Inteligência Artificial

Prof. Robson de Souza

Resolução de problemas por meio de busca

Os agentes inteligentes devem maximizar sua medida de desempenho, esse objetivo é as vezes simplificado se o agente pode adotar um objetivo que deseja satisfazer.
Os agentes mais simples eram os agentes reativos, que baseiam suas ações em um mapeamento direto de estados em ações. Tais agentes não podem operar bem em ambientes para os quais esse mapeamento seria grande demais para se armazenar e levaria muito tempo para se aprender. Por outro lado, os agentes baseados em objetivos, consideram ações futuras e o quanto seus resultados são desejáveis.

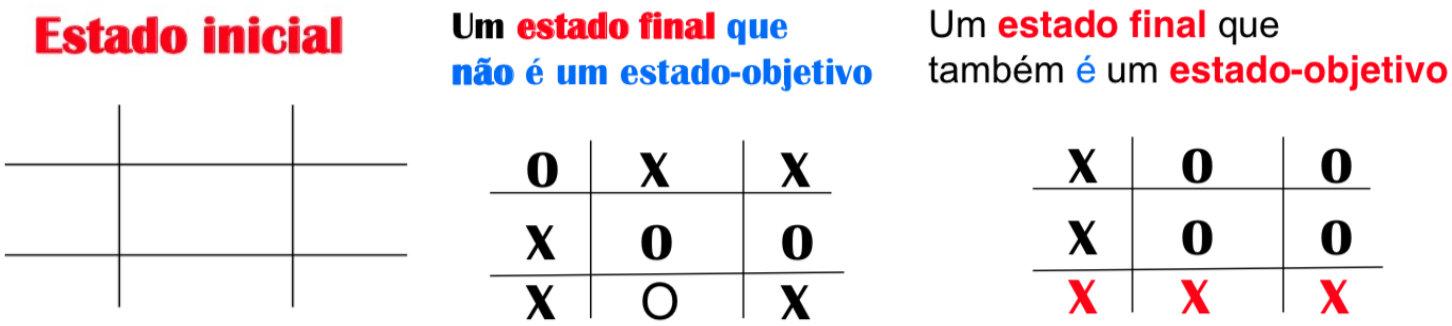
Existe um tipo de agente baseado em objetivo chamado **agente de resolução de problemas**. Os agentes de resolução de problemas utilizam representações atômicas, ou seja, os estados do mundo são considerados como um todo, sem estrutura interna visível para os algoritmos de resolução de problemas.

Resolução de problemas como busca num espaço de estados

Para construir um sistema para resolver um problema dado precisamos **formular objetivos** com base na **situação atual** e numa **medida de desempenho do agente**. A **formulação de um problema** é o processo de escolher, dado um objetivo, quais estados e quais ações devem ser considerados.

Para construir um sistema para resolver um problema em **particular** nós precisamos definir o problema **precisamente** (inclusive definindo quais estados finais constituem soluções para o problema), **analisar** o problema, **identificar e representar** o conhecimento específico à tarefa que é necessário para se resolver o problema e **aplicar as técnicas de resolução** de problemas mais adequadas para o problema em particular.

Exemplo (Jogo da velha):



Podemos representar a solução de problemas como uma sequência de estados, que leva de um estado inicial até um estado final, onde cada estado é um estado admissível, produzido a partir de um estado anterior através de uma ação, ou seja, uma mudança de estados admissível.

Podemos representar as mudanças de estado admissíveis de várias formas, principalmente por meio de regras de produção, que são estruturas da forma SE P ENTÃO R onde: P é o antecedente (ou lado esquerdo) e R é o resultado (ou lado direito) da regra.

Para encontrar a solução de um problema, por meio de busca, é necessários fazer uma **descrição formal do problema**. As descrições formais de problemas são feitas do seguinte modo:

1 - Defina um **espaço de estados** que contenha **todas as possíveis configurações** das entidades relevantes;
2 - Especifique um ou mais dentre os estados que correspondem ao **estados iniciais**, ou seja, estados a partir dos quais o processo de resolução de problemas pode se iniciar;
3 - Especifique um ou mais **estados-objetivo**, ou seja, estados que representam soluções aceitáveis do

problema;
4 - Especifique um conjunto de regras que descreva os operadores, ou seja, as ações disponíveis, em particular, decida quanto do conhecimento sobre a solução do problema será embutido nessas regras.

Busca

A busca na Inteligência Artificial é a técnica geral de exploração do espaço de estados, de modo a se achar uma sequência de estados que leve de um estado inicial a um estado objetivo. A busca se dá construindo (e podando) uma árvore de busca, nessa árvore, **cada nó é um elemento do espaço de estados** e **cada ramo que é aberto define uma sequência de estados**, representando uma possível solução para o problema.

Se, durante a resolução do problema, podemos atingir o mesmo estado por caminhos diferentes, então a árvore de busca é na verdade um grafo. Cada **mudança de estados** pode ser caracterizada pela **aplicação de uma regra de produção**.

Resolução de Problemas = descrição formal de problemas + uso de uma estratégia de controle da busca, que leve de um estado inicial a um estado-objetivo.

Vale ressaltar que a busca é um **mecanismo geral, um método fraco** em oposição a métodos baseados em conhecimento sobre a solução do problema. A busca se dá construindo uma árvore de busca para:

- Sair de um dos estado iniciais e
- aplicar uma sequencia de operadores para
- construir um ramo da árvore que leve a um estado-objetivo.

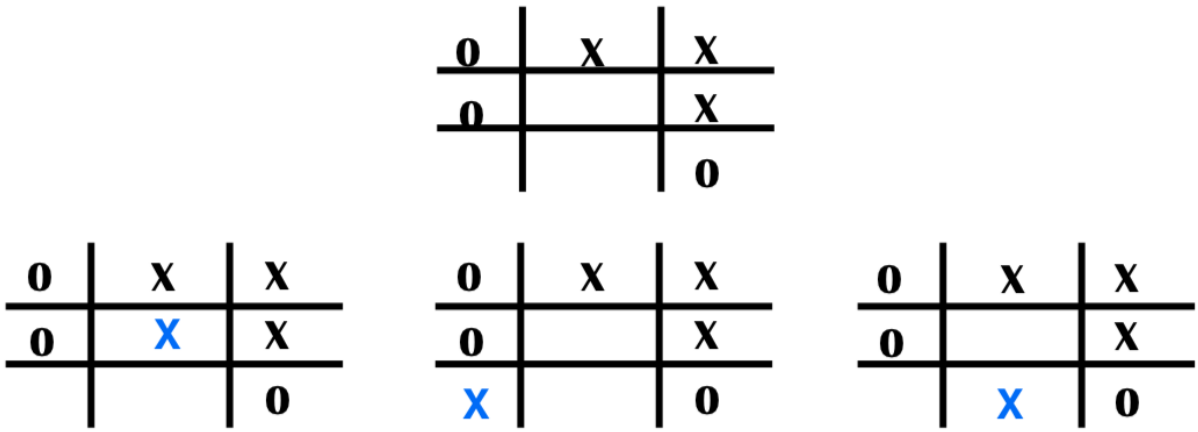
Fica claro que um algoritmo de busca precisará decidir quais nós da árvore deverão ser escolhidos para continuar a busca por um estado-objetivo. Em buscas que **não são guiadas por conhecimento** existem duas principais estratégias, que são a **busca em largura** e a **busca em profundidade**.

Busca em Largura (Breadth-First Search)

Nesse tipo de busca, todos os nós da árvore de busca no mesmo nível de profundidade são explorados antes de explorar nós em níveis mais abaixo. O algoritmo da busca em largura segue as seguintes ações:

- Escolha um estado inicial para raiz da árvore;
- Para cada nó da árvore, recursivamente: gere todos os descendentes deste nó, aplicando todas as regras admissíveis,
- até que se chegue a um estado-objetivo (ou se chegue a um estado final).

A figura abaixo mostra um exemplo de exploração em largura de um conjunto de estados em um jogo da velha.



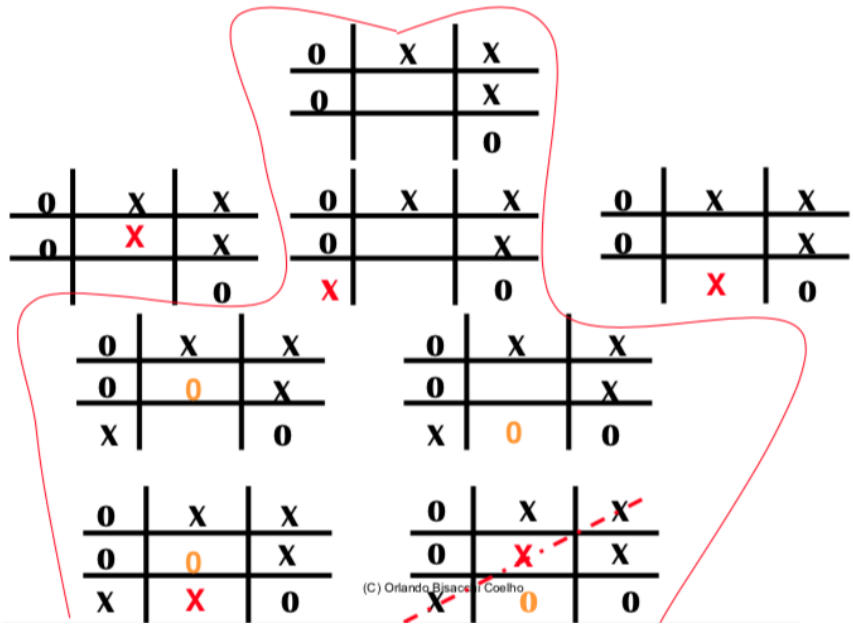
Busca em Profundidade (Depth-First Search)

Nesse tipo de busca, o algoritmo explora apenas um dos descendes de um nó até chegar a um estado-objetivo ou final. Caso a exploração leve a um beco sem saída (estado que não é objetivo e nem final), ela retorna por meio de backtracking a um estado que permite a exploração de outro ramo da árvore que pode levar a um estado-objetivo.

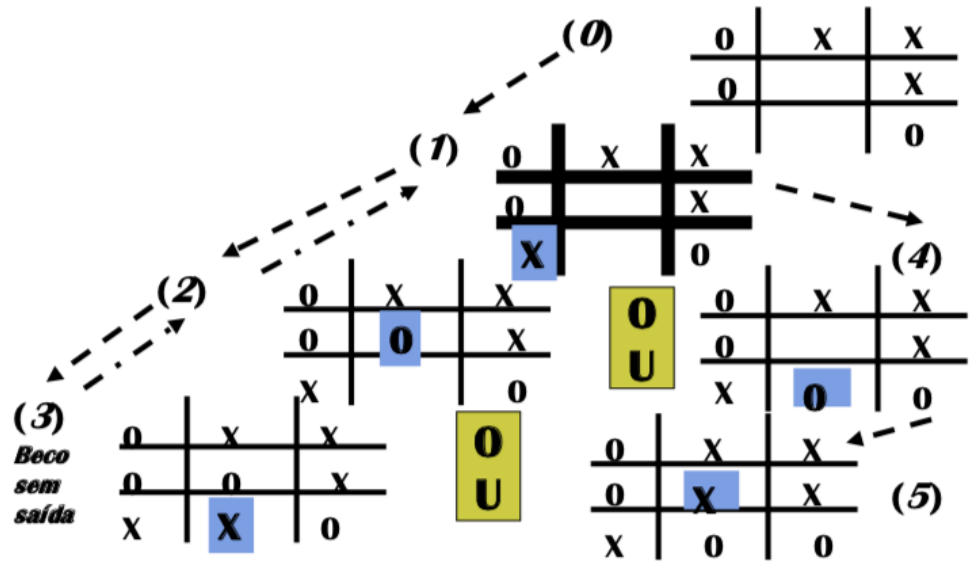
O algoritmo da busca em profundidade executa as seguintes ações:

- Escolha um estado inicial para raiz da árvore;
- Para cada nó da árvore, recursivamente:
  - gere um só dos descendentes deste nó, aplicando somente uma das regras admissíveis;
- até que se chegue a um estado-objetivo (ou a um estado final) ou até que se chegue a um beco-sem-saída;
- caso chegue num beco-sem-saída,
  - [backtrack] gere um dos descendentes alternativos do nó mais recentemente gerado e continue o algoritmo a partir daí.

A figura abaixo mostra um exemplo de exploração em profundidade de um conjunto de estados em um jogo da velha.



A figura abaixo mostra um exemplo do mecanismo de backtracking.



Busca em l

Vantagens da Busca em Largura:

- Se existe uma solução para o problema então é garantido que a busca em largura vai achar essa solução, além do mais, essa solução vai ser o mais curta possível;
- Já a busca em profundidade pode perder um longo tempo explorando um ramo profundo mas infrutífero da árvore de busca, enquanto a solução está num outro ramo, muito menos profundo.
- Não precisa de backtracking. Sem um mecanismo de backtracking não é garantido que a busca em profundidade ache uma solução.

Vantagens da Busca em Profundidade:

- A busca em profundidade requer muito menos memória, já que só os nós no caminho corrente precisam ser armazenados.
- Pode acontecer de a busca em profundidade achar a solução do problema muito rapidamente, sem precisar construir a maior parte da árvore de busca.

Referências bibliográficas:

RUSSELL, Stuart J.; NORVIG, Peter. Inteligência artificial. Elsevier, 2004.

RUSSELL, Stuart; NORVIG, Peter. Inteligência artificial. Rio de Janeiro: GEN LTC, 2013. ISBN 9788595156104.