Expert Systems with Applications 37 (2010) 8027–8035

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Goals description and application in migrating workflow system

Wu Xiuguo [a,b,*], Zeng Guangzhou [a]

[a] School of Computer Science and Technology, Shandong University, 250061 Jinan, China
[b] School of Information Management, Shandong Economics University, 250014 Jinan, China

ARTICLE INFO

Keywords:
Intelligent agent
Goal representation
Goal description logics (*GDLs*)
Migrating workflow system (*MWfS*)

ABSTRACT

Goals, which can be described as states that an agent would like to realize, is an important concept for intelligent agent systems. The representation of goals and the ability to reason about them are the major problems in goal-oriented analysis and modeling techniques, especially in intelligent agent system, as goals are more stable than other abstractions (e.g. user stories). Description Logics (DLs) is a formal tool of knowledge representation and reasoning. In this paper, we construct a framework with explicit representation and formal semantics of goals-Goal Description Logics (*GDLs*), which integrates two aspects of goals: declarative (a description of the state of sought), and procedural (a set of plans for achieving the goal), into one concept based on Description Logics (DLs). In addition, goals reasoning, especially goal matchmaking in *GDLs*, is studied using its effective judgment to concept subsumption. We propose a conceptual model of goal-based migration workflow system (*GMWfS*) based on *GDLs*, and illustrate an application. We also present preliminary experimental results on an implementation of these ideas. Compared to traditional workflow methods, *GOMWfS* is more flexible and intelligent.

1. Introduction

Intelligent agent, which has the ability to operate in dynamic and complex domains, has not only become one of the central topics of artificial intelligence, but also the mainstream of computer science. Many researchers take up agent-oriented programming as a new and exciting paradigm to investigate, and has significant applications in a wide range of domains (Bratman, 1987; Rao & Georgeff, 1995). Although there has been much debate on what constitutes an agent, and which features are important, the consensus is that an intelligent agent is situated, autonomous, reactive, proactive, and social (Wooldridge, 1998).

Goals, is an important concept in intelligent agent systems, which exhibit proactive behaviors. In the context of this paper, a *goal* is a state of the world that an agent wants to achieve. On this view, the explicit representation of goals and the ability to reason about them play an important role in agent systems.

Several works have been and still are concerned with defining appropriate frameworks for advanced, intelligent, and automated problem solving using goal-driven architecture. Hindriks et al. developed an agent programming language called GOAL (Goal Oriented Agent Language) (Hindriks, de boer, van der Hoek, & Meyer, 2000). They provided a logical formalism, a programming language, and a set of formal semantics that relates the logical formalism to the programming language. In contrast to other attempts (Soham, 1993; Wobcke, 2000), it bridges the gap between logical formalisms and a programming language that realizes the logic. Their logic requires that goals not be entailed by beliefs (i.e. they are not already achieved) and that goals be satisfiable. In PRACTIONIS, a goal is considered as an analysis, a design, and an implementation abstraction compliant to the semantics described (Morreale et al., 2006). In other words, PRACTIONIST agents can be programmed in terms of goals, which then will be related to either desires or intentions according to whether some specific conditions are satisfied or not. van Riemsdijk et al. attempt to incorporate both the declarative and procedural aspect of goals into their *Dribble* programming language (van Riemsdijk, van der Hoek, & Meyer, 2003). They represent goals and beliefs as propositional formulae, and provide operational semantics for mental state changes of the agent and the execution of goals. Inspired by *Dribble*, Dastani et al. extended the agent programming language 3APL (An Abstract Agent Programming Language) to include the declarative aspect of goals (Dastani, van Riemsdijk, Dignum, & Meyer, 2003; Thangarajah, 2004). In the extended 3APL, goals represent situations that the agent wants to realize, and plans are introduced as procedures to allow the agent to satisfy these goals.

Systems like Agent-0 (Soham, 1993), AgentSpeak(L) (Rao, 1996), and JACK (Busetta, Rönnquist, Hodgson, & Lucas, 1998), do not represent goals explicitly, but capture goals implicitly. In conclusion, all these notions are structures that built from the actions. Therefore they are similar in nature to plans.

* Corresponding author at: School of Computer Science and Technology, Shandong University, 250061 Jinan, China. Tel.: +1 306 400 2238.
E-mail address: xiuguosd@163.com (X. Wu).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.05.087

Description Logics (*DLs*) are a family of knowledge representation formalisms. They are based on the notion of concepts and roles, and are mainly characterized by constructors that allow complex concepts and roles to be built from atomic ones (Horrocks & Sattler, 1999; Baader, Calvanese, Mc Guinness, Nardi, & Patel-Schneider, 2002). The main benefit from these knowledge languages is that sound and complete algorithms for the subsumption and satisfiability problems can be defined. A *DLs* reasoner solves the problems of equivalence, satisfiability and subsumption.

A key difference between *DLs* and the standard representation formalisms based on First-Order Logic, e.g. relational and deductive databases, is that *DLs* provide an arena for exploring new sets of "logical connectives" – the constructors used to form composite descriptions – that are different from the standard connectives such as conjunction, universal quantifiers, etc. Therefore, *DLs* provide a new space in which to search for expressive and effectively computable representation languages. Moreover, although it is possible to translate many aspects of *DLs* currently encountered into First-Order Logic, reasoning with such a translation would be a very poor substitute because *DLs*-based systems reason in a way that does not resemble standard theorem proving (e.g. by making use of imperative programming features).

We have presented a novel way to develop a representation of goals based on Description Logics (*DLs*), which are able to represent structural knowledge in a formal and well-understood way (Xiuguo & Tongtong, 2008; Xiuguo, Guangzhou, & Gongping, 2008). In those papers, our aim is to allow agent implementation platforms to be more faithful to their theoretical foundations, and to provide a better handling of goals.

Based on above reasons, we construct a framework with explicit representation and formal semantics of goals-Goal Description Logics (*GDLs*), which integrates two types of goals: declarative goals and procedural goals into one concept, using effective representation and reasoning capability of Description Logics (*DLs*), especially its effective judgment to concept subsumption. The main contributions of this paper include:

- A logic framework to express goals description in intelligent agents, including syntax and semantics;
- Reasoning, especially goal hierarchy construction algorithms in Goal Description Logic (*GDLs*);
- A conceptual model of goal-oriented migration workflow system (*GMWfS*);
- Evaluation of goal-based workflow system and process-based workflow system.

The remainder of the paper is structured as follows: in the next section (Section 2), we define a framework of goals and plans-Goal Description Logics (*GDLs*). In Section 3, we present goal hierarchy algorithms, which is an important reasoning problem in *GDLs*. Then, in Section 4 we propose a goal-based migrating workflow model, and report experiments to show the costs and benefits of goal-based workflow system compared to traditional workflow system. Section 5 reports on background work and the related work, to finally draw some conclusions and propose some further research topics in Section 6.

2. Goal description logics (GDLs)

In order to implement goal representation and reasoning, we propose a formal framework-Goal Description Logics (*GDLs*) based on Description Logics (*DLs*). In this section, we will present the formal syntax and semantics of *GDLs*, and then introduce the main means of goals achievement-goals plan.

Let's suppose *Cons* to be a constant set: $Cons = \{a_1, a_2 \ldots\}$; *Var* is a variable set: $Var = \{x_1, x_2 \ldots\}$ and *Fun* is a function set: $Fun = \{f_1, f_2 \ldots\}$.

2.1. Syntax

Definition 2.1 (*Term*). A *term* in *GDLs* can be defined as follows:

(1) If $x \in Var$, then $x \in Term$;
(2) If $x \in Term$ and $x_1, x_2 \ldots x_n \in Term$, then $f(x_1, x_2 \ldots x_n) \in Term$.

Definition 2.2 (*Concept*).

(1) Primitive concept P, top concept $\top$ and bottom concept $\bot$ are *concepts*;
(2) If C and D are *concepts*, then $\neg C$, $C \sqcap D$ and $C \sqcup D$ are *concepts*;
(3) If R is a *role* name, and C is a *concept* name, then $\exists R.C$ and $\forall R.C$ are *concepts*.

Definition 2.3 (*Formula*).

(1) Assertion formula like $C(a)$, $R(a,b)$, where there is no variables;
(2) Common formula like $C(x)$, $R(x_i,x_j)$, where there is at least one variable;
(3) Assertion formula and common formula are *formulae*;
(4) If φ and ψ are *formulae*, then $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \rightarrow \psi$, are all *formulae*.

Definition 2.4 (*Substitution*). A *substitution* θ is a finite set of pairs as the form $\{a_1/x_1, \ldots, a_n/x_n\}$, where $a_i \in Var$ and $x_j \in Term$, and $\forall i \neq j$: $a_i \neq a_j$, $i, j \in [1 \ldots n]$. A ground substitution θ is a substitution such that there is no variable in x_i.

Definition 2.5 (*Instance Formula*). Let ϕ be a *GDLs* formula, $\theta = \{a_1/x_1, \ldots, a_n/x_n\}$ be a substitution. Then ϕ' is called an *instance formula* where all variables in ϕ are simultaneously replaced with θ.

Definition 2.6 (*Atomic Goal*). An *atomic goal* in *GDLs* can be described as follows: $g(x_1, x_2 \ldots x_n) = (P, S, F)$, where.

(1) g is the atomic goal name. $x_1, x_2 \ldots x_n$ are constants or variables;
(2) P is the pre-condition set which is made up from $C(x_i)$ or $R(x_i,x_j)(1 \leqslant i,j \leqslant n)$. P guarantees that the goal g can be satisfiable. In a general way, we call *P satisfaction conditions*;
(3) S is the state of the world once a goal succeeds. $s \in S$ has the form as B_i/H_i, where B_i is a set of *GDLs* formulae; H_i is a common literal, having the form as$C(x)$, $R(x_i,x_j)$ or $\neg C(x)$, $\neg R(x_i,x_j)$, where C is a concept and R is a role. If H_i is positive, then H_i is added to *ABox A*, else H_i is deleted from *ABox A*;
(4) F is a set of failure conditions which indicates a goal must be dropped with failure. One advantage of allowing F to be specified is that it allows for a range of commitment strategies to be realized by specifying appropriate conditions for a goal to be dropped.
Given an atomic goal, we can get a goal instance by a substitution θ. A ground goal is a goal without any variable. An atomic goal can be regarded as a set of ground goals.

Definition 2.7 (*Goal Formula*).
A *goal formula* (*goal*) can be constituted by:

(1) If g is a n-dimension atomic goal name, $t_1, t_2 \ldots t_n$ are *terms*, then $g(t_1, t_2 \ldots t_n)$ is a *goal formula*, called *atomic goal formula*;