resumo TCC

Marcos Paulo Quintão Fernandes

September 2017

1 Artigo 1: A fast algorithm for the maximum clique problem

Este artigo tem por finalidade resolver de forma exata o problema da maior clique em um grafo de maneira mais eficiente que enumerar todos os conjuntos de vértices. O problema da maior clique se consiste em encontrar o maior subconjunto de vértices de um grafo tal que qualquer par de de elementos desse subconjunto são adjacentes. Como esse problema é NP-Completo, não é possível resolvê-lo em tempo exponencial. Dessa forma, um algoritmo que o resolve é exponencial. Usando uma técnica de "branch-and-bound"na recursão, foi possível eliminar a visitação de exponenciais estados que com certeza não estão presente em alguma solução do problema. Dessa forma, foi obtido uma melhoria considerável no desempenho para os grafos testados. Ao final do artigo existe uma tabela que corresponde ao tempo de execução do algoritmo implementado. Observa-se que para instâncias de 378 vértices não foi possível obter alguma solução pelo fato do algoritmo ser extremamente lento.

2 Artigo 2: Monte-Carlo Tree Search: A New Framework for Game AI

Este artigo tem por finalidade apresentar uma ferramenta geral para jogar diversos jogos sem ter muito jogo, muita experiência com o jogo ou uma base de dados grande. A dificuldade de um jogo qualquer se deve em avaliar a qualidade do estado do jogo. No entanto, realizar essa avaliação para um estado não terminal do jogo significa analisar um espaço exponencial de possibilidades. A árvore binária de Monte Carlo, ao realizar explorações randômicas no espaço de possibilidades de um jogo, consegue realizar previsões muito precisas com respeito ao estado atual de jogo e quais é a provável melhor ação a ser feita. De acordo com o artigo, ela tem sido bastante efetiva em diversos jogos como GO, xadrez e Cattan.

3 Artigo 3: A Survey of Monte Carlo Tree Search Methods

Este artigo tem por finalidade descrever como funciona a árvore de busca de Monte Carlo. De acordo com o artigo, o algoritmo é baseado em quatro etapas:

1. Seleção - Suponha que o algoritmo se situa em um estado u e que todos os estados possíveis provenientes de u já foram visitados. Nesta etapa, através de uma política α iremos escolher qual é o melhor estado a ser analisado.

2. Expansão - Suponha que o algoritmo se situa em um estado u e que existe um conjunto S de estados possíveis que ainda não foram visitados. Nesta etapa, iremos escolher aleatoriamente algum estado a ser analisado.

3. Simulação - Suponha que o algoritmo se situa no estado criado pelo método de Expansão. A partir desse estado, iremos executar alguma heurística para avaliar qual o "benefício"desse estado.

4. Propagação Reversa - Ao realizar a simulação e encontrarmos o "benefício"do estado atual, iremos propagar para todo estado u anterior ao estado simulado, o "beneficio"proveniente do estado simulado.

É bem plausível perceber que os métodos de Simulação e Expansão são os métodos que mais contribuem para eficiência do algoritmo. O artigo citou algum dos métodos que usam as propriedades do processo de decisão de Markov, teoria dos jogos e variantes para calcular a prioridade de cada estado em cada iteração do algoritmo.