Comparison Between Two Languages Used to Express Planning Goals: $CTL$ and $E_{AGLE}$

Wei Huang$^1$, Zhonghua Wen$^{1,2}$, Yunfei Jiang$^1$, and Aixiang Chen$^1$

$^1$ Software Research Institute, Sun Yat-sen University
Guangzhou, Guangdong 510275, China
huangbodao@yahoo.com.cn

$^2$ College of Information Engineering, Xiangtan University
Xiangtan, Hunan 411105, China
zhonghua@xtu.edu.cn

Abstract. The extended goals in non-deterministic domains are often expressed in temporal logic, particularly in $CTL$ and $E_{AGLE}$. No work has given a formal comparison between $E_{AGLE}$ and $CTL$ on semantics, though it is said that the capability of representing the “intentional” aspects of goals and the possibility of dealing with failure are the main new features of $E_{AGLE}$ w.r.t. $CTL$.

According to the formal semantics for $E_{AGLE}$ and $CTL$, we prove that all the $E_{AGLE}$ formulas in which only $LV_1$ operators (i.e. the operators representing the “intentional” aspects of goals) appear and some $E_{AGLE}$ formulas including $LV_2$ operators (i.e. the operators dealing with failure and qualitative preferences) can be replaced by some $CTL$ formulas without any change on semantics. Finally, we also find some basic and important goals in non-deterministic domains that exceed the expressive ability of $E_{AGLE}$.

1 Introduction

Unlike classical planning [12], planning for extended goals in non–deterministic domains (e.g., robotics, scheduling, and control) is required to generate plans that satisfy conditions on their whole execution paths in order to deal with non–determinism and possible failures. Planning based on Markov Decision Processes, i.e. MDP–based planning [7,8] and planning based on model checking [3,4,5] are two main approaches to planning for extended goals in non-deterministic domains. In the former approach, planning goals are represented by means of utility functions; while in the latter approach, planning goals are expressed by formulas in temporal logic, particularly in $CTL$ [6] and $E_{AGLE}$ [5].

Though it is said that the capability of representing the “intentional” aspects of goals and the possibility of dealing with failure are the main new features of $E_{AGLE}$ w.r.t. $CTL$ in [5], little research has been devoted to the formal comparison between $E_{AGLE}$ and $CTL$ on semantics. In this paper we prove that all the $E_{AGLE}$ formulas in which only $LV_1$ operators (i.e. the operators representing the “intentional” aspects of goals) appear can be replaced by some
Comparison Between Two Languages Used to Express Planning Goals

CTL formulas without any change on semantics, so can some $E_AG_LE$ formulas including $LV_2$ operators (i.e. the operators dealing with failure and qualitative preferences). In addition, we find that some $CTL$ formulas can not be expressed in $E_AG_LE$. Our purpose is not suggesting designers of systems have a choice between $E_AG_LE$ and $CTL$. Actually, $E_AG_LE$ and $CTL$ have different features.

This paper is structured as follows. Section 2 illustrates the basic concepts of planning for extended goals in non-deterministic domains, and presents the formal semantics for $E_AG_LE$ formulas and $CTL$ formulas. The core of the paper, i.e. section 3 gives a formal comparison between $E_AG_LE$ and $CTL$ on semantics, and shows some limitations of the expressive power of $E_AG_LE$ language. Section 4 concludes the paper by presenting some possibilities for further work.

2 Background

In this section, we briefly illustrate some definitions of planning for extended goals in non-deterministic domains [3] that are relevant to our work, and review the semantics for $CTL$ formulas and $E_AG_LE$ formulas over a Kripke structure [3]–[5]. Some examples of these definitions can be found in [3] and [5].

2.1 Planning for Extended Goals in Non-deterministic Domains

Following [3], a non-deterministic planning domain $D$ is a tuple $(B, Q, A, \rightarrow)$, where $B$ is a finite set of basic propositions, $Q \subseteq 2^B$ is the set of states, $A$ is the finite set of actions, and $\rightarrow \subseteq Q \times A \times Q$ is the transition relation. $q \rightarrow a q'$ denotes $(q, a, q') \in \rightarrow$. The relation $\rightarrow$ is required to be total, i.e. for every $q \in Q$ there is some $a \in A$ and $q' \in Q$ such that $q \rightarrow a q'$.

In planning for extended goals, actions specified by plans do not only depend on the current state of the domain. The sequence of states that have appeared in the plan execution (i.e. the execution context) must be taken into account.

**Definition 1.** A plan for a domain $D$ is a tuple $\pi = \langle C, c_0, act, ctxt \rangle$, where $C$ is a set of contexts, $c_0 \in C$ is the initial context, $act : Q \times C \rightarrow A$ is the action function, and $ctxt : Q \times C \times Q \rightarrow C$ is the context function.

A plan $\pi$ is executable if, whenever $act(q, c) = a$ and $ctxt(q, c, q') = c'$, then $(q, a, q') \in \rightarrow$. A plan $\pi$ is complete if, whenever $act(q, c) = a$ and $(q, a, q') \in \rightarrow$, then there is some context $c'$ such that $ctxt(q, c, q') = c'$ and $act(q', c')$ is defined. In the following discussion, we consider only plans that are executable and complete.

In fact, any $c \in C$ represents a class of sequences of states. To each class correspond the already accomplished subgoals.

**Definition 2.** The execution structure of plan $\pi$ in a domain $D$ from state $q_0$ is the structure $K = \langle S, R, L \rangle$, where $S = \{ (q, c) | act(q, c) \text{ is defined} \}$, $R = \{ ((q, c), (q', c')) \in S \times S | q \rightarrow a q' \text{ with } a = act(q, c) \text{ and } c' = ctxt(q, c, q') \}$, $L(q, c) = \{ b | b \in q \}$.