Managing quality in agent dialogues

Mariela Morveli-Espinoza and Josep Puyol-Gruart

Artificial Intelligence Research Institute (IIIA)
Spanish Scientific Research Council (CSIC)
Campus UAB. 08193 Bellaterra. Spain.
{mmorvelie,puyol}@iiia.csic.es
http://www.iiia.csic.es

Abstract. This article presents quality measures to rate responses given by agents, in order to drive the control of communication and dialogues among agents. Depending on the available time or the urgency of the requirement, agents can produce answers with different levels of quality. Assuming that in real world is normally better to receive an answer with poor quality than no answer, agents are able to give partial answers based on its current knowledge. Autonomy implies taking the best decision with the available information, obtained from perception or from the communication with other agents, avoiding blocking situations and no action. Agents use specialization calculus as inference engine to generate such kind of answers.

Introduction

In multiagent systems, agents have particular goals. Conversations among deliberative agents aim to obtain information in order to produce solutions to those goals. Consider a rule-based agent. The classical inference mechanisms could take a long time to generate definitive results, depending on the availability of external information—from the environment or from other agents. A promising approach is the specialization reasoning model described below, because it can generate partial results or answers in a much shorter period of time.

It may be reasonable to think about different strategies of reasoning using partial information, i.e. when a concrete timeout has been reached or when a value is needed, a less precise but useful answer could be used. Improved accuracy or quality of results can be attained over time, and then the agent goals could persist until it is no possible to obtain more precise values.

Time is another important issue, to deal with this aspect we consider *anytime algorithms*. They were first used by Dean and Boddy in the late 1980's [2]. The main characteristic of these algorithms is that the *quality* of its results can be measured and that it improves gradually as computation time increases. This kind of algorithms are normally related to real time, where time granularity is thinner than the long time needed to calculate a complete solution. They are able to communicate the best result obtained when interrupted or they can establish a compromise to deliver it in a given time. In our approach both absolute and relative *quality* will be introduced as a quality measure of agents answers.

In the context of logics and knowledge-based systems some authors talks about *progressive (or anytime) reasoning* or *deduction* [7, 6]. Anytime concepts are important for this techniques to build intelligent systems, for instance in probabilistic reasoning, ontologies or constrain propagation [3, 13, 15].

In this paper we will introduce how reasoning based on specialization of rule-based systems can be the central mechanism to deliberate and also to produce *reasonable* dialogues among conversational agents [1, 4, 11]. We consider that agents are anytime reasoners producing answers with different levels of quality [5]. We assume that in the real world normally is better to receive an answer with poor quality than no answer. The answer can be good enough for the receiver or the receiver can spend more time to wait for a better answer.

Section 2 is devoted to quality measures. In Section 3 we formally describe the specialization as an anytime mechanism and its impact over quality and precision. We present the description of the agent and its pragmatics in Section 4. Comments on performance and validation of our approach are presented in Section 5. Finally, some conclusions and future work are developed in Section 6.

1 Preliminaries

In this section we will define the language used and the core of our approach: specialization calculus [8–10].

Language $\mathcal{L} = \langle V, \Sigma, \mathcal{S} \rangle$ is defined by:

- Truth-values $V \in [0, 1]$ where 1 and 0 are the booleans *True* and *False* respectively. $Int(V) = \{[i, j] \mid i, j \in V, i \leq j\}$ are intervals of truth-values in V.
- Σ is a set of propositional variables (atoms or facts).
- Sentences S composed by: literals (a, I), $(\neg a, I)$, with $a \in \Sigma$ and $I \in Int(V)$ and rules of the form $(p_1 \wedge p_2 \wedge \cdots \wedge p_n \rightarrow q, [i, 1])$, where $i \in V$, p_i and q are literals, and $\forall i, j (p_i \neq p_j, p_i \neq \neg p_j, q \neq p_j, q \neq \neg p_j)$.

Inference Rules We will use the following inference rules:

- Not-introduction: from $(a, [i, j])$ infer $(\neg a, [1 - j, 1 - i])$
- Not-elimination: from $(\neg a, [i, j])$ infer $(a, [1 - j, 1 - i])$
- Parallel composition[1]: from (a, I_1) and (a, I_2) infer $(a, I_1 \cap I_2)$
- Specialization: from $(p_i, [i, j])$ and
  $(p_1 \wedge \cdots \wedge p_{i-1} \wedge p_i \wedge p_{i+1} \wedge \cdots \wedge p_n \rightarrow q, [k, 1])$
  infer $(p_1 \wedge \cdots \wedge p_{i-1} \wedge p_{i+1} \wedge \cdots \wedge p_n \rightarrow q, [T(i, k), 1])$

The conjunction operator T is a binary operation such that the following properties hold $\forall a, b, c \in V_n$:

- T1: $T(a, b) = T(b, a)$

[1] When the intersection is empty, then it is considered to be a contradiction in the knowledge base.

- T2: $T(a, T(b, c)) = T(T(a, b), c)$
- T3: $T(0, a) = 0$
- T4: $T(1, a) = a$
- T5: if $a \leq b$ then $T(a, c) \leq T(b, c)$ for all c

We can use general triangular norms like:

- Łukasiewicz $T_L(x, y) = max\{0, x + y - 1\}$,
- Product $T_P(x, y) = (x \cdot y)$, or
- Minimum $T_m(x, y) = min(x, y)$

We can see that in this calculus both rules and facts are weighted using intervals of truth-values, and that both conditions and conclusions of rules can be negated. The *specialization rule* defined above is the core of the progressive reasoning algorithm. When a rule is specialized it produces a new rule with less conditions and a new updated truth-value. When a rule is totally specialized (there are no more conditions) it produces a final value for the literal of the conclusion.

Consider a fact q with truth-value $[\alpha, \beta]$ and the set of rules $R_q = R_q^+ \cup R_q^-$ deducing it. We can distinguish between the rules R_q^+ deducing q (positive ones) and R_q^- deducing $\neg q$ (negative ones). Following the specialization inference rule, positive rules (positive evidences) will contribute to the minimum of the interval α and negative ones (negative evidences) to the maximum β.

From T4 and T5 we can deduce the property $T(a, b) \leq min\{a, b\}$. Given a rule $(P \rightarrow q, [k, 1]) \in R_q^+$, we can obtain values $(q, [\rho, 1])$, with $\rho \leq k$. The most precise value $(q, [k, 1])$ for the conclusion will be obtained when P is true, or $[0, 1]$, because of the specialization rule and the T_4 property of T. Similarly, given $(P \rightarrow \neg q, [k, 1]) \in R_q^-$, the most precise value will be $(\neg q, [k, 1])$ and then for the positive literal $(q, [0, 1 - k])$.

Fact q is initially *unknown*, that is, its value is the most imprecise interval $[0, 1]$. Using the values obtained from totally—positive and negative—specialized rules we will obtain a more precise interval for q by means of the applications of *parallel composition rule.*

Most precise value Given an agent A with a set of rules $R_q = R_q^+ \cup R_q^-$ deducing q: r rules R_q^+ with truth-values $\{[\alpha_1, 1], \ldots, [\alpha_r, 1]\}$ and s rules R_q^- with truth-values $\{[\beta_1, 1], \ldots, [\beta_s, 1]\}$. The most precise interval from R_q^+ will be $[max_{i=1}^{r}(\alpha_i), 1]$ and $[0, min_{i=1}^{s}(\beta_i)]$ from R_q^-.
Finally we can say that the expected most precise interval for q from the knowledge R_q of agent A will be

$$[max_{i=1}^{r}(\alpha_i), min_{i=1}^{s}(\beta_i)]$$

We have to take into account that each specialization step produces a new knowledge base and then the expected most precise interval will be changed.

The new rules are provisional if they are deduced with provisional information otherwise they are definitive. Facts are definitive if they are deduced with definitive information and there are no more rules that can improve its value. Depending on this, rules can be deleted or not, see Section 4.3.

2 Quality measures

Quality measures and their properties are important for anytime algorithms [16]. Quality has to be (i) Measurable and recognizable: the quality of an approximate result has to be determined precisely and easily at run time, (ii) Monotonic: the quality of the result is a non-decreasing function of time and input quality, and (iii) Consistent: the quality of the result is correlated with time and input quality.

Quality is evaluated based on a three-dimensional criterion that measures the level of certainty, precision and completeness of a given value. In our case, these three measures will be used to determine the quality of the answers given by agents, which values are intervals of thuthvalues.

The quality is determined based on the following characteristics:

Certainty: In an approximate reasoning context we want to know the degree of truth or falsity of propositions. Then, given a set of knowledge deducing a fact we are interested in using those relations that provides values close to *true* or *false*. We assume a uniform distribution of the certainty in the interval, the mean is then the expected value of the interval and it can be representative of its certainty:

$$C[i,j] = \frac{i+j}{2}$$

Precision: Values of facts are intervals. The most precise information given by the interval is when the difference between the maximum and the minimum is 0, and the least precise is when that difference is 1, that is, the only case $[0,1]$, or *unknown*.

$$P[i,j] = 1-(j-i)$$

Completeness: To determine the value of a fact we need to know the values of other related facts contained in the rules that deduce that fact. Given two facts, with the same level of certainty and precision, we will consider of more quality that with less number of dependencies that could improve the result.

We have to distinguish between the quality of a certainty value in itself and that related to the quality given by a concrete knowledge base (KB). For instance, considering a fact q with truth-value 1, it is obvious that this value is better than any other value less than 1. However, if the KB deducing fact q can not produce a better value than 0.8—there is no rule with truth-value greater than 0.8—this value is the maximum that the fact q can reach and then it is considered of the best quality.

Taking into account the comments above we are going to define both absolute and relative qualities. Absolute quality is used as an internal quality for helping the agent to explore first the more promising rules. To know which is the next rule to be used, we consider the absolute quality of the truth-values of the rules. Both certainty and precision are used to calculate it. On the other hand, relative quality is calculated using precision formula and it also uses completeness as a complementary measure for that cases when more than one answer has the same precision value. We use relative quality as a measure of the quality level of answers given by agents.

2.1 Absolute quality

Precision and certainty are directly related because a good precision is more interesting when the value of the fact is close to true or false. We can use the following expression to calculate it, resulting values are between 0 and 1:

$$f(p, c) = p \cdot |(2c - 1)|$$

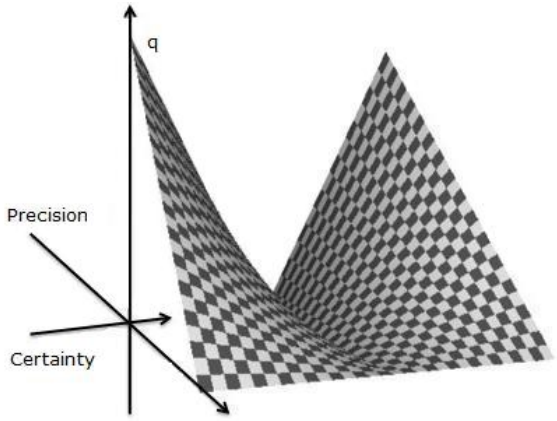

Fig. 1. Representation of the quality function where $x, y, z \in [0, 1]$ are the middle point of the interval (certainty), the amplitude of the interval (precision) and the quality, respectively. It is a symmetric function with respect to the plane $x = 0.5$

The first term p corresponds to the precision of the interval, better when closer to 1. The second term c corresponds to the value represented by the middle point of the interval, better when closer to 0 or 1, that is, true or false. It is a symmetric function with respect to the plane $i + j = 1$. In Figure 1 we can see the shape of the function. Finally, the absolute quality for an interval is:

$$Q_a[i, j] = |P([i, j]) \cdot (2C([i, j]) - 1)| = \left|i^2 - (1 - j)^2\right|$$

2.2 Relative quality

The KB of an agent determines the maximum relative quality that can be obtained for a given fact taking account all the available information. We can consider a KB is *good* deducing a fact f when it can deduce that fact with high precision and with values close to true or false. It depends on the programmer of the KB because all rules and facts of the starting KB are in charge of the system designer. For instance, consider a KB_1 where the maximum expected truth-value for rules deducing f is 0.7. This means that the highest value f can

reach is 0.7. This value is obviously less than 1, but for KB_1 is the maximum possible. On the other hand, the new values that an agent acquires over the time can also modify the truth-values of rules. Therefore, we have to consider relative measures with respect to the maximum quality level that is possible to obtain with the current knowledge. Finally, the result for f will be a combination of the KB and the values of facts used to make the deduction.

In the next Section we will see how this value change with (discrete) time, when new information is acquired by the agent. After each execution cycle—to include the new information—quality of the results can be evaluated. We can know in each moment which is the *most precise value* agent A can obtain for the fact a, $P_m^A(a)$, as explained in Section 1. We will use precision to evaluate the relative quality.

$$Q_r^A(a, [\alpha, \beta]) = \frac{P[\alpha, \beta]}{P_m^A(a)}$$

If all the facts used to deduce the goal would have a definitive value then the completeness will be of 100%. If all those facts would also have values—true of false with the maximum precision—such that the premises of rules are true then we will obtain the maximum quality degree.

3 Deduction and quality

The main component of the mental state of agents is the KB. It contains beliefs (facts) and knowledge (rules) for deliberation [14]. In our model, both facts and rules are weighted with intervals of truth-values.

Specialization can be considered as an anytime algorithm because it allows to obtain information before the completion of the inference process. It can be considered also a mechanism for progressive reasoning because it is a technique that successively refines a solution while making available intermediate solutions. In the following we will show the relation between specialization and both absolute and relative quality.

3.1 Specialization and absolute quality

The main goal of the inference engine is try to find the most precise values for the facts. The nature of the specialization inference, described above, avoids producing less precise values for rules after a specialization step. However, this is not so obvious for the quality of facts. Following parallel composition, we can see that this quality is not a monotonic function with respect to the time. For instance, an agent A can produce for the fact a the provisional value $[0.4, 1]$ with quality $Q_a^A[0.4, 1] = 0.16$. Consider that a new information $[0, 0.6]$ arrives and $Q_a^A([0.4, 1] \cap [0, 0.6]) = Q_a^A[0.4, 0.6] = 0$.

Consider a provisional result $[\alpha, \beta]$. We will analyze the behavior of the specialization with respect to quality:

Positive rules They produce results in the form $[\gamma, 1]$, and the middle point is $C[\gamma, 1] \geq 0.5$. We have to consider the following cases:

- When $C[\alpha,\beta] \geq 0.5$, $\forall\gamma, Q([\alpha,\beta] \cap [\gamma,1]) \geq Q[\alpha,\beta]$
- When $C[\alpha,\beta] \leq 0.5$, and $\beta \geq 0.5$, $Q([\alpha,\beta] \cap [\gamma,1]) \geq Q[\alpha,\beta]$ only for $\sqrt{2(1-\beta)^2 - \alpha^2} \leq \gamma \leq \beta$
- Otherwise $Q([\alpha,\beta] \cap [\gamma,1]) < Q[\alpha,\beta]$

Negative rules They produce results in the form $[0,\delta]$, and the middle point is $C[0,\delta] \leq 0.5$. We have to consider the following cases:

- When $C[\alpha,\beta] \leq 0.5$, $\forall\delta, Q([\alpha,\beta] \cap [0,\delta]) \geq Q[\alpha,\beta]$
- When $C[\alpha,\beta] \geq 0.5$ and $\alpha \leq 0.5$, $Q([\alpha,\beta] \cap [0,\delta]) \geq Q[\alpha,\beta]$ only for $\alpha \leq \delta \leq 1 - \sqrt{2\alpha^2 - (1-\beta)^2}$
- Otherwise $Q([\alpha,\beta] \cap [0,\delta]) < Q[\alpha,\beta]$

These results are obvious in the sense that an interval with a middle point greater that 0.5 is reinforced by positive rules producing a more precise result more close to 1. The same occurs with intervals close to false and negative rules. In the other cases when we combine two intervals with middle points in opposite parts—greater and less than 0.5—it is necessary to compensate with the precision the fact that the middle point of the resulting value now is farther from true or false.

At the meta-level we will be interested in exploring the best rules first. It seems reasonable to try first the rules that can contribute increasing the absolute quality of facts. Notice that the values of rules change in each specialization step and that it can not be guaranteed a monotonic behavior of absolute quality.

3.2 Specialization and relative quality

It is easy to see that precision of facts is a monotonic function with respect to the time, new information will produce more precise facts by firing rules and applying parallel composition with old values of facts

$$P([\alpha,\beta] \cap [\gamma,\delta]) \geq max\{P[\alpha,\beta], P[\gamma,\delta]\}$$

Taking into account rules we can notice that their precision decreases with new information because the specialization rule and the well-known property of t-norms: $T(a,b) \leq min\{a,b\}$

$$\forall\alpha \leq k, P([\alpha,1]) \leq P([k,1])$$

We can conclude that new information will produce less or equal precision for the set of rules and equal or more precision for the set of facts. This implied that we can use provisional values of facts to deduce more provisional values, and no belief revision is needed. We can use relative quality as the external quality of an agent that deals with the more important property of an anytime algorithm, monotonicity.

4 Deliberative agents and anytime reasoning

The model of reasoning described above could take a long time to generate definitive results. This is not a consequence of the complexity of the deductive process. We consider that specialization time is irrelevant for our time restrictions, which are communication time, availability of agents, collaborative behavior, etc. The time granularity depends on the application but we have to take into account that the motivation is not classical real time.

Below an agent definition as anytime entity is presented. It is also showed how agents work internally and how time impacts on its behavior.

4.1 Agents as anytime entities

Consider a multi-agent system with m agents $\mathcal{A}_m = \{A_1, \ldots, A_m\}$. Each agent has the following structure:

Agents A deliberative agent is a tuple $A_i = \langle KB_i, G_i, I_i, O_i, t_i \rangle$ where:

- KB_i is the knowledge base of agent A_i.
- G_i is the set of goals of A_i. A goal g is a tuple $\langle x, A_j, t_b \rangle$, where $x \in \Sigma$, $A_j \in \mathcal{A}$ is the agent that queries A_i about x and t_b is the remaining time for deadline.
- I_i is the input interface of A_i, the set of external facts that can be obtained by querying other agents. These are tuples $\langle x, A_j \rangle$, where $x \in \Sigma$, $A_j \in \mathcal{A}$ and $A_j \neq A_i$.
- O_i is the output interface of A_i; this is, the set of facts agent A_i can answer to other agents.
- t_i is the deadline for giving an answer.

There are two types of anytime algorithms [12]: the *interruptible* one may be halted at any time and produces a result with a more or less good quality and the *contract* one has a contract time—it must know the total allocation of time in advance—if interrupted at any point before the termination of the contract time, it might not produce results. We can consider that our agents could have both anytime behaviors. A contract algorithm because the deadline is known in advance—autonomy gives agents freedom to define its own deadline independently of other agent's deadlines. It also could have an interruptible behavior because it can be asked at any time giving the current value, in the worst case *unknown*.

An agent has a set of processes: (i) an *interface communication manager*, (ii) an *specialization engine*, (iii) an *answering machine*, (iv) an *evaluation machine* and (v) an *integration machine*; and a set of data repositories: the KB are the facts and rules of the problem domain, the current commitments (*goals*), and the data about other agents (*acquaintances*) and data about its abilities and capacities (*competences*). In Figure 2 you can see an scheme of the relations among all these components.

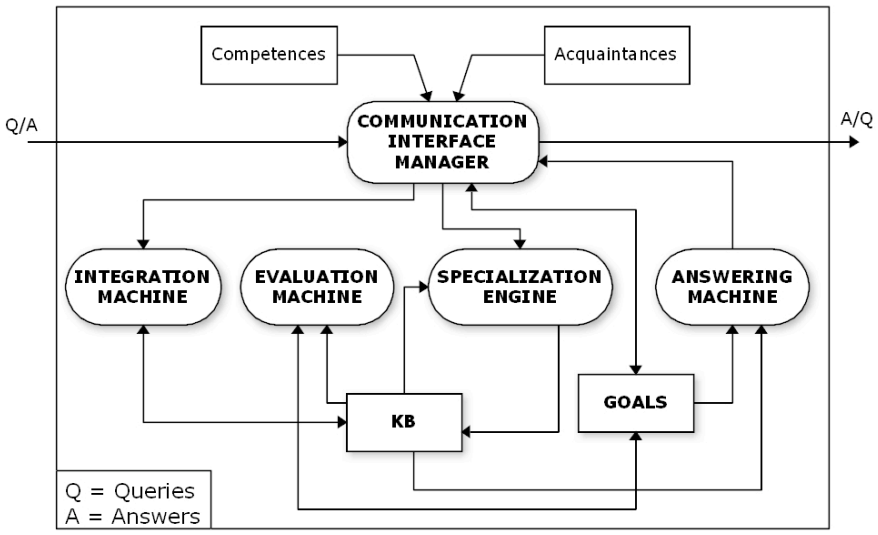

Fig. 2. Agent architecture.

4.2 Agent architecture

- *Communication Interface manager (CIM)* manages the input and output of queries and answers:
  - When it receives a query q and $q \in O_i$, a new goal is added to the goal list: $G_i := G_i \cup \{\langle q, A_j, t_f \rangle\}$.
  - When it receives an answer, it sends it to the *integration machine*.
  - It sends the answers and queries to the other agents, following the correct protocols and reporting all the activity.
- The *specialization engine* receives as inputs fact values and performs a *specialization cycle*: $S : KB \times f \rightarrow KB'$ is a data-driven process that begins when the input is a new fact value f. This triggers a complete specialization process over the KB and a new specialized KB' is generated.
- The *integration machine* receives as input a complete answer (facts and eventually a set of rules) and incorporates them into the KB.
- The *answering machine* receives as input a trigger signal indicating:
  - i A goal deadline ends. If the goal doesn't have a definitive value, then the *answering machine* has to elaborate other kinds of answers (see Section 4.3)
  - ii The definitive value for a goal is found, and then the obvious response is the definitive value.
- The *evaluation machine* is a goal-driven process $I : KB \times g \rightarrow g^*$ that begins when the agent process a goal g. It triggers a complete exploring process obtaining a set of new goals g^*, which are necessary to find values of g with better quality, as seen in Section 3.1.

4.3 Responses

One important topic in our model is the different variety of answers agents can give. Below, the definition of agent responses and the kinds of responses are defined:

Responses A response is a tuple $R = \langle f, V, S, KB \rangle$ where:

- f is the fact which is been answered.
- V is the value of fact f (an interval of truth-values).
- S is the state of the fact f value, i.e. *provisional*, *definitive* or *pending*[2].
- KB is a knowledge base useful to improve the value of f.

1. Definitive value $R = \langle f, V, def, \emptyset \rangle$: this is the most useful result because it means that there is no more information that can improve the result, this is the most precise. After the specialization agents can substitute a rule using it by its specialized version.
2. Provisional value $R = \langle f, V, prov, \emptyset \rangle$: this is not a definitive value, it can be improved later. Agents can use it to produce only more provisional values. Agents can not delete rules that use it because they will be useful to produce more precise values.
3. Provisional value and a set of knowledge related to it, $R = \langle f, V, prov, KB_f \rangle$: this is similar to the case above but the answer includes all the information needed for improving the value. Agents can use this provisional value and start the mechanism to find more information.
4. A set of rules related to the question $R = \langle f, [0, 1], pending, KB_f \rangle$: the same that the case above but without a provisional value.

Since we consider that agents have a deadline to answer a question. When it is not possible to obtain a definitive value for a query and the deadline has been reached, agents can answers with less quality answers as provisional or pending. Provisional values are less precise and can be used also to produce provisional deduction and so provisional values for other facts.

In summary, answers can contain the best value, a provisional one because it can be improved later, or a conditional answer because the agent ignores some information needed to build the answer.

4.4 Evaluation cycle

When an *agent's life* begins and receives a simple query, the agent begins a goal-driven—backward chaining style—work. This task will produce new goals that have to be solved. The *evaluation machine* judges the impact of these new goals in the quality of the original one. Some of them can be internal and others have to be obtained from other agents. Internal goals are considered a self-commitment and the agent starts a search process in order to find which are the new goals it needs.

[2] A pending fact is a fact that is provisionally unknown [4].

When new facts are known—maybe from other agents answers—it is started a data-driven task of specialization—forward chaining style. The transition from one solution to a more precise one happens in this specialization step.

An incomplete answer to a query is generated when there is no enough time to complete the query processing or there are agents that do not answer. Each agent goal could achieve a definitive or a provisional value. The *evaluation machine* decides if this value is enough. If further reasoning is required to improve the quality, new requirements are sent to the corresponding agents.

Agents can send and receive facts and rules as conditional answers or knowledge communication. When the deadline of a goal ends and it has a provisional value, the agent can send rules as part of the answer (see Section 4.3). There are sets of criteria that are out of the scope of this paper like privacy or protocol constraints that can limit the contents of rules in an answer. It is not necessary to send the provisional rules because with the provisional values of facts and the original rules we can easily deduce the provisional ones.

5 On performance and validation

Performance profiles are used to describe the expected output quality value as a function $Q(t)$ with execution time t [16]. Normally they have to be calculated using statistics over a set of inputs and they are normally monotonic functions. In our case it is easy to see that given the quality measures in Section 2 and the analysis in Section 3 the performance profile of an agent is a monotonically non-decreasing function. New information improves the completeness and precision of the results.

There are other forms of performance profiles. For instance the conditional performance profile, where $P(q_{out}|q_{in}, t)$ is the probability of obtaining a result with quality q_{out}, given an input of quality q_{in} at time t. It is desirable that when the input quality improves the output quality also will do. From the specialization and parallel compositions rules we can see that it is true.

It is obvious that an anytime system can produce answers before a standard one. The problem is to determine when this is or not an advantage. An agent can make a decision or execute an action depending on information of different quality. Depending on the available time or the urgency, the agent can accept a level of quality enough to exceed a particular threshold and then take action. Autonomy implies taking the best decision with the available information, avoiding blocking situations and no action.

Performance profiles can be objective measures of validation, but we are also interested in experimenting on subjective aspects of agent's behavior as the *emergence* of conversations among agents. We are interested in seeing if a very simple rule-based mechanism—as presented above—can produce conversations *similar to humans*.

6 Conclusions and future work

In this paper we have presented an anytime mechanism for deliberative agents based on a monotonous reasoning over intervals of truth values. Both absolute and relative quality measures have been defined. After the respective analysis, relative measures fit better with anytime quality measures properties.

Deadline is considered fix for the sake of simplicity, but it could be variable and be calculated to improve agent's performance. Criteria like communication channels cost, confidence and agent's capacity can be considered for its estimation and effects on performance profiles. We have said that when an agent receives a provisional value it can be used to produce more provisional values, but we can think in a timeout or other rational subjective criteria to consider that a provisional value becomes definitive.

We have to study other type of performance profiles taking into account that our system is a multi-agents system. We can think in social or join performance profiles and in the dependency of performance profile measures with respect to the deadline time of agents and other parameters.

We are also designing a protocol to deal with provisional values and the knowledge received. It is reasonable to think that when a provisional value is received, agents can insist later in order to improve the value or use their own means to obtain that information.

Acknowledgements.

Authors acknowledge partial support by the Spanish projects IEA (TIN2006-15662-C02-01), MULOG2 (TIN2007-68005-C04-01) and Agreement Technologies (CONSOLIDER CSD2007-0022, INGENIO 2010). Mariela Morveli-Espinoza is supported by the Programme Alβan, the European Union Programme of High Level Scholarships for Latin America, scholarship No.(E06D101440PE). We would like to thank the referees for their valuable suggestions and comments.

References

1. M Barbuceanu and WK Lo. Conversation oriented programming for agent interaction. In *Issues in Agent Communication*, volume 1916 of *Lecture Notes in Artificial Intelligence*, pages 220–234, 2000.
2. T. L. Dean and M. Boddy. An analysis of time-dependent planning. In *Proceedings of the Seventh National Conference on Artificial Intelligence, AAAI 88*, pages 49—54, 1988.
3. B. Jaumard and A. D. Parreira. An anytime deduction algorithm for the probabilistic logic and entailment problems. *International Journal of Approximate Reasoning (IJAR)*, 50(1):92–103, 2009.
4. Mariela Morveli-Espinoza and Josep Puyol-Gruart. On partial deduction and conversational agents. In Teresa Alsinet, Josep Puyol-Gruart, and Carme Torras, editors, *Artificial Intelligence Research and Development*, volume 184 of *Frontiers in Artificial Intelligence and Applications*, pages 60–69. IOS Press, 2008.

5. Mariela Morveli-Espinoza and Josep Puyol-Gruart. Anytime reasoning mechanism for conversational agents. In Sandra Sandri, Miquel Sànchez-Marrè, and Ulises Cortés, editors, *Artificial Intelligence Research and Development*, volume 202 of *Frontiers in Artificial Intelligence and Applications*, pages 215–223. IOS Press, 2009.
6. Abdel-Illah Mouaddib. A study of a dynamic progressive reasoning system. *Journal of Experimental and Theoretical Artificial Intelligence*, pages 101–122, 2000.
7. Abdel-Illah Mouaddibllah Mouaddib and Shlomo Zilberstein. Knowledge-based anytime computation. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95*, pages 775–783, 1995.
8. J. Puyol, L. Godo, and C. Sierra. A specialisation calculus to improve expert system communication. In Bern Neumann, editor, *Proceedings of the 10th European Conference on Artificial Intelligence, ECAI'92*, pages 144–148, Vienna, August 1992. Jonh Wiley & Sons, New York.
9. J. Puyol-Gruart, L. Godo, and C. Sierra. Specialisation calculus and communication. *International Journal of Approximate Reasoning (IJAR)*, 18(1/2):107–130, 1998.
10. J. Puyol-Gruart and C. Sierra. Milord II: a language description. *Mathware and Soft Computing*, 4(3):299–338, 1997.
11. F Rago. Conversational agent model in intelligent user interface. In *Fuzzy Logic and Applications*, volume 2955 of *Lecture Notes in Artificial Intelligence*, pages 46–54, 2006.
12. S. J. Russell and S. Zilberstein. Composing real-time systems. In *Proceedings of the Twelfth International Joint Conferences on Artificial Intelligence*, pages 212—217, 1991.
13. S. Schlobach, E. Blaauw, M. El Kebir, A. ten Teije, F. van Harmelen, S. Bortoli, M. Hobbelman, K. Millian, S. Stam Y. Ren, P. Thomassen, R. van het Schip, and W. van Willigem. Anytime classification by ontology approximation. In Ruzica Piskac et al., editor, *Proceedings of the workshop on new forms of reasoning for the Semantic Web: scalable, tolerant and dynamic*, pages 60–74, 2007.
14. Yoav Shoham. Agent-oriented programming. *Artificial Intelligence*, 60:51–92, 1993.
15. Alan Verberne, Frank Van Harmelen, and Annette Ten Teije. Anytime diagnostic reasoning using approximate boolean constraint propagation. In *Constraint Propagation, Int. Conference on Principles of Knowledge Representation and Reasoning (KR'00)*, pages 323–332, 2000.
16. Shlomo Zilberstein. Using anytime algorithms in intelligent systems. *The AI magazine*, 17(3):73–83, 1996.