Introducing Relevance Awareness in BDI Agents

Emiliano Lorini, Michele Piunti

Introducing Relevance Awareness in BDI Agents

Emiliano Lorini[1] and Michele Piunti[2]

[1] Université de Toulouse, CNRS, Institut de Recherche en Informatique de Toulouse, France
[2] Università degli studi di Bologna - DEIS, Bologna, Italy

Abstract. Artificial agents engaged in real world applications require accurate allocation strategies in order to better balance the use of their bounded resources. In particular, during their epistemic activities, they should be able to filter out all irrelevant information and just consider what is relevant for the current task that they are trying to solve. The aim of this work is to propose a mechanism of relevance-based belief update to be implemented in a BDI cognitive agent. This is in order to improve the performance of agents in information-rich environments. In the first part of the paper we present the formal and abstract model of the mechanism. In the second part we present its implementation in the *Jason* programming platform and we discuss its performance in simulation trials.

1 Introduction

Realistic cognitive agents are by definition resource-bounded [6], hence they should not waste time and energy in reasoning, fixing and reconsidering their knowledge on the basis of every piece of information they get. For this reason, they require accurate allocation strategies in order to better balance the use of their bounded computational resources. In this paper we present a computational model of a mechanism of relevance-based belief update. This mechanism is responsible for filtering out all non-relevant information and for considering only what is *relevant* for the current task that an agent is trying to solve. We show how such a mechanism can be implemented in a BDI (Belief, Desire, Intention) agent [22]. BDI is a well-established framework which is aimed at describing an agent's mental process of deciding, moment by moment on the basis of current beliefs, which action to perform in order to achieve his goals. The mechanism we propose will accomplish the following general function in an agent reasoning process: (*i*) to signal the inconsistency between the agent's beliefs and an incoming input which is relevant with respect to the agent's current intentions and (*ii*) to trigger a process of belief update in order to integrate such a relevant input in the agent's belief base. More generally, we suppose that at each moment an agent is focused and allocates his attentive resources on a particular task that he is trying to fulfill and on a certain number of intentions which represent the pragmatic solution selected by the agent to accomplish the task [3]. In so doing, the agent ignores all incoming inputs which are not relevant with respect to the current task on which he is focused and only considers the information that is relevant. If a relevant input turns out to be incompatible with respect to the pre-existent beliefs of the agent, the agent reconsiders them.

The approach proposed in this paper is also intended to bridge the existing gap between formal and computational models of belief change and cognitive models of belief

dynamics. Indeed, formal approaches to belief change implicitly assume that when an agent perceives some fact such a perception is always a precursor of a process of belief change. In contrast, we model here these precursors of a process of belief change and, in agreement with cognitive theories of bounded rationality (e.g. [24]), we show that implementing them in a resource-bounded agent can improve his performance in information-rich environments requiring massive perceptive activities.

This proposal is also intended to provide a novel insight in the design and programming of cognitive agents. Whereas in the context of Multi Agent Systems (MAS) mainstream agent platforms provide advanced mechanisms to process messages and Agent Communication Languages (ACLs), the perception of heterogeneous events and information is often shaped on the basis of technical constructs, which are typically designed in a domain-dependent fashion and implemented with constructs built at the language level. Mainstream programming models are often not flexible enough to integrate different perceptive abilities, nor to relate them with cognitive constructs such as the cognitive ones typical of BDI agents. Instead, we propose in this paper a programming model in which the relationship between perception and practical reasoning is clearly specified in terms of the pivotal notion of pragmatic relevance.

The paper is organized as follows. Section 2 abstractly describes the proposed approach and contextualizes it with respect to the related literature on the subject of relevance. Section 3 defines the abstract model of a cognitive agent. This includes informational attitudes (e.g. beliefs which change over time and stable causal knowledge) and motivational attitudes (e.g. intentions and desires). Section 4 applies the agent's abstract model to the formalization of a specific problem domain. In section 5 the cognitive architectures of two general typologies of BDI agents are formally defined—respectively implementing a traditional BDI interpreter and BDI^{rel}with relevance awareness abilities. Section 6 describes a programming model for the BDI^{rel}agent, discussing how it has been implemented by using the *Jason* platform [2]. Finally, Section 7 compares the performance of agents engaged in simulated experiments in the scenario previously described and Section 8 concludes with final remarks and future directions of research.

2 The Concept of Relevance: An Overview

The notion of information relevance is not new in the literature, as it has been extensively investigated in several domains like AI, philosophy and cognitive sciences.

Most theoretical works on relevance have been interested in modeling a particular form of *informational* relevance based on various forms of conditional in/dependence. According to [13,18], for instance, the concept of relevance coincides with the probabilistic concept of conditional dependence and, in particular, irrelevance is identified with conditional independence, and relevance is identified with the negation of irrelevance. Relevance logic [1] proposes alternatives to material implication where the antecedent and consequent are relevantly related.

In the area of belief revision, some authors [7,19] have introduced a primitive notion of relevance of an agent's belief base with respect to an incoming input χ. These authors argue that during belief revision a rational agent does not change his entire belief corpus, but only the portion of it that is relevant to the new incoming information χ, that is,

only the portion that shares common propositional variables with the minimal language of the input χ. Some computational systems, inspired by Information Theory [23], conceive relevance as a quantitative notion of informativeness that can be related to a given datum. Among others, such an approach gave rise to ranking mechanisms used for instance by web research engines, which are almost always based on a quantitative and possibly weighted analysis of the amount of links referable to a web document.

Differently from the above mentioned works, which are mostly interested in a notion of informational relevance, we are interested here in investigating a notion of *pragmatic relevance*[1] and in its application to agent programming. The notion of relevance discussed in this paper is closer to the one considered in several psychological theories of motivations and emotions [14,9], where relevance is related to the subjective appraisal of a certain event with respect to an agent's ongoing goals and intentions. In particular, according to these theories, an agent's perception of a fact which is relevant with respect to his goals and intentions might be responsible for triggering a certain emotion of the agent. For example, imagine an agent perceiving the fact "there is a lion in front of me" which is relevant with respect to his goal of survival. Then, the agent will feel an intense fear caused by the perception of this fact.

Such a notion of pragmatic relevance has been considered in some design models of active perception (see, e.g., [26]) and of low-level mechanisms in which relevance is related to action selection through classifier systems [25]. Some models exist explaining the relationship that exists between perceptive processes and agent reasoning. Among others, Pereira & Tettamanzi recently proposed a formal model of goal generation where both relevance and trustworthiness of information sources are involved in rational goal selection [8]. Few works deal with agent models which are capable of appraising incoming input on the basis of cognitive relevance: [12] proposed a model where percepts are filtered according to a programmable heuristic defined in a so called *impact function*, while in [16] a model relating relevant information to an internal value indicating unexpectedness and surprisingness of percepts is envisaged.

The aim of this work is to ground the concept of relevance on mental states such as beliefs, goals and intentions. Once grounded on mental states, the notion of pragmatic relevance can be easily integrated in the practical reasoning of an agent and, in particular, it can be operationalized in the widely adopted BDI (belief, desire, intention) computational model of cognitive agents [21] and then integrated into a programming model for intelligent agents and multi-agent systems.

Before introducing our computational model of pragmatic relevance and in order to ground it on agents' mental states, in the next section the adopted agent reasoning model is briefly described.

3 The Abstract Agent Model

In this section a definition of an agent's abstract model based on BDI is provided. At the programming level this includes constructs obtained by perceptions, static causal

[1] See, e.g., [10] for a discussion on the distinction between informational relevance and pragmatic relevance.

knowledge, volatile beliefs, desires and intentions, desire-generating and planning rules, and a repertoire of basic actions.

Let $\mathrm{VAR}=\{X_1,\ldots,X_n\}$ be a non-empty set of random variables. We suppose that each random variable $X_i \in \mathrm{VAR}$ takes values from a non-empty set of variable assignments Val_{X_i}. For each set Val_{X_i} we denote by Inst_{X_i} the corresponding set of all possible instantiations of random variable X_i. For example, suppose that $\mathrm{Val}_{X_i}=\{x_1,\ldots,x_r\}$ then $\mathrm{Inst}_{X_i}=\{X_i{=}x_1,\ldots,X_i{=}x_r\}$. We denote by Inst the set of all possible instantiations of all random variables, that is: $\mathrm{Inst}=\bigcup_{X_i\in\mathrm{VAR}}\mathrm{Inst}_{X_i}$.

Perceived data. $\Gamma \subseteq \mathrm{Inst}$ is a set of perceived data which fixes the value of certain variables that an agent perceives at a certain moment. For example, $\Gamma=\{X_i{=}x\}$ means "the agent perceives the event $X_i{=}x$". We denote by

$$\Gamma_{Var}=\{X_i \in \mathrm{VAR} \mid \exists\, x \in \mathrm{Val}_{X_i} \text{ such that } X_i{=}x \in \Gamma\}$$

the subset of VAR which includes the variables that an agent observes at a certain moment. Here we suppose that for all $X_i \in \Gamma_{Var}, \mathrm{Inst}_{X_i} \cap \Gamma$ is a singleton, that is, we suppose that an agent cannot perceive two different instantiations of the same variable. We use the notation $\Gamma(X_i)$ to denote this singleton for every $X_i \in \Gamma_{Var}$, that is, $\Gamma(X_i){=}\mathrm{Inst}_{X_i} \cap \Gamma$.

Stable causal knowledge. K is a Bayesian network which represents the joint probability distribution over the set of random variables VAR. A Bayesian network is a directed acyclic graph (DAG) whose nodes are labeled by the random variables in VAR and the edges represent the causal influence between the random variables in VAR [18]. Given an arbitrary random variable X (i.e. an arbitrary node) in the Bayesian network K we denote by $\mathrm{anc}(X)$ the set of ancestors of X. Formally, Z is an *ancestor* of X in the Bayesian network K if there is a directed path from Z to X in K.

Moreover, given an arbitrary random variable X in the Bayesian network K, we denote by $\mathrm{par}(X)$ the set of parents of X in the Bayesian network. Formally, Z is a *parent* of X in the Bayesian network K if Z is an ancestor of X in K which is directly connected to Z. Finally, we associate to each random variable X in K a conditional probability distribution $P(X \mid \mathrm{par}(X))$. The Bayesian network K encodes the agent's causal knowledge of the environment. Here we suppose that this part of the agent's knowledge is stable and can not be reconsidered.

Volatile beliefs. The abstract agent model also includes beliefs that can change over time, i.e. the agent's volatile beliefs [5]. Given a random variable $X_i \in \mathrm{VAR}$, we denote by $\sum_{X_i}$ the set of all possible probability distributions over the random variable X_i. We denote by BEL the cartesian product of all $\sum_{X_i}$, that is $\mathsf{BEL}=\prod_{X_i\in\mathrm{VAR}}\sum_{X_i}$. BEL includes all possible combinations of probability distributions over the random variables in VAR. Elements in BEL are vectors $\mathsf{B}=\langle \mathsf{B}_1,\ldots,\mathsf{B}_n\rangle, \mathsf{B}'=\langle \mathsf{B}'_1,\ldots,\mathsf{B}'_n\rangle,\ldots$. Every vector B in BEL corresponds to a particular configuration of beliefs of the agent. In this sense, BEL includes all potential configurations of beliefs of the agent.

Suppose that $\mathrm{Val}_{X_i}=\{x_1,\ldots,x_r\}$. Then, every element B_i in a configuration of beliefs B is just a set $\{(X_i{=}x_1){=}a_1,\ldots,(X_i{=}x_r){=}a_r\}$ of probability assignments $a_1,\ldots,a_r \in [0,1]$ to each possible instantiations of the variable X_i.

Given a specific configuration of beliefs $\mathsf{B}=\langle \mathsf{B}_1, \ldots, \mathsf{B}_n \rangle$, we write $\mathsf{B}(X_i{=}x){=}a$ if and only if $(X_i{=}x){=}a \in \mathsf{B}_i$. For example, $\mathsf{B}(X_i{=}x){=}0.4$ means that given the configuration of beliefs $\mathsf{B}=\langle \mathsf{B}_1, \ldots, \mathsf{B}_n \rangle$ the agent assigns probability 0.4 to the fact that variable X_i takes value x. Moreover, we denote by $\mathsf{B}(X_i{=}x)$ the number $a \in [0, 1]$ such that $\mathsf{B}(X_i{=}x){=}a$.

Intentions and desires. We also model motivational attitudes by denoting with INT the set of potential intentions of an agent. Here we suppose that every instantiation of a variable in Inst is a potential intention of the agent, that is, $\mathsf{INT} = \mathrm{Inst}$. We denote by $\mathsf{I}, \mathsf{I}', \ldots \in 2^{\mathsf{INT}}$ specific sets of intentions of the agent. Given a specific set of intentions of the agent I, we denote by I_{Var} the subset of VAR which includes all intended variables, that is, all those variables which have (at least) one instantiation in I. Formally:

$$\mathsf{I}_{Var} = \{X_i \in \mathrm{VAR} \mid \exists\, x \in \mathrm{Val}_{X_i} \text{ such that } X_i{=}x \in \mathsf{I}\}.$$

We call *intention supports* all variables that are parents in the Bayesian network K of some intended variable. The set of intention supports is formally defined as follows:

$$\mathsf{SUPP}_{Var} = \{X_i \in \mathrm{VAR} \mid \exists\, X_j \in \mathsf{I}_{Var} \text{ such that } X_i \in \mathrm{par}(X_j)\}.$$

Note that the set of intention supports includes intention preconditions, that is, all conditions on which the achievement of an intended result of the agent depends.

DES is the set of all potential desires of the agent. As for intentions, we suppose that every instantiation of a variable in Inst is a potential desire of the agent, that is, $\mathsf{DES}{=}\mathrm{Inst}$. We denote by $\mathsf{D}, \mathsf{D}', \ldots \in 2^{\mathsf{DES}}$ specific sets of desires of the agent.

Desire-generating rules and planning rules. We specify a set DG of desire-generating rules and a set PL of planning rules. A desire-generating rule in DG is a desire-generating rule in the style of [11] of the form:

$$\psi_1, \ldots, \psi_s \mid \lambda_1, \ldots, \lambda_j \Longrightarrow \varphi_1, \ldots, \varphi_t.$$

Such a rule is responsible for generating t desires $\varphi_1, \ldots, \varphi_t$ when the agent has s beliefs $\psi_1, \ldots, \psi_s$ and j intentions $\lambda_1, \ldots, \lambda_j$.[2] The set of desire-generating rules DG corresponds to a function $options : \mathsf{BEL} \times 2^{\mathsf{INT}} \mapsto 2^{\mathsf{DES}}$. This function returns a specific set D of desires, given a specific configuration B of beliefs and a specific set I of intentions.

A planning rule in the set of planning rules PL is a plan-generating rule of the form:

$$\psi_1, \ldots, \psi_s \mid \lambda_1, \ldots, \lambda_j \Longrightarrow \alpha_1, \ldots, \alpha_t.$$

Such a rule is responsible for generating t plans $\alpha_1, \ldots, \alpha_t \in \mathsf{ACT}$, where ACT is the repertoire of actions and plans of our agent, when the agent has s beliefs $\psi_1, \ldots, \psi_s$ and j intentions $\lambda_1, \ldots, \lambda_j$. The set of planning rules PL corresponds to a function $plan : \mathsf{BEL} \times 2^{\mathsf{INT}} \mapsto 2^{\mathsf{ACT}}$.

This function returns a set π of plans, given a specific set B of beliefs and specific set I of intentions. To summarize, an agent's abstract model is defined as a tuple $\langle \Gamma, \mathsf{K}, \mathsf{B}, \mathsf{D}, \mathsf{I}, \mathsf{DG}, \mathsf{PL}, \mathsf{ACT} \rangle$, where each element in the tuple is defined as before.

[2] Our desire-generating rules correspond to the goal generation rules of the BOID framework [4].

4 Formalization of the Experimental Scenario

A simple experimental scenario is here introduced for testing relevance aware agents in a concrete problem domain. The scenario has been inspired by *Tileworld*, a simulation game originally introduced in [20]. Despite its simplicity, the experiment has been conceived as a testbed by introducing a highly parameterized environment that can be used to investigate and compare several performances in agents' reasoning processes. In this case, the environment layout is represented by the 12×12 grid in Fig. 1(a). An agent moves in the grid being driven by the goal of finding fruits of a certain color, according to the ongoing season. Indeed, agents look for fruits of different colors in different seasons of the year. We suppose that there are three different seasons and related colors of fruits and trees: the red season, the blue season and the green season. Agents are intrinsically motivated to look for and to eat red fruits during the red season, blue fruits during the blue season and green fruits during the green season. Environmental dynamics are characterized by periodic season cycles: after s_t rounds the season changes on the basis of a periodic function and the intrinsic motivation of an agent changes accordingly. Fruits of any color occupy cells (i, j) (with $1 \leq i \leq 16$ and $1 \leq j \leq 9$), where i indicates the number of the macro-area in the grid and j the number of the cell inside the macro-area. Trees of any color occupy macro-areas i of size 3×3 (with $1 \leq i \leq 16$) in the grid depicted in Fig. 1(a). We suppose that at each moment for every color there is exactly one fruit and tree of that color in the grid. Moreover, we suppose an objective dependence between trees and fruits in the grid: a fruit of a certain color is a sign of the presence of a fruit of the same color in the immediate neighborhood. Agents exploit these signs during their search of fruits. We suppose that a tree of any color is randomly placed in a macro-area i of size 3×3. Given a tree of a certain color in a macro-area i of size 3×3, a fruit of the same color is randomly placed by the simulator in one of the nine cells inside the macro-area i. For example, if a red tree is in the macro-area 1 of the grid then for each cell $(1, i)$ with $1 \leq i \leq 9$, there is $\frac{1}{9}$ of probability that

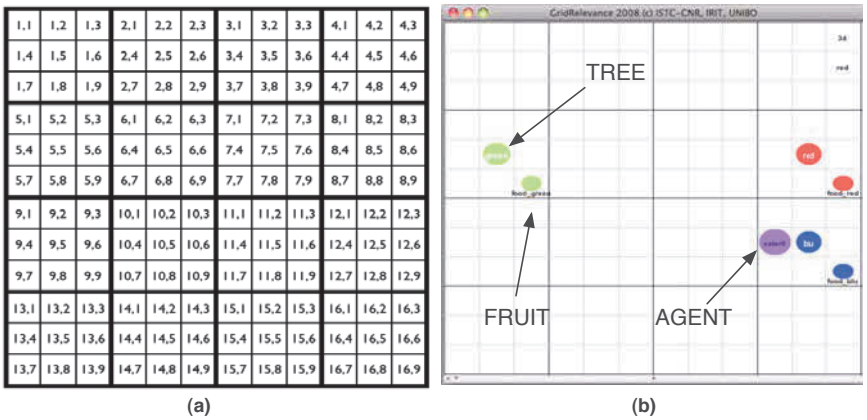

Fig. 1. Environment grid in agent's domain representation (a) and the running experiment (b)

a red fruit is located in that cell. Fruits and trees change periodically their positions. The dynamism factor δ indicates how many seasons have to pass before a tree location changes.

We impose constraints on the perceptual capabilities of agents by supposing that an agent sees only those fruits which are in the cells belonging to the same macro-area in which the agent is. For example, if the agent is in cell $(6, 1)$, he only sees those fruits which are in the cells belonging to the macro-area 6. Moreover we suppose that an agent sees only those trees which are situated in the same macro-area in which the agent is or in the four neighboring macro-areas on the left, right, up or down. For example, if the agent is in cell $(6, 1)$, he only sees those trees which are in macro-areas $2, 5, 7$ or 10.

The knowledge of our agents is encoded by means of the following eight random variables VAR$=\{SEAS, POS, RF, BF, GF, RT, BT, GT\}$. The variables RF, BF, GF, POS take values from the sets $\{(i, j) \mid 1 \leq i \leq 16, 1 \leq j \leq 9\}$, whilst the variables RT, BT, GT take values from the set $\{i \mid 1 \leq i \leq 16\}$. Finally, $SEAS$ takes value from the set $\{r, b, g\}$. Variables RF, BF, GF specify respectively the position of a red/ blue/ green fruit in the grid depicted in Fig. 1 (a). Variables RT, BT, GT specify respectively the position of a red/blue/green tree in the grid. For example, RT=13 means "there is a red tree in the macro-area 13". Variable $SEAS$ specifies the current season. For example, $SEAS$=$blue$ means "it is time to look for blue fruits!". Finally, Variable POS specifies the position of the agent in the grid.

The variables in VAR are organized in the Bayesian network K as follows: $\mathrm{par}(POS)=\{\emptyset\}$, $\mathrm{par}(SEAS)=\{\emptyset\}$, $\mathrm{par}(RT)=\{\emptyset\}$, $\mathrm{par}(BT)=\{\emptyset\}$, $\mathrm{par}(GT)=\{\emptyset\}$, $\mathrm{par}(RF)=\{RT\}$, $\mathrm{par}(BF)=\{BT\}$, $\mathrm{par}(GF)=\{GT\}$. Since there are 144 possible positions of a fruit and 16 possible positions of a tree in the grid, each conditional probability table associated with $P(RF \mid RT)$, $P(BF \mid BT)$ and $P(GF \mid GT)$ has 144×16=2304 entries. We suppose that the knowledge an agent has about the dependencies between trees and fruits perfectly maps the objective dependencies between trees and fruits. Hence, we only specify for each tree of a certain color – and arbitrary macro-area $i \in \{1, \ldots, 16\}$ in the grid in which a tree can appear – the 9 conditional probabilities that a fruit of the same color appears in one cell in that macro-area. We suppose for each of them the same probability value $\frac{1}{9}$. All other conditional probabilities have value 0, that is, given a tree of a certain color which appears in an arbitrary macro-area $i \in \{1, \ldots, 16\}$, the probability that there is a fruit of the same color outside that macro-area is zero. More precisely, we have that for all $1 \leq i, j \leq 16$ and $1 \leq z \leq 9$: *(i)* if i=j then $P(RF{=}(j, z) \mid RT{=}i){=}\frac{1}{9}$; *(ii)* if $i \neq j$ then $P(RF{=}(j, z) \mid RT{=}i){=}0$.

Desire-generating rules in DG are exploited by agents for solving the general task of finding a fruit of a certain color in the grid. Agents are endowed with three general classes of desire-generating rules.

The first class includes desire-generating rules of the following form. For $i \in \mathrm{Val}_{SEAS}$: $(SEAS{=}i){=}1 \Longrightarrow SEAS{=}i$. These desire-generating rules are responsible for changing the intrinsic motivation of an agent, according to the season change, that is: if an agent is certain that it is time to look for fruits of kind i, then he should form the desire to look for fruits of kind i.

The second class includes desire-generating rules shown in Table 1 (DG *Group 1*). These are responsible for orienting the search toward a certain macro-area, according to

Table 1. Desire-generating rules governing agents' intention selection. At an implementation level (i.e. in *Jason*) the formuled expressions are stated in terms of context conditions and used for the applicability of related plans.

DG **Group 1**	DG **Group 2**
For $1 \leq i \leq 16$:	For $1 \leq i \leq 16$ and $1 \leq j \leq 9$:
$(RT{=}i){=}1 \mid SEAS{=}r \Longrightarrow RT{=}i$	$(RF{=}(i,j)){=}1 \mid SEAS{=}r \Longrightarrow RF{=}(i,j)$
$(BT{=}i){=}1 \mid SEAS{=}b \Longrightarrow BT{=}i$	$(BF{=}(i,j)){=}1 \mid SEAS{=}b \Longrightarrow BF{=}(i,j)$
$(RT{=}i){=}1 \mid SEAS{=}g \Longrightarrow GT{=}i$	$(RF{=}(i,j)){=}1 \mid SEAS{=}g \Longrightarrow GF{=}(i,j)$

the current season (i.e. an intention to find fruits of a certain color) and his beliefs about the position of trees in the grid. For instance, if an agent is certain that there is a red tree in the macro-area 3 of the grid (i.e. $(RT{=}3){=}1$) and desires to find a red fruit (i.e. $SEAS{=}red$), then he should form the intention to reach that position of a red tree (i.e. $RT{=}3$). Finally, agents are endowed with the kind of desire-generating rules shown in Table 1 (DG *Group 2*). These desire-generating rules are responsible for orienting the search of an agent toward a certain cell, according to the current season (i.e. an intention to find fruits of a certain color) and his beliefs about the position of fruits in the grid. For example, if an agent desires to find a blue fruit (i.e. $SEAS{=}blue$) and knows/ is certain that there is a blue fruit in cell $(10, 1)$ of the grid (i.e. $(BF{=}(10, 1)){=}1$), then he should form the intention to move toward that position of the blue fruit (i.e. $BF{=}(10, 1)$).

We suppose that agents have five basic actions in repertoire: $MoveDown$, $MoveUp$, $MoveLeft$, $MoveRight$ and $EatFruit$. Indeed, at each round they can only move from one cell to the next one. Planning rules encode approaching policies which depend on the agent's current intentions and his actual position in the grid. Agents have both planning rules for reaching macro-areas in the grid (given their current positions) and planning rules for reaching cells in the grid (given their current positions). The latter planning rules are exploited for the local search of a fruit of a certain color inside a macro-area. An example of these planning rule is the following: $(POS{=}(15, 1)){=}1 \mid RT{=}3 \Longrightarrow MoveUp$. Thus, if an agent intends to reach position 3 of a red tree and is certain to be in cell $(15, 1)$ then he should form the plan to move one step up.

5 Pragmatic Relevance and Belief Update

In this section we present two different architectures and corresponding typologies of cognitive agents to be tested in the scenario described above. The first type of agent corresponds to a standard BDI agent whose control loop is described in the right column of Table 2. The second type of agent, whose control loop is described in the left column of Table 2, is a BDI agent endowed with a relevance-based mechanism of belief update. We call this second type of agent BDI^{rel} agent.

The formal description of the control loop of the standard BDI agent is similar to [27,22]. In lines 1-2 the beliefs and intentions of the agent are initialized. The main

Table 2. Abstract interpreter implemented by the two typologies of agents

BDI^{rel} **agent control loop**	BDI **agent control loop**
1. B:=B_0;	1. B:=B_0;
2. I:=I_0;	2. I:=I_0;
3. while (true) do	3. while (true) do
4. get new percept Γ;	4. get new percept Γ;
5. if $\mathrm{REL}(\mathsf{I}, \Gamma, \mathsf{B}) > \Delta$ then	5. B:=$bu(\Gamma, \mathsf{B})$;
6. B:=$bu^*(\Gamma, \mathsf{B}, \mathsf{I})$;	6. D:=$options(\mathsf{B}, \mathsf{I})$;
7. end-if	7. I:=$filter(\mathsf{B}, \mathsf{D}, \mathsf{I})$;
8. D:=$options(\mathsf{B}, \mathsf{I})$;	8. π:=$plan(\mathsf{B}, \mathsf{I})$;
9. I:=$filter(\mathsf{B}, \mathsf{D}, \mathsf{I})$;	9. $execute(\pi)$;
10. π:=$plan(\mathsf{B}, \mathsf{I})$;	10. end-while
11. $execute(\pi)$;	
12. end-while	

control loop is in lines 3-10. In lines 4-5 the agent perceives some new facts Γ and updates his beliefs according to a function bu. In line 6 the agent generates new desires by exploiting his desire-generating rules. In line 7 he deliberates over the new generated desires and his current intentions according to the function $filter$.[3] Finally, in lines 8-9 the agent generates a plan for achieving his intentions by exploiting his planning rules and he executes an action of the current plan. The main difference between the standard BDI agent and the BDI^{rel}agent is the belief update part in the control loop. We suppose that a process of belief update is triggered in the BDI^{rel}agent only if the agent perceives a fact and evaluates this to be relevant with respect to what he intends to achieve (line 5 in the control loop of the BDI^{rel}agent). In this sense, the BDI^{rel}is endowed with a cognitive mechanism of relevance-based belief update. In fact, this mechanism filters out all perceived facts that are irrelevant with respect to the current intentions. Thus, the BDI^{rel}agent only updates his beliefs by inputs which are relevant with respect to his current intentions. Differently, at each round the standard BDI agent updates his beliefs indiscriminately: for any fact he perceives, he updates his beliefs whether the perceived fact is relevant with respect to his intentions or not.

One might argue that the belief update strategy adopted by the BDI^{rel}agent is somewhat shortsighted. Indeed, the BDI^{rel}agent only considers inputs which are relevant with respect to his current intentions. We postpone to future work the analysis and the design of more sophisticated agents who consider in their belief update strategies also those inputs that they *expect* to be relevant for their future intentions.

In order to design the mechanism of relevance-based belief update, we define a notion of local relevance of an input Γ with respect to an intention $Y{=}y \in \mathsf{I}$, given the configuration of beliefs B. This is denoted by $\mathrm{rel}(Y{=}y,\Gamma,\mathsf{B})$ and is defined as follows, where for every $c \in [-1, 1]$, $\mathsf{Abs}[c]$ returns the absolute value of c.

[3] Space restrictions prevent a formal description of the function $filter$ here (see [27] for a detailed analysis). Only notice that this function is responsible for updating the agent's intentions with his previous intentions and current beliefs and desires (i.e. $filter : \mathsf{B} \times 2^{\mathsf{I}} \times 2^{\mathsf{D}} \mapsto 2^{\mathsf{I}}$).

$$\mathrm{rel}(Y{=}y,\Gamma,\mathsf{B}){=}\begin{cases}\Rightarrow \text{ If } Y \in \Gamma_{Var}: \\ 1{-}\mathsf{B}(\Gamma(Y)) \\ \\ \Rightarrow \text{ If } \mathrm{par}(Y) \subseteq \Gamma_{Var} \text{ and } Y \notin \Gamma_{Var}: \\ \mathsf{Abs}[\mathsf{B}(Y{=}y){-}P(Y{=}y \mid \{X_i{=}x \mid X_i \in \mathrm{par}(Y) \text{ and } X_i{=}x \in \Gamma\})] \\ \\ \Rightarrow \text{ If } \mathrm{par}(Y) \not\subseteq \Gamma_{Var} \text{ and } Y \notin \Gamma_{Var}: \\ 0\end{cases} \quad (1)$$

The degree of local relevance of the percept Γ with respect to intended fact $Y{=}y \in \mathsf{I}$ (given the agent's configuration of beliefs B) is defined on the basis of three conditions.

According to the first condition, if the intended variable Y is also a perceived variable in Γ_{Var} (i.e. there exists an instantiation of Y which is an element of Γ) then, $\mathrm{rel}(Y{=}y,\Gamma,\mathsf{B})$ is equal to the degree of unexpectedness of the perceptΓ(i.e. $1{-}\mathsf{B}(\Gamma(Y))$). The degree of unexpectedness of the percept Γ is inversely proportional to the prior probability assigned by the agent to the perceived instantiation of the intended variable Y (see [15] for an analysis of the notion of unexpectedness).

According to the second condition, if the intended fact $Y{=}y$ is not an instantiation of a perceived variable in Γ_{Var} and the parents of Y in the Bayesian network K are perceived variables in Γ_{Var} then, $\mathrm{rel}(Y{=}y,\Gamma,\mathsf{B})$ is equal to the degree of discrepancy between the intended fact $Y{=}y$ and the percept Γ. The degree of discrepancy between the intended fact $Y{=}y$ and the percept Γ is given by the absolute value of the difference between the probability assigned to $Y{=}y$ (i.e. $\mathsf{B}(Y{=}y)$) and the conditional probability that $Y{=}y$ is true given that the perceived instantiations of the parents of Y are true (i.e. $P(Y{=}y \mid \{X_i{=}x \mid X_i \in \mathrm{par}(Y) \text{ and } X_i{=}x \in \Gamma\})$).

According to the third condition, if the intended fact $Y{=}y$ is not an instantiation of a perceived variable in Γ_{Var} and there is some parent of Y in the Bayesian network K that is not a perceived variable in Γ_{Var} then $\mathrm{rel}(Y{=}y,\Gamma,\mathsf{B})$ is zero. This third condition corresponds to the irrelevance of the incoming input Γ with respect to the agent's intention $Y{=}y$. Under this third condition, the agent simply ignores the input.

Let us now define a notion of global relevance, noted $\mathrm{REL}(\mathsf{I},\Gamma,\mathsf{B})$, as the maximum value of local relevance for each intended fact $Y{=}y \in \mathsf{I}$:

$$\mathrm{REL}(\mathsf{I},\Gamma,\mathsf{B}){=}\max_{Y=y\in\mathsf{I}} \mathrm{rel}(Y{=}y,\Gamma,\mathsf{B}) \quad (2)$$

This notion of global relevance is used in the control loop of the BDI^{rel}agent: if the new percept Γ is responsible for generating a degree of global relevance higher than Δ (with $\Delta \in [0,1]$) then a process of belief update is triggered and the BDI^{rel}agent adjusts his beliefs with the perceived data Γ according to a function bu^*. The belief update function bu^* of the BDI^{rel}agent takes in input the set of intentions I, the belief configuration B and the percept Γ and returns an update belief configuration B', that is:

$$bu^* : 2^{\mathrm{Inst}} \times \mathsf{BEL} \times 2^{\mathsf{INT}} \mapsto \mathsf{BEL}.$$

More precisely, suppose that $bu^*(\Gamma,\mathsf{B},\mathsf{I}){=}\mathsf{B}'$. The set B' is defined according to the following three conditions. For every $Y \in \mathrm{VAR}$ we have:

(A) If $Y \in \mathsf{I}_{Var}$ or $Y \in \mathsf{SUPP}_{Var}$, and $Y \in \Gamma_{Var}$ then:
$\mathsf{B}'(\Gamma(Y)){=}1$ and for every $Y{=}x \in \mathrm{Inst}_Y \setminus \Gamma(Y), \mathsf{B}'(Y{=}x){=}0$.

(B) If $Y \in \mathsf{I}_{Var}$ or $Y \in \mathsf{SUPP}_{Var}$, and $\mathrm{par}(Y) \subseteq \Gamma_{Var}$ and $Y \notin \Gamma_{Var}$ then:
for every $Y{=}y \in \mathrm{Inst}_Y$, $B'(Y{=}y){=}P(Y{=}y \mid \{X_i{=}x \mid X_i \in \mathrm{par}(Y)$ and $X_i{=}x \in \Gamma\})$.
(C) Otherwise:
for every $Y{=}y \in \mathrm{Inst}_Y$, $B'(Y{=}y){=}B(Y{=}y)$.

According to the previous formal characterization of the function bu^*, the BDI^{rel} agent only reconsiders the probability distributions over his intentions $Y \in \mathsf{I}_{Var}$ and over his intention supports $Y \in \mathsf{SUPP}_{Var}$. In fact, we suppose that the BDI^{rel} agent only reconsiders those beliefs which are directly related with his intentions or with his intention supports, since he allocates his attention on the current task he is trying to solve. More precisely: if Y is either an intended random variable in I_{Var} or an intention support in SUPP_{Var}, and Y is a perceived variable in Γ_{Var}, then the updated probability distribution over Y assigns probability 1 to the perceived instantiation $\Gamma(Y)$ of variable Y and probability 0 to all the other instantiations of variable Y (condition A); if Y is either an intended random variable in I_{Var} or an intention support in SUPP_{Var}, Y is not a perceived variable in Γ_{Var}, but Y's parents in the Bayesian network are perceived variables in Γ_{Var}, then the updated probability distribution over Y assigns to each instantiations $Y{=}y$ of variable Y a probability which is equal to the conditional probability that $Y{=}y$ is true given that the perceived instantiations of the parents of Y are true (i.e. $P(Y{=}y \mid \{X_i{=}x \mid X_i \in \mathrm{par}(Y)$ and $X_i{=}x \in \Gamma\})$) (condition B). In all other cases the probability distribution over Y is not updated (condition C).

Space restrictions prevent a formal description of the belief update function bu of the standard BDI agent. Let us only say that function bu (differently from the function bu^* of the BDI^{rel} agent) updates indiscriminately all beliefs of the agent, that is, at each round the standard BDI agent reconsiders the probability distributions over all random variables $Y \in \mathrm{VAR}$. The function bu has the same conditions of function bu^* specified above. The only difference is that in bu the requirement '$Y \in \mathsf{I}_{Var}$ or $Y \in \mathsf{SUPP}_{Var}$' is not specified.

6 Programming Model

This section introduces the programming model implementing the mechanism of relevance-based belief update described above. The experimental platform has been built on top of CArtAgO, a framework for developing MAS environments based on the abstraction of agents and artifacts [17]. Agents have been implemented by extending *Jason*, a programming platform for BDI agents based on AgentSpeak(L) [2].

Environment. The rationale behind the adoption of the Agents and Artifacts (A&A) meta-model for the design of the experimental platform resides in the particular interaction model provided by CArtAgO, where all the mechanisms related to agent's perception and action are regulated, at a system level, by the framework. Agents – independently from their internal model and technology– are allowed to play in CArtAgO environments by interacting with artifacts through operation of *use* which consists in exploiting the artifact's usage interface. Besides, agent's perceptive activities are defined through the notions of *observation* which consists in retrieving the information that artifacts display, and *perception*, enabling agents to sense signals and events

coming from artifacts. In this perspective artifacts are conceived as a target for agents' overall activity, and thus exploited by agents either to execute their actions upon the environment, and to obtain information in a machine-readable format. To implement the scenario described in section 4, the environment has been instrumented with the following artifacts.

- `Timer` provides agents with timing information and enables the automatic mechanisms regulating the dynamism of the environment. Accordingly, it makes available two properties which are related to its internal state: `ticktime` (indicating the actual value of simulated time) and `season` (indicating the value of the ongoing season).
- `GridBoard` provides operations to be used as pragmatic actions by agents (i.e. $Move$, Eat) and feedback information in terms of percepts about the effects of these actions in the environment. In addition, based on the temporal signals generated by the Timer and on the actions performed by the agents, it provides the logic governing the dynamics and the physical consistency of the overall environment.
- `GridboardGUI` is linked to the previously described artifacts and is based on their internal states. It provides the graphical user interface for the system and allows the execution of experiment trials.

It is worth to remark that the adopted artifact-based environment promotes a principled design for agent perceptive activities. Thanks to the defined A&A interaction, events coming from artifacts can signal to the agent situations which require special attention. These signals are automatically sent back to the agent control system in order to be filtered and processed. For instance, once an artifact observable property or a signal is sensed by an agent, it can be filtered according to the given relevance function and possibly become a perceived datum (i.e. a percept). Following the basic idea provided in this work, only if such a percept is relevant it can be used to update the agent's belief base.

Agents. The overall goal for agents is to find and to eat fruit items. At any time step (i.e., round) agents can perfom only one pragmatic action (i.e., a move action in an adjacent cell) while, the score associated to the various fruit items depends on the ongoing season: a fruit of a given color (e.g. blue) provides a reward of $+1$ only if the ongoing season has the same color (e.g. the blue season), otherwise agents obtain no reward. In so doing, agents' performances are straightforwardly related to the environments dynamics. It is supposed that a tree changes its position at regular time intervals due to the ongoing dynamism δ, hence agents need to adaptively govern their behavior with respect to the actual situation. In particular, an agent needs to maintain an updated knowledge of the overall environment in order to avoid wasting resources, thus a certain amount of agent resources need to be allocated to exploration and epistemic activities. Besides, looking for food items in an area which is actually related to a food type that is not consistent with the ongoing season is a disadvantageous behavior in terms of global performance. In these conditions, an effective strategy is to look for fruits by using trees as reference points. Once a tree which is related to the ongoing season is encountered, the agent can perform epistemic actions aimed at updating his beliefs about the presence of fruits in the macro-area in which the agent is located.

At an architectural level, a Bayesian network governing goal deliberation and intention selection has been realized. To have a seamless integration with the *Jason* programming language, the kernel units used by the reasoning engine to perceive and update beliefs have been extended (namely, the `architecture.AgArch` and `asSemantics.Agent` components). In particular, specialized data-types and methods allowing the dynamic query on the probability distribution of domain entities have been introduced. In addition, agent's representation of the problem domain has been realized with a series of dynamic hash-maps. Each hash-map is, in this case, a problem representation related to a given season type. At any given time step this working memory can be accessed by the agent by indicating the coordinates of a given cell, thus returning the content of the cell in terms of entities which are expected to be there. For instance, once a tree is perceived, the agent can control if any information relating to that location is present and possibly update it. Accordingly, when an agent updates his belief about the position of a certain tree, he will also update the belief about the position of the fruit of the same color, given a possibly known probability distribution (we assume that agents know which relation exists between the location of a certain tree and the probability to find food items in the cells belonging to the same area).

Thanks to their perceptive skills, once some particular signal is encountered, agents can exploit it for updating their beliefs, or for reconsidering their intentions. Therefore, after becoming aware of some relevant facts, agents can elicit belief update or an adaptive re-allocation of their computational resources. Artifacts provide, in this case, two kinds of signals: signals for temporal synchronization (agents rule their actions based on a clock signals perceived from the `Timer`) and signals belonging to the set Γ_{Var}, which in turn contains the percepts corresponding to visible entities. For instance, as shown in the following *Jason* cutout, once a clock signal is received from the focused `Timer`, an internal action `gridworld.perceptRel` interacts with the `GridBoard` to retrieve the list of percepts indicating all the visible entities.

```
+tick_time(X)[source("percept"), artifact("timer"), workspace("Relevance")]
  : actual_season(S)
  <- -+time(X);
     gridworld.perceptRel(S);
     !deliberateTarget.
```

The `gridworld.perceptRel` perceptive action has two different implementations respectively for the BDI agent and for the BDI^{rel}agent. `gridworld.perceptRel` is supposed to realize the belief update functions bu and bu^*. Percepts are inserted in the agent working memory and then filtered by the belief update function. In the case of the BDI^{rel}agent, once these percepts are related to the current intention (i.e. actual season) they are stored in the agent memory as permanent belief facts. In particular, bu^* is supposed to retrieve from the `GridBoard` the list of visible entities (these elements become percepts). Moreover, for each retrieved fact, bu^* deletes the beliefs actually referring to entities which are not already present in the actual range of sight (trees and fruits can disappear due to the environment dynamism) and adds a new fact to the belief base only if the scrutinized percept in Γ matches the current intention. For discriminating relevant and not relevant percepts a simple pattern matching is used. Hence, the function of local relevance $\mathrm{rel}(Y{=}y,\Gamma,\mathsf{B})$ is greater than zero when the current season matches the entity

Table 3. Amount of achieved goals and performed belief updates measured at the end of the experiment series for BDI and BDI^{rel} agents in environments with dynamism $\delta \in \{1, 2, 3\}$

	$\delta = 3$		$\delta = 2$		$\delta = 1$	
	BDI	BDI^{rel}	BDI	BDI^{rel}	BDI	BDI^{rel}
$Goal.eff$	42.375	42.375	38.185	37.625	33.937	33.875
$Cost.eff$	92.125	78.437	101.437	85.312	143.125	107.875
$cost.ratio$	2.381	2.012	2.882	2.252	4.379	3.214

type, otherwise $r = 0$. Notice that in the case of the described scenario (where actual agent's intention depends on the current season) the threshold Δ is set to 0.

After the execution of the `perceptRel`, the BDI agent has a complete knowledge of the actual state of its surroundings (i.e. the belief base is supposed to be consistent with the actual state of the visible area). On the other side, BDI^{rel} only considers the information that is relevant with respect to the current situation. Then, to achieve their goals, both BDI and BDI^{rel} agents can adopt the following plans to decide the next course of action.

```
+!deliberateTarget
  : actual_season(S) & food(S,X,Y)
  <- -+targetLoc(X,Y);
     !doAction.
```

```
+!deliberateTarget
   : actual_season(S) & not food(S,_,_)
   & tree(S,X,Y)
   <- -+targetLoc(X,Y);
      !doAction.
+!deliberateTarget <- !doAction.
```

It is worth nothing that, according to the *Jason* transition system, the desire-generating rules described in Tab. 1 are here expressed by means of *context-conditions*, i.e. in form of expression of beliefs (*belief formulae*). The belief `targetLoc(X,Y)` can refer to a given fruit location only if the agent has already located a fruit and has stored a related fact in his belief base. If there are no facts in the belief base concerning fruits or trees related to the ongoing season, the agent will perform an epistemic action, i.e. by exploring the grid in order to discover some new relevant fact. Besides, once beliefs are canceled from the belief base by the internal belief update activities –ruled respectively by bu for the standard BDI agent and bu^* for the BDI^{rel} agent– the agent reconsiders its intentions and selects a new target to be reached.

```
-food(T,X,Y) : targetLoc(X,Y)
        <-  !deliberateTarget.
```

```
-tree(T,X,Y) : targetLoc(X,Y)
        <-  !deliberateTarget.
```

In so doing, both the BDI agent and the BDI^{rel} agent update their belief base in two circumstances: (*i*) when the actual belief base is wrong (not consistent with the perceived state of the environment), and (*ii*) when the actual belief base is incomplete (due to a lack of knowledge). Finally, by using the operations allowed by the `GridBoard` interface, and taking into account the planning rules discussed in section 4, a `!doAction` realizes the basic pragmatic actions (i.e., *eat* a fruit or *move* towards `targetLoc(X,Y)`).

7 Experiment

This section discusses a series of experiments comparing the performances of BDI vs. BDI^{rel} agents engaged in in the scenario presented in section 4.

7.1 Experiment Setting

In our experimental setting we suppose for simplicity that agents have always access to their current position in the grid, and that they are always notified about simulation steps and season changes. Therefore, at the beginning of a new season, an agent always knows that it is time to look for fruits of a different color, and thus adopting the goal to look for fruits belonging to the ongoing color. In order to have a measure about the trade-off between effectiveness and efficiency, agents' performances have been evaluated according to a twofold metric. On the one hand, goal effectiveness ($Goal.eff$) represents the total amount of achieved goals during a trial (i.e., eaten fruits), while cost effectiveness ($Cost.eff$) is the total amount of update operations performed by the agent on his belief base. On the other hand, we define the $cost.ratio$ of an agent in terms of the agent's belief update cost divided by the total amount of achieved goals ($Cost.eff/Goal.eff$). This in particular gives a quantitative measure of efficiency, namely how many units of cost the agent needs to spend for each achieved goal. In other terms, a $cost.ratio = 1$ means that the agents has performed a belief update operation for each achieved goal (i.e., eaten fruit). Besides, since only one $Move$ action is allowed for each time step, the adopted metrics provide insights on how many pragmatic actions are needed for agents to achieve their goals.

The length of experiments has been set to 900 rounds in order to be long enough for the chosen metrics to become stable. The global performance of each agent is measured by averaging $cost.ratio$ of 16 trials in environments with different dynamism δ. Season length s_t is set at 15 rounds, while we consider n=3 seasons (respectively red, blue and green) with three associated types of tree and fruit. The initial placements of entities are randomly selected, while a fruit of a given color is generated at the beginning of each corresponding season. Finally, we assume that for any color at most one fruit is present in the grid at the same time.

7.2 Discussion

Fig. 1 (b) shows a snapshot of the running simulation. Experiments have been conducted using three different variables of dynamism δ for the environment. The first two columns in Tab.3 show the $cost.ratio$ respectively for the BDI agent and the BDI^{rel}agent operating in a static environment ($\delta = 3$, a tree changes its location every 3 seasons, 45 rounds). Both agents attain an average of 43.375 eaten fruits on their trials. However they have to pay quite different costs of belief update (BDI performs an average of 92.125 update operations, while BDI^{rel}78.437). Considering the $cost.ratio$, once agents have overcome their transitory phase they spend respectively 2.381 (BDI) and 2.012 (BDI^{rel}) costs for each eaten fruit.

The central columns show the performances of the two agents in environments with medium dynamism ($\delta = 2$, a tree changes its location every 2 seasons, 30 rounds). Here, due to the frequent tree changes, both agents rely on a less accurate knowledge model. In terms of eaten fruits the BDI agent attains a higher performances (38.185), clearly outperforming the BDI^{rel}agent (37.125). On the other hand, as far as costs of belief update are concerned, BDI^{rel}performs better: in fact, under the same conditions, BDI^{rel}does less belief update operations (85.312 vs. 101.437). As a consequence, BDI^{rel}considerably shows a better effectiveness with respect to the $cost.ratio$

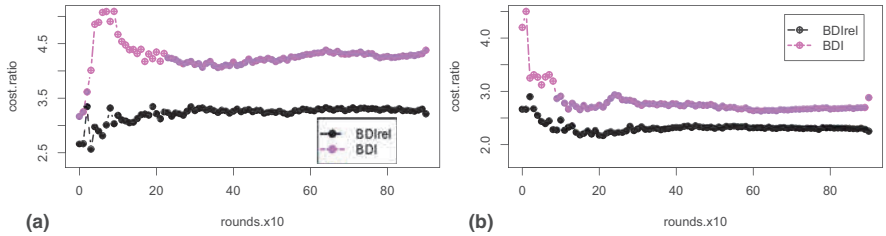

Fig. 2. Cost ratio in environments characterized by different dynamism: δ=1 (a) and δ=2 (b)

(see Fig.2.b) whose value converges, at the end of the experiments, at an average of 2.252 belief update operations per achieved goal (vs. 2.882 for the BDI agent).

The last two columns shows the results in a highly dynamic environment ($\delta = 1$, a tree changes its location every season change, 15 rounds). Here, due to the massive epistemic activities, the standard BDI agent is able to maintain a more consistent and complete knowledge of the environment. Therefore, he performs better than the BDI^{rel}agent in terms of achieved goals (with an average of 33.937 number of eaten fruits against 33.875). In contrast, the BDI agent has to pay higher costs related to his epistemic activities, even beyond the initial transitory phase. In this case, the costs of belief update are 143.125 for the BDI agent and 107.875 for the BDI^{rel}agent. As depicted in Fig.2.a, the $cost.ratio$ reflects this difference by converging to a value of about 4.379 for the BDI and of 3.214 for the BDI^{rel}.

Despite the simplicity of the problem domain, the experiments show a noticeable effect of the relevance-based filter of belief update on the BDI^{rel}performance. By reporting agent effectiveness both in terms of goal achieved and in terms of costs for belief update, $cost.ratio$ is, indeed, a good indicator for analyzing the trade-off in agents performances (see Fig.2). As the results of the experimental analysis show, on the one side the BDI agent is always the best in terms of goal effectiveness, thus able to achieve an higher amount of goals. BDI agents are *passively* affected by all incoming information, hence they obtain a precise knowledge of their surroundings and result in a more acknowledged decision making. Consequently, we may argue that the more an agent spends his resources for belief update, the more his belief base will be correct and adequate with respect to the current state of the environment, and the more the agent will find fruits in the grid. On the other side, BDI^{rel}agents adopt an active perception style and exploits their relevance-based mechanism to filter percepts: if the incoming input is not relevant with respect to the current intention, then the BDI^{rel}agent simply ignores it. Besides a worse performance in terms of goal achieved, this mechanism allows the BDI^{rel}agent to avoid wasting computational resources for belief update, thus resulting in a better global performance in terms of $cost.ratio$. This aspect is worth to be considered in cases where agents have bounded resources and environments are characterized by strong dynamicity or information richness. As the results show, the higher the dynamism of the environment is, the higher the computational costs which are paid by BDI agents for belief update processes. In these conditions, BDI^{rel}agents have a

concrete advantage in filtering noise and in considering only what is expected to be useful for achieving their current goals.

8 Conclusion

We have presented in this work a mechanism of relevance-based belief update providing a computational model for BDI agents, and implemented it using *Jason* platform. As the experimental results show, the costs for belief update are effectively reduced by using the mechanism for filtering relevant percepts. Despite the simple scenario adopted, the experimental results can be easily generalized as well as the computational model for relevance aware agents can be straightforwardly applied to more complex application domains. We think that a notion of pragmatic relevance is a pivotal one for the implementation of forthcoming agent-based systems, for instance in all the cases in which agents have to perform complex and resource demanding activities in highly dynamic and information-rich environments (i.e., the Web, as well as pervasive systems in the real world).

In this line, the directions for future works are manifold. We are actually working on a generalization of our model of pragmatic relevance which consists in adding a quantitative dimension for intentions. In this generalized model, the degree of relevance of a certain input with respect to an agent's intention will also depend on the utility/importance of the intended outcome. Besides, we will further investigate the relationships between the notion of relevance and intention reconsideration mechanisms. Since the persistence of an intention over time depends on the persistence of those beliefs which support this intention (i.e. beliefs are reasons for intending), we will study how the relevance-based filter of belief update discussed in this paper may affect the persistence of intentions in an indirect way. Moreover, we intend to develop in the future a more advanced model extending an agent's abilities to manage a probabilistic belief base (i.e., by introducing salience maps, dynamic Bayesian networks, influence diagrams, etc.).

Finally, as already noticed in section 5, we intend to extend our model to a more sophisticated type of agent than the BDI^{rel} agent who also considers inputs that he expects to be relevant for his future intentions. For instance, if such an agent has the current intention to cook a meal and he expects that tomorrow morning he will have the intention to go from Paris to Rome by train, he will also consider information that are relevant with respect to his expected intention (e.g. the information "there will a train strike").

References

1. Anderson, A.R., Belnap, N.D.: Entailment: the logic of relevance and necessity, vol. 1. Princeton University Press, Princeton (1975)
2. Bordini, R., Hübner, J.F., Wooldridge, M.: Programming Multi-Agent Systems in AgentSpeak Using Jason. John Wiley & Sons, Ltd., Chichester (2007)
3. Bratman, M.: Intentions, plans, and practical reason. Harvard University Press, Cambridge (1987)
4. Broersen, J., Dastani, M., Hulstijn, J., van der Torre, L.: Goal generation in the BOID architecture. Cognitive Science Quarterly 2(3-4), 428–447 (2002)

5. Casati, R., Pasquinelli, E.: How can you be surprised? the case for volatile expectations. Phenomenology and the Cognitive Sciences 6(1-2), 171–183 (2006)
6. Cherniak, C.: Minimal rationality. MIT Press, Cambridge (1986)
7. Chopra, S., Parikh, R.: Relevance sensitive belief structures. Annals of Mathematics and Artificial Intelligence 28(1-4), 259–285 (2000)
8. da Costa Pereira, C., Tettamanzi, A.G.B.: Goal generation with relevant and trusted beliefs. In: 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), pp. 397–404. ACM Press, New York (2008)
9. Ellsworth, P.C., Scherer, K.R.: Appraisal processes in emotion. In: Davidson, R.J., Goldsmith, H.H., Scherer, K.R. (eds.) Handbook of the affective sciences. Oxford University Press, Oxford (2003)
10. Floridi, L.: Understanding epistemic relevance. Erkenntnis 69, 69–92 (2008)
11. Kaelbling, L.P., Rosenschein, S.J.: Action and planning in embedded agents. In: Maes, P. (ed.) Designing autonomous agents, pp. 35–48. MIT Press, Cambridge (1990)
12. Koster, A., Koch, F., Sonenberg, L., Dignum, F.: Augmenting BDI with Relevance: Supporting Agent-based, Pervasive Applications. In: Mobile Interaction Devices (PERMID 2008) Workshop at Pervasive 2008, Sydney (2008)
13. Lakemeyer, G.: Relevance from an epistemic perspective. Artificial Intelligence 97(1-2), 137–167 (2004)
14. Lazarus, R.S.: Emotion and adaptation. Oxford University Press, New York (1991)
15. Lorini, E., Castelfranchi, C.: The unexpected aspects of surprise. International Journal of Pattern Recognition and Artificial Intelligence 20(6), 817–833 (2006)
16. Lorini, E., Piunti, M.: The Benefits of Surprise in Dynamic Environments: From Theory to Practice. In: Paiva, A.C.R., Prada, R., Picard, R.W. (eds.) ACII 2007. LNCS, vol. 4738, pp. 362–373. Springer, Heidelberg (2007)
17. Omicini, A., Ricci, A., Viroli, M.: Artifacts in the A&A meta-model for multi-agent systems. Autonomous Agents and Multi-Agent Systems 17(3) (2008)
18. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufman, Cambridge (1988)
19. Peppas, P., Chopra, S., Foo, N.Y.: Distance semantics for relevance-sensitive belief revision. In: Proceedings of the Ninth International Conference on Principles of Knowledge Representation and Reasoning (KR 2004), pp. 319–328. AAAI Press, Menlo Park (2004)
20. Pollack, M.E., Ringuette, M.: Introducing the tileworld: Experimentally evaluating agent architectures. In: National Conference on Artificial Intelligence (1990)
21. Rao, A.S., Georgeff, M.P.: Modelling rational agents within a BDI-architecture. In: Fikes, R., Sandewall, E. (eds.) Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR 1991), San Mateo, CA, pp. 473–484. Morgan Kaufmann Publishers, San Francisco (1991)
22. Rao, A.S., Georgeff, M.P.: BDI agents: from Theory to Practice. In: Proceedings of the First International Conference on Multi-Agent Systems, ICMAS 1995 (1995)
23. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal 27, 623–656 (1948)
24. Simon, H.: Models of thought, vol. 1. Yale University Press, New Haven (1979)
25. Weiß, G.: Learning the goal relevance of actions in classifier systems. In: Proc. of the Tenth European Conference on Artificial intelligence (ECAI 1992), pp. 430–434 (1992)
26. Weyns, D., Steegmans, E., Holvoet, T.: Model for active perception in situated multi-agent systems. Special Issue of Journal on Applied Artificial Intelligence 18, 200–204 (2003)
27. Wooldridge, M.: An Introduction to Multiagent Systems. John Wiley & Sons, Chichester (2002)