Combining Model-Based Meta-Reasoning and Reinforcement Learning For Adapting Game-Playing Agents

Patrick Ulam, Joshua Jones, and Ashok Goel
College of Computing, Georgia Institute of Technology
Atlanta, USA 30332

Abstract

Human experience with interactive games will be enhanced if the software agents that play the game learn from their failures. Techniques such as reinforcement learning provide one way in which these agents may learn from their failures. Model-based meta-reasoning, a technique in which an agent uses a self-model for blame assignment, provides another. This paper evaluates a framework in which both these approaches are combined. We describe an experimental investigation of a specific task (defending a city) in a computer war strategy game called FreeCiv. Our results indicate that in the task examined, model-based meta-reasoning coupled with reinforcement learning enables the agent to learn the task with performance matching that of an expert designed agent and with speed exceeding that of a pure reinforcement learning agent.

Introduction

Intelligent agents acting as non-player characters (NPC) in interactive games often fail in their tasks. However, NPC's in most commercially available interactive games generally do not learn from these failures. As a result, the human player may tire of playing the game. The human player's experience with an interactive game surely will be enhanced if these agents learned from their failures and minimize their recurrence.

Meta-reasoning provides one method for learning from failures. In *model-based meta-reasoning*, an agent is endowed with a self-model, i.e., a model of its own knowledge and reasoning. When the agent fails to accomplish a given task, the agent uses its self-model, possibly in conjunction with traces of its reasoning on the task, to assign blame for the failure(s) and modify its knowledge and reasoning accordingly. Such techniques have been used in domains ranging from game playing (B. Krulwich and Collins 1992)(Ulam, Goel, and Jones 2004), to route planning (Fox and Leake 1995) and assembly planning (Murdock and Goel 2003).

However, (Murdock and Goel 2008) showed in some cases model-based meta-reasoning can only *localize* the causes for its failures to specific portions of its task structure, but not necessarily *identify* the precise causes or the modifications needed to address them. They used reinforcement learning (RL) to complete the partial solutions generated by meta-reasoning: first, the agent used its self-model to localize the needed modifications to specific portions of its task structure, and then used Q-learning within those parts to identify the necessary modifications.

In this work, instead of using reinforcement learning to identify the modifications necessary in a task model, we evaluate the hypothesis that model-based meta-reasoning may also be used to identify the appropriate RL space for a specific task. The learning space represented by combinations of all possible modifications to an agent's reasoning and knowledge can be too large for RL to work efficiently. One way in which this complexity can be addressed is through the decomposition of the learning problem into a series of smaller sub-problems (e.g. (Dietterich 1998)). This research examines how an agent may *localize* learning within such a decomposition through the use of model-based meta-reasoning. We evaluate this hypothesis in the context of game playing in a highly complex, non-deterministic, partially-observable environment.

Reinforcement Learning

Reinforcement learning (RL) is a machine learning technique in which an agent learns through trial and error to maximize rewards received for taking particular actions in particular states over an extended period of time (Kaelbling, Littman, and Moore 1996). Formally, given a set of environmental states $\mathcal{S}$, and a set of agent actions $\mathcal{A}$, the agent learns a policy, π, which maps the current state of the world $s \in \mathcal{S}$, to an action $a \in \mathcal{A}$, such that the sum of the reinforcement signals r are maximized over a period of time. One popular technique for learning such a policy is called Q-Learning. In Q-Learning, the agent calculates Q-Values, the expected value of taking a particular action in a particular state. The Q-Learning update rule can be formulated as $\mathcal{Q}(s,a) = \mathcal{Q}(s,a) + \alpha(r + \gamma \max_{a'} \mathcal{Q}^*(s,a') - \mathcal{Q}(s,a))$, where *r* is the reward received for taking the action, $\max_{a'} \mathcal{Q}^*(s,a')$ is the reward that would be received by taking the optimal action after that, α is a parameter to control the learning rate, and γ is a parameter to control reward discounting.

Although successful in many domains, the use of RL may be limited in others due to the so-called *curse of dimension-*

Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.