Analogical Reasoning in an Impenetrable-Memory Cognitive Architecture

Sayan Bhattacharyya (BHATTACH@UMICH.EDU)
John E. Laird (LAIRD@UMICH.EDU)
Department of Electrical Engineering & Computer Science
University of Michigan, Ann Arbor MI 48109 USA

Introduction

Much of recent work in cognitive science treats analogy as a separate problem that can be analyzed in isolation from the other cognitive capabilities of an agent. In this article, we describe work in progress that takes a different approach. Our approach is to build an integrated agent in which analogy will be a capability integrated with other architectural capabilities. We start with an existing cognitive architecture and explore the issues involved in getting this architecture to perform analogical reasoning. The point of the work is to build the capability of analogical reasoning without violating the assumptions which are constrained by the necessity to provide the other capabilities of the architecture.

The advantage of this approach is twofold. In practical terms, this aproach is useful because the addition of the new capability does not demand the creation of a completely different type of agent. Also, since we require that constraints derived from other cognitive capabilities of the architecture be also simulataneously satisfied when incorporating the capability of analogical reasoning, this requirement helps to plausibly constrain the design space of the agent.

Description of the Problem

Our cognitive architecture of choice is Soar, which has been proposed as a unified theory of cognition (UTC) and has been shown capable of modeling a wide range of cognitive phenomena (Laird, Newell, & Rosenbloom, 1987). Soar is a production system architecture which solves problems using a problem-space computational model and which has a single general-purpose learning mechanism. Soar has a production memory which contains production rules and a separate working memory. Deliberate problem solving in Soar occurs by means of operators acting on states. The proposal, application and termination of operators are mediated by the firing of production rules.

Our goal is to develop an analogical reasoning capability within the framework of the Soar architecture without undoing any of its architectural assumptions. This capability should make it possible for Soar to learn new new operators by analogy with prior existing operators.

Since Soar's productions are in production memory, and since production memory is Soar's way of modeling long-term memory, it is impenetrable to the architecture (i.e. the architecture does not have read access to the production memory). A production rule can be sensed by the architecture only in the way it changes the current state when its preconditions match the current state. However, in order to derive a new production rule by analogy with an existing production rule, it is necessary to have explicit read access to the contents of the rule so that it can be transformed to yield a new rule.

Description of the Solution

The only way to gain explicit read access to the contents of a rule in an impenetrable memory is to reconstruct the rule based on the result of its action. While it is easy to reconstruct the action side of a rule (by comparing the state before and after its execution), it is much more difficult to reconstruct the precondition of a rule. Our scheme accomplishes this using a combination of an inductive learning method and a recall method which relies on a generate-recognize strategy similar to dual-process theories of recall in cognitive psychology.

A variant of the SCA inductive learner (Miller, 1993) is used for category learning – learning to classify preconditions into categories representing parametrized operators. Each category, like each precondition, is actually a set of attribute-value pairs. Learning takes place as and when operators are proposed during normal problem-solving and is thus interleaved with normal execution. The result of learning is the accummulation of rules which associate partially to fully complete sets of preconditions with categories. During the recall phase, for a given category (i.e. a parametrized operator), values are generated incrementally for its attributes and the rules learned in the learning phase are used to sift the correct values for the attributes at each step (because only the correct values match the learned rules, the rules act as a filter). So, depending on how much learning has taken place, preconditions can be reconstructed with varying degrees of accuracy. Thus, in spite of the impenetrability of production memory, recovery of productions in explicit and readable form is made possible. Rules encoding transformation relations within the domain can now be used to transform both the preconditions (newly made explicit) as well as the parameters of the existing operators. Thus, operators with new parameters and new preconditions can be learned in a systematic way, and the entire process is equivalent to learning by transformational analogy.

References

Laird, J.E., Newell, A. & Rosenbloom, P.S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence, 33*, 1-64.

Miller, C.S. (1993). *Modeling concept acquisition in the context of a unified theory of cognition*. Doctoral dissertation, Dept. of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor.