SOAR – UMA COMPILAÇÃO DOS CONCEITOS DA ARQUITETURA

Rodrigo Silveira Dutra (rodrigo@fem.unicamp.br)
UNICAMP

ABSTRACT

This paper has as objective to present the architecture SOAR, its types of memory and the application of this architecture in works already accomplished. The conceptual base of this work is made herself starting from the compilation of article SOAR and the documentation of the same, tends as objective to centralize concepts dispersed.

Keywords: Architecture Soar, Memory, learning

INTRODUÇÃO

A ciência cognitiva tem como objetivo integrar conceitos de diferentes áreas de pesquisas. De acordo com as arquiteturas existentes que aborda a ciência cognitiva, SOAR representa uma delas, abrangendo aspectos relacionados a diferentes formas de armazenamento e recuperação em memórias de longo e curto prazo. A partir deste conceito, é possível definir arquitetura cognitiva como uma teoria dos mecanismos fixos e estruturas que envolvem a cognição humana. De fato, a maioria de nosso comportamento cotidiano parece exigir algum grau de pensamento, a fim de mensurar nossas percepções e ações. Em geral, a idéia de arquitetura é útil porque nos permitem desmembrar alguns aspectos comuns de comportamentos que caracterizam o conteúdo [1].

Estados e Operadores são as estruturas básicas de apóio à arquitetura, Figura 1. Os estados contêm toda a informação sobre a situação atual, inclusive percepção e descrição de metas atuais e o espaço do problema, enquanto os operadores são os meios para percorrer pelo espaço do problema. A partir do comportamento meta-orientado, a sucessão de operadores que são aplicados, ao estado e suas transformações devem ser gerenciados pelo princípio de racionalidade: "*se um agente tem conhecimento sobre uma aplicação de operador, conduzirá então a um de suas metas e o agente selecionará aquele operador*" [2].

Um estado representa a situação atual do modelo que inclui a meta. Assim, um estado tem que ter alguma característica como meta, cujo valor tem características que descrevem a meta. Deste modo, a meta está disponível para análise durante o problema a ser solucionado avaliando o progresso e direcionando a seleção de operadores. Logo, deve existir conhecimento suficiente que solucione quando o estado satisfaz a meta, que então notifica a arquitetura. O espaço do problema também é representado como uma característica do estado.

O que é importante não é se a aplicação de operador preserva algum tipo de relação de verdade entre o mundo externo e sua simulação interna. O que é importante é que usando estados e operadores é possível modelar qualquer comportamento, agindo ou pensando em agir, de acordo com uma função que retorne um tipo de conhecimento.

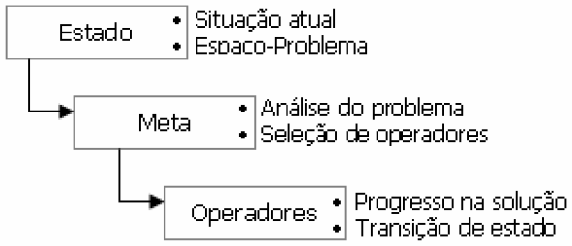

Figura 1

ESTRUTURAS DE MEMÓRIA

Para fazer sentido à realidade e funcionar no mundo real, a utilização do conhecimento sobre objetos e ações em perseguição de nossas metas atuais se faz de grande importância. A divisão entre conhecimento geral e aplicações de um determinado conhecimento é capturada pelo SOAR pela existência de dois tipos diferentes de memória.

Primeiro, o conhecimento que existe, independente da situação atual, é definido pela memória de longo prazo (LTM). SOAR distingue três tipos diferentes de LTM: processual, semântico, e episódico. O conhecimento processual está relacionado como e quando fazer

algo, o conhecimento semântico consiste em fatos sobre o mundo e o conhecimento episódico, de acordo com [3] retrata eventos e história que são adquiridos conforme a experiência, enquanto que a aprendizagem semântica tenta extrair fatos no contexto experimental.

A LTM não está diretamente disponível, mas é acessada de acordo com a situação atual. O segundo tipo de memória corresponde à Memória de Trabalho, a WM também pode conter conhecimento geral que é relacionado à situação atual, fatos gerais ou recordações de situações prévias específicas, que são úteis na tomada de decisão sobre uma situação específica. Em SOAR, WM é representado como um conjunto das características e valores que compõem o estado atual (substates) que poderia incluir representações da meta atual, espaço de problema, e operador.

SOAR distingue entre dois tipos de persistência em memória de trabalho. O trabalhando dos elementos de memória (WMEs) é criado como parte de uma aplicação de um operador persistente, permanecendo em memória de trabalho até que eles sejam removidos, sendo denominados de o-supported. O WMEs restante, denominado de i-supported, são removidos assim que a instanciação da produção que os criou deixe de emparelhar na memória de trabalho. A vantagem de i-supported é que remove automaticamente da WMEs o que não é mais utilizado pela situação atual. Por exemplo, um operador é proposto baseado em características específicas da situação, quando um destas características muda, o operador é definido automaticamente. [4]. A Figura 2 ilustra a estrutura de memória do SOAR.

É útil pensar em LTM como contendo o que pode ser pertinente a muitas situações diferentes, mas deve ser recobrado explicitamente, e WM como contendo o que o modelo pensa com relação à situação atual e específica. O conhecimento move de LTM a WM por recuperação automática e deliberada de estruturas de LTM pertinentes.

Embora todos os tipos de conhecimento de longo prazo (processual, semântico, e episódico) sejam considerados úteis, o conhecimento processual é o principalmente responsável para controlar comportamento e mapeamento direto sobre conhecimento de operadores. Os conhecimentos semântico e episódico normalmente entram em ação quando o conhecimento processual está, de algum modo, incompleto ou inadequado para a situação atual.

Os processos cognitivos são inicializados freqüentemente e terminam em pontos arbitrários. O trabalho de elementos de memória em SOAR dá origem em dois modos, através de percepção ou por recuperações de memória de longo prazo durante o ciclo de decisão.

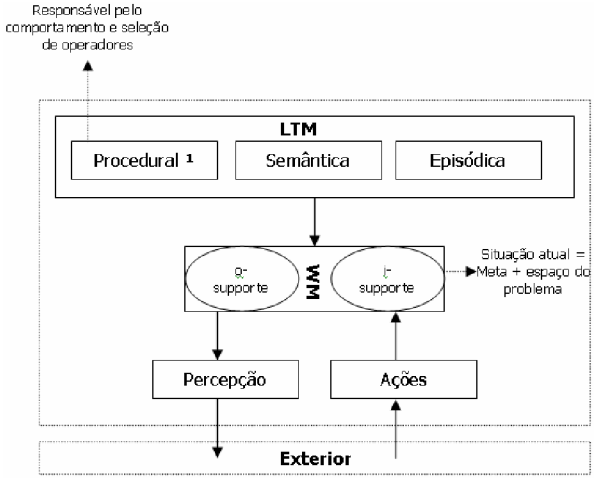

Figura 2

DECISÃO

O ciclo de decisão é o componente de processo que gera comportamento fora do escopo da memória de longo prazo e trabalha com recordações. O propósito do ciclo de decisão é selecionar o próximo operador para aplicar. Um ciclo de decisão é um mecanismo de processo fixo na arquitetura SOAR que faz seu trabalho em cinco fases: introdução, elaboração, decisão, aplicação, e produção. As fases são necessárias para a arquitetura produzir comportamento conforme o princípio de racionalidade

- A introdução está relacionada com a criação de elementos de memória de acordo com a variação da percepção.
- Durante elaboração, os conteúdos de memória de funcionamento são emparelhados de acordo com as regras em memória de longo prazo, desta maneira, a inicialização da memória processual torna-se o centro da execução, logo, resultam em mudanças às características e valores do estado além de sugestões, ou preferências, por selecionar o operador atual. Com o surgimento de mudanças da memória de funcionamento, novas regras podem ser iniciadas. A fase de elaboração é processada até que não existam mais acionamentos de regras. A elaboração é conseqüência da mudança de processamento para a memória de trabalho. Correspondendo a conclusões simples ou

sugestões (no caso de preferências) e não ações no mundo ou mudanças para estruturas internas que deveriam persistir além da situação atual.

De acordo com o princípio de racionalidade, o modelo deve usar seu conhecimento para alcançar suas metas. A introdução de preferências permite a arquitetura colecionar toda a evidência disponível para mudanças de potencial para o estado, antes de produzir qualquer mudança de fato. Assim, a ausência de qualquer disparo adicional de regras de LTM, sinaliza o começo da fase de decisão.

O procedimento de decisão é aplicado ao vocabulário de preferências, independente da semântica do domínio. O vocabulário inclui preferências simbólicas, como um o operador a outro, mas também inclui preferências numéricas, como o valor esperado de um determinado operador.

Claro que, seleção do operador não é bastante. O operador deve ser aplicado para produzir uma transição de estado, a fim de contemplar um movimento no espaço do problema. Em SOAR, só um único operador pode ser selecionado para um estado em um determinado momento. Normalmente, há um recurso que só pode ser usado em modos limitados e uma decisão deve ser feita, como usar isto agora, para a situação atual. O empacotamento permite ações múltiplas junto como um único operador que é selecionado como uma unidade.

Duas melhorias são acrescidas ao sistema de ativação e a experiência de repetição, Figura 3. Modificando o sistema de forma que os conjuntos de o-supported recebem aumentos em ativação, elevou a precisão porque era mais provável que as características corretas (que são o-supported) contribuem para a recuperação de um episódio. Impulsionando novos WMEs o impacto torna-se maior porque um das características mais importantes é criada logo antes de um episódio ser armazenado. De acordo com [4], a modificação impulsiona a direção proposta de característica de movimento e produz uma melhoria significante na habilidade do agente recobrar episódios apropriados e selecionar a ação correta. A maneira mais simples para manter ativações é atualizar os valores de ativação de toda WME em todos os ciclos. Porém, a maioria de WMEs não é acessada em um determinado ciclo, de forma que as ativações calculadas nunca são usadas. Uma aproximação alternativa é atualizar só a ativação quando uma WME é acessada ou quando precisa ser removida.

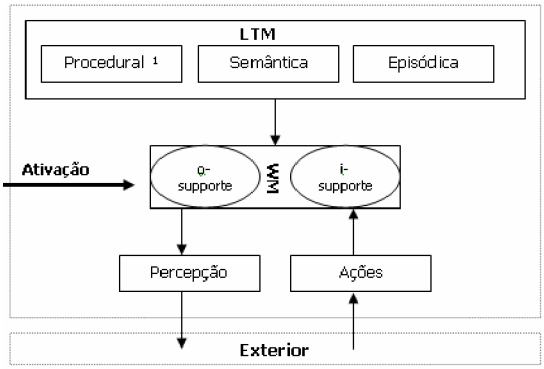

Figura 3

IMPASSE

Um impasse é um evento que surge sempre que o procedimento de decisão não pode solucionar as preferências em memória de trabalho. Logo, o ciclo de decisão não pode decidir. Freqüentemente as razões por selecionar um a outro estão baseado em raciocino deliberado que não é fácil de expressar em uma ou duas regras.

O argumento exigido para selecionar um operador requer decisões e explorações. Para apoiar estes tipos de processo, SOAR automaticamente cria substate em resposta para um impasse. Um substate é um estado novo que representa a informação pertinente para solucionar o impasse.

Para controlar a recuperação de conhecimento de memória episódica, deve ser criada uma sugestão que pode ser usado para procurar na memória. A própria sugestão é criada por um operador que seleciona estruturas do superstate. Uma vez evocada a memória episodica a memória de trabalho contém informação que pode ser usada para tentar solucionar a impasse.

Em geral, a memória de trabalho consiste em uma hierarquia de state/substate onde cada substate existe para solucionar um impasse no espaço do problema. A hierarquia cresce com o surgimento de impasses no decorrer do processo, e encolhe como os impasses estão sendo resolvidos, extraindo o conhecimento exigido no contexto mais alto. Tipicamente, ciclos de decisão correspondem a movimento pelo espaço de problema que está na base da hierarquia, mas não é necessário que isto seja verdade; o ciclo de decisão ainda resulta em só uma única seleção de operador, mas se são sugeridas mudanças múltiplas em estados diferentes, a mudança ocorre no topo da hierarquia.

Um impasse de operador corresponde aos modos nos quais o ciclo de decisão não pode

decidir. Impasses na escolhe de operadores correspondem freqüentemente ao que normalmente é pensado sobre metas deliberadas, onde um problema complexo é decomposto em problemas mais simples. Além disso, impasses podem surgir para todo problema e podem incluir propondo, selecionando, e aplicando um operador, de forma que sempre que o conhecimento disponível possa ser considerado inadequado, adicional e possivelmente armazenado em memória episódica semântica, no substate.

APRENDIZADO

Quando uma tarefa é realizada repetidamente, a cada vez uma melhora é desenvolvida, ou seja, com menos erros ou de maneira mais rápida, ou simplesmente parece exigir menos esforço. De alguma maneira a habilidade é alterada de acordo com o tempo em que ficamos sem executar uma determinada tarefa. Em resumo, resultados são obtidos através do aprendizado, ou seja, aquisição de conhecimento por experiência. Esta questão de adquirir conhecimento muda o comportamento com o passar do tempo.

Devido ao fato que é possível conduzir para uma mudança em comportamento, a arquitetura ou o conteúdo deve ser afetado por experiência. Desde que a arquitetura é, por definição, um conjunto de estruturas fixas e mecanismos, deve ser o conteúdo que muda. E desde que conteúdo de domínio é capturado por regras, fatos e episódios o aprendizado tem que criar essas estruturas. Por um período SOAR suportava um único mecanismo de aprendizagem denominado chunking, sofrendo um *upgrade* e passando a incluir três mecanismos de aprendizagem adicionais: reforço no aprendizado, aprendizagem semântica, e aprendizagem episódica.

CHUNKING

Chunking é um mecanismo de aprendizado que adquire regra de experiência meta-orientada[5]. Quando um impasse surge significa que o sistema não possui regras disponíveis em memória de longo prazo. Expressado um pouco mais intuitivamente, um impasse é o modo da arquitetura sinalizar uma falta de conhecimento, logo, uma falta de conhecimento é uma oportunidade por aprender.

Em resposta para o impasse, um substate é criado para buscar o conhecimento. O conteúdo de domínio ampliou este substate que envolve a revocação, então avalia operadores que provêem o contexto que poderia ativar conhecimento de LTM.

Como resultado de processar no substate, o conhecimento exigido escolhe o operador que satisfaz o conhecimento, logo o impasse poderia ser solucionado. Além disso, nós tivemos uma oportunidade para aprender uma nova regra que fará aquele conhecimento, dada as circunstâncias imediatamente disponíveis que originalmente conduziu ao impasse. De fato, a arquitetura forma uma regra nova automaticamente sempre que são gerados resultados de um impasse, Figura 4.

A distinção entre regras informadas ao sistema e regras desenvolvidas pelo sistema, nós lhes damos um nome especial, *chunks*, e chamamos o processo de aprendizagem que os cria de chunking.

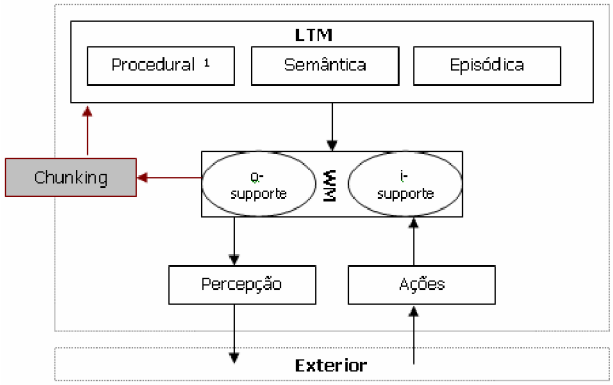

Figura 4

REFORÇO NO APRENDIZADO

Uma fonte de conhecimento que está fora de conhecimento interno é realimentação do ambiente, ou o que é chamado freqüentemente de "recompensa". A recompensa pode vir do "agente" em qual um modelo é embutido, ou pode ser gerado interiormente quando uma meta é alcançada. A recompensa pode ser positiva ou negativa. A realimentação também pode ser tratada a um nível mais abstrato, como uma recompensa por ter sucesso em uma tarefa ou fracasso.

Recompensa pode ter impacto imediato em um comportamento, informando um modelo que deveria ou não evitar a situação atual. Mas a recompensa também pode ser usada para imprensar termo comportamento longo por aprender. Para capturar este tipo de conhecimento SOAR baseia-se em experiência e ajuste de reforço de aprendizado através de predições de recompensas futuras, selecionando ações que

maximizam o futuro e então esperara recompensas.

Dentro da arquitetura SOAR, o reforço no aprendizado possui duas partes, a primeira está relacionada com o aprendizado de regras que testam as características apropriadas dos estados e operadores. Segunda, para as regras solicitadas, SOAR tem que saber identificar os retornos pertinentes a cada regra.

O objetivo é criar regras, inicialmente baseadas nas regras que propõem os operadores, e então especializar essas regras se eles não incluem condições suficientes de forma que prediga o mesmo valor.

Chunking e reforço de aprendizado criam regras novas, mas o reforço de aprendizado só cria seleção de operador e a base pela criação são regularidades estatísticas em recompensas ao invés dos resultados de raciocínio interno. [6] destaca uma melhora no reforço do aprendizado para a arquitetura SOAR, incorporando o ajuste de valores numéricos sobre as preferências, ou seja, criando novas regras que testam um conjunto de características diferentes enquanto criam preferências numéricas.

MEMÓRIA EPISÓDICA

Outra fonte de conhecimento que está disponível é o fluxo de experiência. A memória de experiências é chamada memória episódica. A memória episódica registra eventos e história que são embutidas em experiência.

Um episódio pode ser usado para responder perguntas sobre o passado, predizer o resultado de possíveis cursos de ação, ou ajudar a manter o progresso em metas. Um conjunto de episódios pode melhorar comportamento por outros tipos de aprendizagem, como reforço que aprende ou chunking, tornando possível a obtenção de experiência, quando há tempo para reflexão.

Em SOAR, são registrados episódios automaticamente como um problema é resolvido [3]. Um episódio consiste em um subconjunto dos elementos de memória de funcionamento que existem na hora de registrar. SOAR seleciona esses elementos de memória de trabalho sobre os que foram recentemente usados.

Para recobrar um episódio, uma sugestão é criada em memória de trabalho (através de regras), além de ser identificado de maneira distinta entre o que é uma memória (o episódio) e o que está sendo experiente. Uma vez o episódio é recobrado, pode ativar disparos de regra, ou até mesmo é a base por criar sugestões novas para procuras adicionais de memória episódica.

A memória episódica difere de reforço de aprendizado e chunking em muitas formas, sendo um mecanismo de aprendizagem passivo que não faz nenhuma generalização. Outra diferença é que os conteúdos de um episódio só são indiretamente determinados argumentando. Conforme Framework [3], a simulação computacional para a memória episódica pode ser decomposta de acordo com as seguintes fases: Codificação, armazenamento, recuperação e uso.

Finalmente, para o aprendizado episódico, não há nenhuma distinção entre condições para recuperação e o que deveria ser recobrado. Em contraste, o conhecimento adquirido por chunking e reforço de aprendizado é assimétrico, há condições distintas e ações, e as condições têm que emparelhar para recobrar as ações exatamente.

MEMÓRIA SEMÂNTICA

A fonte final de conhecimento é a co-ocorrência de estruturas em memória de trabalho. Estas co-ocorrências são mais gerais que episódios que representam apenas ocorrências. Elas são distintas do que pode ser construído em regras por chunking ou reforço de aprendizado porque elas estão sobre uma estrutura estática em vez de regras baseadas em derivação. Estas são estruturas declarativas, conhecimento semântico também é conhecido pelo “o que você saiba” (em contraste com isso que você “se lembra” da memória episódica).

Uma estrutura de memória semântica é recobrada da mesma maneira como é um episódio, uma sugestão é criada deliberadamente em memória de trabalho através de regras, a sugestão busca a melhor partida, em termos de memória longa, e então a memória completa é recobrada. Em geral, uma memória semântica é muito menor e mais compacta que um episódio e representa as relações entre alguns elementos em memória de trabalho em vez da memória completa.

APLICAÇÃOES

Como exemplos de aplicação da arquitetura SOAR, é importante destacar alguns projetos, dentre eles:

- NTD-Soar – utilizada pela NASA para auxilio no controle de lançamentos.
- TacAir-Soar – utilizada para treinamentos de pilos e situações de missões críticas.
- Instructo-Soar – aprendizado interativo por instrução.

REFERÊNCIAS

[1] Lehman, J. F., J. Laird, e P. Rosenbloom (2006). Gentle introduction to soar, an architecture for human cognition: 2006 update.

[2] Newell, A. (1982). The knowledge level. artificial intelligence, 18, 87-127.

[3] Nuxoll, A. e J. E. Laird (2004). A cognitive model of episodic memory integrated with a general cognitive architecture.

[4] Nuxoll, A., J. E. Laird, e M. R. James (2004). Comprehensive working memory activation in soar.

[5] Laird, J. E., P. S. Rosenbloom, e A. Newell (2003). Chunking in soar: The anatomy of a general learning mechanism.

[6] Nason, S. e J. E. Laird (2004). Soar-RL: Integrating reinforcement learning with soar.