

Explainability for optimisation-based decision support systems

Roger Xavier Lera Leri

School of Engineering

Universitat Autònoma de Barcelona

Submitted in partial satisfaction of the requirements for the
Degree of Doctor of Philosophy
in Computer Science

Supervisor Dr. Filippo Bistaffa
Second Supervisor Dr. Juan Antonio Rodríguez Aguilar

March, 2026

Abstract

Decision Support Systems (DSS) based on optimisation techniques are widely deployed to support decision-making in complex environments. Although optimisation approaches are often perceived as more transparent than machine learning models, their outcomes (i.e., the recommendations that will inform the decision-making process) remain difficult for non-expert users to understand. This drawback stems from the inherent complexity of real-world optimisation problems, which often involve many variables and constraints. As a result, users may struggle to understand why a particular decision is optimal, hindering their trust in and adoption of such decisions. For optimisation-based DSS to be effectively integrated into practice, providing clear explanations of their outcomes is fundamental. However, in contrast to the abundant work in explainable AI for machine learning models, considerably less research has been devoted to explainability methods for optimisation approaches.

This thesis addresses this research gap by proposing a general, domain-independent approach that combines constraint reasoning techniques with Large Language Models (LLMs) to generate explanations for optimisation-based DSS. Our proposed approach enables users of optimisation-based DSS to pose questions about their recommendations. To answer such questions, which suggest alternative decisions to the DSS's outcome, our approach generates textual explanations by identifying the reasons why the optimal decision is preferable to the user-suggested alternative. Specifically, we exploit constraint reasoning techniques to structure and semantically characterise the information required by the LLM, which we then use to generate a textual explanation that answers the user's question.

Since this thesis develops a general explainability approach for optimisation-based decision support systems, we tackle three novel and challenging optimisation prob-

lems across distinct real-world domains. Specifically, we consider these problems because (i) they are novel and have not previously been formalised as optimisation tasks, and (ii) they serve as diverse case studies for evaluating the proposed explainability approach. First, we formalise Value System Aggregation as an optimisation problem that aims to derive a consensus value system from a set of individual value systems. Second, we tackle the Research Project Planning Problem, which aims to find the optimal allocation of researchers' dedication to research projects. Third, we formulate the Bachelor's Degree Planning Problem as a scheduling problem, whose goal is to find a degree plan, i.e., an allocation of university subjects across semesters, that maximises the acquisition of skills required for students' preferred jobs. These problems belong to different families of optimisation problems, namely aggregation, resource allocation, and scheduling, thereby demonstrating the generality of the proposed explainability approach.

More specifically, we thoroughly evaluate our explainability approach in terms of (i) completeness, that is, the ratio of identified reasons mentioned in a textual explanation, and (ii) readability, which represents the difficulty of understanding an explanation. Furthermore, we evaluate the cost of generating explanations by applying our approach to two additional hard benchmark problems from the combinatorial optimisation literature: the Resource-Constrained Project Scheduling Problem and the Winner Determination Problem. The empirical evaluation on real-world datasets demonstrates that the proposed methodology generates complete explanations without increasing the difficulty in understanding across diverse optimisation domains. Specifically, providing the structured necessary information as input to an LLM substantially improves the completeness of the resulting textual explanations. Moreover, experimental results indicate that explanations can typically be generated in real time, often in a few seconds.

Resum

Els Sistemes de Suport a la Decisió (SSD) basats en tècniques d'optimització s'implanten àmpliament per donar suport a la presa de decisions en entorns complexos. Tot i que els algorítmes d'optimització sovint es perceben com a més transparents que els models d'aprenentatge automàtic, els seus resultats (és a dir, les recomanacions que informaran els processos de pressa de decisions) continuen sent difícils d'entendre per a usuaris no experts. Aquest repte deriva de la complexitat inherent dels problemes d'optimització del món real, que habitualment impliquen un gran nombre de variables i restriccions. Com a conseqüència, els usuaris poden tenir dificultats per comprendre per què una decisió determinada és òptima, cosa que pot dificultar que confiïn en aquestes decisions i les adoptin. Per tal que els DSS basats en optimització s'integrin de manera efectiva a la pràctica, han de ser capaços de proporcionar explicacions clares dels seus resultats. Tanmateix, en contrast amb l'abundant recerca en IA explicable per a models d'aprenentatge automàtic, s'ha dedicat considerablement menys investigació als mètodes d'explicabilitat per a tècniques d'optimització.

Aquesta tesi aborda aquesta bretxa proposant un mètode general i independent del domini d'aplicació que combina tècniques de raonament de restriccions amb Models de Llenguatge Grans (MLGs) per generar explicacions per a SSD basats en optimització. La nostra proposta permet als usuaris de SSD basats en optimització formular preguntes sobre les seves recomanacions. Per respondre aquestes preguntes, les quals suggereixen decisions alternatives al resultat del SSD, el nostre mètode genera explicacions textuals identificant les raons per les quals la decisió òptima és "*preferible*" respecte l'alternativa suggerida per l'usuari. Concretament, explotem tècniques de raonament amb restriccions per estructurar i caracteritzar semànticament la informació requerida pel MLG, que posteriorment utilitzem per generar una explicació textual que respon a la pregunta de l'usuari.

Atès que aquesta tesi desenvolupa un mètode general d'explicabilitat per a sistemes de suport a la decisió basats en optimització, abordem tres problemes d'optimització nous i desafiants en diferents dominis del món real. Concretament, considerem aquests problemes perquè (i) són nous i no s'havien formalitzat prèviament com a tasques d'optimització, i (ii) serveixen com a casos d'estudi diversos per avaluar el mètode d'explicabilitat proposat. En primer lloc, formalitzem l'Agregació de Sistemes de Valors com un problema d'optimització que té com a objectiu derivar un sistema de valors de consens a partir d'un conjunt de sistemes de valors individuals. En segon lloc, abordem el Problema de Planificació de Projectes de Recerca, que té com a objectiu trobar l'assignació òptima de la dedicació dels investigadors als projectes de recerca. En tercer lloc, formulem el Problema de Planificació del Grau com un problema de planificació, l'objectiu del qual és trobar un pla d'estudis, és a dir, una assignació d'assignatures universitàries al llarg dels semestres, que maximitzi l'adquisició de les competències necessàries per als llocs de treball preferits pels estudiants. Aquests problemes pertanyen a diferents famílies de problemes d'optimització, concretament l'agregació, l'assignació de recursos i la planificació, cosa que demostra la generalitat del mètode d'explicabilitat proposat.

Concretament, avaluem exhaustivament la nostra metodologia d'explicabilitat en termes de (i) completesa, és a dir, la proporció de raons identificades que s'esmenten en una explicació textual, i (ii) llegibilitat, que representa la dificultat d'entendre una explicació. A més, avaluem el cost de generar explicacions aplicant el nostre mètode a dos problemes de referència addicionals i computacionalment durs de la literatura d'optimització combinatòria: el Problema de Planificació de Projectes amb Recursos Limitats i el Problema de Determinació del Guanyador. L'avaluació empírica amb dades del món real demostra que la metodologia proposada genera explicacions completes sense augmentar la dificultat de comprensió en diversos dominis d'optimització. En concret, proporcionar la informació necessària de manera estructurada com a entrada a un LLM millora substancialment la completesa de les explicacions textuals resultants. A més, els resultats experimentals indiquen que les explicacions es poden generar habitualment en temps real, sovint en pocs segons.

Acknowledgements

The past few years have been both intense and deeply rewarding. They have been filled with excitement, the opportunity to work on fascinating problems, and the privilege of meeting remarkable people. At the same time, this journey has brought significant challenges, moments that tested my resilience and patience. During those times, I was fortunate to be surrounded by people who encouraged me to persevere and continue moving forward. I would therefore like to express my sincere gratitude for their guidance, support, and reassurance.

First and foremost, I would like to express my deepest gratitude to my supervisors, Filippo Bistaffa and Juan A. Rodríguez-Aguilar. I am sincerely thankful not only for their technical guidance and insightful feedback throughout my PhD, but also for their constant support and the many good moments we shared along the way. I have been incredibly fortunate to work with such brilliant mentors, who treated me as a colleague, challenged me to grow, and were always available whenever I needed advice or direction.

Filippo, thank you for first guiding me during my Master's thesis, for sparking my interest in academia and research, and for later placing your trust in me by offering me the opportunity to continue my doctoral studies under your supervision. Over these years, I have learned immensely from your way of thinking, your rigour, and your approach to research. On a more personal note, I will always remember our after-work climbing sessions —and how effortlessly you would flash the boulders that had taken me countless attempts. Those moments made this journey even more memorable.

JAR, thank you for all the enlightening conversations we shared and for the generosity you showed in dedicating your time and energy to my work, despite not being

my "*formal*" supervisor. I am deeply grateful for the many hours you invested in carefully reviewing my manuscripts, pushing me to refine and improve them iteration after iteration. Your commitment to excellence has left a lasting mark on my work. And, of course, thank you for feeding me countless croissants during meetings — I may have gained a few kilograms over these years, but I did so very happily.

I would also like to thank my friends and colleagues at the institute, who shared with me not only countless hours in the lab but also the ups and downs that inevitably come with research. We shared knowledge, exchanged ideas, and celebrated one another's successes — all of which really helped me during the thesis. Beyond work, I am especially grateful for the (maybe many) hours spent playing table tennis in the robotics room. Although my PhD chapter now comes to an end, I sincerely hope that this is not the end of the experiences we share.

Finally, I would like to express my deepest gratitude to my family: my mother, Blanca, my sisters, Anna and Nuria, and my girlfriend, Laia. Thank you for your unconditional love, patience, and constant support throughout these years.

List of Tables

3.1 Policy options in the Súdwest-Fryslân PVE 54
3.2 *VO* matrices representing the relationship between alternatives and values 56
3.3 Value preferences (columns 2 and 3) and value judgement functions (columns 4–7) for the countries in the EVS. 60
3.4 Computed consensus ranking for different ethical principles p. 70
3.5 Value preferences (columns 2, 3, and 8) and value judgement functions (columns 4–7) of the consensus European value system when considering equal social influence. 74
3.6 Value preferences (columns 2, 3, and 8) and value judgement functions (columns 4–7) of the consensus European value system when social influence considers population. 74

4.1 Dataset information (anonymous project names). 99
4.2 Distributions used for RPP instance generation. 108

5.1 Elective subjects computed for the degree plans targeting *Penetration Tester* and *Software Engineer* jobs. Units within both degree plans are in **bold**. 137
5.2 Skills required by the considered jobs and their Required Level (RqL), along with the Achieved Level (AcL) after completing the computed degree path and the unit that provides such skill level. Units in black are core units. Units in blue are electives that belong to the major program selected to complete by the approach. Units in brown are electives that do not belong to the major program selected by the approach. 138

6.1 Information of the key subjects of the optimal degree plan of Example 6.2.1. We provide the semesters where the subject is planned (and the season), the seasonal offer of the subject, its predecessors, and whether they are core subjects. In case a subject is not planned, we indicated with a − sign in the semester column. 156
6.2 Set of possible user questions, query constraints and constraint encodings for the DPP formalisation. 157
6.3 *IIS* from Example 6.2.3. We indicate the constraint types (in different colours) and the elements of each constraint. 161
6.4 *IIS* from Example 6.2.3. We indicate the constraint type (in different colours) and the mathematical encoding of the constraints. 162
6.5 Natural Language Templates for each type of DPP constraint discussed in Section 5.2. 162
6.6 Explanations generated using a prompt with the graph of reasons (left) and a prompt with the IIS (right) to answer the query of the running example: "*Why is Programming Fundamentals not planned at semester 4?*". We highlight in *blue* the reasons. 172
6.7 Reasons mentioned in the explanations (shown in Table 6.6) generated using a prompt with the graph of reasons and a prompt with the IIS. 172
6.8 Average completeness, number of words, and GFI per explanation for both prompt types. 174

7.1 General features for user queries. 195
7.2 IIS runtime prediction results using problem and query features as input for the different trained models: *Linear Regression* (LR), *Quadratic Regression* (QR), *Regression Tree* (RT), *Random Forest* (RF), *Shallow Learning* (SL), and *Deep Learning* (DL). We report 10-fold cross-validation RMSE and MAE (lower values are better). Bold indicates the (significantly) best-performing model. 198
7.3 IIS runtime prediction comparison between models using only problem instance features and models using query and problem instance features. We report 10-fold cross-validation RMSE (lower values are better) for both optimisation problems, the WDP and the RCPSP. In the third row, we report the gain (in percentage) of using query features. Bold indicates the (significantly) best-performing model. . . 200

B.1 *IIS* from Example 6.2.3. We indicate the constraint type (in different colours) and the mathematical encoding of the constraints. 220

B.2 Set of possible user questions, query constraints and constraint encodings for the Aggregation of Value Systems formalisation in the European Value Study case. 221
B.3 Set of possible user questions, query constraints and constraint encodings for the Research Project Planning Problem formalisation. . . 222
B.4 Set of possible user questions, query constraints and constraint encodings for the Degree Planning Problem formalisation. 223
B.5 Set of possible user questions, query constraints and constraint encodings for the Resource-Constrained Project Scheduling Problem formalisation. 224
B.6 Set of possible queries, the query constraints and the constraint encoding of the Winner Determination Problem formalisation. 225
B.7 Prompt using the Graph of Reasons to answer the query of the running example. 226
B.8 Prompt using the IIS (with their constraints translated) to answer the query of the running example. 227

C.1 Specific features for the WDP instances defined by Leyton-Brown, Nudelman and Shoham [114]. 229
C.2 Specific features for the RCPSP instances. 230
C.3 Solving time prediction results for the different models and different problem instances. We report 10-fold cross-validation RMSE and MAE (lower values are better). Bold indicates the (significantly) best-performing model. 231

List of Figures

1.1 Main components of an explainable DSS showing its interaction with users. 5
1.2 Classes of MIP problems. While MILPs consider linear objective and constraints, MIQP considers a quadratic objective function, and MIQCP comprise problems with quadratic constraints. 7
1.3 Scheme of the main research questions and contributions of this thesis. 11

2.1 Classes of MIP problems. In particular, MILP, MIQP, and MIQCP can be further divided into two subclasses: those with only discrete variables, such as ILP, IQP, and IQCP, and those with only continuous variables, such as LP, QP, and QCP. MILP, MIQP, and MIQCP are specific cases of MICP. 31
2.2 Subclasses of MIPs and p-norm minimisation problems depending on the value of p. 35

3.1 The impact of the ethical principle p on the computed consensus of a set of individuals' judgement of two objects: $R_i[1]$ and $R_i[2]$. Circles show the individuals' judgement and squares represent the consensus computed. Squares are filled with the colour of the ethical principle used to compute the consensus. (Best viewed in colour). 51

3.2 Distance between the consensus computed according to ethical principle p and the consensuses computed according to $p = 1$ (fully utilitarian, black line) and $p = \infty$ (fully egalitarian, red line). The transition point $\bar{p}$ is the ethical principle producing a consensus equidistant from the fully utilitarian and fully egalitarian ones. Thus, $\bar{p}$ divides the space of ethical principles into an *utilitarian zone* (more similar to the fully utilitarian consensus, green) and an *egalitarian zone* (more similar to the fully egalitarian consensus, light blue). The *fully egalitarian* (dark blue) zone marks the ethical principles that produce a consensus approximately equal (to a small ϵ) to the fully egalitarian one. PVE case study on top, and EVS case study for the population-based influence case at the bottom. 76

3.3 Boxplots of residuals for different ethical principles in the value preference and value-alternative aggregation for the PVE (top) and EVS (bottom) case study. The diamond represents the mean and the circles represent the outliers. 80

4.1 Results in the *sufficient budget* scenario for $\gamma = \infty$ (best viewed in colours). 102

4.2 Results in the *abundant budget* scenario for $\gamma = \infty$. 104

4.3 Target deviation (4.3a), budget excess (4.3b), and comparison between the contracted researchers' dedication with their target (4.3c), for different γ and for $\alpha = 0.25$, $\beta = 0$. 105

4.4 Fig. 4.4a reports the number of researchers generated with respect to the number of projects $|P|$ and the maximum number of work packages per project W_{max}. Fig. 4.4b reports the total solution runtime. Each cell reports the average and the standard error of the mean over the 10 instances generated for each combination of $|P|$ and W_{max}. . . 107

4.5 Results in the *sufficient budget* scenario for $\gamma = \infty$ considering uncertainty. 111

4.6 Fig. 4.6a reports the number of researchers generated with respect to the number of projects $|P|$ and the maximum number of work packages per project W_{max}. Fig. 4.6b reports the total runtime required to compile and solve RPP. Each cell reports the average and the standard error of the mean over the 10 instances generated for each combination of $|P|$ and W_{max} (best viewed in colours). 113

5.1 Overview of our automated solution approach. We refer the reader to Sections 5.2 and 5.3 for the technical details. 117
5.2 Histogram of job affinities for all jobs. We report the distribution of the job affinity values in two frequency metrics: the absolute number of jobs (left axis), and the percentage of jobs (right axis). 140
5.3 Evolution of the job affinity during the degree plan. The shadowed region represents the standard deviation of the job affinity for the different target jobs. 141
5.4 Cactus plot showing the job affinity for the degree plan of each target job. We order degree plans from the lowest job affinity to the highest on the x-axis. We plot the job affinity considering degree plans without the elective units (in blue); whole degree plans (in brown); and all units from the degree syllabus, $\mathcal{U}$ (in red). 142
5.5 Average RJA along the size of the set of target jobs. 145
5.6 Evolution of job affinity increase along semesters. 146
5.7 Screenshot of JobIQ showing the skills required by the *ICT Business and Systems Analysts* job and which skills are acquired after completing the Bachelor's Degree in ICT. 147

6.1 Overview of our explanatory approach. Section 6.2 discusses its steps in detail. 153
6.2 Process to build a graph of reasons applied to our running example: (1) relating constraints within the IIS to yield its dual graph, and (2) labelling constraints in the dual graph to build the graph of reasons. . 163
6.3 Average percentage of reasons mentioned in the explanations generated using the IIS (red) and using the graph of reasons (green). We group the explanations into three groups depending on the number of reasons: (i) small (less than five reasons), (ii) medium (between five and ten reasons), and (iii) large (more than ten reasons). 176

7.1 Distribution of IIS size reduction computed by FORQES versus CPLEX (green, left) and runtime increase to compute the smallest IIS (red, right). 185
7.2 Runtimes to solve the MIP problem and compute IIS (left y-axis), and time overhead (right y-axis), for DPP. 186
7.3 Average IIS computation time (lines, left y-axis) and overhead (dots, right y-axis) for different query types for the DPP. 186

7.4 Runtimes to solve the MIP problem and compute IIS (left y-axis), and time overhead (right y-axis), for the RPP problem. 187
7.5 Average IIS computation time (lines, left y-axis) and overhead (dots, right y-axis) for different query types for the RPP problem. 187
7.6 Runtimes to solve the optimisation problem and compute IIS (left y-axis), and time overhead (right y-axis), for the aggregation of value systems. 188
7.7 Boxplot of the Solving time of the MIP problem, the IIS runtime (left y axis), and the Overhead for the RCPSP (right y axis). 189
7.8 Average IIS computation time (lines, left y-axis) and overhead (dots, right y-axis) for different query types for the RCPSP. 189
7.9 Average IIS runtimes (lines) and overhead (dots) for different WDP instance distributions: *scheduling*, *matching*, *paths*, and *regions*. . . . 190
7.10 IIS Runtime predictions on previously unseen test instances for Random Forest (best performing model). The $x-$axis denotes the runtime prediction, and the $y-$axis denotes the real runtime. Each data point represents one instance. We show the Coefficient of Determination (R^2), which measures how well the model predicts runtime (1 indicates a perfect fit). 199

C.1 Solving time predictions on previously unseen test instances for Random Forest (best performing model). The $x-$axis denotes the runtime prediction, and the $y-$axis denotes the real runtime. Each data point represents one instance. We show the Coefficient of Determination (R^2), which measures how well the model predicts runtime (1 indicates a perfect fit). 232

Table of Contents

1 Introduction **1**

1.1 Motivation . . . 2

1.1.1 The Need for Explanations . . . 2

1.1.2 Mathematical Optimisation . . . 5

1.1.3 Optimisation Problem Domains . . . 7

1.1.3.1 Aggregating Value Systems . . . 8

1.1.3.2 Research Project Planning . . . 8

1.1.3.3 Bachelor's Degree Planning . . . 9

1.2 Research Questions . . . 10

1.2.1 Applied Research Questions . . . 11

1.2.1.1 Aggregating Value Systems . . . 11

1.2.1.2 Research Project Planning . . . 12

1.2.1.3 Bachelor's Degree Planning . . . 13

1.2.2 Methodological Research Questions . . . 13

1.3 Contributions . . . 15

1.3.1 Applied Contributions . . . 15

1.3.1.1 Solving the Value Systems Aggregation Problem . . . 15

1.3.1.2 Solving the Research Project Planning Problem . . . 17

1.3.1.3 Solving the Bachelor's Degree Planning Problem . . . 19

1.3.2 Methodological Contributions . . . 19

1.4 Thesis Outline . . . 23

1.5 Publications . . . 24

2 Background and Related Work **29**

2.1 Mixed-Integer Programming . . . 29

2.1.1 Convex Optimisation . . . 31

2.1.2 Objective Functions Involving Norms . . . 33

2.2 Infeasibility in Optimisation Problems . . . 35

2.3 Related Work on Optimisation . . . 37

2.3.1 Differences Between the RCPSP and the problems addressed in this thesis . . . 39
2.4 Related Work on Explainability . . . 40

3 Aggregating Value Systems **44**
3.1 Introduction . . . 44
3.2 Preliminaries: Distance Functions . . . 47
3.3 Value Systems: From Theory to Practice . . . 52
3.3.1 Participatory Value Evaluation . . . 54
3.3.1.1 Definition of Value Systems . . . 55
3.3.1.2 Value System Estimation . . . 56
3.3.2 European Value Study . . . 57
3.3.2.1 Definition of Value Systems . . . 58
3.3.2.2 Value System Estimation . . . 59
3.4 Formalising the Aggregation of Value Systems . . . 61
3.5 An ℓ_p-regression Approach to Aggregate Value Systems . . . 63
3.5.1 Solving the ℓ_p-regression Problem . . . 64
3.6 Experimental Results . . . 68
3.6.1 Case study: Participatory Value Evaluation . . . 69
3.6.1.1 Analysing the Consensus on Value Preferences . . . 69
3.6.1.2 Analysing the Consensus on Values-Alternatives Relationships . . . 71
3.6.2 European Value Study . . . 73
3.6.3 Characterising the Space of Ethical Principles . . . 75
3.6.3.1 Takeaways for Decision-Makers . . . 78
3.6.4 The Relationship between Consensus and Individual Value Systems . . . 79
3.6.4.1 Takeaways for Decision-Makers . . . 81
3.7 Conclusions and Future Work . . . 82

4 Research Project Planning **86**
4.1 Introduction . . . 86
4.2 Stochastic Robust Approximation . . . 88
4.3 Problem Definition . . . 89
4.3.1 Basic Definitions . . . 90
4.3.2 Optimisation Objective . . . 90
4.3.3 Hard Constraints . . . 91
4.4 Solving the RPP Problem . . . 92

4.4.1 Introducing Soft Budget Constraints . . . 94
4.4.2 Introducing Uncertainty with Stochastic Instances . . . 96
4.5 Experimental Analysis . . . 99
4.5.1 Discussion of the Results . . . 100
4.5.1.1 Sufficient Budget Scenario . . . 103
4.5.1.2 Abundant Budget Scenario . . . 103
4.5.1.3 Scarce Budget Scenario . . . 105
4.5.2 Scalability Evaluation . . . 107
4.5.3 Experiments with Uncertainty . . . 109
4.5.3.1 Experimental methodology . . . 109
4.5.3.2 Results . . . 110
4.6 Conclusions . . . 112

5 Bachelor's Degree Planning **115**
5.1 Introduction . . . 115
5.2 Formalising the Degree Planning Problem as an MIP . . . 118
5.2.1 Basic Definitions . . . 118
5.2.2 Objective of the Degree Planning Problem . . . 120
5.2.3 Constraints . . . 121
5.2.4 Completed Units . . . 126
5.3 Solving DPP by Linearising the Argument of the p-norm . . . 127
5.3.1 Equivalence Between MIP and NMP . . . 130
5.4 Experimental Analysis . . . 135
5.4.1 Dataset . . . 135
5.4.2 Implementation . . . 136
5.4.3 Use Case: Beginning-of-Degree Planning . . . 137
5.4.4 Use Case: Most Attainable Jobs on the Market . . . 139
5.4.5 Use Case: Multi-Job Target . . . 142
5.4.6 Use Case: Mid-Degree Planning . . . 146
5.5 Real-world Applicability of our Approach . . . 147
5.5.1 Current Deployment . . . 147
5.5.2 Extending Our DPP Model to Other Curricula . . . 147
5.6 Conclusions . . . 149

6 Explainability for Optimisation **151**
6.1 Introduction . . . 151
6.2 Building Explanations for Optimisation-Based Decisions . . . 155
6.2.1 Query Translation . . . 156

6.2.2 The User-Desired Satisfiability Problem . . . 159
6.2.3 Computing the Infeasible Constraints . . . 160
6.2.4 Constructing the *Graph of Reasons* . . . 162
6.2.5 Providing Textual Explanations with LLMs . . . 165
6.3 Experimental Evaluation . . . 168
6.3.1 Experimental Setting . . . 168
6.3.1.1 Optimisation-Based Benchmark Scenarios . . . 168
6.3.1.2 Experimental Methodology . . . 170
6.3.2 Completeness and Readability Results . . . 173
6.4 Conclusions . . . 176

7 On the Cost of Explanations 180
7.1 Introduction . . . 180
7.2 Empirical Hardness Evaluation . . . 182
7.2.1 Experimental Setting . . . 183
7.2.1.1 Problem Instances . . . 183
7.2.1.2 Implementation and Computational Setup . . . 184
7.2.2 Is it worth computing the smallest IIS? . . . 184
7.2.3 Empirical hardness results . . . 185
7.3 Predicting explanations' empirical hardness . . . 192
7.3.1 Machine learning techniques . . . 193
7.3.1.1 Linear Regression . . . 193
7.3.1.2 Quadratic Regression . . . 193
7.3.1.3 Regression Tree and Random Forest . . . 194
7.3.1.4 Neural Networks . . . 194
7.3.2 Feature Engineering . . . 195
7.3.2.1 Query Features . . . 195
7.3.2.2 Problem Instance Features . . . 197
7.3.3 Empirical analysis . . . 197
7.3.3.1 Comparing predictors . . . 198
7.3.3.2 Quantifying the benefit of using query features . . . 199
7.4 Conclusions . . . 200

8 Conclusions and Future Work 204
8.1 Key Contributions . . . 204
8.1.1 Aggregating Value Systems . . . 205
8.1.2 Research Project Planning . . . 206
8.1.3 Bachelor's Degree Planning . . . 207

8.1.4 Explaining the Outcome of an Optimisation-Based DSS . . . 208
8.2 Future Work . . . 210
8.2.1 Aggregating Value Systems . . . 211
8.2.2 Research Project Planning . . . 212
8.2.3 Bachelor's Degree Planning . . . 213
8.2.4 Explaining the Outcome of an Optimisation-Based DSS . . . 214

A Winner Determination Problem **217**
A.1 Formalisation of the Winner Determination Problem . . . 217

B Explainability Supplementary Material **219**
B.1 IIS example . . . 219
B.2 Query Examples . . . 219
B.2.1 Aggregation of Value Systems . . . 219
B.2.2 Research Project Planning Problem . . . 220
B.2.3 Degree Planning Problem . . . 222
B.2.4 Resource-Constrained Project Scheduling Problem . . . 223
B.2.5 Winner Determination Problem . . . 224
B.3 LLM Prompt . . . 226

C Runtime Prediction Supplementary Material **228**
C.1 Problem Instance Features . . . 228
C.1.1 WDP Features . . . 228
C.1.2 RCPSP Features . . . 230
C.2 Solving Time Prediction . . . 231

Chapter 1

Introduction

In an increasingly complex and data-driven world, organisations, institutions, and individuals are often faced with decisions that require careful analysis of multiple factors, including data, individual or collective preferences, constraints, and objectives. To aid decision-making, *Decision Support Systems* (DSSs) have emerged as essential tools. In general, a DSS is an information system designed to support (rather than replace) human decision-making by providing recommendations and facilitating their evaluation [102].

In many real-world domains, decision-making is driven by a DSS that employs some optimisation algorithm. In this framework, an optimisation algorithm computes recommendations on decisions by solving some optimisation problem. Examples from application domains include workforce management [22], food industry production [193], and construction projects [128], among others.

In this thesis, we focus on formalising and solving optimisation problems to support decision-making in real-world applications. However, optimisation-based DSSs produce outputs that remain difficult for users to understand. This difficulty arises because real-world optimisation problems typically involve a large number of variables and constraints, making the resulting solutions complex and difficult to interpret. As a result, the lack of explainability in these systems hinders users' trust in and adoption of the recommendations. To address this gap, we aim to develop DSSs that are not only capable of generating recommendations based on optimal solutions but also of providing human-understandable explanations that justify why a DSS has recommended a given decision.

Our focus lies in explaining the outcomes of optimisation-based DSSs. To achieve this, we employ *Artificial Intelligence* (AI) techniques to generate explanations that are accessible to DSS users. In contrast to much of the research in explainability, which focuses on explaining the behaviour of *Machine Learning* (ML) models, our work instead aims at the outcomes of optimisation problems. This distinction clearly differentiates our work from the contributions of explainable AI research.

The remainder of this chapter is structured as follows. Section 1.1 presents the motivation behind this research. Section 1.2 introduces the research questions addressed in this thesis, followed by Section 1.3, which outlines the main contributions proposed to answer these questions. Section 1.4 provides an overview of the thesis structure, and finally, Section 1.5 lists the publications derived from this work.

1.1 Motivation

In this section, we present the motivation behind the research questions addressed in this thesis. In detail, Section 1.1.1 discusses the need for providing explanations in DSSs. Next, Section 1.1.2 introduces basic concepts in optimisation addressed in this thesis, i.e., MIP problems. Lastly, Section 1.1.3 outlines the different application domains considered in this thesis.

1.1.1 The Need for Explanations

In recent years, AI has achieved notable momentum, enabling impressive results in many application domains [12]. The integration of AI into DSSs that assist human decision-making is expanding both in our daily lives and across a wide range of industries, including education, healthcare, finance, logistics, and law enforcement [151].

Alongside such growth, there has been an increasing interest in developing human-centred AI that is trustworthy and aligned with the ethical values that characterise human society. In this context, the European Union has established guidelines to define its vision for ethical and trustworthy AI systems [47]. To achieve trustworthy

AI systems, one key requirement is to ensure transparency. Transparency allows users to understand how decisions are made, fostering confidence in the system's decisions. In this regard, a critical aspect of transparency is the ability of AI systems to provide explanations for their outcomes. Thus, there is a growing consensus within the AI community that explainability is essential to justify the rationale behind their decisions to human users. Furthermore, this requirement is not only a matter of good practice but is also increasingly reflected in legal frameworks. For instance, the European General Data Protection Regulation explicitly guarantees individuals the *right to an explanation* when decisions made by automated systems have legal or significant consequences for them [46]. Similarly, the Digital Services Act reinforces the need for transparency in technology, highlighting explainability as a fundamental mechanism to achieve it [49].

Consequently, there has been a recent resurgence of interest in the field of *eXplainable Artificial Intelligence* (XAI) three decades after its inception [37, 184], as researchers and practitioners seek to provide more transparency for their algorithms [126]. Namely, XAI refers to a class of AI systems whose decisions or behaviours are explained and understood by humans [151]. More specifically, XAI primarily focuses on explaining the outcomes of ML algorithms to provide insights into the internal functionalities of models that typically operate as "black boxes".

However, there is still a significant lack of research on explaining the outcomes of optimisation problems. While it is possible to explain the formalisation of an optimisation problem, including the definition of decision variables, constraints, and the objective function, the resulting solution may still appear unintuitive or difficult to interpret for end users. For example, consider a DSS used to schedule university subjects across semesters. Such a problem typically involves thousands of decision variables (e.g., assigning courses to specific semesters) and numerous constraints (e.g., precedence requirements among subjects, where some must be completed before starting others, credit limits per semester, and course availability). The objective function may aim to maximise the acquisition of the skills required for a desired job profile by appropriately planning courses. The DSS may then propose a specific study plan indicating which subjects a student should take in each semester. How-

ever, for an end user, it is often difficult to understand why this particular plan is recommended over many alternative feasible ones, as the system does not explain the reasons underlying its recommendation. This, in turn, can hinder the practical adoption of optimisation-based DSSs [207].

Indeed, explanations that justify "*why*" a system recommends a decision are more relevant to non-expert users than those that describe "*how*" a decision was obtained [206, 207], which are typically meaningful to experts. Importantly, in the context of social sciences, research has shown that explanations are most useful and satisfying when they do not merely state why a certain outcome was recommended, but also why it was chosen over an alternative. Such explanations are referred to as *contrastive explanations* [126].

In line with this, some approaches in the literature aim to explain the outcomes of optimisation problems by generating contrastive explanations [76, 211]. Typically, they do so by numerically comparing the original recommendation with an alternative suggested by the user. While effective at highlighting differences, these methods require solving an additional optimisation problem for each explanation, which can be computationally expensive. Moreover, these techniques are often tailored to specific application domains [76, 150]. Furthermore, they rely on abstract representations [211] or fixed templates [76, 150] —which may be inadequate when presenting explanations as part of a conversation with the user [126]—, overlooking the recent advances in *Large Language Model*s (LLMs) to produce natural language explanations.

Against this background, our main objective is to propose a general methodology for optimisation-based DSS that generates contrastive explanations in response to users' questions. A user question, henceforth also referred to as "*query*", suggests alternative recommendations to the optimal recommendation. For instance, in the case of the academic planning DSS, a user might ask "*Why has Object-Oriented Programming been scheduled instead of Parallel Programming?*". Notice that this query suggests an alternative recommendation (planning Parallel Programming) over the optimal solution (planning Object-Oriented Programming). Therefore, the challenge in this

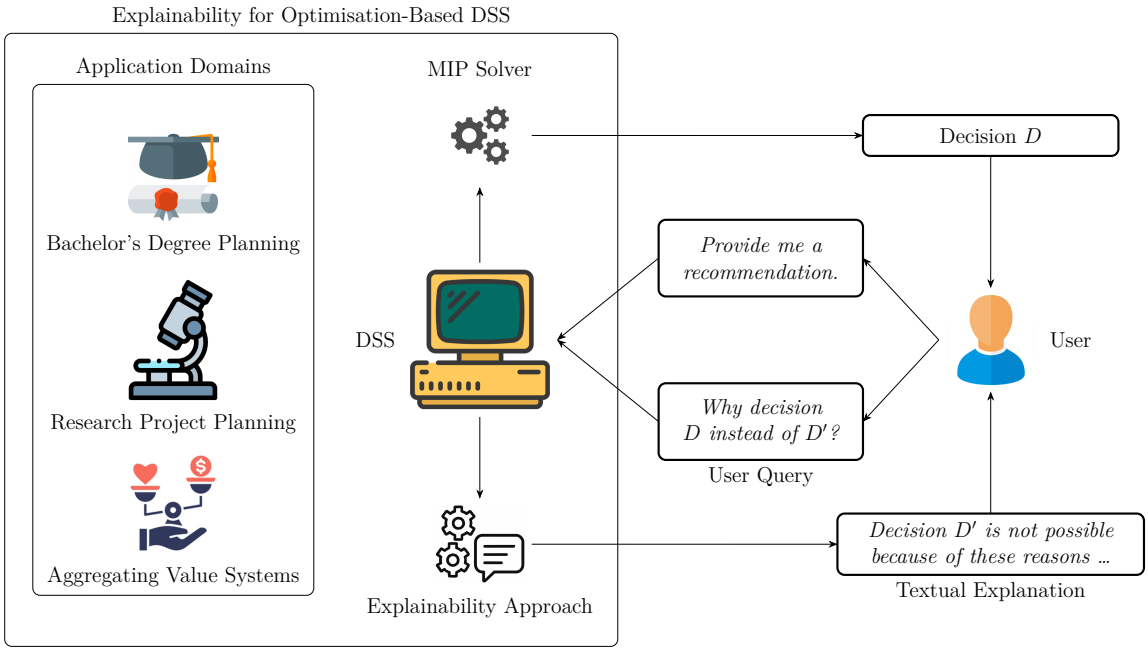


Figure 1.1: Main components of an explainable DSS showing its interaction with users.

thesis is to explain why a recommended decision is better than a user-suggested alternative.

Figure 1.1 outlines our notion of explainable DSS. First, a user requests a recommendation to the DSS. Once a recommendation is provided, we allow the user to pose queries that challenge the recommendation. Then, our explainability approach generates an explanation in response to the user query.

1.1.2 Mathematical Optimisation

As previously mentioned, this thesis proposes a general methodology to explain the recommendations of optimisation-based DSS. To illustrate the generality of this approach, we address multiple optimisation problems. In this section, we introduce the basic concepts to formalise the optimisation problems addressed.

Mathematical optimisation focuses on finding the best possible solution to a problem subject to constraints. A solution corresponds to a numerical assignment to a set of decision variables, where these variables encode concrete real-world decisions. An optimisation problem therefore seeks an assignment that (i) minimises or maximises

a given objective function and (ii) satisfies all imposed constraints. For example, solving a scheduling problem usually amounts to finding the schedule of tasks that respects the particular schedule restrictions and minimises the total activity duration.

A well-established technique to formalise optimisation problems is *Mixed-Integer Programming* (MIP). MIP problems are those that have discrete and continuous decision variables [198]. In simple terms, the outcome of MIP problems includes some variables that take fractions or real values and others that are restricted to integers. MIP represents a general class of optimisation problems. Indeed, *Integer Programming* (IP) is a special case of MIP in which all variables are restricted to integer values [199]. Conversely, continuous optimisation is another special case of MIP in which all variables are continuous. In general, MIP problems (with integer variables) are NP-Hard [198] and are expensive to solve.

In more detail, a MIP is an optimisation problem in the form of

$$\text{minimise} \quad f_0(\mathbf{x}, \mathbf{y}) \tag{1.1a}$$

$$\text{subject to} \quad f_i(\mathbf{x}, \mathbf{y}) \leq 0, \quad \forall i = 1, \ldots, m, \tag{1.1b}$$

where $\mathbf{x} \in \mathbb{R}^n$ is a vector of continuous decision variables, $\mathbf{y} \in \mathbb{Z}^k$ is a vector of integer decision variables, and $f_i : \mathbb{R}^n \times \mathbb{Z}^k \to \mathbb{R}$ is any real-valued function. Notice that we do not assume any form for functions f_i. More specifically, the family of MIPs where the objective function f_0 is linear or quadratic is characterised as *Mixed-Integer Linear Programming* (MILP) and *Mixed-Integer Quadratic Programming* (MIQP) respectively. In Section 2.1, we further discuss the different classes of MIP problems shown in Figure 1.2.

Given that MIP is a general technique capable of capturing a wide variety of optimisation problems, we use it to formulate the underlying optimisation problems of the DSSs considered in this thesis. Furthermore, since one of the main goals of this thesis is to explain a solution to a given optimisation problem, we assume that such a solution can be computed in the first place by means of an optimisation solver within a reasonable amount of time. Along these lines, with a small abuse of terminology, in the remainder of this thesis, we will refer to as "*optimisation problems*" the famil-

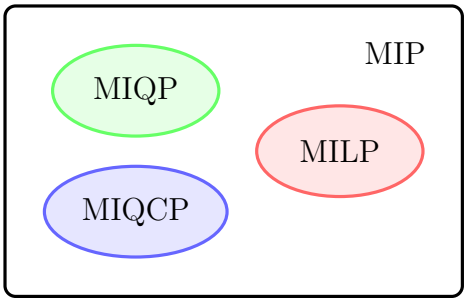


Figure 1.2: Classes of MIP problems. While MILPs consider linear objective and constraints, MIQP considers a quadratic objective function, and MIQCP comprise problems with quadratic constraints.

ies of optimisation problems that a given solver can solve. More specifically, in this thesis, we employ CPLEX [55], which can solve the families of optimisation problems shown in Figure 1.2 (such families will be discussed in Section 2.1). We adopt this approach since the explanation technique proposed in this thesis is solver-agnostic, i.e., it can be applied to any optimisation problem, provided that a solver capable of computing a solution exists.[1]

In the remainder of this subsection, we motivate the need for DSS across the different domains addressed in this thesis.

1.1.3 Optimisation Problem Domains

In this section, we introduce the three specific optimisation problems tackled in this thesis. These problems have been selected for two main reasons. First, each one addresses a real-world scenario that has not previously been studied from an optimisation perspective. Second, the optimisation problems fall into three categories: aggregation, resource allocation, and scheduling. All of these can be formalised and solved as MIP problems. This diversity enables us to demonstrate the generality of our explanation methodology across varied problem structures and application domains.

[1]For instance, our technique can be directly extended to *Mixed-Integer Convex Programming* (MICP) (i.e., a family of problems that comprises and extends the three subclasses shown in Figure 1.2) by switching to the MOSEK solver [9], which is capable of solving such a family of optimisation problems.

1.1.3.1 Aggregating Value Systems

As discussed in Section 1.1.1, there is growing interest in the development of human-centred AI that is both trustworthy and ethical. In this context, value alignment has emerged as a central challenge in AI & Ethics. Value alignment aims to ensure that the goals, behaviours, and decisions of powerful AI systems are aligned with human values [163].

A core component in value alignment is the concept of a *value system*, which models how an entity (e.g., an individual or an organisation) interprets and prioritises values. Most state-of-the-art research in value alignment assumes that an AI system must align with a *single* value system, be it that of an individual or a society. However, as Gabriel [72] argues, humans often hold multiple, reasonable yet conflicting beliefs about values. In other words, we live in a pluralistic world where individuals maintain diverse value systems. For example, consider the moral value of "*respect*" in the context of a funeral. In Western cultures, wearing black clothing is considered a sign of respect, whereas in some Asian cultures, wearing white is seen as the respectful choice.

Furthermore, in a pluralistic society, policymakers make decisions or create policies aligned with the society's moral values. This motivates the need for a system that informs policymakers about the consensus value system that represents a society. Once such a consensus is established, policymakers can make decisions aligned with it, ensuring they reflect society's values as a whole.

1.1.3.2 Research Project Planning

Effectively managing *academic research projects* presents several challenging aspects. A key difficulty lies in planning how researchers should allocate their working hours across the multiple projects they contribute to. Each project is typically divided into *work packages* (i.e., groups of related tasks within the project that form an isolated unit of work), each requiring a specific number of hours to meet project goals and timelines. However, allocating such dedication is subject to multiple constraints.

These constraints are both legal and personal. For instance, legal regulations may

limit the maximum number of hours a researcher can be assigned within a given period, such as per day, week, or month. At the same time, researchers may have individual preferences for how they allocate their time across the projects they are involved in. Budget availability further restricts the allocation process, as funding limits the number of person-hours available and must be allocated accordingly to researchers' professional levels and expertise.

Moreover, research project conditions change over time (e.g., researchers can join or leave the team, new projects can be approved, and deadlines might get extended). Due to the evolution of research projects, managers must account for uncertainty when planning them.

Despite these challenges, the majority of research institutions still handle project planning manually, as automated solutions are lacking. This approach not only increases the workload of research project managers (who manually assign researchers' hours to projects) but also results in suboptimal allocation of resources. These limitations highlight the need for a DSS that recommends optimal allocations of researchers' hours across multiple projects, accounting for institutional constraints, individual preferences, and changing conditions.

1.1.3.3 Bachelor's Degree Planning

In recent years, there has been a growing interest in developing technologies and tools for personalised education [P8]. Within this landscape, *Educational Recommender Systems* (EdRecSys) have emerged as a central area of research [61], aiming to assist students in making informed decisions throughout their academic journey. One of the main topics of interest for EdRecSys research has been recommending university courses that not only satisfy degree requirements but also align with their long-term career goals. This decision-making process can significantly influence their professional trajectories. Indeed, in a recent survey carried out by the educational platform Forage,[2] 1000 U.S. university students were interviewed about several aspects of education in relation to their professional prospects. One of the most important

[2] https://www.theforage.com/blog/news/forage-career-readiness-survey, accessed on 23/05/2024.

findings of this survey was that 71% of the students evaluated as *extremely important* to "identify the most important skills [needed] to land [their] dream job", but only 39% considered the "tools or information to do this" as adequate. Along these lines, developing an automated approach to guide students throughout their educational pathway toward the acquisition of the skills required for their preferred job(s) is key.

Furthermore, university students must comply with the requirements of their Bachelor's degree curricula when planning which subjects to enrol in each semester. For instance, due to limited resources, universities typically offer specific subjects only during certain semesters. In addition, many subjects have prerequisites, such as the prior completion of foundational or related courses. Some degree programmes also require students to complete a major or specialisation. These institutional constraints, combined with the need to select subjects that best equip students for their preferred career paths, make the manual construction of an optimal degree plan highly complex.

1.2 Research Questions

This thesis aims to develop explainability tools for optimisation-based DSSs in different application domains. Our main **methodological** research question is:

- Can we develop a general methodology to explain the optimal decisions recommended by a DSS obtained by solving a MIP problem?

Then, to illustrate how our methodology can be used in real-world application domains, we also consider the following **applied** research question:

- Can the various real-world optimisation problems from our application domains of interest be effectively formalised as MIP problems?

Figure 1.3 summarises the main research questions and contribution of this thesis. In the remainder of this section, we further elaborate on each of these research questions in more detail. Then, in Section 1.3, we discuss how our contributions provide answers to such research questions.

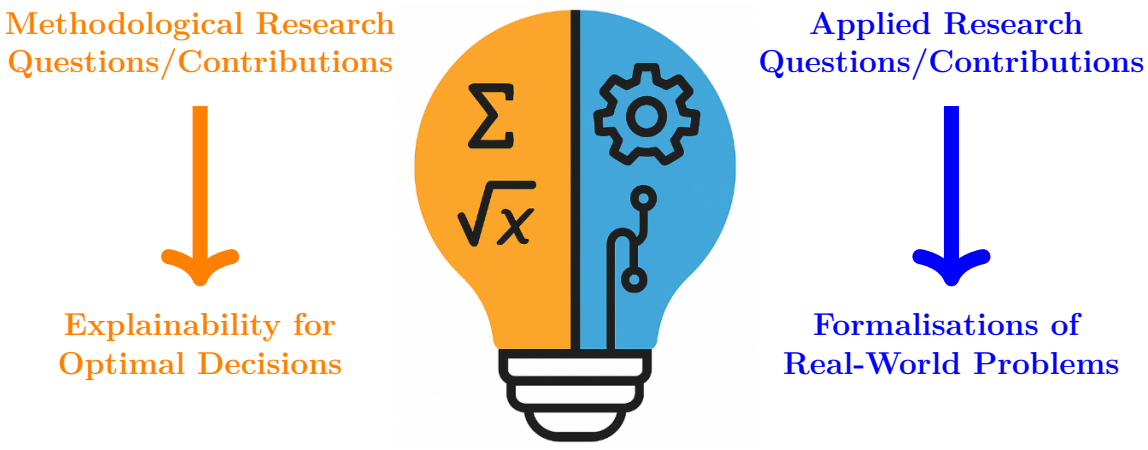


Figure 1.3: Scheme of the main research questions and contributions of this thesis.

1.2.1 Applied Research Questions

1.2.1.1 Aggregating Value Systems

As mentioned above, *value systems* model how an entity interprets and prioritises moral values. Although some approaches in AI consider moral values [132, 131], they do not explicitly define a notion of value system. Other approaches [18, 172, 123] do define a value system, but they focus only on value preferences, without addressing the fact that moral values are abstract and can be interpreted differently in a pluralistic world. To aggregate value systems, we first need to identify the key elements that characterise how values are interpreted and prioritised. Consequently, we pose the following research question:

Question Q1: How do we formalise a Value System?

After formalising our notion of a value system, we next address the challenge of aggregating individual value systems into a consensus value system. Based on the social choice literature, various aggregation criteria, referred to as *ethical principles*, have been proposed for aggregating *judgements* or *preferences* [82]. For example, the *utilitarian* ethical principle [19] yields a consensus that maximises the total welfare, which favours outcomes that benefit the majority. In contrast, the *egalitarian* ethical principle [153], aims to maximise *fairness*, whose outcomes reduce the minority dissatisfaction. Along these lines, we adopt a parameterised distance function from

prior work [78, 79, 81] to aggregate individual judgements or preferences. The key advantage of this approach is that the resulting consensus varies with the ethical principle used for aggregation: at one extreme, it yields a utilitarian consensus that maximises total welfare, while at the other, it produces an egalitarian consensus that emphasises fairness and reduces minority dissatisfaction. By employing this distance metric as the objective function, we now formulate the aggregation of value systems as a MIP problem. This leads to the following research question:

Question Q2: How can we formally define and solve the problem of aggregating individual value systems as an optimisation problem?

Given that the resulting consensus is sensitive to the parameter choice of the distance function, i.e., the so-called ethical principle, providing guidelines that support the selection of an ethical principle is essential for aggregating value systems. Indeed, as noted by Baum and Slavkovik [15], the choice of how values are aggregated is not "*morally neutral*." Thus, we are naturally led to the following question:

Question Q3: How to support decision-makers in choosing the ethical principle to compute a consensus value system?

1.2.1.2 Research Project Planning

As mentioned in Section 1.1.3.2, administrative staff from research institutions still manually allocate researchers' hours in research projects. Planning such dedication allocations while considering typical constraints in research project planning and respecting researchers' preferences is a convoluted and time-consuming task. Furthermore, the conditions of a research project may change over time. For instance, researchers might join or leave projects, new projects might get accepted, and deadlines might get extended, among other changes. As a result, the allocation plan computed for the upcoming months may become invalid. For this reason, explicitly accounting for uncertainty in project conditions is particularly critical in this domain.

Although similar problems, such as the *Workforce Assignment Problem* [166], have been studied in the literature, the specific problem of allocating researchers' hours

in this context has not yet been formally addressed. Thus, the following research question is to formalise the problem itself:

Question Q4: How can we formalise the *Research Project Planning* problem as a MIP while accounting for uncertainty?

1.2.1.3 Bachelor's Degree Planning

As discussed in Section 1.1.3.3, selecting university courses is a complex task that must comply with the constraints of bachelor's curricula while considering students' career objectives. Given the growing need for automated tools that assist students in planning their academic paths toward desired professions, this thesis aims to formalise the degree planning process as an optimisation problem. This leads us to the following research question:

Question Q5: How do we formalise the problem of computing a Bachelor's Degree Plan that considers students' career preferences as a MIP?

1.2.2 Methodological Research Questions

As discussed in Section 1.1.1, providing explanations to users of an optimisation-based DSS fosters trust in the system's recommendations. Hence, this thesis is devoted to explaining the outcomes of such systems. Specifically, we aim to generate contrastive explanations that answer user queries. Since the underlying optimisation problems of our DSSs are formalised as MIPs, we focus on explaining the outcomes when solving an optimisation problem modelled in this way. Therefore, we pose the following research question:

Question Q6: Can we develop a general methodology to generate contrastive explanations that answer the questions posed by a user about the outcome of optimisation problems?

Due to the complexity of this question, we can further divide it into two sub-questions.

Question Q6.1: How can we extract and structure the relevant information required to generate an explanation for a user question?

As mentioned above, the social sciences literature suggests that humans tend to seek contrastive explanations that refer to reasons or causes, which clarify why one outcome occurred instead of another [126]. Therefore, extracting and structuring the reasons that justify the outcome of an optimisation problem is a core challenge of this thesis.

Furthermore, effective explanations should not be presented in isolation but rather as part of an interactive dialogue with the user [112, 126]. Given this requirement, a natural next step is to consider generating textual explanations that can support a conversation. To this end, we investigate whether LLMs, which exhibit outstanding language processing capabilities, can be used to generate complete and accurate explanations. This step leads to the following research question:

Question Q6.2: Can we develop an explanatory methodology for MIPs that leverages LLMs to generate contrastive explanations?

When presenting explanations in a conversation, latency becomes a critical factor. To maintain a fluid interaction with the user, we need to generate explanations in real time. As discussed in Section 1.1.2, MIP problems are NP-Hard [21] and consequently, reasoning about their solutions can be computationally expensive, especially for large instances. In such cases, informing users about the estimated latency to generate an explanation could improve their experience interacting with DSSs. Accordingly, we consider the following two research questions:

Question Q7.1: When do the computational costs of our explanability methodology allow supporting real-time, lively interactions with users?

When computational costs are high, users may not receive an explanation in real time. However, providing an estimate of the expected waiting time could still be valuable. This raises the question of whether it is possible to predict the latency required to compute explanations.

Question Q7.2: Can we predict the computation time required to generate an explanation?

This brings us to a final research question that summarises the concerns addressed by all the previous questions in this section.

Question Q8: Is our explanatory methodology general enough to construct explanations for the optimisation problems addressed in this thesis?

1.3 Contributions

This dissertation addresses all the research questions stated in the previous section through the chapters that follow. Specifically, we advance the state of the art through both theoretical and applied advances. On the one hand, we propose a novel methodology for explaining the outcomes of optimisation problems. For the sake of clarity, in the remainder of this thesis, we will refer as "*optimal decisions*" the recommendations given by a DSS derived from the solution of an optimisation solver. On the other hand, we formalise real-world optimisation problems and solve them using existing optimisation techniques. Importantly, these problems also provide meaningful and diverse test cases for evaluating the generality and effectiveness of the proposed explainability approach. Figure 1.3 summarises the main contributions of this thesis. In the following subsections, we discuss these contributions in relation to the research questions.

1.3.1 Applied Contributions

1.3.1.1 Solving the Value Systems Aggregation Problem

Chapter 3 addresses the research questions related to the aggregation of value systems. In short, our goal is to formalise and solve this problem. To achieve this, we first define our notion of a value system, which encompasses (i) preferences over values and (ii) interpretations of values. Establishing this definition is necessary to answer research question Q1.

Contribution C1: General definition of the key components of a value system.

Section 3.3 outlines the key general components that a value system must include in order to capture pluralistic perspectives on moral values. In detail, a value system must comprise a set of moral values and a set of alternatives (e.g., actions, policies) among which an individual can make value-laden choices. Including alternatives enables defining relationships between them and moral values, thereby capturing value interpretations by specifying the extent to which each alternative promotes or demotes a moral value. Lastly, a value system must also incorporate the preferences over values. Furthermore, we illustrate how to adapt the general definition of value systems to two real-world cases.

Having established our notion of a value system, next we formally define the problem of aggregating multiple individual value systems, which is directly related to question Q2. The challenge of aggregating various individual "*judgements*" or "*preferences*" into a consensus has been widely studied in disciplines such as social choice [25, 40] and multi-expert decision-making [189, 93, 33, 140]. In our work, we address this aggregation by minimising a p-parameterised distance function, borrowed from the social choice literature [78, 79, 80], which possesses desirable social choice properties. As discussed in Section 3.2, this choice is motivated by the fact that such a distance function allows computing distinct consensuses by varying the p parameter, the so-called "*ethical principle*", which ranges from utilitarian (maximum utility) to egalitarian (maximum fairness). Given such a choice of distance function, we produce the following contribution:

Contribution C2: We formulate the aggregation of value systems as an ℓ_p-*regression problem* (also known as *norm approximation problem* [24]).

Contribution C2.1: We solve the aggregation of value systems as a two-step optimisation problem to yield a consensus on (i) the value interpretations from individuals and (ii) the preferences of individuals over moral values.

Having formalised the aggregation of value systems, we illustrate how to efficiently solve the resulting ℓ_p-regression problem using state-of-the-art algorithms on real data from two case studies. Notice that an ℓ_p-*regression* problem is a specific instance of a convex optimisation, which itself can be formulated as a particular

case of MIP. In our experimental evaluation, we analyse the impact of the ethical principle p on the obtained consensus value system. Moreover, we quantify the price to be paid when deviating from the majority to include minority views. Thus, with respect to research question Q3, we contribute as follows:

Contribution C3: We provide guidelines for decision-makers seeking to derive a consensus from individual value systems in accordance with a chosen ethical principle.

Although the results differ between the two case studies, because each involves distinct data, we identify common findings that can guide decision-makers in selecting the ethical principle for computing a consensus value system. This choice is crucial, as the selected ethical principle determines whether the consensus more closely reflects the majority's value system or provides greater fairness toward the moral values of minorities.

1.3.1.2 Solving the Research Project Planning Problem

In Chapter 4, we present our approach for solving the *Research Project Planning* (RPP) problem. In this section, we outline the main contributions, which address research question Q4.

As previously discussed, research project planning is a complex endeavour that involves numerous constraints. A research institution must allocate researchers' hours to projects while ensuring that individual monthly hour limits are not exceeded, project dedication requirements are met, and project budgets are not surpassed. Moreover, researchers have preferences regarding the number of hours devoted to each project they are involved in.

Specifically, we cast such a problem as a constrained ℓ_2-regression problem, which can be solved with state-of-the-art solvers. In particular, we examine two model variants: (i) a hard-constrained model, in which exceeding project budgets is strictly prohibited, and (ii) a soft-constrained model, where budget exceedance is permitted but penalised in the allocation. Additionally, both models are parameterised to

balance the importance of satisfying researchers' preferences with the efficient use of project budgets.

Up to this point, we have assumed that the initial projects' data remain unchanged throughout the planning horizon. However, in reality, the conditions of research projects evolve over time as new projects are initiated, deadlines shift, and researchers join and leave teams. To address this inherent uncertainty, we enhance our model with *Stochastic Robust Approximation* (SRA) [24]. This approach accounts for the fact that project data is not fixed but can vary across different scenarios. Instead of minimising a deterministic objective function, *Stochastic Robust Approximation* (SRA) minimises the expected value of the objective function under such variations. Although solving SRA problems is often computationally challenging, prior work [17] has shown that convex formulations such as *Linear Programming* (LP) and *Quadratic Programming* (QP) can be tractable. In our case, tractability is achieved by assuming that the input data may take only a finite number of variations. This allows us to obtain robust allocations that remain reliable even as project conditions evolve. Thus, we contribute as follows:

Contribution C4: We formulate the *Research Project Planning* as a MIP problem that allocates researchers' hours to projects with the objective of meeting researchers' preferences while efficiently utilising project budgets.

Contribution C4.1: We extend this formulation to account for uncertainty by modelling the problem as a *Stochastic Robust Approximation* with a finite number of variations in the input data.

Having modelled the RPP problem, we solve it using state-of-the-art solvers and evaluate our approach with real project data from our institution, the *Artificial Intelligence Research Institute* (IIIA-CSIC). Our evaluation focuses on analysing the impact of the model parameters that balance satisfying researchers' preferences with efficiently utilising project budgets. The results demonstrate that our approach is both scalable and robust to uncertainty. Finally, we implemented our approach into a prototype DSS, which is currently being tested within our institution.

1.3.1.3 Solving the Bachelor's Degree Planning Problem

Planning the courses of a bachelor's degree is a complex task because students must comply with university constraints while also considering their career objectives. At the same time, students may not always know which skills are required for their desired job(s), nor which courses provide those skills. To address this challenge, we allow students to express their preferences for the job(s) they wish to pursue. With this in mind, we formalise the *Degree Planning Problem* (DPP), which aims to schedule courses across semesters so that students acquire as many of the skills relevant to their preferred job(s) as possible. Along these lines, our contribution is the following:

Contribution C5: We formalise the *Degree Planning Problem* as a MIP that schedules university subjects across semesters while maximising the "coverage" of the required skills for a student's desired job(s).

As we later discuss in Chapter 5, the objective of covering the required skills for a student's preferred job(s) is inherently non-linear, which prevents the direct use of state-of-the-art solvers. To overcome this limitation, we introduce a non-trivial linearisation of our model by defining additional auxiliary variables.

Through several use cases, we demonstrate that our approach serves as a valuable decision-support tool in two key contexts: (i) for students, to plan their academic pathways and (ii) for curriculum designers, to evaluate how effectively university programs cover the skill requirements of the job market. To this end, we validate our approach using real-world data from the Bachelor's Degree in Information and Communications Technology at Western Sydney University, together with the Skills Framework for the Information Age (SFIA) dataset [29]. Lastly, we implement a demo prototype of our scheduling approach.

1.3.2 Methodological Contributions

After formalising the underlying optimisation problems of the DSSs as MIPs and evaluating their solutions, we now address the challenge of explaining the outcomes

of the various DSSs to foster trust in their recommendations. In particular, we focus on generating contrastive explanations to answer user questions.

While there exist works on explainability for optimisation problems, to the best of our knowledge, no prior work has addressed contrastive explanations for MIP problems in general, since they are tailored to specific optimisation problems [76, 150]. These approaches allow users to challenge a solution by posing queries about why a particular outcome was obtained instead of an alternative. Then, they recompute a solution that incorporates the suggested alternative and compares the two outcomes numerically. However, such methods do not explicitly identify the underlying reasons why one recommendation is preferable to another, which is key in explainability. Consequently, our challenge is to develop a general approach, i.e., capable of explaining the outcomes of various optimisation problems, that generates "good" explanations according to the social sciences [126].

Leveraging the fact that we encode each optimisation problem as a MIP problem, we propose a general approach that provides contrastive explanations for MIPs. As in existing works, we allow users to challenge the optimal solution to a problem by posing queries. However, instead of re-solving the problem with additional constraints derived from the query, we identify the constraints that prevent the MIP solver from finding a better solution that accommodates the user's query. We then exploit the relationships among these constraints to construct a graph, the so-called *graph of reasons*, which captures the reasons why the original optimal solution is better than the suggested alternative. Finally, we use the information captured by the graph as structured input to a LLM, which generates a textual explanation.

Chapter 6 discusses our explainability approach. Thus, with respect to research question Q6, we make the following contribution to the state of the art.

Contribution C6: We propose a general methodology that leverages AI techniques to generate contrastive explanations for users' queries about optimal decisions.

To achieve this, we first identify and structure the necessary information to build our explanations.

Contribution C6.1: Thanks to constraint reasoning techniques, we extract and relate the reasons into the so-called graph of reasons why an optimal decision is better than a user-suggested alternative.

The next challenge is to construct a textual explanation from the reasons encoded in the graph. This step is crucial, since our explanations are conceived as part of a dialogue.

In that context, we investigate whether it is possible to exploit the natural language processing capabilities of LLMs to generate textual explanations. However, explaining why a particular solution is proposed instead of another is highly challenging, even for human experts, because optimisation problems often involve many constraints and decision variables. Given that LLMs are known to struggle with abstract reasoning tasks [75], we propose to use them instead as a tool for transforming the information captured in the graph into textual explanations. Accordingly, our contribution regarding question Q6.2 is the following:

Contribution C6.2: We leverage *Large Language Model*s to generate textual explanations based on a *graph of reasons*.

Chapter 6 discusses how we place constraint reasoning techniques at the service of LLMs to generate textual explanations that are complete and informative for users.

Furthermore, since our goal is to provide explanations in a conversation, it is crucial to compute them in real time. Although our approach avoids re-solving the optimisation problem to generate an explanation, extracting the constraints that prevent a better solution accommodating the user's query can still be computationally costly. Consequently, we must evaluate the empirical hardness of computing an explanation. This motivates the following contribution regarding research question Q7.1.

Contribution C7.1: We quantify the empirical hardness of computing an explanation.

In chapter 7, we present the results of our experimental evaluation. We test our approach on (i) the problems addressed in this thesis, and (ii) *Resource-Constrained Project Scheduling Problem* [91] and the *Winner Determination Problem* [56]. We

choose to evaluate our approach on these additional problems because they are well studied in the optimisation field and provide benchmark instances suitable for assessing scalability [181, 116], i.e., the cost of explanations for very large optimisation problem instances.

For cases where generating explanations is computationally expensive, informing the user of a DSS about the estimated computation time would be valuable. Indeed, providing users with the estimated waiting time has been shown to increase users' tolerance for delays [136] and reduce frustration during system interaction [135]. Thus, this presents us with a key challenge: predicting the runtime required to generate an explanation for a given problem instance. While prior work exists on predicting the runtime of *solving* optimisation [114, 115, 96] or satisfiability problems [203, 96], to the best of our knowledge, no method currently exists to predict the computation time needed to generate an explanation.

A key challenge in predicting the explanation runtime is that it depends not only on the computational cost of solving the underlying problem instance but also on the specific query posed by the user, as shown by the results from Contribution C7.1. Specifically, for the same problem instance, different user queries can yield explanations that require vastly different runtimes. Therefore, to predict the runtime required to compute an explanation, we adapt a well-established technique from the literature that has proven effective for predicting solving times for combinatorial optimisation problems using ML models [96]. In detail, we extend this technique by training ML models employing features that jointly represent the optimisation problem instance and the user query. Specifically, to reliably estimate the explanation runtime, we combine (i) established features that capture the specific problem instance hardness [114] with (ii) novel features defined to capture the query to be explained. We therefore address Question Q7.2 with the following contribution discussed in Section 7.3.

Contribution C7.2: Using ML techniques, we demonstrate how to learn models that predict explanation time.

In particular, we define a novel set of features to characterise user queries, which

we employ to train ML models. We evaluate the proposed approach across problem instances in the *Resource-Constrained Project Scheduling Problem* (RCPSP) and *Winner Determination Problem* (WDP) domains. Results show that features characterising user queries significantly reduce prediction error.

We apply the proposed methodology within the optimisation-based DSSs addressed in this thesis. Chapter 6 showcases its application across the three domains, and we evaluate it with respect to both computational cost and the quality of the generated explanations, particularly their completeness and readability. This brings us to the final contribution of the thesis.

Contribution C8: We illustrate how to apply our methodology to generate explanations for the three optimisation problems addressed in this thesis.

1.4 Thesis Outline

Following the concepts introduced in this chapter, the remainder of this thesis is structured as follows:

- **Chapter 2** introduces well-known optimisation concepts, which constitute the basis to formalise the optimisation problems addressed in this dissertation. Then, it reviews the related work and positions the contributions of this thesis with respect to existing literature in optimisation and explainability.
- **Chapter 3** presents our notion of value systems and formalises the problem of aggregating individual value systems into a consensus. We cast such a problem as a ℓ_p-regression problem, which we solve with state-of-the-art optimisation techniques. Then, we evaluate it with real-world case studies and provide guidelines for decision-makers on selecting an appropriate ethical principle.
- **Chapter 4** presents the *Research Project Planning* (RPP) problem. We formalise the problem as a constrained ℓ_2-regression model and propose both hard- and soft-constraint models to address budget constraints. Then, we extend the model to account for uncertainty through *Stochastic Robust Approxima-*

tion. We evaluate the proposed approach with real project data and showcase a prototype DSS tailored to our institution.

- **Chapter 5** introduces the *Degree Planning Problem* (DPP). We formalise the problem as a MIP that generates personalised degree plans maximising skill acquisition for students' desired career paths while complying with bachelor's degree requirements. We also propose a non-trivial linearisation to enable the use of state-of-the-art solvers and validate our approach with real-world data.
- **Chapter 6** develops our general methodology for generating explanations of optimisation-based DSS. We present an approach to compute contrastive explanations for MIP problems by combining constraint reasoning techniques and LLMs. We apply our explanability methodology to the three optimisation problems presented in previous chapters and assess the quality of the generated explanations in terms of completeness and readability.
- **Chapter 7** provides an extensive experimental evaluation. We investigate the empirical harness to compute explanations. Moreover, we predict the computational cost of computing explanations with ML models that use features to characterise the optimisation problem instances and the user queries.
- **Chapter 8** concludes the thesis by summarising the main contributions and outlining directions for future research.

1.5 Publications

This section presents all the scientific publications derived from this dissertation. In addition, we discuss other publications related to the topics addressed in this thesis and conclude with a list of papers currently under review (as of November 2025).

First, we list the publications that disseminate the work presented in this thesis.

- [P1] R. Lera-Leri, F. Bistaffa, M. Serramia, M. Lopez-Sanchez and J. Rodriguez-Aguilar, 'Towards pluralistic value alignment: Aggregating value systems through

ℓ_p-regression,' in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 780–788

- [P2] R. X. Lera-Leri, E. Liscio, F. Bistaffa, C. M. Jonker, M. Lopez-Sanchez, P. K. Murukannaiah, J. A. Rodriguez-Aguilar and F. Salas-Molina, 'Aggregating value systems for decision support,' *Knowledge-Based Systems*, vol. 287, p. 111 453, 2024
- [P3] R. X. Lera-Leri and F. Bistaffa, 'A robust and scalable approach to meet user preferences in research project planning,' in *Proceedings of the European Conference on Artificial Intelligence 2024*, IOS Press, 2024, pp. 4516–4523
- [P5] R. X. Lera-Leri, F. Bistaffa, T. Trescak and J. A. Rodríguez-Aguilar, 'Computing job-tailored degree plans towards the acquisition of professional skills,' *Annals of Operations Research*, vol. 351, pp. 2095–2128, 2025
- [P4] R. X. Lera-Leri, F. Bistaffa, A. Georgara and J. A. Rodríguez-Aguilar, 'Exploiting constraint reasoning to build graphical explanations for mixed-integer linear programming,' in *Proceedings of the International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*, Springer, 2025, pp. 39–57 (Published in Lecture Notes in Artificial Intelligence)

Our approach to aggregating value systems is first introduced in [P1], where we define a notion of value system, formalise the aggregation as an ℓ_p-regression problem, and evaluate it through a use case. This work is later extended in [P2], where we generalise the notion of value system to apply the approach to additional use cases and provide guidelines to support decision-makers in selecting ethical principles to obtain a consensus value system.

The other two optimisation problems addressed in this thesis, the RPP and the *Degree Planning Problem* (DPP), are formalised and solved in [P3] and [P5], respectively. In both works, we test our approaches using real-world datasets.

Lastly, in [P4], we present our methodology, based on constraint reasoning, for extracting and structuring the information required to build explanations. In that

work, we also empirically evaluate the computational cost of extracting such information for well-established optimisation problems.

Furthermore, we published additional papers related to this thesis.

- [P6] E. Liscio, R. Lera-Leri, F. Bistaffa, R. I. Dobbe, C. M. Jonker, M. Lopez-Sanchez, J. A. Rodriguez-Aguilar and P. K. Murukannaiah, 'Value inference in sociotechnical systems,' in *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '23, London, United Kingdom: IFAAMAS, 2023, pp. 1774–1780
- [P7] E. Liscio, R. Lera-Leri, F. Bistaffa, R. I. Dobbe, C. M. Jonker, M. Lopez-Sanchez, J. A. Rodriguez-Aguilar and P. K. Murukannaiah, 'Inferring values via hybrid intelligence,' in *HHAI 2023: Augmenting Human Intellect*, IOS Press, 2023, pp. 373–378
- [P8] T. Trescak, R. Lera-Leri, F. Bistaffa and J. A. Rodriguez-Aguilar, 'Agent-assisted life-long education and learning,' in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022

In [P6] and [P7], we propose a pipeline of technical AI-based components to extract the societal value systems. In [P8], we discuss how agent technologies can be employed to empower education, develop skills, and maximise career prospects by curating personalised education pathways.

In addition to scientific publications, our institution, the "*Consejo Superior de Investigaciones Científicas*" (CSIC), has registered two software prototypes developed as part of this thesis.

1. R. X. Lera-Leri, F. Bistaffa, J. A. Rodríguez-Aguilar, T. Trescak, "Bachelor's Degree Planner". 2025
2. R. X. Lera-Leri, F. Bistaffa, J. A. Rodríguez-Aguilar, "Learning Path Planner". 2024

Software 1 implements our approach to solving the DPP problem within a prototype designed for students who wish to plan their courses to maximise the acquisition of

skills relevant to their preferred job(s). Software 2 adapts this approach to recommend personalised learning paths composed of online courses rather than a bachelor's degree curriculum. As in the bachelor's degree case, the objective is to maximise the coverage of the skills required for students' desired job(s).

Lastly, we cite two publications related to Chapters 6 and 7. The first work is currently under review, while the second has been accepted for publication.

1. R. X. Lera-Leri, F. Bistaffa, A. Georgara and J. A. Rodríguez-Aguilar, "Constraint Reasoning and Large Language Models Join Forces: Providing Contrastive Explanations for Real-World Optimisation-Based Applications."

2. R. X. Lera-Leri, F. Bistaffa and J. A. Rodríguez-Aguilar, "Predicting the Cost of Explanations in Optimisation Problems," in *Proceedings of the International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*.

In the first work, we deploy constraint reasoning techniques (as discussed in [P4]) to support LLMs in generating textual explanations. We evaluate the proposed approach on two optimisation problems: the DPP and the RCPSP. In the second work, we train ML models to estimate the runtime to generate explanations.

Chapter 2

Background and Related Work

This chapter introduces the background concepts and reviews related work in the areas addressed by this thesis. Specifically, we first present the basic concepts of *Mixed-Integer Programming* (MIP), which will later be used to formalise the optimisation problems studied here. We then survey prior work on the specific optimisation problems considered in this thesis. Finally, we examine the literature on explaining the outcomes of optimisation problems.

2.1 Mixed-Integer Programming

Mathematical optimisation is concerned with finding the best possible solution to a problem subject to a set of constraints. More precisely, an optimisation problem seeks a solution that (i) satisfies all given constraints and (ii) minimises or maximises a specified objective function. For example, in scheduling problems, the goal is typically to build a schedule of tasks that respects the scheduling constraints while minimising the total project duration [89]. In the case of packing problems, the objective is to select a subset of items that maximises revenue without exceeding capacity constraints [103]. Similarly, in combinatorial auctions, the objective is to select the subset of bins (collections of items with associated prices) that maximises total revenue [56].

A well-established approach to formalising optimisation problems is *Mixed-Integer Programming* (MIP). MIP problems involve both discrete and continuous decision variables [198]. In simple terms, their solutions include variables that can take frac-

tional or real values, alongside others that are restricted to integers. More formally, a MIP problem can be expressed in the following form:

$$\text{minimise} \quad f_0(\mathbf{x}, \mathbf{y}) \tag{2.1a}$$

$$\text{subject to} \quad f_i(\mathbf{x}, \mathbf{y}) \leq 0, \quad \forall i = 1, \ldots, m, \tag{2.1b}$$

where $\mathbf{x} \in \mathbb{R}^n$ is a vector of continuous decision variables, $\mathbf{y} \in \mathbb{Z}^k$ is a vector of integer decision variables, and $f_i : \mathbb{R}^n \times \mathbb{Z}^k \to \mathbb{R}$ is any real-valued function. In more detail, (2.1a) defines the objective function, while (2.1b) represents the problem constraints. Solving this problem amounts to finding the values of the decision variables that minimise the objective function while satisfying all constraints. Although (2.1b) is expressed as a set of inequality constraints, MIPs can also encode equality constraints by enforcing two inequalities, namely $f_i \leq 0$, $-f_i \leq 0$, which implies $f_i = 0$.

MIP constitutes a general class of optimisation problems. A well-known special case is *Integer Programming* (IP), where all decision variables are restricted to integer values [199]. Conversely, continuous optimisation corresponds to the subclass of problems in which all variables are continuous [7]. Furthermore, there exist different subclasses of MIP depending on the type of objective function and constraints. For example, *Mixed-Integer Linear Programming* (MILP) is the class of MIP problems whose objective and constraints are linear, i.e.,

$$\text{minimise} \quad \mathbf{c}^T\mathbf{x}, \tag{2.2a}$$

$$\text{subject to} \quad D\mathbf{x} \leq \mathbf{d}, \tag{2.2b}$$

$$A\mathbf{x} \sim \mathbf{b}, \tag{2.2c}$$

where $\mathbf{c} \in \mathbb{R}^n$, $D \in \mathbb{R}^{m \times n}$, $\mathbf{d} \in \mathbb{R}^m$, $A \in \mathbb{R}^{k \times n}$, and $\mathbf{b} \in \mathbb{R}^k$. *Mixed-Integer Quadratic Programming* (MIQP) allows for quadratic objective functions [145]; and *Mixed-Integer Quadratically Constrained Programming* (MIQCP) further extends this by including quadratic constraints [32]. More generally, *Mixed-Integer Convex Programming* (MICP) comprises the optimisation problems where the functions f_i are convex [122]. Figure 2.1 represents the different classes of MIP problems. Due to this gener-

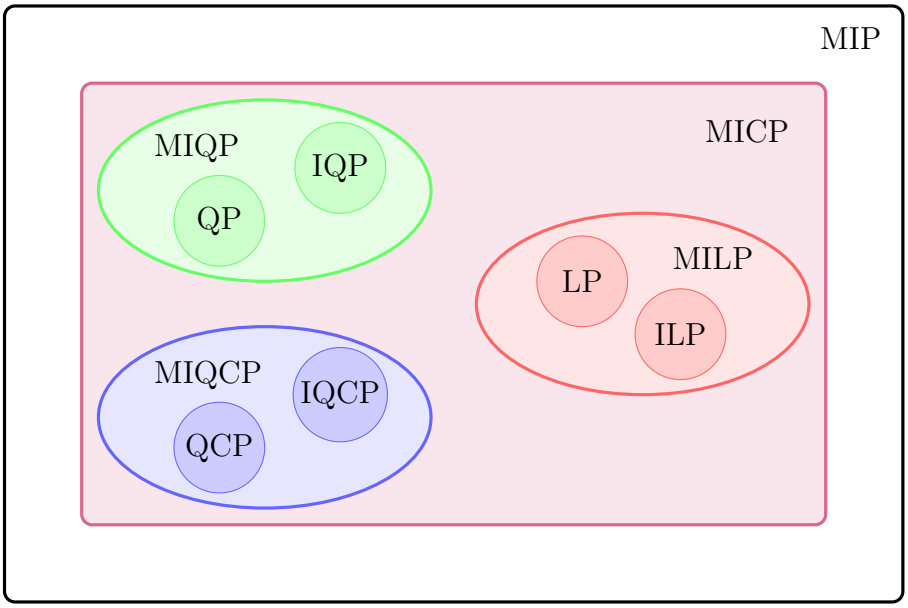


Figure 2.1: Classes of MIP problems. In particular, MILP, MIQP, and MIQCP can be further divided into two subclasses: those with only discrete variables, such as ILP, IQP, and IQCP, and those with only continuous variables, such as LP, QP, and QCP. MILP, MIQP, and MIQCP are specific cases of MICP.

ality, MIP enables the formalisation of a wide range of optimisation problems across diverse application domains.

In general, MIP problems (with integer variables) are NP-Hard [198] and are extremely expensive to solve. On the other hand, optimal solvers such as CPLEX [55] and Gurobi [86] can tackle some particular classes of MICP problems such as MILP, MIQP, and MIQCP. In the next section, we introduce basic concepts of convex optimisation, which serve as the foundation for formalising the optimisation problems addressed in this dissertation.

2.1.1 Convex Optimisation

A convex function f is a function such that

$$f\Big(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}\Big) \leq \alpha f(\mathbf{x}) + (1-\alpha)f(\mathbf{y}), \tag{2.3}$$

where $0 \leq \alpha \leq 1$ and $\forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. A *convex optimisation problem in standard form* [24, Section 4.2.1] is one of the form

$$\text{minimise} \;\; f_0(\mathbf{x}),$$

$$\text{subject to } f_i(\mathbf{x}) \leq 0, \qquad i = 1, 2, \ldots, m \tag{2.4}$$

$$\mathbf{a}_i^T \mathbf{x} - b_i = 0, \qquad i = 1, 2, \ldots, p \tag{2.5}$$

where $f_0, \ldots, f_m$ are convex functions, (2.4) are inequality constraints and (2.5) are linear equality constraints. Convex optimisation problems are very appealing from a computational perspective because of their fundamental property that ensures that any locally optimal point is also (globally) optimal [24].

A well-known example of convex optimisation problem is *Least Squares* (LS),[1] whose objective function is of the form

$$||A\mathbf{x} - \mathbf{b}||_2^2 = \mathbf{x}^T A^T A\mathbf{x} - 2\mathbf{b}^T A\mathbf{x} + \mathbf{b}^T \mathbf{b}, \tag{2.6}$$

where $A \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$. Specifically, the *unconstrained* version of the problem has a well-known analytical solution $\mathbf{x} = (A^T A)^{-1} A^T \mathbf{b}$, where A^T is the transpose matrix of A. On the other hand, there is no analytical solution for the *constrained* version of the problem (i.e., with equality or inequality constraints). However, numerical methods can solve it very efficiently.

More in general, a wealth of commercial and non-commercial off-the-shelf solvers are available for solving a wide range of types of convex optimisation problems, such as *Linear Programming* (LP), *Quadratic Programming* (QP), or *Quadratically Constrained Programming* (QCP) by implementing state-of-the-art algorithms based on the simplex method or interior-point methods [24]. Furthermore, advanced solvers can also handle problems with discrete variables when their relaxation, that is, the optimisation problem obtained by removing the integrality constraints on the decision variables, is convex. In the following section, we introduce p-norms, a widely used convex objective function. As will be shown later, we employ p-norms as the objective function in each of the optimisation problems addressed in this dissertation.

[1] According to Boyd and Vandenberghe [24], "it is well-known that *Linear Programming* and *Least Squares* have a fairly complete theory, arise in a variety of applications, and can be solved numerically very efficiently".

2.1.2 Objective Functions Involving Norms

A p-norm [24] is defined as

$$\|\mathbf{x}\|_p = \left(\sum_i |x_i|^p \right)^{1/p}, \tag{2.7}$$

where $p \in [1..\infty]$. Importantly, p-norms are convex functions for all values of p within this range.

Theorem 2.1.1. *p-norms are convex functions.*

Proof. To demonstrate that norms are convex functions, we must demonstrate that they comply with the definition of a convex function (2.3), i.e., we must demonstrate the following:

$$\|\alpha\mathbf{x} + (1-\alpha)\mathbf{y}\| \leq \alpha\|\mathbf{x}\| + (1-\alpha)\|\mathbf{y}\|.$$

To do so, we develop the left-hand side by applying the triangle inequality [69] such that

$$\|\alpha\mathbf{x} + (1-\alpha)\mathbf{y}\| \leq \|\alpha\mathbf{x}\| + \|(1-\alpha)\mathbf{y}\|.$$

Lastly, we apply $\|k\mathbf{x}\| = |k| \cdot \|\mathbf{x}\|$, and considering that $0 \leq \alpha \leq 1$,

$$\begin{aligned} \|\alpha\mathbf{x} + (1-\alpha)\mathbf{y}\| &\leq |\alpha| \cdot \|\mathbf{x}\| + |1-\alpha| \cdot \|\mathbf{y}\| \\ &\leq \alpha\|\mathbf{x}\| + (1-\alpha)\|\mathbf{y}\|. \end{aligned}$$

□

Norm Minimisation Problem (NMP) are a family of optimisation problems well-known in the literature [24, 35] defined as

$$\text{minimize} \quad \|A\mathbf{x} - \mathbf{b}\|_p, \tag{2.8a}$$

$$\text{subject to} \quad \mathbf{x} \in \mathcal{F}, \tag{2.8b}$$

where $A \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathcal{F}$ is the set of feasible solutions. Notice that the convexity of norms guarantees that every local optimum is also a global optimum [24] in *Norm Minimisation Problem* (NMP). Importantly, this property holds only when

the argument of the p-norm is a linear function. NMPs are particularly interesting because they can be transformed either to a MILP, a MIQPs, or a MIQCP depending on the value of p (see Figure 2.2), i.e., they are solvable with off-the-shelf solvers for any p.[2]

The most common p-norms used in the literature are $p = \{1, 2, \infty\}$ [24, 165]. In the case of minimising the *Manhattan* norm ($p = 1$), i.e., $\|A\mathbf{x} - \mathbf{b}\|_1 = |A_1\mathbf{x} - b_1| + \cdots + |A_m\mathbf{x} - b_m|$, the problem can be cast as an MILP such that

$$\text{minimise} \quad \mathbf{1}^T\mathbf{t}, \tag{2.9a}$$

$$\text{subject to} \quad -\mathbf{t} \leq A\mathbf{x} - \mathbf{b} \leq \mathbf{t}, \tag{2.9b}$$

$$\mathbf{x} \in \mathcal{F}, \tag{2.9c}$$

where $\mathbf{t} \in \mathbb{R}^m$ is a vector of auxiliary variables [24]. In the minimization of the Maximum norm ($p = \infty$), i.e., $\|A\mathbf{x} - \mathbf{b}\|_\infty = \max_{i=1,\ldots,m} \left(|A_i\mathbf{x} - b_i|\right)$, the problem can be also cast as an MILP such that

$$\text{minimise} \quad t, \tag{2.10a}$$

$$\text{subject to} \quad -t\mathbf{1} \leq A\mathbf{x} - \mathbf{b} \leq t\mathbf{1}, \tag{2.10b}$$

$$\mathbf{x} \in \mathcal{F}, \tag{2.10c}$$

where $t \in \mathbb{R}$ is an auxiliary variable [24]. In addition, in the minimisation of the Euclidean norm ($p = 2$), i.e., $\|A\mathbf{x} - \mathbf{b}\|_2$, the problem can be cast as an MIQP by squaring the 2-norm such that

$$\text{minimise} \quad \|A\mathbf{x} - \mathbf{b}\|_2^2 = \mathbf{x}^T A^T A\mathbf{x} - 2\mathbf{b}^T A\mathbf{x} + \mathbf{b}^T\mathbf{b}, \tag{2.11a}$$

$$\text{subject to} \quad \mathbf{x} \in \mathcal{F}. \tag{2.11b}$$

The optimisation problems involving the minimisation of norms with $p \notin \{1, 2, \infty\}$ can be solved by employing the techniques discussed by Alizadeh and Goldfarb [6, Section 2.3g], who show how to represent inequalities involving p-norms as MIQCP, hence enabling the use of off-the-shelf solvers such as CPLEX or Gurobi. Figure

[2]The widely used optimisation library CVXPY can automatically transform a NMP to the appropriate problem depending on p.

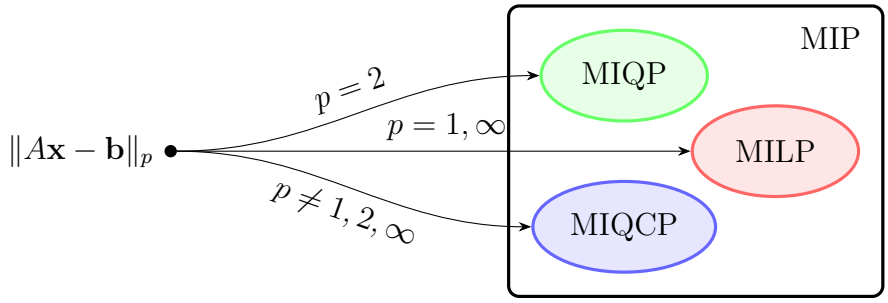


Figure 2.2: Subclasses of MIPs and p-norm minimisation problems depending on the value of p.

2.2 summarises the above discussion, showing the resulting optimisation problems depending on the choice of p-norm.

Having provided the necessary background on MIP to formalise the optimisation problems tackled in this dissertation, we now introduce basic concepts in the infeasibility of optimisation problems. Such definitions will be later used in Chapter 6 to generate contrastive explanations.

2.2 Infeasibility in Optimisation Problems

After introducing MIP, a common technique used to formalise and solve complex optimisation problems, we now focus on infeasibility, which plays a crucial role in the construction of explanations throughout this thesis. In particular, analysing infeasibilities will allow us to reason about why certain user-suggested alternatives cannot be realised within the optimisation model, as we later explain in Chapter 6.

Infeasibility in optimisation problems occurs when no solution satisfies all the problem's constraints simultaneously, often due to conflicting constraints. A common technique employed to detect and analyse infeasibilities is the extraction of the *Irreducible Infeasible Subsystem* (IIS) of constraints. In our explainability approach, we employ IISs as a technique to identify and extract the minimal set of constraints that explain why an optimal solution is preferred over an alternative suggested by the user. In the remainder of this section, we further the concept of IIS.

More precisely, the IIS is a minimal set of inconsistent constraints. However, to

properly define the notion of IIS, we must first define the notions of constraint and system of constraints. We borrow the definition of constraint from [124]:

Definition 2.2.1. *Given a sequence of distinct variables $V = \langle x_1, \ldots, x_k \rangle$ and their associated finite domains $\mathcal{D}_1, \ldots, \mathcal{D}_k$, a constraint c is a subset of $\mathcal{D}_1 \times \cdots \times \mathcal{D}_k$. Thus, a constraint c specifies the assignments allowed by the variables of V.*

In other words, a constraint is a condition that variables must satisfy. Then, a constraint system is a set of constraints C. Hence, we define an infeasible system as follows.

Definition 2.2.2. *An Infeasible System of constraints is a set of constraints C such that no assignment of V exists that satisfies all the constraints in C.*

Having defined an infeasible system of constraints, we define an IIS borrowing the definition from [43]:

Definition 2.2.3. *An IIS is a subset of constraints within an infeasible system that is in itself infeasible, but any proper subset of the IIS is feasible.*

Finding an IIS is crucial in *Constraint Satisfaction Problem* (CSP) because it identifies the constraints preventing us from obtaining a feasible solution. Thus, many algorithms have been developed to find the IIS for LP [190, 77, 141, 42], MILP [84] or Non-Linear Integer Programming [41]. Finding the IIS involves iteratively analysing its infeasible system and identifying the minimal subset of constraints responsible for infeasibility. Some examples of algorithms that find IISs are: the Deletion Filter [44], which iteratively deletes constraints and checks the feasibility of the system; the Additive Method [187], which iteratively adds constraints to an initially empty set and tests whether the set is feasible; or the combination of both methods [84]. Although computing IISs can be costly, state-of-the-art solvers efficiently compute them.

2.3 Related Work on Optimisation

In this section, we introduce a well-studied optimisation problem from the Operations Research (OR) literature: the *Resource-Constrained Project Scheduling Problem* (RCPSP). In brief, the RCPSP consists of scheduling a set of activities while satisfying a given set of resource and precedence constraints. Although numerous variants of the problem exist, the primary objective is typically to minimise the total project duration.

We focus on the RCPSP because of its similarities with both the *Research Project Planning* (RPP) and *Degree Planning Problem* (DPP) problems formalised and solved in Chapters 4 and 5, respectively. Moreover, because of its widespread use as a benchmark problem [91], we employ the RCPSP to assess our explainability methodology in a domain that is well-studied by the OR community and facilitates comparison with existing work. Finally, the RCPSP and its variants have been extensively studied with benchmark problem instances, which are suitable for studying the scalability of our explainability approach.

Intuitively, the RCPSP is an NP-hard problem [21] consisting of activities that must be scheduled to meet precedence and resource constraints while minimising the project's makespan [90, 91]. We borrow this formalisation from the single mode variant of the RCPSP problem in [108]. We consider a project with J activities to complete. W.o.l.o.g. activity 1 is the only start activity, and activity J is the only finish activity. Each activity j has a given processing time d_j, and a set of R_j resources that j needs to complete. Moreover, each activity j has a set of activity predecessors $\mathcal{P}_j$, which implies that activity j can only start when each of its predecessors $h \in \mathcal{P}_j$ has been completed. The planning horizon is divided into T time units labelled as $t = 1, \ldots, T$, where T is a makespan upper bound. Given the precedence relations and the activity duration, we can calculate the time windows, i.e., intervals $[EF_j, LF_j]$, with the earliest and latest finishing times of each activity. In addition, we consider that at each time t, there are K_r units of resource $r \in R$ available. A given activity j requires an amount k_{jr} of resource r per period. Given those ele-

ments, the goal is to find a feasible schedule that minimises the total duration, i.e., minimises the finishing time of activity J.

We present the formalisation of the RCPSP as a MILP as follows.

$$\text{minimise} \quad \Phi(x) = \sum_{t=EF_J}^{LF_J} t \cdot x_{Jt} \tag{2.12a}$$

$$\text{s.t.} \sum_{t=EF_j}^{LF_j} x_{jt} = 1,\ j = 1, \ldots, J, \tag{2.12b}$$

$$\sum_{t=EF_h}^{LF_h} t \cdot x_{ht} \leq \sum_{t=EF_j}^{LF_j} (t - d_j) \cdot x_{jt},\ j = 2, \ldots, J,\ h \in \mathcal{P}_j, \tag{2.12c}$$

$$\sum_{j=1}^{J} k_{jr} \sum_{q=\max(t,EF_j)}^{\min(t+d_j-1,LF_j)} x_{jq} \leq K_r,\ r \in R,\ t = 1, \ldots, \bar{T}, \tag{2.12d}$$

$$x_{jt} \in \{0, 1\},\ j = 1, \ldots, J, \quad t = EF_j, \ldots, LF_j \tag{2.12e}$$

where x_{jt} are the decision variables expressing whether an activity j is completed on time t. The goal of RCPSP (2.12a) is to minimise the finishing time of the last activity J. Constraint (2.12b) forces the scheduling of each activity once. Constraint (2.12c) ensures the precedence relations, i.e., every activity must start after its predecessors have been completed. Constraint (2.12d) limits the use of renewable resources.

There exist numerous variations of RCPSP formalisation. Hartmann and Briskorn [90, 91] provide a survey of such variants, which include activities with multiple modes [92, 107, 181] (i.e., multiple procedures to complete an activity using different resources or amount of resources with distinct processing times), nonrenewable resources [179], diverse objectives such as the minimization of costs and the consumption of nonrenewable resources [58] or the maximization of the net present value [125]. Due to this broad spectrum of models, Brucker, Drexl, Möhring, Neumann and Pesch [30] propose a unified notation for RCPSP as well as an $\alpha|\beta|\gamma-$scheme to classify variants of RCPSP. α indicates the *resource environment*, i.e., whether the instance of the problem considers multi-modes for executing the activities and the number of resources. β establishes the *activity characteristics* such as the existence of preced-

ence relations, deadlines, or allowing preemption, i.e., allowing activity interruptions during its execution [59]. Finally, γ describes the objective function.

RCPSP is a general framework that is able to capture very similar problems as special cases of RCPSP [90]. For instance, Sprecher [180] shows that *Open Shop*, *Flow Shop*, *Job Shop*, and *Assembly Line* problems are special cases of RCPSP. Furthermore, project scheduling concepts are powerful enough to capture problems from different scenarios [90]. Hartmann [88] demonstrates that *Bin Packing*, *Knapsack Packing* and *Strip Packing* can be viewed as special cases of RCPSP for one, two, and three dimensions. In addition, Brucker and Knust [31] show that *High School Timetabling*, *University Course Timetabling*, and *Audit Scheduling Problem* can be captured as special cases of RCPSP. Finally, *Workforce Assignment Problem* [166] is a special case of RCPSP whose goal is to assign a set of activities to a set of workers. In this sense, this problem is similar to RPP, also due to the presence of constraints on the maximum number of working hours and the required dedication.

2.3.1 Differences Between the RCPSP and the problems addressed in this thesis

Although the RCPSP shares similar concepts with both the RPP and the DPP problems addressed in this thesis, these problems are formally distinct in terms of their optimisation goals and constraints, hence motivating a novel formalisation for them.

In both RPP and DPP, some elements can indeed be mapped to components of the RCPSP. For instance, work packages in the RPP or university courses in the DPP can be viewed as activities with precedence or timing constraints. Similarly, researchers' available time or course credits can be interpreted as renewable resources, and projects' duration in the RPP or sequence of semesters in the DPP can be seen as a planning horizon. These analogies explain the conceptual proximity between these problems.

However, the fundamental differences lie in their optimisation objectives. The goal of the RCPSP is typically to minimise the project's total duration, i.e., the makespan,

whereas the RPP and DPP pursue entirely different objectives. In the RPP, the objective is to maximise the alignment between researchers' preferred dedication and the actual assigned workload while efficiently allocating the available budget. Minimising the makespan would not even be meaningful in this context, as work packages have fixed start and end dates.

Similarly, in the DPP, the objective is to generate a Bachelor's Degree plan that best matches a student's desired job-related skill set. Here, the total completion time of the degree is predetermined rather than a variable to optimise. Moreover, not all subjects need to be included in the degree plan; only those that contribute most effectively to the target skill profile. In contrast, all activities must be scheduled in the RCPSP. Furthermore, some of the constraints in DPP are domain-specific and cannot be directly expressed within the RCPSP formulation. For example, at least one major has to be completed during the Bachelor's Degree. Therefore, a set of compulsory and elective subjects related to the major has to be scheduled.

Overall, while the RPP and DPP share certain structural aspects with the RCPSP, their differing objectives and domain-specific constraints prevent the direct application of existing RCPSP algorithms. These differences justify the development of novel optimisation models tailored to each problem within this thesis.

2.4 Related Work on Explainability

In this section, we begin by outlining the main properties that characterise a "*good*" explanation in the context of *Artificial Intelligence* (AI). Next, we examine recent explainability approaches for optimisation problems. Finally, we discuss research on *Minimal Unsatisfiable Subset* (MUS), an analogous concept to IIS and extensively studied in the Boolean Satisfiability (SAT) literature.

In the last decade, building explanations to justify the outcomes of "*black box*" *Machine Learning* (ML) models has been mainly the focus of the so-called *eXplainable Artificial Intelligence* (XAI) community [2, 212], while optimisation (our topic in this paper) has been studied less often in this context. Nonetheless, in this disserta-

tion, we aim to build our explanations following the properties identified by Miller *et al.* [126] in one of the most prominent works of the XAI literature. According to Miller *et al.*, a *good* explanation has to be: (i) *contrastive* (in response to particular counterfactual cases); (ii) *social* (to transfer knowledge as a part of a conversation); (iii) *selected* (humans select one or two reasons as an explanation); and (iv) *causal* (explanations with causal relationships are more important than referring to probabilities or statistical relations). Along these lines, our explanations comply with the properties outlined by Miller *et al.*because: (i) they are *contrastive*, as they explain why user-desired scenarios are not as good as the optimal solution; (ii) they are *social*, as they can be delivered in an interaction dialogue between a user and a *Decision Support System* (DSS); (iii) they are *selected*, as they are based on an irreducible infeasible set of constraints; and (iv) they are *causal*, since they reference the constraints and the objective function that restrict the output of the optimisation problem.

As mentioned above, the research on providing explanations for optimisation-based DSS is not as extensive. Some of the most commonly used techniques in explainability for optimisation include: (i) what-if analysis [76, 111], which involves comparing the optimal solution with a user-desired alternative; (ii) constraint or rule analysis [23, 100], which seeks to explain outcomes by reasoning about the problem's constraints; and (iii) visualisation methods [167, 191], which help interpret the performance of the optimisation algorithm. Since we leverage constraint reasoning techniques to provide contrastive explanations, here we discuss what-if and constraint analysis techniques.

Some approaches have been developed to explain the outcome of different classical combinatorial problems by analysing what-if scenarios [110, 76]. However, the explanations generated by such approaches lack some key properties enumerated by Miller *et al.* [126]. Although Georgara *et al.* [76] provide contrastive explanations, they refer to numerical reasons instead of referring to causal relationships. Similarly, Korikov *et al.* [111, 110] generate counterfactual explanations that indicate the necessary numerical changes in the data of the problem to comply with a user-desired solution. Moreover, these approaches are not general; each one addresses a particular

application domain. In contrast, in this thesis, we propose a domain-independent explanatory approach that generates contrastive, social, and selected explanations that refer to reasons indicating why the optimal solution is better than the one posed by the user.

Now, there exist approaches focused on constraint or rule analysis that help interpret the outcome of constrained problems. For instance, Nardi *et al.* [137] find justifications for the outcome of collective decision algorithms via analysing the fulfilled voting rules. Bogaerts *et al.* [23] give step-wise explanations for CSP via iteratively analysing the problem constraints. Additionally, some works focus on interpreting the infeasibility in SAT problems by extracting a MUS of clauses [178], a concept analogous to an IIS in the context of optimisation problems. Several works discuss how to extract multiple MUSs [118], since an infeasible system of constraints may possess an exponential number of MUSs [36]. Along these lines, some methods have been proposed to select the MUS whose constraints are the most relevant according to user preferences [100, 73]. Furthermore, there has been an effort in computing the MUS with minimal cardinality, i.e., the MUS with a minimum number of constraints [117, 98]. Crucially, the above-mentioned works on MUS focus on the interpretability of the *infeasibility* of a given *SAT Problem* by extracting a MUS. In this dissertation, we pursue a *different* goal. We aim to provide *textual explanations* by answering a user's query on the optimal solution. We do so by computing the IIS of a specially-defined CSP and identifying the *reasons* that constitute such an explanation, which are then represented as a graph, and then used to build textual explanations with *Large Language Model*s (LLMs).

Chapter 3

Aggregating Value Systems

3.1 Introduction

As we motivated in Chapter 1, the vision of human-centred Artificial Intelligence (AI) has spurred research on trustworthy, ethical AI that enhances human capabilities and empowers citizens and society to deal with the globalised world's challenges effectively. Thus, developing trustworthy AI [38] that abides by human values is a primary AI concern, as explicitly stated by the European Commission's Ethics Guidelines for Trustworthy AI [47], the Artificial Intelligence Act [48], and the IEEE's Global Initiative on Ethics of Autonomous and Intelligent Systems [63]. Within this vision, a core problem is AI value alignment, which aims to ensure that AI is properly aligned with human values [163]. Designing an AI agent to align with human values means that the agent does "what it morally ought to do, as defined by the individual or society"[1] [72].

The problem of value alignment has spurred research on different aspects of the challenge such as formalising the value alignment problem [177], identifying relevant values [121], value-sensitive design [197, 71, 139], learning value-aligned behaviours [157, 1, 201, 138, 14, 158, 159], reasoning about values to act ethically [5, 186, 185], and aligning norms with values [171, 131, 132]. A core component of the different value alignment endeavours is the concept of a *value system*, which models how an entity (e.g., an individual or an organisation) interprets and prioritises values. A

[1] Therefore, hereafter we shall refer to human values as *moral values*, or simply as *values* for shorter when there is no risk to incur in confusion.

common assumption in most state-of-the-art research on value alignment is that an AI system must align with *one* value system, be it that of an individual or a society. However, as Gabriel [72] argues, by following Rawls, humans may hold various reasonable but contrasting beliefs about values. That is, we live in a pluralistic world where people hold different value systems. Designing an AI system that aligns with a group of people with different value systems poses the *pluralistic value alignment* problem [72]. This is the case, for instance, when making policy decisions that align with stakeholders having a variety of value systems (e.g., when deciding over water governance [147, 146]) or when designing human-agent teams involving humans with differing ethical perspectives (the most apparent examples exist in the medical field, where teams are constantly tasked with scenarios that require ethical consideration) [68].

Overall, in this chapter, we address the problem of determining the value system(s) that an AI system should align with (considering value diversity). More precisely, we make headway in the pluralistic value alignment problem by addressing the aggregation of different value systems to yield a *consensus* value system. We assume that the value systems to aggregate belong to any form of individual entities, be they citizens, stakeholders, teammates, countries, or even supra-national entities. To succeed in this endeavour, we identify three major challenges, which are directly related to the research questions Q1, Q2, and Q3 that we address in this chapter.

First, as noted by Mittelstadt [129], existing ethical codes are rather abstract about moral values, hence not specific enough to guide actions. This is also typically the case in the AI literature on value systems (e.g., [131, 172, 123, 18]). Therefore, it is necessary to precisely identify the key elements that a value system must include. Importantly, this effort must consider that: (i) value systems are *contextual* [121, 94, 26, 106], i.e., the way in which we reason about and prioritise values is influenced by contextual factors such as actors and actions [195, 192, 169]; and (ii) individuals may ascribe different *interpretations* to the same value system. Second, from a social choice perspective, value systems can be aggregated following different *ethical principles* (e.g., utilitarian or egalitarian). Therefore, we need a principled and general aggregation method that allows us to set the ethical principle of choice. To this

end, we formalise the aggregation problem as the minimisation of a parameterised distance function. Depending on the parameter value, the aggregation yields a utilitarian consensus, an egalitarian consensus, or any point along the spectrum between the two. Finally, by analysing the impact of the ethical principle on the resulting consensus value system, we provide guidelines for decision-makers concerned with obtaining a consensus on individual value systems in selecting the ethical principle most appropriate for their goals.

We approach the challenge of pluralistic value alignment by studying two real-world cases. On the one hand, we consider the data from a Participatory Value Evaluation (PVE) [133] process conducted in 2020 to gauge the opinion of the residents of a municipality in the Netherlands over energy policies [99]. From this data, we estimate the value system of each citizen who participated in the PVE. On the other hand, we consider data from the European Values Study (EVS) [65], an extensive survey conducted on European citizens to characterise the value systems of European countries.

We build on preliminary works on value estimation [175] to make the following novel contributions:

- We extend the definition of value system to account for *contextuality* and *pluralism*. We show how the novel definition can be employed to represent individual value systems in the two case studies that we analyse.
- Based on the social choice (distance) functions from González-Pachón and Romero [82], we formalise the problem of aggregating different value systems following a given ethical principle (e.g., utilitarian or egalitarian), and we cast it as a two-step optimisation problem to obtain: (i) the aggregation of value interpretations from individuals, and (ii) the aggregation of the preferences of individuals over moral values.
- We show that the problem of computing the consensus value system can be cast as an *ℓ_p-regression problem* [3] (also called *norm approximation problem* [24]). By doing so, we provide a general theoretical framework that allows us to solve the above-mentioned problem for a range of ethical principles—from

utilitarian (maximum utility) to egalitarian (maximum fairness)—in a scalable and reliable way, thanks to recent results in the machine learning literature [3].

- We illustrate our value aggregation approach with real-world data from the PVE [99] and the EVS [65]. We conduct a thorough empirical evaluation that: (i) shows the impact of choosing different ethical principles on the resulting consensus value system; (ii) characterises the space of ethical principles to determine whether a given ethical principle produces a consensus leaning towards the utilitarian or the egalitarian ethical principle; and (iii) quantifies the price to pay when moving away from the majority to include the minority. The aggregation results differ in the two case studies since they handle different data. However, we delineate the common trends we observe. Finally, we provide practical insights for decision-makers concerned with obtaining a consensus on different value systems according to an ethical principle of choice.

In summary, we provide both the computational means and the guidelines for a decision-maker to conduct the principled aggregation of value systems.

Organisation Section 3.2 provides background on the distance functions that we require from the social choice literature. Section 3.3 identifies the key elements of a value system and introduces the value systems in our two case studies. Section 3.4 formalises our aggregation problem and Section 3.5 shows that it can be cast as an ℓ_p-regression problem that can be solved as described in Subsection 3.5.1. Section 3.6 reports on our empirical findings and provides guidelines for the decision-makers in charge of the value system aggregation. Finally, Section 3.7 draws conclusions and sets paths to future research.

3.2 Preliminaries: Distance Functions

As mentioned above, the main goal of this chapter is to aggregate the values and preferences on values (i.e., the value systems) of different individuals to yield a *consensus* value system. The basic problem in designing socially optimal decisions is aggregating individual preferences on multiple alternatives into a collective pref-

erence representing a consensus. This problem is the core of disciplines such as social choice (e.g., [25, 40]), multi-expert decision-making (e.g., [189, 93]), or group decision-making (e.g., [33, 205, 140, 202]). Therefore, many procedures for undertaking this aggregation task have been proposed in the literature. However, methods for aggregating preferences holding "good" theoretical properties are needed to facilitate the acceptance of the resulting consensus by the group of individuals involved in the decision [82]. Designing aggregation functions that exhibit good *social choice* properties (e.g., unanimity, anonymity, non-dictatorship) is a major goal in the social choice literature [25]. For this reason, to tackle the aggregation problem that we pose in this chapter, we resort to the tools in the social choice literature.

This section provides a background on the social choice functions that we employ. We borrow from existing work [78, 79, 80, 82], which define a generator of social choice functions (as a p-parameterised distance function) to obtain a consensus in a society of individuals. The choice of this generator of social choice functions is motivated by several reasons. First, it is well-founded on the social choice literature and multi-criteria decision-making literature, following earlier work by Cook [52, 53, 50, 51] and Yu's p-metric distance [208, 209] respectively. Second, the literature (e.g., [78, 81]) has already studied the social choice properties underlying the compromise consensuses derived from the selected social choice function (e.g., neutrality, monotonicity, anonymity). Third, the work in [79] shows that our social choice function allows us to solve a wide range of aggregation situations (involving both ordinal and cardinal preferences of individuals). Finally, the works in [78, 82] offer interpretations of the consensus that the social choice function produces. This includes a study (limited to the cases of $p = 1$ and $p = \infty$) of the *ethical* interpretation of the resulting consensus within a context of social choice. This means that our generator produces social functions that vary depending on ethical principles (e.g., egalitarian, utilitarian, equity)[2]. Such ethical interpretations provide us with foundations to analyse consensuses. Thus, one of the major benefits of our generator of social choice functions is that it allows decision-makers to compute a consensus fol-

[2]In this chapter, we do not consider the consensus computed considering the principle of equity (i.e., the so-called Marxian solution) since, as noted by the authors of [82], it often results in an over-constrained optimisation problem that yields no solution.

lowing different *ethical principles* according to different values of the parameterised distance function.

The general setting of González-Pachón and Romero [78] considers a society of n members ($i = 1, \ldots, n$). Each member of the society provides judgement on m objects ($j = 1, \ldots, m$), which can be candidates, criteria, alternatives, etc. In the case of aggregating value systems, the objects represent moral values. As argued in Section 3.3, rankings are typically used to describe value systems. Thus, we consider that each member of the society ranks their moral value preferences. Furthermore:

- w_i is the weight (social influence) of the i-th member.
- $R_i[j]$ is the rank position provided by the i-th member of the society for the j-th object (in our case, with the j-th value) within the ranking.
- $R_S[j]$ is the *consensus* position assigned by the society as a whole to the j-th object (in our case, with the j-th value). R_S is the unknown consensus ranking that we seek to obtain.
- p is a metric parameter (i.e., an integer ≥ 1) that determines the *ethical principle* used to compute the consensus, in accordance with the terminology established in the social choice literature [78, 79, 80, 82].

From the previous definitions, a generator of social choice functions based on the weighted Minkowski p-metric distance function (U_p) is introduced and described in several works [78, 79, 80, 82] as:

$$U_p = \left[\sum_{i=1}^{n} \sum_{j=1}^{m} w_i |R_i[j] - R_S[j]|^p \right]^{1/p}. \tag{3.1}$$

Given the distance function U_p and a value of p, the goal is to find the *consensus* values of $R_S[j]$ that minimise the deviation between the judgements provided by the members of the society (data of the problem) and the consensus (the unknown).

In addition, González-Pachón and Romero [82] modify (3.1) to aggregate more com-

plex objects besides rankings. In this case, they propose a distance function to aggregate a 2-dimensional vector such that

$$U_p = \left[\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1,k\neq j}^{m} w_i|R_i[j,k] - R_S[j,k]|^p\right]^{1/p}, \tag{3.2}$$

where $R_i[j,k]$ is the judgement value provided by the i-th member of the society when comparing the j-th and the k-th object. It is relevant that González-Pachón and Romero [79] shows how the social choice function in Equation 3.2 can handle a wide range of aggregation situations. This includes aggregations when information about preferences is: ordinal and complete (complete rankings); ordinal and partial (partial rankings); and cardinal and complete (pairwise comparisons). Thus, when using Boolean values in the preference (R) matrix, we can express ordinal and qualitative preferences, whereas we can also express graded quantitative preferences through cardinal values. Furthermore, as argued in [82], the information in the preference matrix can be complete or incomplete.

From the U_p distance function, González-Pachón and Romero [82] derive two cases of interest. First, by setting $p = 1$, the general distance in (3.1) yields

$$U_B = \left[\sum_{i=1}^{n}\sum_{j=1}^{m} w_i|R_i[j] - R_S[j]|\right]. \tag{3.3}$$

The consensus that minimises U_B provides the social optimum from the point of view of the majority, i.e., the *utilitarian* solution (or Benthamite solution [19]) that maximises the total welfare.

By setting $p = \infty$, the distance function in (3.1) yields

$$U_R = \max_{i,j}\left[|R_i[j] - R_S[j]|\right]. \tag{3.4}$$

(3.4) represents the Chebyshev distance, which is equivalent to the weighted Minkowski distance for $p = \infty$. In this case, finding the consensus implies minimising the disagreement of the member of the society most displaced with respect to the majority solution defined by the utilitarian case above (3.3). This solution is *egalitarian* [39] since it represents the social optimum from the point of view of the minority (from

the perspective of the worst-off member of the society according to Rawls' principle [153]), leading to the point of maximum fairness. Note that, when considering the limit case $p \to \infty$, the weighting scheme in (3.1) vanishes, hence the weight w_i does not appear in (3.4).

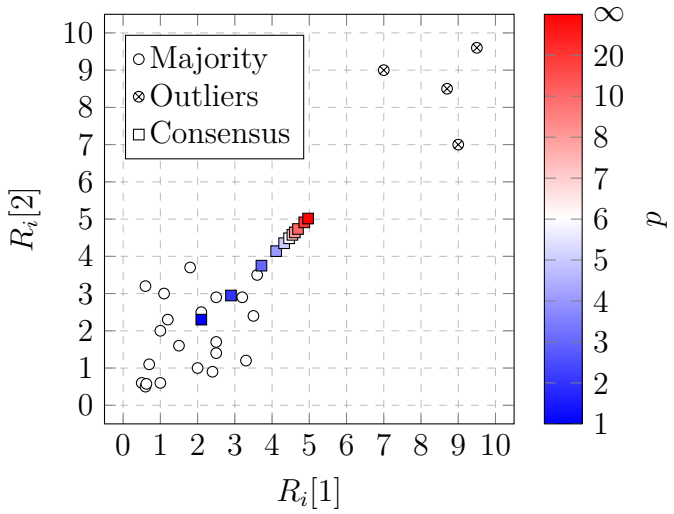


Figure 3.1: The impact of the ethical principle p on the computed consensus of a set of individuals' judgement of two objects: $R_i[1]$ and $R_i[2]$. Circles show the individuals' judgement and squares represent the consensus computed. Squares are filled with the colour of the ethical principle used to compute the consensus. (Best viewed in colour).

In addition to the utilitarian and egalitarian cases, we can use $p \in [2..\infty)$ for computing different consensus. To illustrate the semantics of the ethical principle p and its impact on the consensus, we show a test case (with fabricated data) in Figure 3.1, which plots the judgements of 25 individuals on two objects. The circles represent the individuals' judgements $R_i[1]$ and $R_i[2]$ about objects 1 and 2 within the x and y axis, respectively.

We clearly distinguish two groups of individuals: (1) a clustered set of individuals that represent the majority with values for x and y smaller than 4, and (2) a few individuals representing outliers distant from the majority (each shown in a crossed circle). In addition, the squares represent the position of the computed judgement consensus with different ethical principles, p's, whose values are represented with a colour scale from blue ($p = 1$) to red ($p = \infty$). As we can observe, the utilitarian consensus ($p = 1$) is at the centre of the majority. As we increase p, the consensus

moves towards the outliers, converging to the egalitarian solution ($p = \infty$), which reduces the distance of the consensus to the worst-off member of the society.

In Section 3.4, we employ the general distance function in (3.6) to pose our problem of aggregating value systems as that of computing a consensus. By leveraging this general distance function, Section 3.6 analyses how the ethical principle determined by p (including the utilitarian and egalitarian principles) affects the consensus in particular case studies.

3.3 Value Systems: From Theory to Practice

Moral values are the principles deemed valuable by society [148] and involved in ethical choices [54]. Our preference over relevant, competing values guides our decision-making process [18]. To this extent, *value systems* are the structures employed to represent our moral values and associated preferences [172, 123, 18]. Rankings are typically used to describe value preferences because they are the least restrictive preference structure satisfying totality. However, recent works [121] contend that representing value systems solely as rankings is incomplete for concrete applications.

Computer scientists [121, 106] and social scientists [94, 26, 195] argue that value systems are *contextual*. That is, the way in which we reason about and prioritise values is influenced by the context we are in. For instance, one may value freedom over safety but prioritise safety over freedom during a global pandemic [121]. Furthermore, considering values is challenging for humans, as it requires significant cognitive effort [113, 149]. Thus, allowing humans to express their preferences over value-laden contextual alternatives (as opposed to competing values) is both easier and more insightful [70].

Along the lines of Gabriel [72], we consider value systems as *pluralistic*. As argued by Gabriel [72], humans hold various reasonable but contrasting beliefs about values. Therefore, we live in a pluralistic world where individuals ascribe to different value systems. Thus, individuals from different cultural backgrounds might judge the same action differently regarding the same moral value, which we describe as having

different *value interpretations*. For instance, if we consider the moral value of *respect* in the context of a funeral, Western cultures consider wearing black as promoting the value (and failing to do it as demoting it), whereas Asian cultures favour a white dress code.

Considering the *contextuality* and *pluralism* of value systems, we identify four key elements that a value system must include:

1. a set of **values** relevant to the context under discussion;
2. a set of contextual **alternatives** (e.g., actions, policies) over which an individual ought to take value-laden choices;
3. a **relationship between alternatives and values** that captures value interpretation by specifying the degree to which an individual deems an alternative as *promoting* or *demoting* a value; and
4. a set of individual **preferences over values**.

On the one hand, the presence of alternatives and the relationship between alternatives and values delimit a context for situated value reasoning. On the other hand, the individual interpretation of the relationship between alternatives and values and consequent individual value preferences reflects the pluralism of values. To this extent, Chisholm [45] links these two aspects (i.e., alternatives or actions and values) by arguing that individuals may judge an action as either good or bad to perform (or not to perform) depending on the value under consideration.

Our goal is to aggregate value systems to obtain a consensus value system. To this end, we introduce two case studies aimed at estimating value systems (Subsections 3.3.1 and 3.3.2). In each case study, we formally define the corresponding value system and show how individual value systems are obtained. Subsequently, in Section 3.6, we compute the consensus value system by aggregating both value interpretations and value preferences.

3.3.1 Participatory Value Evaluation

Participatory Value Evaluation (PVE) [133] is a digital framework for eliciting citizens' preferences over governmental policy options. We use data from a PVE conducted in 2020 to gauge the opinion of the residents of a municipality in the Netherlands over energy policies [99].

The main question asked in this PVE was: *what do you find important for future decisions on energy policy?* As potential answers, six *policy options* were developed by policymakers in collaboration with a panel of 45 citizens. Each PVE participant was asked to distribute 100 points among the six options and then to motivate each option to which they had assigned points with a textual *motivation*. Table 3.1 shows the six policy options and the mean points allocated to each option by the PVE participants.

Table 3.1: Policy options in the Súdwest-Fryslân PVE

Policy option	Description	Mean points
o_1	The municipality takes the lead and unburdens you	29.05
o_2	Inhabitants do it themselves	21.72
o_3	The market determines what is coming	9.39
o_4	Large-scale energy generation will occur in a small number of places	15.01
o_5	Betting on storage (Súdwest-Fryslân becomes the battery of the Netherlands)	12.96
o_6	Become a major energy supplier in the Netherlands	4.71

The mean points allocated to each option in a PVE indicate *what* options the PVE participants prefer. However, these points do not indicate *why* the participants have that preference. To answer the why question, Siebert, Liscio, Murukannaiah, Kaptein, Spruit, Hoven and Jonker [175] estimate the value system underlying each participant's preferences.

In Subsection 3.3.1.1, we describe how the previously introduced value system's four key components connect to this case study and provide a formal definition of a value system. In Subsection 3.3.1.2, we introduce the method that Siebert, Liscio, Murukannaiah, Kaptein, Spruit, Hoven and Jonker [175] propose for estimating an individual's value system from their PVE data.

3.3.1.1 Definition of Value Systems

We adapt the definition of value system that Siebert, Liscio, Murukannaiah, Kaptein, Spruit, Hoven and Jonker [175] propose to incorporate the four key components introduced earlier.

1. The *value list* V is the set of values relevant to the discussion.

2. The *alternatives* over which participants make value-laden choices are the policy options presented in the PVE (Table 3.1): $O = \{o_1, \ldots, o_6\}$.

3. The *relationship between alternatives and values* captures value interpretation by means of a binary individual value-option matrix VO_i with $|V|$ (number of values) rows and $|O|$ (number of options) columns, where:

$$VO_i(v, o) = \begin{cases} 1, & \text{if value } v \text{ is relevant for an individual } i \text{ for option } o \\ 0, & \text{otherwise.} \end{cases}$$

   In Subsection 3.3.1.2, we explain how an individual's VO_i is calculated based on their answers to the survey.

4. The *value preferences* are an individual's ranking R_i of V, which is a reflexive, transitive, and total binary relation, noted as $v_a \succeq v_b$. Given $v_a, v_b \in V$, if $v_a \succeq v_b$, we say v_a is more preferred than v_b. If $v_a \succeq v_b$ and $v_b \succeq v_a$, then we note it as $v_a \sim v_b$ and consider v_a and v_b indifferently preferred. However, if $v_a \succeq v_b$ but it is not true that $v_b \succeq v_a$ (i.e., $v_a \neq v_b$), then we note it as $v_a \succ v_b$.

Considering the four aspects, we define a value system in a PVE as follows.

Definition 3.3.1. *A value system in a Participatory Value Evaluation is a tuple* $\mathcal{V}_i = \langle V, O, VO_i, R_i \rangle$*, where* V *is a non-empty set of values,* O *is a set of available policy options,* VO_i *is a matrix describing the relevance of a value for an individual for a given option, and* R_i *is the ranking of* V *which represents an individual's value preference.*

3.3.1.2 Value System Estimation

The designers of the energy policy PVE identify the relevant values by analysing the participants' motivations for policy choices using a grounded theory approach. The five most commonly mentioned values they identified are: cost-effectiveness (v_1), nature and landscape preservation (v_2), leadership (v_3), cooperation (v_4), and self-determination (v_5). However, these five values may not be relevant to each of the six policy options in Table 3.1.

Siebert *et al.*[175] propose computational methods for estimating a participant's value system based on the choices and motivations they provide in the PVE. They compute an initial VO matrix, shown in Table 3.2a, as the first guess of value preferences based on the available choices in the PVE. It is intended to be the starting point for estimating the individual participants' value-option matrices (VO_i) and value rankings (R_i).

Table 3.2: VO matrices representing the relationship between alternatives and values

(a) Initial VO for the energy transition PVE

Values \ Options	o_1	o_2	o_3	o_4	o_5	o_6
v_1	1	1	1	1	1	1
v_2	1	1	0	1	1	1
v_3	1	1	1	0	0	0
v_4	1	1	1	1	0	1
v_5	1	1	0	0	1	0

(b) Example of estimated individual VO_i

Values \ Options	o_1	o_2	o_3	o_4	o_5	o_6
v_1	1	1	1	1	0	1
v_2	1	1	0	1	1	0
v_3	1	1	1	0	0	0
v_4	1	1	1	0	1	1
v_5	1	1	0	0	1	0

Siebert, Liscio, Murukannaiah, Kaptein, Spruit, Hoven and Jonker [175] propose four methods for estimating participants' value systems. Their approach is based on the theory that "valuing is deliberatively consequential" [168]. That is, if a participant's choice is based on a deliberation of value preferences, the value preferences can be recognised in the motivation provided for the choice. The best-performing approach is the sequential combination of three proposed methods (MO, MC, and TB):

- MO addresses inconsistencies between motivations provided for different choices. For example, consider that an individual selected options o_4 and o_6, for which v_2 and v_4 are deemed relevant. Further, assume that they motivated o_4 with value v_2 and o_6 with value v_4. Following the notion of valuing as deliberatively

consequential, from the two motivations, one can both infer that $v_2 \succ v_4$ and that $v_4 \succ v_2$. Thus, the method *MO* updates VO_i by considering irrelevant the value that is part of the inconsistency but was not mentioned in the motivation (in practice, by setting the cells (v_4, o_4) and (v_2, o_6) to 0, as in Table 3.2b).

- *MC* targets inconsistencies between choices and motivations. Assume that a participant allocates some points to option o_5 where, according to *VO*, v_1 is relevant, but v_4 is not, but mentions v_4 in the motivation. Then, *MC* adjusts VO_i to prefer v_4 over v_1 for o_5 (in practice, by setting the cell (v_1, o_5) to 0 and the cell (v_4, o_5) to 1, as in Table 3.2b).

- *TB* reduces the number of indifferent preferences in an individual's value ranking. First, it computes the importance of values for an individual by weighing the values supported in VO_i with the points that the participant assigned to the options. For instance, consider that individual i distributes their points to the six policy options as: $\{30, 40, 10, 20, 0, 0\}$. The multiplication of VO_i by the vector of this policy scores results in a vector of importance scores for values in V: $\{100, 90, 80, 80, 70\}$. Then, the individual's value ranking R_i is inferred by ordering the values in V according to their importance score: $v_1 \succ v_2 \succ v_3 \sim v_4 \succ v_5$, where $v_a \sim v_b$ indicates that there is a tie, i.e., the participant has no preference between v_a and v_b. However, if one of the motivations provided by the participant mentions v_4 but no motivations mention v_3, then the *TB* method breaks the tie by setting $v_4 \succ v_3$, thus resulting in $R_i : v_1 \succ v_2 \succ v_4 \succ v_3 \succ v_5$.

3.3.2 European Value Study

The European Values Study (EVS) [65] is a large-scale survey research program on European values. It collaborates with the World Values Survey [200]. The program provides data about the variety of positions that citizens from different European countries have on basic values such as well-being, solidarity, and democracy. Although the EVS survey covers a wide range of questions and values, here we focus on two values and three questions for 34 European countries. Our goal is not to

comprehensively analyse European values but to illustrate our aggregation approach with a simplified example.

3.3.2.1 Definition of Value Systems

Similarly to the case of the PVE, we identify the four key elements of the value system $\mathcal{V}$ of the EVS.

1. $V = \{v_1, \ldots, v_n\}$ is the set of values relevant to European citizens.

2. The alternatives are a set of actions, $A = \{a, \ldots, a_m\}$, which give information about the value interpretation of citizens.

3. We relate values with actions to interpret each value $v \in V$ for every country i. To do so, for each value $v \in V$ and the actions in A, we define the action judgement function $\alpha_v^i : A \to [-1, 1]$ for country i as the function that evaluates the promotion or demotion of value v when performing action $a \in A$. These evaluations are real numbers in the interval $[-1, 1]$: a positive number indicates the degree to which a value is being promoted, whereas a negative one indicates demotion. For instance, in the "funeral dress code" example from Section 3.3, a western-raised individual would consider wearing black clothes (wbc) as an action that promotes the value respect ($\alpha^i_{respect}(wbc) > 0$), and wearing colourful clothes (wcc) is an action that demotes the value respect ($\alpha^i_{respect}(wcc) < 0$). We represent such evaluations in a value-action matrix such that

$$VA_i = \begin{bmatrix} \alpha^i_{v_1}(a_1) & \cdots & \alpha^i_{v_1}(a_m) \\ \vdots & \ddots & \vdots \\ \alpha^i_{v_n}(a_1) & \cdots & \alpha^i_{v_n}(a_m) \end{bmatrix}, \tag{3.5}$$

   where each row corresponds to the action judgement functions of a value.

4. We define a country i's value preferences via a preference matrix $P_i \in [0, 1]^{n \times n}$, where $P_i[v_j, v_k] \in [0, 1]$ represents i's graded preference when comparing v_j and v_k. The value preferences in P_i are *pairwise comparisons*, where 0.5 stands for indifference. For instance, individuals from a country may prefer environmental protection over economic development with a grade of 0.75.

Given these elements, we formally define the value system as follows.

Definition 3.3.2. *Given a set of values V and a set of actions A, a value system for the European Values Study is a tuple $\mathcal{V}_i = \langle V, A, VA_i, P_i \rangle$ for individual i , where VA_i is the value-action matrix containing the evaluation of values with relation to actions, and P_i is the preference matrix containing the preference pairwise comparisons between values.*

3.3.2.2 Value System Estimation

We resort to the EVS data [183] to create the value system of each country. We consider two values: *religiosity* (rl) and *permissiveness* (pr).[3] We characterise these two values in terms of their action judgement functions. For simplicity, we consider the judgement of two actions: *homosexual couples' parenthood* (ho) and *divorcing* (dv). To characterise Europeans' position on religiosity, we consider the EVS question "Q1F: How important is religion in your life?" and partition possible answers so that we can discern the percentage of citizens who consider religion important from the ones who do not. Columns 2 and 3 of Table 3.3 show the respective percentages per country, which we also interpret as the degree of preference of each value. For conciseness, Table 3.3 lists 12 out of the 34 countries considered in our computation. Formally, we denote the preference degree of value rl over pr in country i as $P_i[rl, pr]$. Conversely, we denote as $P_i[pr, rl]$ the preference degree of pr over rl. As a consequence, we assume that those countries in which religion is important for the majority of the population (i.e., $P_i[rl, pr] > P_i[pr, rl]$) will prefer religiosity over permissiveness, whereas we consider that permissiveness is preferred over religiosity if $P_i[rl, pr] < P_i[pr, rl]$.

Next, we employ two additional EVS questions to characterise the value judgement functions of the values under consideration: "Q27A: How much do you agree or disagree with the statement: Homosexual couples are as good parents as other

[3]Although the values of *religiosity* [130] and *permissiveness* [105] can be related to the values of *tradition* and *tolerance* from the Schwartz's revised model of values [170], we choose them to fit EVS's data better. In fact, one may even think that secularism seems a better alternative to permissiveness when comparing it to religiosity. However, we argue that permissiveness [105] is better suited, as it is specifically related to sexual freedom [196], and the data from EVS we use relates to homosexual couples and divorce.

Table 3.3: Value preferences (columns 2 and 3) and value judgement functions (columns 4–7) for the countries in the EVS.

Country	$P_i[rl, pr]$	$P_i[pr, rl]$	$\alpha^i_{rl}(ho)$	$\alpha^i_{rl}(dv)$	$\alpha^i_{pr}(ho)$	$\alpha^i_{pr}(dv)$
AL	0.72	0.28	−0.64	−0.33	−0.58	−0.18
AM	0.83	0.17	−0.73	−0.48	−0.71	−0.45
AT	0.44	0.56	0.06	0.19	0.32	0.54
AZ	0.60	0.40	−0.83	−0.32	−0.85	−0.13
BA	0.81	0.19	−0.46	−0.09	−0.31	0.15
BG	0.61	0.39	−0.42	−0.05	−0.32	0.16
BY	0.56	0.44	−0.67	−0.04	−0.52	0.20
CH	0.32	0.68	0.01	0.20	0.30	0.56
CZ	0.22	0.78	−0.09	0.01	0.12	0.32
DE	0.36	0.64	0.20	0.25	0.42	0.59
DK	0.24	0.76	0.25	0.53	0.46	0.75
EE	0.26	0.74	−0.52	−0.07	−0.23	0.24
ES	0.40	0.60	0.15	0.20	0.44	0.50
FI	0.35	0.65	0.02	0.30	0.32	0.62
FR	0.36	0.64	0.29	0.26	0.51	0.54
GB	0.38	0.62	0.20	0.20	0.39	0.50
GE	0.94	0.06	−0.71	−0.50	−0.60	−0.32
HR	0.61	0.39	−0.41	−0.17	−0.02	0.28
HU	0.46	0.54	−0.27	0.00	−0.10	0.25
IS	0.40	0.60	0.54	0.41	0.70	0.71
IT	0.66	0.34	−0.15	0.12	0.05	0.51
LT	0.48	0.52	−0.56	0.03	−0.50	0.21
ME	0.87	0.13	−0.45	−0.34	−0.26	−0.15
MK	0.81	0.19	−0.52	−0.09	−0.30	0.27
NL	0.32	0.68	0.26	0.19	0.52	0.63
NO	0.35	0.65	0.33	0.38	0.64	0.68
PL	0.80	0.20	−0.49	−0.17	−0.20	0.34
PT	0.67	0.33	0.11	0.02	0.19	0.35
RO	0.82	0.18	−0.62	−0.25	−0.29	0.00
RS	0.75	0.25	−0.62	−0.09	−0.51	0.21
RU	0.51	0.49	−0.59	0.08	−0.56	0.20
SE	0.31	0.69	0.44	0.56	0.61	0.76
SI	0.36	0.64	−0.22	0.16	0.10	0.53
SK	0.58	0.42	−0.53	−0.05	−0.31	0.31

couples?" and "Q44G: Can divorce be always justified, never justified, or something in between?". By correlating these answers with those about religion, we obtain the judgements of religious citizens of each country on homosexual parenthood and divorce (columns 4–5 of Table 3.3). Similarly, we obtain the judgements of non-religious people (columns 6–7 of Table 3.3).

3.4 Formalising the Aggregation of Value Systems

After formally describing a value system $\mathcal{V}$ and two real-world examples, we proceed to the problem of aggregating value systems, $\mathcal{V}_1, \ldots, \mathcal{V}_N$, of individuals $i = 1, \ldots, N$. Specifically, we compute a value system $\mathcal{V}_S$ that *best* represents the aggregation of $\mathcal{V}_1, \ldots, \mathcal{V}_N$ according to an *ethical principle* p.[4]

Recall that a value system has four key components. We assume that the first two components, the set of values and the set of alternatives, are the same across all individuals. So it is not necessary to aggregate them. Then, the problem of aggregating multiple value systems boils down to aggregating the last two components, the relationship between values and alternatives and the value preferences, which are specific to the individuals.

We pose the overall aggregation problem as a two-step procedure to compute: (1) a *consensus* values-alternatives relation, and (2) a *consensus* of preferences over values. We represent the objects for aggregation (i.e., the individual values-alternatives relations and the individual preferences over values) as q-dimensional vectors. This representation provides sufficient expressiveness since it generalises 1-dimensional vectors and matrices, which are commonly used in the literature [78, 79, 80, 82] and in the test cases described in Subsection 3.3.1 and Subsection 3.3.2 for these purposes.

Considering a unique general representation for individual objects allows us to pose the aggregation problem in a unique general way. Following the social choice literature (Section 3.2), we cast this problem as the minimisation of a distance measure defined along the lines of Equations 3.1 and 3.2 (for 1 and 2-dimensional cases, respectively). To accommodate a q-dimensional representation, we generalise these distance functions as:

$$U_p = \left[\sum_{i=1}^{n} \sum_{j_1=1}^{J_1} \cdots \sum_{j_q=1}^{J_q} w_i |T_i[j_1, \ldots, j_q] - T_S[j_1, \ldots, j_q]|^p\right]^{1/p}, \tag{3.6}$$

[4] We refer to p as the ethical principle used to compute the aggregation, in accordance with the social choice literature (Section 3.2). This should not be confused with the objects of our aggregation, i.e., the value systems.

where $T_i[j_1, \ldots, j_q]$ is the judgement or preference value provided by the i-th member of the society to the particular combination over q features[5]. Notice that the generalisation that we propose in Equation 3.6 does not assume any underlying vectorial structure. We propose such generalisation to aggregate vectors, matrices, $q-$dimensional vectors, and even scalars when considered as one-component vectors.

Following Definition 3.3.1, (3.6) can be readily particularised for the PVE case study to define the distance between the individual value-option matrices VO_i and the aggregated value-option matrix VO_S as

$$U_p^{(VO)} = \left[\sum_{i=1}^{N} w_i \sum_{j=1}^{|V|} \sum_{k=1}^{|O|} \left|VO_i[v,o] - VO_S[v,o]\right|^p\right]^{1/p}. \tag{3.7}$$

From (3.6), we can also derive a definition for the distance between the individual rankings R_i and the aggregated ranking R_S as

$$U_p^{(R)} = \left[\sum_{i=1}^{N} w_i \sum_{j=1}^{|V|} \left|R_i[j] - R_S[j]\right|^p\right]^{1/p}. \tag{3.8}$$

Definitions for the EVS case can be derived similarly. We do not report these definitions for the sake of conciseness.

For a general value system defined in Section 3.3, we denote the distance function referring to the relationships between values and alternatives, and the value preferences as $U_p^{(3)}$ and $U_p^{(4)}$, respectively. Then, we can pose *value system aggregation* as a two-step problem aiming to compute

$$T_S^{(3)} = \arg\min U_p^{(3)}, \tag{3.9}$$

$$T_S^{(4)} = \arg\min U_p^{(4)}, \tag{3.10}$$

where $T_S^{(3)}$ and $T_S^{(4)}$ denote the consensus among values-alternatives relationships and value preferences, respectively.

[5]The judgements or preferences of the individuals $T_i[j_1, \ldots, j_q]$ can be represented with binary, integer or real numbers, depending on the case study domain.

3.5 An ℓ_p-regression Approach to Aggregate Value Systems

We show how Equations 3.9 and 3.10 can be cast as ℓ_p-regression (also known as *norm approximation* [24]) problems. Such a transformation yields obvious computational benefits as it allows us to efficiently solve the above-mentioned optimisation problems for any p, as explained in Subsection 3.5.1.

Since solving (3.9) and (3.10) can be seen as the minimisation of the same general q-dimensional distance function (i.e., (3.6)), in Theorem 3.5.1 we directly consider this problem.

Theorem 3.5.1. *Computing the solution* $T_S = \arg\min U_p^{(T)}$ *is equivalent to computing the solution* x *of the* ℓ_p*-regression problem*

$$\textit{minimise } \|A\mathbf{x} - \mathbf{b}\|_p, \tag{3.11}$$

where $A \in \mathbb{R}^{N \cdot J_1 \cdots J_q \times J_1 \cdots J_q}$ *and* $\mathbf{b} \in \mathbb{R}^{N \cdot J_1 \cdots J_q}$ *are*

$$A = \begin{bmatrix} w_1^{1/p} \cdot I \\ \vdots \\ w_N^{1/p} \cdot I \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} w_1^{1/p} \cdot \overrightarrow{T_1} \\ \vdots \\ w_N^{1/p} \cdot \overrightarrow{T_N} \end{bmatrix},$$

$I \in \mathbb{R}^{J_1 \cdot J_2 \cdots J_q \times J_1 \cdot J_2 \cdots J_q}$ *is the identity matrix of size* $J_1 \cdot J_2 \cdots J_q$, $\overrightarrow{(\cdot)}$ *is the* vectorisation *operation that turns a* $q-$*dimensional vector into a* $1-$*dimensional vector, and the* p*-norm* $\|\mathbf{x}\|_p$ *of a vector* $\mathbf{x}$ *is defined as* $\|\mathbf{x}\|_p = \left(\sum_i |x[i]|^p\right)^{1/p}$.

Proof. As a first step, we rewrite (3.6) as

$$\left[\sum_{i=1}^{N} w_i \sum_{h=1}^{J_1 \cdot J_2 \cdots J_q} \left|T_i[h] - T_S[h]\right|^p\right]^{1/p} \tag{3.12}$$

and, subsequently, as

$$\left[\sum_{i=1}^{N} \left\| w_i^{1/p} \cdot \overrightarrow{T_i} - w_i^{1/p} \cdot \overrightarrow{T_S} \right\|_p^p\right]^{1/p}. \tag{3.13}$$

To express (3.13) as an ℓ_p-regression problem, we define $A \in \mathbb{R}^{N \cdot J_1 \cdot J_2 \cdots J_q \times J_1 \cdot J_2 \cdots J_q}$ and $b \in \mathbb{R}^{N \cdot J_1 \cdot J_2 \cdots J_q}$ as

$$A = \begin{bmatrix} w_1^{1/p} \cdot I \\ \vdots \\ w_N^{1/p} \cdot I \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} w_1^{1/p} \cdot \overrightarrow{T_1} \\ \vdots \\ w_N^{1/p} \cdot \overrightarrow{T_N} \end{bmatrix},$$

We can finally formulate the problem of minimizing (3.6) as

$$\text{minimise } \|A\mathbf{x} - \mathbf{b}\|_p.$$

The solution of the above-defined problem (i.e., the vector $\mathbf{x}$) is T_S. □

3.5.1 Solving the ℓ_p-regression Problem

We now discuss a computational solution to (3.11). This solution applies to aggregating both case studies' value-alternative relations and value preferences, as particular cases of (3.11).

For $p = 1$, (3.11) represents an *absolute residuals approximation* problem, i.e.,

$$\text{minimise } \|A\mathbf{x} - \mathbf{b}\|_1 = \sum_{i=1}^{N \cdot J_1 \cdots J_q} |\mathbf{a}_i \cdot \mathbf{x} - b_i|, \tag{3.14}$$

which can be encoded as an *Linear Programming* (LP) problem [24] as follows:

$$\text{minimise } \mathbf{1}^T\mathbf{t}, \tag{3.15a}$$

$$\text{subject to } -\mathbf{t} \leq A\mathbf{x} - \mathbf{b} \leq \mathbf{t}, \tag{3.15b}$$

where $\mathbf{t} \in \mathbb{R}^{N \cdot J_1 \cdots J_q}$ is a vector of auxiliary variables. For $p = \infty$, we are dealing with a *Chebyshev approximation* (or *Min-Max approximation*) problem, i.e.,

$$\text{minimise } \|A\mathbf{x} - \mathbf{b}\|_\infty = \max_{i=1,\ldots,N \cdot J_1 \cdots J_q} \Big\{ |\mathbf{a}_i \cdot \mathbf{x} - b_i| \Big\}. \tag{3.16}$$

Similarly to $p = 1$, (3.16) can be solved via *Linear Programming* [24] as follows:

$$\text{minimise } t, \tag{3.17a}$$

$$\text{subject to } -t\mathbf{1} \leq A\mathbf{x} - \mathbf{b} \leq t\mathbf{1}, \tag{3.17b}$$

where $t \in \mathbb{R}$ is an auxiliary variable. Contrarily to the $p = 1$, for $p = \infty$, (3.17) can be solved analytically as we demonstrate in Theorem 3.5.2.

Theorem 3.5.2. *The optimal value of the problem in Equation (3.17) is:*

$$t^* = \max_h \left(\frac{\max_i(T_i[h]) - \min_i(T_i[h])}{2} \right), \tag{3.18}$$

and the optimal consensus is

$$T_S^*[h] \in [\max_i(T_i[h]) - t^*, \min_i(T_i[h]) + t^*], \quad \textit{for all } h = 1, \ldots, J_1 \cdot J_2 \cdots J_q. \tag{3.19}$$

Proof. By further developing the linear program (3.17) and replacing $\mathbf{x}$ by T_S, we obtain

$$\begin{aligned} \text{minimise} \quad & t, \\ \text{subject to} \quad & w_i^{1/p} \cdot T_S[h] - w_i^{1/p} \cdot T_i[h] \leq t, \quad \text{for all } i \in [1, N], h \in [1, J_1 \cdots J_q], \\ & w_i^{1/p} \cdot T_S[h] - w_i^{1/p} \cdot T_i[h] \geq -t, \quad \text{for all } i \in [1, N], h \in [1, J_1 \cdots J_q], \end{aligned}$$

which is simplified considering that for $p = \infty$, the weights vanish such that

$$\begin{aligned} \text{minimise} \quad & t, \\ \text{subject to} \quad & T_S[h] - T_i[h] \leq t, \quad \text{for all } i \in [1, N], h \in [1, J_1 \cdots J_q], \\ & T_S[h] - T_i[h] \geq -t, \quad \text{for all } i \in [1, N], h \in [1, J_1 \cdots J_q], \end{aligned}$$

Then, by simply isolating $T_S[h]$, we obtain the following system of inequality constraints.

$$T_S[h] \leq T_i[h] + t, \quad \text{for all } i \in [1, N], h \in [1, J_1 \cdots J_q], \tag{3.22a}$$

$$T_S[h] \geq T_i[h] - t, \quad \text{for all } i \in [1, N], h \in [1, J_1 \cdots J_q]. \tag{3.22b}$$

Notice that for each variable $T_S[h]$, there are N pairs of (two for each individual i), each differing only in the constant term $T_i[h]$. Geometrically, each inequality constraint corresponds to a half-space whose boundary is a hyperplane. For example, in two dimensions, an inequality constraint defines a line that partitions the plane into a feasible and an infeasible region.

In our case, for a given h, the constraints form two families of parallel hyperplanes: their coefficients with respect to the decision variables are identical (all equal to 1), and they differ only in their constant term $T_i[h]$. Since the hyperplanes are parallel, only the "*extreme*" ones, i.e., those with the smallest or largest value of $T_i[h]$, depending on the inequality direction, actually restrict the feasible region.

Concretely, in (3.22a), the restricting constraint is the one with the minimum value of $T_i[h]$, while in (3.22b), it is the one with the maximum value. Therefore, the system of constraints can be reduced by this set of extreme inequalities.

$$T_S[h] \leq \min_{i=1,\ldots,N}(T_i[h]) + t, \quad \text{for all } h \in [1, J_1 \cdots J_q], \tag{3.23a}$$

$$T_S[h] \geq \max_{i=1,\ldots,N}(T_i[h]) - t, \quad \text{for all } h \in [1, J_1 \cdots J_q]. \tag{3.23b}$$

Thus, the optimal solution is restricted to the closed set

$$T_S[h] \in [\max_{i=1,\ldots,N}(T_i[h]) - t^*, \min_{i=1,\ldots,N}(T_i[h]) + t^*], \quad \text{for all } h \in [1, J_1 \cdots J_q]. \tag{3.24}$$

Then, we proceed to isolate t. Hence, by simply reducing the inequality system (3.23), we obtain

$$t \geq \frac{\max_{i=1,\ldots,N}(T_i[h]) - \min_{i=1,\ldots,N}(T_i[h])}{2}, \quad \text{for all } h \in [1, J_1 \cdots J_q]. \tag{3.25}$$

Since the optimisation problem goal is to minimise t, the optimal value t^* will be the minimum feasible value of t. This results in the maximum value of the right-hand side of the inequality (3.25), i.e.,

$$t^* = \max_{h \in [1, J_1 \cdots J_q]} \left(\frac{\max_{i=1,\ldots,N}(T_i[h]) - \min_{i=1,\ldots,N}(T_i[h])}{2} \right) \tag{3.26}$$

□

Furthermore, for $p = 2$, (3.11) can also be solved analytically by treating it as *Least Squares* problem,[6] whose optimal solution is

$$\mathbf{x} = (A^T A)^{-1} A^T \mathbf{b}. \tag{3.27}$$

We employ this analytical solution in Theorem 3.5.3 below, where we show that, for $p = 2$, the aggregation of a general q-dimensional vector can be obtained as the *weighted arithmetic mean* of the individual q-dimensional vectors $(T_1, \ldots, T_N)$.

Theorem 3.5.3. *For $p = 2$, T_S can be analytically computed as the* weighted arithmetic mean *of the individual T q-dimensional vectors $(T_1, \ldots, T_N)$, where the weights are $w_1, \ldots, w_N$.*

Proof. As a first step, we explicitly compute $(A^T A)^{-1}$ as

$$\begin{aligned} (A^T A)^{-1} &= \left(\begin{bmatrix} w_1^{1/2} \cdot I & \cdots & w_N^{1/2} \cdot I \end{bmatrix} \begin{bmatrix} w_1^{1/2} \cdot I \\ \vdots \\ w_N^{1/2} \cdot I \end{bmatrix} \right)^{-1} \\ &= \left(\sum_{i=1}^{N} w_i \cdot I \right)^{-1} = \left(\sum_{i=1}^{N} w_i \right)^{-1} I. \end{aligned} \tag{3.28}$$

Notice that (3.28) is a diagonal matrix whose elements on the diagonal are all equal

[6] The Least Squares problem is obtained by squaring the objective of the original ℓ_2-regression problem. The obtained problem is equivalent to the original one (i.e., it has the same optimal solution), but it has the advantage that it can be solved analytically by expressing the objective as a convex quadratic function [24].

to the inverse of the sum of the weights. By making use of the above result, we explicitly compute x as

$$\mathbf{x} = \left(\sum_{i=1}^{N} w_i\right)^{-1} \overbrace{\begin{bmatrix} w_1^{1/2} \cdot I & \cdots & w_N^{1/2} \cdot I \end{bmatrix}}^{A^T} \begin{bmatrix} w_1^{1/2} \cdot \overrightarrow{T_1} \\ \vdots \\ w_N^{1/2} \cdot \overrightarrow{T_N} \end{bmatrix} \tag{3.29}$$
$$= \frac{\sum_{i=1}^{N} w_i \cdot \left[\overrightarrow{T_i}\right]}{\sum_{i=1}^{N} w_i} = \frac{\sum_{i=1}^{N} w_i \cdot \overrightarrow{T_i}}{\sum_{i=1}^{N} w_i}.$$

Thus, each element of x (i.e., of T_S) is the weighted mean of the corresponding elements of T_i, according to the weights $w_1, \ldots, w_N$. □

For any $p \notin \{1, 2, \infty\}$, (3.6) represents a non-linear problem. Nonetheless, by exploiting the structure of (3.11) as an ℓ_p-regression problem, we can solve it for any p. To do so, we choose the state-of-the-art *Iteratively Reweighted Least Squares* (IRLS) algorithm [3], the only approach for ℓ_p-regression that is guaranteed to converge for any value of p.[7]

3.6 Experimental Results

We empirically investigate our method for computing the consensus value systems for the case studies in Subsections 3.3.1 and 3.3.2. Our analysis shows that our approach is flexible enough to aggregate value preferences encoded differently (i.e., as value rankings in the PVE and pairwise comparisons in the EVS case study). In particular, we aim to:

1. Illustrate how the choice of a given ethical principle (from 1 to ∞) affects the resulting consensus value system;
2. Understand the impact of employing different social weights for the individual value systems on the resulting consensus value system;
3. Analyse the classification of the available ethical principles based on the *prox-*

[7] Our source code: https://github.com/RogerXLera/ValueSystemsAggregation. We use the publicly-available IRLS code [3]: https://github.com/fast-algos/pIRLS.

imity of the consensus they produce with respect to the utilitarian and egalitarian consensuses; and

4. Study at a fine-grained level how the relationship between the distribution of individual value systems and the resulting consensus varies (with respect to the majority and the minority) as the value of the ethical principle increases.

Finally, we distil our empirical analysis to delineate the guidelines a policymaker should follow in choosing an ethical principle when performing the aggregation of ethical principles.

In Subsections 3.6.1 and 3.6.2, we analyse the results for the two case studies, respectively, to address the first two goals. Then, Subsection 3.6.3 tackles the characterisation of the space of ethical principles as pursued by the third goal. Finally, Subsection 3.6.4 investigates the relationship between consensus and individuals as the ethical principle changes (i.e., 4th goal).

3.6.1 Case study: Participatory Value Evaluation

The PVE case study comprises five values, six options, and 795 individual value systems estimated via the method in Subsection 3.3.1.2. In aggregating these value systems, we treat all individuals as equal within society. Thus, all individuals have the same weight in the distance functions employed to compute the consensus among values-alternatives relationships and value preferences in Equations 3.9 and 3.10, i.e., $\forall i \in \{1, 2, \ldots, 795\}, w_i = 1$.

3.6.1.1 Analysing the Consensus on Value Preferences

We show the results of aggregating the rankings over values corresponding to the 795 individual value systems according to different ethical principles.

Encoding individual value preferences First, we build the ranking of values for each individual. For each individual i, we set $R_i[v_j]$ to the position of value v_j in i's ranking. For two values, v_j and v_k, that are equally preferred, we assign the same position within the ranking, i.e., $R_i[v_j] = R_i[v_k]$.

Aggregating individual value preferences Next, we compute the consensus ranking R_S by solving Equation 3.10 for the distance function in Equation 3.8, following the method in Subsection 3.5.1. When computing the consensus, we do not enforce the position of a value j to be an integer. Thus, for instance, a consensus of $R_S[v_j] = 2.5$ indicates that the society considers that value v_j has a position between the second and third position in the consensus ranking. We consider that value v_j is preferred over value v_k ($v_j \succ v_k$) when their positions in the consensus ranking differ such that $R_S[v_k] - R_S[v_j] > \epsilon$, where ϵ is a small positive number ($\epsilon = 0.05$ in our experiments). Otherwise, we say the two values are equally preferred or indifferent ($v_j \sim v_k$).

Table 3.4 shows the consensus rankings resulting from the aggregation of individual rankings for different ethical principles (p): from 1 (utilitarian) to 10, and ∞ (egalitarian). Each column $R_S[v_i]$ indicates the position of value v_i in the ranking computed by our aggregation. Note that we obtained a partial ranking as a consensus ranking for each ethical principle. That is, the order (preferences) between values in each consensus ranking is not strict since it contains ties between values. For instance, in the first row, value v_1 is equally preferred to value v_2 (because $R_S[v_1] = R_S[v_2] = 2$).

Table 3.4: Computed consensus ranking for different ethical principles p.

p	$R_S[v_1]$	$R_S[v_2]$	$R_S[v_3]$	$R_S[v_4]$	$R_S[v_5]$	Consensus ranking
1	2.00	2.00	3.00	3.00	3.00	$v_1 \sim v_2 \succ v_3 \sim v_4 \sim v_5$
2	2.14	2.53	2.90	2.90	2.91	$v_1 \succ v_2 \succ v_3 \sim v_4 \sim v_5$
3	2.31	2.65	2.95	2.96	2.94	$v_1 \succ v_2 \succ v_3 \sim v_4 \sim v_5$
4	2.42	2.73	2.97	2.98	2.96	$v_1 \succ v_2 \succ v_3 \sim v_4 \sim v_5$
5	2.51	2.78	2.98	2.99	2.97	$v_1 \succ v_2 \succ v_3 \sim v_4 \sim v_5$
6	2.58	2.81	2.99	3.00	2.98	$v_1 \succ v_2 \succ v_3 \sim v_4 \sim v_5$
7	2.63	2.84	2.99	3.00	2.98	$v_1 \succ v_2 \succ v_3 \sim v_4 \sim v_5$
10	2.74	2.89	2.99	3.00	2.99	$v_1 \succ v_2 \succ v_3 \sim v_4 \sim v_5$
∞	3.00	3.00	3.00	3.00	3.00	$v_1 \sim v_2 \sim v_3 \sim v_4 \sim v_5$

From Table 3.4, we distinguish three types of consensus rankings.

- $p = 1$ (utilitarian): v_1 and v_2 are equally preferred. They are both preferred over the others (v_3, v_4, v_5), which are equally preferred.
- $p \in [2..10]$ (intermediate): v_1 becomes more preferred than v_2, while both are

still more preferred than the rest of values (v_3, v_4, v_5). The indifference between v_3, v_4 and v_5 holds.

- $p = \infty$ (egalitarian): all values are equally preferred.

We make two interesting observations from Table 3.4. First, recall (Subsection 3.5.1) that for $p = 2$, the consensus ranking results from computing the mean of the individual rankings to aggregate. Thus, our consensus ranking for $p = 2$ is the same as obtained by Siebert, Liscio, Murukannaiah, Kaptein, Spruit, Hoven and Jonker [175], which solely employs the mean of individual rankings to compute a consensus ranking. Second, the consensus position R_S for *all* moral values converges to 3 (central position in the ranking) as the value of parameter p increases.

3.6.1.2 Analysing the Consensus on Values-Alternatives Relationships

After analysing the impact of choosing different ethical principles on the consensus value preferences, we now discuss the results of the consensus value-option matrices. Recall from Subsection 3.3.1.2 that each individual holds a different view of the relationship between values and options. Computing the consensus on the value-option matrices unveils the most representative value-option matrix (i.e., the most representative interpretation of values) on which the consensus value preferences apply. Recall (Subsection 3.3.1.1) that the binary value-option matrix VO_i indicates whether a value v is promoted by option o or not according to individual i. We remark that considering binary matrices for expressing the relationship between values and alternatives is the result of applying the methodology proposed by Siebert, Liscio, Murukannaiah, Kaptein, Spruit, Hoven and Jonker [175]. However, in the EVS case study presented in the following section, we will discuss a different instance of the relationship between alternatives and values that contain more gradual evaluations [-1,1], representing the degree of promotion or demotion of a given value when performing a specific action.

For brevity, we only analyse the consensus in three cases of interest: the utilitarian ($p = 1$), the *mean principle* ($p = 2$), and the egalitarian ($p = \infty$) cases. For ease of understanding, we first discuss the mean principle case.

Mean principle ($p = 2$) (3.30) shows the resulting consensus value-option matrix. From Theorem 3.5.3, we know that aggregating individual value-option matrices results in a value-option matrix containing the mean values of the individual matrices. By multiplying the consensus value-option matrix by 100, we obtain the percentage of individuals considering value v relevant for option o. For example, v_3 is deemed relevant to 45.2% of the participants to justify option o_1; similarly, v_5 and v_2 are related to options o_2 and o_4, respectively, for more than 30% of the participants.

$$VO_S^{(p=2)} = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} o_1 & o_2 & o_3 & o_4 & o_5 & o_6 \end{array} \\ \begin{pmatrix} 0.116 & 0.116 & 0.134 & 0.117 & 0.121 & 0.075 \\ 0.065 & 0.039 & 0.015 & 0.349 & 0.037 & 0.061 \\ 0.452 & 0.054 & 0.055 & 0.017 & 0.015 & 0.004 \\ 0.110 & 0.177 & 0.048 & 0.026 & 0.016 & 0.041 \\ 0.048 & 0.396 & 0.012 & 0.011 & 0.028 & 0.022 \end{pmatrix} \end{array} \tag{3.30}$$

Utilitarian principle ($p = 1$) The consensus value-option matrix VO_S is a zero matrix, i.e., $VO_S[v, o] = 0$ for every value $v \in V$ and option $o \in O$. This indicates that no value $v \in V$ is promoted in any option $o \in O$. Despite Table 3.2a showing that there are values that are frequently annotated for some options, most individual value-option matrices are sparse (have many zeroes). This is reflected in (3.30), where there are no options in which a majority of individuals indicated a specific value to be relevant (i.e., no entry of (3.30) is larger than 0.5). This leads the method to compute a consensus of zeroes because ℓ_1-regression tends to have residuals $(VO_i[v, o] - VO_S[v, o])$ equal to 0. This is well-known in the optimisation literature [24].

Egalitarian principle ($p = \infty$) The consensus value-option matrix has $VO_S[v, o] = 0.5$ for every value $v \in V$ and option $o \in O$. This is because the egalitarian principle minimises the disagreement with the most displaced elements of the society, i.e., the distance between the consensus for each value-option pair and the individuals assigning the largest and lowest judgement to the value-option pair. Recall from Table 3.2 that our value-option matrices are binary. By peeking into the individual option

matrices, we observe that for every value-option pair (v, o), the maximum $VO_i[v, o]$ is 1 (at least one individual mentions value v to justify option o). The minimum $VO_i[v, o]$ is 0. Hence, $VO_S[v, o] = 0.5$ for every moral value $v \in V$, and option $o \in O$ because 0.5 is the consensus value that minimises the distance to the most displaced individual.

3.6.2 European Value Study

We now study the aggregation of value systems for the European Values Study case. Similar to the PVE case, we explore how the resulting value system is affected by the parameters of our aggregation approach. However, unlike the PVE case, in the EVS case, we consider two cases for social influence: (1) where all countries have the same influence ($w_i = 1$), and (2) where social influence w_i of a country i is proportional to the population of that country (i.e., $w_i = {}^{\text{population}_i}/_{\sum_j \text{population}_j}$).[8]

Recall from Subsection 3.3.2 that this case considers 34 European countries, two values (religiosity (rl) and permissiveness (pr)), and two actions (divorce (dv) and parenthood by homosexual couples (ho)). Tables 3.5 and 3.6 show the consensus value systems for different ethical principles, considering equal and population-based social influence, respectively. We report results for the utilitarian ($p = 1$) and egalitarian cases ($p = \infty$) and $p = 2$ and $p = 3$. We choose to show these values of parameter p as a change of value preferences can be observed in the consensus computed with $p = 2$ and $p = 3$ in the population-based social influence scenario (Table 3.6). In both tables, P_S stands for the value of the consensus on moral value preferences, whereas the α values contain the consensus values on moral value-action relationships. For instance, $P_S[rl, pr]$ indicates the consensus grade of preference of the religiosity moral value over the permissiveness moral value. Further, $\alpha^S_{rl}(dv)$ denotes the consensus grade of promotion of the religiosity value when performing action divorce.

[8]The population data is accessed from *Worldometers* (https://www.worldometers.info/world-population/population-by-country) in Sept. 2021.

Table 3.5: Value preferences (columns 2, 3, and 8) and value judgement functions (columns 4–7) of the consensus European value system when considering equal social influence.

p	$P_S[rl, pr]$	$P_S[pr, rl]$	$\alpha^S_{rl}(ho)$	$\alpha^S_{rl}(dv)$	$\alpha^S_{pr}(ho)$	$\alpha^S_{pr}(dv)$	Value pref.
1	0.504	0.496	−0.288	0.013	−0.054	0.318	$rl \succ pr$
2	0.535	0.465	−0.224	0.031	−0.032	0.315	$rl \succ pr$
3	0.548	0.452	−0.200	0.029	−0.029	0.280	$rl \succ pr$
∞	0.580	0.420	−0.148	0.027	−0.077	0.159	$rl \succ pr$

Table 3.6: Value preferences (columns 2, 3, and 8) and value judgement functions (columns 4–7) of the consensus European value system when social influence considers population.

p	$P_S[rl, pr]$	$P_S[pr, rl]$	$\alpha^S_{rl}(ho)$	$\alpha^S_{rl}(dv)$	$\alpha^S_{pr}(ho)$	$\alpha^S_{pr}(dv)$	Value pref.
1	0.444	0.556	0.007	0.123	0.187	0.503	$pr \succ rl$
2	0.495	0.505	−0.128	0.111	0.059	0.400	$pr \succ rl$
3	0.521	0.479	−0.154	0.080	0.015	0.364	$rl \succ pr$
∞	0.580	0.420	−0.148	0.027	−0.077	0.159	$rl \succ pr$

Equal social influence In Table 3.5, all countries have the same social influence. We observe that all the *consensus European value systems* prefer religiosity over permissiveness. This is because the computed preference of religiosity over permissiveness is larger than 0.5 ($P_S[rl, pr] > 0.5$) for all ethical principles (p) considered. However, for the utilitarian case ($p = 1$), this preference is barely noticeable (i.e., both $P_S[rl, pr]$ and $P_S[pr, rl]$ are close to 0.5), whereas the difference between $P_S[rl, pr]$ and $P_S[pr, rl]$ increases as we increase the value of parameter p. This transition happens because, for greater values of p, the consensus tends to reduce the maximum disagreement with the most "extreme" preferences, which in our case is by Georgia ($P_i[rl, pr] = 0.94$). In addition, for all p's, the parenthood by homosexual couples is valued negatively (see the values of $\alpha^S_{rl}(ho)$ and $\alpha^S_{pr}(ho)$), whereas divorce is slightly accepted (see the values of $\alpha^S_{rl}(dv)$ and $\alpha^S_{pr}(dv)$).

Population-based social influence When considering social influence based on population size (Table 3.6), we observe significant changes compared to the previous case. Specifically, for the utilitarian principle ($p = 1$), permissiveness is preferred over religiosity (since $P_S[pr, rl]$ is larger than $P_S[rl, pr]$), and both adoptions by homosexual couples and divorce promote both values. This consensus shift is due to the

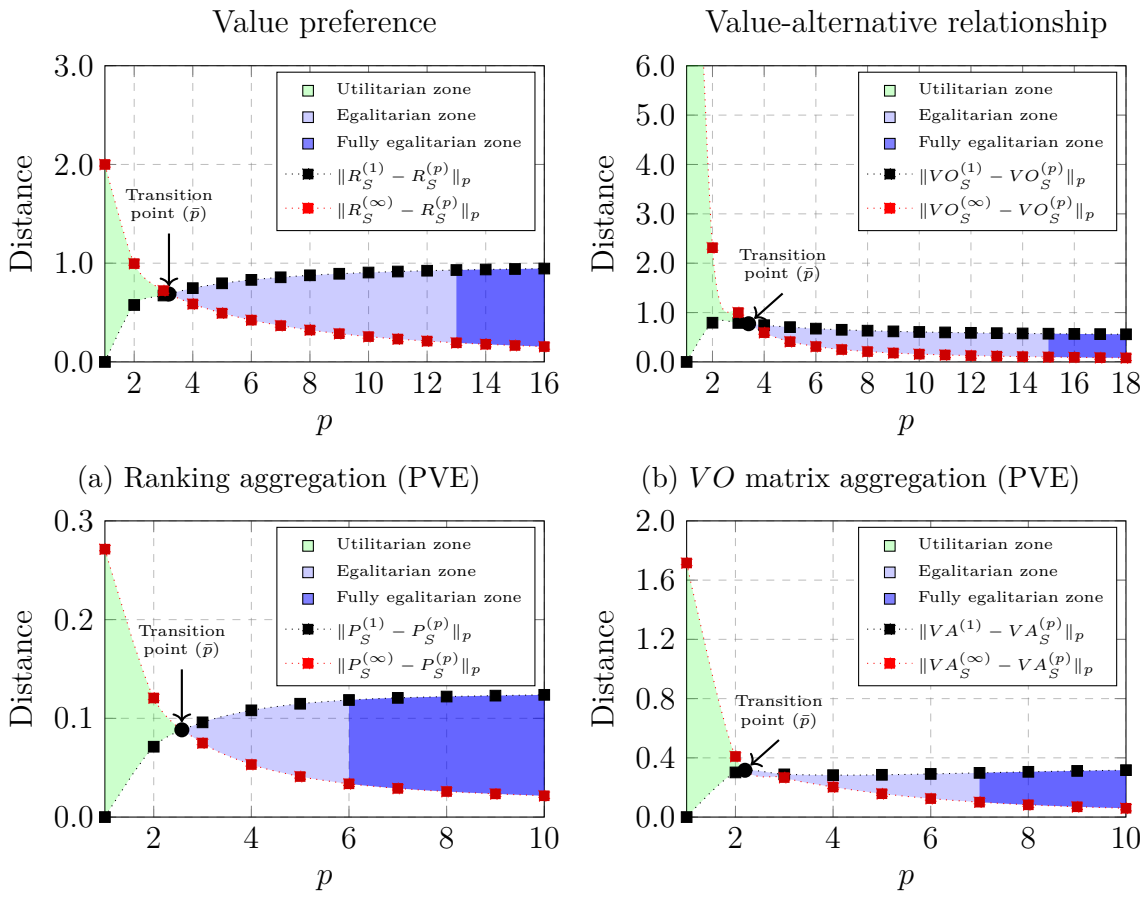
social influence that highly populated European countries, such as Germany, France, and Great Britain, wield, which prefer permissiveness over religiosity. Nonetheless, as we increase p, we obtain the same transition towards religiosity as observed in Table 3.5. Indeed, for values of parameter p equal or larger than 3, we obtain the same trend on all consensus values (for P_S and α) for both the population-based (weighted) and the equal social influence (unweighted) cases. This confirms that the impact of weights (w_i) vanishes when considering high values of parameter p in accordance with (3.4).

Overall, choosing the aggregation parameters (p and w_i) determines between swaying towards prevailing (majority) value systems or value systems that lie closer to divergent opinions. In the EVS case, we even obtain a swap for the preferences of *religiosity* and *permissiveness*.

3.6.3 Characterising the Space of Ethical Principles

We characterise the *whole* space of ethical principles (from utilitarian to egalitarian) available to a decision-maker when computing a consensus value system. We do so to determine whether an ethical principle p produces a consensus leaning towards utilitarian ($p = 1$) or egalitarian ($p = \infty$). To achieve our objective in the PVE case, we compute the consensus ranking R_S considering a given p (denoted as $R_S^{(p)}$). We measure the distance between $R_S^{(p)}$ and the one corresponding to $p = 1$ and $p = \infty$, denoted as $R_S^{(1)}$ and $R_S^{(\infty)}$ respectively. Formally, we denote these two distances as $\left\|R_S^{(1)} - R_S^{(p)}\right\|_p$ and $\left\|R_S^{(p)} - R_S^{(\infty)}\right\|_p$. Analogously, we define $\left\|VO_S^{(1)} - VO_S^{(p)}\right\|_p$ and $\left\|VO_S^{(p)} - VO_S^{(\infty)}\right\|_p$ for value-option matrices. In the same vein, for the EVS case, we denote the distances between value preferences as $\left\|P_S^{(1)} - P_S^{(p)}\right\|_p$ and $\left\|P_S^{(p)} - P_S^{(\infty)}\right\|_p$, and denote the distances between value-action matrices as $\left\|VA_S^{(1)} - VA_S^{(p)}\right\|_p$ and $\left\|VA_S^{(p)} - VA_S^{(\infty)}\right\|_p$.

By making use of the above-defined distances, we can determine a *transition point* (denoted as $\bar{p}$) that is the equidistant ethical principle whose computed consensus is between the fully utilitarian and the fully egalitarian consensuses. In addition, we define the *limit point*, $\hat{p}$, as the ethical principle such that all $p > \hat{p}$ produce



(a) Ranking aggregation (PVE)

(b) VO matrix aggregation (PVE)

(c) Preference matrix aggregation (EVS)

(d) VA matrix aggregation (EVS)

Figure 3.2: Distance between the consensus computed according to ethical principle p and the consensuses computed according to $p = 1$ (fully utilitarian, black line) and $p = \infty$ (fully egalitarian, red line). The transition point $\bar{p}$ is the ethical principle producing a consensus equidistant from the fully utilitarian and fully egalitarian ones. Thus, $\bar{p}$ divides the space of ethical principles into an *utilitarian zone* (more similar to the fully utilitarian consensus, green) and an *egalitarian zone* (more similar to the fully egalitarian consensus, light blue). The *fully egalitarian* (dark blue) zone marks the ethical principles that produce a consensus approximately equal (to a small ϵ) to the fully egalitarian one. PVE case study on top, and EVS case study for the population-based influence case at the bottom.

a consensus that is approximately equal (to a small ϵ) to the fully egalitarian one ($p = \infty$). We compute $\hat{p}$ that satisfies:

$$\frac{U_{p-1}^{(R)} - U_p^{(R)}}{U_p^{(R)}} < \epsilon,$$

where $U_p^{(R)}$ is the value of the distance function defined by (3.8).

Because of the transition and limit points, we can characterise different zones within the space of ethical principles as Figure 3.2 illustrates.

- The *utilitarian zone* is composed of all ethical principles leaning towards the fully utilitarian case ($p < \bar{p}$).
- The *egalitarian zone* is composed of all ethical principles leaning towards the fully egalitarian case but before surpassing the *limit point* ($\hat{p}$), i.e., $p \in [\bar{p}..\hat{p}]$.
- The *fully egalitarian zone* is composed of the set of all ethical principles greater than the limit point, i.e., $p > \hat{p}$.

Figure 3.2a plots the distances between the consensus ranking ($R_S^{(p)}$) with respect to the fully utilitarian ($R_S^{(1)}$) and fully egalitarian consensus rankings ($R_S^{(\infty)}$) as the value of the ethical principle p increases. Figure 3.2b shows analogous results for aggregating value-option matrices. We observe that the transition point is near 3 ($\bar{p} \sim 3$) for both the ranking and value-option matrix aggregation. However, the limit point varies for both cases ($\hat{p} = 13$ for ranking aggregation and $\hat{p} = 15$ for value-option matrix aggregation). This is because such points depend on the data of the problem.

The data dependency is corroborated when considering the EVS case. Figures 3.2c and 3.2d plot the distance between consensuses for value preference matrix aggregation and values-action matrix aggregation, respectively. In these figures, we observe that the transition and limit points are different with respect to those computed for the PVE case. The transition point for value preference and value-action matrices aggregation is between 2 and 3, whereas the limit point is $\hat{p} = 6$ in Figure 3.2c and $\hat{p} = 7$ in Figure 3.2d.

Therefore, a given ethical principle p can be interpreted as "more utilitarian" or "more egalitarian" depending on its relative position with respect to the transition point. However, notice that if the transition point $\bar{p}$ turns out to be different for the two aggregations that we perform to compute a consensus value system, then if we choose the same ethical principle p to apply to both aggregations, it may lie in different regions.

3.6.3.1 Takeaways for Decision-Makers

The comparison between the two case studies shows that the regions characterising the space of ethical principles depend on the domain and the data. Hence, a decision-maker must carefully examine the different ethical principle zones when choosing the ethical principle p. The decision-maker may desire to choose an ethical principle that leads to either a utilitarian or an egalitarian consensus for both aggregations (value preferences and value interpretations). However, the same value of parameter p could fall into different zones for moral value preferences and moral value-alternatives relationships aggregation. This motivates the need for characterising the *joint* utilitarian and egalitarian zones for value preferences and value-alternatives relationship aggregations before choosing an ethical principle.

The visual analysis displayed in Figure 3.2 intends to provide useful guidance for decision-makers concerned with obtaining a consensus on different value systems following an ethical principle of choice. In general, we propose the following guidelines:

1. Plot the distance between consensuses, as we do in Figure 3.2.
2. Plot the utilitarian, egalitarian, and fully egalitarian zones for value preference aggregation and values-alternatives relationship aggregation.
3. Compute the joint utilitarian and egalitarian zones for both value preference and value-alternative relationship. For instance, in the PVE case study (the case for EVS is analogous), we define:
    - $[1, \bar{p}_m)$, where $\bar{p}_m = min(\bar{p}_R, \bar{p}_{VO})$, as the joint utilitarian zone;
    - $(\bar{p}_M, \infty)$, where $\bar{p}_M = max(\bar{p}_R, \bar{p}_{VO})$, as the joint egalitarian zone;
    - $[\bar{p}_m, \bar{p}_M]$ as a *mixed* zone that contains ethical principles that lie in different zones, considering the consensus of ranking and value-option matrices.

After following the steps above, we obtain a *joint* space of ethical principles, which is partitioned into three zones of ethical principles:

- Utilitarian: ethical principles for which consensus on value preferences and value-alternatives relationship aggregation are both utilitarian.

- Egalitarian: ethical principles for which consensus on value preferences and value-alternatives relationship are both egalitarian.
- Mixed: ethical principles for which consensus on value preferences and value-alternatives relationship aggregation are not aligned (one is utilitarian, whereas the other is egalitarian).

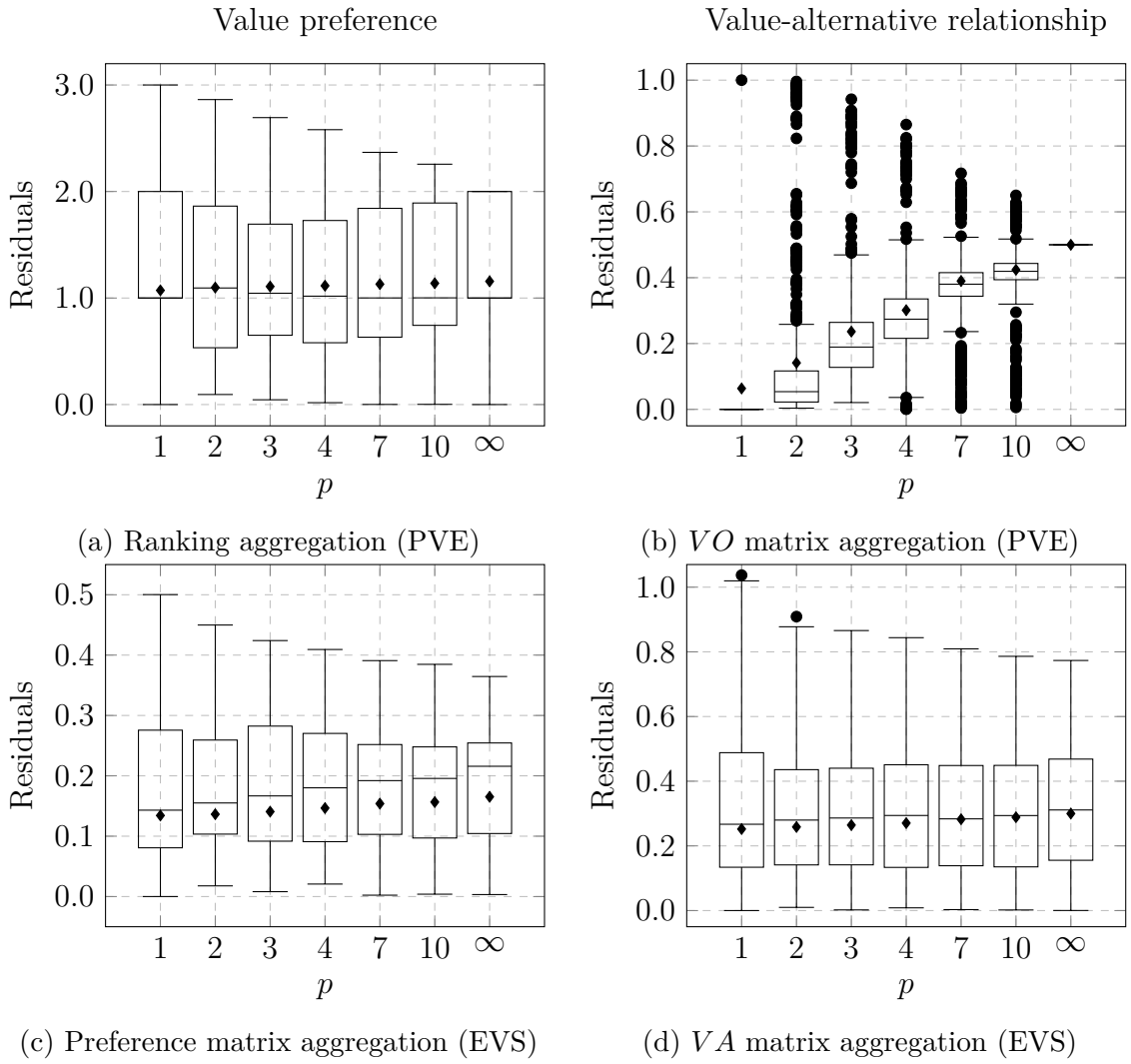
This joint ethical principle space provides the decision-maker with the necessary information on choosing an ethical principle p to compute the consensus.

3.6.4 The Relationship between Consensus and Individual Value Systems

In Subsections 3.6.1 and 3.6.2, we showed that the choice of the ethical principle impacts the resulting consensus value system. A further *micro-level* analysis helps us investigate how the relationship between the consensus and the individual value systems changes as the ethical principle changes. Such a micro-level analysis helps quantify the trade-off when moving away from the majority (utilitarian case) to include the minority (egalitarian case). This analysis complements the one conducted in Subsection 3.6.3 (when characterising ethical principles) to help the policymaker choose the ethical principle to employ when aggregating value systems.

In what follows, we analyse the distribution of *residuals*, which represent the *gap* between given consensus value systems and the individual value systems aggregated to obtain the consensus. For instance, when considering the PVE case study, $|R_i[v] - R_S[v]|$ yields the residual for the ranking of individual value system i with respect to the consensus value system regarding value v. Analogously, we can also calculate the residual for the value-option matrix of individual value system i ($|VO_i[v,o] - VO_S[v,o]|$). Figure 3.3 plots the distribution of the residuals obtained when computing consensus value systems for both case studies as the value of the ethical principle (p) increases.

First, we focus on the PVE case study. Figures 3.3a and 3.3b plot the distributions of the residuals for the ranking aggregation and the VO-matrix aggregation,



(a) Ranking aggregation (PVE)

(b) VO matrix aggregation (PVE)

(c) Preference matrix aggregation (EVS)

(d) VA matrix aggregation (EVS)

Figure 3.3: Boxplots of residuals for different ethical principles in the value preference and value-alternative aggregation for the PVE (top) and EVS (bottom) case study. The diamond represents the mean and the circles represent the outliers.

respectively. In Figure 3.3a, we observe that the utilitarian ethical principle ($p = 1$) yields the maximum residuals, hence confirming that the utilitarian principle considers less the individuals within the minority of society. As the value of parameter p increases towards the egalitarian principle, the maximum residuals decrease. For the egalitarian ethical principle ($p = \infty$), we obtain that the maximum residual is 33% smaller than for the utilitarian principle (from 3.0 down to 2.0). This is because the egalitarian ethical principle aims to reduce the distance between the consensus and the most distant individuals of society. Further, we observe that the mean of the residuals gradually increases as the value of parameter p increases (up to an

8% increase for the egalitarian case). The observations above are more pronounced when we analyse the residual distribution for aggregating individual VO matrices using different ethical principles (Figure 3.3b). On the one hand, the value of the maximum residual halves when moving from $p = 1$ (1.0) to $p = \infty$ (0.5). On the other hand, the mean of the residuals dramatically increases from a value close to 0 until reaching 0.5 for the egalitarian case ($p = \infty$), which amounts to a x7 increase of the mean[9].

We confirm the observations above when analysing the EVS case study. Figures 3.3c and 3.3d show the distribution of residuals for the aggregation of individual preference matrices P_i and the aggregation of individual value-action matrices VA_i, respectively for the weighted scenario (different social influences per country). On the one hand, the maximum residual decreases as we increase p for both preference matrix and value-action matrix aggregations. On the other hand, there is a smooth increase in the mean of the residuals as we increase p for both the preference matrix and value-action matrix aggregations. However, the values of residuals differ compared to the PVE case. This is due to the differences in each case study's set of individual value systems.

3.6.4.1 Takeaways for Decision-Makers

Our analysis shows that moving away from the majority (utilitarian) to include the minority (egalitarian) leads to an increase in the mean of the residuals and a decrease in the maximum residuals. While the guidelines provided in Subsection 3.6.3.1 specify macro-directives for choosing the ethical principle p, the analysis of residuals gives concrete information about the trade-offs between choosing different ethical principles. The decision-maker must weigh the extent to which the increase in the average residual of a large portion of individuals compensates for minimising the distance to the minority (when moving from utilitarian to egalitarian principle).

[9]Recall that value-option matrices are binary. We highlight that we obtain a consensus matrix full of zeros for the utilitarian principle ($p = 1$). Because the value-option matrices are binary, all the residuals are either 0 or 1. In fact, more than 75% of the residuals are 0. As to the egalitarian principle ($p = \infty$), we obtain a consensus matrix full of 0.5 values. Hence, all the residuals are 0.5.

3.7 Conclusions and Future Work

The main contribution of this chapter is to provide novel computational means and guidelines for a decision-maker to conduct a principled aggregation of value systems. This is a fundamental step towards designing AI systems that align with a group of individuals with different value systems, namely, towards addressing the *pluralistic value alignment* problem defined by Gabriel [72]. Our contributions make headway in developing trustworthy AI [47] systems that adhere to ethical principles and values.

We show that the problem of computing a consensus value system can be cast as an ℓ_p*-regression* [3] (or *norm approximation* [24]) problem. By doing so, we provide a principled and general theoretical framework to solve the above-mentioned problem for a range of ethical principles—from utilitarian (maximum utility) to egalitarian (maximum fairness)—in a scalable and reliable way, thanks to recent results in the machine learning literature [3]. Importantly, our approach also allows us to compute the consensus for *any* single value of p, which was not possible before for the generator of social choice functions that we employ (González-Pachón and Romero [82] can only deal with $p = 1$ and $p = \infty$).

We study the aggregation of value systems using real-world data from two case studies: the Participatory Value Evaluation (PVE) [99] process and the European Values Study (EVS) [65]. Our empirical evaluation of the case studies draws insights for a decision-maker about how to employ our computational tools to perform value system aggregation. In particular, we show how to proceed to: (i) quantify the impact of choosing different ethical principles on the resulting consensus value system; (ii) characterise the semantics of the available ethical principles (to determine whether a given ethical principle produces a consensus leaning towards the utilitarian or the egalitarian ethical principle); and (iii) quantify the trade-off when moving away from the majority to include the minority. Importantly, our observations vary per case study since they handle different value systems.

In this chapter, we treat the aggregation of individual value systems. However, the aggregation must be preceded by the *estimation* of individual value systems [P6]. To

this end, surveys such as PVE and EVS can be employed to collect value-laden input from the participants, and methods like Siebert, Liscio, Murukannaiah, Kaptein, Spruit, Hoven and Jonker [175] (Subsection 3.3.1.2) are employed to compound the survey answers into individual value systems. Such methods could be further automated using natural language processing to automatically detect the values that are motivating the survey answers [10, 120]. Ultimately, the aggregation of value systems is the result of pipelining the estimation of individual value systems with the computation of a consensus value system.

In conclusion, the contributions of this chapter address the first three research questions outlined in Section 1.2.1.1, which correspond to the first three contributions of this thesis.

- **Question Q1:** How do we formalise a Value System? We formalise a Value System by identifying its key components: a set of values, a set of contextual alternatives, the relationships between alternatives and values, and the preferences over values. We then adapt this definition to two real-world case studies, showing that both the relationships between alternatives and values, as well as the preferences over values, can be represented either as cardinal (binary) relations or as graded relations.

- **Question Q2:** How can we formally define and solve the problem of aggregating individual value systems as an optimisation problem? We draw on the social choice literature and employ a p-parameterised distance function. The problem of computing a consensus value system is then formalised as the minimisation of this distance function, which we cast as an ℓ_p-regression problem. We demonstrate how this problem can be efficiently solved for any choice of ethical principle p using state-of-the-art algorithms.

- **Question Q3:** How to support decision-makers in choosing the ethical principle to compute a consensus value system? We evaluate the impact of different ethical principles on the resulting consensus value system by analysing (i) the distances between consensuses obtained under distinct ethical principles and (ii) the gap between each consensus and the individual value systems. Based

on this analysis, we provide guidelines to support decision-makers in selecting an appropriate ethical principle.

Chapter 4

Research Project Planning

4.1 Introduction

Project planning is a notoriously complex endeavour that has received significant attention in multiple scientific fields [104] due to the involvement of multifaceted and challenging problems, including but not limited to scheduling, task assignment, prediction of future events, formation of well-performing teams, coordination, preference elicitation, and preference satisfaction. Along these lines, providing recommendations to guide the planning and management of early-stage projects is proven to be crucial to project success [154, 144].

It should come as no surprise that planning *academic research projects* also presents several challenging aspects. The work of researchers involved in each project must be accurately organised in order to ensure that the dedication required by each *work package* (i.e., a group of related tasks within the project that form an isolated unit of work) is satisfied. Nonetheless, such dedication is also subject to constraints of both legal (e.g., researchers cannot be assigned more than a well-defined amount of working hours during a day or a month) and personal nature (e.g., researchers have preferences over the amount and the distribution of their research effort among the projects they are involved in). Additionally, budget availability is obviously a crucial aspect that determines the number of person-hours that can be devoted to each project, also taking into account the experience and the professional degree of each researcher. Finally, projects change over time (e.g., researchers can join or

leave the team, new projects can be approved, etc.). Therefore, one must account for uncertainty when planning academic research projects.

Perhaps unsurprisingly, to the best of our knowledge, the majority of research institutions still tackle project planning by manually carrying out the above-mentioned tasks due to the lack of automated solutions. Against this background, in this chapter, we formally define the *Research Project Planning* (RPP) problem and we put forward an automated solution that can compute the optimal dedication for a set of researchers involved in several projects, to match the preferred dedication of researchers as much as possible while efficiently utilising the available budget. To achieve this objective, and therefore, answer research question Q4, we propose a novel ℓ_2-*norm approximation* [24] model that enables the use of off-the-shelf solvers to compute optimal solutions in a matter of seconds. Our solutions are also robust against the uncertainty inherent in academic project planning, thanks to the proposed use of *Stochastic Robust Approximation* (SRA) [24].

Our approach has already been implemented as part of a planning prototype (shown in Figure 4.6) that is currently being tested in our research institution; hence, we believe our contributions could pave the way to a practical solution for academic project planning.

More specifically, we make the following contributions to the state-of-the-art:

- We model RPP as an ℓ_2-norm approximation problem, discussing both a hard and a soft-constrained model. Our model allows the user to directly express preferences (i.e., the importance of matching the specified target dedication versus the importance of efficiently spending the available budget) as model parameters.
- We evaluate our approach on real data provided by our institution, illustrating the impact of the parameters that reflect the users' preferences on the computed solutions.
- Results also show that our approach is capable of accounting for uncertainty within real-world projects by balancing the dedication of researchers in case

of unexpected changes in projects' conditions. Moreover, our approach can compute optimal solutions for thousands of projects and researchers within minutes.

The rest of this chapter is organised as follows. Section 4.2 provides background on the *Stochastic Robust Approximation* (SRA), which allows our model to account for uncertainty. In Section 4.3 we formally define the RPP problem, while in Section 4.4 we show how we cast RPP as a norm approximation problem, both as a hard and soft-constrained version. In this section, we also explain how our model accounts for uncertainty. Section 4.5 presents our experimental analysis considering real-world data provided by our research institution. Finally, Section 4.6 concludes this chapter.

4.2 Stochastic Robust Approximation

In optimisation problems, we usually assume that the input data is certain. Still, in some cases, it is desirable to take into account some uncertainty or a possible variation in the data [24]. As an example, consider an approximation problem such as LS (2.6), where matrix A and vector $\mathbf{b}$ represent the input data. To incorporate uncertainty, we allow both A and $\mathbf{b}$ to vary statistically. More precisely, we treat $A \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$ as random variables such that

$$\begin{aligned} A =& \bar{A} + U, \\ \mathbf{b} =& \bar{\mathbf{b}} + u, \end{aligned}$$

where $\bar{A}$ and $\bar{\mathbf{b}}$ are the constant matrix and vector indicating the average value of A and $\mathbf{b}$ respectively, while $\bar{U}$ and $\bar{\mathbf{u}}$ are a random matrix and vector with zero mean. Hence, since A and $\mathbf{b}$ are random variables, it is reasonable to minimise the expected value of the norm such that

$$\text{minimise} \;\; \mathbf{E}\|A\mathbf{x} - \mathbf{b}\|_2^2. \tag{4.1}$$

This problem is the so-called *Stochastic Robust Approximation* (SRA) problem [24]. It is usually not tractable since, in most cases, it is very difficult to evaluate the

objective or its derivatives. However, Ben-Tal and Nemirovski [17] study the cases where the convex SRA is tractable, such as LP, QP, and others. A case where the SRA problem in (4.1) can be solved occurs when the input data assumes only a finite number of values, i.e.,

$$\mathbf{prob}(A = A_i, \mathbf{b} = \mathbf{b}_i) = p_i, \quad i = 1, \ldots, k,$$

where $A_i \in \mathbb{R}^{m \times n}$, $\mathbf{b}_i \in \mathbb{R}^m$, $1^T p = 1$, $p \geq 0$. In this case (4.1) has the form

$$\text{minimise } p_1 \|A_1 \mathbf{x} - \mathbf{b}_1\|_2^2 + \cdots + p_k \|A_k \mathbf{x} - \mathbf{b}_k\|_2^2,$$

and can be rewritten as

$$\begin{aligned} &\text{minimise } \mathbf{p}^T \mathbf{t} \\ &\text{subject to } \|A_i \mathbf{x} - \mathbf{b}_i\| \leq t_i, \quad i = 1, \ldots, k, \end{aligned}$$

where the variables are $\mathbf{x} \in \mathbb{R}^n$.

Furthermore, Mulvey, Vanderbei and Zenios [134] propose robust models for convex-constrained optimisation problems, where the constraints have to be satisfied for each scenario $k \in \Omega$. Taking as an example the *convex optimisation problem in standard form*, the robust optimisation model is defined as

$$\begin{aligned} \text{minimise } & \sum_{k \in \Omega} p_k \cdot f_{0,k}(\mathbf{x}), \\ \text{subject to } & f_{i,k}(\mathbf{x}) \leq 0, && i = 1, 2, \ldots, m \quad \text{for all } k \in \Omega \\ & \mathbf{a}_{i,k}^T \mathbf{x} - b_{i,k} = 0, && i = 1, 2, \ldots, p \quad \text{for all } k \in \Omega. \end{aligned} \tag{4.2}$$

4.3 Problem Definition

In this section, we formally define the RPP problem. Intuitively, we tackle the problem of planning the dedication of a set of researchers who are involved in a set of research projects, to match the preferred dedication of each researcher as much as

possible and efficiently utilise the available budget, while satisfying the constraints inherent in research project planning.

4.3.1 Basic Definitions

We consider a *planning horizon* $M = (a, \ldots, b)$ as a sequence of months. We assume that M is sufficiently large to cover the duration of all the projects.

We consider a set of researchers R, a function $\in : R \to \mathbb{R}^+$ that provides the time slot (i.e., person-hour) cost for each researcher (e.g., based on their experience within the institution) and a function $\tau : R \times M \to \mathbb{N}$ that provides the maximum number of working time slots per month for each researcher.

We consider a set of work packages $W \subseteq M \times M$, i.e., $w = (w_s, w_e) \in W$ is a couple representing the start month w_s and end month w_e of work package w. We consider a function $involved : R \times W \to \{\top, \bot\}$ that expresses whether or not a researcher is involved in a work package and a vector $\mathbf{d} \in \mathbb{R}^{|W|}$, such that d_w measures the minimum total dedication (number of time slots) required by each work package. We consider a set of projects P as a *partition* of W, i.e., each $p \in P$ is a set of work packages and each $w \in W$ belongs to one project.

To formally pose our optimisation problem, we also consider a set $X_W \subseteq R \times W \times M$ of tuples such that $(r, w, m) \in X_W$ if $involved(r, w) = \top$ and $m \in [w_s, w_e]$. We also define the set $X_P \subseteq R \times P \times M$ in the corresponding way. Finally, we consider a vector $\mathbf{x} \in \mathbb{R}^{|X_W|}$ of $|X_W|$ real decision variables each associated to a tuple in X_W, such that $x_{r,w,m} \in [0, \tau(r, m)]$. Intuitively, $x_{r,w,m}$ represents the number of time slots that the researcher $r \in R$ devotes to work package $w \in W$ during the month $m \in M$. We assume that one-time slot contributes the same amount of research work independently of the researcher, following the standard practice of European projects' proposals.

4.3.2 Optimisation Objective

The main objective of our approach is to compute the optimal value of each decision variable of the vector X_W to achieve the following objectives.

Firstly, we aim to match a given *target dedication* associated with each researcher $r \in R$ for each month and each project in which they are involved. Formally, we consider a vector $\mathbf{t} \in \mathbb{R}^{|X_P|}$ of $|X_P|$ values such that $t_{r,p,m}$ represents the target dedication of researcher $r \in R$ to project $p \in P$ during month $m \in M$. This target dedication may stem from the fact that researchers prefer certain workloads during specific periods of the year, or because the research institution wants to assign appropriate workloads to researchers to reflect their roles (e.g., principal investigator, contracted researcher, etc.).

Secondly, assuming that each project $p \in P$ is associated with a *budget* b_p (where $\mathbf{b} \in \mathbb{R}^{|P|}$), our approach aims at maximising the expenditure for contracts of the researchers involved in p without exceeding b_p. This second criterion is motivated by the fact that a research project (when granted) is allocated a given budget that usually has to be entirely spent during the duration of such a project. Henceforth, it is reasonable to assume that research institutions aim to efficiently use the budget of the project as much as possible. Notice that, for simplicity, here we do not consider other expenses such as equipment, travel costs, etc. Nonetheless, our approach can be easily adapted by defining b_p as the portion of the total budget that has been allocated to the personnel.

4.3.3 Hard Constraints

We now specify some constraints that will allow us to reflect the restrictions inherent in real-world scenarios.

1. The total cost of the researchers working in a given project $p \in P$ does not exceed b_p. We achieve this by defining the matrix $B \in \mathbb{R}^{|P| \times |X_W|}$ where each row corresponds to a project $p \in P$ and each column to a tuple $(r, w, m) \in X_W$, such that

$$B_{p,(r,w,m)} = \begin{cases} €(r), & \text{if } w \in p, \\ 0, & \text{otherwise,} \end{cases} \tag{4.3}$$

   and by imposing the linear constraint $B\mathbf{x} \leq \mathbf{b}$.

2. The required dedication for each work package is satisfied. We achieve this by

defining the matrix $D \in \{0,1\}^{|W| \times |X_W|}$ where each row corresponds to a work package $w \in W$ and each column to a tuple $(r, w, m) \in X_W$, such that

$$D_{w,(r,w',m)} = \begin{cases} 1, & \text{if } w = w', \\ 0, & \text{otherwise,} \end{cases} \tag{4.4}$$

and by imposing the linear constraint $D\mathbf{x} \geq \mathbf{d}$.

3. The total monthly dedication of each researcher r in a given month m is at most $\tau(r, m)$. We achieve this by defining the matrix $T \in \{0,1\}^{(|R| \cdot |M|) \times |X_W|}$ where each row corresponds to a researcher $r' \in R$ and a month $m' \in M$ and each column to a tuple $(r, w, m) \in X_W$, such that

$$T_{(r',m'),(r,w,m)} = \begin{cases} 1, & \text{if } r' = r,\ m' = m, \\ 0, & \text{otherwise,} \end{cases} \tag{4.5}$$

and by imposing the linear constraint $T\mathbf{x} \leq \tau$.

Notice that, by abuse of notation, here we consider τ as a vector containing all the $|R| \cdot |M|$ values of the function τ defined in Section 4.3.

We remark that we do not aim to specify an exhaustive list of constraints that might apply during the research planning phase of every possible research institution. Nonetheless, our framework allows one to easily implement additional restrictions that might apply in specific scenarios.[1]

4.4 Solving the RPP Problem

Having defined the objective and the constraints of our optimisation approach, we are now ready to formally define RPP as a constrained *norm approximation problem* [24].

To this end, we define the matrix $A \in \{0,1\}^{|X_P| \times |X_W|}$ with $|X_P|$ rows (each corres-

[1] As a matter of fact, our prototype mentioned in Section 4.6 includes minor additional constraints that are specific to our research institution, which we decided to leave out of the formalisation discussed here for the sake of simplicity.

ponding to a tuple $(r, p, m) \in X_P$) and $|X_W|$ columns (each corresponding to a tuple $(r', w, m') \in X_W$), such that

$$A_{(r,p,m),(r',w,m')} = \begin{cases} 1, & \text{if } w \in p, r = r', m = m', \\ 0, & \text{otherwise.} \end{cases} \tag{4.6}$$

By using (4.3), (4.4), (4.5), and (4.6), we formally define the optimisation problem with the objectives discussed in Section 4.3.2 subject to the constraints discussed in Section 4.3.3 as

$$\begin{aligned} \text{minimise} \quad & \|A\mathbf{x} - \mathbf{t}\|_2^2 + \|B\mathbf{x} - \mathbf{b}\|_2^2, \\ \text{subject to} \quad & B\mathbf{x} \leq \mathbf{b}, \\ & D\mathbf{x} \geq \mathbf{d}, \\ & T\mathbf{x} \leq \tau. \end{aligned} \tag{4.7}$$

The objective function of (4.7) involves two components that correspond to the two objectives discussed in Section 4.3.2. Specifically, $\|A\mathbf{x} - \mathbf{t}\|_2^2$ denotes the deviation between the computed dedication $A\mathbf{x}$ and the target dedication $\mathbf{t}$. Similarly, $\|B\mathbf{x} - \mathbf{b}\|_2^2$ denotes the deviation in terms of budget utilisation. In this formulation, $B\mathbf{x}$ cannot exceed $\mathbf{b}$ since we also impose the linear constraint $B\mathbf{x} \leq \mathbf{b}$. We discuss a formulation without this constraint in the next section.

We also remark that in (4.7) we directly show our norm approximation formalisation as one that employs the ℓ_2 norm, i.e., the p-norm $\|\mathbf{x}\|_p = (\sum_i |x_i|^p)^{1/p}$ for $p = 2$.[2] To be more precise, following a common practice in the optimisation and machine-learning literature, we consider the square of the ℓ_2 norm, hence obtaining a constrained LS problem [24]. It is very well known that LS problems can be solved very efficiently by employing a wealth of very mature off-the-shelf solvers, further motivating our formalisation of choice for the RPP problem.

The above computational discussion holds for *real* decision variables (see Section 4.3.1). On the other hand, if one considers *integer* decision variables, the problem

[2]Other values of p could also be considered, resulting in different types of norm approximation problems [24]. An in-depth analysis of the impact of the parameter p on our approach is beyond the scope of this thesis.

in (4.7) becomes a MIQP problem, which can also be solved by off-the-shelf solvers such as CPLEX (despite being an NP-Hard problem in general).

4.4.1 Introducing Soft Budget Constraints

In our current formalisation of the RPP problem, the budget b_p of a project p cannot be exceeded in any way, as imposed by Constraint 1. However, in real-world research planning scenarios, it could be useful to relax such a constraint, thus making it a *soft constraint*. For example, funding entities could allow for small deviations from the predetermined budget via a proper justification. Furthermore, research institutions might have some funds reserved to integrate small extra expenses that cannot be covered by the budget of the project. Nonetheless, we desire to be able to control the positive values in the vector $B\mathbf{x} - \mathbf{b}$, which should be penalised differently with respect to the elements with a negative value. Along these lines, we define a "soft" version of (4.7) in which we do not consider Constraint 1 and we assign different weights to the positive and negative components of the elements of the vector $B\mathbf{x}-\mathbf{b}$. For the sake of coherence, we also assign a weight to the $\|A\mathbf{x} - \mathbf{t}\|_2^2$ term, obtaining

$$\begin{aligned} \text{minimise } & \alpha \cdot \|A\mathbf{x} - \mathbf{t}\|_2^2 + \beta \cdot \|\min(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2 + \gamma \cdot \|\max(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2, \\ \text{subject to } & D\mathbf{x} \geq \mathbf{d}, \\ & T\mathbf{x} \leq \tau, \end{aligned} \tag{4.8}$$

where the parameters $\alpha, \beta, \gamma \in \mathbb{R}^+$ control the above-mentioned weights and the min (resp. max) function represents the *element-wise* minimum (resp. maximum) between two vectors. It should be noted that the terms $\|\min(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2$ and $\|\max(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2$, although of very similar form and both subject to minimisation as part of the objective function, are of very different nature. On the one hand, $\|\min(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2$ is being minimised as it is reflecting our second optimisation criterion discussed in Section 4.3.2. On the other hand, $\|\max(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2$ is being minimised as it is effectively implementing a soft constraint taking the place of Constraint 1. Along these lines, we can characterise the semantics of the parameters α, β, and γ as follows. On the one hand, α represents the importance of matching

the target dedication $\mathbf{t}$, whereas β is the importance of spending the budget $\mathbf{b}$ as efficiently as possible. Finally, γ is the importance of exceeding the available budget as little as possible. By setting $\gamma = \infty$ we obtain a formulation equivalent to the hard-constrained one in (4.7).

The rest of this section discusses the computational complexity of (4.8). More specifically, we turn our attention to showing that the soft-constrained problem in (4.8) is still a constrained LS problem, hence maintaining the same desirable properties of the computational efficiency of (4.7).

Proposition 4.4.1. *The optimisation problem in* (4.8) *can be posed as the constrained LS problem*

$$\begin{aligned} \textit{minimise}\ & \alpha \cdot \|A\mathbf{x} - \mathbf{t}\|_2^2 + \beta \cdot \|\mathbf{v}\|_2^2 + \gamma \cdot \|\mathbf{u}\|_2^2, \\ \textit{subject to}\ & D\mathbf{x} \geq \mathbf{d}, \\ & T\mathbf{x} \leq \tau, \\ & B\mathbf{x} - \mathbf{b} \leq \mathbf{u}, \\ & \mathbf{u} \geq \mathbf{0}, \\ & B\mathbf{x} - \mathbf{b} \geq \mathbf{v}, \\ & \mathbf{v} \leq \mathbf{0}, \end{aligned} \tag{4.9}$$

where $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{|P|}$ *are additional auxiliary variables.*

Proof. As a first step of our proof, we show that

$$\text{minimise}\ \ \|\max(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2$$

can be represented as a constrained LS problem with an additional variable $\mathbf{u} \in \mathbb{R}^{|P|}$, as follows:

$$\begin{aligned} \text{minimise}\ & \|\mathbf{u}\|_2^2, \\ \text{subject to}\ & B\mathbf{x} - \mathbf{b} \leq \mathbf{u}, \\ & \mathbf{0} \leq \mathbf{u}. \end{aligned} \tag{4.10}$$

In a similar way we can obtain a constrained LS problem corresponding to the minimisation of $\|\min(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2$:

$$\begin{aligned} &\text{minimise} && \|\mathbf{v}\|_2^2, \\ &\text{subject to} && B\mathbf{x} - \mathbf{b} \geq \mathbf{v}, \\ & && \mathbf{0} \geq \mathbf{v}. \end{aligned} \tag{4.11}$$

We can now replace (4.10) and (4.11) with the corresponding terms within (4.8), obtaining the problem in (4.9). □

4.4.2 Introducing Uncertainty with Stochastic Instances

In the above-defined formalisation of the problem (4.9), we assumed that the data is fully known a priori, i.e., matrices A, D, T and B and vectors $\mathbf{t}, \mathbf{d}, \tau$ and $\mathbf{b}$ are fixed. However, in real-world research planning scenarios, the data and conditions of the plan can change throughout the planning horizon. For instance, a new project may be assigned to the research institution, new researchers may start working on ongoing projects, the requirements of the project may change, etc. Hence, it is crucial to handle such uncertainty inherent in research projects.

Usually, when an unexpected change happens in the middle of the duration of a research project, a straightforward strategy is to recompute the planning for the rest of the project. To implement such a strategy, we can compute the planning for the following months, considering the dedication and the budget already spent on the work packages. Unfortunately, this simple strategy might not work in some situations. For instance, consider the case when new projects are assigned to the research institution. In such cases, researchers' time distribution may adapt to cover the dedication of ongoing and new projects. This may probably cause an abrupt increase in researchers' efforts to satisfy the dedication of the projects in the following months. Furthermore, in the most extreme cases, the dedication of the projects may not be possible to satisfy without the involvement of more researchers. Along these lines, we now discuss how we explicitly account for uncertainty thanks to a technique called SRA [24], which we introduced in Section 4.2.

First, we start by considering a set of possible scenarios $\Omega = \{\omega_0, \omega_1, \ldots, \omega_K\}$, where ω_0 is the scenario with the current data, and $\omega_1, \ldots, \omega_K$ are possible variations of ω_0. A^k, D^k, T^k and B^k for all $k = 0, 1, \ldots, K$ are the matrices for the particular scenarios, using the same notation for the vectors. By using such a general methodology, the new scenario can cope with variations of any kind: additions of new projects, changes in the preferences of researchers, changes of budgets, and involvement of new researchers in projects are some examples.

Then, by following the methodology previously introduced in Section 4.2 [24, 134], we aim to minimise the expected value of the objective function, which represents the weighted mean of the objective functions of each scenario. Thus, we minimise the expected value of the problem (4.9) as follows.

$$
\begin{aligned}
\text{minimise} \quad & \mathbf{p}^T \cdot \Big(\alpha \cdot \mathbf{A} + \beta \cdot \mathbf{B} + \gamma \cdot \mathbf{\Gamma}\Big), \\
\text{subject to} \quad & \|A^k\mathbf{x} - \mathbf{t}^k\|_2^2 \leq \mathbf{A}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& \|\mathbf{v}^k\|_2^2 \leq \mathbf{B}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& \|\mathbf{u}^k\|_2^2 \leq \mathbf{\Gamma}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& B^k\mathbf{x} - \mathbf{b}^k \leq \mathbf{u}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& B^k\mathbf{x} - \mathbf{b}^k \geq \mathbf{v}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& \mathbf{u}^k \geq \mathbf{0}, \quad \forall\, k = 0, 1, \ldots, K, \\
& \mathbf{v}^k \leq \mathbf{0}, \quad \forall\, k = 0, 1, \ldots, K.
\end{aligned}
\tag{4.12}
$$

where $p \in [0, 1]^{K+1}$ is a vector with the probability that a given scenario occurs and $\mathbf{A}, \mathbf{B}, \mathbf{\Gamma} \in \mathbb{R}^{+K+1}$ are auxiliary variables.

Now, we need to deal with the hard constraints of the formalisation. Mulvey, Vanderbei and Zenios [134] propose to enforce as hard constraints the constraints belonging to all scenarios, i.e., $\forall\, \omega_k \in \Omega$. We choose to enforce only as hard constraints those constraints belonging to the real scenario ω_0. Constraints belonging to the scenario with variations in the data ω_k for $k = 1, \ldots, K$ are encoded as soft constraints, i.e., preferences that should be fulfilled as much as possible. We impose the hard constraints on ω_0 because it is the "real" scenario, while we impose soft constraints on

the hypothetical scenarios because imposing hard constraints might abruptly change the dedication distribution, yielding infeasible problems. Hence, we add the following set of constraints to (4.12):

$$
\begin{aligned}
& D^k\mathbf{x} \geq \mathbf{d}^k - l^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& T^k\mathbf{x} \leq \tau^k + h^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& l^k, h^k \geq 0, \quad \forall\, k = 1, \ldots, K, \\
& l^0, h^0 = 0,
\end{aligned}
\tag{4.13}
$$

where $\mathbf{l}, \mathbf{h} \in \mathbb{R}^{+K+1}$ are auxiliary variables indicating dissatisfaction with the hypothetical scenarios. We enforce l^0, h^0 to zero, hence encoding such constraints as hard for the "real" data scenario. By adding (4.13) to (4.12), we obtain:

$$
\begin{aligned}
\text{minimise} \quad & \mathbf{p}^T \cdot \Big(\alpha \cdot \mathbf{A} + \beta \cdot \mathbf{B} + \gamma \cdot \mathbf{\Gamma} + \delta \cdot \mathbf{l} + \rho \cdot \mathbf{h}\Big), \\
\text{subject to} \quad & \|A^k\mathbf{x} - \mathbf{t}^k\|_2^2 \leq \mathbf{A}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& \|\mathbf{v}^k\|_2^2 \leq \mathbf{B}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& \|\mathbf{u}^k\|_2^2 \leq \mathbf{\Gamma}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& B^k\mathbf{x} - \mathbf{b}^k \leq \mathbf{u}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& B^k\mathbf{x} - \mathbf{b}^k \geq \mathbf{v}^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& \mathbf{u}^k \geq \mathbf{0}, \quad \forall\, k = 0, 1, \ldots, K, \\
& \mathbf{v}^k \leq \mathbf{0} \quad \forall\, k = 0, 1, \ldots, K, \\
& D^k\mathbf{x} \geq \mathbf{d}^k - l^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& T^k\mathbf{x} \leq \tau^k + h^k, \quad \forall\, k = 0, 1, \ldots, K, \\
& l^k, h^k \geq 0, \quad \forall\, k = 1, \ldots, K, \\
& l^0, h^0 = 0.
\end{aligned}
\tag{4.14}
$$

After discussing our solution approach, we present our experimental evaluation using real-world data in the next section.

4.5 Experimental Analysis

The main objective of our experimental evaluation is to assess the impact of the parameters α, β, and γ on the solutions computed by our approach, in order to provide guidance for the final user (e.g., research institutions) when setting these values so as to obtain the desired behaviour. We refer the reader to Section 4.4.1 for an explanation of the semantics of the parameters α, β, and γ.

In our tests, we consider real-world data regarding the research projects currently active in our research institution. Specifically, from the projects' proposals, we collect data about the set of researchers R, the set of projects P, the set of work packages W, and the required dedication d_w for each $w \in W$. Overall, our dataset involves 6 projects and 27 work packages, whose dedication must be allocated during a planning horizon M of 67 months (Table 4.1).

According to our data, we consider two types of researchers for the set R: *permanent researchers* (e.g., associate and full professors), who are characterised by a higher degree of experience and a higher salary, and *contracted researchers*, whose contract is finite and whose salary is usually lower than that of senior colleagues.

Another crucial information that we collect from our dataset is the budget b_p for each project $p \in P$ and the target dedication vector $\mathbf{t}$. As expected, the budget and the target dedication specified in the proposal of each project were determined to be "just enough" to cover the dedication required by each work package, as a result of a manual planning phase. Nonetheless, since we are proposing and evaluating a tool for project planning, we are interested in testing it in situations where the budget is not so well calibrated, in order to measure its impact on the solution computed by our approach (also in relation to the above-mentioned parameters). To this end,

Table 4.1: Dataset information (anonymous project names).

Project	A	B	C	D	E	F
Work Packages	9	8	1	5	2	2
Researchers	8	5	5	4	6	4
Duration (months)	36	48	36	36	39	39

in addition to the original budget data, which we employ in the so-called *sufficient budget* scenario, we derive two additional scenarios from our real-world data: the *abundant budget* scenario where the original budget of each project is increased by 5% and the *scarce budget* scenario with a budget reduced by 5% for each project. In all these scenarios, we maintain the original target dedication vector $\mathbf{t}$ to better assess the impact of the budget variation.

Since our approach relies on a LS model that can be solved very efficiently (see Section 4.4), we do not show results about the runtime, which is negligible (< 1 second) for the dataset considered here. We employ the CVXPY v1.2.0 Python library to model the constrained LS models in (4.7) and (4.8), and we use the IBM CPLEX v22.1.0.0 solver to solve such models.[3] We run our experiments on a machine with a 12-core 3.70GHz CPU and 64GB of RAM.

4.5.1 Discussion of the Results

To analyse our solutions quantitatively, we will consider the following metrics. First, we define the *budget residual* as the mean percentage deviation between the cost computed by our approach and the budget available for each project. More formally,

$$\text{Budget residual} = 100 \cdot \frac{1}{|P|} \sum_{p \in P} \left| \frac{\min((B\mathbf{x} - \mathbf{x})_p, 0)}{b_p} \right| = \frac{100}{|P|} \sum_{p \in P} \left| \frac{v_p}{b_p} \right|. \tag{4.15}$$

Such a metric is a variation of *Mean Absolute Percentage Error* (MAPE), which is always positive, and in this specific case is bounded within [0,100].[4] Intuitively, a budget residual close to 0% indicates that we are efficiently using the whole budget. On the other hand, a budget residual close to 100%, indicating a minimal expenditure of the budget, is difficult to achieve in practice, since such a budget is required to guarantee a certain dedication of the researchers.

[3] The code is available on: https://github.com/RogerXLera/ResearchProjectPlanning.

[4] MAPE is a well-known metric commonly used in statistics defined such that $\text{MAPE} = 100 \cdot \text{mean}\left(\frac{|A-F|}{|A|}\right)$, where F is the regression value and A is the actual data [11].

Similarly, we consider the *budget excess* defined as

$$\text{Budget excess} = 100 \cdot \frac{1}{|P|} \sum_{p \in P} \left| \frac{\max((B\mathbf{x} - \mathbf{b})_p, 0)}{b_p} \right| = \frac{100}{|P|} \sum_{p \in P} \left| \frac{u_p}{b_p} \right|. \tag{4.16}$$

This metric gives information about the mean percentage of the excess between the computed cost and the budget for each project. While the expression in (4.16) is always positive, it is not upper-bounded, since there is no constraint bounding the exceeding in the soft-constrained formalisation in (4.9). In the case of the hard-constrained formalisation in (4.7), the budget excess is always 0%. Moreover, we consider the *dedication residual* defined as

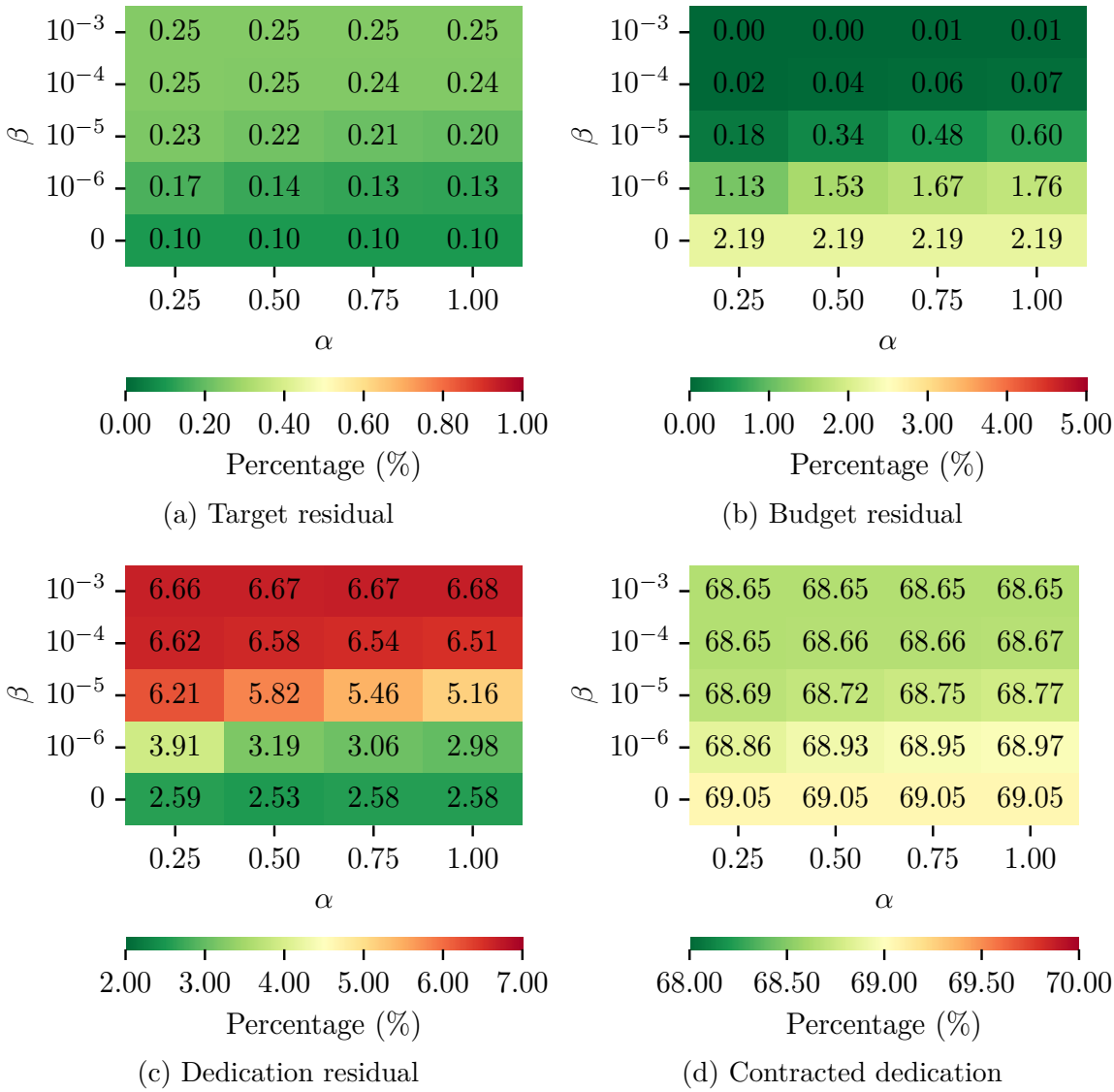
$$\text{Dedication residual} = 100 \cdot \frac{1}{|W|} \sum_{w \in W} \left| \frac{(D\mathbf{x} - \mathbf{d})_w}{d_w} \right| \tag{4.17}$$

to represent the relative percentage difference between the dedication corresponding to the computed solution and the required dedication of each work package. A dedication residual close to 0% indicates that the dedication assigned to work packages is strictly the required one. Finally, we define the *target deviation* as the mean percentage deviation from the target dedication associated with researchers. Formally,

$$\text{Target deviation} = 100 \cdot \frac{1}{|X_P|} \sum_{i=1}^{|X_P|} \left| \frac{(A\mathbf{x} - \mathbf{t})_i}{\max(t_i) - \min(t_i)} \right|. \tag{4.18}$$

The target deviation can also be seen as a variation of the MAPE. We divided the difference between the computed and the target dedication for the range of the target instead of the target dedication. This variation avoids the situation where having small differences between the computed distribution and the target dedication for values of $t_i \to 0$ yield big target deviation values. A target deviation close to 0% indicates that the computed plan is close to the researchers' target preferences. Finally, since our dataset includes permanent and contracted researchers, we also measure the *contracted researchers' dedication*, i.e., the percentage of time slots assigned to contracted researchers with respect to the total.

Our objective is to assess the impact of α, β, γ in the three budget scenarios (i.e., *scarce*, *sufficient*, and *abundant*). More specifically, in all our experiments we con-



(a) Target residual

(b) Budget residual

(c) Dedication residual

(d) Contracted dedication

Figure 4.1: Results in the *sufficient budget* scenario for $\gamma = \infty$ (best viewed in colours).

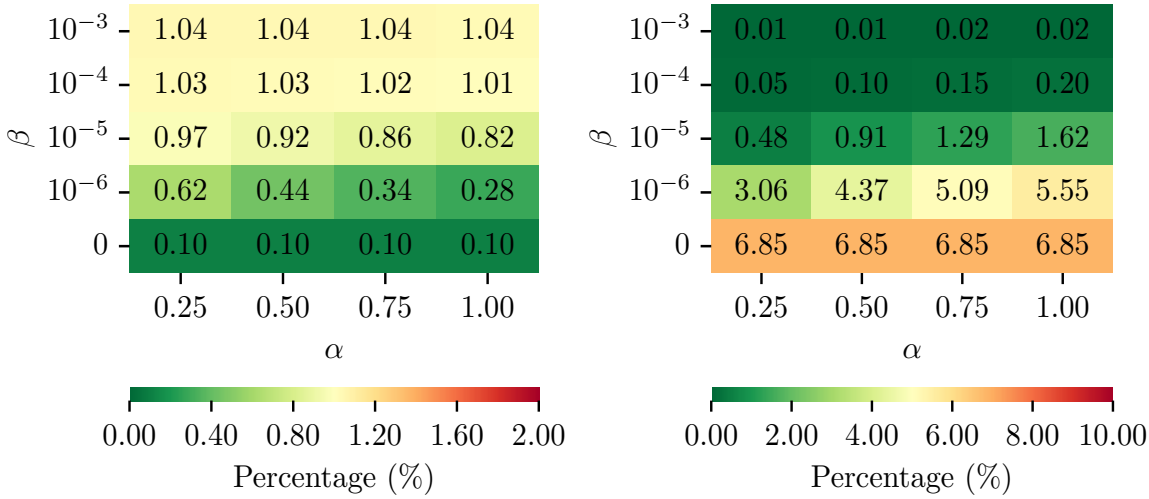
sider $\alpha \in \{0.25, 0.50, 0.75,\ 1.00\}$ and $\beta \in \{0, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$. We consider different values for α and β since they are associated with components with different units of measurement in our objective function (4.8), i.e., $\|A\mathbf{x} - \mathbf{t}\|_2^2$ and $\|\mathbf{v}\|_2^2$, respectively. As for the parameter γ, we consider $\gamma \in \{0, 10^{-5}, \infty\}$ in order to account for an *unconstrained* scenario ($\gamma = 0$), a *soft-constrained* scenario ($\gamma = 10^{-5}$), and a *hard-constrained* scenario ($\gamma = \infty$).

4.5.1.1 Sufficient Budget Scenario

Figure 4.1 shows the results for the *sufficient budget* scenario. Overall, we observe that, for the three different values of γ parameter, the solutions computed by our approach do not vary by a significant amount; therefore, we only report results for $\gamma = \infty$, which accounts for the *hard-constrained* formulation. This behaviour is expected, since in this scenario we consider a budget that has been carefully calibrated to be sufficient for satisfying the dedication required by the work packages (see Section 4.5). Indeed, in this scenario, the resulting target deviation is negligible (the maximum displacement is 0.25%), i.e., our approach matches researchers' dedication preferences quite precisely. Nonetheless, we observe a variation when changing β, i.e., the parameter that controls the importance of efficiently allocating the budget. Indeed, in Figure 4.1b we notice that when increasing β we achieve a budget residual that progressively approaches 0 (i.e., all the budget is efficiently spent). This is achieved by assigning a slightly larger dedication (upper rows of Figure 4.1c) of up to 6.7%. Regarding the dedication of contracted researchers, there is no significant variation for the different configurations of α and β. Finally, we correctly observe that the budget excess is always 0% since the budget is sufficient to satisfy the required dedication in the sufficient budget case.

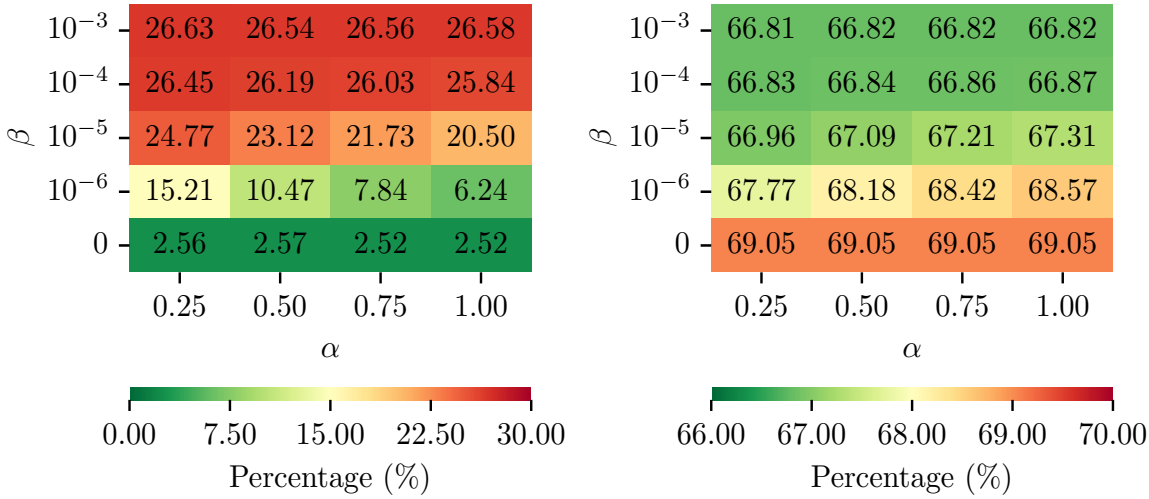
4.5.1.2 Abundant Budget Scenario

In Figure 4.2 we report the results for the *abundant budget* scenario. Here we consider a situation in which the budget is not a limiting factor; hence, we observe that varying γ (i.e., the parameter that controls the importance of not surpassing the available budget) does not impact the behaviour of the algorithm. Thus, similarly to the previous case, we only report results for $\gamma = \infty$. On the other hand, in this scenario, we observe a significant impact of the parameters α and β. First, we notice that for $\beta = 0$ the approach computes a solution equivalent to the *sufficient budget* scenario. In fact, for such parameter configuration, the target deviation (0.10%) and the contracted dedication (69.1%) are equivalent in both scenarios (bottom row of Figures 4.2a and 4.2d). In addition, we notice that the budget residual for $\beta = 0$ varies between the abundant and the sufficient budget scenario (bottom row of Figure



(a) Target residual

(b) Budget residual

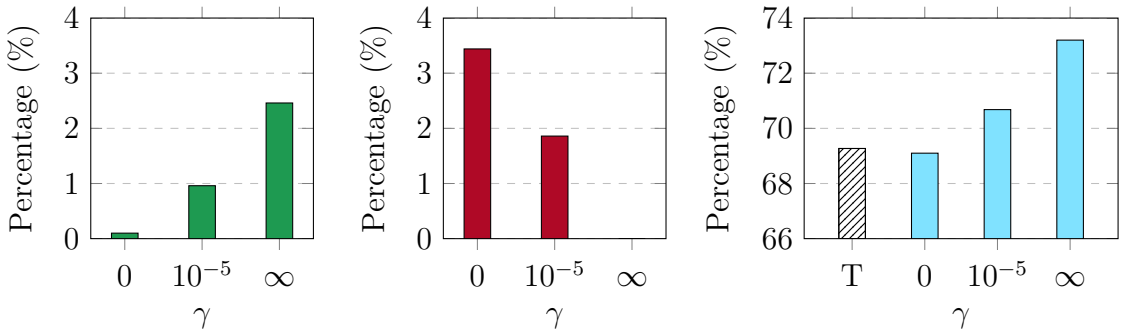


(c) Dedication residual

(d) Contracted dedication

Figure 4.2: Results in the *abundant budget* scenario for $\gamma = \infty$.

4.2b). This increase is due to the fact that the available budget (b) has increased by +5% with respect to the sufficient budget scenario, but the computed cost of the researchers (Bx) is indeed equivalent to the original for $\beta = 0$. This behaviour is due to the absence of any incentive to spend all the budget; thus, the algorithm aims to match the target dedication as much as possible (bottom row of Figure 4.2a). On the other hand, when increasing β, the algorithm gives more importance to spending the available budget (upper rows of Figure 4.2b). This is achieved by assigning a larger overall dedication to the researchers (upper rows of Figure 4.2c), but also by employing more "expensive" (permanent) researchers, since contracted researchers' dedication decreases when increasing β (Figure 4.2d). By doing so, we see that the



(a) Target deviation (b) Budget excess (c) Contracted dedication

Figure 4.3: Target deviation (4.3a), budget excess (4.3b), and comparison between the contracted researchers' dedication with their target (4.3c), for different γ and for $\alpha = 0.25$, $\beta = 0$.

target deviation also gets affected, although the deviation increases only by 1% at most (Figure 4.2a). Also, for the intermediate values of β ($\beta = 10^{-4}, 10^{-5}, 10^{-6}$), when we increase α (i.e., the parameter that controls the importance of matching the target dedication), we obtain a less efficient budget spending but a dedication that closely matches the target one. Again, we do not show the budget excess because it is always 0% due to the same reasons discussed in the sufficient budget scenario.

4.5.1.3 Scarce Budget Scenario

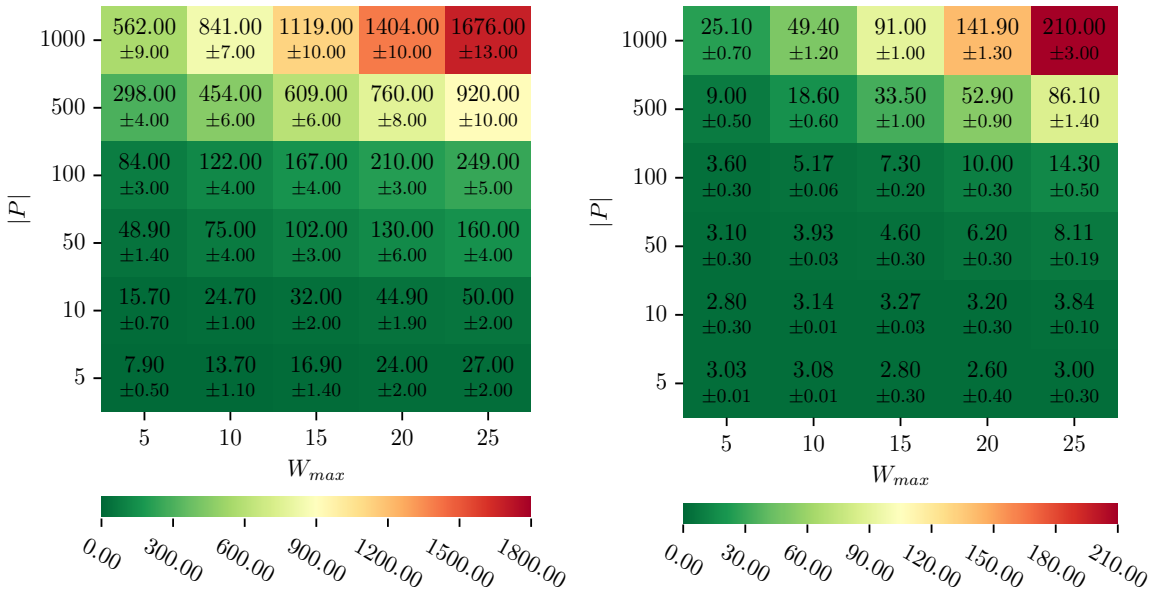
Figure 4.3 shows the results for the *scarce budget* scenario. We immediately observe that budget scarcity has three important consequences. First, as expected, γ now has a significant impact on the results, since the budget is not sufficient to cover the required dedication, and hence, the budget constraint in our formulation has to be violated, i.e., $\|\max(\mathbf{0}, B\mathbf{x} - \mathbf{b})\|_2^2 > 0$. Second, since the available budget will always be entirely allocated (budget residual $\approx 0\%$), β loses any impact on the considered measures. Third, in this scenario, we observe that the dedication surplus is negligible for all parameter configurations (since the algorithm will always allocate the minimum amount of dedication given the budget scarcity). Finally, α has a small impact on the metrics defined for the scarce budget scenario. Given the above-mentioned discussion, we only report results for different values of γ, disregarding the values of α and β.

For $\gamma = 0$ (unconstrained case), we correctly obtain a behaviour equivalent to the

sufficient budget case, similarly to what we observed in the *abundant* scenario for $\beta = 0$. This is due to the absence of any constraint that limits the spending of the budget; hence, the algorithm is free to allocate any amount of money in order to match the target dedication as much as possible. More specifically, we obtain exactly the same target deviation (0.10%) and contracted researchers' dedication (69.1%) as in the other scenarios. Again, the budget excess is almost ($\sim$ 3.5%) due to the reduced budget, although the computed cost is the same as in the other scenarios. For $\gamma = 10^{-5}$ (soft-constrained case), the budget excess is reduced to 1.85%. This is achieved by increasing the dedication of contracted researchers (whose salary is lower than that of permanent researchers), which is reflected in the contracted researchers' dedication and the target deviation by an increase of up to 70.7% and 0.95% respectively. Finally, for $\gamma = \infty$ (hard-constrained case), the algorithm can compute a plan without exceeding the budget, by assigning more dedication to contracted researchers (up to 73.2%). In general, our algorithm opts for higher contracted researchers' dedications when forced to allocate a lower budget, when setting a greater γ.

Overall, our results demonstrate that α, β, and γ have a significant impact on the metrics defined to represent the outcome of the algorithm, depending on the budget availability in relation to the required and target dedication. For our dataset, we show that α and β have a considerable impact on the *abundant* and *sufficient budget* scenario, whereas γ has an important impact in the *scarce budget* scenario, where the budget has to be exceeded by a larger amount.

In this respect, we also considered a *highly scarce budget* scenario, i.e., a scenario where each project budget has been reduced by 10% with respect to the original budget. In this case, the algorithm could not find any feasible solution for $\gamma = \infty$, since the available budget is simply not enough to cover the minimum required dedication. This result confirms the importance of the availability of a soft-constrained formulation, which allows one to obtain feasible solutions while exceeding the budget as little as possible.



(a) Number of researchers (b) Total runtime

Figure 4.4: Fig. 4.4a reports the number of researchers generated with respect to the number of projects $|P|$ and the maximum number of work packages per project W_{max}. Fig. 4.4b reports the total solution runtime. Each cell reports the average and the standard error of the mean over the 10 instances generated for each combination of $|P|$ and W_{max}.

4.5.2 Scalability Evaluation

In the experiments of Section 4.5.1, we have considered real data provided by our institution, for which our approach can compute the optimal solution in less than a second. Therefore, to properly evaluate the scalability of our approach, we consider the methodology discussed hereafter to generate realistic problem instances[5] with a large number of projects and researchers, and then we measure the runtime needed to calculate an optimal solution.

More precisely, we generate realistic RPP instances in two phases. The first phase consists of randomly generating multiple projects, i.e., the starting date and its budget, as well as the number of work packages. For each work package of the project, we randomly assign its required dedication and duration. The second phase involves the generation of the set of researchers that are involved in the projects, along with their availability, cost, and target dedication for a given project. These

[5]RPP is a novel problem, hence no instance generators are readily available in the literature. Here we employ a procedure along the lines of the one by Drexl, Nissen, Patterson and Salewski [62].

features are generated according to probability distributions reported in Table 4.2. We verified that such distributions result in RPP instances whose features resemble the realistic data provided by our institution (which we employed in the experiments in previous sections). We then generate a set of RPP problem instances varying the number of projects $|P|$ and the maximum number of work packages per project, namely W_{max}, to evaluate the scalability of our approach. Specifically, we consider all combinations of $|P| \in \{5, 10, 50, 100, 500, 1000\}$ and $W_{max} \in \{5, 10, 15, 20, 25\}$, generating 10 instances for each pair of parameters, for a total of 300 instances.

Table 4.2: Distributions used for RPP instance generation.

Distribution	Feature
$Exp(1.8 \cdot 10^{-1})$	Start date of project
$U(26.4, 44.3)$	Mean budget per hour per project
$U(1, W_{max})$	Number of work packages per project
$Exp(7.6 \cdot 10^{-4})$	Dedication of work packages
$N(38.1, 4.7)$	Duration of work packages
$U(120, 140)$	Availability of researchers
$U(17.2, 54.0)$	Cost per hour for researchers
$Bern(3.9 \cdot 10^{-1})$	Is a researcher in a work package?
$U(0, \tau(r, m))$	Target dedication for r during m

Notice that, while we generate test instances with a fixed number of projects, the total number of researchers $|R|$ depends on the dedication demand of such projects. Figure 4.4a shows how the total number of researchers varies according to the number of projects $|P|$ and the maximum number of work packages per project W_{max}. That is, more researchers are generated to fulfil the demand required by a higher number of projects.

Figure 4.4b reports the total runtime of our approach (including both the runtime required for compiling the norm approximation problem and the runtime required for solving it) with respect to the number of projects and the maximum number of work packages per project. Results show that instances with up to 100 projects are solved in at most a few tens of seconds. Moreover, for (arguably unrealistic) very large instances considering 1000 projects, a maximum of 25 work packages per project, and more than 1600 researchers, our approach is able, on average, to compute the optimal solution in 210 seconds. This capability of scaling to very large

instances is also thanks to the employment of *sparse matrices* to represent A, B, D, and T matrices, which allows us to achieve a dramatic reduction (up to 5 orders of magnitude) in terms of memory consumption with respect to the standard dense matrix representation.

Overall, results show that our approach makes it viable to compute optimal solutions even for very large RPP instances with realistic features.

4.5.3 Experiments with Uncertainty

Now that we have assessed the impact of the parameters and the scalability of our approach, we aim to assess the impact of uncertainty in our approach. To do so, we evaluate our approach described in section 4.4.2 by using the metrics described in section 4.5.1 and by generating possible hypothetical scenarios Ω. The rest of this section is organised as follows. In section 4.5.3.1, we provide more details on our methodology for the experiments concerning uncertainty. In section 4.5.3.2 we discuss our results.

4.5.3.1 Experimental methodology

To evaluate our approach with uncertainty, we need to generate the hypothetical scenarios in the set $\Omega = \{\omega_0, \omega_1, \ldots, \omega_K\}$, i.e., the variations of the initial data ω_0. We consider $K = 10$. Notice that such hypothetical scenarios represent the possibility of having unexpected events that change the initial conditions of the projects. For the sake of simplicity, given that these changes could affect all the input data of our model (the assignment of a new project, the renounce or involvement of a researcher, the changes in the condition of a project, i.e., variations in the budget, dedication, or deadline, among others), we choose to consider one of the most common situations tackled in real-world project planning, i.e., the addition of new projects during the planning horizon. Each hypothetical scenario ω_k has the same initial projects as ω_0 plus the hypothetical new projects generated via the methodology described in

the supplementary material. The number of additional projects is uniformly chosen between 1 and 3.[6]

Note that the same methodology can be applied to other above-mentioned situations that introduce changes in the input of the problem.

4.5.3.2 Results

As mentioned before, our objective is to evaluate the impact of considering uncertainty in the data. Hence, we aim to compare our approach with and without uncertainty. To do so, we run the experiments for the same values of α and β parameters discussed in Section 4.5.1. Moreover, as we are generating random hypothetical scenarios, we rerun them for different seeds. For the sake of brevity, we just show the results for the sufficient budget scenario for $\gamma = \infty$, i.e., the hard-constrained formalisation, in Figure 4.5.

Figure 4.5a shows that β and α have a significant impact on the target deviation. Indeed, by comparing the results of the approach with and without uncertainty, we observe that the maximum value of the target deviation for the uncertainty approach is $12.8 \pm 1.2\%$, while for the formalisation without uncertainty is 0.25% (2.5% for the scarce budget scenario). This result is expected for the uncertainty approach, as the algorithm pursues a trade-off between satisfying the preferences of the researchers and saving dedication for the projects of the hypothetical scenarios. We also observe that as we increase α, we reduce the target deviation, and as we increase β, we reduce the budget residual (Figure 4.5b). This behaviour is similar to the case without uncertainty, as expected.

We also observe a significant reduction in the dedication residual (Figure 4.5c) compared to the approach without uncertainty (Figure 4.1c). More specifically, the maximum value of the dedication residual for the approach considering uncertainty is 2.4% while for the approach not considering uncertainty is 6.7%.

Finally, in Figure 4.5d we observe a significant drop in the contracted researchers'

[6]We choose this number of additional projects as we consider it appropriate for our dataset, which contains 6 projects. For larger institutions, a larger value would be considered.

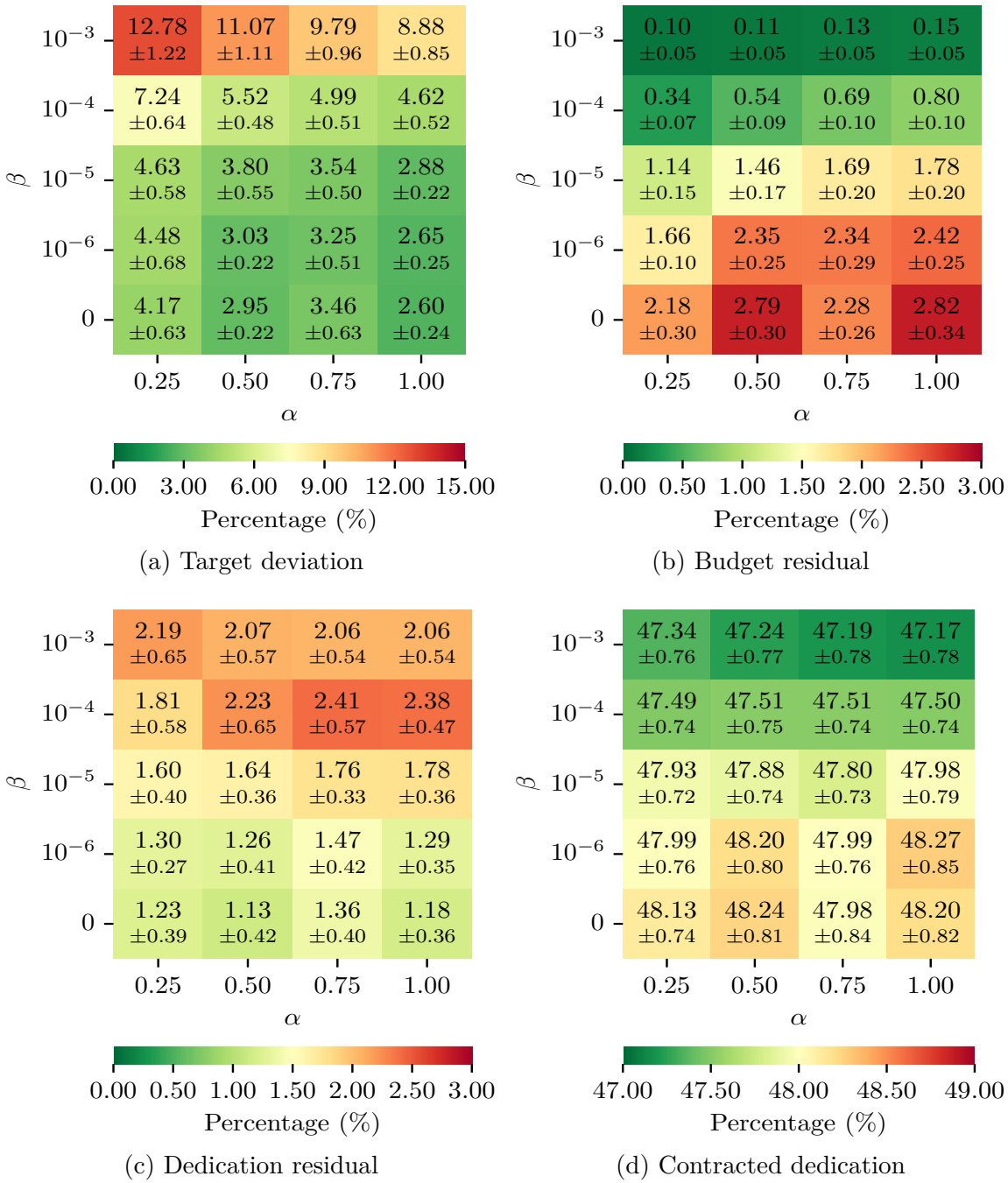


Figure 4.5: Results in the *sufficient budget* scenario for $\gamma = \infty$ considering uncertainty.

dedication compared to the approach without uncertainty (48.2% vs 69.1%). This difference can be explained as follows. As we discuss in section 4.5.1, contracted researchers usually execute most of the workload compared to "permanent" researchers. Hence, to account for the possible addition of new hypothetical projects, our approach balances the dedication of permanent and contracted researchers. This allows

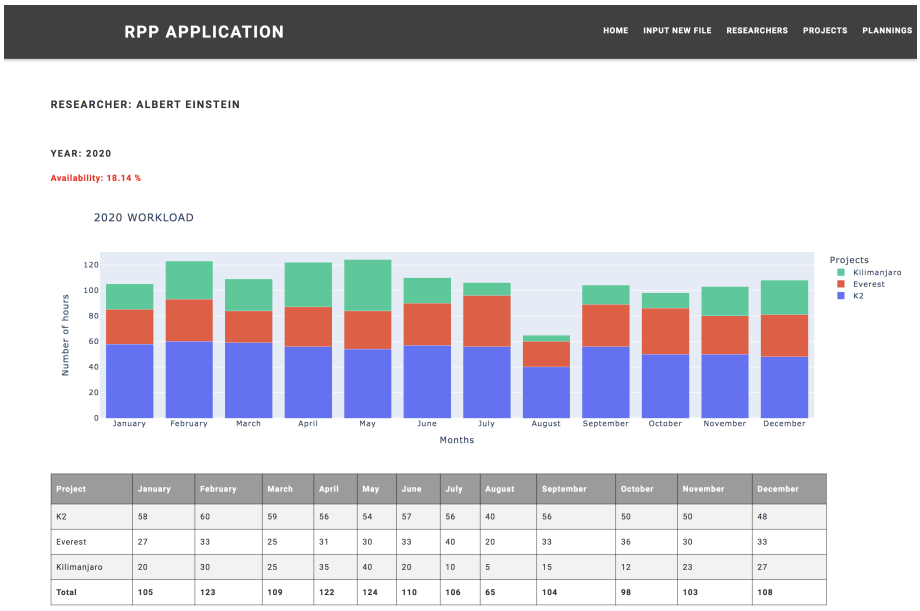
contracted researchers to be involved in new projects by reducing their dedication to the current projects and increasing the dedication of the permanent researchers.

Overall, we observe that our approach is capable of accounting for uncertainty by balancing the dedication of researchers.

4.6 Conclusions

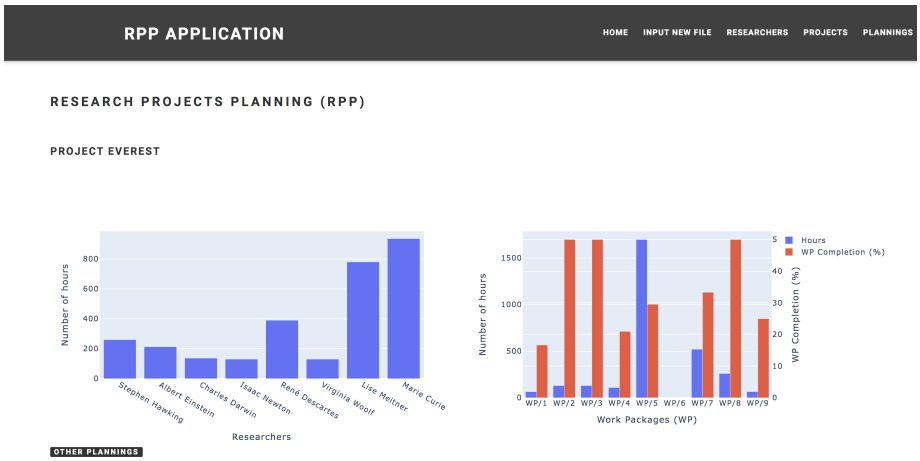
We considered the RPP problem, a central task routinely tackled by research institutions when planning the dedication of a set of researchers involved in a set of research projects. We put forward a norm approximation approach to compute plans for the user (e.g., the administrative staff) that can account for uncertainty, and we evaluated it on a real-world dataset provided by our research institution. Results showed that our approach can compute optimal solutions that reflect the preferences of the user in real-time, including instances that are larger than the ones managed by an average-sized research institution, as shown in Section 4.5.2. Our approach has been implemented as part of a planning prototype (shown in Figure 4.6) that is being tested in our institution.

In summary, this chapter provides an answer to research question **Q4**: How can we formalise the Research Project Planning problem as a MIP while accounting for uncertainty? We achieved this by formally defining the RPP problem as an ℓ_2-regression problem (which results in a MIQP problem), and employing the *Stochastic Robust Approximation* to account for uncertainty.



Project	January	February	March	April	May	June	July	August	September	October	November	December
K2	58	60	59	56	54	57	56	40	56	50	50	48
Everest	27	33	25	31	30	33	40	20	33	36	30	33
Kilimanjaro	20	30	25	35	40	20	10	5	15	12	23	27
Total	**105**	**123**	**109**	**122**	**124**	**110**	**106**	**65**	**104**	**98**	**103**	**108**

(a) Researcher planning recommendation



(b) Project planning recommendation

Figure 4.6: Fig. 4.6a reports the number of researchers generated with respect to the number of projects $|P|$ and the maximum number of work packages per project W_{max}. Fig. 4.6b reports the total runtime required to compile and solve RPP. Each cell reports the average and the standard error of the mean over the 10 instances generated for each combination of $|P|$ and W_{max} (best viewed in colours).

Chapter 5

Bachelor's Degree Planning

5.1 Introduction

The United Nations' 2030 Agenda for Sustainable Development includes quality education as a Sustainable Development Goal. This goal calls for the development of technologies and tools that make education management and provision more equitable, inclusive, and *personalised* [P8]. Indeed, personalisation in education has recently been the focus of significant research efforts within the scientific community, mainly from the perspective of *learning analytics* [176, 74, 188]. Learning analytics research focuses on measuring, collecting, analysing, and reporting data about learners and their contexts to understand and optimise learning and the environments in which it occurs. In this context, one of the most active and relevant areas of research concerns *Educational Recommender Systems* (EdRecSys) [61]. One of the main topics of interest for EdRecSys research has been recommending university courses since planning the set of courses to study during a university degree is one of the most crucial tasks that impacts the future professional life of a student. Indeed, in a recent survey carried out by the educational platform Forage,[1] 1000 U.S. university students were interviewed about several aspects of education in relation to their professional prospects. One of the most important findings of this survey was that 71% of the students evaluated as *extremely important* to "identify the most important skills [needed] to land [their] dream job", but only 39% considered the "tools or information to do this" as adequate. Along these lines, developing an auto-

[1] https://www.theforage.com/blog/news/forage-career-readiness-survey, accessed on 23/05/2024.

mated approach to guide students throughout their educational pathway toward the acquisition of the skills required by their preferred job(s) is key.

Of course, the problem of planning a set of activities within a time horizon has also been extensively studied in the planning and scheduling literature. However, despite the abundance of works in these fields (especially on the RCPSP [142, 101, 64] introduced in Chapter 2), to the best of our knowledge, the problem of computing a degree path that allows a student to acquire the skills required by their desired job(s) has not been studied yet.

Against this background, in this chapter, we propose an automated solution (outlined in Figure 5.1) able to provide recommendations to students during the completion of an undergraduate university degree by solving a novel planning problem, a problem that we denote as *Degree Planning Problem* (DPP). Specifically, the goal of the DPP is to compute degree plans that achieve the set of skills that best cover the requirements of the job(s) preferred by a student, i.e., maximising the so-called *job affinity*. More precisely, we make the following contributions to the state-of-the-art:

- We propose a formal definition of the DPP as a MIP problem. Solving such an optimisation problem yields the degree paths that maximise the "coverage" of the professional skills required by a student's desired job(s). While some aspects of our model are based on our real-world test case, it is straightforward to adapt it to other university curricula without changing the proposed solution method.

- We present a non-trivial linearisation of our optimisation problem so that we can solve it with standard optimisation tools.

- We show, through several use cases, that our approach can be a valuable decision-support tool for (i) students planning their academic pathways (even when they are uncertain about their desired job or they have already initiated their academic career) and (ii) curriculum designers assessing how well a university's degrees satisfy the skill requirements of the job market. With this aim, we employ real-world data from the Bachelor's Degree of Information

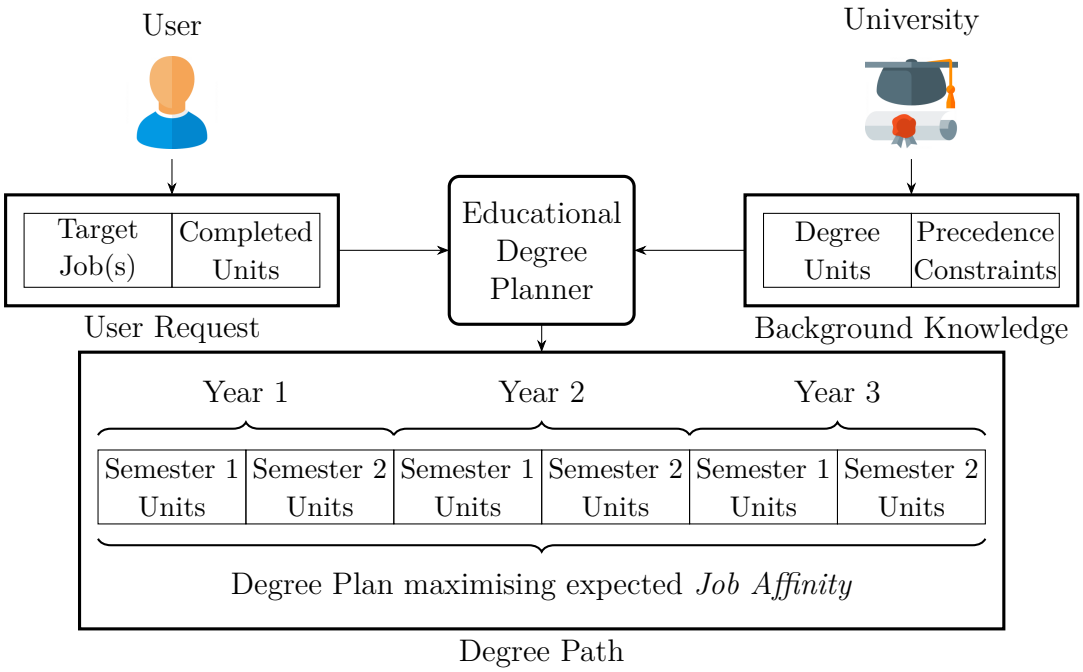


Figure 5.1: Overview of our automated solution approach. We refer the reader to Sections 5.2 and 5.3 for the technical details.

and Communications Technology of Western Sydney University and the *Skills Framework for the Information Age* (SFIA) dataset [29].

Organisation. In Section 5.2, we discuss the formalisation of the DPP. We show how we solve the DPP by casting it as a MIP problem in Section 5.3. In Section 5.4, we empirically analyse real-world data considering different potential use cases. In Section 5.5, we discuss the real-world applicability of our approach and propose some extensions to other university curricula. Finally, Section 5.6 concludes the chapter and discusses future research directions.

5.2 Formalising the Degree Planning Problem as an MIP

The objective of the DPP is planning a feasible degree path (i.e., one that complies with the rules imposed by a university)[2] for a student to maximize the level of skills required by the job (or a set of jobs) preferred by the student.

5.2.1 Basic Definitions

First, we consider a sequence of semesters $\mathcal{L} = \langle l_1, \ldots, l_n \rangle$. Each semester is associated with a season of the year in the set $\Gamma = \{\text{winter, spring, summer, autumn}\}$.[3] To represent such an association, we define function $\rho : \mathcal{L} \to \Gamma$, which assigns a season to each semester. We also consider a set of skills $\mathcal{S} = \{s_1, \ldots, s_w\}$. Then, we denote a university subject as *unit*, representing the fundamental element used to build a student's degree plan.

Definition 5.2.1 (Unit)**.** *A unit is a tuple $\langle S, c, \sigma, \gamma \rangle$, where $S \subseteq \mathcal{S}$ is the set of skills that a student obtains when coursing the unit, $c \in \mathbb{N}$ is the number of credits rewarded to a student completing the unit, $\sigma : S \to [1..lev_{max}]$ is a function that indicates the level for a given skill acquired when completing a unit (lev_{max} denotes the maximum skill level that can be achieved) and $\gamma : \mathcal{L} \to \{\top, \bot\}$ is a Boolean function that indicates whether a unit is offered during a given semester l_i (i.e., $\gamma(l_i) = \top$) or it is not offered (i.e., $\gamma(l_i) = \bot$).*[4]

Henceforth, we denote the set of all units as $\mathcal{U}$.[5] We refer to the components of a

[2] Some aspects of our model (e.g., the major definition, the precedence relations, or the credits specification restrictions) are based on our real-world test case. Nonetheless, it is straightforward to adapt it to other university curricula without changing the proposed solution method.

[3] Notice that universities usually organise courses during two semesters: autumn and spring. However, some universities, such as the one considered in our test case, offer shorter semesters during summer vacations. Thus, for the sake of generality, we consider all the seasons.

[4] Universities have limited resources. Hence, it is reasonable to assume that not all units will be offered during all semesters.

[5] In this chapter, we are taking the perspective of a single student, hence we assume that each unit that appears in the set $\mathcal{U}$ has the capacity of hosting such a student, i.e., we do not enforce any capacity constraint. With minimal changes to our model, one could easily consider the problem of recommending degree plans to *several students*, in which capacity constraints could be enforced straightforwardly. Nonetheless, by taking such a multi-student perspective, the problem would

unit $u \in \mathcal{U}$ by subscripts (e.g., S_u and c_u denote the skills and number of credits of unit u, while σ_u and γ_u refer to its skill level and offer functions).

It is common for degrees' syllabi to have requisites to course some subjects, as students may be required to possess some basic knowledge before pursuing a given subject. Hence, we introduce precedence relationships that hold between units. Following the structure of the real-world syllabi, we consider two types of precedence relationships:

- AND: a unit requires that *all* its prerequisite units have been completed.
- OR: a unit can be undertaken by a student if *at least one* of its prerequisite units has been completed.

Formally, we capture these two types of precedence relationships as follows.

Definition 5.2.2 (AND precedence)**.** *The AND precedence graph is a directed acyclic graph* $(\mathcal{U}, E_{and})$*, where* $E_{and} \subseteq \mathcal{U} \times \mathcal{U}$*. If* $(u, u') \in E_{and}$*, we say that unit* u *is a prerequisite of* u'*.*

Definition 5.2.3 (OR precedence)**.** *The OR precedence between units is a function* $or : 2^{|\mathcal{U}|} \to \mathcal{U}$*. If* $or(U) = u'$*, with* $U \subseteq \mathcal{U}$*, we say that all units in* U *are an OR prerequisite of* u'*.*

Notice that a unit can have multiple precedence relations of both the previously defined types.

In addition, we consider a subset of units $\mathcal{U}^{core} \subset \mathcal{U}$ called *core units*, which are mandatory for a student during their degree. Moreover, each student must pursue at least one *major*. A *major* is a specialisation within the degree. We formally define a major as follows.

Definition 5.2.4 (Major)**.** *A major is a set of units* $M \subset \mathcal{U}$ *such that : (i) there is a set of core units* $M^{core} \subseteq M$ *to be completed; and (ii) there is a set of eligible units* $M^{elig} \subset M$ *distributed into subsets of units* $SM^{elig} \subseteq M^{elig}$*, such that a student has to choose a certain number of units* $n_{SM^{elig}}$ *for each* SM^{elig}*.*

resemble more of an *assignment* problem of students to units, which is not our purpose here. For this reason, we leave this exercise out of this thesis.

We denote the set of all majors by $\mathcal{M}$. Moreover, we introduce the notion of *credits specification* as a tuple $\langle c_T, \mathbf{d} \rangle$ where $c_T \in \mathbb{N}$ is the total number of credits to complete to obtain a bachelor and $\mathbf{d} \in \mathbb{N}^{|\mathcal{L}|}$ is a vector containing the number of credits that a student has to complete during a semester.

Finally, we define the notion of *job*.

Definition 5.2.5 (Job)**.** *A job j is a tuple $\langle S_j, \sigma'_j \rangle$, where $S_j \subseteq \mathcal{S}$ is the set of skills required to take on the job, and $\sigma'_j : S_j \to [1..lev_{max}]$ is a skill requirement function defining the minimum level required per skill to undertake the job.*

5.2.2 Objective of the Degree Planning Problem

We now characterise the objective of the DPP. Specifically, the DPP aims at finding the degree plan whose units help a student achieve the skills required by the job(s) pursued by the student. Formally, we define a target as follows.

Definition 5.2.6 (Target)**.** *Given a set $\mathcal{J}$ of jobs defined according to Definition 5.2.5, the target $\mathbf{t} \in \mathbb{N}^{|\mathcal{S}|}$ is a vector such that t_s represents the target level required by each skill s required by a job $j \in \mathcal{J}$. Hence, $t_s = \max_{j \in \mathcal{J}} \sigma'_j(s)$. We assume $t_s = 0$ for the skills that are not required.*

Therefore, given a student who pursues the target $\mathbf{t}$, solving the DPP amounts to finding a degree plan as an *allocation of units to semesters*, such that: (i) the skills acquired by the student after completing the degree plan bring them as close as possible to the target, i.e., the degree plan maximizes a similarity measure (*job affinity*) between the achieved skills and the skills required by the job(s), and (ii) the student is not penalized for acquiring further skills not required by the job(s).[6] Therefore, the DPP amounts to solving

$$\text{minimize} \left(\sum_{s \in \mathcal{S}} \left| \min \left(\max_{u,l} \left(\sigma_u(s) \cdot x_{u,l} \right) - t_s, 0 \right) \right|^p \right)^{1/p}, \tag{5.1}$$

[6]Not penalising further skills is reasonable because degrees are conceived to provide a wider range of skills than those strictly required. Furthermore, from the point of view of a student, acquiring further skills than required is an advantage: they make the student more competitive in the market, and they are valuable to change or evolve their professional career.

where the binary decision variable $x_{u,l}$ denotes whether the unit u has been scheduled or not during semester l, and $\sigma_u(s)$ is the level of skill s for the unit corresponding to the variable $x_{u,l}$. The $(\sum |\cdot|^p)^{1/p}$ notation represents a p-norm that measures the *distance* between the vector of acquired skills and the target, where $p \in [1, \infty]$ is a metric parameter.

5.2.3 Constraints

After defining the objective of DPP, we now specify a number of feasibility constraints that must be included to comply with university rules. We remark that these constraints are specific to the degree structure of Western Sydney University. Nonetheless, adapting our formalisation to other scenarios and implementing additional constraints by adopting a similar methodology is easy.

1. A unit cannot be scheduled more than once in a degree plan. We achieve this by defining a matrix $B \in \mathbb{N}^{|\mathcal{U}| \times |X|}$ where each row corresponds to a unit $u \in \mathcal{U}$ and each column to a decision variable $x_{u,l} \in X$ such that

$$B_{u,(u',l)} = \begin{cases} 1, & \text{if } u = u', \\ 0, & \text{otherwise,} \end{cases} \tag{5.2}$$

   and by imposing the linear constraint

$$B\mathbf{x} \leq \mathbf{1}, \tag{5.3}$$

   where $\mathbf{x} \in \{0, 1\}^{|X|}$ is the vector of decision variables, and $\mathbf{1} \in \mathbb{N}^{|\mathcal{U}|}$ is a vector with all components equal to 1.

2. As mentioned in the basic definitions, some units are not offered during some semesters. We fulfil such constraint by defining a diagonal matrix $P \in \mathbb{N}^{|X| \times |X|}$

where each row corresponds to a decision variable $x_{u,l} \in X$ and each column to a decision variable $x_{u',l'} \in X$ such that

$$P_{(u,l),(u',l')} = \begin{cases} 1, & \text{if } u = u' \text{ and } l = l' \text{ and } \gamma_u(l) = \perp, \\ 0, & \text{otherwise,} \end{cases} \tag{5.4}$$

and by imposing the linear constraint

$$P\mathbf{x} = \mathbf{0}, \tag{5.5}$$

where $\mathbf{0} \in \mathbb{N}^{|\mathcal{P}|}$ is a null vector.

3. The degree plan has to fulfil a total number of credits. We achieve this by defining a vector $\mathbf{c} \in \mathbb{N}^{|X|}$ where each position corresponds to the number of credits assigned to the unit associated with variable $x_{u,l} \in X$ such that $\mathbf{c}_{(u,l)} = c_u$, and by imposing the linear constraint

$$\mathbf{c}^T\mathbf{x} = c_T, \tag{5.6}$$

where c_T is the total number of credits.

4. In addition, during each semester, we have to assign a certain number of credits. We achieve this by defining a matrix $D \in \mathbb{N}^{|\mathcal{L}| \times |X|}$ where each row corresponds to a semester $l \in \mathcal{L}$ and each column to a decision variable $x_{u',l'} \in X$ such that

$$D_{l,(u',l')} = \begin{cases} c(u'), & \text{if } l = l', \\ 0, & \text{otherwise,} \end{cases} \tag{5.7}$$

and by imposing the linear constraint

$$D\mathbf{x} = \mathbf{d}, \tag{5.8}$$

where $\mathbf{d} \in \mathbb{N}^{|\mathcal{L}|}$ is a vector whose components are the total amount of credits per semester.[7]

5. Now, we introduce the constraints related to the precedence relations between units. As mentioned in Section 5.2.1, two types of precedence relations exist: AND and OR. However, we can formalise the AND precedence relation as a specific case of OR precedence, where the set of prerequisites contains one unit for each AND precedence relation. Consequently, we denote $\mathcal{R}$ as the set of all precedence relations. To formally code precedence relations we define matrices $R, r \in \mathbb{N}^{|\mathcal{R}|\cdot|\mathcal{L}|\times|X|}$ where each row corresponds to a precedence relation $or(U) \in \mathcal{R}$ and a semester $l \in \mathcal{L}$ and each column to a decision variable $x_{u,l} \in X$ such that

$$R_{(or(U),l'),(u,l)} = \begin{cases} 1, & \text{if } u \in U \text{ and } l' > l, \\ 0, & \text{otherwise,} \end{cases} \tag{5.9}$$

$$r_{(or(U),l'),(u',l)} = \begin{cases} 1, & \text{if } u' = or(U) \text{ and } l' = l, \\ 0, & \text{otherwise.} \end{cases} \tag{5.10}$$

Then, we impose

$$R\mathbf{x} \geq r\mathbf{x}. \tag{5.11}$$

Concretely, this linear constraint enforces assigning one or more prerequisites $u \in U$ of unit $u' = or(U)$ in previous semesters than when this unit is assigned, in case u' is scheduled. If u' is not scheduled, the constraint does not enforce assigning any prerequisite.

6. All core units (i.e., $u \in \mathcal{U}^{core}$) must be scheduled. We achieve this by defining

[7] Notice that (5.6) is self-contained in (5.8). However, we impose both constraints in case a user wants to impose a lower or upper bound threshold for constraint (5.8) such that $d_{min} \leq D\mathbf{x} \leq d_{max}$, which is a common real-world case. In that case, constraint (5.6) is needed.

a matrix $E \in \mathbb{N}^{|\mathcal{U}^{core}| \times |X|}$ where each row corresponds to a unit $u \in \mathcal{U}^{core}$ and each column to a decision variable $x_{u',l'} \in X$ such that

$$E_{u,(u',l')} = \begin{cases} 1, & \text{if } u = u', \\ 0, & \text{otherwise.} \end{cases} \tag{5.12}$$

and by imposing the linear constraint

$$E\mathbf{x} = \mathbf{1}, \tag{5.13}$$

where $\mathbf{1} \in \mathbb{N}^{|\mathcal{U}^{core}|}$ is a vector of 1.

7. Finally, a student has to complete at least one major. To enforce this constraint, we define three sub-constraints. First, we guarantee the assignment of at least one major by imposing the linear constraint

$$\mathbf{1}^T\mathbf{v} \geq 1, \tag{5.14}$$

where the auxiliary variable $v_M = 1$ indicates that the plan will contain the requirements to complete major M. Then, we enforce scheduling all core units of the major we assign, namely M^{core}. We achieve this by defining a matrix $F \in \mathbb{N}^{|\mathcal{U}_{\mathcal{M}}^{core}| \times |X|}$, where $\mathcal{U}_{\mathcal{M}}^{core}$ denote the set of core units of all majors, and each row corresponds to a core unit $u \in M^{core}$ for a major $M \in \mathcal{M}$ and each column to a decision variable $x_{u',l'} \in X$ such that

$$F_{u,(u',l')} = \begin{cases} 1, & \text{if } u = u', \\ 0, & \text{otherwise.} \end{cases} \tag{5.15}$$

We then define a matrix $f \in \mathbb{N}^{|\mathcal{U}_{\mathcal{M}}^{core}| \times |\mathcal{M}|}$ where each each row corresponds to

a core unit $u \in M^{core}$ for a major $M \in \mathcal{M}$ and each column to a decision variable v_M such that

$$f_{u,v_M} = \begin{cases} 1, & \text{if } u \in M^{core}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.16}$$

Then, we impose the linear constraint

$$F\mathbf{x} \geq f\mathbf{v}. \tag{5.17}$$

Concretely, this constraint enforces scheduling core units of a major when this is coursed. Otherwise, the units can be undertaken, but it is not a requirement. Furthermore, we consider that majors have different subsets of elective units, namely $\mathcal{SM}^{elig}$. By $\mathcal{SM}^{elig}$, we denote the set of elective unit subsets. In order to enforce that at least $n_{SM^{elig}}$ units $u \in SM^{elig}$ are assigned, we define a matrix $G \in \mathbb{N}^{|\mathcal{SM}^{elig}| \times |x|}$ where each row corresponds to a subset of electives units $SM^{elig} \in \mathcal{SM}^{elig}$ and each column to a decision variable $x_{u',l'} \in X$ such that

$$G_{SM^{elig},(u,l')} = \begin{cases} 1, & \text{if } u \in SM^{elig}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.18}$$

Then, we define a vector $g \in \mathbb{N}^{|\mathcal{SM}^{elig}| \times |\mathcal{M}|}$ where each row corresponds to a subset of electives units $SM^{elig} \in \mathcal{SM}^{elig}$ and each column to a decision variable $x_{u',l'} \in X$ such that

$$g_{SM^{elig},v_M} = \begin{cases} n_{SM^{elig}}, & \text{if } SM^{elig} \text{ is associated with } M, \\ 0, & \text{otherwise,} \end{cases} \tag{5.19}$$

where $n_{SM^{elig}}$ is the minimum number of elective units completed per subset SM^{elig}. Then, we impose

$$G\mathbf{x} \geq g\mathbf{v}. \tag{5.20}$$

Concretely, this linear constraint enforces that the required number of elective

units is scheduled if a given major M is pursued. Otherwise, elective units can be scheduled, but it is not a requirement.

5.2.4 Completed Units

In our formalisation discussed so far, we assume that (i) no unit had been completed and (ii) there were no changes in the academic curriculum along the degree plan. However, in practice, students might have completed some units and want to receive a new plan for the remaining part of their bachelor's degree. The reasons for computing a new plan might be multiple: students might want to target a different desired job(s), conditions in the curriculum might change, and students might need to compute a new valid degree plan (some units might not be offered anymore, new units might be offered, skill levels acquired by units might change, prerequisites of the units have changed, etc.), or the required skills for a given job might change. Thus, we must consider the scenario where students are in the middle of the degree and have completed some units. To do so, we first formally define the concept of a completed unit.

Definition 5.2.7 (Completed units)**.** *A completed unit is a tuple $\langle u, l\rangle$ where u is the completed unit and l is the semester when the unit has been completed. We denote the set of all completed units by $\mathcal{Q}$.*

Then, we define a matrix $Q \in \mathcal{N}^{|\mathcal{Q}|\times|X|}$ where each row corresponds to a completed unit $\langle u, l\rangle \in \mathcal{Q}$ and each column to a variable $x_{u',l'} \in X$ such that

$$Q_{(u,l),(u',l')} = \begin{cases} 1, & \text{if } u = u' \text{ and } l = l', \\ 0, & \text{otherwise.} \end{cases} \tag{5.21}$$

Finally, we consider completed units by imposing the linear constraint

$$Q\mathbf{x} = \mathbf{1}. \tag{5.22}$$

As a conclusion, by combining the objective function in (5.1) with the collection of

constraints (5.3), (5.5), (5.6), (5.8), (5.11), (5.13), (5.14), (5.17), (5.20), and (5.22), we obtain the following MIP formalization of the DPP:

$$\begin{aligned} \text{minimise} \quad & \left(\sum_{s \in \mathcal{S}} \left| \min \left(\max_{u,l} \left(\sigma_u(s) \cdot x_{u,l} \right) - t_s, 0 \right) \right|^p \right)^{1/p}, \\ \text{subject to} \quad & B\mathbf{x} \leq \mathbf{1}, \quad P\mathbf{x} = \mathbf{0}, \quad \mathbf{c}^T\mathbf{x} = c_T, \quad D\mathbf{x} = \mathbf{d}, \\ & R\mathbf{x} \geq r\mathbf{x}, \quad E\mathbf{x} = \mathbf{1}, \quad \mathbf{1}^T\mathbf{v} \geq 1, \\ & F\mathbf{x} \geq f\mathbf{v}, \quad G\mathbf{x} \geq g\mathbf{v}, \quad Q\mathbf{x} = \mathbf{1}. \end{aligned} \tag{5.23}$$

5.3 Solving DPP by Linearising the Argument of the p-norm

We are now ready to detail our approach to solve (5.23). With that aim, we show how to cast the MIP formulation of the DPP as a NMP problem with a linear argument within a p-norm objective function. Specifically, our goal is to linearize the argument of the p-norm, i.e., $\min \left(\max_{u,l} \left(\sigma_u(s) \cdot x_{u,l} \right) - t_s, 0 \right)$, in the objective function of (5.23). This will allow us to solve the DPP as one of the manageable optimisation problems discussed in Section 2.1.2.

In detail, we discuss how we reformulate the DPP in two steps. First, we consider the goal of ensuring that the skills acquired by a student are as close as possible to their target job(s). Hence we simplify (5.1) to

$$\text{minimise} \quad \left(\sum_{s \in \mathcal{S}} \left| \max_{u,l} \left(\sigma_u(s) \cdot x_{u,l} \right) - t_s \right|^p \right)^{1/p}, \tag{5.24}$$

for the sake of clarity. Second, we consider the goal of not penalising the acquisition of non-compulsory skills for the target job(s), whose objective function is (5.1). To address the first goal, we introduce a new set of auxiliary, binary decision variables $y_{s,lev}$ with $s = 1, \ldots, |\mathcal{S}|$ and $lev = 1, \ldots, lev_{max}$. Given a degree path (obtained by setting some decision variables $x_{u,l}$ to 1), setting $y_{s,lev}$ to 1 indicates that the degree plan achieves at least level lev for skill s. Furthermore, on the one hand, we introduce a matrix of binary values $A \in \mathbb{N}^{lev_{max} \cdot |\mathcal{S}| \times |X|}$ that encodes the skill levels

acquired by each unit. Thus, by setting $A_{(s,lev),(u,l)}$ to 1, we indicate that if unit u is allocated to semester l, it helps a student achieve at least level lev for skill s (this is because $\sigma_u(s) \geq lev$). Otherwise, we set $A_{(s,lev),(u,l)}$ to 0. On the other hand, we introduce another matrix of binary values $K \in \mathbb{N}^{|\mathcal{S}| \times lev_{max} \cdot |\mathcal{S}|}$ to encode that decision variables $\{y_{s,lev} \mid lev = 1, \ldots, lev_{max}\}$ refer to skill s. Thus, we set $K_{s',(s,level)}$ to 1 when $s = s'$, and to 0 otherwise.

We are now ready to formulate a version of the DPP that only addresses the goal of selecting a degree path *as close as possible to* the target job(s):

$$\text{minimise} \quad \|K\mathbf{y} - \mathbf{t}\|_p, \tag{5.25a}$$

$$\text{subject to} \quad A\mathbf{x} \leq \Lambda \cdot \mathbf{y}, \tag{5.25b}$$

$$A\mathbf{x} \geq \Lambda \cdot (\mathbf{y} - \mathbf{1}) + \mathbf{1}. \tag{5.25c}$$

and also subject to the side constraints in the DPP defined in the previous section, i.e., constraints (5.3), (5.5), (5.6), (5.8), (5.11), (5.13), (5.14), (5.17), (5.20), and (5.22). Notice that $\Lambda \in \mathbb{R}$ is a large enough number to ensure the non-violation of constraints [83].[8] Notice also that the introduction of constraints (5.25b) and (5.25c) force to set an auxiliary variable $y_{s,lev}$ to 1 when there is some unit u scheduled during some semester l in the degree plan ($x_{u,l} = 1$) whose completion leads to level lev for skill s (i.e., $\sigma_u(s) \geq lev$); and to 0 otherwise. Then, the product $K\mathbf{y}$ obtains the maximum level for each skill in the degree plan.

Now, the objective function above is already an p-norm with a linear argument. However, this formulation penalises acquiring skills not required by the target job(s). We will address this issue next. We achieve this by reformulating the cost function in (5.25a) as follows:

$$\text{minimise} \quad \|\min(K\mathbf{y} - \mathbf{t}, \mathbf{0})\|_p. \tag{5.26}$$

This new formulation ensures that acquiring skills not required by the target job(s) does not contribute to the cost function. However, notice that we have introduced

[8] To ensure this condition, it is enough to consider $\Lambda = |X|$.

non-linearity in the argument of the p-norm of the objective function in (5.26), which we must eliminate to have a convex objective function. We achieve that by introducing a new set of auxiliary decision variables $z_s \in \mathbb{Z}$, with $s = 1, \ldots, |S|$. Decision variable z_s takes the value of the *level shortage* for skills s, considering the acquired level of s thanks to a degree plan and the required level of s for the target job(s). Therefore, we enforce that z_s is set to 0 when a given degree plan leads to acquiring a greater (or equal) level for skill s than required by the target job(s) ($Ky_s \geq t_s$). Otherwise, z_s is set to the difference, shortage, between acquired skills and required skills ($Ky_s - t_s$).

With these new variables, we reformulate the objective in (5.26) as follows:

$$\text{minimise} \quad \|\mathbf{z}\|_p, \tag{5.27a}$$

$$\text{subject to} \quad \mathbf{z} \leq K\mathbf{y} - \mathbf{t}, \tag{5.27b}$$

$$\mathbf{z} \leq \mathbf{0}. \tag{5.27c}$$

Notice that the argument of the p-norm in the objective function (5.27a) is linear. Therefore, we are in the position of formalising the DPP as an NMP, which considers the objective function (5.27a), the constraints required by the original formulation of the DPP, and the new constraints added by the reformulation in this section, i.e., (5.25b), (5.25c), (5.27b), (5.27c). Formally:

$$\begin{aligned} \text{minimise} \quad & \|\mathbf{z}\|_p, \\ \text{subject to} \quad & \mathbf{z} \leq \mathbf{0}, \quad \mathbf{z} \leq K\mathbf{y} - \mathbf{t}, \quad A\mathbf{x} \leq \Lambda \cdot \mathbf{y}, \\ & A\mathbf{x} \geq \Lambda \cdot (\mathbf{y} - \mathbf{1}) + \mathbf{1}, \quad B\mathbf{x} \leq \mathbf{1}, \quad P\mathbf{x} = \mathbf{0}, \\ & \mathbf{c}^T\mathbf{x} = c_T, \quad D\mathbf{x} = \mathbf{d}, \quad R\mathbf{x} \geq r\mathbf{x}, \quad E\mathbf{x} = \mathbf{1}, \\ & \mathbf{1}^T\mathbf{v} \geq 1, \quad F\mathbf{x} \geq f\mathbf{v}, \quad G\mathbf{x} \geq g\mathbf{v}, \quad Q\mathbf{x} = \mathbf{1}. \end{aligned} \tag{5.28}$$

Although the formulation above uses four different sets of decision variables ($\mathbf{x}, \mathbf{v}, \mathbf{y}$ and $\mathbf{z}$), we remark that our primary decision variables are $\mathbf{x}$, which are the ones employed to build a degree plan.

Since (5.28) is an NMP, we can solve it with the solution techniques discussed in Section 2.1.2.

5.3.1 Equivalence Between MIP and NMP

Having presented such a non-trivial linearization of our optimisation problem, we now provide the formal proof of the equivalence of the initial MIP (5.23) and the final NMP (5.28). To do so, we must prove that: (i) the formalisation in (5.28) does not change the space of feasible solutions of $\mathbf{x}$; and (ii) the ordering of each pair of feasible solutions in both problems is the same. The latter statement is necessary to guarantee that our transformation preserves optimal solutions.

To prove statement (i), we formally define the space of feasible solutions.

Definition 5.3.1 (Space of feasible solutions)**.** *The space of feasible solutions $\mathcal{F}$ is the set of points within the domain of the optimisation problem for which all the constraints are satisfied [24].*

Formally, for the formalisation (5.23) of DPP, we define $\mathcal{F}$ as follows.

$$\mathcal{F}_{\mathcal{C}} = \Big\{ \mathbf{x} \in \{0,1\}^{|X|} \,|\, \mathbf{x} \text{ satisfies } \mathcal{C} \Big\}, \tag{5.29}$$

where $\mathcal{C}$ is the set of constraints (5.3), (5.5), (5.6), (5.8), (5.11), (5.13), (5.14), (5.17), (5.20), and (5.22). To prove (ii), we demonstrate that both objective functions are equivalent, which is a *sufficient* condition to guarantee (ii).

For clarity, we proceed in two steps as done in Section 5.3. We first demonstrate that the formalisation (5.25) is equivalent to (5.24). Secondly, we demonstrate that the formalisation (5.27) is equivalent to (5.26).

Step 1 First, we introduce Lemma 5.3.1.

Lemma 5.3.1. *The space of feasible solutions of problems* (5.24) *and* (5.25) *is the same.*

Proof. Notice that the final problem (5.25) considers the set of constraints in the

initial problem $\mathcal{C}$ plus constraints (5.25b) and (5.25c). For the sake of simplicity we use the notation $\mathcal{C}'$ for the set of constraints

$$\mathcal{C}' = \mathcal{C} \cup \{A\mathbf{x} \leq \Lambda \cdot \mathbf{y}\,,\; A\mathbf{x} \geq \Lambda \cdot (\mathbf{y} - 1) + \mathbf{1}\}.$$

Formally, we define the space of feasible solutions of problem (5.25) as

$$\mathcal{F}_{\mathcal{C}'} = \Big\{\mathbf{x} \in \{0,1\}^{|X|} \,|\, \mathbf{x} \,\text{satisfy}\;\; \mathcal{C}'\Big\}. \tag{5.30}$$

Due to the fact that problem (5.25) considers two additional constraints, $\mathcal{F}_{\mathcal{C}'} \subseteq \mathcal{F}_{\mathcal{C}}$ is ensured. Now, we evaluate whether constraints (5.25b) and (5.25c) make that $\mathcal{F}_{\mathcal{C}'}$ contains less feasible solutions than $\mathcal{F}_{\mathcal{C}}$. Therefore we examine constraints (5.25b) and (5.25c), which can be expressed as

$$\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} \leq \Lambda \cdot y_{s,lev}, \tag{5.31}$$

$$\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} \geq \Lambda \cdot (y_{s,lev} - 1) + 1, \tag{5.32}$$

for all $s \in \mathcal{S}$; $lev = 1, \ldots, lev_{max}$, where $U_{s,lev} \subseteq \mathcal{U}$ is the set of units u that provides skill s such that when u is completed, the student acquires a skill level higher or equal than lev, i.e., $\sigma_u(s) \geq lev$. Thus, $\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} > 0$ indicates that after completing the considered degree plan, a student has acquired at least level lev for skill s. We remark that $y_{s,lev} \in \{0,1\}$ is a binary auxiliary variable which is set to 1 only when the degree plan has scheduled a unit whose completion leads to at least level lev for skill s. Considering that $\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} \in [0..\Lambda]$, we derive (5.31) and (5.32) for both possible values of $y_{s,lev}$ such that

$$\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} \leq \begin{cases} 0, & \text{if } y_{s,lev} = 0, \\ \Lambda, & \text{if } y_{s,lev} = 1. \end{cases}$$

$$\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} \geq \begin{cases} 1 - \Lambda, & \text{if } y_{s,lev} = 0, \\ 1, & \text{if } y_{s,lev} = 1. \end{cases}$$

for all $s \in \mathcal{S}$; $lev = 1, \ldots, lev_{max}$. As we can see, for any value $\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l}$ there exists only one value of $y_{s,lev}$ that satisfies the constraints. Hence, problem (5.25) does not lose solutions when constraints (5.25b) and (5.25c) are added.[9] Thus, we conclude that $\mathcal{F}_{\mathcal{C}'} = \mathcal{F}_{\mathcal{C}}$. □

Now, we prove (ii): that the ordering of each pair of feasible solutions in both problems is the same. To do so, it is sufficient to prove Lemma 5.3.2.

Lemma 5.3.2. *The objective function* (5.25a) *is equivalent to* (5.24)*, i.e.*

$$\|K\mathbf{y} - \mathbf{t}\|_p = \left(\sum_{s \in \mathcal{S}} \left| \max_{u,l} \left(\sigma_u(s) \cdot x_{u,l} \right) - t_s \right|^p \right)^{1/p}$$

Proof. First, we develop (5.25a):

$$\|K\mathbf{y} - \mathbf{t}\|_p = \left(\sum_{s \in \mathcal{S}} \left| K_s\mathbf{y} - t_s \right|^p \right)^{1/p} =$$
$$= \left(\sum_{s \in \mathcal{S}} \left| \sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s \right|^p \right)^{1/p}.$$

From the discussion in Proof 5.3.1, we remark that $y_{s,lev}$ can only be set to 1 when at least one unit that delivers level lev (or higher) for skill s is recommended, namely is part of the solution. For example, if there is planned a unit u that delivers level i for skill s (and no other scheduled unit gives a higher level for skill s), i.e., $\sigma_u(s) = i$, then $y_{s,lev} = 1$, for $lev = 1, \ldots, i$; and $y_{s,lev} = 0$, for $lev = i+1, \ldots, lev_{max}$. Hence, $\sum_{lev=1}^{lev_{max}} y_{s,lev} = i$ for skill s, which indicates that at least there is a unit u which $\sigma_u(s) = i$ is part of the solution. If we consider all scheduled units in the degree plan instead of only a unit, $\sum_{lev=1}^{lev_{max}} y_{s,lev}$ is equivalent to obtaining the maximum level of skill s of all scheduled units, which can be encoded as $\max_{u,l} \left(\sigma_u(s) \cdot x_{u,l} \right)$. Thus,

$$\|K\mathbf{y} - \mathbf{t}\|_p = \left(\sum_{s \in \mathcal{S}} \left| \sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s \right|^p \right)^{1/p} =$$
$$= \left(\sum_{s \in \mathcal{S}} \left| \max_{u,l} \left(\sigma_u(s) \cdot x_{u,l} \right) - t_s \right|^p \right)^{1/p}.$$

[9] Notice that when $\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} > 0$, only $y_{s,lev} = 1$ satisfies constraints (5.31) and (5.32). Otherwise, when $\sum_{u \in U_{s,lev}} \sum_{l \in \mathcal{L}} x_{u,l} = 0$, only $y_{s,lev} = 0$ satisfies constraints (5.31) and (5.32).

□

Then, we can introduce Theorem 5.3.3.

Theorem 5.3.3. *Problems* (5.24) *and* (5.25) *are equivalent.*

Proof. The proof is immediate by applying Lemma 5.3.1 and 5.3.2. □

Step 2 Now, we discuss that both (5.26) and (5.28) formalizations are equivalent. First, we remark that the space of feasible solutions $\mathcal{F}$ of formalisation (5.26) is equal to the one of (5.25). Second, we define the space of feasible solutions $\mathcal{F}_{\mathcal{C}''}$ of formalization (5.28) such that

$$\mathcal{F}_{\mathcal{C}''} = \Big\{ \mathbf{x} \in \{0,1\}^{|X|} \,|\, \mathbf{x} \,\text{satisfy}\ \ \mathcal{C}'' \Big\}, \tag{5.33}$$

where $\mathcal{C}''$ is the set of constraints such that

$$\mathcal{C}'' = \mathcal{C}' \cup \{\mathbf{z} \leq \mathbf{0}\,,\, \mathbf{z} \leq K\mathbf{y} - \mathbf{t}\}.$$

Then, we introduce Lemma 5.3.4.

Lemma 5.3.4. *The space of feasible solutions of problems* (5.26) *and* (5.28) *is the same.*

Proof. Thus, we must evaluate whether constraints (5.27b) and (5.27c) make $\mathcal{F}_{\mathcal{C}''}$ contain fewer feasible solutions over auxiliary variable $\mathbf{y}$. Note that both constraints can be expressed as

$$z_s \leq \sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s, \tag{5.34}$$

$$z_s \leq 0, \tag{5.35}$$

$\forall\, s \in \mathcal{S}$. From (5.35) we remark that we encode $\mathbf{z}$ as a non-positive vector, i.e., $\mathbf{z} \in \mathbb{Z}^{-|\mathcal{S}|}$. From the discussion in Proof 5.3.1, we notice that $0 \leq \sum_{lev=1}^{lev_{max}} y_{s,lev} \leq lev_{max}$. Notice that from Section 5.2 we remark that $0 \leq t_s \leq lev_{max}$. Hence, for any value $-lev_{max} \leq \sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s \leq lev_{max}$ there exist a value z_s that satisfies

the constraints. Thus, the formalization (5.28) does not lose feasible solutions when adding (5.27b) and (5.27c), i.e., $\mathcal{F}_{\mathcal{C}''} = \mathcal{F}_{\mathcal{C}'} = \mathcal{F}_{\mathcal{C}}$. □

Having proved Lemma 5.3.4 and guaranteed (i), we now move to Lemma 5.3.5.

Lemma 5.3.5. *The objective function* (5.26) *is equivalent to* (5.27a)*, i.e.,*

$$\|\min(K\mathbf{y} - \mathbf{t}, \mathbf{0})\|_p = \|\mathbf{z}\|_p.$$

Proof. To prove so, we examine both the constraints (5.34), (5.35) and the objective function (5.27a). We remark that $\mathbf{z} \in \mathbb{Z}^{|\mathcal{S}|}$ is a non-positive set of auxiliary variables due to constraint (5.35). However, the domain of $\mathbf{z}$ can be even more restricted by (5.34) when $\sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s < 0$. In addition, we know that we aim at minimising the absolute values of vector $\mathbf{z}$ to a power. Therefore, $\mathbf{z}$ variables' values will be the maximum values of their domain restricted by constraints (5.34) and (5.35). Formally:

$$\begin{aligned} z_s &= \begin{cases} \sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s, & \text{if } \sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s < 0, \\ 0, & \text{otherwise,} \end{cases} \\ &= \min(\sum_{lev=1}^{lev_{max}} y_{s,lev} - t_s, 0) \\ &= \min(K_s\mathbf{y} - t_s, 0). \end{aligned}$$

$\forall\, s \in \mathcal{S}$. As we have shown, both objective functions are equivalent and, consequently, (ii) is guaranteed. □

Finally, we can prove Theorem 5.3.6.

Theorem 5.3.6. *Problems* (5.23) *and* (5.28) *are equivalent.*

Proof. By direct applications of Lemmas 5.3.1, 5.3.2, 5.3.4 and 5.3.5. □

5.4 Experimental Analysis

The main objective of this section is to show the potential of our approach as a valuable decision-supporting tool for students and curriculum designers. To this end, we consider four scenarios we deem representative and realistic use cases for our approach. First, we focus on a student at the beginning of their degree who wants to plan the best path to achieve the skills required by some desired job(s). Secondly, we consider a student who wants to identify the jobs that are more *attainable* in the job market after completing the degree. Thirdly, we focus on the case when a student may be uncertain about the desired target job, hence they specify a set of multiple candidate jobs to receive a broader recommendation by our algorithm. Finally, we consider a student who has already started their degree and requests recommendations to move ahead. The student will request degree plans that lead to maximising the acquisition of the skills required by a desired job.

5.4.1 Dataset

Our use cases employ a real-world dataset of skills, i.e., the SFIA skill framework [29]. We consider version 8 of this dataset (with 121 skills), adopted by governments and institutions in Australia[10], the USA, the UK, and the EU to define jobs and positions that require a certain level of professional skills. As to degree data, we consider a real-world bachelor's degree, i.e., the Bachelor of Information and Communication Technology (ICT) of the Western Sydney University,[11] which involves $|\mathcal{U}| = 82$ units ($\mathcal{U}^{core} = 16$ of which are core units) and 7 different majors, together with the corresponding precedence constraints among units. Importantly, our dataset contains the courses in the syllabus of the Bachelor of ICT together with information on the skills acquired when completing each course. For that, we tasked university experts with associating, by hand, each unit with the skills in the SFIA framework that a student would acquire after completing the unit. Moreover, experts were asked to set the level of each skill acquired when completing a unit. Each unit rewards a

[10] https://sfia-online.org/en/tools-and-resources/standard-industry-skills- profiles.

[11] https://hbook.westernsydney.edu.au/programs/bachelor-information- communications-technology/.

student with 10 credits (i.e., $c_u = 10\ \forall u \in \mathcal{U}$). To obtain a Bachelor of ICT, a student must obtain $c_T = 240$ credits, completing 40 credits each semester. Therefore, a student completes a bachelor's degree in 6 semesters. Finally, we consider a dataset of 118 jobs obtained from the real-world *APS Digital Career Pathways* dataset [13], which specifies the level of SFIA skills required by each job.

5.4.2 Implementation

Even though our model can be used with any p-norm, following the discussion by Salas-Molina, Bistaffa and Rodríguez-Aguilar [165, Section 6], we restrict our attention to $p \in \{1, 2, \infty\}$, since those are the values that can be better semantically characterised. Specifically, the optimal solution for $p = 1$ aims to minimise the *total* (or, equivalently, the *average*) residual $r_i = z_i$, while for $p = 2$ we aim to minimise the *standard deviation* of the residual. Finally, the optimal solution for $p = \infty$ aims to minimise the maximum residual. Among the three above-mentioned values of p, we argue that $p = 1$ is the best-suited for our cases since we aim to reduce the total gap between the acquired skills and those required by the target job. For $p = 1$, the DPP amounts to a MILP. We remark that solving DPP for different values of p is indeed technically possible, but such an exercise has been left out for the sake of simplicity.

We implement and solve our problem in (5.28) with CPLEX v22.1.0.[12] We run our tests on a machine with an 8-core 1.00GHz CPU and 16.0GB of RAM. The average runtime for computing the optimal solution is 0.025 ± 0.001 seconds in all our experiments for $p = 1$. We also evaluate the computational benefit of solving the initial problem in (5.23) to solving our transformed problem in (5.28). We do so by comparing the runtime to solve the problem in (5.28) with the ILOG CPLEX solver to the runtime required to solve the original problem in (5.23) with the ILOG CP solver. We observe that while the optimal solution is computed in 0.025 seconds on average with our approach, the average runtime using the CP solver is 10.4 ± 2.2 seconds, i.e., our approach outperforms the CP solver by 2 orders of magnitude.

[12] Our implementation is available at https://github.com/RogerXLera/DegreePlanning.

At this point, it is important to mention that computing degree plans in real time have value from an application perspective. Indeed, we conceive our DPP as part of an *interactive* decision-support system for students and curriculum designers. Therefore, such a system must rapidly respond to requests issued by students and curriculum designers.

5.4.3 Use Case: Beginning-of-Degree Planning

Penetration Tester		*Major: Cybersecurity*
Semester	Unit	Unit type
2 (Sp)	Object Oriented Programming	Elective
3 (Au)	**Data Structures and Algorithms**	Major
4 (Sp)	**Network Security**	Major
5 (Au)	**Discrete Mathematics**	Elective
5 (Au)	Ethical Hacking Principles and Practice	Major
6 (Sp)	Computer Organisation	Major
6 (Sp)	Information Security	Major
6 (Sp)	**Formal Software Engineering**	Elective
Software Engineer		*Major: Mobile Computing*
Semester	Unit	Unit type
2 (Sp)	Social Computing	Elective
3 (Au)	**Data Structures and Algorithms**	Elective
3 (Au)	**Discrete Mathematics**	Elective
4 (Sp)	Information Systems Deployment and Management	Major
4 (Sp)	Mobile Applications Development	Major
5 (Au)	Wireless and Mobile Networks	Major
6 (Sp)	**Network Security**	Major
6 (Sp)	**Formal Software Engineering**	Elective

Table 5.1: Elective subjects computed for the degree plans targeting *Penetration Tester* and *Software Engineer* jobs. Units within both degree plans are in **bold**.

In our first use case, we consider students at the beginning of their degree (i.e., $\mathcal{Q} = \emptyset$); thus, they need a plan for all semesters of their degrees. Therefore, the goal of our first experiment is to show the degree paths computed by our approach depending on the skills required by two selected example jobs: *Penetration Tester* (a

Penetration Tester ($\alpha = 93\%$)			
Skill	RqL	AcL	Unit
Systems Design	4	4	Professional Experience
Software Design	3	3	Formal Software Engineering
Programming	3	3	Object Oriented Programming & Data Structures and Algorithms
Penetration Testing	4	3	Ethical Hacking Princ. and Practice
Software Engineer ($\alpha = 85\%$)			
Skill	RqL	AcL	Unit
Porting	4	3	Mobile Applications Development & Web Systems Development
Testing	3	2	Mobile Applications Development & Web Systems Development
Methods and Tools	4	4	Formal Software Engineering
Systems Integration	5	3	Info. Systs. Depl. & Manag. & Web Systems Development
Systems Design	4	4	Professional Experience
Software Design	3	3	Formal Software Engineering
Programming	4	4	Mobile Applications Development

Table 5.2: Skills required by the considered jobs and their Required Level (RqL), along with the Achieved Level (AcL) after completing the computed degree path and the unit that provides such skill level. Units in black are core units. Units in blue are electives that belong to the major program selected to complete by the approach. Units in brown are electives that do not belong to the major program selected by the approach.

highly-specific and technical job) and *Software Engineer* (a more general job). Table 5.1 shows the elective units of the computed degree paths coursed in each semester,[13] where the numbers are semesters' labels and "Au" and "Sp" stand for Autumn and Spring semesters, respectively. Our results show that, for the *Penetration Tester* job, the *Cybersecurity* major is planned to be coursed as the one that best matches the skills required. For the *Software Engineer* job, our approach plans to pursue the *Mobile Computing* major. Table 5.2 shows the skills acquired in each of the

[13]We do not show the degree path with the core units for the sake of brevity.

above-mentioned degree paths and the units providing such skills after completion. For both degree paths, we see that there are units related to the major that provide the required skills, justifying the pursuit of that major. For the *Penetration Tester* degree path example, we show that the *Ethical Hacking Principles and Practices* unit, which is compulsory to complete the *Cybersecurity* major, is the only unit that gives 3 out of 4 levels of the *Penetration Testing* skill. In addition, there are elective units that help students to achieve the required level for some skills. It is the case for the *Formal Software Engineering* unit, which provides students with level 3 of the *Software Design* skill and level 4 of the *Methods and Tools* skill. Other elective units are planned since they are the prerequisites for other electives. For instance, *Discrete Mathematics* is a prerequisite of *Formal Software Engineering*.

In such examples, a student would achieve almost all the required skills. However, we observe that some skills required by more multidisciplinary jobs cannot be covered. This result is expected due to the lack of units that can provide some skills or their required level. Indeed, it is reasonable to expect that, for some technical and multidisciplinary jobs, students might not reach the required level of skills after completing their bachelor's. Nonetheless, our approach allows students to start their professional careers in the best position based on the curriculum offered by the university.

5.4.4 Use Case: Most Attainable Jobs on the Market

In our second use case, the student's goal is to identify the degree plans (computed by our algorithm) whose completion leads to the most attainable jobs on the market. For this purpose, we first consider a *job affinity* function that measures the achieved skills with respect to the ones required by the target job(s).

Definition 5.4.1 (Job affinity)**.** *Given a target vector of skills* $\mathbf{t}$ *and a degree plan encoded by the decision variable* $\mathbf{z}$*, we define the job affinity function* $\alpha(\mathbf{t}, \mathbf{z})$ *as*

$$\alpha(\mathbf{t}, \mathbf{z}) = 100 \cdot \frac{\|\mathbf{t} + \mathbf{z}\|_1}{\|\mathbf{t}\|_1}. \tag{5.36}$$

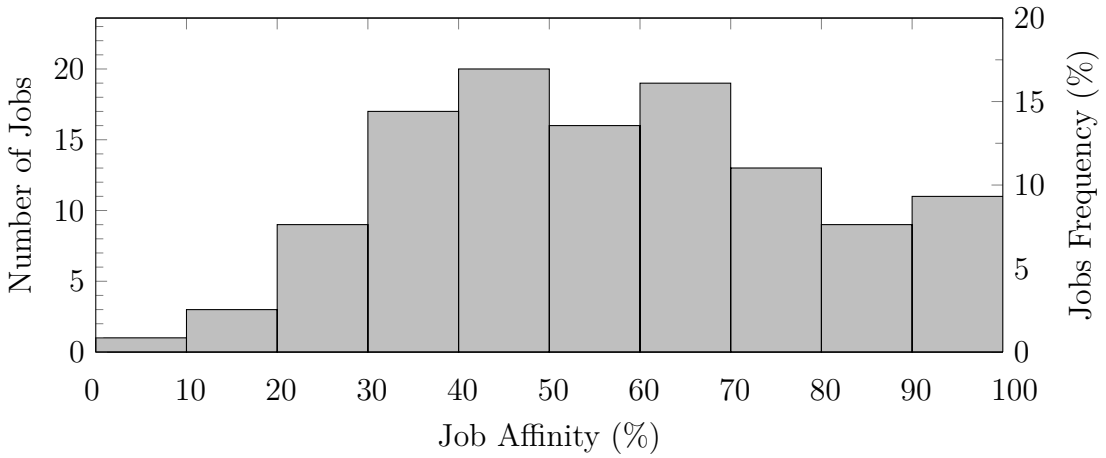


Figure 5.2: Histogram of job affinities for all jobs. We report the distribution of the job affinity values in two frequency metrics: the absolute number of jobs (left axis), and the percentage of jobs (right axis).

We define α in terms of $\mathbf{z}$, which measures the *shortage* between the achieved level of skills and the target $\mathbf{t}$. Hence, when $\alpha = 100\%$, the student has fully achieved the desired job requirements. On the other hand, when $\alpha = 0\%$, the student does not possess any skills out of those required by the job.

Given the above-defined job affinity function, we solve the DPP for all the jobs in our dataset and compute the corresponding job affinity values. Notice that the capability of our approach of solving the DPP within seconds is fundamental for this kind of analysis, for which we must solve the DPP multiple times (as many as jobs). Figure 5.2 reports the histogram of job affinity values. Results show that for almost 10% of target jobs (11 out of 118 target jobs), the optimal degree paths achieve a job affinity greater than 90%. Furthermore, for 58% of the jobs, the computed degree paths achieve job affinities greater than 50%. In more detail, for 8 jobs, the job affinity for their optimal degree plans is 100%, i.e., students acquire all the skills required by the jobs when completing the recommended optimal degree plans.

Furthermore, we illustrate the evolution of skills learned along semesters in Figure 5.3. We show how job affinity progressively increases over semesters. In more detail, by the end of the third semester (half of the degree plan), students acquire 40% of the required skills on average. Moreover, once the degree plan is completed, the average job affinity is 56% as shown in Figure 5.4. Therefore, 29% of job affinity

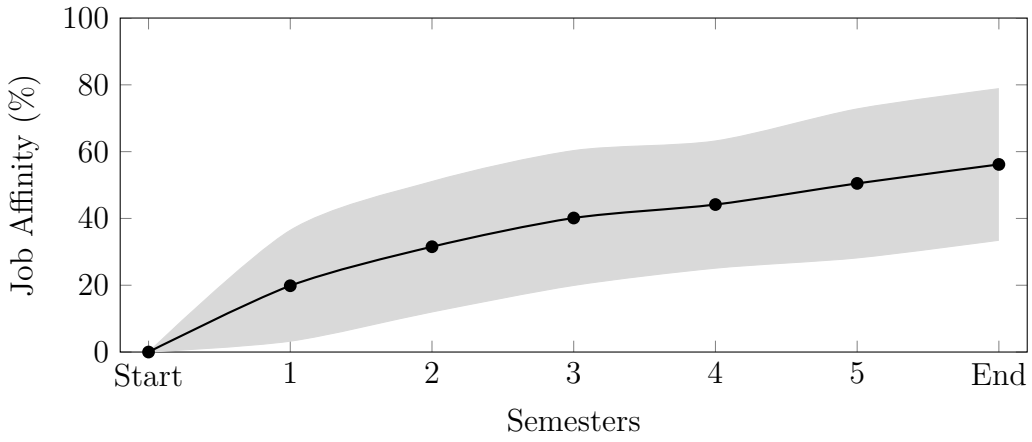


Figure 5.3: Evolution of the job affinity during the degree plan. The shadowed region represents the standard deviation of the job affinity for the different target jobs.

value is gained in the second part of a degree plan, where most of the elective units are completed. We argue that most of the skills not acquired can not be learned with any of the units of the degree curriculum. This is usual since some jobs require high-level skills that can only be acquired with additional specialised education or professional experience.

To better understand and illustrate the value of electives, in Figure 5.4, we show in greater detail the job affinity values for the degree plans computed considering each job in our dataset. Specifically, we plot the job affinity values considering (i) degree plans without elective units (considering only core and major units), (ii) whole degree plans (with electives), and (iii) all units from the syllabus (all $u \in \mathcal{U}$). Notice that elective units significantly increase the job affinity values of degree plans. In detail, the average job affinity without considering electives is 41.5%, while the average job affinity is 56.2%. Thus, elective units lead to a 26% relative increase in job affinity. Furthermore, the recommended degree plans have the highest job affinity values possible in most cases since coursing all available courses in the degree syllabus does not significantly increase job affinity. This indicates that most of the skills not acquired in the recommended degree plans can not be learned with any of the units of the degree curriculum. As mentioned above, this is usual since some jobs require high-level skills that can only be acquired with additional specialised education or professional experience.[14]

[14]For example, the role of "*Software Engineer*" requires level 5 of skill "*Systems Integration*"

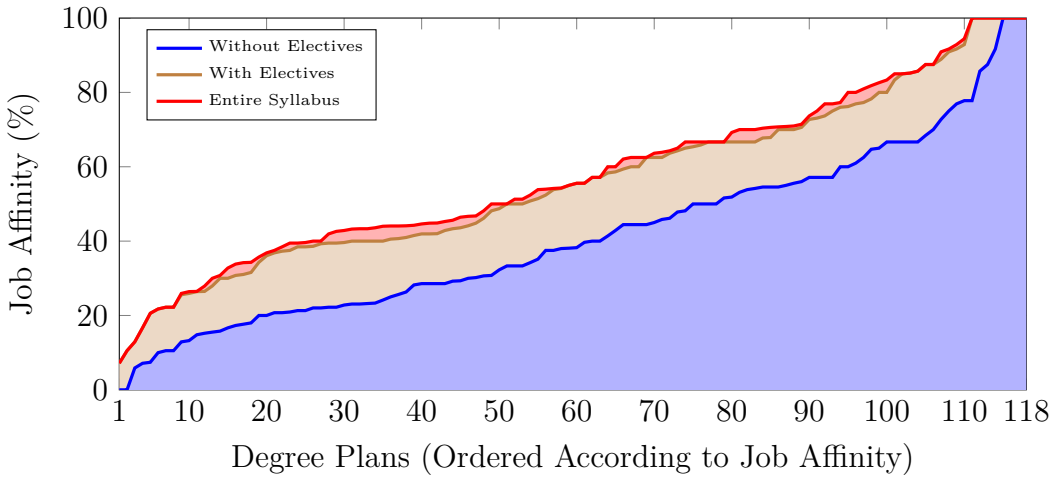


Figure 5.4: Cactus plot showing the job affinity for the degree plan of each target job. We order degree plans from the lowest job affinity to the highest on the x-axis. We plot the job affinity considering degree plans without the elective units (in blue); whole degree plans (in brown); and all units from the degree syllabus, $\mathcal{U}$ (in red).

We believe these analyses could be useful for students and curriculum designers. On the one hand, students can identify which jobs are more aligned with their degree, helping them better understand whether it is appropriate for the professional career they aim to pursue. Moreover, students can track their skill acquisition during their studies. On the other hand, curriculum designers can evaluate which portion of the job market is "covered" by the degree in terms of professional skills and, if deemed appropriate, adjust the offered units in a principled way.

5.4.5 Use Case: Multi-Job Target

Our third use case focuses on a scenario in which a student is uncertain about the specific job to pursue but has several potential candidate jobs in mind. Along these lines, our algorithm could benefit the student since it could recommend a *joint* degree path to cover the professional skills required by such potential jobs. Nonetheless, such a joint degree plan aiming at several target jobs at once could provide, for each individual job, a lower skill coverage compared to the degree paths specifically aimed at each job separately.

(see Table 5.2). According to the SFIA, such a high level indicates that "*the employee is capable of providing authoritative guidance in such skill and is accountable for delivering significant work outcomes*". Therefore, such a skill level can only be acquired after years of professional experience.

Thus, this experiment aims to quantify the skill coverage of joint degree plans *with respect to* individual degree plans. Hence, the larger the skill coverage of a joint degree plan *relative* to such individual plans, the better, because it is closer to the best possible individual plans.

To quantify the skill coverage of joint degree plans, we first define an auxiliary metric to compute the skill coverage of each job in a candidate job set given a joint degree plan. Recall that a degree plan (as well as a joint degree plan) is characterised by a set of decision variables. Along these lines, we use the vector $\mathbf{z}$ encoding the computed plan to determine the skill coverage of a joint degree plan. In detail, $z_s \in \mathbb{Z}^-$ encodes the level shortage of skill s of a joint degree plan, i.e., the difference between the target skill level t_s and the skill level that would be achieved by following the plan.

Definition 5.4.2 (Single Job Affinity (SJA))**.** *Given a candidate job $j = \langle S_j, {\sigma'}_j \rangle$, a candidate job set $\mathcal{J}$ such that $j \in \mathcal{J}$, a target vector of the candidate job set required skills $\mathbf{t}$, and a joint degree plan for the jobs in $\mathcal{J}$ whose skill level shortage is encoded by the decision variable vector $\mathbf{z}$, we define the Single Job Affinity of job j for the joint degree plan as:*

$$\alpha_{j,\mathcal{J}}(\mathbf{t}, \mathbf{z}) = 100 \cdot \frac{\sum_{s \in S_j} \min({\sigma'}_j(s), t_s + z_s)}{\sum_{s \in S_j} {\sigma'}_j(s)}, \tag{5.37}$$

where $t_s + z_s$ is the achieved level of skill s after completing the joint degree plan.

After that, we define the notion of *Relative* Job Affinity to compute the skill coverage of a job in a joint degree plan with respect to an individual degree plan.

Definition 5.4.3 (Relative Job Affinity (RJA))**.** *Given a candidate job $j \in \mathcal{J}$, the target vector $\mathbf{t}$ of the skills required by the candidate jobs in $\mathcal{J}$, the skill level shortage $\mathbf{z}$ of a joint degree plan for $\mathcal{J}$, the target vector $\hat{\mathbf{t}}^j$ of the skills required by job j, and the skill level shortage $\hat{\mathbf{z}}^j$ of an individual job degree plan for job j, we define the Relative Job Affinity of the joint degree plan with respect to the individual degree plan as*

$$RJA(j, \mathbf{t}, \mathbf{z}, \hat{\mathbf{t}}^j, \hat{\mathbf{z}}^j) = 100 \cdot \frac{\alpha_{j,\mathcal{J}}(\mathbf{t}, \mathbf{z})}{\alpha_j(\hat{\mathbf{t}}^j, \hat{\mathbf{z}}^j)}, \tag{5.38}$$

where $\alpha_{j,\mathcal{J}}(\mathbf{t},\mathbf{z})$ is the SJA of job j and $\alpha_j(\hat{\mathbf{t}}^j,\hat{\mathbf{z}}^j)$ is the job affinity of job j individual job degree plan.[15]

Given a set of candidate jobs and a joint degree plan for them, we quantify their skill coverage in three steps. First, we compute the degree plan for each job in the candidate job set. Second, we compute the RJA for each job in the candidate job set. Finally, we need a global measure to quantify the skill coverage of a joint degree plan with respect to all the individual plans. Thus, we aggregate the candidate jobs' RJAs to obtain the skill coverage of the joint degree plan. To this end, we define the Average RJA as follows.

Definition 5.4.4 (Average Relative Job Affinity (ARJA))**.** *Given a candidate job set $\mathcal{J}$, the target vector $\mathbf{t}$ of the skills required by the candidate jobs in $\mathcal{J}$, the skill level shortage $\mathbf{z}$ of a joint degree plan for $\mathcal{J}$, the set of target vectors (denoted as $\hat{T}$) of the skills required by each job $j \in \mathcal{J}$, and the set of skill level shortages (denoted as $\hat{Z}$) of an individual job degree plan for each job $j \in \mathcal{J}$, we define the Average Relative Job Affinity of the joint degree plan as*

$$ARJA(\mathcal{J},\mathbf{t},\mathbf{z},\hat{T},\hat{Z}) = \frac{1}{|\mathcal{J}|}\cdot\sum_{j\in\mathcal{J}} RJA(j,\mathbf{t},\mathbf{z},\hat{\mathbf{t}}^j,\hat{\mathbf{z}}^j) = \frac{100}{|\mathcal{J}|}\cdot\sum_{j\in\mathcal{J}}\frac{\alpha_{j,\mathcal{J}}(\mathbf{t},\mathbf{z})}{\alpha_j(\hat{\mathbf{t}}^j,\hat{\mathbf{z}}^j)}. \quad (5.39)$$

Once we have defined our metric for skill coverage of a joint degree plan, we describe our experimental setting. We aim to quantify how well a joint degree plan covers a student's requirements as their uncertainty increases, assuming that if students are more uncertain, they will specify more candidate jobs. Thus, we consider up to $|\mathcal{J}| = 10$ for cases with maximum uncertainty, and we refer to cases with minimum uncertainty when $|\mathcal{J}| = 2$. To compute the ARJA for different sizes of candidate job sets, we solve the DPP for 4500 unique candidate job sets of size $|\mathcal{J}| = 2, 3, \ldots, 10$, i.e., 500 combinations of jobs per set size. Notice that computing the degree plan for all the possible job combinations is impossible because of the so-called "Curse of Dimensionality". For this experimental setting, the total number of combinations is $\sum_{k=1}^{10}\binom{118}{k} = 1.07\cdot 10^{14}$. Therefore, given the impossibility of computing the

[15] Notice that RJA $\in [0, 100]$.

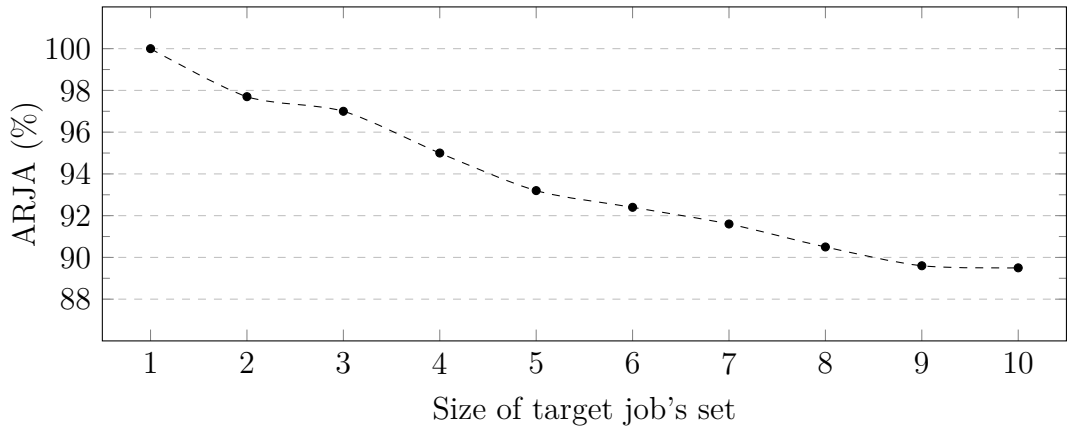


Figure 5.5: Average RJA along the size of the set of target jobs.

degree plan due to the unmanageable number of possible combinations of jobs, the capability of our approach to compute degree plans in real time is crucial.

Figure 5.5 shows how the skill coverage decreases as the student's uncertainty increases. In detail, we notice a decrease in the ARJA as the number of target jobs increases, i.e., as the number of target jobs (i.e., the student's uncertainty) increases. We do not plot the statistical error since it is negligible ($\approx 0.2\%$). Overall, even in the case with maximum uncertainty, i.e., the case where candidate job sets contain 10 jobs, the ARJA is high, i.e., 10.5% decrease with respect to no uncertainty about their target job (i.e., $|\mathcal{J}| = 1$). This shows that the joint degree plans that DPP computes are valuable recommendations for a student despite uncertainty. Specifically, for instances considering up to 4 target jobs, the average ARJA is 95%. Moreover, the average ARJA decreases below the 90% exclusively for candidate job sets with a big number of jobs, i.e., $|\mathcal{J}| \geq 9$. Therefore, when a student is uncertain about their desired career path, our approach can compute degree paths for multiple target jobs with a minimum decrease in skill coverage with respect to the case where students know their desired job. This is due to the fact that, in our dataset, different jobs may require common skills, as well as to the fact that units may provide skills that are required by multiple jobs.

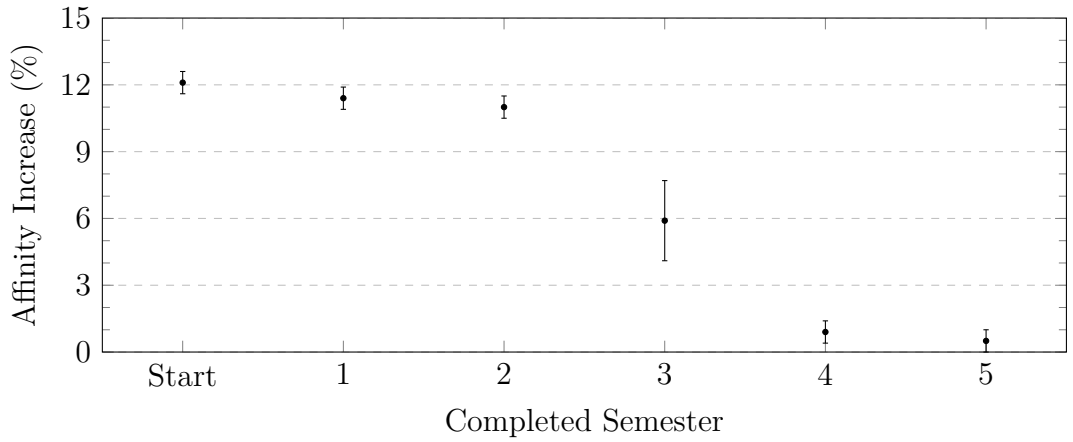


Figure 5.6: Evolution of job affinity increase along semesters.

5.4.6 Use Case: Mid-Degree Planning

In our fourth use case, we consider that a student has already started their bachelor's degree and they have already completed some units. Henceforth, we employ our approach to plan a path for the remaining part of their degree. We consider 12 "reference" degree paths based on the recommended ones in the bachelor's syllabus. Then, for each reference degree path, we consider that there is a student who partially completed such a degree path (i.e., 1 semester, 2 semesters, etc.). We then complete each partial path with our approach and calculate the job affinity increase by comparing the computed path with the reference one. We analysed the 20 jobs with the highest job affinity for each "reference" degree path.

Figure 5.6 shows the increase in the job affinity's mean and standard deviation across all the above-mentioned experiments. Results indicate that our approach can indeed provide an improvement with respect to the reference path indicated by the bachelor's syllabus. We also observe that the maximum increase is obtained when our approach is employed early in the degree path. This result is expected since, in this case, the optimisation is less constrained, thus it can reach a higher job affinity. Moreover, we observe a sharp drop in the job affinity increase when students complete half of their degree. As expected, choosing a target during the first half of the degree is crucial for skill achievement.

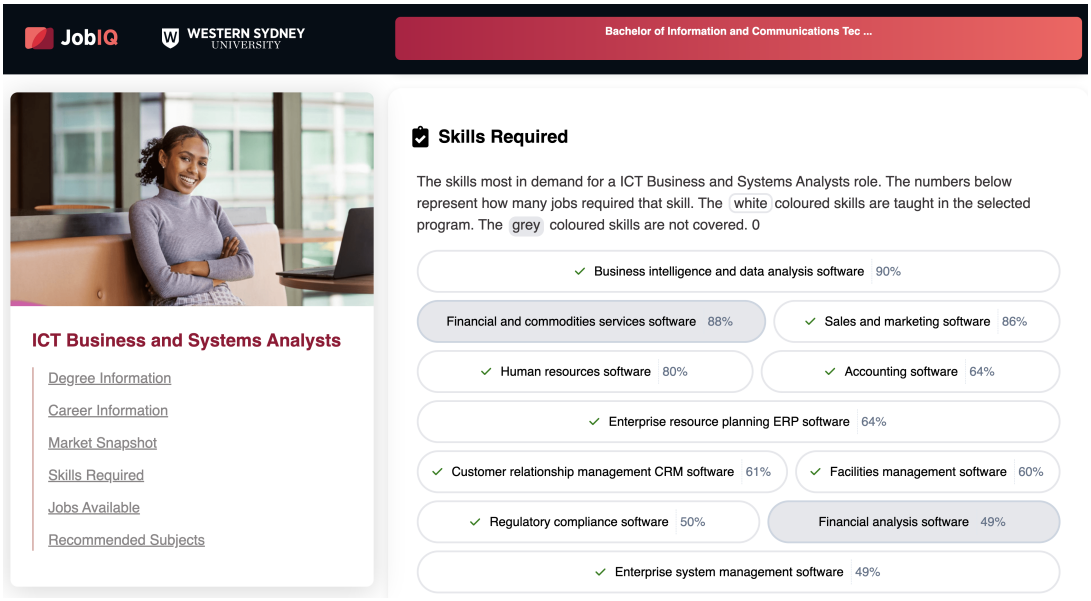


Figure 5.7: Screenshot of JobIQ showing the skills required by the *ICT Business and Systems Analysts* job and which skills are acquired after completing the Bachelor's Degree in ICT.

5.5 Real-world Applicability of our Approach

5.5.1 Current Deployment

To showcase the applicability of our approach to real-world university systems, we embedded our approach in JobIQ[16]. Such a system recommends Western Sydney University degrees based on students' career preferences. The website informs students which skills they acquire during their study (as shown in Figure 5.7) and which job markets they can target after graduation. Furthermore, it shows them current job offers from the countries where such a university is present, i.e., Australia, New Zealand, and Indonesia, including financial prospects or demanding employers. This approach provides an idea of what career prospects there are for our graduates. Therefore, JobIQ has become one of the main recruitment tools for our hiring strategy.

5.5.2 Extending Our DPP Model to Other Curricula

Even though our formulation discussed in Section 5.2 involves some aspects specific

[16] https://www.jobiq.com.au. Last Consulted: February 2025.

to the Bachelor's Degree in ICT from Western Sydney University (such as the major requirements in Section 5.2.1), we will now discuss how it can be adapted to other common scenarios by means of additional linear constraints.

As a first example, it is common for students to be forced to enrol in a minimum number of credits each semester. Furthermore, to avoid students suffering from an excessive workload, degree curricula usually specify a maximum number of credits [34]. In such cases, we would replace constraint (16) by the set of constraints $\mathbf{d_{min}} \leq D\mathbf{x} \leq \mathbf{d_{max}}$, where $\mathbf{d_{min}}, \mathbf{d_{max}} \in \mathbb{N}^{|\mathcal{L}|}$ are vectors whose components are the minimum and maximum number of credits allowed to enrol per semester. Furthermore, it is common to require that students complete a minimum number of credits to graduate instead of allowing them to just enrol in a given number of credits. In such a case, we would modify constraint (14) such that $\mathbf{c}^T\mathbf{x} \geq c_T$.

Another common scenario is when academic curricula require completing a given number of "*basic-level*" units to access "*higher-level*" units. In such a case, we formalise such a restriction with a new type of prerequisite constraint that enforces having completed a certain number of basic-level units before enrolling in a higher-level course. We define a subset of units containing the basic-level units $\mathcal{B} \subset \mathcal{U}$ and a second subset of units containing the higher-level units $\mathcal{H} \subset \mathcal{U}$. Then, we impose a set of linear constraints such that

$$\sum_{u \in \mathcal{B}} \sum_{i=1}^{l-1} x_{u,i} \geq n_h \cdot x_{h,l}, \quad \text{for all } h \in \mathcal{H}, l \in \mathcal{L},$$

where n_h is the number of basic-level units to be completed to enrol in unit h.

Moreover, some institutions require students to complete special courses (or a certain number) to finish their bachelor's degree. For example, in the US, some degrees require the completion of "*Liberal Education*" courses [162]. In such a case, we define a special subset of units $\mathcal{I} \subset \mathcal{U}$. Then, we impose a set of linear constraints such that

$$\sum_{u \in \mathcal{I}} \sum_{l \in \mathcal{L}} x_{u,l} \geq n_{\mathcal{I}}$$

where $n_{\mathcal{I}}$ is the number of special units to be completed to finish a bachelor's degree.

5.6 Conclusions

As mentioned above, developing decision-support tools for guiding students throughout their educational pathways toward the acquisition of the skills required by their preferred job(s) is key. In this chapter, our goal was to contribute to the development of such decision-support tools.

First, we formalised the problem of computing the degree path for a student to be employable in their desired jobs as an optimisation problem. Solving such an optimisation problem yields the degree paths that maximise the "coverage" of the professional skills required by a student's desired job(s). Second, because of the non-linearity of the objective function in that optimisation problem, we introduced a non-trivial linearization of the problem so that we can solve it with standard integer optimisation tools. We formally prove that our transformation produces an equivalent optimisation problem.

Throughout our experimental evaluation with real-world data from the Bachelor of ICT of the University of Western Sydney, we showed that our approach can be a valuable decision-support tool for (i) students planning their academic pathways and (ii) curriculum designers assessing the matching between a university's degrees and the job market. Despite showcasing it for an actual real-world example, our model is straightforward to adapt to other university curricula without changing the proposed solution method.

In conclusion, this chapter is devoted to answering **Research Question Q5**: How do we formalise the problem of computing a Bachelor's Degree Plan that considers students' career preferences as a MIP? We answer this by (i) formally defining the requirements of a Bachelor's Degree as linear constraints, (ii) formulating an objective function that captures the goal of maximising the acquisition of skills relevant to students' preferred jobs, and (iii) linearising this objective to enable efficient solution of the resulting MILP with state-of-the-art solvers.

Chapter 6

Explainability for Optimisation

6.1 Introduction

In the preceding chapters, we introduced different optimisation-based approaches that compute recommendations (i.e., the solution to an optimisation problem) to support a variety of decision-making processes. For instance, in Chapter 5, we presented an algorithm that recommends a personalised degree plan, specifying which university subjects a student should take in each semester in order to acquire the skills most relevant to their preferred career paths. Likewise, in Chapter 4, we proposed an approach for research project planning that recommends how researchers should allocate their time across project tasks over the project's duration, with the goal of balancing researchers' preferences and the efficient use of the project's budget.

Even though these application domains often rely on classical optimisation techniques, which are arguably more interpretable than modern "*black box*" ML models, their outcomes could still be difficult to understand for the user, hence making the adoption of the computed recommendations more difficult. Furthermore, users may need to ask questions to make informed decisions and ensure the human oversight required by modern international guidelines [47].

To address the above-mentioned needs, in this chapter, we focus on developing a methodology to answer the questions that users could pose about the recommendations produced by an optimisation approach and, consequently, we tackle research question **Q6**. Such questions reflect a human tendency to seek contrastive explanations —those that do not merely state why something was recommended, but why it

was recommended rather than an alternative. This perspective is supported by the social sciences, where it is argued that explanations are most useful and satisfying when they are contrastive [126].

In line with this, some works in the XAI field aim to generate contrastive explanations for justifying the outcome of specific ML algorithms [182, 127, 97]. However, to the best of our knowledge, no prior work has addressed contrastive explanations for MIP problems. Existing approaches to explaining the outcomes of optimisation problems typically work by numerically comparing the original recommendation with an alternative proposed by the user [76, 211]. While effective in highlighting differences, these methods require solving an additional optimisation problem to generate each explanation, which can be computationally expensive. Moreover, these techniques are often tailored to specific application domains [76, 150]. More importantly, they do not take advantage of recent advances in LLM to produce natural language explanations. Instead, they rely on abstract representations [211] or fixed templates [76, 150], which is not desirable when providing explanations as part of an interaction [126].

Along these lines, we investigate (i) how to extract and structure the relevant information to answer a user question and (ii) whether LLMs can effectively be leveraged to answer users' questions about DSS recommendations (i.e., research questions **Q6.1** and **Q6.2**). In that context, while LLMs have shown remarkable capabilities in natural language processing, generating meaningful explanations for optimisation-based decisions remains highly challenging, even for human experts. This difficulty stems from the complexity of the optimisation process, which often involves numerous restrictions and domain-specific rules. For example, in the case of the DPP, subjects may have strict precedence requirements, some subjects might not be offered in particular semesters, or students must complete a certain number of credits. Thus, solely asking an LLM (as done, for instance, by Rajani *et al.* [152]) to explain complex recommendations may lead to incorrect or misleading explanations [119, 155], since LLMs are known to struggle with abstract reasoning tasks [75]. Lastly, unlike existing approaches that develop domain-specific explainability methods, and in line with Research Question **Q8**, in this chapter, we investigate whether it is possible to

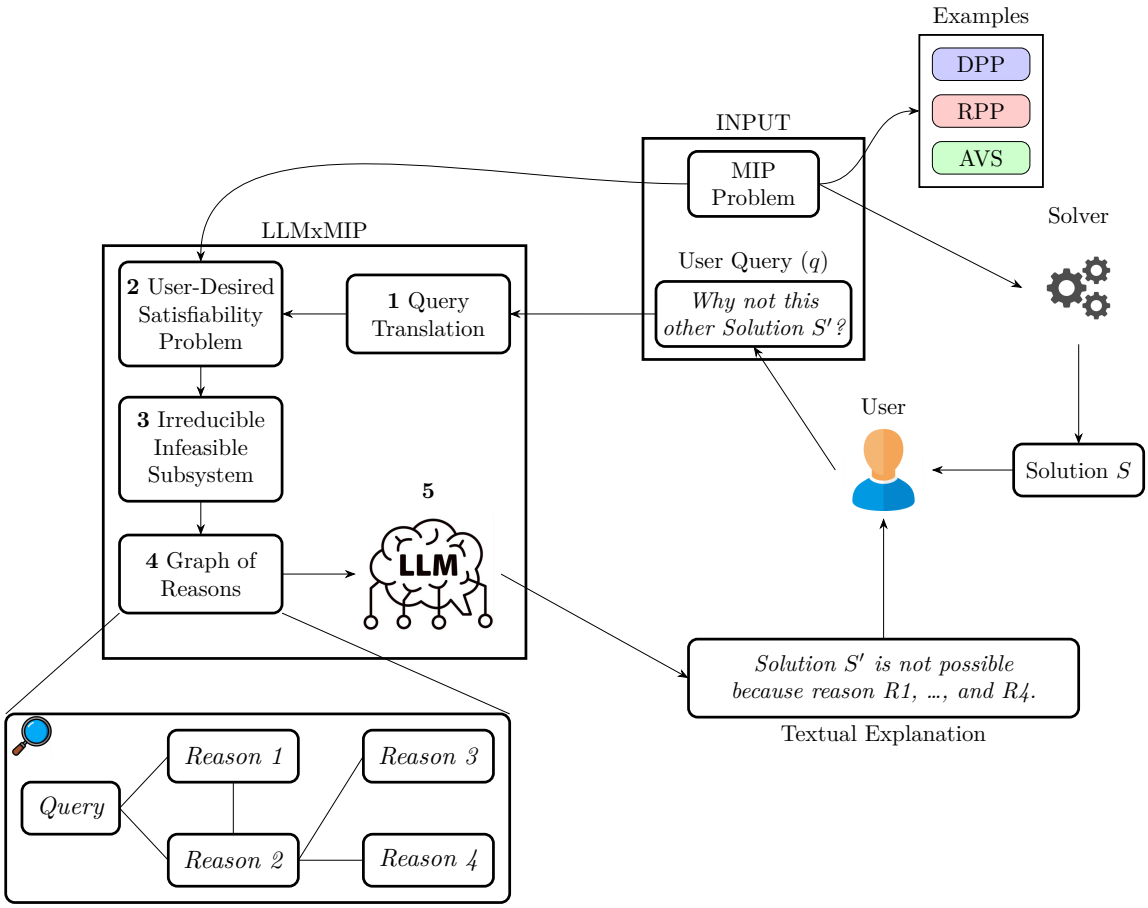


Figure 6.1: Overview of our explanatory approach. Section 6.2 discusses its steps in detail.

design a general explanation method that can be applied to any MIP problem and, by extension, to all optimisation problems addressed in this thesis.

To overcome such a limitation, in this thesis, we propose *Large Language Models for Explainable MIP* (LLMxMIP), a novel methodology to generate explanations for optimisation-based application domains that combines constraint reasoning with LLMs to generate textual explanations that are *complete* —i.e., they include all sufficient reasons relevant to answer a user's question— and *accurate* —i.e., they are free from errors or inaccuracies. Within LLMxMIP, we exploit constraint reasoning techniques to structure and semantically characterise the information required by the LLM, which is in charge of generating a textual explanation answering the user's question.

More specifically, LLMxMIP is a process that follows five main steps, outlined in

Figure 6.1. First, it transforms a user's query, i.e., the question posed by the user, into a constraint and reformulates the original optimisation problem as a CSP that is compliant with the user's query (first and second steps, respectively). Then, it finds the conflicts that prevent us from obtaining a better solution that accommodates the user query. This amounts to extracting the set of constraints in the CSP that make it infeasible (third step).[1] Our approach exploits constraint reasoning to compute such a set of constraints, i.e., the minimal subset of constraints that make the CSP infeasible, namely the IIS [43]. Afterwards, LLMxMIP transforms the unstructured collection of mathematical constraints in the IIS into a structured graph, the so-called *graph of reasons*. Such a graph captures the relationships between constraints and assigns them a semantic interpretation —which we call "*reason*"— by means of *Natural Language* (NL) templates. Finally, an LLM leverages the graph of reasons to generate a textual explanation.

Overall, we make the following novel contributions to the state-of-the-art:

- We propose LLMxMIP, a novel methodology that integrates state-of-the-art AI techniques —combining constraint reasoning with LLMs— to explain the outcomes of an optimisation-based process encoded as a MIP.
- We employ constraint reasoning techniques to identify the underlying reasons that explain why it is not possible to obtain a better solution that satisfies the user's query. Then, we transform such an unstructured set of reasons into a graph, whose structure allows one to capture the relationships between reasons.
- We leverage LLMs to generate textual explanations based on the graph of reasons, whose structured representation allows LLMs to produce complete and accurate explanations.
- We test LLMxMIP on the three real-world optimisation-based scenarios discussed in this thesis, i.e., the DPP (Chapter 5), the RPP (Chapter 4), and the Aggregation of Value Systems in (Chapter 3). Additionally, we test LLMxMIP on another well-known problem in the optimisation literature, the RCPSP. Our

[1] The infeasibility of the CSP proves that the query of the user leads to a solution that is worse than the computed optimal one.

results indicate that using graphs of reasons as input to LLMs results in textual explanations that are significantly better in terms of completeness and readability.

The remainder of this chapter is organised as follows. In Section 6.2, we propose our approach. In Section 6.3, we evaluate the quality of the explanations generated with LLMs. Section 6.4 concludes the chapter and discusses the research questions tackled in this chapter.

6.2 Building Explanations for Optimisation-Based Decisions

This section presents LLMxMIP, our methodology for generating contrastive explanations for users interacting with optimisation-based DSS. As illustrated in Figure 6.1, our approach combines constraint reasoning techniques with LLMs to produce textual explanations. Specifically, the figure outlines the key components of LLMxMIP, which we explain in detail in the remainder of this section. While the methodology is general —applicable to any application domain involving a MIP problem— we use the DPP, formalised in Chapter 5, as an illustrative example throughout this section.

In a MIP problem, constraints encode restrictions that the solution must satisfy, while the objective function encodes some preferences or quantities to be optimised [24]. Thus, a solver output is affected by the constraints and the objective function. In the case of the DPP, the solver output is the degree plan, which is affected by the syllabi restrictions and the skills required for the desired job. Hence, we propose to exploit the information provided by the constraints and the objective function to build contrastive explanations. The following running example illustrates the concepts of our approach.

Example 6.2.1. *Consider that we aim to compute a degree plan for a real-world Bachelor's degree in Information and Communication Technology for a student who pursues a career in Software Engineering.*[2] *Such a Bachelor's Degree lasts six*

[2]Section 5.4.1 provides the details of the problem data.

Table 6.1: Information of the key subjects of the optimal degree plan of Example 6.2.1. We provide the semesters where the subject is planned (and the season), the seasonal offer of the subject, its predecessors, and whether they are core subjects. In case a subject is not planned, we indicated with a − sign in the semester column.

Subject	Semester	Offer	Predecessors	Core
Prog. Fundamentals (PF)	2 (Spr.)	Both	-	Yes
Prog. Techniques (PT)	4 (Spr.)	Spr.	PF	Yes
Object Oriented Prog. (OOP)	–	Spr.	PF	No
Operating Systems Prog. (OSP)	5 (Aut.)	Aut.	PT or OOP	Yes

semesters (the first semester starts in autumn) and students must complete 40 credits each semester (completing a subject rewards students with 10 credits). As mentioned in Section 5.2, solving the DPP amounts to maximising the required skills' level of the desired job, while satisfying a set of constraints C. For the sake of clarity, we only show a part of the optimal degree plan $\boldsymbol{x}^*$ in Table 6.1.

Some approaches in the literature build a contrastive explanation by comparing an optimal solution with a user-desired solution [76, 211]. The latter results from translating a user's "*query*", i.e., a question for which a user expects an explanation, into *additional constraints* that the decision (optimal solution) should satisfy. While our methodology does not directly compare both solutions as in previous work, we similarly encode a user's query as additional constraints on the optimisation problem —a process we describe in detail in the next section.

6.2.1 Query Translation

Here, we characterise the types of queries that a user can pose and how to encode them into query constraints. More precisely, a user asks "*why*" questions (in natural language) that challenge a decision or solution computed by an optimisation-based DSS. For example, regarding the DPP, a user might ask "Why is subject Programming Fundamentals not planned at semester 4?". Hence, a contrastive explanation should tell the user why an alternative solution complying with the query, the so-called user-desired solution, is not at least as good as the optimal solution. In the case of the previous example, this alternative solution would be a degree plan in which Programming Fundamentals is scheduled in semester 4, as suggested by the

Table 6.2: Set of possible user questions, query constraints and constraint encodings for the DPP formalisation.

	Question	Query constraints	Constraint encoding (C_Q)
Q1.	Why is subject u planned at semester l?	**veto** to plan u at semester l	$x_{u,l} = 0$
Q2.	Why is subject u not planned at semester l?	**enforce** to plan u at semester l	$x_{u,l} = 1$
Q3.	Why is subject u **not** planned before semester l?	**enforce** to plan u before semester l	$\sum_{i=0}^{l-1} x_{u,i} = 1$
Q4.	Why is subject u **not** planned after semester l?	**enforce** to plan u after l	$\sum_{i=l+1}^{l_n} x_{u,i} = 1$
Q5.	Why is subject u planned?	**veto** u	$\sum_{l \in \mathcal{L}} x_{u,l} = 0$
Q6.	Why is subject u not planned?	**enforce** u	$\sum_{l \in \mathcal{L}} x_{u,l} = 1$
Q7.	Why is the group of subjects U planned at semester l?	**veto** to plan all $u \in U$ at semester l	$\sum_{u \in U} x_{u,l} < \lvert U \rvert$
Q8.	Why is the group of subjects U **not** planned at l?	**enforce** to plan all $u \in U$ at semester l	$\sum_{u \in U} x_{u,l} = \lvert U \rvert$
Q9.	Why is subject u planned at semester l instead of unit u'?	**enforce** to plan u' at l and **veto** u at l	$x_{u',l} = 1$, $x_{u,l} = 0$
Q10.	Why is subject u planned instead of subject u'?	**enforce** u' and **veto** u	$\sum_{l \in \mathcal{L}} x_{u',l} = 1$, $\sum_{l \in \mathcal{L}} x_{u,l} = 0$

user's query. Henceforth, since a user's query challenges a recommendation computed by an optimisation-based DSS, we assume that in case of a MIP problem, a user's query refers to value assignments that do occur in an optimal solution [210]. In other words, a user's query is related to the decision variables of the MIP at hand. In optimisation problems, decision variables are related to objects, activities, agents, etc. For instance, in the DPP, the decision variables indicate whether a subject is planned for a given semester. Hence, there is a decision variable for every pair of subjects and semesters.

Example 6.2.2. *Following Example 6.2.1, a user may ask "*Why is *Programming Fundamentals* not planned at semester 4?*" about the solution in Table 6.1. Building an explanation to this query requires enforcing to plan Programming Fundamentals at semester 4, which is encoded by constraint*

$$x_{PF,4} = 1, \tag{6.1}$$

(see Table 6.2). Enforcing this constraint would allow obtaining a user-desired solution.

To formally encode a user's query as additional constraints in the MIP problem, we adopt the classification proposed by Georgara et al. [76].

- **enforce** constraints to express that some value assignments that do not hold in an optimal solution are required to hold in the user-desired solution; and
- **veto** constraints to express that some value assignments that do hold in an optimal solution must not hold in the user-desired solution.

Following such classification, Table 6.2 lists possible query templates for the DPP. The table also shows the translations of queries into (linear) query constraints. While the table focuses on the DPP, the process of (i) identifying queries, (ii) defining query templates, and (iii) translating them into query constraints is general and can be applied to other MIP problems. We show the translation of the queries for the other problems addressed in this thesis in the appendix.

To illustrate how user queries are systematically translated into formal query constraints within our approach, we now discuss concrete examples based on the DPP (see Table 6.2).

- Query Q1 "*Why is subject u planned at semester l?*" translates into **vetoing** the planning of subject u at semester l (i.e., $x_{u,l} = 0$) to enforce an alternative solution to the optimal solution. Notice that this query refers to a **single assignment** occurring in the optimal solution.
- To comply with query Q7 "*Why is the group of subjects U planned at semester l?*" we must **veto** the planning of the entire set of subjects U at semester l (i.e., $\sum_{u \in U} x_{u,l} < |U|$). In this case, the query refers to the planning of multiple subjects, i.e., **multiple assignments**, in the optimal solution.
- When addressing query Q2 "*Why is subject u **not** planned to be completed at semester l?*" we must **enforce** the hypothetical planning of unit u, which the current (optimal) degree plan disregards.

- To answer the query Q8 "*Why is subject u planned to be completed at semester l instead of subject u'?*", we must: (1) enforce subject u'; and (2) veto u at semester u. Thus, we consider combinations of **enforce** and **veto** constraints to answer more specific queries about value assignments that did and did not occur.

6.2.2 The User-Desired Satisfiability Problem

Since numerical comparisons are not desirable in explanations [126], we propose instead to find the reasons why a user query, which challenges the optimal solution by posing a user-desired scenario, cannot lead to a better solution than the optimal solution. To do so, we cast the problem of obtaining the user-desired solution as a CSP. Such a CSP contains the problem constraints, the query constraints, and an additional constraint enforcing a solution at least as good as the optimal solution to the "*main*" problem. This formulation will serve to extract the constraints that restrict us from obtaining a user-desired solution with optimal value.

Definition 6.2.1 (User-Desired Satisfiability Problem)**.** *Let M be a MIP, a set of query constraints C_Q associated with a query, and $f^* = f(\boldsymbol{x}^*)$ the value of the optimal solution $\boldsymbol{x}^*$ for M. The* User-Desired Satisfiability Problem *(UDSP) for M is a CSP composed of the following constraints:*

$$(i)\ f(\boldsymbol{x}) \leq f^*, \quad (ii)\ C, \quad (iii)\ C_Q, \tag{6.2}$$

where $\boldsymbol{x}$ are variables and (i) is a minimality constraint.[3]

Example 6.2.3. *We define a UDSP for example 6.2.2 as the following CSP with variables $\boldsymbol{x}$:*

$$(i)\ f(\boldsymbol{x}) \leq f^*, \quad (ii)\ C, \quad (iii)\ x_{PF,4} = 1, \tag{6.3}$$

where (i) is the minimality constraint, (ii) are the main problem constraints, and (iii) is the query constraint.

[3]We consider M a minimisation problem. In maximisation problems, we impose a maximality constraint such that $f_M(\mathbf{x}) \geq f^*$.

Based on a user's query, the literature typically considers user-desired optimisation scenarios that accommodate such a query, aiming at quantifying how worse a user-desired solution would be in terms of optimality. In this context, the literature typically defines an extended optimisation problem, adding the query constraints to the constraint set [211, 76]. Then, it solves the extended problem and compares its solution with respect to the optimal one. Notice that we neither solve an extended MIP problem that accommodates a user's query (as in [211]) nor quantify how worse the user-desired solution is (as in [76]). In contrast, we find the reasons why the query posed by the user leads to worse solutions. In the next section, we discuss how to compute such reasons from the constraints of the UDSP.

6.2.3 Computing the Infeasible Constraints

Having defined the UDSP, we extract the minimal set of constraints that prevent us from finding a feasible solution to the UDSP, i.e., we extract the IIS. Notice that there might be multiple IISs corresponding to the same set of constraints since the number of IISs increases exponentially with the size of a constraint set [36]. More precisely, different IISs (despite being all "irreducible") could contain different numbers of constraints (i.e., some IISs might be smaller than others). Hence, this would subsequently lead us to explanations of varying sizes, since we build our explanations based on the constraints of an IIS. Following [161],[4] we should aim to compute the smallest possible IIS because the more concise an explanation, the easier it is to understand.

On the one hand, some algorithms in the literature can compute the "smallest" IIS [117, 98]. However, computing the "smallest" IIS is very costly. On the other hand, off-the-shelf solvers such as CPLEX use specialised heuristics to efficiently compute a single IIS that is not guaranteed to be the smallest one. In the next chapter, in Section 7.2.2, we experimentally compare the solutions computed by an algorithm for computing the smallest IIS, adapted from [98], and CPLEX. We observe that, in

[4]The number of "rules" (i.e., constraints in our case) in an explanation is a commonly used metric to evaluate its simplicity.

Table 6.3: IIS from Example 6.2.3. We indicate the constraint types (in different colours) and the elements of each constraint.

Id	Type	Values
q	Query	Programming Fundamentals, Semester 4
c_1	Completion	Programming Fundamentals
c_2	Offer	Programming Techniques **not offered** in Autumn
c_3	Offer	Object Oriented Programming **not offered** in Autumn
c_4	Offer	Operating System Programming **not offered** in Spring
c_5	Precedence	Prog. Fundamentals **before** Prog. Techniques
c_6	Precedence	Prog. Fundamentals **before** Object Oriented Prog.
c_7	Precedence	Prog. Tech. **or** Obj. Oriented Prog. **before** Op. Syst. Prog.
c_8	Core	Operating System Programming

practice, CPLEX computes the smallest IIS for most instances with a significantly lower computational cost.

Having computed the IIS, the resulting IIS might give us information about the UDSP instance at hand. If an UDSP is **feasible**, i.e., at least one solution exists, the alternative proposed by the user is as good as the optimal solution of the main problem. In such a case, there exists no set of inconsistent constraints, i.e., $IIS(C_{UDSP}) = \emptyset$, where C_{UDSP} is the set of constraints of an UDSP.[5] If the UDSP is infeasible, i.e., no solution exists, we differentiate two scenarios:

- **Query Infeasibility**. In this case, the query constraints are incompatible with the main problem restrictions. As a result, some query constraints conflict with the problem constraints in (6.2), causing infeasibility. Consequently, the IIS contains a set of query constraints and main problem constraints, i.e., $IIS(C_{UDSP}) = C'_Q \cup C'$ with $C'_Q \subseteq C_Q$, $C' \subseteq C$, and $C'_Q, C' \neq \emptyset$.
- **Minimality Infeasibility**. Here, some query constraints conflict with some main problem constraints, and the minimality constraint in (6.2). More intuitively, while the user's query is compatible with the problem restrictions, it leads to a worse solution in terms of optimality. Consequently, the IIS contains a set of query constraints, main problem constraints, and the minimality constraint.

[5] We denote $IIS(C)$ the function computing an IIS of constraint set C.

Table 6.4: *IIS* from Example 6.2.3. We indicate the constraint type (in different colours) and the mathematical encoding of the constraints.

Id	Type	Constraint Encoding
q	Query	$x_{PF,4} = 1$
c_1	Completion	$\sum_{l=1}^{6} x_{PF,l} \leq 1$
c_2	Offer	$\sum_{l=1,3,5} x_{PT,l} = 0$
c_3	Offer	$\sum_{l=1,3,5} x_{OOP,l} = 0$
c_4	Offer	$\sum_{l=2,4,6} x_{OSP,l} = 0$
c_5	Precedence	$x_{PF,1} \geq x_{PT,2}, \sum_{l=1}^{3} x_{PF,l} \geq x_{PT,4}$
c_6	Precedence	$x_{PF,1} \geq x_{OOP,2}, \sum_{l=1}^{3} x_{PF,l} \geq x_{OOP,4}$
c_7	Precedence	$\sum_{l=1}^{2} \sum_{u=\{PT,OOP\}} x_{u,l} \geq x_{OSP,3}, \sum_{l=1}^{4} \sum_{u=\{PT,OOP\}} x_{u,l} \geq x_{OSP,5}$
c_8	Core	$\sum_{l=1}^{6} x_{OSP,l} = 1$

Example 6.2.4. *Following example 6.2.3, Table 6.3 shows the IIS for the UDSP in equation (6.3). In this example, the IIS contains 8 main problem constraints (1 completion constraint, 3 offer constraints, 3 precedence constraints, 1 core constraint) and the query constraint. Hence, since the minimality constraint is not within the IIS, this case belongs to the Query Infeasibility category. For completeness, Table 6.4 shows the mathematical encoding of the constraints within the IIS.*

6.2.4 Constructing the *Graph of Reasons*

Having established how we compute the set of infeasible irreducible constraints (i.e., the IIS), we now address the challenge of relating and providing a semantic interpretation to the constraints within an IIS in a way that enables the LLM to generate more

Table 6.5: Natural Language Templates for each type of DPP constraint discussed in Section 5.2.

(i) Completion (5.3):	"*Subject u cannot be planned more than once*"
(ii) Offer (5.5):	"*Subject u cannot be planned at autumn/spring semesters*"
(iii) Credits (5.8):	"*d_l credits must be planned at semester l*"
(iv) Precedence (5.11):	"*Subject h must be planned before Subject j*"
(v) Core (5.13):	"*Subject u is compulsory*"
(vi) Major (5.14):	"*Students must complete at least one major*"
(vii) Major core (5.17):	"*Subject u is compulsory to complete major M*"
(viii) Eligible (5.20):	"*$n_{SM^{elig}}$ eligible subjects from the set of subjects U must be complete to obtain major M*"

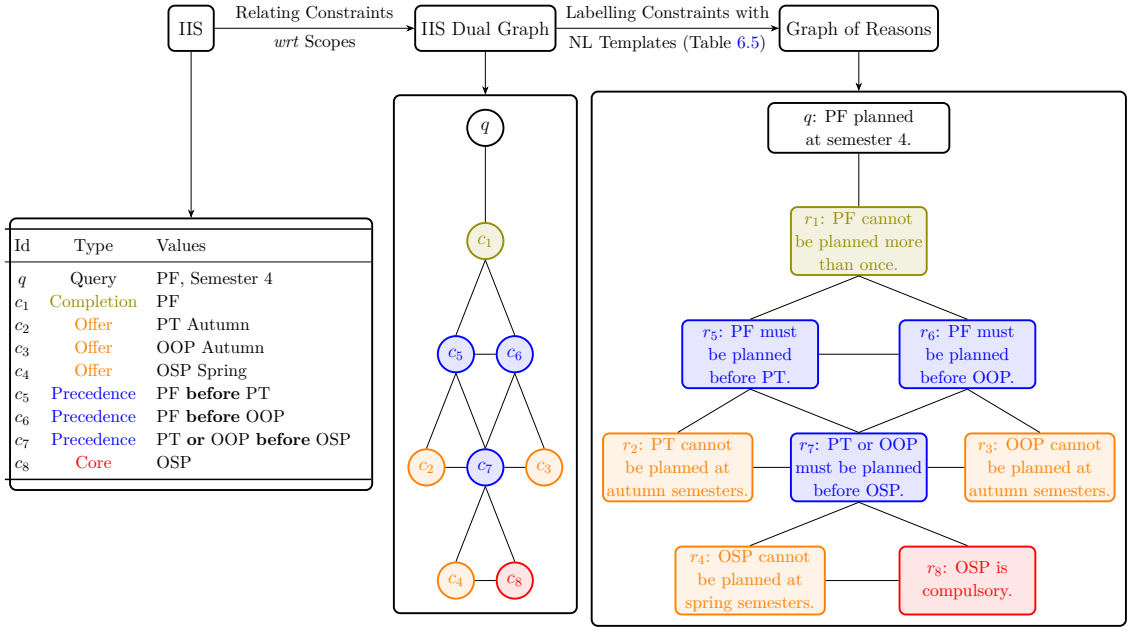


Figure 6.2: Process to build a graph of reasons applied to our running example: (1) relating constraints within the IIS to yield its dual graph, and (2) labelling constraints in the dual graph to build the graph of reasons.

precise and meaningful explanations —ones that convey why the user query cannot be satisfied. This is necessary because, as we later demonstrate in our experiments, inferring the relationships between constraints from an IIS to build explanations is not an easy *reasoning* task for an LLM. Indeed, LLMs might hallucinate and generate factually incorrect explanations when tackling such a complex task because they have limitations with abstract reasoning [75] and summarisation [4].

Along these lines, we show that, by exploiting the structural relations among constraints, we can organise them in a graph, the so-called *graph of reasons*, whose structure eases the understanding of the conflicts between constraints within the IIS (in contrast with the unstructured list of constraints in the IIS). Ultimately, our purpose is to use such a graph as a prompt to make the knowledge inherent in the constraints available to the LLM. This enables the LLM to generate explanations that are more complete and readable.

To this end, we rely on well-known concepts from the Constraint Reasoning literature, as explained hereafter. Hypergraphs [57] represent how a set of constraints relates to decision variables. A *hypergraph* $\mathcal{H} = (V, S)$ is a pair where V is a set of

vertices, and $S = \{S_1, \ldots, S_l\}$, $S_i \subseteq V$, is a set of subsets of V called hyperedges. Notice that the vertices represent the set of decision variables of our model, and a hyperedge S_i is the scope of constraint c_i, which is the subset of decision variables involved in the constraint. However, hypergraphs do not directly represent the relations among constraints, which is, instead, the purpose of the concept of dual graphs. Formally, a *dual graph* [57] $D = \{S, I\}$ is an undirected graph whose vertices S are the hyperedges of $\mathcal{H}$ and $(S_i, S_j) \in I$ iff $S_i \cap S_j \neq \emptyset$. Intuitively, each node of a dual graph corresponds to a constraint, and each edge indicates that two constraints share some variables. Therefore, the dual graph of an IIS captures the relations between its constraints.

Figure 6.2 showcases our approach for building the graph of reasons from the IIS in Table 6.3. For instance, constraint c_1, which ensures that Programming Fundamentals is planned at most once, is related to the query constraint enforcing the planning of Programming Fundamentals in the fourth semester. Thus, both constraints restrict the assignment of the same subject, Programming Fundamentals. More formally, both constraints are related since they share decision variables, hence there is an edge connecting such constraints in the graph of reasons. In contrast, other constraints such as c_2, which vetoes to plan Programming Techniques in Autumn Semesters, are not *directly* related to the query constraint. Nonetheless, c_2 is linked to the precedence constraint c_5, which ensures that Programming Fundamentals is planned before Programming Techniques. Since c_5 is also related to c_1, an indirect relationship between c_2 and the query constraint can be established through this chain of dependencies, formally represented by a path of edges between c_2 and the query constraint.

After having established the dual graph of constraints, following the approach by [150], we use NL templates to provide a semantic interpretation to each constraint in such a graph, as shown in Figure 6.2.[6] Ultimately, a "*reason*" takes the form of the semantic interpretation of a constraint by means of a NL template. Therefore, we

[6] Notice that translating queries and labelling constraints using NL templates are the only problem-dependent steps that must be carried out by an expert, in contrast with the rest of our approach, which is domain-independent. As an example, Table 6.5 shows the NL templates involved in our running example for each type of DPP constraints.

refer to the dual graph of the semantically-characterised constraints as the *graph of reasons*.

A desirable property that characterises the graph of reasons is that it is not "disconnected", i.e., it involves reasons that are always related to each other. This property can be directly expressed in terms of the graph-theory concept of *connectedness*, as formalised by Theorem 6.2.1.

Theorem 6.2.1. *Given an IIS, the IIS dual graph D is connected.*

Proof. We prove by contradiction that a disconnected IIS dual graph, i.e., a graph with multiple disconnected components, is not possible to obtain due to the infeasible and irreducible properties of an IIS (Definition 2.2.3).

Consider an IIS graph that is disconnected in two subgraphs, $A = (C_A, E_A)$ and $B = (C_B, E_B)$ such that

$$C_A \cup C_B = IIS,$$
$$C_A \cap C_B = \emptyset,$$
$$S_A \cap S_B = \emptyset,$$

where S_A, S_B are the scopes of the set of constraints C_A and C_B respectively. Notice that, since the graph is disconnected, A and B subgraphs do not share variables, i.e., $S_A \cap S_B = \emptyset$. Then, since the subset of constraints C_A and C_B are (disjoint) subsets of the IIS, they are feasible (see Definition 2.2.3). However, since both subsets of constraints have different variables (different domains), the union of both constraint sets would not affect the feasibility status of the resulting system of constraints. As a result, we would have a feasible IIS, which is a contradiction in itself. Therefore, it is not possible to have a disconnected IIS graph. □

6.2.5 Providing Textual Explanations with LLMs

LLMs have emerged as a powerful tool for a wide range of tasks, including, for example, machine translation, reasoning, summarisation, and question-answering

[87, 213]. However, it is well-known that LLMs present limitations with abstract reasoning [75] and summarisation [4].

As far as we are concerned, a consequence of these limitations is that LLMs' responses tend to take the implicit assumption made in the user prompt for granted, rather than independently verifying their truthfulness.

To illustrate this phenomenon, we revisit the degree plan from Example 6.2.1 and consider the following user query: "*Why is Programming Fundamentals not planned at semester 3?*". Notice that, in contrast with the query in Example 6.2.2 "*Why is Programming Fundamentals not planned at semester 4?*" —suggesting an invalid alternative solution (i.e., one that violates some hard constraints)—, this query suggests an alternative solution that is equally valid and optimal, i.e., a solution that does not violate any constraint and that has the same optimal total utility. In this case, our approach yields an empty IIS, correctly representing the fact that the user-suggested alternative is equally good.[7] In contrast, when straightforwardly prompting an LLM to generate an explanation in response to the query at hand, the LLM produces the following explanation:[8] "*Programming Fundamentals is not planned in semester 3 because doing so would violate prerequisite timing for other subjects that are scheduled earlier in the plan...*" Notice that the LLM implicitly assumes that the suggested alternative is indeed invalid, without independently verifying the truthfulness of this assumption. In fact, LLM are notoriously reluctant to contradict the user, a phenomenon —known as "*sycophancy*" [174, 143]— that can lead to false statements, as in this case. Notice that this aspect is particularly important in the context of explaining the solutions of optimisation problems, for which multiple, equally good solutions may exist, as discussed in Section 6.2.3.

Against this background, our methodology, which provides the information encoded by the IIS as a structured, semantically-characterised graph, as discussed in the above section, is key to guiding LLMs in generating *correct* textual explanations.

[7] In such cases, the explanation provided by LLMxMIP consists of indicating that the alternative suggested yields a degree plan that is as good as the optimal one. Additionally, we can provide a solution compliant with the user query by solving the resulting UDSP discussed in Section 6.2.2.

[8] For this test, we provided the LLM only with the problem solution and the instance data. We show the first sentence of the explanation for the sake of brevity.

In practice, we build a prompt that uses the graph of reasons as an input to the LLM to generate such an explanation. We employ a simple zero-shot prompt strategy [164] that contains the following information.

i. A **context** of the optimisation problem to solve. In our case, a description of the DPP, its goal, and its restrictions. Additionally, we provide the optimal solution, i.e., the optimal degree plan.

ii. The **question** posed by the user.

iii. A **textual encoding** of the graph of reasons. As the graph of reasons is a visual representation, we include a textual encoding of this graph in the prompt. This encoding specifies the node labels —representing the reasons— and the relationships between them, corresponding to the graph's edges.

iv. The **instructions** of the task to do, i.e., writing a structured explanation using the encoding of the graph of reasons.

A myriad of prompt engineering methods has been discussed in the literature. For instance, techniques such as chain-of-thought prompting [194] and prompt ensembling [214] have demonstrated noteworthy improvements in tasks requiring multi-step reasoning or factual accuracy. Contrarily, for other more general tasks such as translation, zero-shot prompts can significantly outperform few-shot prompts due to semantic contamination from the examples (shots) [156]. Additionally, minor variations in phrasing or formatting can lead to substantial differences in model outputs across a wide range of tasks [194, 214]. This variability highlights the importance of prompt engineering.

Despite the variety of prompt engineering methods discussed in the literature, we adopt a zero-shot prompting strategy, as the scope of this chapter is to showcase how LLMs can be leveraged to provide explanations, rather than to conduct an extensive evaluation of different prompting strategies.

Example 6.2.5. *Considering the computed IIS and its graph of reasons in Figure 6.2, we provide the textual explanation generated with a LLM to answer the question* "Why is Programming Fundamentals not planned at semester 4?"*. The resulting ex-*

planation is the following.[9] "Programming Fundamentals is not planned in semester 4 because it must be taken earlier to satisfy the prerequisite requirements of later subjects, particularly Programming Techniques and Object Oriented Programming. These subjects, in turn, need to be completed before taking Operating Systems Programming, a compulsory subject that can only be taken in the autumn. Additionally, Programming Techniques and Object Oriented Programming cannot be scheduled in the autumn semesters, further constraining their placement. Since subjects cannot be planned more than once, Programming Fundamentals must be scheduled early enough to enable this sequence, making semester 4 too late for it."

Notice that the explanation in Example 6.2.5 is correct, i.e., there are no imprecisions or mistakes, and complete, it mentions all the reasons in the graph. In B.3, we discuss the exact prompt used in this example.

6.3 Experimental Evaluation

In this chapter, the goal of the experimental evaluation is to evaluate whether our approach can produce explanations that are complete and easy to understand. In the next chapter, we further evaluate LLMxMIP in terms of the empirical hardness of computing our explanations, allowing us to characterise which problem instances can be tackled in real-time for users (Section 7.2).

6.3.1 Experimental Setting

We now describe our experimental setting, including the considered optimisation -based benchmark (Section 6.3.1.1) scenarios and the methodology used to conduct our experiments (Section 6.3.1.2).

6.3.1.1 Optimisation-Based Benchmark Scenarios

We test our approach in four different optimisation problems. In detail, we evaluate LLMxMIP in the three optimisation domains tackled in this thesis, the DPP, the RPP and the aggregation of value systems. Moreover, we validate our approach

[9] We used *gpt-4o* to compute textual explanations.

by testing it on challenging instances of a well-studied problem of the optimisation literature, the RCPSP. We do so to showcase its generality and applicability to real-world problems. Furthermore, in the next chapter, we evaluate its scalability with computationally hard problem instances typically studied in the optimisation literature. In this section, we detail the problem instances used in our experimental evaluation.

DPP In this experimental section, we employ the problem instance described in Section 5.4.1. In detail, we evaluate our approach on the Bachelor's Degree in Information and Communication Technology of the Western Sydney University. For such a Bachelor's Degree, we computed a customised degree plan for each of 118 target jobs from the real-world *APS Digital Career Pathways* dataset [13].

RPP Similarly to the DPP case, we evaluate our explainability approach using the RPP instance introduced in Section 4.5. In this analysis, we focus on the *sufficient budget scenario*, in which each project's budget matches its original allocation. We examine three configurations of the objective-function parameters α, β, and γ in Equation (4.9). Specifically, we fix $\alpha = 1.0$ and $\gamma = \infty$ (corresponding to the hard-constrained setting), and vary $\beta = \{0.0, 10^{-5}, 1.0\}$. Finally, for the sake of simplicity, we do not incorporate uncertainty in the RPP instance used in this evaluation.

Aggregation of Value Systems In the case of aggregating value systems, presented in Section 3.6, we relied on two different case studies to demonstrate the applicability of our aggregation approach across multiple domains. However, since our goal here is to evaluate the quality of the explanations generated for the value-system aggregation problem, we focus, for brevity, on a single domain: the European Value Study. We analyse six consensus value systems obtained by combining three ethical principles —the utilitarian principle ($p = 1$), the egalitarian principle ($p = \infty$), and the mean principle ($p = 2$)— with two weighting configurations: one that weights each country by its population and another that assigns equal weight to all countries.

RCPSP Among the many variants of the RCPSP [91], we consider the single-mode RCPSP, where each activity can only be completed in one way. The problem instances used come from the *Project Scheduling Problem Library* (PSPLIB) [108].[10] Such a library employs ProGen [109], an algorithm that generates problem instances with different parameters such as number of activities, network complexity, or resource strength. We use 1440 instances with $|J| \in \{30, 60, 90\}$, generated with 48 combinations of parameters (see [108]) and 10 different seeds.

6.3.1.2 Experimental Methodology

As our first step, we solve the benchmark problem instances obtained as described in Section 6.3.1.1. We then generate queries as described in Section 6.2.1. Specifically, we described 10 different query types for the DPP. In Section B.2 of the appendix, we enumerate 10, 4, and 8 query types for the RPP, the aggregation of value systems, and the RCPSP, respectively. In greater detail, for the DPP, we generate five different queries for each type of query, i.e., a total of 50 queries for each optimal solution. Since we use a dataset of 118 jobs (and each job has a different optimal solution), we generate a total of 5900 queries. In the case of the RPP, we generate ten queries for each query type, resulting in a total of 100 queries per optimal solution. Given that we consider three different parameter configurations, this amounts to a total of 300 queries. In the case of value system aggregation, we generate five distinct queries for each component of the consensus solution. In this case study, the consensus consists of four action–judgement functions[11] and one value preference, resulting in 25 queries per consensus solution. Since we analyse six consensus value systems, this yields a total of 150 queries. Finally, for the RCPSP, we generate one query per query type for each problem instance. Since we consider 1440 problem instances, this results in a total of 11520 queries. Overall, across all optimisation problems considered in this experimental section, we generate a total of 17900 queries.

Once the queries are generated, we generate their corresponding UDSPs according

[10] https://www.om-db.wi.tum.de/psplib/. Last consulted: April 2024.
[11] In this case study, we consider two values and two actions.

to Section 6.2.2, and we calculate their IISs according to Section 6.2.3. Finally, we compute the graphs of reasons according to Section 6.2.4.

We solve all instances and compute IISs with CPLEX v22.1.0.[12] We run our tests on a machine with a 10-core 2.2GHz CPU and 16GB of RAM. We limit solving time to 10 hours per problem instance and per explanation.

Then, we proceed to generate our LLM-generated textual explanations that answer the above-mentioned queries. Since we aim to evaluate whether the structure conveyed by the graph of reasons is beneficial to the LLM,[13] we prompt it with two different strategies. On the one hand, we employ a prompt that provides the constraints of the IIS semantically characterised with natural language templates, but without any structure. On the other hand, we use a prompt that contains a textual encoding of the graph of reasons. In detail, this encoding specifies the graph's nodes, i.e., the constraints of the IIS semantically characterised with natural language templates (as in the prompt with only the constraints of the IIS), and the graph's edges, i.e., the relations between the constraints in the IIS. We show examples of both types of prompts in the appendix.

To quantify the benefits of using the graph of reasons as input to the LLM, we compute two classic NL metrics on the generated explanation:

- *completeness*: the percentage of reasons in the explanation with respect to the number of reasons in the IIS related to the query; and
- *readability*: difficulty in understanding the text of an explanation.

In general, to compute the readability of an explanation, we use the *Gunning Fog Index* (GFI) [85], which estimates the years of formal education a person needs to understand a text. Additionally, we count the number of words in an explanation as a measure of its length.

To conduct the above-mentioned readability and completeness analysis, we select a

[12] Code for the experimental evaluation available at: https://github.com/RogerXLera/LLMxMIP.

[13] We use *gpt-4o* (https://openai.com/es-ES/index/hellox-gpt-4o/) to generate our textual explanations.

Table 6.6: Explanations generated using a prompt with the graph of reasons (left) and a prompt with the IIS (right) to answer the query of the running example: "*Why is Programming Fundamentals not planned at semester 4?*". We highlight in *blue* the reasons.

Programming Fundamentals is not planned in semester 4 because *it must be taken earlier to satisfy the prerequisite requirements of later subjects*, particularly *Programming Techniques and Object Oriented Programming*. *These subjects*, in turn, *need to be completed before taking Operating Systems Programming*, *a compulsory subject* that *can only be taken in the autumn*. Additionally, *Programming Techniques* and *Object Oriented Programming cannot be scheduled in the autumn semesters*, further constraining their placement. Since *subjects cannot be planned more than once, Programming Fundamentals* must be scheduled early enough to enable this sequence, making semester 4 too late for it.	Programming Fundamentals is scheduled in semester 2 rather than semester 4 to comply with prerequisite requirements and ensure a feasible progression through the curriculum. Since *this subject must be taken before others like Programming Techniques and Object Oriented Programming*, delaying it to a later semester would prevent the student from enrolling in these dependent courses on time. Additionally, university rules may restrict when certain subjects can be taken, and *each subject can only be planned once, so placing Programming Fundamentals* earlier allows for a coherent and complete academic path within the 6-semester limit.

Table 6.7: Reasons mentioned in the explanations (shown in Table 6.6) generated using a prompt with the graph of reasons and a prompt with the IIS.

Reasons	Graph prompt	IIS prompt
r_1: PF cannot be planned more than once	✓	✓
r_2: PT cannot be planned at autumn semesters	✓	✗
r_3: OOP cannot be planned at autumn semesters	✓	✗
r_4: OSP cannot be planned at spring semesters	✓	✗
r_5: PF must be planned before PT	✓	✓
r_6: PF must be planned before OOP	✓	✓
r_7: PT or OOP must be planned before OSP	✓	✗
r_8: OSP is compulsory	✓	✗

representative sample of different queries both for the DPP and the RCPSP, such that (i) it contains different types of query constraints, and (ii) the IIS computed from an UDSP is not the empty set.[14] More specifically, we selected 300 distinct queries for both the DPP and the RCPSP to compare two types of explanations: those generated using a graph of reasons and those based on an IIS.

6.3.2 Completeness and Readability Results

Results show that providing the graph of reasons to an LLM leads to several benefits when generating textual explanations. As an illustrative example for the DPP, Table 6.6 shows the explanations generated with and without a graph of reasons. Additionally, Table 6.7 shows which reasons out of the IIS are mentioned in each explanation. Observe that the explanation using the graph of reasons in the prompt is more complete since it mentions all the reasons, while the one using the IIS only mentions three (out of eight).

Such an example illustrates a general trend observed in the results presented in Table 6.8, which reports the average completeness, the average GFI, and the average number of words of 65 pairs of explanations (one generated using the graph of reasons prompt and the other using the IIS prompt) for both problem domains.

As for the DPP, the average completeness is 84% for explanations generated with a graph of reasons, while the average completeness for explanations generated with an IIS is 64%. This represents a statistically significant gain of 20% when using the graph of reasons as a prompt. We also notice that the more complete explanations, thanks to graphs of reasons, are also longer, as shown in Table 6.8. Indeed, explanations generated with a graph of reasons prompt are 32% longer (20 words more) on average than explanations generated with an IIS prompt. Nonetheless, although an explanation generated from the graph of reasons uses, on average, more words, its readability (GFI) is similar to explanations generated using the IIS.

As for the RPP, explanations generated using a graph of reasons achieve a higher average completeness (93%) than those generated using an IIS prompt (86%). This corresponds to a statistically significant improvement of 7% in completeness when employing the graph of reasons. With respect to explanation length, the number of words is comparable across the two prompt types: explanations generated with a graph of reasons contain, on average 61 words, while those generated with an IIS prompt contain 59 words, and the difference is not statistically significant. Similarly,

[14] If the IIS is empty, the UDSP is feasible, which indicates that the user-desired solution is also optimal. In such a case, the explanation mentions that the proposed optimisation scenario is also optimal; thus, it is possible to fulfil such a query.

Table 6.8: Average completeness, number of words, and GFI per explanation for both prompt types.

Problem	Metric	Graph Prompt	IIS Prompt	$p-$value
DPP	Completeness (%)	84 ± 3	64 ± 4	$2.4 \cdot 10^{-8}$
	Number of words	82 ± 4	62 ± 2	$2.1 \cdot 10^{-12}$
	GFI	20.8 ± 0.3	20.9 ± 0.3	0.86
RPP	Completeness (%)	93 ± 2	86 ± 2	$1.5 \cdot 10^{-2}$
	Number of words	61 ± 2	59 ± 1	$1.1 \cdot 10^{-1}$
	GFI	18.4 ± 0.3	18.1 ± 0.3	0.82
Value System	Completeness (%)	56 ± 3	55 ± 3	$7.6 \cdot 10^{-1}$
	Number of words	76 ± 2	66 ± 1	$3.3 \cdot 10^{-8}$
	GFI	18.0 ± 0.3	17.6 ± 0.3	$5.9 \cdot 10^{-2}$
RCPSP	Completeness (%)	85 ± 3	69 ± 4	$2.1 \cdot 10^{-7}$
	Number of words	107 ± 4	85 ± 3	$9.7 \cdot 10^{-13}$
	GFI	15.6 ± 0.2	16.5 ± 0.3	$5.7 \cdot 10^{-3}$

the readability of the explanations, as measured by the GFI, remains essentially unchanged across prompt types, with average values of 18.4 for the graph of reasons and 18.1 for the IIS prompt. These results indicate that, for the RPP, providing a graph of reasons improves completeness without increasing explanation length or reducing readability.

As to the RCPSP, the average completeness is 85% for explanations generated with a graph of reasons prompt, while the average completeness for explanations generated with a IIS prompt is 69% (therefore, the gain in completeness is 16%). On the other hand, in contrast with the DPP case, for the RCPSP the readability index is significantly better for explanations generated with a graph of reasons prompt than explanations generated with an IIS prompt. Indeed, providing the graph of reasons results in explanations that are easier to understand (GFI = 15.6) with respect to those generated with the IIS (GFI = 16.5).[15]

In the case of aggregating value systems, the average completeness of explanations generated using a graph of reasons (56%) is comparable to that of explanations generated using an IIS prompt (55%), with no statistically significant difference between the two approaches. This suggests that, for this problem, providing a graph of reas-

[15]Explanations generated using a graph of reasons contain more concise sentences, which is an indicator directly considered to compute the GFI.

ons does not yield a noticeable improvement in completeness. However, explanations generated with a graph of reasons are significantly longer, with an average length of 76 words, compared to 66 words for explanations generated with an IIS prompt. This corresponds to an average increase of approximately 15% in explanation length. Despite this increase in length, the readability of the explanations remains similar across prompt types, as indicated by comparable GFI values (18.0 for the graph of reasons and 17.6 for the IIS prompt). Overall, these results indicate that, for the Aggregation of Value Systems, the use of a graph of reasons leads to longer explanations without a corresponding gain in completeness or a loss in readability. We observe that the graph of reasons is less effective for this problem because, in contrast to the other problems considered in this thesis, its formalisation does not involve explicit constraints beyond those introduced to encode the objective function. In particular, the aggregation of value systems is formulated solely as the minimisation of a distance function. Although for the cases where $p = 1, 2, \infty$, the problem can be encoded as a constrained MIP, all constraints arise from the linearisation of the objective through the introduction of an auxiliary variable. As a consequence, the extracted IISs always exhibit the same structure: they consist of the query constraint, a set of constraints bounding the auxiliary variable, and the minimality constraint. The constraints bounding the auxiliary variable reflect the individual value systems of the countries that prevent the consensus value system proposed in the user query from being optimal. In a nutshell, because the structure of the problem induces IISs with limited variability and explanatory power, the graph of reasons does not provide additional insight for the aggregation of value systems, unlike the other problems addressed in this thesis, where rich and heterogeneous constraints play a central role.

Now, to better understand how using a graph of reasons affects the completeness of the explanations, Figure 6.3 shows the aggregated average completeness for all the addressed domains. We report the average completeness across different explanation groups, categorised by size —defined as the number of reasons in the graph of reasons (and the corresponding IIS): (i) small (fewer than five reasons), (ii) medium (between five and ten reasons), and (iii) large (more than ten reasons). For small explanations

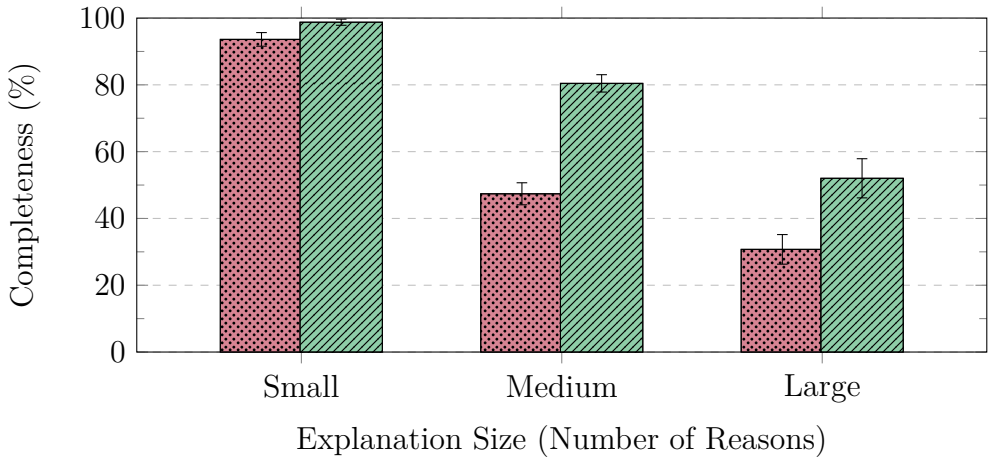


Figure 6.3: Average percentage of reasons mentioned in the explanations generated using the IIS (red) and using the graph of reasons (green). We group the explanations into three groups depending on the number of reasons: (i) small (less than five reasons), (ii) medium (between five and ten reasons), and (iii) large (more than ten reasons).

(i.e., with less than five reasons), the LLM can generate complete explanations even without a graph of reasons. However, for medium and large explanations (i.e., more than five reasons), the completeness drops significantly when the LLM is given only an IIS. In more detail, for instances with a medium number of reasons (between five and ten reasons), the explanations generated using a graph of reasons mention $\sim 30\%$ more reasons on average than the explanations generated with only an IIS. Similarly, large explanations (> 10 reasons) using a graph of reasons mention $\sim 20\%$ more reasons on average than the explanations generated with an IIS.

6.4 Conclusions

In this chapter, we have tackled the fundamental question of explaining the outcomes of the optimisation processes that drive *Decision Support System* (DSS) in many real-world application domains. Properly explaining these outcomes and answering the questions posed by the users is indeed fundamental to guarantee the necessary human oversight required by modern AI international regulations. To achieve this goal, we have shown how we leveraged the recent advances of LLMs within our approach, namely LLMxMIP. However, despite their impressive performance across various language tasks, LLMs still struggle with abstract reasoning and

are prone to producing incorrect and uninformative explanations. To overcome this, within LLMxMIP we employ constraint reasoning techniques to distil the information necessary to produce the explanation, which is then passed as input to the LLM as the so-called *graph of reasons*. Such a graph captures both the semantics and relationships between constraints of the optimisation problem.

We evaluate our approach on the three different optimisation problems addressed in this thesis to showcase its generality. Additionally, to showcase its applicability to a well-studied problem in the literature, we evaluate our approach on realistic instances of the *Resource-Constrained Project Scheduling Problem*. and scalability. Our results demonstrate that providing the graph of reasons to an LLM significantly improves the quality of the generated textual explanations in terms of completeness, for all the evaluated problems except the aggregation of value systems. This is due to the lack of variety in constraints that restrict the solution to that problem, which consists of minimising a distance function. However, for this problem domain, our approach is still capable of generating explanations mentioning the reasons why the recommended solution is better than the user-suggested solution.

In conclusion, the contributions of this chapter address the research questions concerned with explainability for optimisation-based decision support systems, namely Research Questions **Q6**, **Q6.1**, **Q6.2**, and **Q8**. The main contributions can be summarised as follows:

- **Question Q6:** Can we develop a general methodology to generate contrastive explanations that answer the questions posed by a user about the outcome of optimisation problems? We propose *Large Language Models for Explainable MIP* (LLMxMIP), a general methodology for explaining the outcomes of optimisation processes by answering users' contrastive questions. Our approach grounds explanations in the underlying optimisation model by explicitly reasoning over constraints and the user query, thereby enabling explanations that go beyond surface-level descriptions of the solution.
- **Question Q6.1:** How can we extract and structure the relevant information required to generate an explanation for a user question? We compute

the so-called *graph of reasons*, a structured representation that captures the relevant constraints and their relations, which play a role in justifying why alternative solutions suggested by the user are not possible or worse than the optimal solution. This representation distils the essential information required for explanation while preserving the semantics of the optimisation problem.

- **Question Q6.2:** Can we develop an explanatory methodology for MIPs that leverages LLMs to generate contrastive explanations? We show that LLMs can generate quality textual explanations when provided with structured input grounded on constraint reasoning. By using the graph of reasons as the input to the LLM, LLMxMIP mitigates common issues such as incorrect or trivial explanations, resulting in explanations that are more complete than those generated from unstructured inputs.

- **Question Q8:** Is our explanatory methodology general enough to construct explanations for the optimisation problems addressed in this thesis? We demonstrate that LLMxMIP is domain-agnostic and can be applied to a wide range of optimisation problems, including all problems studied in this thesis as well as the well-known RCPSP. Our experimental results confirm that the proposed method scales to realistic problem instances and generates complete explanations.

Chapter 7

On the Cost of Explanations

7.1 Introduction

Having proposed a methodology for generating textual explanations for optimisation-based decision-support systems in the previous chapter, we now evaluate the computational cost of producing these explanations. This need stems from the complexity of computing the IIS, a fundamental step in LLMxMIP. In practice, as shown in our experiments in Section 7.2, this step can range from a few milliseconds to hours, depending on the problem instance and the user query.

Consequently, this chapter focuses on an empirical analysis of the computational cost associated with generating explanations using LLMxMIP. In particular, we address **Research Question Q7.1** by characterising which queries, posed to challenge the recommendation of the DSS, can be answered with an explanation in real time (i.e., in under a minute) across different problem instances. Moreover, when computational costs are high, users may not receive an explanation immediately. In such situations, providing users with the estimated waiting time has been shown to increase users' tolerance for delays [136] and reduce frustration during system interaction [135]. Accordingly, the second objective of this chapter is to estimate the explanation generation time, thereby addressing **Research Question Q7.2**.

Since the empirical results presented in Section 7.2 show that explanations for all optimisation problems studied in this thesis can be generated in real time, we further evaluate our explainability approach on more challenging instances drawn from the combinatorial optimisation literature. In particular, we consider two representat-

ive families of combinatorial optimisation problems, namely scheduling and packing, in addition to the three application domains studied in this thesis. Specifically, we employ two well-studied benchmark problems: the *Resource-Constrained Project Scheduling Problem* (RCPSP), which has already been used in Section 6.3 to evaluate the quality of the generated explanations, and the *Winner Determination Problem* (WDP) from combinatorial auctions [56], which can be formulated as a multidimensional knapsack problem. By evaluating our approach on these problems, for which computing IISs and the resulting explanations are particularly demanding, as shown in Section 7.2, we assess the scalability and practical feasibility of our explainability methodology.

To answer research question 7.2, i.e., to predict the explanation generation time, we adopt a well-established technique from the literature that has proven effective for predicting solving times for combinatorial optimisation problems using ML models [96, 203, 204]. However, since the existing technique is designed to predict the time required to solve an optimisation problem, to estimate the explanation generation time, we adapt it by characterising the expected cost of computing an IIS with specific features of the optimisation problem instance [114] and a set of proposed features that capture the user query to be explained.

More specifically, we make the following contributions in this chapter:

- We empirically evaluate the computational cost of generating explanations with LLMxMIP. Our analysis investigates whether explanations can be produced in real time across a diverse set of optimisation problems, including challenging benchmark problems from the combinatorial optimisation literature. Our results show that despite the hardness of solving challenging optimisation problems, LLMxMIP computes explanations in less than 60 seconds in most cases.
- We build a predictor to estimate explanation generation time based on the features of the optimisation problem to be solved and the features of the query that triggers the explanation. Our results show that the *Root Mean Squared Error* (RMSE) of our predictions is 0.75 and 0.66 for the WDP and the RCPSP, respectively. This indicates that the predicted runtimes are, on average, close

to the observed explanation times, with errors that remain within a comparable time scale. Among the evaluated methods, random forest models achieve the best predictive performance, reducing prediction error by up to 19% compared to alternative approaches.

- Additionally, our results show that extending problem-instance features with query-specific features is crucial to obtain good estimations, since employing query-specific features helps reduce prediction error by up to 22% compared to using problem-instance features alone.

7.2 Empirical Hardness Evaluation

The main goal of this section is to evaluate the empirical hardness [114] of computing explanations. As a result, this will enable us to characterise the problem instances for which explanations can be computed in real time for users. Our study will measure (i) the time needed to compute IISs to build explanations, and (ii) the *time overheads* of building explanations with respect to solving optimisation problems. Thus, given a problem and a query, we define the *time overhead* to build an explanation for the query as the ratio between the time to compute the IIS and the time to compute the optimal solution for the problem. Therefore, our study compares the hardness of solving optimisation problems with that of computing their explanations.

Recall that generating an explanation involves additional steps beyond computing IISs (see Section 6.2), including the construction of a graph of reasons and the formulation of textual explanations using an LLM. However, the computational cost of generating textual explanations mainly depends on the cost of computing IISs. Indeed, while explanation generation relies on lightweight graph processing and a single inference step of a LLM, which can usually take a few seconds, computing an IIS can take hours for the hardest queries. Consequently, we consider only the IIS runtime when evaluating the empirical complexity of explanation generation.

An additional goal is to empirically learn whether the computational effort required to compute the smallest IIS with state-of-the-art techniques pays off. This would

be the case if we do obtain IISs that are significantly smaller than those computed by CPLEX,[1], which does not necessarily compute the smallest IIS. To evaluate the trade-off between efficiently computing IIS and extracting the smallest IIS, we compare the size (i.e., the number of constraints) of the IIS computed by CPLEX and the smallest IIS, as well as the respective runtimes of CPLEX and a state-of-the-art algorithm for computing the smallest IIS.

7.2.1 Experimental Setting

We now detail our experimental setting, including the description of the MIP problem instances used to provide explanations, the state-of-the-art algorithms employed, and the hardware used to run our experiments.

7.2.1.1 Problem Instances

For the experimental evaluation presented in this chapter, we employ the same problem instances and query instances described in Section 6.3.1.2 of the previous chapter.

The only additional optimisation problem considered in this chapter is the WDP, which we include to evaluate the scalability of our explainability approach on optimisation instances that are significantly harder than those addressed elsewhere in this thesis. The WDP is a well-studied benchmark from the combinatorial optimisation literature, which generalises other well-known problems such as the multi-dimensional knapsack problem [103]. For the sake of completeness, we provide the details of the WDP formalisation in Appendix A. Below, we briefly describe the WDP instances used in our experiments.

WDP We use the *Combinatorial Auction Test Suite* (CATS) generator [116] to create realistic instances of *Combinatorial Auction* (CA).[2] Specifically, we consider four commonly used bid-generation distributions, namely *paths*, *regions*, *matching*, and *scheduling*. We vary the number of bids as $|B| \in 20, 50, 100, 200, 500, 1000$. For each bid setting, we consider three corresponding settings for the number of goods,

[1]We employ CPLEX since it is the solver used throughout this thesis, but other off-the-shelf solvers, such as Gurobi [86], can compute IIS.

[2]https://github.com/kevinlb1/CATS.

resulting in a total of 18 distinct bid–good configurations. For every combination of distribution and bid–good configuration, we generate 100 independent problem instances. Overall, this results in a total of 7200 WDP instances. Once each problem instance is solved, we randomly generate two distinct queries from the query types list in Table B.6, i.e., we compute a total of 14400 IIS.

7.2.1.2 Implementation and Computational Setup

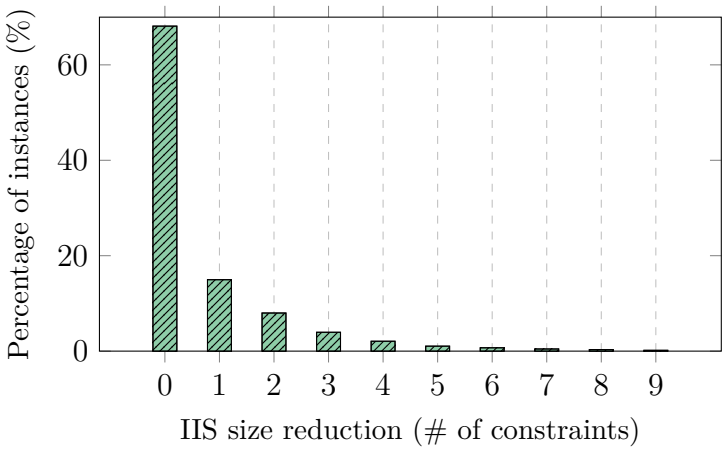
We solve all problem instances and compute IISs with CPLEX v22.1.0.[3] We also employ FORQES [98] to compute the smallest IIS because, to the best of our knowledge, it is the most recent algorithm for computing the smallest IIS.[4] We run our tests on a machine with a 10-core, 2.2 GHz CPU and 16GB of RAM. We limit solving time to 10 hours per problem instance and per explanation.

7.2.2 Is it worth computing the smallest IIS?

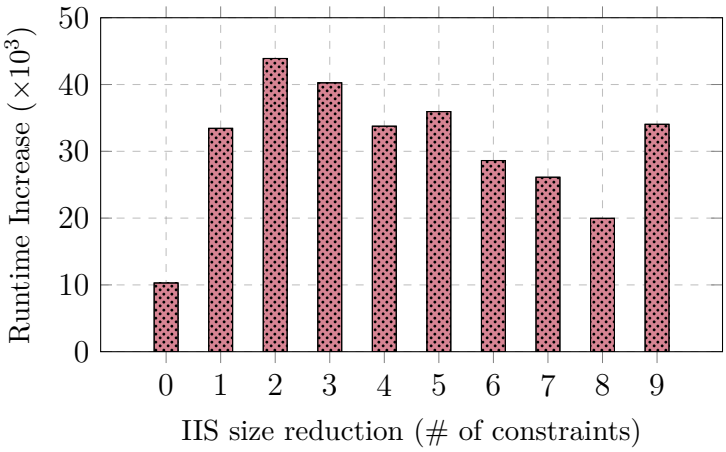
We compute the IISs resulting from all generated queries with CPLEX and FORQES, and we compare their runtimes as well as the difference in the sizes of the computed IISs. Figure 7.1 shows such a difference in IISs' number of constraints and the mean runtime increase of computing the smallest IIS with FORQES. We report the runtime increase as the ratio between FORQES's runtime and CPLEX's runtime. Results show that for the majority of instances (∼70%), both algorithms obtain IISs of the same size. FORQES computes IISs with a size reduction greater than 2 constraints in only 7% of the cases. In contrast, FORQES requires $4 \cdot 10^4$ times as much runtime as CPLEX on average. Overall, our results show that computing the smallest IIS via FORQES does *not* result in IISs that are significantly smaller than those computed by CPLEX, which, in contrast, terminates in a significantly shorter runtime. Henceforth, in the rest of our experimental evaluation, we only employ the CPLEX runtimes to compute IISs.

[3] Code for the experimental evaluation at: https://github.com/RogerXLera/ExMIPExperimentalEvaluation.

[4] FORQES was originally designed to compute the smallest MUS, a concept closely related to IISs (see Section 2.4). To compute the smallest MUS, FORQES iteratively computes the *Minimal Correction Subset* (MCS) of different sets of constraints. We adapt FORQES by iteratively



(a) Frequency of instances



(b) Runtime increase

Figure 7.1: Distribution of IIS size reduction computed by FORQES versus CPLEX (green, left) and runtime increase to compute the smallest IIS (red, right).

7.2.3 Empirical hardness results

Having evaluated the cost of computing the smallest IIS, we now report the detailed results regarding the runtimes and the overhead of computing an IIS for the three application domains addressed in this thesis, the RCPSP, and the WDP.

DPP. Figure 7.2 shows the MIP solving times, the IIS computation times, and the time overhead to compute explanations for DPP problem instances. Notice that solving the problem and computing IISs is very fast ($\sim 10^{-2}$ seconds) for the majority of cases. Thus, we can provide instant explanations for such a problem. We observe

computing the *Maximal Feasible Subsystem* (MFS) instead, an analogous problem to MCS in the constraint programming literature [43]. Our adaptation does not affect the algorithm's complexity.

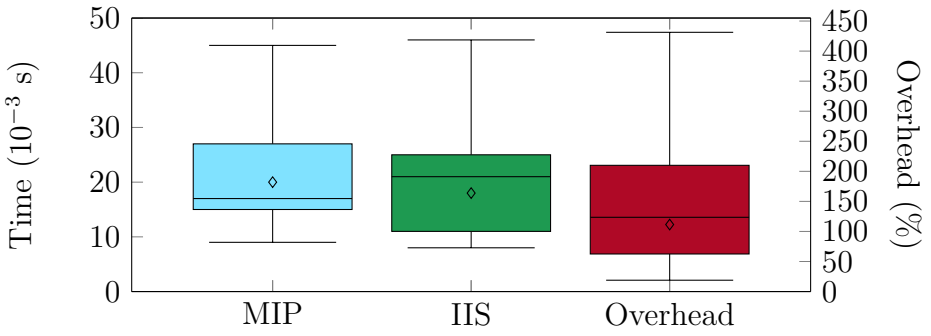


Figure 7.2: Runtimes to solve the MIP problem and compute IIS (left y-axis), and time overhead (right y-axis), for DPP.

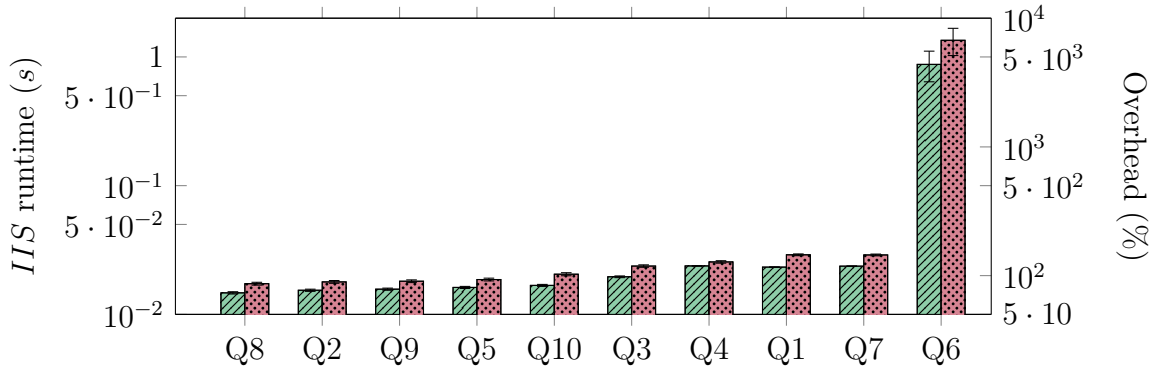


Figure 7.3: Average IIS computation time (lines, left y-axis) and overhead (dots, right y-axis) for different query types for the DPP.

that the average overhead is close to 100%, indicating that explanations take the same time to compute as to solve the optimisation problem on average, thereby being both tasks very fast to complete. In more detail, only 25% of explanations take more than twice as long to compute as to solve the MIP problem.

Figure 7.3 shows the impact of the query type (see Table 6.2) on the average runtime to compute an IIS, and consequently, the required overhead to compute an explanation. On the one hand, most query types are solved on average in approximately a hundredth of a second and have $\sim$ 100% of time overhead. This result confirms that it is equally computationally costly to compute an explanation as it is to compute the optimal solution, enabling real-time interaction with a user, since the mean runtime is less than two hundredths of a second. On the other hand, computing explanations for query type Q6 is significantly harder than the other query types. Although the overhead indicates that computing an IIS is harder than solving the optimisation

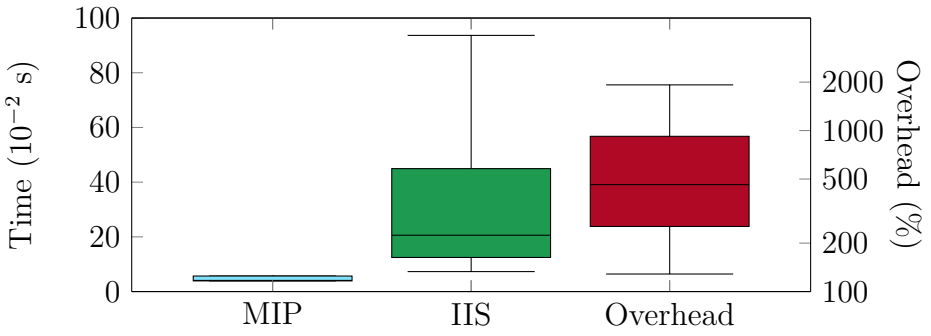


Figure 7.4: Runtimes to solve the MIP problem and compute IIS (left y-axis), and time overhead (right y-axis), for the RPP problem.

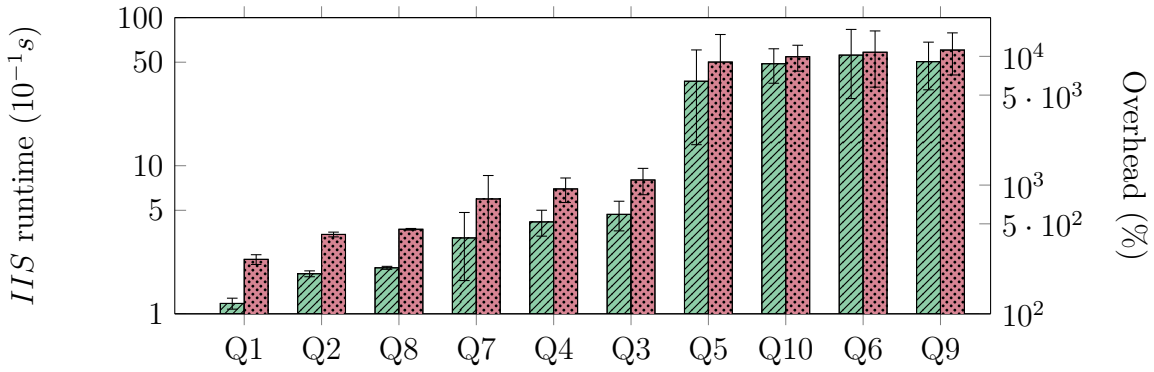


Figure 7.5: Average IIS computation time (lines, left y-axis) and overhead (dots, right y-axis) for different query types for the RPP problem.

problem for this query type, we still compute explanations in real time. It remains future work to establish why such a query is harder to explain than others.

RPP. Figure 7.4 reports the MIP solving times, the IIS computation times, and the corresponding time overhead required to compute explanations for RPP instances. Overall, solving the optimisation problem is very efficient, with median solving times of approximately 4×10^{-2} seconds. This confirms that the underlying RPP model can be solved in near real time for the considered instances. In contrast, computing the IIS is more expensive. While the median IIS computation time is still below one second (approximately 0.21 seconds), the distribution is much more skewed than in the DPP case, with an upper whisker close to one second. As a consequence, the overhead introduced by explanation generation is substantially higher. These results indicate that, although explanations for the RPP can still be computed in real time, explanation generation is more demanding for RPP than for DPP.

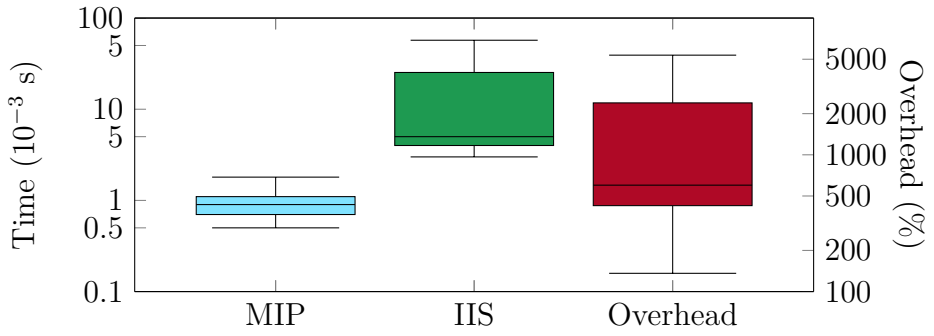


Figure 7.6: Runtimes to solve the optimisation problem and compute IIS (left y-axis), and time overhead (right y-axis), for the aggregation of value systems.

Figure 7.5 further analyses the impact of the query type on the runtime of IIS computation and, consequently, on the explanation overhead. On the one hand, several query types (e.g., Q1, Q2, Q7, and Q8) can be explained relatively efficiently, with average runtimes below 0.5 seconds and overheads in the range of 250% to 800%. For these queries, explanation generation remains feasible in real time. On the other hand, a subset of query types (notably Q5, Q6, Q9, and Q10) is substantially harder to explain. For these queries, the average time to compute an IIS increases by more than an order of magnitude, reaching between 3.7 and 5.6 seconds. Overall, these results show that, while LLMxMIP can provide explanations for RPP instances in real time, the computational cost is highly sensitive to the type of query posed by the user. This highlights the importance of predicting explanation runtimes and managing user expectations when deploying explainability mechanisms for complex optimisation problems.

Aggregation of Value Systems. Figure 7.6 reports the solving times, IIS computation times, and the corresponding overhead required to generate explanations for the aggregation of value systems. As in the DPP and RPP cases, computing the optimal solution is very fast. Moreover, although computing the IIS is more expensive than solving the problem, it remains well within real-time bounds. For instance, the median time required to compute an IIS is approximately 5 milliseconds, while the maximum observed runtime slightly exceeds 50 milliseconds. Overall, these results indicate that explanations for the aggregation of value systems can be generated in real time, even in the worst cases observed. Nevertheless, due to the extremely

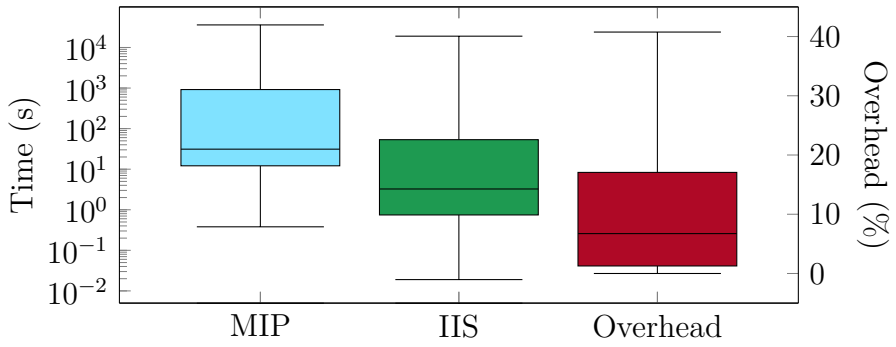


Figure 7.7: Boxplot of the Solving time of the MIP problem, the IIS runtime (left y axis), and the Overhead for the RCPSP (right y axis).

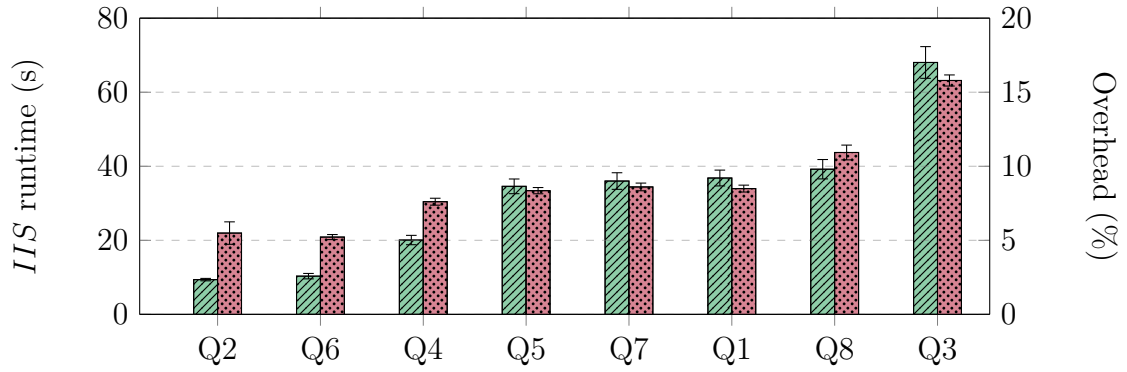
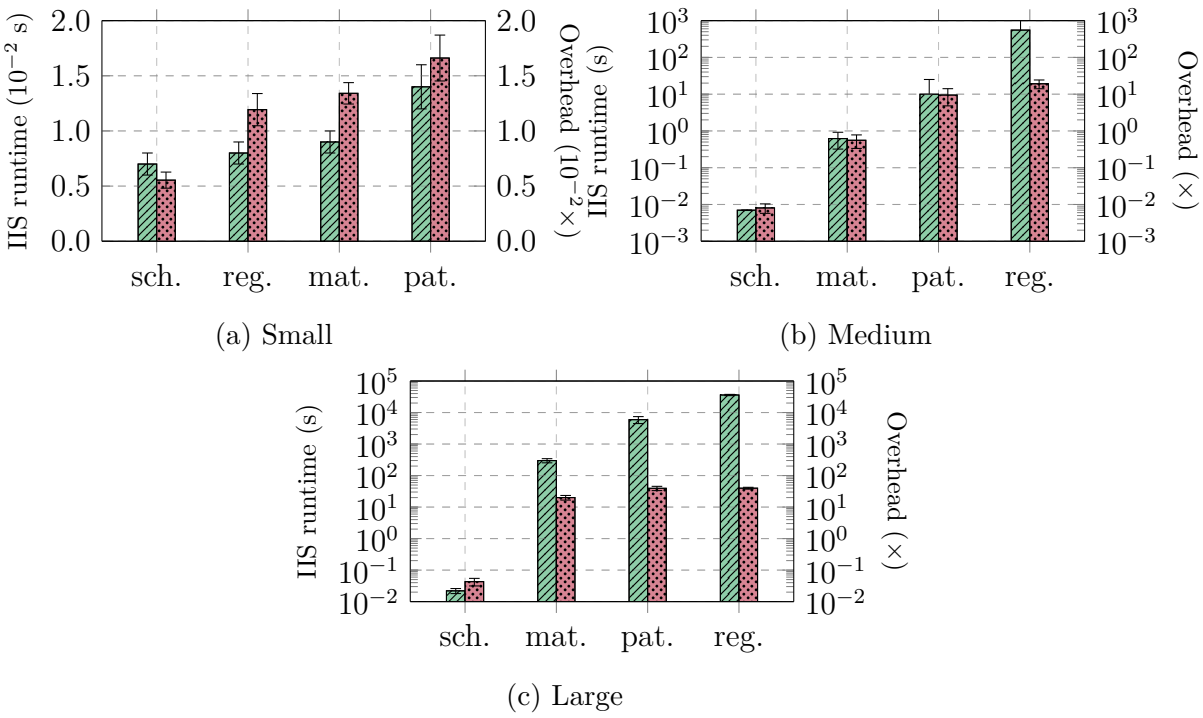


Figure 7.8: Average IIS computation time (lines, left y-axis) and overhead (dots, right y-axis) for different query types for the RCPSP.

small solving time of the underlying optimisation problem, explanation generation incurs a substantial relative overhead.

RCPSP. Similarly to Figures 7.2, 7.4, and 7.6, Figure 7.7 shows the MIP solving times, the IIS computation times, and the time overhead to compute explanations for RCPSP problem instances. Notice that solving such RCPSP instances is harder than the considered real instances of the optimisation problems tackled in this thesis, since harder instances take more than 10^4 seconds to solve. However, for such a case, we also notice that computing IISs is significantly faster than solving MIP problems. We observe that, in most cases, the overhead is lower than 40%, indicating that most explanations can be computed faster than half the time required to solve a problem instance. On the one hand, the median overhead to compute an explanation is 6.7%, and the median time is 3.3 seconds. On the other hand, the media MIP solving time is 31.2 seconds. Finally, we compute more than 75% of the explanations in less than



(a) Small

(b) Medium

(c) Large

Figure 7.9: Average IIS runtimes (lines) and overhead (dots) for different WDP instance distributions: *scheduling*, *matching*, *paths*, and *regions*.

1 minute. Notice that generating fast explanations is a key requirement for users of explainable systems to interact, according to the study by [112].

Figure 7.8 shows the impact of the query type (see Table B.5) on the average runtime to compute an IIS, and the average overhead to compute an explanation. On the one hand, most query types are solved on average in less than 60 seconds and have less than 10% of time overhead. On the other hand, some query types are significantly harder than others, e.g., Q3. Similarly to query Q6 for the DPP, or queries Q5, Q6, Q9, and Q10 for the RPP, it remains future work to establish why such a query is harder to explain than others.

WDP. Figure 7.9 shows the average IIS computation time and overhead for each distribution. We plot the overhead with "×" units, indicating the cost of computing IISs compared to the time to solve problem instances. We organise the results in three plots of different instance sizes based on the number of bids $|B|$: small ($|B| \in \{20, 50\}$), medium ($|B| \in \{100, 200\}$), and large ($|B| \in \{500, 1000\}$). We generally observe that, as instance sizes increase, average IIS computation times and

overheads increase as well. In detail, for small-size instances (Fig. 7.9a), computing explanations is much cheaper than solving problems (i.e., the overhead is $\ll 1\times$), and fast (no more than $2 \cdot 10^{-2}$ seconds on average). Moreover, there are significant differences in the average IIS computation time for the different distribution types for medium-size instances (Fig. 7.9b). While computing an IIS is fast for the scheduling and matching distributions (less than 1 second), it is much harder for the region's distribution (550 seconds, with an overhead of $20\times$ on average). Finally, computing an IIS is hard for large instances, and hence the overhead increases for almost all distributions except the scheduling distribution. Notice that we observe differences between distributions in IIS computation time. Such differences have also been reported in the empirical hardness of solving instances [114].

Results Summary. Across the optimisation problems addressed in this thesis, i.e., the *Degree Planning Problem*, *Research Project Planning*, and the *Aggregation of value systems*, we observe that explanation generation is consistently fast. For these real-world domains, LLMxMIP can extract the IIS in less than a second in the majority of cases, and, therefore, generate explanations in real time. To assess the behaviour of our approach on computationally harder optimisation problems, we additionally evaluated it on two well-known benchmarks from the combinatorial optimisation literature: the *Resource-Constrained Project Scheduling Problem* and the WDP. For the RCPSP, our results show that extracting the IIS is significantly faster than solving the problem. In detail, the median overhead of extracting the IIS is 6.7% with respect to solving the optimisation problem. Consequently, LLMxMIP can generate explanations in real time, i.e., within a few seconds, even for challenging scheduling instances. For the WDP, the computational cost of explanation generation depends strongly on both the instance size and the structure of the bid distribution. Generally, the outcome of small and medium-sized problem instances can be explained in real time. In contrast, for region-based distributions, generating explanations for medium-sized instances already becomes computationally expensive, as it typically takes a few minutes. In large instances, explanation generation is costly, except for the scheduling distribution, whose outcomes can be explained in less than a second. Overall, these results indicate that explanation generation be-

comes computationally demanding only in the presence of arguably unrealistic large instances. Nevertheless, when they do occur, the variability in explanation runtime motivates the need to inform users of the waiting time for an explanation. This leads us to the next section, where we propose a predictor to estimate the runtime required to generate an explanation.

7.3 Predicting explanations' empirical hardness

While LLMxMIP can generate explanations in real-time (which is fundamental to provide our explanations as part of an interaction [112]), for some large instances, it may fail to compute an IIS in real-time. In such cases, providing the user with an estimate of the computation time can be helpful.

Hence, in this section, we aim to predict the time required to generate explanations (i.e., the IIS running time) using ML techniques. Specifically, since our experimental results in Section 7.2 show that the explanation generation time depends strongly on the user's query, we identify multiple features that characterise both the query and the underlying optimisation problem instance, and use them as input to train our ML prediction models. In this way, we build upon prior work on solving time prediction [114, 96]. We extend this line of research by training models that incorporate not only instance-specific features but also query features. Following the methodology of the previous section, here we consider the same benchmark problems, i.e., RCPSP and WDP.

The remainder of this section is structured into three parts. We first describe the ML models employed to predict explanation runtime in Section 7.3.1. We then introduce the features used to characterise optimisation problem instances and user queries in Section 7.3.2. Lastly, in Section 7.3.3, we present and discuss the experimental results, comparing the performance of the different models and highlighting the benefits of incorporating features that capture user queries.

7.3.1 Machine learning techniques

In this section, we briefly describe the ML methods that we used to predict the runtime required to compute explanations: linear and quadratic regression, regression trees, random forests, and neural networks. We use these ML techniques because they are commonly used in the literature to build models that predict the solving time of optimisation problems [114, 96].

7.3.1.1 Linear Regression

Linear Regression (LR) is the most basic regression technique since it involves fitting a linear function $f_{\mathbf{w}}(\mathbf{x}) = w_0 + w_1 \cdot x_1 + \cdots + w_n \cdot x_n$ to its input $\mathbf{x}$, where $\mathbf{w}$ are the parameters of the model. In our case, our inputs are the features characterising the UDSP instance $\mathbf{z}$, and the coefficients $\mathbf{w}$ weigh such features in our fitted model.

This technique aims to obtain the weights $\mathbf{w}$ that minimise the *Mean Squared Error* (MSE), i.e., minimising $\sum_{i=1}^{N}(y_i - f_{\mathbf{w}}(\mathbf{x}))^2$, where y_i is the output to predict.

7.3.1.2 Quadratic Regression

Following Leyton-Brown, Nudelman and Shoham [114], we consider employing *Quadratic Regression* (QR) by constructing additional features that capture nonlinear interactions among the original ones, and subsequently fitting a linear model over this expanded feature set. Specifically, we augment the original features with all pairwise products, yielding a second-degree polynomial feature representation.

However, with such an approach, the number of features increases polynomially. This might cause problems during training when the number of features, and consequently the linear regression coefficients, exceeds the number of training instances. In addition, some features might be uninformative or strongly correlated to others. To mitigate the problem of having too many features, previous works in the literature use feature selection methods to reduce the number of features [96, 203, 204]. Therefore, we use forward selection to select the most relevant original features. This method iteratively chooses the best feature to add to the output feature set based on the cross-validation MSE of linear regression, and it stops when it has selected

N features [67]. Then, we perform linear regression on all pairs of products from the selected feature set and its linear features. We use $N = 20$ as our parameter configuration for this method.

7.3.1.3 Regression Tree and Random Forest

Regression trees are tree-based regression models [28]. A *Regression Tree* (RT) is a tree where each non-leaf node is associated with an input feature. At each node, there's a binary split into two child nodes based on a splitting rule for the associated input feature and its possible values. A tree is built by recursively splitting the training dataset into subsets, each of which becomes a successor child. This process is repeated on each derived subset until the data in the leaf node share the same feature values. Such a process can lead to *overfitting*, which can be mitigated with *pruning* methods that eliminate branches that do not improve the model's accuracy.

A *Random Forest* (RF) is a method consisting of a set of regression trees [27]. Such trees can lead to different predictions because they are trained on different training sample sets or use different feature subsets to generate splitting rules. A RF prediction is computed by averaging the predictions of all the trees, hence preventing overfitting. The prediction quality improves as the number of trees, B, increases. We use $B = 100$ in the experiments.

7.3.1.4 Neural Networks

*Neural Network*s (NNs) are models inspired by the structure and function of the brain. They consist of layers of interconnected nodes, the so-called *neurons*, which process data to enable the network to learn patterns and make predictions or decisions. The *Multi-Layer Perceptron* (MLP) is one of the most basic NN architectures, where neurons are organised in layers. In detail, each neuron n_i receives k inputs $\mathbf{x_i} \in \mathbb{R}^k$ from neurons of the previous layer. Then, a neuron has associated a set of weights $\mathbf{w_i}$ and a bias term b_i which are used to aggregate the inputs such that $\mathbf{w_i} \cdot \mathbf{x_i} + b_i$. Finally, each neuron propagates the result to the neurons of the next layer through a nonlinear activation function. The weights of each neuron, $\mathbf{w_i} \in \mathbb{R}^k$,

Table 7.1: General features for user queries.

Size

1-2 **Number of variables** and **number of constraints**.

Interaction with the Problem Constraints

Scope

3 **Number of constraints sharing scope with query constraints.**

4-6 **Number of variables shared statistics:** max, mean, and standard deviation of the number of variables shared with main problem constraints.

Structural

7-10 **Query Clustering Coefficient statistics:** For each query constraint q_i, we calculate the number of edges between the neighbours N_i divided by the number of possible edges of the neighbours $k_i(k_i - 1)/2$, where k_i is the number of neighbours. Thus, the clustering coefficient of a node C_i is $\frac{2 \cdot |\{e_{jk} : v_j, v_k \in N_i, e_{jk} \in E\}|}{k_i(k_i-1)}$. We record the min, max, mean, and standard deviation of the clustering coefficients.

11-13 **Query shortest path length:** max, mean, and standard deviation of the minimum path length, over all pairs of query-constraints and main-problem-constraints.

14 **Ratio** of the average clustering coefficient to the average query shortest path length.

Interaction with the Problem Objective

15 Number of **query constraints variables shared** with the minimality constraint.

16 **Shortest path length** from the query constraints to minimality constraint.

are adjusted so that they minimise a loss function, a process carried out using an optimisation algorithm.

We employ the well-known Python library TensorFlow [60] to implement two MLP models. We implement a model using one hidden layer of 128 neurons, which we call *Shallow Learning* (SL) model. We use a second model using four hidden layers of 128 neurons, which we call *Deep Learning* (DL) model. Both models use ReLU activation functions and are trained using the "*Adam*" optimisation algorithm to minimise the MSE.

7.3.2 Feature Engineering

In this section, we define the query and problem instance features used to train models predicting explanation generation time.

7.3.2.1 Query Features

As shown by our experimental analysis in section 7.3, the hardness of computing an explanation depends on the query posed by the user. Hence, characterising a query is important for predicting the time required to compute an explanation. To this end, we identify query features based on the analysis of: (i) the dimensions of the query constraints; (ii) the relationships between the query constraints and the original problem constraints; and (iii) the relationships between the query constraints and the objective of problem (i.e., the minimality constraint introduced in Section

6.2.2). Table 7.1 compiles the query features that we have identified. Importantly, we restrict attention to features that can be computed for *any* query constraint, independently of domain-specific knowledge. The proposed features are described in more detail below.

- **Internal Features**. The first group of features focus on internal properties of the query constraints. In detail, we include the number of variables and constraints used to encode a user's query. We introduce them to record the size of the query constraints added to the problem constraints, which may affect the runtime for computing an IIS.
- **Features from the interaction with the problem constraints**. This second group of features captures the extent to which the query constraints interact with the original problem constraints.
  - **Scope Features**. We record the number of main problem constraints that share at least one variable with the query constraint. Moreover, we compute the maximum, average, and standard deviation of the number of shared variables between the query constraint and the main problem constraints.
  - **Structural Features**. To further capture the relationship between the query constraints and the main problem constraints, we represent the set of main problem constraints and query constraints as a dual graph (see Section 6.2.4), where each node corresponds to a constraint and edges connect constraints that share at least one variable. Using this representation, we extract structural features that characterise the relationships of the query constraint(s) within the graph. In particular, we compute the clustering coefficient of the node(s) associated with the query constraint(s).[5] as well as the shortest path lengths between the query constraints and each of the original problem constraints.
- **Features from the interaction with the problem objective**. Finally, the

[5]When a query is encoded by multiple constraints, we record the minimum, maximum, average, and standard deviation of their clustering coefficients.

last group of features captures the relationships between the query constraints and the objective function of the optimisation problem, that is, the *minimality constraint* introduced in Section 6.2.2. Specifically, we record the number of variables shared between the query constraint(s) and the minimality constraint, as well as the shortest path length from the query constraint(s) to the minimality constraint.

7.3.2.2 Problem Instance Features

To characterise problem instances, we identify specific problem features. For the WDP, we employ features previously used to predict the runtime of solving the WDP, as specified in [114]. For the RCPSP, we define features related to the generation of artificial problem instances [109, 108]. In the Appendix C, we show the full list of problem instance features.

7.3.3 Empirical analysis

The main goal of this section is to compare the prediction error of the trained ML models of the techniques discussed in Section 7.3.1 to estimate the time to compute an explanation.[6] Furthermore, we aim to quantify the reduction in prediction error achieved by incorporating the query features in Table 7.1, compared to models trained using only problem-instance features.

To do so, we compute the problem instance and query features of the problem instances discussed in Section 7.2.1.1. Thus, we obtain the explanation computation time of 11520 and 14400 query instances for the RCPSP and WDP, respectively. However, we discard the optimisation problem instances that exceed 10 hours in solving time (since we limit the computation to 10 hours, we do not have the true runtime). As shown in Section 7.2, the explanation generation time (EGT) ranges

[6] Following prior work [115, 96], we can also train predictors to estimate the solving time of optimisation problems. However, since this task is beyond the scope of this thesis, we report the results of predicting the solving time for a problem instance in Appendix C.2.

Table 7.2: IIS runtime prediction results using problem and query features as input for the different trained models: *Linear Regression* (LR), *Quadratic Regression* (QR), *Regression Tree* (RT), *Random Forest* (RF), *Shallow Learning* (SL), and *Deep Learning* (DL). We report 10-fold cross-validation RMSE and MAE (lower values are better). Bold indicates the (significantly) best-performing model.

Problem Domain	**RMSE**						**MAE**					
	LR	**QR**	**RT**	**RF**	**SL**	**DL**	**LR**	**QR**	**RT**	**RF**	**SL**	**DL**
WDP	1.12	1.06	1.02	**0.75**	0.89	0.91	0.93	0.88	0.58	**0.46**	0.61	0.62
RCPSP	0.82	0.78	0.92	**0.66**	0.77	0.80	0.78	0.74	0.55	**0.43**	0.56	0.59

between 10^{-3} seconds to 10 hours ($3.6 \cdot 10^4$ seconds). Due to such a broad range of EGT, we predict the $\log_{10}$(runtime) instead of the EGT. [7]

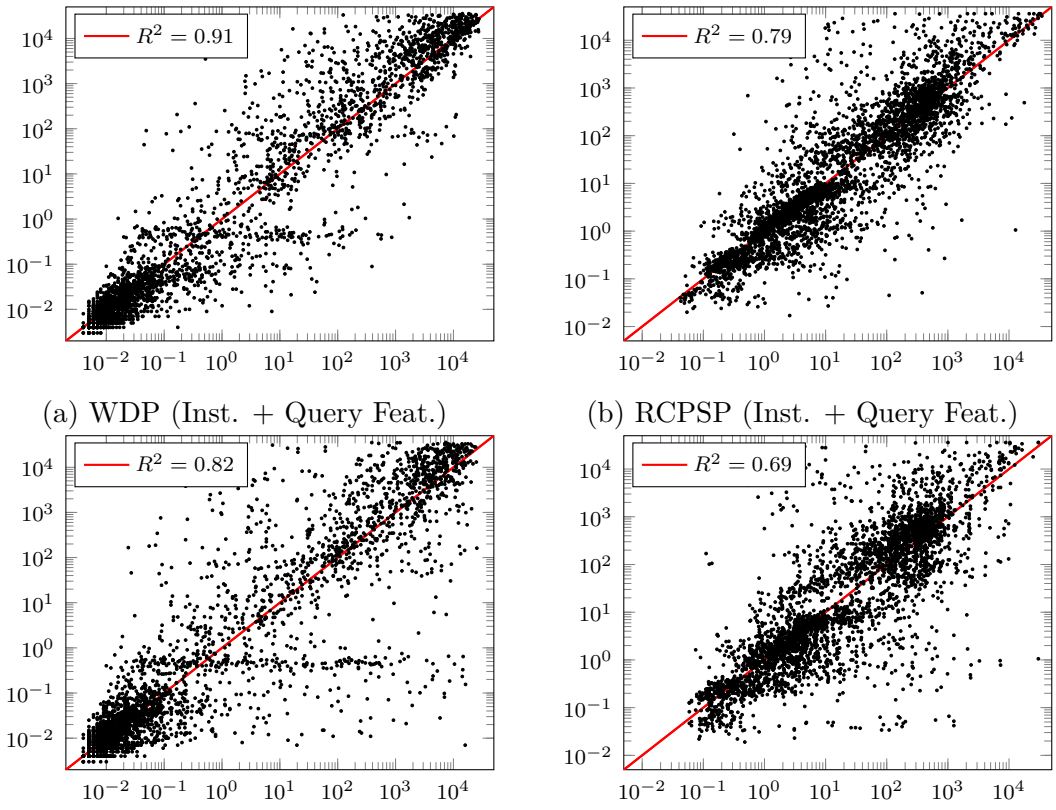
Then, we train several predictors using the ML techniques discussed in section 7.3.1. Our training encompasses a 10-fold cross-validation using 80% of our dataset, leaving the remaining 20% to test each model. We report the RMSE because we use the MSE as the loss function for all the ML methods we employ. For completeness, we additionally report the *Mean Absolute Error* (MAE), which is easily interpretable because it represents the average absolute deviation between predicted and actual runtimes.

7.3.3.1 Comparing predictors

In Table 7.2, we show the RMSE and MAE of each prediction model over the WDP and RCPSP test data. Overall, we observe that the RF model is the best-in-class for predicting the IIS runtime, both for the WDP and the RCPSP. Indeed, the RF model outperforms the SL model, achieving a 19% and 17% reduction in prediction error for the WDP and the RCPSP, respectively. Specifically, we observe that the lowest prediction errors are below < 1, e.g., the RMSE of the RF predictor is 0.75 and 0.66 for the WDP and the RCPSP, respectively. Intuitively, this indicates that the predicted runtimes are, on average, close to the observed explanation generation times, with errors that remain within a comparable time scale.

Figure 7.10 shows the scatter plots and the adjusted R^2 of the runtime predictions

[7] If we predict the absolute running time, the model would strongly penalise low relative error for hard instances and would not penalise huge relative errors in easy instances.



(a) WDP (Inst. + Query Feat.)

(b) RCPSP (Inst. + Query Feat.)

(c) WDP (Inst. Feat.)

(d) RCPSP (Inst. Feat.)

Figure 7.10: IIS Runtime predictions on previously unseen test instances for Random Forest (best performing model). The $x-$axis denotes the runtime prediction, and the $y-$axis denotes the real runtime. Each data point represents one instance. We show the Coefficient of Determination (R^2), which measures how well the model predicts runtime (1 indicates a perfect fit).

of RF on test data. Notice that, except for some outliers, the data points tend to cluster tightly around the diagonal line, indicating a low prediction error for both the WDP (Figure 7.10a) and the RCPSP (Figure 7.10b).

7.3.3.2 Quantifying the benefit of using query features

In this section, we aim to quantify the benefit of incorporating query features as input to train our ML prediction models, as opposed to a model trained only with problem instance features. To this end, we train the ML approaches without using query features and compare their prediction errors with those of models that do use query features. Table 7.3 compares the RMSE obtained with the models using query features and the models that only employ problem instance features. We observe that incorporating query features substantially improves predictive accuracy for the

Table 7.3: IIS runtime prediction comparison between models using only problem instance features and models using query and problem instance features. We report 10-fold cross-validation RMSE (lower values are better) for both optimisation problems, the WDP and the RCPSP. In the third row, we report the gain (in percentage) of using query features. Bold indicates the (significantly) best-performing model.

Input Features	**WDP**						**RCPSP**					
	LR	**QR**	**RT**	**RF**	**SL**	**DL**	**LR**	**QR**	**RT**	**RF**	**SL**	**DL**
Query + Problem Instance	1.12	1.06	1.02	**0.75**	0.89	0.91	0.82	0.78	0.92	**0.66**	0.77	0.80
Problem Instance	1.14	1.04	1.15	0.96	**0.92**	**0.92**	0.83	0.80	**0.78**	**0.78**	0.83	0.84
Gain (%)	1.8	-1.9	11.3	**21.9**	3.3	1.1	1.2	2.5	-15.2	**15.4**	7.2	4.7

RF model, reducing the RMSE by 22% for the WDP and by 15% for the RCPSP. Overall, introducing query features improves prediction performance across almost all models.

Figures 7.10c and 7.10d show the runtime prediction for computing an IIS with only problem instance features. Importantly, we note that the prediction performance improves significantly when using query features, as shown in Figures 7.10a and 7.10b. In contrast, Figures 7.10c and 7.10d show a much greater dispersion of data points, indicating a worse prediction when the models rely solely on problem instance features. Notice that the adjusted R^2 significantly improves both for the RCPSP and the WDP.

7.4 Conclusions

In this chapter, we addressed a critical aspect of explainability for optimisation-based decision support systems: the computational cost of generating explanations. While the previous chapter focused on the quality of explanations generated by LLMxMIP, here we evaluate whether these explanations can be produced in real time to present them as part of a conversation. To this end, we conducted an extensive empirical evaluation of the runtime required to generate explanations across a diverse set of optimisation problems. Our analysis covered all optimisation problems addressed in this thesis, as well as two well-known and computationally challenging benchmarks from the combinatorial optimisation literature, namely the RCPSP and the WDP. Our results indicate that, in the case of the problems addressed in this thesis, LL-

MxMIP typically provides explanations in less than a second. Moreover, for the RCPSP, our method computes explanations much faster than solving the optimisation problem instance. This is a key aspect of LLMxMIP since state-of-the-art approaches require solving additional MILP problems to generate explanations [76, 211]. For the WDP, we can compute explanations in less than one minute for small and medium instances, though computing explanations for large instances can be more challenging.

However, our experiments also reveal that the computational cost of explanation generation can vary substantially across problem instances and, crucially, across different user queries. Indeed, in some cases, LLMxMIP cannot compute an explanation in real time. In such cases, providing an estimate of the time required to compute an explanation might be valuable to users. Along these lines, the second contribution of this chapter is to develop a methodology for predicting explanation runtimes. Building on established work on optimisation solving time prediction, we trained a variety of machine learning models using features that characterise both the optimisation problem instance and the user query. Importantly, we identify user query features by analysing the relationships between the query and the optimisation problem. Our results show that by explicitly modelling query-specific features, we significantly reduce prediction error, enabling reliable estimates of the runtime needed to compute an explanation. More specifically, Random Forest is the model that significantly outperforms others, confirming results from related work [96].

In conclusion, the contributions of this chapter address the research questions concerned with the computational feasibility of generating explanations for optimisation-based decision support systems, namely **Research Questions Q7.1** and **Q7.2**. The main contributions can be summarised as follows:

- **Question Q7.1:** When do the computational costs of our explanability methodology allow supporting real-time, lively interactions with users? We provide an empirical evaluation of the computational cost of generating explanations with LLMxMIP across a wide range of optimisation problems, including hard benchmark instances. Our results show that LLMxMIP typically generates ex-

planations fast enough to present them in real-time as part of a conversation, even for computationally demanding problems.

- **Question Q7.2:** Can we predict the computation time required to generate an explanation? We propose a runtime prediction methodology based on machine learning models that use as input features characterising the optimisation problem instance and the user's query. Our results show that accurate runtime estimates can be obtained, with random forest models consistently outperforming alternative approaches.

Chapter 8

Conclusions and Future Work

In this thesis, we study how to formalise, solve, and explain optimisation problems that guide the recommendations produced by *Decision Support System* (DSS).

In real-world settings, such optimisation problems typically involve a large number of variables and constraints, making their outcomes complex and difficult to interpret. As a result, users may struggle to understand, trust, and ultimately adopt the recommendations provided by optimisation-based DSSs. Along these lines, generating explanations that help users understand recommendations and support the decision-making process is crucial for the deployment of DSSs in real-world scenarios [47, 46]. In this dissertation, we propose a domain-independent method for generating explanations that explicitly answer users' questions about the outcome of an optimisation problem. Furthermore, we evaluate our method on the optimisation problems studied throughout this thesis, demonstrating its generality and applicability.

For the remainder of this Chapter, we first summarise the main contributions and key findings of this thesis in Section 8.1. We then discuss promising directions for future research in Section 8.2.

8.1 Key Contributions

We now outline the main contributions and key results of this dissertation, following the structure of the thesis. First, we summarise the contributions related to each real-world optimisation problem addressed. Then, we outline the contributions of the explainability methodology developed on top of these optimisation models.

8.1.1 Aggregating Value Systems

The first application addressed in this thesis concerns value alignment, which aims to ensure that AI systems act in accordance with a set of moral values. This problem is particularly relevant for policymakers, who must make decisions or create policies aligned with a society's moral values. In such a scenario, decision-making often must take into account that individuals in a society hold multiple, conflicting beliefs about moral values [72]. These beliefs can be formalised as *value systems*, which capture how individuals interpret and prioritise moral values [18, 172]. To tackle this challenge, we focus on developing a consensus-based value system that best represents a society's moral values.

Unlike prior works in the Ethics literature, where value systems typically account only for preferences among values [18, 123], we explicitly account for these value interpretations. Accordingly, our first main contribution (**Contribution C1**) is a general definition of a value system that captures both preferences over values and interpretations of values. Specifically, our notion of a value system comprises a set of values, a set of *contextual* alternatives, a set of preferences among values, and a set of relationships between values and alternatives. These relationships encode value interpretations by specifying the degree to which an individual perceives that an alternative promotes (or demotes) a given moral value. In Chapter 3, we demonstrate the applicability of this definition through two real-world case studies: a Participatory Value Evaluation (PVE) conducted in the Netherlands and the European Value Study (EVS).

Having proposed a general definition of a value system, we then formalise the problem of aggregating individual value systems to derive a consensus on both value preferences and value interpretations. We address this problem by minimising a p-parameterised distance function, which enables the computation of different consensuses by varying the parameter p, referred to as the "*ethical principle*". This ethical principle ranges from *utilitarian*, which maximises the total welfare, to *egalitarian*, which minimises the disagreement of the individual who is most distant from the majority. This leads to our **Contribution C2**: we cast the aggregation of value

systems as an ℓ_p-regression problem [24], thereby providing a unified framework for computing consensuses under different ethical principles.

Finally, through an extensive empirical evaluation on two real case studies, i.e., the PVE and the EVS, we analysed how different ethical principles influence the resulting consensus value systems. In particular, our experiments allowed us to (i) quantify the impact of selecting different ethical principles on the consensus, (ii) characterise the semantics of these principles by identifying whether they yield a more utilitarian or egalitarian consensus, and (iii) measure the trade-offs involved when moving away from the majority to accommodate minority perspectives. While the specific results vary across the two case studies due to differences in the datasets, we extract common insights into the impact of the ethical principle. Based on these insights, we present **Contribution C3**, a set of practical guidelines to support policymakers in selecting an ethical principle that aligns with their goals when aggregating individual value systems.

8.1.2 Research Project Planning

The second application addressed in this thesis concerns the Research Project Planning (RPP) problem, a task routinely faced by research institutions when allocating researchers' time across multiple projects. This problem is inherently complex, since it requires satisfying a wide range of constraints, including researchers' workload limits, project dedication requirements, and budgetary restrictions, while also accounting for (i) researchers' personal preferences over how their time is distributed, and (ii) an efficient budget expenditure. To date, in the absence of dedicated automated software, research institutions typically tackle this problem manually, which often results in suboptimal allocation of researchers' available hours and imposes an additional workload on administrative staff.

To address this challenge, in Chapter 4, we formalised the RPP as a constrained optimisation problem that balances institutional requirements with individual preferences. In particular, we modelled such a problem as a constrained ℓ_2-regression problem [24], enabling the computation of allocations that minimise deviations from

researchers' preferred workloads while respecting hard and soft constraints. This formulation yields a Mixed-Integer Quadratic Programming (MIQP) model that can be efficiently solved with state-of-the-art solvers.

A key limitation of traditional planning approaches is their assumption that project conditions remain unaltered over the project duration. In practice, however, research projects are highly dynamic: project deadlines shift, team compositions might evolve as researchers leave or enter an institution, and new projects might start. To capture this inherent uncertainty, we extended our model using the Stochastic Robust Approximation (SRA) [24]. This allowed us to reason over multiple possible future scenarios and compute research plans that remain reliable even as conditions change. This constitutes **Contribution C4** of this thesis: an MIP formulation of the RPP problem that accounts for both individual preferences and institutional constraints, while explicitly accounting for uncertainty.

We evaluated our approach using real-world data from our institution, the IIIA. Our results show that the method is robust and, thanks to its parameterised objective function, can produce allocations that reflect the planners' preferences. In addition, we analysed how different parameter configurations affect the trade-off between satisfying researchers' preferences and efficiently utilising project budgets, thereby providing practical insights for administrative staff. Furthermore, we evaluate the scalability of our approach with realistic problem instances. Our results show that our approach is scalable even for problem sizes beyond those typically encountered in research institutions.

Finally, we implemented our approach in a decision-support prototype currently being tested within our institution. This demonstrates the practical applicability of our method and highlights its potential to support real-world planning processes.

8.1.3 Bachelor's Degree Planning

The last optimisation-based application we addressed in this thesis is the problem of Bachelor's Degree Planning, which concerns the automatic computation of Bachelor's Degree plans, i.e., a distribution of Bachelor's Degree subjects into semesters,

that satisfy university curricula restrictions while supporting students in achieving their career objectives. This problem is particularly challenging because students often lack precise knowledge of the skills required for their desired jobs and of how individual courses contribute to acquiring those skills. As a result, degree planning is frequently performed without systematic support, potentially leading to suboptimal educational trajectories.

To tackle this challenge, in Chapter 5, we formalised the Degree Planning Problem as an optimisation problem that schedules university subjects across semesters so as to maximise the acquisition of skills relevant to a student's preferred job(s), while respecting all academic constraints. This constitutes **Contribution C5** of this thesis. Since the objective function of this problem is inherently non-linear, we introduced a non-trivial linearisation using auxiliary variables and formally proved that this transformation is equivalent to the original formulation in terms of optimality and feasibility. In other words, both formalisations yield the same optimal solution, as well as the same set of solutions that satisfy the constraints. This enables the resulting model to be efficiently solved using standard MILPs solvers.

We evaluated our approach using real-world data from the Bachelor of ICT at Western Sydney University, combined with the SFIA skills framework. Our experimental results show that the proposed method can effectively support two complementary use cases: (i) assisting students in planning their academic trajectories in line with their career aspirations, and (ii) helping curriculum designers assess how well a degree program aligns with labour market needs. Moreover, although we instantiated our model on a specific curriculum, it is adaptable to other academic programs without requiring major changes to our approach. Finally, we implemented our approach as a prototype decision-support system and registered it as software.

8.1.4 Explaining the Outcome of an Optimisation-Based DSS

In this part of the thesis, we addressed a fundamental challenge in optimisation-based decision support systems: how to explain their recommendations so that users can understand and trust them. While optimisation techniques can efficiently generate

high-quality solutions, their complexity often renders them opaque, limiting their practical usefulness in real-world decision-making contexts. This issue is particularly critical in settings where transparency, accountability, and human oversight are required [46, 47].

Given this context, in chapter 6, we developed a general explainability methodology for optimisation problems, grounded in well-established insights from the social sciences about how humans generate explanations. In contrast to existing approaches that rely on numerical comparisons or are tailored to a specific problem domain, our method focuses on identifying and structuring the reasons why an optimal solution is preferable to a user-suggested alternative. This led to **Contribution C6.1**: the development of the graph of reasons, a structured, graphical representation that captures the relevant constraints and their interactions and serves as the basis for our explanation.

Since structured representations alone may still be difficult for non-expert users to interpret, we further investigated how to transform these graphs into natural-language explanations. Rather than relying solely on Large Language Models (LLMs) to perform complex reasoning and explain the outcomes of optimisation problems, which may produce incorrect and uninformative explanations, we used them as tools to generate explanations based on the reasons and their interactions captured in the graph of reasons. This hybrid approach, which constitutes the **Contribution C6.2**, combines the reliability of constraint reasoning techniques with the natural language processing capabilities of language models, resulting in explanations that are more complete, coherent, and less prone to hallucination.

Another key contribution of this thesis, **Contribution C8**, is to demonstrate the generality of our methodology by applying it to all optimisation problems addressed in this thesis, as well as to well-known benchmark problems such as the Resource-Constraint Project Scheduling Problem (RCPSP) and the Winner Determination Problem (WDP). Across these domains, our approach consistently generated contrastive, reason-based explanations that clarified why specific recommendations were made rather than alternatives suggested by users.

Beyond explanation quality, we also investigated the real-time computational feasibility of our approach, which is essential for enabling interactive, conversational decision support. Our empirical analysis showed that explanations can typically be generated in real time for most of the considered problems, often much faster than resolving the optimisation problem itself. Moreover, we identified the problem-query instances for which explanation generation becomes computationally challenging, constituting **Contribution C7.1**. In such cases, providing an estimate of the time required to find an explanation might still be valuable to the user. This leads to **Contribution C7.2**, where we train a variety of machine learning models to predict explanation generation time. Importantly, we identify a set of features that characterise the relationship between the user query and the optimisation problem. By employing these query features, along with those that characterise the problem instance, to train ML prediction algorithms, we significantly improved prediction accuracy, allowing systems to provide reliable estimates of explanation waiting time. Our results indicate that a Random Forest predictor consistently outperforms the other approaches.

8.2 Future Work

While this thesis formalises and solves novel optimisation problems in real-world domains and advances the state of the art in explainability for optimisation, it also opens several research avenues for future work. In this subsection, we outline promising directions that could further extend the proposed optimisation formalisations to improve their applicability in real-world domains. Moreover, we identify avenues for future work that seek to refine and extend our explainability approach, addressing some of the limitations identified throughout this dissertation. Similarly to Section 8.1, we first discuss future research directions for the optimisation problems addressed in this thesis, and in Subsection 8.2.4 we outline future research directions for our explainability approach.

8.2.1 Aggregating Value Systems

As future work, we identify three main research directions that naturally extend the framework developed in this thesis.

First, we aim to move beyond the computation of a single global consensus value system. In the current framework, aggregation yields one consensus intended to represent the entire population. However, real-world societies are rarely homogeneous and often consist of distinct groups or clusters of individuals who share similar value systems. Capturing such internal structure is crucial, as a single aggregated outcome may not inform decision-makers of important moral disagreements. Moreover, this would allow decision-makers to better understand a society's moral landscape, identify where value conflicts arise, and design policies that are more sensitive to the needs and perspectives of different social groups. As future work, we therefore plan to investigate methods for identifying communities of individuals with similar value systems and computing a consensus value system for each group. To this end, clustering methods such as k-means, DBSCAN, or OPTICS could be employed to identify such groups. While k-means partitions individuals into a predefined number of clusters by minimising within-group variance around centroids, DBSCAN and OPTICS identify clusters as dense regions of similar individuals, allowing the detection of irregularly shaped groups [160]. Beyond clustering approaches, another promising research direction concerns coalition formation methods. Such methods analyse how groups of individuals may strategically organise around their consensus value systems. In such approaches, it is essential to define a characteristic function that assigns to each possible coalition a numerical value capturing the benefit that its members can jointly achieve.

Second, we plan to further explore connections with the literature on multi-expert decision-making (e.g., [189, 93]). In particular, the distance function employed in this thesis can be interpreted as a penalty function, as defined in [16]. While this connection is promising, fully leveraging aggregation operators from that literature would require a careful analysis of their social choice properties, such as fairness, representativeness, and robustness to manipulation. Although this lies beyond the

scope of the present thesis, investigating whether such operators can offer similar normative guarantees at a lower computational cost constitutes a compelling direction for future research.

Finally, we plan to generalise our approach by allowing each individual to adopt their own ethical principle prior to aggregation. In the current framework, a single ethical principle is selected by the decision-maker and applied uniformly across all individuals. However, as argued by Baum and Slavkovik [15], the way in which values are aggregated (in our case, the ethical principle used for aggregation) is not "*morally neutral*". Indeed, in realistic pluralistic settings, individuals may differ not only in their value systems but also in how these values should be aggregated. Addressing this challenge would require the development of new social choice functions and corresponding computational methods capable of handling heterogeneous ethical principles.[1]

8.2.2 Research Project Planning

Several directions for future research naturally emerge from our work on the Research Project Planning (RPP) problem.

First, as an initial step, we plan to deploy and evaluate our approach in different research institutions involving a wider variety of project types. Such an evaluation would require extending our current formalisation to accommodate additional project-specific requirements, such as heterogeneous funding schemes or different contractual constraints. Extending the model in this direction would strengthen its generality and applicability across diverse institutional settings.

Second, an important avenue for future work is incorporating fairness into the planning process. In the current formulation, allocations are driven by researchers' preferences and project costs, which may result in uneven workload distributions, with some researchers dedicating substantially more hours than others. Introducing explicit fairness criteria, for instance by constraining or penalising imbalances in workload,

[1] Along these lines, Salas-Molina, Bistaffa and Rodríguez-Aguilar [165] proposes a method for aggregating multiple individual judgements under different ethical principles, which could serve as a valuable starting point for extending our framework in this direction.

would allow the model to better support equitable and sustainable research planning practices.

Finally, we aim to validate our approach through user-centred evaluations involving administrative staff from different research institutions. Such empirical validation would provide valuable feedback on the usability, transparency, and practical relevance of the proposed decision-support tool and help identify additional requirements to improve its adoption in real-world planning processes.

8.2.3 Bachelor's Degree Planning

Several avenues remain open for extending the Bachelor's Degree Planning framework proposed in this thesis.

A first promising direction concerns the further personalisation of degree plans. In its current form, our model focuses on maximising the coverage of skills relevant to a student's preferred job(s), while ensuring compliance with institutional constraints. However, students differ widely in their interests, learning pace, and academic strengths. Incorporating individual preferences over courses or workload distribution would enable the generation of more tailored degree paths that better reflect each student's personal goals and circumstances.

A closely related direction is to account for students' academic performance. Rather than treating all courses as equally attainable for each student, future extensions could integrate predictive models of student performance based on historical data. This would allow the system to estimate the expected difficulty of a given degree path and adapt recommendations accordingly. In contrast to existing educational recommender systems, which typically suggest individual courses without explicitly planning them [61, 173], we envision combining optimisation and learning techniques to generate full degree plans that optimise skill acquisition and are performance-aware. To achieve this, future work could leverage two main sources of data: (i) the observed performance of students in completed subjects, and (ii) historical data from past students. Such information could be used to train predictive models of expected performance, which could then be incorporated into the optimisation objective.

Predicting such expected performance is important as an indicator for students of the difficulty of the degree plan they are about to follow. As students progress, the expected performance can be refined, and our approach can learn to anticipate problems and recommend variations of a degree plan.

8.2.4 Explaining the Outcome of an Optimisation-Based DSS

While this thesis advances the state of the art in explainability for optimisation-based decision-support systems, it also opens several promising directions for future research.

A first research line concerns improving the interactivity of our explainability framework. Although our approach is domain-independent, it currently assumes that user queries are manually translated into additional mathematical constraints by an expert. This requirement limits the range of questions users can ask and might reduce the system's accessibility for non-expert users. A natural extension is to automate this translation process, enabling users to express their queries in natural language. Leveraging recent advances in LLMs [215], future work could investigate how to map free-form user questions into the corresponding mathematical constraints that encode user-suggested alternatives. This would allow for more flexible and intuitive interactions, facilitating a dialogue-based form of explanation.

A second important direction is the empirical validation of our approach through user studies. While our evaluation focuses on the computational and formal properties of the proposed explanations, assessing their satisfaction, usefulness, clarity, and trust requires direct feedback from end users [95]. Conducting controlled user studies would enable us to analyse how different types of explanations affect user understanding, satisfaction, and decision-making behaviour. Such studies would also help refine the explanation strategies and tailor them to the needs of different user profiles.

A third promising avenue is to extend our explainability methodology beyond optimisation problems formalised as Mixed-Integer Programming and solved by exact solvers. Many real-world problems are computationally intractable and are instead

addressed using tailored heuristics, metaheuristics, or hybrid methods that yield sub-optimal solutions. In these settings, the need for explanations may be even more critical for user trust. Future work could investigate how to adapt our approach to such scenarios, developing techniques to explain why a heuristic solution is preferable to alternatives.

Together, these directions aim to transform our explainability approach into a fully interactive, user-centred, and broadly applicable method for optimisation-based decision-making, further bridging the gap between complex optimisation decisions and non-expert users.

Appendix A

Winner Determination Problem

A *Combinatorial Auction* (CA) is a type of auction where bidders can place bids for sets of goods. Given a combinatorial auction that receives a set of bids over a set of goods, the WDP is that of selecting the subset of bids that maximise revenue [56]. We follow the standard approach to solve a WDP by casting it as a *Weighted Set Packing* (WSP) problem modelled as an MILP, where decision variable $x_b \in \{0, 1\}$ indicates whether bid b is selected. The WDP's constraints force that every good is allocated up to one winning bid.

A.1 Formalisation of the Winner Determination Problem

We consider the WSP for the WDP formalisation. Our choice of employing a WSP formalisation is also motivated by the fact that such a problem has been used in real-world application domains such as *shared mobility* [20], *cooperative learning* [8], and, according to [66], any scenario involving the formation of collectives of agents.

There is a set of goods G and a set of bids B. Each bid $b \in B$ is a tuple $\langle w_b, G_b \rangle$, where w_b is the value of the bid and $G_b \subseteq G$ is the set of goods within the bid. We formalise the WSP as follows.

$$\text{maximise} \quad \Phi(x) = \sum_{b \in B} w_b \cdot x_b \tag{A.1a}$$

$$\text{s.t.} \sum_{b \in B | g \in G_b} x_b \leq 1, \qquad g \in G, \tag{A.1b}$$

$$x_b \in \{0,1\}, \qquad b \in B, \tag{A.1c}$$

where x_b are the decision variables expressing whether a bid b is selected. The goal of WSP (A.1a) is to maximise the weighted sum of the bids. Constraint (A.1b) ensures that selected bids are pairwise disjoint.

Appendix B

Explainability Supplementary Material

B.1 IIS example

In Table B.1, we provide the mathematical encoding of the constraints within the IIS of the running example in the left table of Figure 6.2. Notice that each precedence constraint c_5, c_6, and c_7 is a system of multiple mathematical constraints, instead of a single mathematical constraint. We join them into a single restriction because we need to enforce multiple mathematical constraints per semester to represent the precedence relation of "*planning a subject after completing its prerequisite*" (see DPP formalisation in Section 5.2).

B.2 Query Examples

In this section, we describe the query types employed in our experiments for each optimisation problem addressed in this thesis and both benchmark problems used to test LLMxMIP, the RCPSP and the WDP.

B.2.1 Aggregation of Value Systems

In the case of the Aggregation of Value Systems, the queries concern the obtained consensus, either in terms of how an action promotes (or demotes) a moral value or

Table B.1: *IIS* from Example 6.2.3. We indicate the constraint type (in different colours) and the mathematical encoding of the constraints.

Id	Type	Constraint Encoding
q	Query	$x_{PF,4} = 1$
c_1	Completion	$\sum_{l=1}^{6} x_{PF,l} \leq 1$
c_2	Offer	$\sum_{l=1,3,5} x_{PT,l} = 0$
c_3	Offer	$\sum_{l=1,3,5} x_{OOP,l} = 0$
c_4	Offer	$\sum_{l=2,4,6} x_{OSP,l} = 0$
c_5	Precedence	$x_{PF,1} \geq x_{PT,2}, \sum_{l=1}^{3} x_{PF,l} \geq x_{PT,4}$
c_6	Precedence	$x_{PF,1} \geq x_{OOP,2}, \sum_{l=1}^{3} x_{PF,l} \geq x_{OOP,4}$
c_7	Precedence	$\sum_{l=1}^{2} \sum_{u=\{PT,OOP\}} x_{u,l} \geq x_{OSP,3}, \sum_{l=1}^{4} \sum_{u=\{PT,OOP\}} x_{u,l} \geq x_{OSP,5}$
c_8	Core	$\sum_{l=1}^{6} x_{OSP,l} = 1$

how strongly one value is preferred over another. Table B.2 summarises these query types and their corresponding constraint encodings.

For instance, the query Q1 "*Why does action a promote value v less than α'?*" is handled by **enforcing** the value interpretation to be higher than α'. This query refers to a **single consensus assignment**, namely the aggregated impact of one action on one value. Other queries concern **pairwise value preferences**. For example, query type Q3 "*Why is the preference of value v over value v' less than $\hat{p}$?*" is translated into enforcing the value preference of v over value v' to be greater than threshold $\hat{p}$. Intuitively, the user challenges the degree to which the consensus prefers v over v'. Similarly, when the query asks *Why is the preference of value v over value v' more than $\hat{p}$?*", we enforce a lower assignment on the consensus of the value preference.

B.2.2 Research Project Planning Problem

Table B.3 presents the set of possible queries for the RPP. In the context of the RPP, decision variables $x_{r,w,m}$ represent the number of hours that researcher r dedicates to work package w during month m. Hence, most queries concern whether a given dedication is "*too low*" or "*too high*" with respect to a user's alternative dedication.

For example, the query Q1, "*Why is researcher r assigned less than h hours in work package w during month m?*" is tackled by **enforcing** $x_{r,w,m} \geq h$. In other words,

Table B.2: Set of possible user questions, query constraints and constraint encodings for the Aggregation of Value Systems formalisation in the European Value Study case.

	Question	**Query constraints**	**Constraint encoding (C_Q)**
Q1.	Why does action a promote [1] value v *less* than α'?	**enforce** the consensus $\alpha_v^S(a)$ to be higher than α'	$\lvert\alpha_v^S(a)\rvert \geq \lvert\alpha'\rvert$
Q2.	Why does action a promote [2] value v *more* than α'?	**enforce** the consensus $\alpha_v^S(a)$ to be lower than α'	$\lvert\alpha_v^S(a)\rvert \leq \lvert\alpha'\rvert$
Q3.	Why is the preference of value v over value v' less than $\hat{p}$?	**enforce** the consensus $P_S[v, v']$ to be higher than $\hat{p}$	$P_S[v, v'] \geq \hat{p}$
Q4.	Why is the preference of value v over value v' more than $\hat{p}$?	**enforce** the consensus $P_S[v, v']$ to be lower than $\hat{p}$	$P_S[v, v'] \leq \hat{p}$

we impose the user-suggested alternative in which researcher r dedicates at least h hours to w in month m, and we analyse why such an alternative conflicts with the optimal plan. This query refers to a **single assignment** in the solution. Conversely, the query Q2 "*Why is researcher r assigned more than h hours in work package w during month m?*" is encoded by enforcing $x_{r,w,m} \leq h$, thus constraining the solution to allocate fewer hours than in the optimal plan.

Other queries concern **multiple assignments**. For instance, query Q3 "*Why is researcher r assigned less than h hours in month m?*" is encoded by enforcing the aggregation of the dedication over all work packages in which r is involved during month m to be greater than h. Similarly, queries Q5 and Q6 aggregate over the entire duration of a work package, while Q9 and Q10 aggregate over all the work packages and the whole planning horizon for a given researcher. In each case, we **enforce** an alternative lower or upper bound on the total dedication. Finally, some queries focus on the work packages' dedication rather than the researchers' dedication. For example, query Q7 "*Why is the assigned dedication to work package w less than h hours during month m?*" is encoded by enforcing more hours in the sum of the dedication over all researchers involved in w during month m.

Table B.3: Set of possible user questions, query constraints and constraint encodings for the Research Project Planning Problem formalisation.

	Question	Query constraints	Constraint encoding (C_Q)
Q1.	Why is researcher r assigned less than h hours in work package w during month m?	**enforce** researcher r to dedicate more than h hours in work package w at month m	$x_{r,w,m} \geq h$
Q2.	Why is researcher r assigned more than h hours in work package w during month m?	**enforce** researcher r to dedicate less than h hours in work package w at month m	$x_{r,w,m} \leq h$
Q3.	Why is researcher r assigned less than h hours in month m?	**enforce** researcher r to dedicate more than h hours in month m	$\sum_{w \in W \mid \text{involved}(r,w)=\top} x_{r,w,m} \geq h$
Q4.	Why is researcher r assigned more than h hours in month m?	**enforce** researcher r to dedicate less than h hours in month m	$\sum_{w \in W \mid \text{involved}(r,w)=\top} x_{r,w,m} \leq h$
Q5.	Why is researcher r assigned less than h hours in work package w?	**enforce** researcher r to dedicate more than h hours in work package w	$\sum_{m \in [w_s,w_e]} x_{r,w,m} \geq h$
Q6.	Why is researcher r assigned more than h hours in work package w?	**enforce** researcher r to dedicate less than h hours in work package w	$\sum_{m \in [w_s,w_e]} x_{r,w,m} \leq h$
Q7.	Why is the assigned dedication to work package w less than h hours during month m?	**enforce** to plan more than h hours to work package w during month m	$\sum_{r \in R \mid \text{involved}(r,w)=\top} x_{r,w,m} \geq h$
Q8.	Why is the assigned dedication to work package w more than h hours during month m?	**enforce** to plan less than h hours to work package w in month m	$\sum_{r \in R \mid \text{involved}(r,w)=\top} x_{r,w,m} \leq h$
Q9.	Why is the assigned dedication to researcher r less than h hours along the planning horizon?	**enforce** researcher r to dedicate more than h hours	$\sum_{w \in W \mid \text{involved}(r,w)=\top} \sum_{m \in [w_s,w_e]} x_{r,w,m} \geq h$
Q10.	Why is the assigned dedication to researcher r more than h hours along the planning horizon?	**enforce** researcher r to dedicate less than h hours	$\sum_{w \in W \mid \text{involved}(r,w)=\top} \sum_{m \in [w_s,w_e]} x_{r,w,m} \leq h$

B.2.3 Degree Planning Problem

Table B.4 lists possible queries for the DPP. The table also shows the translations of queries into (linear) query constraints.

Query Q1 "*Why is subject u planned at semester l?*" translates into **vetoing** the planning of subject u at semester l (i.e., $x_{u,l} = 0$) to enforce an alternative solution to the optimal solution. Notice that this query refers to a **single assignment** occurring in the optimal solution. To comply with query Q7 "*Why is the group of subjects U planned at semester l?*" we must **veto** the planning of the entire set of subjects U at semester l (i.e., $\sum_{u \in U} x_{u,l} < |U|$). In this case, the query refers to the planning of multiple subjects, i.e., **multiple assignments**, in the optimal solution. When addressing query Q2 "*Why is subject u **not** planned to be completed at semester l?*" we must **enforce** the hypothetical planning of unit u, which the current (optimal)

Table B.4: Set of possible user questions, query constraints and constraint encodings for the Degree Planning Problem formalisation.

	Question	**Query constraints**	**Constraint encoding (C_Q)**
Q1.	Why is subject u planned at semester l?	**veto** to plan u at semester l	$x_{u,l} = 0$
Q2.	Why is subject u not planned at semester l?	**enforce** to plan u at semester l	$x_{u,l} = 1$
Q3.	Why is subject u **not** planned before semester l?	**enforce** to plan u before semester l	$\sum_{i=0}^{l-1} x_{u,i} = 1$
Q4.	Why is subject u **not** planned after semester l?	**enforce** to plan u after l	$\sum_{i=l+1}^{l_n} x_{u,i} = 1$
Q5.	Why is subject u planned?	**veto** u	$\sum_{l \in \mathcal{L}} x_{u,l} = 0$
Q6.	Why is subject u not planned?	**enforce** u	$\sum_{l \in \mathcal{L}} x_{u,l} = 1$
Q7.	Why is the group of subjects U planned at semester l?	**veto** to plan all $u \in U$ at semester l	$\sum_{u \in U} x_{u,l} < \lvert U \rvert$
Q8.	Why is the group of subjects U **not** planned at l?	**enforce** to plan all $u \in U$ at semester l	$\sum_{u \in U} x_{u,l} = \lvert U \rvert$
Q9.	Why is subject u planned at semester l instead of unit u'?	**enforce** to plan u' at l and **veto** u at l	$x_{u',l} = 1$, $x_{u,l} = 0$
Q10.	Why is subject u planned instead of subject u'?	**enforce** u' and **veto** u	$\sum_{l \in \mathcal{L}} x_{u',l} = 1$, $\sum_{l \in \mathcal{L}} x_{u,l} = 0$

degree plan disregards. To answer the query Q8 "*Why is subject u planned to be completed at semester l instead of subject u'?*", we must: (1) enforce subject u'; and (2) veto u at semester u. Thus, we consider combinations of **enforce** and **veto** constraints to answer more specific queries about value assignments that did and did not occur.

B.2.4 Resource-Constrained Project Scheduling Problem

Table B.5 lists possible queries for the RCPSP. The table also shows the translations of queries into (linear) query constraints.

In the case of the RCPSP, for example, the query "*Why is activity j scheduled to be completed at time t?*" translates into **vetoing** the completion of activity j at time t (i.e., $x_{j,t} = 0$) to enforce an alternative solution to the optimal solution. Notice that this query refers to a **single assignment** occurring in the optimal solution.

Additionally, to comply with the query "*Why is the group of activities A scheduled*

Table B.5: Set of possible user questions, query constraints and constraint encodings for the Resource-Constrained Project Scheduling Problem formalisation.

	Question	Query constraints	Constraint encoding (C_Q)
Q1.	Why is activity j completed at time t?	**veto** j complete at time t	$x_{j,t} = 0$
Q2.	Why is activity j **not** completed at time t?	**enforce** j complete at time t	$x_{j,t} = 1$
Q3.	Why is activity j **not** completed before time t?	**enforce** j complete before time t	$\sum_{t'=EF_j}^{t-1} x_{j,t'} = 1$
Q4.	Why is activity j **not** completed after time t?	**enforce** j to complete after t	$\sum_{t'=t+1}^{LF_j} x_{j,t'} = 1$
Q5.	Why is the group of activities A completed at t?	**veto** to complete $j \in A$ at time t	$\sum_{j \in A} x_{j,t} < \lvert A \rvert$
Q6.	Why is the group of activities A **not** completed at t?	**enforce** to complete all of them at time t	$\sum_{j \in A} x_{j,t} = \lvert A \rvert$
Q7.	Why is activity j completed at t instead of t'?	**enforce** complete j at t' and **veto** j at t	$x_{j,t'} = 1$, $x_{j,t} = 0$
Q8.	Why is activity j completed at t instead of j'?	**enforce** j' and **veto** j complete at time t	$x_{j',t} = 1$, $x_{j,t} = 0$

to be completed at time t?" we must **veto** the completion of all activities in A at t (i.e., $\sum_{j \in A} x_{j,t} = 0$). In this case, the query refers to the schedule of multiple activities, i.e., **multiple assignments**, in the optimal solution. Furthermore, when addressing the query "*Why is activity j **not** scheduled to be completed at time t?*" we must **enforce** the hypothetical scheduling of activity j, which the current scheduling disregards.

Moreover, to answer the query "*Why is activity j scheduled to be completed at time t instead of activity j'?*", we must: (1) enforce activity j' ; and (2) veto j at time t. Thus, we consider combinations of **enforce** and **veto** constraints to answer more specific queries about value assignments that did and did not occur.

B.2.5 Winner Determination Problem

Table B.6 presents the set of possible queries for the WDP, together with their translation into (linear) query constraints.

In the context of the WDP, the query "*Why is bid b selected?*" is addressed by **vetoing** bid b, that is, by adding the constraint $x_b = 0$ to rule out the current

Table B.6: Set of possible queries, the query constraints and the constraint encoding of the Winner Determination Problem formalisation.

Question	**Query constraints**	**Constraint encoding,** C_Q
Q1. Why is bid b selected?	**veto** b	$x_b = 0$
Q2. Why is bid b **not** selected?	**enforce** b	$x_b = 1$
Q3. Why is bid b selected instead of bid b'?	**enforce** b', **veto** b	$x_{b'} = 1, x_b = 0$
Q4. Why is good g selected?	**veto** g	$\sum_{b \in B \mid g \in G_b} x_b = 0$
Q5. Why is good g **not** selected?	**enforce** g	$\sum_{b \in B \mid g \in G_b} x_b = 1$
Q6. Why is good g selected instead of good g'?	**enforce** g', **veto** g	$\sum_{b \in B \mid g' \in G_b} x_b = 1$, $\sum_{b \in B \mid g \in G_b} x_b = 0$
Q7. Why are goods g and g' in the same bid?	**veto** g and g' in the same selected bid	$\sum_{b \in B \mid g', g \in G_b} x_b = 0$
Q8. Why are goods g and g' **not** in the same bid?	**enforce** g and g' in the same bid	$\sum_{b \in B \mid g', g \in G_b} x_b = 1$

optimal choice and enforce an alternative solution. This query refers to a **single assignment** in the optimal solution, namely the selection of a specific bid.

Similarly, the query *Why is bid b **not** selected?*" is handled by **enforcing** the selection of b (i.e., $x_b = 1$), thereby imposing a hypothetical alternative that contradicts the optimal assignment. Another example is query Q3, "*Why is bid b selected instead of bid b′?*", which requires a combination of constraints: we **enforce** b' and simultaneously **veto** b. Hence, these queries involve combinations of enforce and veto constraints to compare assignments that did and did not occur in the optimal solution.

Beyond individual bids, some queries concern **goods** rather than bids. For instance, the query *Why is good g selected?*" is translated into **vetoing** all bids that contain g, that is, imposing $\sum_{b \in B \mid g \in G_b} x_b = 0$. In contrast, the query *Why is good g **not** selected?*" is handled by **enforcing** that exactly one bid containing g be selected, i.e., $\sum_{b \in B \mid g \in G_b} x_b = 1$. These queries refer to **multiple assignments**, since the selection of a good depends on multiple bids which include it.

Overall, as in the RCPSP, all queries in the WDP can be encoded through combinations of **enforce** and **veto** constraints, applied either to single decision variables (bids) or to aggregated sets of variables (goods and their co-occurrence in bids).

Table B.7: Prompt using the Graph of Reasons to answer the query of the running example.

The bachelor's degree planning problem is the problem of finding a plan of university subjects that respect some precedence and university restrictions while maximising the skills learnt. Specifically, the goal is to acquire the required level for the student's desired job role by coursing subjects. In this context, provide an explanation to answer the following question about the solution to the 6-semester bachelor's degree planning problem. To do so, use the reasons below, which are related to the bachelor's degree restrictions. Moreover, such reasons are connected to each other like nodes in a graph. Use the reasons and their connections to write the explanation.

Partial Solution:

Subject	Semester	Offer	Predecessors	Core
Prog. Fundamentals (PF)	2 (Spr.)	Both	-	Yes
Prog. Techniques (PT)	4 (Spr.)	Spr.	PF	Yes
Object Oriented Prog. (OOP)	–	Spr.	PF	No
Operating Systems Prog. (OSP)	5 (Aut.)	Aut.	PT or OOP	Yes

Question: "Why is Programming Fundamentals not planned at semester 4? "

Reasons:
0: Programming Fundamentals is not planned at semester 4.
1: Programming Fundamentals cannot be planned more than once.
2: Programming Fundamentals must be planned before Programming Techniques.
3: Programming Fundamentals must be planned before Object Oriented Programming.
4: Programming Techniques cannot be planned during autumn semesters.
5: Programming Techniques or Object Oriented Programming must be planned before Operating Systems Programming.
6: Object Oriented Programming cannot be planned during autumn semesters
7: Operating Systems Programming cannot be planned during spring semesters.
8: Operating Systems Programming is compulsory.

Connections:
(0-1), (1-2), (1-3), (2-3), (2-4), (2-5), (3-5), (3-6), (4-5), (5-6), (5-7), (5-8), (7-8)

Write a brief explanation without listing reasons and do not refer to the number of the reason.

B.3 LLM Prompt

Now, we provide the prompt templates used to generate the explanations of the experimental evaluation (see Section 6.3). In detail, we provide two prompt templates: (i) a template that uses a graph of reason to build an explanation, and (ii) a template that uses the translation of the mathematical constraints of the IIS.

Table B.7 shows a prompt using the graph of reasons to generate an explanation. Table B.8 shows a prompt using the translation of the constraints within the IIS to generate an explanation. Both prompts are used to generate the explanations of

Table B.8: Prompt using the IIS (with their constraints translated) to answer the query of the running example.

The bachelor's degree planning problem is the problem of finding a plan of university subjects that respect some precedence and university restrictions while maximising the skills learnt. Specifically, the goal is to acquire the required level for the student's desired job role by coursing subjects. In this context, provide an explanation to answer the following question about the solution to the 6-semester bachelor's degree planning problem. To do so, use the reasons below, which are related to the bachelor's degree restrictions.

Partial Solution:

Subject	Semester	Offer	Predecessors	Core
Prog. Fundamentals (PF)	2 (Spr.)	Both	-	Yes
Prog. Techniques (PT)	4 (Spr.)	Spr.	PF	Yes
Object Oriented Prog. (OOP)	–	Spr.	PF	No
Operating Systems Prog. (OSP)	5 (Aut.)	Aut.	PT or OOP	Yes

Question: "Why is Programming Fundamentals not planned at semester 4? "

Reasons:
0: Programming Fundamentals is not planned at semester 4.
1: Programming Fundamentals cannot be planned more than once.
2: Object Oriented Programming cannot be planned during autumn semesters.
3: Programming Techniques cannot be planned during autumn semesters.
4: Operating Systems Programming cannot be planned during spring semesters.
5: Programming Fundamentals must be planned before Object Oriented Programming.
6: Programming Fundamentals must be planned before Programming Techniques.
7: Programming Techniques or Object Oriented Programming must be planned before Operating Systems Programming.
8: Operating Systems Programming is compulsory.

Write a brief explanation without listing reasons and do not refer to the number of the reason.

the running example, which are shown in Table 6.6. Notice that the only difference between both prompts is that the prompt using the graph of reasons displays the relations between reasons. Additionally, the order of the reasons is different because, in the case of the graph of reasons, we organise the constraints according to their scope, as mentioned in Section 6.2.4. Lastly, in the case of the prompt using the graph of reasons, we add the sentence "*Moreover, such reasons are connected to each other like nodes in a graph.* " to provide the necessary context to the LLM.

Appendix C

Runtime Prediction Supplementary Material

C.1 Problem Instance Features

There is no automatic way to identify the features of a problem instance. To do so, we use specific problem features previously defined to predict the runtime of solving the WDP [114]. For the RCPSP, we use features defined to generate instances of the problem [109, 108]. In addition, we add features that have a statistical relationship with the hardness of solving the problem instance.

C.1.1 WDP Features

Table C.1 shows a list of the problem-specific features for the WDP. The first two features indicate the number of bids and goods of the instance, which in turn are the number of variables and constraints of the MILP encoding.

Secondly, we can define a bipartite graph representing the relations between bids (variables) and goods (constraints) of a WDP instance [114]. The nodes of the graphs are both the bids and the goods of the problem, and an edge indicates that a good is in a bid. We measure the node degree for each node and we compute the minimum, maximum, average, and standard deviation for each group of nodes. Intuitively, the average bid node degree indicates the average number of goods per bid. In contrast, the average good node degree indicates the average number of bids a given node is within.

Table C.1: Specific features for the WDP instances defined by Leyton-Brown, Nudelman and Shoham [114].

1-2 **Number of bids** (number of variables) and **number of goods** (number of constraints)

Bid-Good Graph Features

3-6 **Bid node degree statistics:** min, max, mean, and standard deviation of the bid node degrees.

7-10 **Good node degree statistics:** min, max, mean, and standard deviation of the good node degrees.

Bid Graph Features

11 **Edge Density:** number of edges in the BG divided by the number of edges in a complete graph (i.e., a *clique*).

12-15 **Node degree statistics:** min, max, mean, and standard deviation of the node degrees.

16-19 **Clustering Coefficient statistics.** A measure of "local cliqueness". For each node v_i, we calculate the number of edges between the neighbours N_i divided by the number of possible edges of the neighbours $k_i(k_i-1)/2$, where k_i is the number of neighbours. Thus, the clustering coefficient of a node C_i is $\frac{2\cdot|\{e_{jk}:v_j,v_k\in N_i,e_{jk}\in E\}}{k_i(k_i-1)}$. We record the min, max, mean, and standard deviation of the clustering coefficients.

20-22 **Minimum path length:** max, mean, and standard deviation of the minimum path length, over all pairs of bids.

23 **Ratio of the average clustering coefficient to the average minimum path length.**

24-27 **Node eccentricity statistics:** min, max, mean, and standard deviation of the node eccentricity. The node eccentricity is the length of the shortest path to its furthest node.

28 **Graph width.** A measure of how sparse the graph is, also known as "*graph degeneracy*". See [57].

LP-Based Features

29-31 L_1, L_2, and L_∞ norms of the integer slack vector.

Price-Based Features

32-35 **Bid price statistics:** min, max, mean, and standard deviation of the bid prices.

36-37 **Price per number of goods statistics:** mean and standard deviation of the bid prices per number of goods, i.e., $w_b/|G_b|$.

38-39 **Price per square root of the number of goods statistics:** mean and standard deviation of the bid prices per the square root of the number of goods, i.e., $w_b/\sqrt{|G_b|}$.

Furthermore, we define a second graph, called the Bid Graph, which represents relations between bids [114]. An edge between bids indicates that they share at least one good. We measure different statistics to capture the structure of such a graph. Such features encompass the edge density, the node degree, the clustering coefficient, the minimum path length (between all pairs of nodes), the node eccentricity, and the graph width [57].

In addition, we compute metrics related to the LP relaxation of the problem instance. We do so since CPLEX uses the LP relaxation as a search instance [114]. Thus, we solve the LP relaxation of a WDP instance and we compute the integer slack vector of the solution, i.e., $min(|x_i|, |1-x_i|)$. Such a vector gives us knowledge about the goodness of the initial non-integer solution. We measure the ℓ_1, ℓ_2, and ℓ_∞ norms of the integer slack vector.

Finally, we compute features related to the distribution of bid prices. In addition, we measure the price per good within a bid.

Table C.2: Specific features for the RCPSP instances.

1-2 **Number of variables** and **number of constraints**.

Variable-Constraint Graph Features

3-6 **Variable node degree statistics:** min, max, mean, and standard deviation of the variable node degrees.

7-10 **Constraint node degree statistics:** min, max, mean, and standard deviation of the constraint node degrees.

Activity Features

11 **Number of activities.**

12-15 **Processing time statistics:** min, max, mean, and standard deviation of the activity processing times.

16-18 **Successors statistics:** max, mean, and standard deviation of the number of activity successors.

19-22 **Time range statistics.** Range of the activity time intervals, i.e., $LF_j - EF_j$. We record the min, max, mean, and standard deviation.

23-25 **Number of resources per activity statistics.** max, mean, and standard deviation of the number of different types of resources required per activity.

26-28 **Number of resource units per activity statistics.** max, mean, and standard deviation of the number of resource units required per activity.

Resource Features

29 **Number of resources.**

30-33 **Resource availability statistics:** min, max, mean, and standard deviation of the number of resource units available each time period.

34 **Average resource factor:** the ratio of resource used. See [108] for details.

35 **Average resource strength:** the strength of resource constraints. See [108] for details.

Precedence Graph (PG) Features

36 **PG Edge Density:** number of edges in the PG divided by the number of edges in a complete graph.

37-40 **PG Node degree statistics:** min, max, mean, and standard deviation of the node degrees.

41-43 **PG Minimum path length:** max, mean, and standard deviation of the minimum path length, over all pairs of activities.

44-47 **PG Node eccentricity statistics:** min, max, mean, and standard deviation of the node eccentricity.

Activity-Resource Graph (ARG) Features

48 **ARG Edge Density:** number of edges in the ARG divided by the number of edges in a complete graph.

49-52 **ARG Node degree statistics:** min, max, mean, and standard deviation of the node degrees.

53-54 **ARG Clustering Coefficient statistics.** We record the mean and the standard deviation of the clustering coefficients.

55-57 **ARG Minimum path length:** max, mean, and standard deviation of the minimum path length, over all pairs of nodes.

58 **ARG Ratio of the average clustering coefficient to the average minimum path length.**

59-62 **ARG Node eccentricity statistics:** min, max, mean, and standard deviation of the node eccentricity.

LP-Based Features

63-65 L_1, L_2, and L_∞ norms of the integer slack vector.

C.1.2 RCPSP Features

Table C.2 shows a list of the problem-specific features for the RCPSP. The first two features indicate the number of variables and constraints of the MILP encoding. Similarly to the WDP, we can define a bipartite graph representing the relations between variables and constraints of a RCPSP instance, and we extract the same metrics for such a problem.

Then, we define features to characterise the activities of a problem instance. We measure statistics in relation to their processing time, the number of successors, the time range interval an activity j can be completed, i.e., $LF_j - EF_j$, as well as the total number of resources and resource units an activity needs to be completed. In addition, we gather information about the resources. We measure the number of resource types, the number of resource units available for each time period, as well as the average resource factor and the average resource strength. Both metrics are used to generate different random instances of the RCPSP [108, 109].

Table C.3: Solving time prediction results for the different models and different problem instances. We report 10-fold cross-validation RMSE and MAE (lower values are better). Bold indicates the (significantly) best-performing model.

Prediction Domain	RMSE						MAE					
	LR	QR	RT	RF	SL	DL	LR	QR	RT	RF	SL	DL
WDP	0.27	0.20	0.25	0.17	**0.10**	0.14	0.18	0.35	0.13	0.10	**0.08**	0.12
RCPSP	0.49	**0.41**	0.55	**0.41**	0.45	0.56	0.36	0.52	0.36	**0.27**	0.35	0.48

Likewise, in the case of the WDP, we define a Precedence Graph. We do so to gather information about the precedence relations in the problem instance. In such a graph, the nodes of the graphs are the activities of the problem instance. The edges between activities indicate a precedence relationship between them. Following the case of the WDP, we measure statistics such as the edge density, the node degree, the minimum path length (between all pairs of nodes), and the node eccentricity.

Furthermore, we define a second graph, called the Activity-Resource Graph, which represents relations between activities that share resources. Each node is an activity, and an edge between activities indicates that such activities share at least one resource in a given time period. In other words, an edge represents whether two activities are in the scope of a resource constraint. Similarly to the Precedence Graph, we measure the following statistics: the edge density, the node degree, the clustering coefficient, the minimum path length (between all pairs of nodes), and the node eccentricity.

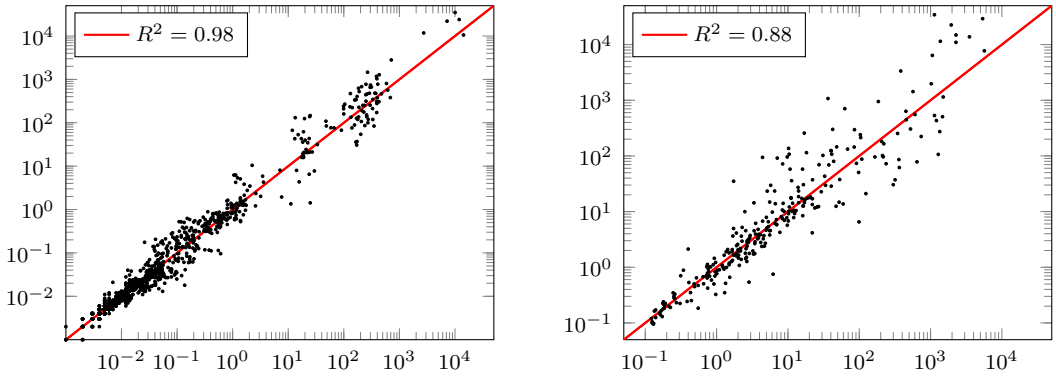
Finally, as we implemented in the WDP, we compute metrics related to the LP relaxation of the problem instance. We measure the ℓ_1, ℓ_2, and ℓ_∞ norms of the integer slack vector.

C.2 Solving Time Prediction

In this section, we show the results of applying the methodology of previous works on runtime prediction [116] to predict the solving time of the WDP and the RCPSP.

Following a well-known methodology in the literature [96], we predict solving times to evaluate whether the problem instance features properly characterise the optimisation problem's hardness. In Table C.3, we show the results of our predictions.

We show that both the RMSE and the MAE are low for all the prediction models, even for simple models such as LR, since the errors are $\ll 1$. This indicates that the runtime prediction is of the same order of magnitude as the ground-truth value. In detail, we show that the "*shallow learning*" model achieves the best performance on the WDP, closely followed by the deep learning model and the RF. For the RCPSP, the best models are QR with feature selection and the RF (although the RF has a lower MAE).



(a) Solving WDP

(b) Solving RCPSP

Figure C.1: Solving time predictions on previously unseen test instances for Random Forest (best performing model). The $x-$axis denotes the runtime prediction, and the $y-$axis denotes the real runtime. Each data point represents one instance. We show the Coefficient of Determination (R^2), which measures how well the model predicts runtime (1 indicates a perfect fit).

Figure C.1 shows the scatter plots and the adjusted R^2 of the solving time predictions of RF on test data for both MIP problems in Figures C.1a and C.1b. Notice that both Regression Coefficients (R^2) are close to 1.

References

[P1] R. Lera-Leri, F. Bistaffa, M. Serramia, M. Lopez-Sanchez and J. Rodriguez-Aguilar, 'Towards pluralistic value alignment: Aggregating value systems through ℓ_p-regression,' in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 780–788 (cit. on pp. 24, 25).

[P2] R. X. Lera-Leri, E. Liscio, F. Bistaffa, C. M. Jonker, M. Lopez-Sanchez, P. K. Murukannaiah, J. A. Rodriguez-Aguilar and F. Salas-Molina, 'Aggregating value systems for decision support,' *Knowledge-Based Systems*, vol. 287, p. 111 453, 2024 (cit. on p. 25).

[P3] R. X. Lera-Leri and F. Bistaffa, 'A robust and scalable approach to meet user preferences in research project planning,' in *Proceedings of the European Conference on Artificial Intelligence 2024*, IOS Press, 2024, pp. 4516–4523 (cit. on p. 25).

[P4] R. X. Lera-Leri, F. Bistaffa, A. Georgara and J. A. Rodríguez-Aguilar, 'Exploiting constraint reasoning to build graphical explanations for mixed-integer linear programming,' in *Proceedings of the International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*, Springer, 2025, pp. 39–57 (cit. on pp. 25, 27).

[P5] R. X. Lera-Leri, F. Bistaffa, T. Trescak and J. A. Rodríguez-Aguilar, 'Computing job-tailored degree plans towards the acquisition of professional skills,' *Annals of Operations Research*, vol. 351, pp. 2095–2128, 2025 (cit. on p. 25).

[P6] E. Liscio, R. Lera-Leri, F. Bistaffa, R. I. Dobbe, C. M. Jonker, M. Lopez-Sanchez, J. A. Rodriguez-Aguilar and P. K. Murukannaiah, 'Value inference in sociotechnical systems,' in *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '23, London, United Kingdom: IFAAMAS, 2023, pp. 1774–1780 (cit. on pp. 26, 82).

[P7] E. Liscio, R. Lera-Leri, F. Bistaffa, R. I. Dobbe, C. M. Jonker, M. Lopez-Sanchez, J. A. Rodriguez-Aguilar and P. K. Murukannaiah, 'Inferring values via hybrid intelligence,' in *HHAI 2023: Augmenting Human Intellect*, IOS Press, 2023, pp. 373–378 (cit. on p. 26).

[P8] T. Trescak, R. Lera-Leri, F. Bistaffa and J. A. Rodriguez-Aguilar, 'Agent-assisted life-long education and learning,' in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022 (cit. on pp. 9, 26, 115).

[1] D. Abel, J. MacGlashan and M. L. Littman, 'Reinforcement learning as a framework for ethical decision making,' in *AAAI Workshop: AI, Ethics, and Society*, 2016 (cit. on p. 44).

[2] A. Adadi and M. Berrada, 'Peeking inside the black-box: A survey on explainable artificial intelligence (xai),' *IEEE access*, vol. 6, pp. 52 138–52 160, 2018 (cit. on p. 40).

[3] D. Adil, R. Peng and S. Sachdeva, 'Fast, provably convergent IRLS algorithm for p-norm linear regression,' in *Proceedings of Advances in Neural Information Processing Systems*, 2019, pp. 14 189–14 200 (cit. on pp. 46, 47, 68, 82).

[4] A. Afzal, J. Vladika, D. Braun and F. Matthes, 'Challenges in domain-specific abstractive summarization and how to overcome them,' *arXiv preprint arXiv:2307.00963*, 2023 (cit. on pp. 163, 166).

[5] N. Ajmeri *et al.*, 'Engineering multiagent systems for ethics and privacy-aware social computing.,' 2018 (cit. on p. 44).

[6] F. Alizadeh and D. Goldfarb, 'Second-order cone programming,' *Mathematical Programming*, vol. 95, Dec. 2001 (cit. on p. 34).

[7] N. Andréasson, A. Evgrafov and M. Patriksson, *An introduction to continuous optimization: foundations and fundamental algorithms*. Courier Dover Publications, 2020 (cit. on p. 30).

[8] E. Andrejczuk, F. Bistaffa, C. Blum, J. A. Rodríguez-Aguilar and C. Sierra, 'Synergistic team composition: A computational approach to foster diversity in teams,' *Knowledge-Based Systems*, vol. 182, p. 104 799, 2019 (cit. on p. 217).

[9] M. ApS, 'Mosek optimization toolbox for matlab,' *User's Guide and Reference Manual, Version*, vol. 4, no. 1, p. 116, 2019 (cit. on p. 7).

[10] O. Araque, L. Gatti and K. Kalimeri, 'Moralstrength: Exploiting a moral lexicon and embedding similarity for moral foundations prediction,' *Know.-Based Syst.*, vol. 191, no. C, Mar. 2020, ISSN: 0950-7051. DOI: 10.1016/j.knosys.2019.105184. [Online]. Available: https://doi.org/10.1016/j.knosys.2019.105184 (cit. on p. 83).

[11] J. S. Armstrong and L.-R. Forecasting, 'From crystal ball to computer,' *New York ua*, vol. 348, 1985 (cit. on p. 100).

[12] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, 'Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai,' *Information fusion*, vol. 58, pp. 82–115, 2020 (cit. on p. 2).

[13] Australian Digital Transformation Agency, *Aps digital career pathways dataset*, https://data.gov.au/data/dataset/aps-digital-career-pathways. Online, 2019 (cit. on pp. 136, 169).

[14] A. Balakrishnan, D. Bouneffouf, N. Mattei and F. Rossi, 'Incorporating behavioral constraints in online AI systems,' in *Proceedings of AAAI Conference on Artificial Intelligence*, 2019, pp. 3–11 (cit. on p. 44).

[15] K. Baum and M. Slavkovik, 'Aggregation problems in machine ethics and ai alignment,' in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, vol. 8, 2025, pp. 355–366 (cit. on pp. 12, 212).

[16] G. Beliakov, H. B. Sola and T. C. Sánchez, *A practical guide to averaging functions*. Springer, 2016, vol. 329 (cit. on p. 211).

[17] A. Ben-Tal and A. Nemirovski, 'Robust convex optimization,' *Mathematics of Operations Research*, vol. 23, no. 4, pp. 769–805, 1998 (cit. on pp. 18, 89).

[18] T. J. M. Bench-Capon and K. Atkinson, 'Abstract argumentation and values,' in *Argumentation in Artificial Intelligence*, 2009, pp. 45–64 (cit. on pp. 11, 45, 52, 205).

[19] J. Bentham, *An Introduction to the Principles of Morals and Legislation*. London: Payne and Son, 1789 (cit. on pp. 11, 50).

[20] F. Bistaffa, C. Blum, J. Cerquides, A. Farinelli and J. A. Rodríguez-Aguilar, 'A computational approach to quantify the benefits of ridesharing for policy makers and travellers,' *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, pp. 119–130, 2019 (cit. on p. 217).

[21] J. Blazewicz, J. K. Lenstra and R. Kan, 'Scheduling subject to resource constraints: Classification and complexity,' *Discrete Applied Mathematics*, vol. 5, no. 1, pp. 11–24, 1983 (cit. on pp. 14, 37).

[22] G. Bocewicz, P. Golińska-Dawson, E. Szwarc and Z. Banaszak, 'Preventive maintenance scheduling of a multi-skilled human resource-constrained project's portfolio,' *Engineering Applications of Artificial Intelligence*, vol. 119, p. 105 725, 2023 (cit. on p. 1).

[23] B. Bogaerts, E. Gamba and T. Guns, 'A framework for step-wise explaining how to solve constraint satisfaction problems,' *Artificial Intelligence*, vol. 300, p. 103 550, 2021 (cit. on pp. 41, 42).

[24] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004 (cit. on pp. 16, 18, 31–34, 46, 63–65, 67, 72, 82, 87, 88, 92, 93, 96, 97, 130, 155, 206, 207).

[25] F. Brandt, V. Conitzer, U. Endriss, J. Lang and A. D. Procaccia, *Handbook of computational social choice*. Cambridge University Press, 2016 (cit. on pp. 16, 48).

[26] J. Brännmark, 'Moral disunitarianism,' *The Philosophical Quarterly*, vol. 66, no. 264, pp. 481–499, Nov. 2015. [Online]. Available: https://doi.org/10.1093/pq/pqv114 (cit. on pp. 45, 52).

[27] L. Breiman, 'Random forests,' *Machine learning*, vol. 45, pp. 5–32, 2001 (cit. on p. 194).

[28] L. Breiman, *Classification and regression trees*. Routledge, 2017 (cit. on p. 194).

[29] British Computer Society, *Skills framework for the information age*, https://sfia-online.or. Online, 2021 (cit. on pp. 19, 117, 135).

[30] P. Brucker, A. Drexl, R. Möhring, K. Neumann and E. Pesch, 'Resource-constrained project scheduling: Notation, classification, models, and methods,' *European Journal of Operational Research*, vol. 112, no. 1, pp. 3–41, 1999 (cit. on p. 38).

[31] P. Brucker and S. Knust, 'Resource-constrained project scheduling and timetabling,' in *International Conference on the Practice and Theory of Automated Timetabling*, Springer, 2000, pp. 277–293 (cit. on p. 39).

[32] S. Burer and A. Saxena, 'The milp road to miqcp,' *Mixed integer nonlinear programming*, pp. 373–405, 2011 (cit. on p. 30).

[33] C. L. Butler and A. Rothstein, *On conflict and consensus: A handbook on formal consensus decision making*. Takoma Park, 2006 (cit. on pp. 16, 48).

[34] C. Castro and S. Manzano, 'Variable and value ordering when solving balanced academic curriculum problems,' in *6th Workshop of the ERCIM WG on Constraints*, 2001 (cit. on p. 148).

[35] D. Chakrabarty and C. Swamy, 'Approximation algorithms for minimum norm and ordered optimization problems,' in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, 2019, pp. 126–137 (cit. on p. 33).

[36] N. Chakravarti, 'Some results concerning post-infeasibility analysis,' *European Journal of Operational Research*, vol. 73, no. 1, pp. 139–143, 1994 (cit. on pp. 42, 160).

[37] B. Chandrasekaran, M. C. Tanner and J. R. Josephson, 'Explaining control strategies in problem solving,' *IEEE Intelligent Systems*, vol. 4, no. 01, pp. 9–15, 1989 (cit. on p. 3).

[38] R. Chatila, V. Dignum, M. Fisher, F. Giannotti, K. Morik, S. Russell and K. Yeung, 'Trustworthy AI,' *Reflections on Artificial Intelligence for Humanity*, pp. 13–39, 2021 (cit. on p. 44).

[39] Y. Chevaleyre, P. Dunne, U. Endriss, J. Lang, M. Lemaitre, N. Maudet, J. Padget, S. Phelps, J. Rodrguez-Aguilar and P. Sousa, 'Issues in multiagent

resource allocation,' *Informatica*, vol. 30, pp. 3–31, 2006, ISSN: 0350-5596 (cit. on p. 50).

[40] Y. Chevaleyre, U. Endriss, J. Lang and N. Maudet, 'A short introduction to computational social choice,' in *International conference on current trends in theory and practice of computer science*, Springer, 2007, pp. 51–69 (cit. on pp. 16, 48).

[41] J. W. Chinneck, 'Analyzing infeasible nonlinear programs,' *Computational Optimization and Applications*, vol. 4, pp. 167–179, 1995 (cit. on p. 36).

[42] J. W. Chinneck, 'Finding a useful subset of constraints for analysis in an infeasible linear program,' *INFORMS Journal on Computing*, vol. 9, no. 2, pp. 164–174, 1997 (cit. on p. 36).

[43] J. W. Chinneck, *Feasibility and Infeasibility in Optimization:: Algorithms and Computational Methods*. Springer Science & Business Media, 2007, vol. 118 (cit. on pp. 36, 154, 185).

[44] J. W. Chinneck and E. W. Dravnieks, 'Locating minimal infeasible constraint sets in linear programs,' *ORSA Journal on Computing*, vol. 3, no. 2, pp. 157–168, 1991 (cit. on p. 36).

[45] R. M. Chisholm, 'Supererogation and offence: A conceptual scheme for ethics,' *Ratio (Misc.)*, vol. 5, no. 1, 1963 (cit. on p. 53).

[46] E. Comission, *General data protection regulation*, https://gdpr-info.eu/. Online, 2016 (cit. on pp. 3, 204, 209).

[47] E. Comission, *Ethics guidelines for trustworthy ai*, https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai. Online, 2019 (cit. on pp. 2, 44, 82, 151, 204, 209).

[48] E. Comission, *Artificial Intelligence Act*, https://digital-strategy.ec.europa.eu/en/library/proposal-regulation-laying-down-harmonised-rules-artificial-intelligence. Online, 2021 (cit. on p. 44).

[49] E. Comission, *Digital service act*, https://digital-strategy.ec.europa.eu/en/policies/digital-services-act-package. Online, 2022 (cit. on p. 3).

[50] W. D. Cook and M. Kress, *Ordinal information and preference structures: decision models and applications*. Prentice-Hall, Inc., 1992 (cit. on p. 48).

[51] W. D. Cook, M. Kress and L. M. Seiford, 'A general framework for distance-based consensus in ordinal ranking models,' *European Journal of Operational Research*, vol. 96, no. 2, pp. 392–397, 1997 (cit. on p. 48).

[52] W. D. Cook and L. M. Seiford, 'Priority ranking and consensus formation,' *Management Science*, vol. 24, no. 16, pp. 1721–1732, 1978 (cit. on p. 48).

[53] W. D. Cook and L. M. Seiford, 'On the borda-kendall consensus method for priority ranking problems,' *Management Science*, vol. 28, no. 6, pp. 621–637, 1982 (cit. on p. 48).

[54] D. Cooper, *Value Pluralism and Ethical Choice*. New York: St. Martin Press, Inc., 1993 (cit. on p. 52).

[55] I. I. CPLEX, 'V12. 1: User's manual for CPLEX,' *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009 (cit. on pp. 7, 31).

[56] S. De Vries and R. V. Vohra, 'Combinatorial auctions: A survey,' *INFORMS Journal on computing*, vol. 15, no. 3, pp. 284–309, 2003 (cit. on pp. 21, 29, 181, 217).

[57] R. Dechter, *Reasoning with probabilistic and deterministic graphical models: Exact algorithms*. Springer Nature, 2022 (cit. on pp. 163, 164, 229).

[58] E. Demeulemeester, B. Reyck, B. Foubert, W. Herroelen and M. Vanhoucke, 'New computational results on the discrete time/cost trade-off problem in project networks,' *Journal of The Operational Research Society*, vol. 49, pp. 1153–1163, Nov. 1998 (cit. on p. 38).

[59] E. L. Demeulemeester and W. S. Herroelen, 'An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem,' *European Journal of Operational Research*, vol. 90, no. 2, pp. 334–348, 1996 (cit. on p. 39).

[60] T. Developers, 'Tensorflow,' *Zenodo*, 2022 (cit. on p. 195).

[61] H. Drachsler, K. Verbert, O. Santos and N. Manouselis, 'Panorama of recommender systems to support learning,' in *Recommender Systems Handbook*, Springer, 2015, pp. 421–451 (cit. on pp. 9, 115, 213).

[62] A. Drexl, R. Nissen, J. H. Patterson and F. Salewski, 'Progen/πx–an instance generator for resource-constrained project scheduling problems with partially renewable resources and further extensions,' *European Journal of Operational Research*, vol. 125, no. 1, pp. 59–72, 2000 (cit. on p. 107).

[63] I. of Electrical and E. Engineers, *Ieee global initiative on ethics of autonomous and intelligent systems*, https://standards.ieee.org/industry-connections/ec/autonomous-systems.html. Online, 2019 (cit. on p. 44).

[64] A. Etminaniesfahani, H. Gu, L. M. Naeni and A. Salehipour, 'An efficient relax-and-solve method for the multi-mode resource constrained project scheduling problem,' *Annals of Operations Research*, pp. 1–28, 2023 (cit. on p. 116).

[65] *European values study*, Online, 2021. [Online]. Available: https://europeanvaluesstudy.eu (cit. on pp. 46, 47, 57, 82).

[66] A. Fenoy, F. Bistaffa and A. Farinelli, 'An attention model for the formation of collectives in real-world domains,' *Artificial Intelligence*, vol. 328, p. 104 064, 2024 (cit. on p. 217).

[67] F. J. Ferri, P. Pudil, M. Hatef and J. Kittler, 'Comparative study of techniques for large-scale feature selection,' in *Machine intelligence and pattern recognition*, vol. 16, Elsevier, 1994, pp. 403–413 (cit. on p. 194).

[68] C. Flathmann, B. G. Schelble, R. Zhang and N. J. McNeese, 'Modeling and guiding the creation of ethical human-AI teams,' in *Proceedings of AAAI/ACM Conference on AI, Ethics, and Society*, 2021, pp. 469–479 (cit. on p. 45).

[69] G. B. Folland, *Real analysis: modern techniques and their applications*. John Wiley & Sons, 1999 (cit. on p. 33).

[70] K. B. Francis, C. Howard, I. S. Howard, M. Gummerum, G. Ganis, G. Anderson and S. Terbeck, 'Virtual Morality: Transitioning from Moral Judgment to Moral Action?' *PLOS ONE*, vol. 11, no. 10, X. Wan, Ed., pp. 1–22, Oct. 2016, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0164374. [Online]. Available: https://dx.plos.org/10.1371/journal.pone.0164374 (cit. on p. 52).

[71] B. Friedman and D. G. Hendry, *Value sensitive design: Shaping technology with moral imagination*. 2019 (cit. on p. 44).

[72] I. Gabriel, 'Artificial intelligence, values, and alignment,' *Minds and Machines*, vol. 30, no. 3, pp. 411–437, 2020 (cit. on pp. 8, 44, 45, 52, 82, 205).

[73] E. Gamba, B. Bogaerts and T. Guns, 'Efficiently explaining csps with unsatisfiable subset optimization,' *Journal of Artificial Intelligence Research*, vol. 78, pp. 709–746, 2023 (cit. on p. 42).

[74] D. Gašević, S. Dawson and G. Siemens, 'Let's not forget: Learning analytics are about learning,' *TechTrends*, vol. 59, no. 1, pp. 64–71, 2015 (cit. on p. 115).

[75] G. Gendron, Q. Bao, M. Witbrock and G. Dobbie, 'Large language models are not strong abstract reasoners,' *arXiv preprint arXiv:2305.19555*, 2023 (cit. on pp. 21, 152, 163, 166).

[76] A. Georgara, J. A. Rodriguez Aguilar and C. Sierra, 'Building contrastive explanations for multi-agent team formation,' in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 516–524 (cit. on pp. 4, 20, 41, 152, 156, 158, 160, 201).

[77] J. Gleeson and J. Ryan, 'Identifying minimally infeasible subsystems of inequalities,' *ORSA Journal on Computing*, vol. 2, no. 1, pp. 61–63, 1990 (cit. on p. 36).

[78] J. González-Pachón and C. Romero, 'Distance-based consensus methods: A goal programming approach,' *Omega*, vol. 27, no. 3, pp. 341–347, 1999 (cit. on pp. 12, 16, 48, 49, 61).

[79] J. González-Pachón and C. Romero, 'Aggregation of ordinal and cardinal preferences: A framework based on distance functions,' *Journal of Multi-Criteria Decision Analysis*, vol. 15, no. 3-4, pp. 79–85, 2008 (cit. on pp. 12, 16, 48–50, 61).

[80] J. González-Pachón and C. Romero, 'The design of socially optimal decisions in a consensus scenario,' *Omega*, vol. 39, no. 2, pp. 179–185, 2011 (cit. on pp. 16, 48, 49, 61).

[81] J. González-Pachón and C. Romero, 'Properties underlying a preference aggregator based on satisficing logic,' *International Transactions in Operational Research*, vol. 22, no. 2, pp. 205–215, 2015 (cit. on pp. 12, 48).

[82] J. González-Pachón and C. Romero, 'Bentham, Marx and Rawls ethical principles: In search for a compromise,' *Omega*, vol. 62, pp. 47–51, 2016 (cit. on pp. 11, 46, 48–50, 61, 82).

[83] I. Griva, S. Nash and A. Sofer, *Linear and Nonlinear Optimization*. Siam, 2009, vol. 108 (cit. on p. 128).

[84] O. Guieu and J. W. Chinneck, 'Analyzing infeasible mixed-integer and integer linear programs,' *INFORMS Journal on Computing*, vol. 11, no. 1, pp. 63–77, 1999 (cit. on p. 36).

[85] R. Gunning, *The Technique of Clear Writing*. New York: McGraw-Hill, 1952 (cit. on p. 171).

[86] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2023. [Online]. Available: https://www.gurobi.com (cit. on pp. 31, 183).

[87] M. U. Hadi, R. Qureshi, A. Shah, M. Irfan, A. Zafar, M. B. Shaikh, N. Akhtar, J. Wu, S. Mirjalili *et al.*, 'A survey on large language models: Applications, challenges, limitations, and practical usage,' *Authorea Preprints*, 2023 (cit. on p. 166).

[88] S. Hartmann, 'Packing problems and project scheduling models: An integrating perspective,' *Journal of the Operational Research Society*, vol. 51, no. 9, pp. 1083–1092, 2000 (cit. on p. 39).

[89] S. Hartmann and D. Briskorn, 'A survey of variants and extensions of the resource-constrained project scheduling problem,' *European Journal of operational research*, vol. 207, no. 1, pp. 1–14, 2010 (cit. on p. 29).

[90] S. Hartmann and D. Briskorn, 'A survey of variants and extensions of the resource-constrained project scheduling problem,' *European Journal of Operational Research*, vol. 207, pp. 1–14, Nov. 2010 (cit. on pp. 37–39).

[91] S. Hartmann and D. Briskorn, 'An updated survey of variants and extensions of the resource-constrained project scheduling problem,' *European Journal of Operational Research*, vol. 297, no. 1, Feb. 2022 (cit. on pp. 21, 37, 38, 170).

[92] S. Hartmann and A. Drexl, 'Project scheduling with multiple modes: A comparison of exact algorithms,' *Networks: An International Journal*, vol. 32, no. 4, pp. 283–297, 1998 (cit. on p. 38).

[93] E. Herrera-Viedma, F. Herrera and F. Chiclana, 'A consensus model for multiperson decision making with different preference structures,' *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 32, no. 3, pp. 394–402, 2002 (cit. on pp. 16, 48, 211).

[94] P. L. Hill and D. K. Lapsley, 'Persons and situations in the moral domain,' *Journal of Research in Personality*, vol. 43, no. 2, pp. 245–246, 2009. DOI: 10.1016/j.jrp.2008.12.034 (cit. on pp. 45, 52).

[95] R. R. Hoffman, S. T. Mueller, G. Klein and J. Litman, 'Metrics for explainable ai: Challenges and prospects,' *arXiv preprint arXiv:1812.04608*, 2018 (cit. on p. 214).

[96] F. Hutter, L. Xu, H. H. Hoos and K. Leyton-Brown, 'Algorithm runtime prediction: Methods & evaluation,' *Artificial Intelligence*, vol. 206, pp. 79–111, 2014 (cit. on pp. 22, 181, 192, 193, 197, 201, 231).

[97] A. Ignatiev, N. Narodytska, N. Asher and J. Marques-Silva, 'From contrastive to abductive explanations and back again,' in *AIxIA 2020–Advances in Artificial Intelligence: XIXth International Conference of the Italian Association for Artificial Intelligence, Virtual Event, November 25–27, 2020, Revised Selected Papers*, Springer, 2021, pp. 335–355 (cit. on p. 152).

[98] A. Ignatiev, A. Previti, M. Liffiton and J. Marques-Silva, 'Smallest mus extraction with minimal hitting set dualization,' in *International Conference on Principles and Practice of Constraint Programming*, Springer, 2015, pp. 173–182 (cit. on pp. 42, 160, 184).

[99] A. Itten and N. Mouter, 'When digital mass participation meets citizen deliberation: Combining mini- and maxi-publics in climate policy-making,' *Sustainability*, vol. 14, no. 8, 2022 (cit. on pp. 46, 47, 54, 82).

[100] U. Junker, 'Quickxplain: Preferred explanations and relaxations for over-constrained problems,' in *Proceedings of the 19th national conference on Artifical intelligence*, 2004, pp. 167–172 (cit. on pp. 41, 42).

[101] M. Karnebogen and J. Zimmermann, 'Generation schemes for the resource-constrained project scheduling problem with partially renewable resources and generalized precedence constraints,' *Annals of Operations Research*, pp. 1–20, 2024 (cit. on p. 116).

[102] P. G. Keen, 'Decision support systems: A research perspective,' in *Decision support systems: Issues and challenges: Proceedings of an international task force meeting*, 1980, pp. 23–44 (cit. on p. 1).

[103] H. Kellerer, U. Pferschy and D. Pisinger, *Knapsack Problems*. Springer, 2004 (cit. on pp. 29, 183).

[104] H. Kerzner, *Project Management: a Systems Approach to Planning, Scheduling, and Controlling*. John Wiley & Sons, 2017 (cit. on p. 86).

[105] C. Knill, C. Adam and S. Hurka, *On the Road to Permissiveness? Change and Convergence of Moral Regulation in Europe*. Oxford University Press, 2015 (cit. on p. 59).

[106] I. Kola, R. Isufaj and C. M. Jonker, 'Does Personalization Help? Predicting How Social Situations Affect Personal Values,' in *HHAI2022: Augmenting Human Intellect*, 2022, pp. 157–169. DOI: 10.3233/FAIA220196 (cit. on pp. 45, 52).

[107] R. Kolisch and A. Drexl, 'Local search for nonpreemptive multi-mode resource-constrained project scheduling,' *IIE transactions*, vol. 29, no. 11, pp. 987–999, 1997 (cit. on p. 38).

[108] R. Kolisch and A. Sprecher, 'Psplib-a project scheduling problem library: Or software-orsep operations research software exchange program,' *European journal of operational research*, vol. 96, no. 1, pp. 205–216, 1997 (cit. on pp. 37, 170, 197, 228, 230).

[109] R. Kolisch, A. Sprecher and A. Drexl, 'Characterization and generation of a general class of resource-constrained project scheduling problems,' *Management science*, vol. 41, no. 10, pp. 1693–1703, 1995 (cit. on pp. 170, 197, 228, 230).

[110] A. Korikov and J. C. Beck, 'Counterfactual explanations via inverse constraint programming,' in *27th International Conference on Principles and Practice of Constraint Programming (CP 2021)*, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2021, pp. 35–1 (cit. on p. 41).

[111] A. Korikov, A. Shleyfman and C. Beck, 'Counterfactual explanations for optimization-based decisions in the context of the gdpr,' in *ICAPS 2021 Workshop on Explainable AI Planning*, 2021 (cit. on p. 41).

[112] H. Lakkaraju, D. Slack, Y. Chen, C. Tan and S. Singh, 'Rethinking explainability as a dialogue: A practitioner's perspective,' *arXiv:2202.01875*, 2022 (cit. on pp. 14, 190, 192).

[113] C. A. Le Dantec, E. S. Poole and S. P. Wyche, 'Values as lived experience,' in *Proceedings of the 27th international conference on Human factors in computing systems*, ser. CHI '09, New York: ACM Press, 2009, p. 1141 (cit. on p. 52).

[114] K. Leyton-Brown, E. Nudelman and Y. Shoham, 'Learning the empirical hardness of optimization problems: The case of combinatorial auctions,' in *Principles and Practice of Constraint Programming-CP 2002: 8th Interna-*

tional Conference, CP 2002 Ithaca, NY, USA, September 9–13, 2002 Proceedings 8, Springer, 2002, pp. 556–572 (cit. on pp. xi, 22, 181, 182, 191–193, 197, 228, 229).

[115] K. Leyton-Brown, E. Nudelman and Y. Shoham, 'Empirical hardness models: Methodology and a case study on combinatorial auctions,' *Journal of the ACM (JACM)*, vol. 56, no. 4, pp. 1–52, 2009 (cit. on pp. 22, 197).

[116] K. Leyton-Brown, M. Pearson and Y. Shoham, 'Towards a universal test suite for combinatorial auction algorithms,' in *Proceedings of the 2nd ACM conference on Electronic commerce*, 2000, pp. 66–76 (cit. on pp. 22, 183, 231).

[117] M. Liffiton, M. Mneimneh, I. Lynce, Z. Andraus, J. Marques-Silva and K. Sakallah, 'A branch and bound algorithm for extracting smallest minimal unsatisfiable subformulas,' *Constraints*, vol. 14, pp. 415–442, 2009 (cit. on pp. 42, 160).

[118] M. H. Liffiton and K. A. Sakallah, 'On finding all minimally unsatisfiable subformulas,' in *International conference on theory and applications of satisfiability testing*, Springer, 2005, pp. 173–186 (cit. on p. 42).

[119] Z. Lin, S. Guan, W. Zhang, H. Zhang, Y. Li and H. Zhang, 'Towards trustworthy llms: A review on debiasing and dehallucinating in large language models,' *Artificial Intelligence Review*, vol. 57, no. 9, p. 243, 2024 (cit. on p. 152).

[120] E. Liscio, A. E. Dondera, A. Geadau, C. M. Jonker and P. K. Murukannaiah, 'Cross-Domain Classification of Moral Values,' in *Findings of the Conference of the North American Chapter of the Association for Computational Linguistics*, ser. NAACL '22, Seattle, USA: ACL, 2022, pp. 2727–2745 (cit. on p. 83).

[121] E. Liscio, M. van der Meer, L. C. Siebert, C. M. Jonker and P. K. Murukannaiah, 'What Values should an Agent Align with? An Empirical Comparison of General and Context-Specific Values,' *Autonomous Agents and Multi-Agent Systems*, vol. 36, no. 1, p. 23, 2022. DOI: 10.1007/s10458-022-09550-0 (cit. on pp. 44, 45, 52).

[122] M. Lubin, E. Yamangil, R. Bent and J. P. Vielma, 'Extended formulations in mixed-integer convex programming,' in *International Conference on Integer Programming and Combinatorial Optimization*, Springer, 2016, pp. 102–113 (cit. on p. 30).

[123] J. Luo, J. C. Meyer and M. Knobbout, 'Reasoning about opportunistic propensity in multi-agent systems,' in *Lecture Notes in Computer Science*, vol. 10642, 2017, pp. 203–221 (cit. on pp. 11, 45, 52, 205).

[124] P. Meseguer, F. Rossi and T. Schiex, 'Soft constraints,' in *Foundations of Artificial Intelligence*, vol. 2, Elsevier, 2006, pp. 281–328 (cit. on p. 36).

[125] M. Mika, G. Waligóra and J. Węglarz, 'Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models,' *European Journal of Operational Research*, vol. 164, no. 3, pp. 639–668, 2005 (cit. on p. 38).

[126] T. Miller, 'Explanation in artificial intelligence: Insights from the social sciences,' *Artificial intelligence*, vol. 267, pp. 1–38, 2019 (cit. on pp. 3, 4, 14, 20, 41, 152, 159).

[127] T. Miller, 'Contrastive explanation: A structural-model approach,' *The Knowledge Engineering Review*, vol. 36, e14, 2021 (cit. on p. 152).

[128] S.-A. Mirnezami, R. Tavakkoli-Moghaddam, R. Shahabi-Shahmiri and M. Ghasemi, 'An integrated chance-constrained stochastic model for a preemptive multi-skilled multi-mode resource-constrained project scheduling problem: A case study of building a sports center,' *Engineering Applications of Artificial Intelligence*, vol. 126, p. 106 726, 2023 (cit. on p. 1).

[129] B. Mittelstadt, 'Principles alone cannot guarantee ethical AI,' *Nature Machine Intelligence*, vol. 1, no. 11, pp. 501–507, 2019 (cit. on p. 45).

[130] F. Molteni, R. Ladini, F. Biolcati, A. M. Chiesi, G. M. D. Sani, S. Guglielmi, M. Maraffi, A. Pedrazzani, P. Segatti and C. Vezzoni, 'Searching for comfort in religion: Insecurity and religious behaviour during the covid-19 pandemic in italy,' *European Societies*, vol. 23, S704–S720, 2021 (cit. on p. 59).

[131] N. Montes and C. Sierra, 'Value-guided synthesis of parametric normative systems,' 2021 (cit. on pp. 11, 44, 45).

[132] N. Montes and C. Sierra, 'Synthesis and properties of optimally value-aligned normative systems,' *Journal of Artificial Intelligence Research*, vol. 74, pp. 1739–1774, 2022 (cit. on pp. 11, 44).

[133] N. Mouter, P. Koster and T. Dekker, 'Contrasting the recommendations of participatory value evaluation and cost-benefit analysis in the context of urban mobility investments,' *Transportation Research Part A: Policy and Practice*, vol. 144, pp. 54–73, 2021 (cit. on pp. 46, 54).

[134] J. M. Mulvey, R. J. Vanderbei and S. A. Zenios, 'Robust optimization of large-scale systems,' *Operations Research*, vol. 43, no. 2, pp. 264–281, 1995 (cit. on pp. 89, 97).

[135] B. A. Myers, 'The importance of percent-done progress indicators for computer-human interfaces,' *ACM SIGCHI Bulletin*, vol. 16, no. 4, pp. 11–17, 1985 (cit. on pp. 22, 180).

[136] F. F.-H. Nah, 'A study on tolerable waiting time: How long are web users willing to wait?' *Behaviour & Information Technology*, vol. 23, no. 3, pp. 153–163, 2004 (cit. on pp. 22, 180).

[137] O. Nardi, A. Boixel and U. Endriss, 'A graph-based algorithm for the automated justification of collective decisions,' in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 935–943 (cit. on p. 42).

[138] R. Noothigattu, D. Bouneffouf, N. Mattei, R. Chandra, P. Madan, R. Kush, M. Campbell, M. Singh and F. Rossi, 'Teaching AI agents ethical values using reinforcement learning and policy orchestration,' *IBM Journal of Research and Development*, vol. 63, no. 4/5, pp. 6377–6381, 2019 (cit. on p. 44).

[139] P. Noriega, H. Verhagen, J. Padget and M. d'Inverno, 'Ethical online AI systems through conscientious design,' *IEEE Internet Computing*, vol. 25, no. 6, pp. 58–64, 2021 (cit. on p. 44).

[140] F. A. Ortega Riejos, M. Á. Pozo Montaño and J. Puerto Albandoz, 'Modelling and planning public cultural schedules for efficient use of resources,' *Computers and Operations Research*, vol. 58, pp. 9–23, 2015 (cit. on pp. 16, 48).

[141] M. Parker and J. Ryan, 'Finding the minimum weight iis cover of an infeasible system of linear inequalities,' *Annals of Mathematics and Artificial Intelligence*, vol. 17, no. 1, pp. 107–126, 1996 (cit. on p. 36).

[142] A. Pass-Lanneau, P. Bendotti and L. Brunod-Indrigo, 'Exact and heuristic methods for anchor-robust and adjustable-robust rcpsp,' *Annals of Operations Research*, pp. 1–34, 2023 (cit. on p. 116).

[143] E. Perez, S. Ringer, K. Lukosiute, K. Nguyen, E. Chen, S. Heiner, C. Pettit, C. Olsson, S. Kundu, S. Kadavath *et al.*, 'Discovering language model behaviors with model-written evaluations,' in *Findings of the association for computational linguistics: ACL 2023*, 2023, pp. 13 387–13 434 (cit. on p. 166).

[144] C. Pérez, L. Climent, G. Nicoló, A. Arbelaez and M. A. Salido, 'A hybrid metaheuristic with learning for a real supply chain scheduling problem,' *Engineering Applications of Artificial Intelligence*, vol. 126, p. 107 188, 2023 (cit. on p. 86).

[145] A. D. Pia, S. S. Dey and M. Molinaro, 'Mixed-integer quadratic programming is in np,' *Mathematical Programming*, vol. 162, no. 1, pp. 225–240, 2017 (cit. on p. 30).

[146] K. Pigmans, H. Aldewereld, V. Dignum and N. Doorn, 'The role of value deliberation to improve stakeholder participation in issues of water governance,' *Water Resources Management*, vol. 33, no. 12, pp. 4067–4085, 2019 (cit. on p. 45).

[147] K. Pigmans, N. Doorn, H. Aldewereld and V. Dignum, 'Decision-making in water governance: From conflicting interests to shared values,' in *Responsible Innovation*, Springer, 2017, pp. 165–178 (cit. on p. 45).

[148] I. van de Poel and L. Royakkers, *Ethics, Technology, and Engineering: An Introduction*. Hoboken, NJ: Wiley-Blackwell, 2011 (cit. on p. 52).

[149] A. Pommeranz, C. Detweiler, P. Wiggers and C. M. Jonker, 'Elicitation of Situated Values: Need for Tools to Help Stakeholders and Designers to Reflect and Communicate,' *Ethics and Information Technology*, vol. 14, no. 4, pp. 285–303, 2012. DOI: 10.1007/s10676-011-9282-6 (cit. on p. 52).

[150] A. Pozanco, F. Mosca, P. Zehtabi, D. Magazzeni and S. Kraus, 'Explaining preference-driven schedules: The expres framework,' in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 32, 2022, pp. 710–718 (cit. on pp. 4, 20, 152, 164).

[151] A. Rai, 'Explainable ai: From black box to glass box,' *Journal of the Academy of Marketing Science*, vol. 48, no. 1, pp. 137–141, 2020 (cit. on pp. 2, 3).

[152] N. F. Rajani, B. McCann, C. Xiong and R. Socher, 'Explain yourself! leveraging language models for commonsense reasoning,' *arXiv:1906.02361*, 2019 (cit. on p. 152).

[153] J. Rawls, *A Theory of Justice*. Oxford: Oxford University Press, 1973 (cit. on pp. 11, 51).

[154] L. Raymond and F. Bergeron, 'Project management information systems: An empirical study of their impact on project managers and project success,' *International Journal of Project Management*, vol. 26, no. 2, pp. 213–220, 2008 (cit. on p. 86).

[155] T. Rehman, R. Mandal, A. Agarwal and D. K. Sanyal, 'Hallucination reduction in long input text summarization,' in *International Conference on Security, Surveillance and Artificial Intelligence (ICSSAI-2023)*, CRC Press, 2024, pp. 307–316 (cit. on p. 152).

[156] L. Reynolds and K. McDonell, 'Prompt programming for large language models: Beyond the few-shot paradigm,' in *Extended abstracts of the 2021 CHI conference on human factors in computing systems*, 2021, pp. 1–7 (cit. on p. 167).

[157] M. O. Riedl and B. Harrison, 'Using stories to teach human values to artificial agents,' in *Proceedings of AAAI Workshop: AI, Ethics, and Society*, 2016 (cit. on p. 44).

[158] M. Rodriguez-Soto, M. Lopez-Sanchez and J. A. Rodriguez Aguilar, 'Multi-objective reinforcement learning for designing ethical environments,' in *Proceedings of International Joint Conference on Artificial Intelligence*, 2021, pp. 545–551 (cit. on p. 44).

[159] M. Rodriguez-Soto, M. Serramia, M. Lopez-Sanchez and J. A. Rodriguez-Aguilar, 'Instilling moral value alignment by means of multi-objective rein-

forcement learning,' *Ethics and Information Technology*, vol. 24, no. 1, pp. 1–17, 2022 (cit. on p. 44).

[160] L. Rokach and O. Maimon, 'Clustering methods,' in *Data mining and knowledge discovery handbook*, Springer, 2005, pp. 321–352 (cit. on p. 211).

[161] A. Rosenfeld, 'Better metrics for evaluating explainable artificial intelligence,' in *Proceedings of the 20th international conference on autonomous agents and multiagent systems*, 2021, pp. 45–50 (cit. on p. 160).

[162] M. S. Roth, *Beyond the university: Why liberal education matters*. Yale University Press, 2014 (cit. on p. 148).

[163] S. Russell, *Human Compatible: Artificial Intelligence and the Problem of Control*. Penguin, 2019 (cit. on pp. 8, 44).

[164] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal and A. Chadha, 'A systematic survey of prompt engineering in large language models: Techniques and applications,' *arXiv preprint arXiv:2402.07927*, 2024 (cit. on p. 167).

[165] F. Salas-Molina, F. Bistaffa and J. A. Rodríguez-Aguilar, 'A general approach for computing a consensus in group decision making that integrates multiple ethical principles,' *Socio-Economic Planning Sciences*, vol. 89, p. 101 694, 2023 (cit. on pp. 34, 136, 212).

[166] M. Sarihi, V. Shahhosseini and M. Banki, 'Multiskilled project management workforce assignment across multiple projects regarding competency,' *Journal of Construction Engineering and Management*, vol. 146, no. 12, 2020 (cit. on pp. 12, 39).

[167] C. C. Sartori, C. Blum and G. Ochoa, 'Stnweb: A new visualization tool for analyzing optimization algorithms,' *Software impacts*, vol. 17, p. 100 558, 2023 (cit. on p. 41).

[168] S. Scheffler, 'Valuing,' in *Equality and Tradition: Questions of Value in Moral and Political Theory*, 1st, Oxford University Press, 2012, ch. 7, p. 352 (cit. on p. 56).

[169] C. Schein, 'The Importance of Context in Moral Judgments,' *Perspectives on Psychological Science*, vol. 15, no. 2, pp. 207–215, 2020. DOI: 10.1177/1745691620904083 (cit. on p. 45).

[170] S. Schwartz, J. Cieciuch, M. Vecchione, E. Davidov, R. Fischer, C. Beierlein, A. Ramos, M. Verkasalo, J.-E. Lönnqvist, K. Demirutku *et al.*, 'Refining the theory of basic individual values,' *Journal of Personality and Social Psychology*, vol. 103, no. 4, p. 663, 2012 (cit. on p. 59).

[171] M. Serramia, M. Lopez-Sanchez and J. A. Rodriguez-Aguilar, 'A qualitative approach to composing value-aligned norm systems,' in *Proceedings of Inter-*

national Conference on Autonomous Agents and Multi-Agent Systems, 2020, pp. 1233–1241 (cit. on p. 44).

[172] M. Serramia, M. Lopez-Sanchez, J. A. Rodriguez-Aguilar, M. Rodriguez, M. Wooldridge, J. Morales and C. Ansotegui, 'Moral values in norm decision making,' in *Proceedings of International Conference on Autonomous Agents and Multi-Agent Systems*, 2018, pp. 1294–1302 (cit. on pp. 11, 45, 52, 205).

[173] E. Shao, S. Guo and Z. A. Pardos, 'Degree planning with plan-bert: Multi-semester recommendation using future courses of interest,' in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 14 920–14 929 (cit. on p. 213).

[174] M. Sharma, M. Tong, T. Korbak, D. Duvenaud, A. Askell, S. R. Bowman, N. Cheng, E. Durmus, Z. Hatfield-Dodds, S. R. Johnston *et al.*, 'Towards understanding sycophancy in language models,' *arXiv preprint arXiv:2310.13548*, 2023 (cit. on p. 166).

[175] L. C. Siebert, E. Liscio, P. K. Murukannaiah, L. Kaptein, S. L. Spruit, J. van den Hoven and C. M. Jonker, 'Estimating Value Preferences in a Hybrid Participatory System,' in *HHAI2022: Augmenting Human Intellect*, Amsterdam, the Netherlands: IOS Press, 2022, pp. 114–127 (cit. on pp. 46, 54–56, 71, 83).

[176] G. Siemens and R. S. d. Baker, 'Learning analytics and educational data mining: Towards communication and collaboration,' in *International Conference on Learning Analytics and Knowledge*, 2012, pp. 252–254 (cit. on p. 115).

[177] C. Sierra, N. Osman, P. Noriega, J. Sabater-Mir and A. Perello-Moragues, 'Value alignment: A formal approach,' in *Proceedings of Responsible Artificial Intelligence Agents Workshop*, 2019 (cit. on p. 44).

[178] J. P. M. Silva, 'Minimal unsatisfiability: Models, algorithms and applications (invited paper),' in *ISMVL*, vol. 10, 2010, pp. 9–14 (cit. on p. 42).

[179] R. Słowinski, 'Multiobjective network scheduling with efficient use of renewable and nonrenewable resources,' *European Journal of Operational Research*, vol. 7, no. 3, pp. 265–273, 1981 (cit. on p. 38).

[180] A. Sprecher, *Project Management: A Systems Approach to Planning, Scheduling, and Controlling*. Springer Science & Business Media, 2012, vol. 409 (cit. on p. 39).

[181] A. Sprecher, S. Hartmann and A. Drexl, 'An exact algorithm for project scheduling with multiple modes,' *Operations-Research-Spektrum*, vol. 19, no. 3, pp. 195–203, 1997 (cit. on pp. 22, 38).

[182] I. Stepin, J. M. Alonso, A. Catala and M. Pereira-Fariña, 'A survey of contrastive and counterfactual explanation generation methods for explainable

artificial intelligence,' *IEEE Access*, vol. 9, pp. 11 974–12 001, 2021 (cit. on p. 152).

[183] E. V. Study, *Integrated dataset, gesis data archive, version 4.0.0*, Online, 2017. DOI: https://doi.org/10.4232/1.13560 (cit. on p. 59).

[184] W. R. Swartout and J. D. Moore, 'Explanation in second generation expert systems,' in *Second generation expert systems*, Springer, 1993, pp. 543–585 (cit. on p. 3).

[185] J. Szabo, J. M. Such and N. Criado, 'Understanding the role of values and norms in practical reasoning,' in *Multi-Agent Systems and Agreement Technologies*, Springer, 2020, pp. 431–439 (cit. on p. 44).

[186] J. Szabo, J. M. Such, N. Criado and S. Modgil, 'Integrating quantitative and qualitative reasoning for value alignment,' in *European Conference on Multi-Agent Systems*, Springer, 2022, pp. 383–402 (cit. on p. 44).

[187] M. Tamiz, S. J. Mardle and D. F. Jones, 'Detecting iis in infeasible linear programmes using techniques from goal programming,' *Computers & operations research*, vol. 23, no. 2, pp. 113–119, 1996 (cit. on p. 36).

[188] Y.-S. Tsai and D. Gasevic, 'Learning analytics in higher education—challenges and policies: A review of eight learning analytics policies,' in *International Conference on Learning Analytics and Knowledge*, 2017, pp. 233–242 (cit. on p. 115).

[189] E. Tsiporkova and V. Boeva, 'Multi-step ranking of alternatives in a multi-criteria and multi-expert decision making environment,' *Information Sciences*, vol. 176, no. 18, pp. 2673–2697, 2006 (cit. on pp. 16, 48, 211).

[190] J. Van Loon, 'Irreducibly inconsistent systems of linear inequalities,' *European Journal of Operational Research*, vol. 8, no. 3, pp. 283–288, 1981 (cit. on p. 36).

[191] D. J. Walker and M. J. Craven, 'Identifying good algorithm parameters in evolutionary multi-and many-objective optimisation: A visualisation approach,' *Applied Soft Computing*, vol. 88, p. 105 902, 2020 (cit. on p. 41).

[192] C. Warren, A. P. Mcgraw and L. Van Boven, 'Values and preferences: Defining preference construction,' *Wiley Interdisciplinary Reviews: Cognitive Science*, vol. 2, no. 2, pp. 193–205, 2011 (cit. on p. 45).

[193] T. Wauters, K. Verbeeck, P. Verstraete, G. Vanden Berghe and P. De Causmaecker, 'Real-world production scheduling for the food industry: An integrated approach,' *Engineering Applications of Artificial Intelligence*, vol. 25, no. 2, pp. 222–228, 2012 (cit. on p. 1).

[194] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, 'Chain-of-thought prompting elicits reasoning in large language mod-

els,' *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022 (cit. on p. 167).

[195] J. de Wet, D. Wetzelhütter and J. Bacher, 'Revisiting the trans-situationality of values in Schwartz's Portrait Values Questionnaire,' *Quality and Quantity*, vol. 53, no. 2, pp. 685–711, 2018 (cit. on pp. 45, 52).

[196] Wikipedia, *Definition of permissive society*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Permissive_society (cit. on p. 59).

[197] T. Winkler and S. Spiekermann, 'Twenty years of value sensitive design: A review of methodological practices in vsd projects,' *Ethics and Information Technology*, vol. 23, no. 1, pp. 17–21, 2021 (cit. on p. 44).

[198] L. A. Wolsey, *Integer programming*. John Wiley & Sons, 2020 (cit. on pp. 6, 29, 31).

[199] L. A. Wolsey and G. L. Nemhauser, *Integer and combinatorial optimization*. John Wiley & Sons, 1999 (cit. on pp. 6, 30).

[200] *World Values Survey*, Online, 2021. [Online]. Available: https://www.worldvaluessurvey.org/wvs.jsp (cit. on p. 57).

[201] Y.-H. Wu and S.-D. Lin, 'A low-cost ethics shaping approach for designing reinforcement learning agents,' in *Proceedings of AAAI Conference on Artificial Intelligence*, 2018, pp. 1687–1694 (cit. on p. 44).

[202] J. Xiao, X. Wang and H. Zhang, 'Exploring the ordinal classifications of failure modes in the reliability management: An optimization-based consensus model with bounded confidences,' *Group Decision and Negotiation*, vol. 31, no. 1, pp. 49–80, 2022 (cit. on p. 48).

[203] L. Xu, F. Hutter, H. H. Hoos and K. Leyton-Brown, 'Satzilla-07: The design and analysis of an algorithm portfolio for sat,' in *Principles and Practice of Constraint Programming–CP 2007: 13th International Conference, CP 2007, Providence, RI, USA, September 23-27, 2007. Proceedings 13*, Springer, 2007, pp. 712–727 (cit. on pp. 22, 181, 193).

[204] L. Xu, F. Hutter, H. H. Hoos and K. Leyton-Brown, 'Satzilla: Portfolio-based algorithm selection for sat,' *Journal of artificial intelligence research*, vol. 32, pp. 565–606, 2008 (cit. on pp. 181, 193).

[205] Z. Xu and X. Cai, 'Group consensus algorithms based on preference relations,' *Information Sciences*, vol. 181, no. 1, pp. 150–162, 2011 (cit. on p. 48).

[206] L. R. Ye and P. E. Johnson, 'The impact of explanation facilities on user acceptance of expert systems advice,' *Mis Quarterly*, pp. 157–172, 1995 (cit. on p. 4).

[207] I. B. Ji-Ye Mao, 'The use of explanations in knowledge-based systems: Cognitive perspectives and a process-tracing analysis,' *Journal of Management Information Systems*, vol. 17, no. 2, pp. 153–179, 2000 (cit. on p. 4).

[208] P.-L. Yu, 'A class of solutions for group decision problems,' *Management science*, vol. 19, no. 8, pp. 936–946, 1973 (cit. on p. 48).

[209] P.-L. Yu, *Multiple-criteria decision making: concepts, techniques, and extensions*. Springer Science & Business Media, 2013, vol. 30 (cit. on p. 48).

[210] M. Zečević, D. S. Dhami, C. A. Rothkopf and K. Kersting, 'Causal explanations of structural causal models,' *arXiv preprint arXiv:2110.02395*, 2021 (cit. on p. 157).

[211] P. Zehtabi, A. Pozanco, A. Bolch, D. Borrajo and S. Kraus, 'Contrastive explanations of centralized multi-agent optimization solutions,' in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 34, 2024, pp. 671–679 (cit. on pp. 4, 152, 156, 160, 201).

[212] Y. Zhang, X. Chen *et al.*, 'Explainable recommendation: A survey and new perspectives,' *Foundations and Trends® in Information Retrieval*, vol. 14, no. 1, pp. 1–101, 2020 (cit. on p. 40).

[213] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, 'A survey of large language models,' *arXiv preprint arXiv:2303.18223*, vol. 1, no. 2, 2023 (cit. on p. 166).

[214] Z. Zhao, E. Wallace, S. Feng, D. Klein and S. Singh, 'Calibrate before use: Improving few-shot performance of language models,' in *International conference on machine learning*, PMLR, 2021, pp. 12 697–12 706 (cit. on p. 167).

[215] Q. Zhou and D. Huang, 'Towards generating math word problems from equations and topics,' in *Proceedings of the 12th international conference on natural language generation*, 2019, pp. 494–503 (cit. on p. 214).