Observation-based Model for BDI-Agents*

Kaile Su[1,2], Abdul Sattar[1], Kewen Wang[1], Xiangyu Luo[2], Guido Governatori[3], Vineet Padmanabhan[3]
[1]Institute for Integrated and Intelligent Systems, Griffith University, Brisbane, Australia
[2]Department of Computer Science, Sun Yat-sen University, Guangzhou, P.R.China
[3]School of ITEE, The University of Queensland, Brisbane, Australia
{k.su, a.sattar, k.wang}@griffith.edu.au; xiang_yu_luo@yahoo.com.cn; {guido,vnair}@itee.uq.edu.au

Abstract

We present a new computational model of BDI-agents, called the observation-based BDI-model. The key point of this BDI-model is to express agents' beliefs, desires and intentions as a set of runs (computing paths), which is exactly a *system* in the interpreted system model, a well-known agent model due to Halpern and his colleagues. Our BDI-model is *computationally grounded* in that we are able to associate the BDI-agent model with a computer program, and formulas, involving agents' beliefs, desires (goals) and intentions, can be understood as properties of program computations. We present a sound and complete proof system with respect to our BDI-model and explore how symbolic model checking techniques can be applied to model checking BDI-agents. In order to make our BDI-model more flexible and practically realistic, we generalize it so that agents can have multiple sources of beliefs, goals and intentions.

Introduction

The *possible worlds semantics* (Kripke 1963) is a fruitful approach to formalizing agent systems via modal logics, because internal mental attitudes of an agent, such as beliefs and goals, can be characterized conveniently with a model theoretic feature in terms of certain accessibility relations. The well-known theory of intention (Cohen & Levesque 1990) and the formalism of the belief-desire-intention (BDI) paradigm (Rao & Georgeff 1998), for example, are along this line. Some of these logics can be reduced to standard modal logics such as mu-calculus (Schild 2000). However, it is still not clear how to get concrete agent models with the belief, desire and intention accessibility relations from specific agent programs.

A number of researchers have attempted to develop executable agent languages such as AgentSpeak(L) (Rao 1996), AGENT0 (Shoham 1993) and 3APL (Hindriks *et al.* 1999), but these agent languages are comparatively simple (van der Hoek & Wooldridge 2003).

The *interpreted system* model (Fagin *et al.* 1995; Halpern & Zuck 1992) offers a natural interpretation, in terms of the states of computer processes, to S5 epistemic logic. The salient point of the interpreted system model is that we can get an interpreted system from a computer program, and formulas in epistemic logic that are valid with respect to the interpreted system can be understood as properties of program computations. In this sense, the interpreted system model is *computationally grounded* (Wooldridge 2000). Thus, it is interesting to extend the interpreted system model to capture an agent's belief, desire and intention as well.

The aim of this paper is to present a computationally grounded model of general BDI-agents by extending the interpreted system model. The key point of our BDI-model is that we characterize an agent's belief, desire and intention as sets of runs (computing paths), which is exactly a *system* in the interpreted system model. Let $\mathcal{B}_i$, $\mathcal{D}_i$ and $\mathcal{I}_i$ be sets of runs related to agent i's belief, desire and intention, respectively. Then, runs in $\mathcal{B}_i$ are possible computing paths from the viewpoint of the agent; those in $\mathcal{D}_i$ are the computing paths that the agent desires; and those in $\mathcal{I}_i$ are computing paths with the agent's intentional choices of the possible actions. Clearly, it is reasonable to assume that $\mathcal{D}_i \subseteq \mathcal{B}_i$, that is, every desired computing path is possible one. Nevertheless, we need not assume that $\mathcal{I}_i \subseteq \mathcal{D}_i$ or even $\mathcal{I}_i \subseteq \mathcal{B}_i$ because an agent's intention may fail to achieve its goal and the real computing path may be beyond the agent's belief even though the agent has chosen and completed an intentional series of actions.

The advantage of our BDI-model is that the basic element, a system, can be symbolically and compactly represented via the ordered binary decision diagram (OBDD) (Bryant 1986). This plays an essential role in model checking general BDI-agents with practical efficiency. Moreover, the presented BDI-model naturally characterizes the relationship between the agent's observations or local states and the dynamics of its belief, desire and intention.

The structure of the paper is as follows. In the next section, we briefly introduce the interpreted system model. Then, we define a new model of BDI-agents, which is called interpreted BDI-system model. A computationally grounded BDI logic, called observation-based BDI logic (OBDI), is introduced and interpreted via two different semantics. We explore how symbolic model checking techniques can be applied in model checking BDI-agents and report some preliminary experimental results. Finally, we discuss some related

*This work was supported by the Australian Research Council grant DP0452628, and partially by the NSFC grants 60496327, 10410638 and 60473004.

work and conclude this paper.

Preliminaries

We start by giving some notions concerning the interpreted system model. Consider a system composed of multiple agents in an environment. We represent the system's state or the *global state* as a tuple $(s_e, s_1, \cdots, s_n)$, where s_e is the *environment's local state* and, for each i, s_i is *agent i's local state*.

Let L_e be a set of possible local states of the environment and L_i a set of possible local states for agent i, for $i = 1, \cdots, n$. We take $G \subseteq L_e \times L_1 \times \cdots \times L_n$ to be the set of *reachable global* states of the system. A *run* over G is a function from the time domain–the natural numbers in our case–to G. Thus, a run over G can be identified with a sequence of global states in G. For convenience, we use g^* to denote the sequence of infinitely repeated global state g; so, $g_0 g_1^*$, for example, indicates a run with the initial state g_0 followed by infinitely repeated state g_1.

We refer to a pair (r, m) consisting of a run r and time m as a *point*. Given a point (r, m), we denote the first component of the tuple $r(m)$ by $r_e(m)$ and, for each i $(1 \leq i \leq n)$, the $i+1$'th component of the tuple $r(m)$ by $r_i(m)$. Thus, $r_i(m)$ is the local state of agent i in run r at "time" m.

The idea of the interpreted system semantics is that a run represents one possible computation of a system and a system may have a number of possible runs, so we say a *system* is a set of runs.

Assume that we have a set Φ of primitive propositions, which we can think of as describing basic facts about the system. An *interpreted system* I consists of a pair $(\mathcal{R}, \pi)$, where $\mathcal{R}$ is a set of runs over a set of global states and π is a valuation function, which gives the set of primitive propositions true at each point in $\mathcal{R}$ (Fagin *et al.* 1995).

For every agent i, let $(r, u) \sim_i (r', v)$ denote that $r_i(u) = r'_i(v)$. Intuitively, $(r, u) \sim_i (r', v)$ means that (r, u) and (r', v) are indistinguishable to agent i. We also use $(r, u) \sim_i^{spr} (r', v)$ to denote $u = v$ and, for every $j \leq u$, $r_i(j) = r'_i(j)$ (here *spr* stands for *synchronous* systems with *perfect recall*).

Let KL_n denote the language of propositional logic augmented by the future-time connectives $\bigcirc$ (next) and $\mathbf{U}$ (until) and a modal operator K_i for each agent i. The language KL_n can be interpreted by using an interpreted system. The related satisfaction relationship $\models_{KL_n}$ is as follows: Given $I = (\mathcal{R}, \pi)$ and a point (r, u) in $\mathcal{R}$, we define $(I, r, u) \models_{KL_n} \psi$ by induction on the structure of ψ. When ψ is of the form $K_i\varphi$, $(I, r, u) \models_{KL_n} \psi$ iff $(I, r', v) \models_{KL_n} \varphi$ for all (r', v) such that $(r, u) \sim_i (r', v)$. It may also be standardly dealt with when ψ is atomic or of the form $\neg\varphi$, $\varphi \wedge \varphi'$, $\bigcirc\varphi$ or $\varphi \mathbf{U} \varphi'$.

The Interpreted BDI-system Model

In this section, we present a new model of BDI-agents, called the interpreted BDI-system model, which extends the interpreted system model. The key point of our approach is that we define agents' belief, desire and intention as sets of runs, that is, *systems* in the interpreted system model.

Formally, given a set G of global states and system $\mathcal{K}$ over G, an agent's *mental state* over system $\mathcal{K}$ is a tuple $\langle \mathcal{B}, \mathcal{D}, \mathcal{I} \rangle$, where $\mathcal{B}$, $\mathcal{D}$ and $\mathcal{I}$ are systems (sets of runs over G) such that $\mathcal{I} \subseteq \mathcal{K}$ and $\mathcal{D} \subseteq \mathcal{B} \subseteq \mathcal{K}$. A *BDI-system* is a structure $\langle \mathcal{K}, \mathcal{M}_1, \cdots, \mathcal{M}_n \rangle$, where $\mathcal{K}$ is a system and for every i, $\mathcal{M}_i$ is agent i's mental state over $\mathcal{K}$.

Assume that we have a set Φ of primitive propositions, which we can think of as describing basic facts about our BDI-agents and their environment. An *interpreted BDI-system* I consists of a pair $(\mathcal{S}, \pi)$, where $\mathcal{S}$ is a BDI-system and π is a valuation function, which gives the set of primitive propositions true at each point in G.

Example 1 Let us consider the scenario of a robot cleaning a room. There are two global states: s_0 and s_1. In state s_0, the room is dirty, and in state s_1, the room is clean. Moreover, the robot believes that the room is dirty, and the goal of the robot is to clean the room.

Now we define a BDI-system S_0=$\langle \mathcal{K}_0, \langle \mathcal{B}_0, \mathcal{D}_0, \mathcal{I}_0 \rangle \rangle$, where

- $\mathcal{K}_0$ is the set of those runs r such that, for every natural number m, if $r(m) = s_1$ then $r(m+1) = s_1$, which means that if the room is clean, then it will keep so.
- $\mathcal{B}_0$ is a set of those runs $r \in \mathcal{K}_0$ with $r(0) = s_0$. This means that the robot believes the room is dirty at first. Notice that *belief* is just the information state of the robot, and there is no guarantee that robot will make the room clean.
- $\mathcal{D}_0$ is a subset of $\mathcal{B}_0$ such that, for every run $r \in \mathcal{D}_0$, there is a number m with $r(m) = s_1$. This means that the robot desires to clean the room.
- Finally, we may define $\mathcal{I}_0$ to be the set of runs $s_0 {s_1}^*$ and ${s_1}^*$. This indicates the robot will clean the room immediately.

We may take $\{dirty\}$ as the set Φ of primitive propositions, representing whether the room is clean or dirty. Clearly, we may naturally define $\pi_0(s_0)(dirty) = 0$ and $\pi_0(s_1)(dirty) = 1$. Thus, we get the interpreted BDI-system (S_0, π_0).

OBDI Logic

In this section, we introduce a multimodal logic of belief, desire and intention, referred to as Observation-based BDI logic (OBDI). As shown below, the semantics of OBDI logic is given in terms of the interpreted BDI-system model illustrated above; According to this semantics, the computation of agents' beliefs, desires, and intentions are based on agents' *observations* (i.e. local states).

Syntax

Given a set Φ of propositional atoms, the language of OBDI logic is defined by the following BNF notations:

$$\begin{aligned} \langle wff \rangle ::= \quad & \text{any element of } \Phi \mid \neg\langle wff \rangle \mid \langle wff \rangle \wedge \langle wff \rangle \mid \\ & \bigcirc\langle wff \rangle \mid \langle wff \rangle \mathbf{U} \langle wff \rangle \mid \\ & \mid B_i\langle wff \rangle \mid D_i\langle wff \rangle \mid I_i\langle wff \rangle \end{aligned}$$

Informally, $B_i\varphi$ and $D_i\varphi$ means that agent i believes and desires φ, respectively, while $I_i\varphi$ denotes that φ holds un-

der the assumption that agent i acts based on his intention. The formulas not containing modalities B_i, D_i and I_i $(i = 1, \cdots, n)$ are called LTL formulas.

Semantics

We now proceed to interpret OBDI logic formulas in terms of interpreted BDI-systems. In what follows, we inductively define the satisfaction relation $\models_{BDI}$ between a formula φ and a pair of interpreted BDI-system and a point. Given an interpreted BDI-system $I = (\mathcal{S}, \pi)$, suppose that $\mathcal{S} = \langle \mathcal{K}, \mathcal{M}_1, \cdots, \mathcal{M}_n \rangle$ and for every i, $\mathcal{M}_i = \langle \mathcal{B}_i, \mathcal{D}_i, \mathcal{I}_i \rangle$. Let r be a run in $\mathcal{K}$ and u a natural number, then we have that:

- $(I, r, u) \models_{BDI} B_i\varphi$ iff $(I, r', v) \models_{BDI} \varphi$ for those (r', v) such that $r' \in \mathcal{B}_i$ and $(r, u) \sim_i (r', v)$;
- $(I, r, u) \models_{BDI} D_i\varphi$ iff $(I, r', v) \models_{BDI} \varphi$ for those (r', v) such that $r' \in \mathcal{D}_i$ and $(r, u) \sim_i (r', v)$;
- $(I, r, u) \models_{BDI} I_i\varphi$ iff $(I, r', v) \models_{BDI} \varphi$ for those (r', v) such that $r' \in \mathcal{I}_i$ and $(r, u) \sim_i (r', v)$;
- $(I, r, u) \models_{BDI} \bigcirc\varphi$ iff $(I, r, (u + 1)) \models_{BDI} \varphi$;
- $(I, r, u) \models_{BDI} \varphi \mathbf{U} \varphi'$ iff $(I, r, u') \models_{BDI} \varphi'$ for some $u' \geq u$ and $(I, r, u'') \models_{BDI} \varphi$ for all u'' with $u \leq u'' < u'$.

Other cases are trivial and omitted here. We say that a formula φ is valid in an interpreted BDI system I, denoted by $I \models_{BDI} \varphi$, if $(I, r, u) \models_{BDI} \varphi$ holds for every point (r, u) in $\mathcal{I}$. We use $\models_{BDI} \varphi$ to denote that φ is valid in every interpreted BDI-system.

According to our definition, $D_i\varphi$ is true iff φ is true along those runs that are desirable to agent i and consistent to agent i's observation. Thus, $D_i\varphi$ intuitively means that agent i's goal implies that φ holds.

For those agents with perfect recall and a global clock, we may use $\sim_i^{spr}$ instead of $\sim_i$ to interpret those formals with modalities B_i, D_i and I_i and get an alternative satisfaction relationship $\models_{BDI}^{spr}$.

Proposition 2 *The following formulas are valid with respect to both $\models_{BDI}$ and $\models_{BDI}^{spr}$:*

- $X(\varphi \Rightarrow \psi) \Rightarrow (X\varphi \Rightarrow X\psi)$
  $X\varphi \Rightarrow YX\varphi$
  $\neg X\varphi \Rightarrow Y\neg X\varphi$
  where X and Y stand for B_i, D_i or I_i (for the same i).
- *Relationship between belief and desire*
  $B_i\varphi \Rightarrow D_i\varphi$
- *Temporal operators*
  $\bigcirc(\varphi \Rightarrow \psi) \Rightarrow (\bigcirc\varphi \Rightarrow \bigcirc\psi)$
  $\bigcirc(\neg\varphi) \Rightarrow \neg \bigcirc \varphi$
  $\varphi \mathbf{U} \psi \Leftrightarrow \psi \vee (\varphi \wedge \bigcirc(\varphi \mathbf{U} \psi))$

Proposition 3 *The following formulas are valid with respect to $\models_{BDI}^{spr}$: $X \bigcirc \varphi \Rightarrow \bigcirc X\varphi$, where X stands for any modality of B_i, D_i and I_i $(i = 1, \cdots, n)$.*

The formula $D_i \bigcirc \varphi \Rightarrow \bigcirc D_i\varphi$ says that if agent i's current goal implies φ holds next time, then at the next time her goal will imply φ, in other words, agent i persists on her goal.

Axiomatization

We give a proof system, called OBDI proof system, for those BDI-agents with perfect recall and a global clock, which contains the axioms of propositional calculus plus those formulas in Propositions 2 and 3. The proof system is closed under the propositional inference rules plus: $\frac{\vdash \varphi}{\vdash D_i\varphi}$ and $\frac{\vdash \varphi}{\vdash I_i\varphi}$ for every agent i.

Theorem 4 *The OBDI proof system for agents with perfect recall and a global clock is sound and complete with respect to interpreted BDI-systems with satisfaction relation $\models_{BDI}^{spr}$.*

Proof: The soundness part is easy and the completeness can follow the similar lines as (Halpern, van der Meyden, & Vardi 2004). ∎

Rational Interpreted BDI-Systems

The drawback of the presented approach is that it is possible for an agent in our BDI-model to have inconsistent beliefs, goals, and intentions. This is because each agent has only single source of belief, single goal and single intention, which may conflict to the agent's observations. To make our model more rational, we assume that agents have multiple sources of beliefs, multiple goals and multiple intentions.

Formally, given a set G of global states and system $\mathcal{K}$ over G, agent i's *BDI-state* over system $\mathcal{K}$ is defined to be a tuple $\overline{\mathcal{M}}_i = \langle \mathcal{B}_i^1, \cdots, \mathcal{B}_i^{k_1}, \mathcal{D}_i^1, \cdots, \mathcal{D}_i^{k_2}, \mathcal{I}_i^1, \cdots, \mathcal{I}_i^{k_3} \rangle$, where $\mathcal{B}_i^j$, $\mathcal{D}_i^j$ and $\mathcal{I}_i^j$ are subsets of $\mathcal{K}$. The structure $\overline{\mathcal{S}} = \langle \mathcal{K}, \overline{\mathcal{M}}_1, \cdots, \overline{\mathcal{M}}_n \rangle$ is said to be a *rational BDI-system*. And similarly, given a valuation function π, we call the pair $\overline{I} = (\overline{\mathcal{S}}, \pi)$ a *rational interpreted BDI-system*.

We may interpret OBDI logic formulas in terms of rational interpreted BDI-systems as follows. Denote the satisfaction relation by $\models_{RBDI}$. Given the rational interpreted BDI-system $\overline{I}$ as above, run r and number u, let j_1 be the least number j such that there is a point (r', v) with $(r, u) \sim_i (r', v)$ and $r' \in \mathcal{B}_i^j$. Let j_2 be the least number j such that (1) $\mathcal{D}_i^j \subseteq \mathcal{B}_i^{j_1}$; and (2) there is a point (r', v) and number j' with $(r, u) \sim_i (r', v)$ and $r' \in \mathcal{D}_i^j \cap \mathcal{I}_i^{j'}$. Let j_3 be the least number j such that there is a point (r', v) with $(r, u) \sim_i (r', v)$ and $r' \in \mathcal{D}_i^{j_2} \cap \mathcal{I}_i^j$. Intuitively, agent i first determines her source of belief (the j_1th one). Then, agent i chooses her goal j_2. In order to achieve goal j_2 , agent i acts by the intention j_3. Thus, we define

- $(\overline{I}, r, u) \models_{RBDI} B_i\varphi$ iff $(\overline{I}, r', v) \models_{RBDI} \varphi$ for those (r', v) such that $r' \in \mathcal{B}_i^{j_1}$ and $(r, u) \sim_i (r', v)$;
- $(\overline{I}, r, u) \models_{RBDI} D_i\varphi$ iff $(\overline{I}, r', v) \models_{RBDI} \varphi$ for those (r', v) such that $r' \in \mathcal{D}_i^{j_2}$ and $(r, u) \sim_i (r', v)$;
- $(\overline{I}, r, u) \models_{RBDI} I_i\varphi$ iff $(\overline{I}, r', v) \models_{RBDI} \varphi$ for those (r', v) such that $r' \in \mathcal{I}_i^{j_3}$ and $(r, u) \sim_i (r', v)$.

Under some constraints, the rational interpreted BDI-system model results in more rational properties of BDI-agents (For example, agents could update or change her belief by her observations and make her belief consistent).

Model Checking BDI-Agents

The model checking problem for our OBDI logic is quite simple: given an interpreted BDI-system I and an OBDI formula φ, determine whether or not $I \models_{BDI} \varphi$.

In order to make our model checking algorithm practically useful, we must consider where our model, an interpreted BDI-system comes from. It would be satisfying if we can derive our model from a program implemented in, say C or Java. However, to make the things simpler, we may consider some abstract programs such as *finite-state programs*, which is expressive enough from the standpoint of theoretical computer science. Moreover, to make our model checking system practically efficient, we use the symbolic model checking techniques. Thus, a finite-state program in our approach is represented in a symbolic way.

Symbolic representation of interpreted BDI-agents

A system as a set of infinite runs is not well suited to model checking directly as it is generally applied to the finite state systems. However, we can represent an interpreted system as a *finite-state program* (G, G_0, R, V), where G is a set of states, G_0 a set of initial states, and R a total "next time" relation, and V associates each state with a truth assignment function. A set of infinite runs is then obtained by "unwinding" the relation R starting from initial states in G_0.

More significantly, we can present a symbolic representation of a finite-state program (G, G_0, R, V) as follows.

1. We use a tuple of boolean variables $\mathbf{x} = \{x_1, \cdots, x_k\}$ and encode a state as an assignment for $\mathbf{x}$, or a subset of $\mathbf{x}$. (For convenience, we do not distinguish a set and its characteristic function.) Thus, G_0 and any set of states can be represented as a propositional formula over $\mathbf{x}$.
2. Another tuple of boolean variables $\mathbf{x}' = \{x'_1, \cdots, x'_k\}$ is used to represent the "next time" relation R between two states as a propositional formula τ over $\mathbf{x} \cup \mathbf{x}'$. Specifically, for two assignments s and s' for $\mathbf{x}$, sRs' holds iff $\tau(\mathbf{x}, \mathbf{x}')$ is satisfied by the assignment $s \cup N(s')$, where $N(s')$ denotes $\{x'_j \mid x_j \in s' \text{ and } 0 < j \leq k\}$.
3. We assume that for each s, $V(s)$ equals s, that is, for each variable x_j $(1 \leq j \leq k)$, $V(s)(x_j) = 1$ iff $s(x_j) = 1$.

Omitting the component V, we represent the finite-state program (G, G_0, R, V) just as $(\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'))$.

Hence, we formally define a (symbolic) *finite-state program with n agents* as a tuple $\mathcal{P} = (\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \cdots, O_n)$, where

1. $\mathbf{x}$ is a set of system variables;
2. θ is a boolean formula over $\mathbf{x}$, called the *initial condition*;
3. τ is a boolean formula over $\mathbf{x} \cup \mathbf{x}'$, called the *transition relation*; and
4. for each i, $O_i \subseteq \mathbf{x}$, containing agent i's *local variables*, or *observable variables*.

Given a state s, we define agent i's local state at state s to be $s \cap O_i$. For convenience, we denote $(s, s \cap O_1, \cdots, s \cap O_n)$ by $g(s)$. We associate with $\mathcal{P}$ the interpreted system $I_{\mathcal{P}} = (\mathcal{R}, \pi)$, where $\mathcal{R}$ is a set of those runs r satisfying that

1. for each m, $r(m)$ is of the form $g(s) = (s, s \cap O_1, \cdots, s \cap O_n)$ where s is a state in $\mathcal{P}$ and the assignment $\pi(s)$ is the same as s;
2. $r(0)$ is $g(s)$ for some assignment s that satisfies θ;
3. for each natural number m, if $r(m) = g(s)$ and $r(m+1) = g(s')$ for some assignments s and s' for $\mathbf{x}$, then $s \cup N(s')$ is an assignment satisfying $\tau(\mathbf{x}, \mathbf{x}')$.

The interpreted system $I_{\mathcal{P}}$ is called *the generated interpreted system of* $\mathcal{P}$.

For convenience, we may use $\mathcal{P}(\theta, \tau)$ to denote a finite-state program with n agents $(\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \cdots, O_n)$, if $\mathbf{x}$ and $O_1, \cdots, O_n$ are clear from the context.

Given a finite-state program $\mathcal{P}(\theta, \tau)$ with n agents , we define an *agent's internal program* over $\mathcal{P}(\theta, \tau)$ as a tuple $\langle \mathcal{P}(\theta_1, \tau_1), \mathcal{P}(\theta_2, \tau_2), \mathcal{P}(\theta_3, \tau_3) \rangle$, where, for $j = 1, 2, 3$, $(\theta_j \Rightarrow \theta) \wedge \tau_j \Rightarrow \tau$ is valid, and $(\theta_2 \Rightarrow \theta_1) \wedge \tau_2 \Rightarrow \tau_1$ is valid.

Clearly, an agent's internal program is exactly related with an agent's mental state. Thus, we define a (symbolic) *BDI-program with n agents* as a tuple $P_A = (\mathcal{P}_K, P_1, \cdots, P_n)$, where $\mathcal{P}_K$ is a finite-state program with n agents and for each agent i, P_i is agent i's internal program over $\mathcal{P}_K$. We use I_{P_A} to denote the corresponding interpreted BDI-system.

Model checking OBDI logic

We need to introduce some notations concerning building boolean formulas. Given a propositional variable p, and a formula φ, we use $\exists p\varphi$ and $\forall p\varphi$ to denote $\varphi[\mathbf{true}/p] \vee \varphi[\mathbf{false}/p]$ and $\varphi[\mathbf{true}/p] \wedge \varphi[\mathbf{false}/p]$, respectively, where $\mathbf{true}$ is some valid boolean formula and **false** is the negation of $\mathbf{true}$. For a set of boolean variables $\mathbf{v} = \{v_1, \cdots, v_m\}$, $\exists \mathbf{v}\varphi$ $(\forall \mathbf{v}\varphi)$ stand for $\exists v_1 \cdots \exists v_m \varphi$ $(\forall v_1 \cdots \forall v_m \varphi)$.

Let ξ be an operator from the set of boolean formulas over $\mathbf{x}$ to the set of boolean formulas over $\mathbf{x}$. We say ψ is a *fixed point* of ξ, if $\models \xi(\psi) \Leftrightarrow \psi$. We say a formula ψ_0 is a *greatest fixed point* of ξ, if ψ_0 is a fixed point of ξ and for every fixed point ψ of ξ, we have that $\models \psi \Rightarrow \psi_0$. Clearly, any two greatest fixed points are logically equivalent to each other. Thus, we denote a greatest fixed point of ξ by $\mathbf{gfp}Z\xi(Z)$. Similarly, we say a ψ_0 is a *least fixed point* of ξ, if ψ_0 is a fixed point of ξ and for every fixed point ψ of ξ, we have that $\models \psi_0 \Rightarrow \psi$. A least fixed point of ξ is denoted by $\mathbf{lfp}Z\xi(Z)$.

To prove the main result in this section, we need the following two lemmas:

Lemma 5 *Given boolean formula ψ over $\mathbf{x}$ and a subset V of $\mathbf{x}$, an assignment (subset) s of $\mathbf{x}$, suppose that, for every $s' \models \psi$, we have $s \cap V \neq s' \cap V$, then $s \models \forall(\mathbf{x} - V)(\neg\psi)$.*

The proof of the above lemma is easy in that it concerns only boolean formulas and their assignments.

Given a finite-state program $\mathcal{P}(\theta, \tau)$ with n agents, let $G(\mathcal{P}(\theta, \tau)) = \mathbf{lfp}Z\,[\theta(\mathbf{x}) \vee (\exists \mathbf{x}(Z \wedge \tau(\mathbf{x}, \mathbf{x}')))[\mathbf{x}/\mathbf{x}']]$. The boolean formula $G(\mathcal{P}(\theta, \tau))$ expresses the set of *reachable global states* .

The next lemma is a variation of Proposition 10 in (Su 2004), whose validity can be seen from the proof of that proposition.

Lemma 6 *Given a finite-state program with n agents $(\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \cdots, O_n)$ and an LTL formula φ, Then, $I_{\mathcal{P}} \models_{KL_n} K_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta, \tau)) \Rightarrow \Gamma(\varphi, \theta, \tau))$ where $\Gamma(\varphi, \theta, \tau)$ is a boolean formula built from φ, θ, τ by using quantifications and fixed-point operations* **lft** *and* **gft**.

Notice that some new propositional variables may be introduced in $\Gamma(\varphi, \theta, \tau)$, but $\Gamma(\varphi, \theta, \tau)$ has no free propositional variables other than $\mathbf{x}$. We omit the details of formula $\Gamma(\varphi, \theta, \tau)$, but what is significant is that $\Gamma(\varphi, \theta, \tau)$ is an expression built by boolean operators and quantifications over propositional variables as well as fixed-point operators. Such expressions can be conveniently dealt with by OBDD as done in main stream computer science.

Theorem 7 *Given a BDI-program with n agents $P_A = (\mathcal{P}_K, P_1, \cdots, P_n)$, suppose that $\mathcal{P}_K = (\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \cdots, O_n)$, and for every i, $P_i = \langle \mathcal{P}(\theta_1^i, \tau_1^i), \mathcal{P}(\theta_2^i, \tau_2^i), \mathcal{P}(\theta_3^i, \tau_3^i) \rangle$. Then, for each LTL formula φ and agent i, the following are valid in $I_{\mathcal{P}_A}$,*

1. $B_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_1^i, \tau_1^i)) \Rightarrow \Gamma(\varphi, \theta_1^i, \tau_1^i))$
2. $D_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_2^i, \tau_2^i)) \Rightarrow \Gamma(\varphi, \theta_2^i, \tau_2^i))$
3. $I_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_3^i, \tau_3^i)) \Rightarrow \Gamma(\varphi, \theta_3^i, \tau_3^i))$

Proof: We prove only the first one, the other two can be obtained in a similar way. Let r be a run in the interpreted system $I_{\mathcal{P}_K}$, m a natural number. To show the formula

$B_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_1^i, \tau_1^i)) \Rightarrow \Gamma(\varphi, \theta_1^i, \tau_1^i))$

is valid in $(I_{\mathcal{P}_A}, r, m)$, we first consider the case that for every point (r', m') in the interpreted system $I_{\mathcal{P}(\theta_1^i, \tau_1^i)}$, we have $r(m) \cap O_i \neq r'(m') \cap O_i$, i.e., $r_i(m) \neq r_i'(m')$. In this case, we have that $I_{\mathcal{P}_A}, r, m \models_{BDI} B_i\varphi$. On the other hand, for every assignment s' over $\mathbf{x}$ with $s' \models G(\mathcal{P}(\theta_1^i, \tau_1^i))$, there is a point (r', m') in $I_{\mathcal{P}(\theta_1^i, \tau_1^i)}$ with $s' = r'(m')$, and we thus have that $r(m) \cap O_i \neq s' \cap O_i$. Therefore, we have that $r(m) \models \forall(\mathbf{x} - O_i)(\neg G(\mathcal{P}(\theta_1^i, \tau_1^i))$ (by Lemma 5), and hence $I_{\mathcal{P}_A}, r, m \models_{BDI} \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_1^i, \tau_1^i)) \Rightarrow \Gamma(\varphi, \theta_1^i, \tau_1^i))$ holds. Thus, the claim holds in this case.

Suppose that for some point (r', m') in the interpreted system $I_{\mathcal{P}(\theta_1^i, \tau_1^i)}$, we have $r(m) \cap O_i = r'(m') \cap O_i$, i.e., $r_i(m) = r_i'(m')$. Then, it is clear that $\forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_1^i, \tau_1^i)) \Rightarrow \Gamma(\varphi, \theta_1^i, \tau_1^i))$ has the same truth value in $(I_{\mathcal{P}_A}, r, m)$ and in $(I_{\mathcal{P}_A}, r', m')$. By the semantics definition, we have that $B_i\varphi$ has the same truth value in $(I_{\mathcal{P}_A}, r, m)$ and in $(I_{\mathcal{P}_A}, r', m')$. Thus, we only need to show that the validity of $B_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_1^i, \tau_1^i)) \Rightarrow \Gamma(\varphi, \theta_1^i, \tau_1^i))$ in $(I_{\mathcal{P}_A}, r', m')$, which is equivalent to that

$$I_{\mathcal{P}(\theta_1^i, \tau_1^i)}, r', m' \models_{KL_n} K_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i) \begin{pmatrix} G(\mathcal{P}(\theta_1^i, \tau_1^i)) \\ \Rightarrow \Gamma(\varphi, \theta_1^i, \tau_1^i) \end{pmatrix}$$

because r' is a run in the interpreted system $I_{\mathcal{P}(\theta_1^i, \tau_1^i)}$. Thus, the claim holds in this case also. ∎

Experimental Results

Theorem 7 provides a reduction of OBDI to LTL, while Proposition 9 in (Su 2004) gives an OBDD-based method of model checking LTL formulas. The complexity of our reduction of logic OBDI to LTL is $PSPACE$-complete. However, because the quantification of Boolean functions and fixed-point operators can be dealt with by any OBDD package, the reduction can be based on OBDDs. Thus, the OBDI model checking algorithm with the above results can be implemented with considerable efficiency.

In this study, we implemented a prototype of the OBDI model checker using CUDD library developed by Fabio Somenzi, and obtained preliminary experimental results.

The scenario we use here involves two robots that are cleaning 9 squares in a `3x3` square, some of which are dirty. We assume that Robot `r1` has initial belief about which squares are dirty. Robot `r1` searches for dirty squares according to her belief; and when one is found, the robot calls robot `r2` to go to the location of the dirty square and then they clean the dirty square together. After cleaning the dirty square, `r2` remains at the same location, and `r1` looks for another dirty square.

As our system is OBDD based, we describe interpreted systems (basic elements in our BDI-model) by the language similar to the input language of the very influential model checking tool NuSMV based on OBDD.

The environment consists of a Boolean array `d`[1..3][1..3] indicating the dirty squares and a Boolean variable `calling`, which expresses that `r1` is calling `r2`.

An agent is defined as a module, which consists of a set of observable variables and three submodules for the agent's beliefs, desires and intentions, respectively. The observable variables includes all of the variables defined in the module and the formal parameters explicitly labelled observable (e.g., one agent's observable parameters includes the other's position and `calling`). Each submodule includes a set of triggering events and its corresponding actions, and shares the observable variables and the initial beliefs of its main module. An action will be executed once its event is triggered and the *context* for the action is satisfied based on the agent's current beliefs. We omit the modelling details here.

Assume that the initial state is $\mathtt{d}[1][3] \wedge \neg \mathtt{d}[2][1] \wedge \mathtt{d}[3][3]$ and agent `r1` initially believes that positions $(1,3)$, $(2,1)$ and $(3,3)$ are dirty. The model checker verified the following specifications by using a PC with CPU P4 2.4G and memory 512M.

1. $\mathbf{G}((B_1\mathtt{d}[3][3]) \Rightarrow \mathbf{F}(\neg \mathtt{d}[3][3]) \wedge \mathtt{pos}(\mathtt{r1}, 3, 3) \wedge \mathtt{pos}(\mathtt{r2}, 3, 3))$
   Result: `true`, BDD nodes allocated: `313240`, Time: `0.68s`.
2. $\mathbf{G}((\mathtt{pos}(\mathtt{r1}, 3, 3) \wedge \mathtt{clean}(\mathtt{r1}, 3, 3)) => D_1(\mathbf{F}\mathtt{continue}))$
   Result: `true`, BDD nodes allocated: `357293`, Time: `1.04s`.
3. $\mathbf{F}(I_1\mathtt{goto}(\mathtt{r1}, 2, 1) \wedge \mathbf{F}\ \mathtt{clean}(\mathtt{r1}, 2, 1))$
   Result: `false`, BDD nodes allocated: `288275`, Time: `0.60s`.

where $\mathbf{F}\varphi \equiv true\mathbf{U}\varphi$ and $\mathbf{G}\varphi \equiv \neg\mathbf{F}\neg\varphi$. The first specification says that in any situation, if `r1` believes that $(3,3)$ is dirty, then `r1` and `r2` will go there and clean it eventually. The second indicates that in any situation, if `r1` is at square $(3,3)$ and will clean it next time, then she desires to continue the search. The third means that `r1` will eventually clean square $(2,1)$ once she intends to go there. It is false, because `r1` need not clean $(2,1)$ after she detects that it is actually clean.

Related Work

Computationally grounded logics: Besides epistemic logic with the interpreted system model, another computa-

tionally grounded logic is $\mathcal{VSK}$ logic (Wooldridge & Lomuscio 2001). However, belief, desire or intention of an agent is not considered in $\mathcal{VSK}$; the technical difficulty behind this seems that modalities for those notions do not satisfy S5 axioms. Thus, it is interesting to formalize the internal attitudes of an agent by extending the interpreted system model.

Alternating-time temporal logics: Alternating-time temporal logic (ATL) and its extensions (ATEL, for example) were proposed recently to tackle the verification of multi-agent system properties (Alur, Henzinger, & Kupferman 2002; van der Hoek & Wooldridge 2002). Using ATL one can express that a coalition of agents can achieve some properties. For example, assume that p and q are local variables of agents 1 and 2, respectively; a coalition of agents 1 and 2 can achieve $p \Leftrightarrow q$. Nevertheless, there is no practical strategy to guarantee that $p \Leftrightarrow q$, because agent 1 can only observe and change the value of p and so can agent 2 for variable q. It is not very convenient to use ATL to deal with incomplete information or agents' local observations, as the model checking problem of ATL with incomplete information is generally undecidable (Alur, Henzinger, & Kupferman 2002). Moreover, ATL and its extensions do not deal with agents' mental state such as belief and desire.

Model checking BDI-agents: Model checking multi-agents systems has become an active topic in the community of multi-agents systems. Many efforts have been devoted to model checking knowledge in multi-agents systems (Hoek & Wooldridge 2002; van der Meyden & Su 2004; Su 2004; Raimondi & Lomuscio 2004). However, comparatively little work has been carried out on model checking BDI-agents. Some general approaches to model checking BDI-agents were proposed in (Rao & Georgeff 1993; Benerecetti & Cimatti 2002), but no method was given for generating models from actual systems, and so the techniques given there could not easily be applied to verifying real multi-agent systems. In (Bordini *et al.* 2003), model checking techniques for AgentSpeak(L) (Rao 1996) were reported; however, AgentSpeak(L) has very simple semantics. The salient point of our work is that we present a general form of BDI agent program, from which BDI agent models are generated and specifications in full BDI logics can be verified by symbolic model checking techniques.

Concluding Remarks

We have proposed a new BDI-model by using interpreted systems, and developed a computationally grounded BDI logic, called OBDI logic. We interpret OBDI logic via two different semantics. One of them is based on the assumption that agents have perfect recalls and a global clock, with respect to which a sound and complete proof system has been presented. We have explored how symbolic model checking techniques can be applied to model checking BDI-agents and got some preliminary experimental results. As for future work, we plan to further investigate rational interpreted BDI-systems and to link to known axiomatizations of belief, desire, and intention.

References

Alur, R.; Henzinger, T.; and Kupferman, O. 2002. Alternating-time temporal logic. *Journal of the ACM* 49:672–713.

Benerecetti, M., and Cimatti, A. 2002. Symbolic model checking for multi-agent systems. In *Proc. MoChart-02*, 1–8.

Bordini, R.; Fisher, M.; Pardavila, C.; and Wooldridge, M. 2003. Model checking agentspeak. In *Proc. AAMAS-03*, 14–18.

Bryant, R. 1986. Graph-based algorithms for boolean function manipulation. *IEEE Transaction on Computers* (C-35(8)).

Cohen, P., and Levesque, H. 1990. Intension is choice with commitment. *Artificial Intelligence* 42:23–261.

Fagin, R.; Halpern, J.; Moses, Y.; and Vardi, M. 1995. *Reasoning about knowledge*. Cambridge, MA: MIT Press.

Halpern, J., and Zuck, L. 1992. A little knowledge goes a long way: Simple knowledge based derivations and correctness proofs for a family of protocols. *Journal of the ACM* 39(3):449–478.

Halpern, J.; van der Meyden, R.; and Vardi, M. Y. 2004. Complete axiomatizations for reasoning about knowledge and time. *SIAM Journal of Computing* 33(3):647–703.

Hindriks, K.; de Boer, F.; van der Hock, W.; and Meyer, J.-J. C. 1999. Agent programming in 3apl. *Autonomous Agents and Multi-Agent Systems* 2(4):357–402.

Hoek, W. v. d., and Wooldridge, M. 2002. Model checking knowledge and time. In *9th Workshop on SPIN (Model Checking Software)*.

Kripke, S. 1963. A semantical analysis of modal logic. i: Normal modal propositional calculi. *Z. Math. Logik Grundl. Math.* 9:67–96.

Raimondi, F., and Lomuscio, A. 2004. Verification of multiagent system via ordered binary decision diagrams: an algorithm and its implementation. In *Proc. of AAMAS-04*.

Rao, A., and Georgeff, M. 1993. A model theoretic approach to the verification of situated reasoning systems. In *Proc. 13th International Joint Conference on Artificial Intelligence*, 318–324.

Rao, A., and Georgeff, M. 1998. Decision procedures for BDI logics. *Journal of Logic and Computation* 8(3):293–344.

Rao, A. 1996. Bdi agent speak out in a logical computable language. In *LNAI*, volume 1038, 42–55.

Schild, K. 2000. On the relationship between BDI logics and standard logics of concurrency. *Autonomous Agents and Multi-Agent Systems* 3:259–283.

Shoham, Y. 1993. Agent oriented programming. *Artificial Intelligence* 60(1):51–92.

Su, K. 2004. Model checking temporal logics of knowledge in distributed systems. In *AAAI-04*, 98–103. AAAI.

van der Hoek, W., and Wooldridge, M. 2002. Tractable multi-agent planning for epistemic goals. In *Proc. AAMAS-02*, 1167–1174.

van der Hoek, W., and Wooldridge, M. 2003. Towards a logic of rational agency. *L.J. of IGPL* 11(2):135–159.

van der Meyden, R., and Su, K. 2004. Symbolic model checking the knowledge of the dining cryptographers. In *Proc. of IEEE CSFW-04*, 280–291.

Wooldridge, M., and Lomuscio, A. 2001. A computationally grounded logic of visibility, perception, and knowledge. *Logic Journal of the IGPL* 9(2):273–288.

Wooldridge, M. 2000. Computationally grounded theories of agency. In Durfee, E., ed., *ICMAS-00*, 13–22. IEEE Press.