Reasoning about Sensing Actions and Reactivity

Son Cao Tran
Computer Science Department
University of Texas at El Paso
El Paso, Texas 79968
Email: tson@cs.utep.edu

Advisor: Dr. Chitta Baral

This thesis focuses on the problem of reasoning about sensing actions and the relationship between action theory and reactive control. The first approach to reasoning about sensing actions is due to Moore[1]. He used the possible world semantics to represent knowledge and treated the accessible relation between world models as a fluent. Later, Hass proposed another way of formulating sensing actions using first order logic. Moore's formulation was then adapted to reasoning about sensing actions in situation calculus by Scherl and Levesque.

In 1997, Lobo, Taylor, and Mendez extended the language $\mathcal{A}$, a high-level action description language of Gelfond and Lifschitz, to allow sensing actions and called the new language $\mathcal{A}_K$. Lobo et al. defined the semantics of $\mathcal{A}_K$, which will be denoted by $\models_{LTM}$ hereafter, by situation transition functions, which are extensions of the state transition functions of $\mathcal{A}$. Baral and Son proposed different approximations for the semantics of $\mathcal{A}_K$, denoted by $\models_a$ [2]. However, it was not clear whether there is a situation calculus counterpart of $\mathcal{A}_K$, as Kartha proved for $\mathcal{A}$, or not. Another question was to prove the soundness of $\models_a$ with respect to $\models_{LTM}$.

We will present a new approach to reasoning about sensing actions in $\mathcal{A}_K$, in which transition functions are defined over *knowledge states* (or *k-states*). A k-state is a pair $\langle s, \Sigma \rangle$ where s is a state and Σ is a set of states. Intuitively, s represents the real state of the world and Σ represents the possible states of the world in which an agent thinks it may be in. We denote the new semantics by $\models_{\mathcal{A}_K}$ and prove that Kartha's results can be extended to domains with sensing actions. More importantly, we prove the soundness of the different approximations of $\mathcal{A}_K$, $\models_a$, with respect to $\models_{\mathcal{A}_K}$ and $\models_{LTM}$. To compute $\models_{\mathcal{A}_K}$, we translate domain descriptions in $\mathcal{A}_K$ into extended logic programs where the answer set semantics of the latter coincides with $\models_{\mathcal{A}_K}$. It is also expected that our formalism corresponds to POMDP (Partial Observable Markov Decision Process), a well-known statistical approach to reasoning about sensing actions.

Theories of actions can be used to model deliberate agents which achieve their goals by repeatedly (a) making observations, (b) creating a plan, and (c) executing the plan. Since planning in general is NP-complete, deliberate agents cannot react effectively to changes caused by exogenous actions. Reactive agents have been proposed to overcome this weakness of deliberate agents.

There have been many formalisms for reactive agents, for example, Schopper's universal planner or Kaelbling and Rosenschein's situated agent architecture. However, the relationship between reactive control and theories of actions has not been well established. Brooks even argued that perhaps reactive behaviors do not need reasoning and knowledge representation at all.

We define a control module as a set of *condition-action rules* of the form "**if** α **then** γ" where α is a condition and γ is a sequence of actions. Intuitively, a control module could be viewed as a case statement which tells the agent what it needs to do in a given state. We formalize precisely what it means for a control module to achieve a goal with respect to a set of initial states. We also give sufficient conditions for such control modules to achieve (or maintain) a goal with respect to a set of states and a theory of action. More importantly, we develop algorithms that automatically construct a control module that achieves a goal given an action theory, a set of initial states, and the goal. These algorithms are not planners, rather they are built upon planners.

We expect to have an implemenation of the aforementioned algorithms when the thesis is done. Also, we would like to improve our robot control software[3] by developing new planning and scheduling modules which will allow users to specify dynamic domains in a high-level action description language and write robot programs. To achieve a goal means then to ask the robot to execute a corresponding program. For this purpose, we intend to use ConGolog, a logic programming language[4] that offers a number of desirable features for modeling dynamic domains such as reactivity, concurrency, and non-determinism. We will add to ConGolog a new construct which can simplify ConGolog-programming in domains where partial order between actions and temporal constraints are important.

[1] Precise references are omitted due to space constraint.

[2] a is 0, 1, or ∞.

[3] We participated in the *"Scheduling a meeting"* and *"Tidy Up"* in AAAI'96 and AAAI'97 robot competitions and won the third and first prize respectively.

[4] Developed by the Cognitive Robotics group at the University of Toronto