Soar: A Functional Approach to General Intelligence

Soar is a theory, implemented as a software architecture, that seeks to describe and realize the fundamental, functional components of intelligence [1]. It grew out of research in problem solving and learning spanning computer science and cognitive psychology [2]. Building on this foundation, in the last 10 years, much Soar research has focused on building intelligent systems that interact autonomously with complex environments that are inhabited by other entities, including other intelligent agents and human agents.

This paper describes the elements that comprise Soar, focusing on its role in the creation of intelligent systems. We illustrate with examples of a military aircraft pilot's tactical control and decision making. Researchers and developers in the Soar community have developed TacAir-Soar, a real-time Soar system that approximates such pilot behavior in simulation for all U.S. Navy fixed-wing air missions [3].

Functional mechanisms of Soar

The continuing thread of Soar research has been to search for a minimal set of mechanisms that are sufficient to realize the complete range of intelligent behavior. An important dimension of this research is to test the sufficiency of the mechanisms by creating real implementations that require increasingly broad capability and complexity. The hypothesis is that only by pushing Soar systems in complex domains will we be able to understand where the software implementation fails to meet the goals of the theory. Over the course of the Soar project, Soar itself has changed many times, as new experiences led researchers to reconsider some basic mechanisms and, in some cases, to create new ones. Today, research continues in the search for new mechanisms and the application of the mechanisms to difficult problems. The following describes the current mechanisms of the Soar architecture.

- **Parallel, associative memory**

  Soar uses an associative mechanism to identify knowledge relevant to current problems and to bring that knowledge to bear on potential solutions. An extremely efficient symbolic pattern matcher compares a representation of the current context to the activation conditions for each element in the system's knowledge base. Any relevant knowledge activates, leading to further elaboration of the context. The agent's reasoning context includes its current beliefs about the state of the world, proposals to perform different tasks, actions to be executed, etc. Thus, the flow of control in Soar is determined by the associations made in memory, not by the sequential, deterministic control structure used in most programming languages. Because many of these associations can be made independently from each other, Soar allows them to occur in parallel. The functional role of parallelism is that many types of relevant knowledge can be brought to bear on a problem. Thus, for a single problem, a Soar agent can simultaneously consider problem decomposition, analogy, experimentation, or other solution methods.

- **Belief maintenance**

  Soar agents are typically situated in complex, dynamic environments. Soar employs a computationally inexpensive truth maintenance algorithm to update beliefs about the world. For example, a pilot agent might believe that an enemy aircraft is pointed at it. This belief would have been derived from environmental inputs such as the enemy heading. When the enemy's heading changes, the agent will automatically update its belief about the enemy being pointed at it. Belief maintenance ensures that agents are responsive to their environments. It also embeds the dynamics of maintenance in Soar itself, so that an agent developer is freed from having to create extra, explicit knowledge to manage belief changes.

 3600 Green Road suite 600, Ann Arbor, MI 48105 • 734-327-8000 • Fax: 734-913-8537 • info@soartech.com • www.soartech.com

- **Preference-based deliberation**

  While reactivity to the environment is important, an agent in a dynamic environment must also be able to deliberate and commit to achieving lasting goals. Soar balances automatic belief maintenance with a mechanism of deliberation. Deliberation lets an agent make more of a commitment to certain beliefs and goals than normal belief maintenance allows. A Soar pilot agent could commit to a particular course based on a target's position at a particular point in time. Even as the target's position changes the agent remembers its previously derived course.

  Deliberation in Soar is mediated by explicit representations of preferences. At any particular point in time, an agent may have any number of goals or activities it could consider. In such cases, Soar allows an agent to bring further knowledge to bear on its decisions. For instance, at some point in its execution, an agent might be faced with one of multiple targets to intercept. Recognizing these potential choices, a Soar agent then would apply further knowledge to assert preferences about the different targets; using a variety of measures to determine the primary threat. Soar's decision procedure sorts through these preferences to make a decision. The key idea is that Soar does not make an "automatic" choice. Rather, it uses agent knowledge to reach each decision.

- **Automatic subgoaling**

  While it is useful to be able to choose between options, an agent could find itself in a situation where it recognizes no available options or can not decide between options. The pilot agent might have preferences indicating that it would be desirable to intercept both the target with the longest range weapon and also the target nearest in range. If these two targets are different, there is a conflict. Soar recognizes such a conflict, marks it as a reasoning "impasse", and automatically creates a subgoal to resolve the impasse. This automatically generates a new problem context on which the agent can bring new knowledge to bear. The agent might use planning knowledge to consider possible futures and determine an appropriate course of action. It might make a choice by comparing this situation to others it knows about and then drawing an analogy.

  Automatic subgoaling gives Soar agents a meta-level reasoning capability, the ability to reason about their own reasoning. Further, Soar's meta-level reasoning is not a separate component of the overall system. Soar agents can use identical knowledge to both act in the world (push the fire button) and reason about actions (what will happen if I push the fire button?).

- **Decomposition via problem spaces**

  In addition to not knowing what to do, or how to select among alternatives, an agent may be faced with knowing what it wants to do, but not having any immediate operational way to achieve that goal. For example, once a pilot agent decides to intercept a particular target, it still must make a series of decisions about *how* to prosecute the intercept. This situation also results in an impasse in Soar. In this case, there is a selected commitment, and the impasse (of not having an immediate way to achieve that commitment) creates a goal to find a way to achieve it. The impasse goal corresponds to a task goal, such as "execute a mission" or "intercept a target". Importantly, Soar makes a distinction between the goal itself and how the goal is achieved. Thus, the same goal can be attacked in different ways at different times, depending on the situation.

  Automatic subgoaling enables task decomposition. At each step in the decomposition, the agent is able to focus its knowledge on the particular options at just that level and ignore considerations at other levels. At the level of executing its mission, the pilot agent considers patrolling and intercepting; once it has decided to intercept, it then considers actions appropriate to the intercept mission, such as closing range with the target and employing weapons. The process of decomposition narrows a potentially exponential number of considerations into a much smaller set of choices, which is important for agents that have finite computational resources and that must remain reactive to the environment. Recognizing and focusing on only knowledge appropriate to the current problem is the essence of the problem space hypothesis, one of Soar's theoretical foundations [2].

3600 Green Road Suite 600r, MI 48105 ● 734-327-8000 ● Fax: 734-913-8537 ● info@soartech.com ● www.soartech.com

- **Adaptation via generalization of experience**

  Soar's mechanism of adaptation is both conceptually simple and powerful. All impasses represent a lack of immediately applicable knowledge. However, once the agent reflects and comes to a decision that resolves the impasse, it summarizes and generalizes the reasoning that occurred during the impasse. This process results in new knowledge that will allow the agent to avoid an impasse when encountering a similar situation. Soar's learning mechanism has been used to learn new conceptual knowledge, to learn new procedures, and even to correct its knowledge as it gains feedback through experience in its environment. There are a variety of different styles of learning that an intelligent agent might exhibit. Past research demonstrates that Soar's simple generalization method can implement many of these styles. The research hypothesis is that Soar's method can implement *all* of them. However, the precise way to implement some styles of learning is still a research question, and even some well understood methods are still difficult to implement in individual Soar agents. Other adaptive intelligent architectures pragmatically finesse the difficulty of having a single, primitive method for adaptation by directly implementing a variety of high-level learning styles.

- **Emphasis on efficiency and performance**

  Although not a fundamental aspect of the theory, Soar researchers have devoted significant energy, effort and creativity in engineering an efficient, high performance reasoning system. All other things being equal, an intelligent system with a faster cycle time will outperform one with a slower cycle. Thus maximizing the speed of reasoning continues to be an important component of the overall aims of the Soar project. These efforts focus on both low-level efficiency (e.g., making Soar's pattern matcher and decision cycle as fast as possible) and high-level efficiency (e.g., engineering agent knowledge bases so they do not waste time examining irrelevant courses of action).

Soar is an attempt to define a small set of general mechanisms of intelligence that are sufficient to generate all types of intelligence. The downside of the Soar approach is that, by dictating very general mechanisms, it underspecifies the capabilities that must be built into intelligent agents. Most of an agent's competence arises from the encoded knowledge (i.e., the set of rules) that Soar's mechanisms operate on. Thus, agent knowledge must be created to realize any high-level intelligent behavior. For instance, Soar has been used to build planning systems. In comparison to other AI planning systems, Soar offers little immediately evident power. Soar only specifies very low-level constraints on how planning can occur, so Soar agent designers must develop their own plan languages and algorithms, while these are provided in most planning systems. However, what Soar does provide is a natural, scalable methodology for integrating planning with plan execution -- as well as natural language understanding, reasoning by analogy, etc. Too often, AI researchers develop powerful single-purpose systems that then create new problems when different sub-systems must be integrated to realize a complete intelligent agent. By focusing on a uniform substrate that allows knowledge to mediate any decision, Soar provides a ready tool with which to realize integrated approaches. Soar therefore trades off powerful, but overly constrained processes for the flexibility to integrate solutions. As TacAir-Soar suggests, Soar has been a critical foundation on which successful complex intelligent systems have been – and continue to be – created.

For further information

[1] Laird, J. E., Newell, A., and Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence, 33*(1): 1-64.
[2] Newell, A. (1990). *Unified Theories of Cognition.* Harvard Press: Cambridge, MA.
[3] Jones, R.M. Laird, J. E., Nielsen, P. E., Coulter, K. J., Kenny, P., Koss, F. V. (1999). Automated Intelligent Pilots for Combat Flight Simulation. *AI Magazine*. Spring, 1999.

3600 Green Road Suite 600r, MI 48105 ● 734-327-8000 ● Fax: 734-913-8537 ● info@soartech.com ● www.soartech.com