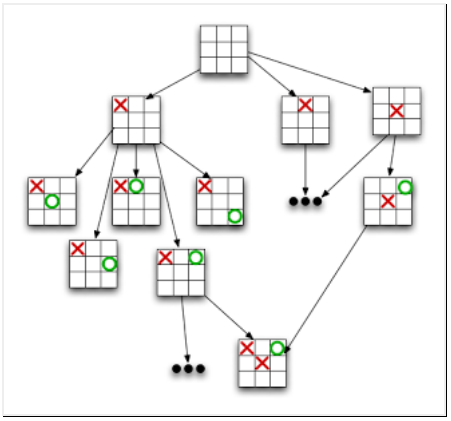
Fala Péricles

Um blog sobre algoritmos, heurísticas e horóscopos.

sexta-feira, 22 de maio de 2009

Grafo de estados e suas aplicações

Muitos problemas envolvendo jogos (quebra-cabeças) podem ser resolvidos usando backtracking ou uma BFS no grafo de estados do problema. Neste post, tratarei a aplicação do BFS no grafo de estados do problema para encontrar a solução ótima para um quebra-cabeça.



O que é um grafo de estados?

Lembra quando você estava no ensino médio e seu professor apresentou aquela árvore de possibilidades para ensinar analise combinatória? Então, o grafo de estados é uma generalização desse conceito só que ao invés de usarmos uma árvore simples, usamos um grafo que é uma estrutura mais genérica que a árvore. No grafo de estados, cada vértice é uma configuração do problema e as arestas representam uma jogada que pode ser efetuada em um turno do jogo. Como na maioria dos problemas, o grafo de estados pode ser gigantesco, criar o grafo de estados pode ser intratável. Mas o que fazer? Não tema, o GRAFO DE ESTADO NÃO PRECISA SER CONSTRUÍDO EXPLICITAMENTE!!!

Sim, o algoritmo constrói as transições implicitamente durante sua execução. E é algo realmente simples. A parte que pode complicar um pouco é a representação de cada estado, é preciso construir um modo de mapear os estados para marcar que estados foram visitados e pra controlar a distância percorrida até uma determinada configuração. O método que gosto de usar é por meio de hasheamento e representar os estados por palavras binárias (é divertido brincar com deslocamento de bits e operações lógicas para construir essas estruturas [=D]).

Existem duas formas de se utilizar o grafo de estados: uma usando uma espécie de DFS (backtracking) em que você vai recursivamente percorrendo o grafo de estados marcando os estados já visitados, outra é usando BFS. A BFS tem a vantagem de encontrar uma seqüencia ÓTIMA que leva a configuração vencedora, ou seja, a seqüencia de menor tamanho que me leva a vitória se o grafo de estados tiver pesos iguais para todas jogadas (arestas). Caso tenha pesos, só fazer um Dijkstra no grafo de estados.

Abaixo temos a ilustração do grafo de estados para um jogo da velha (bem sem graça, sei, mas o principio é o mesmo usado no jogo de Xadrez, damas, Warcraft, etc...[=P]).

Bem, a primeira coisa que temos de fazer depois de definido o grafo de estados é decidir que estado é o estado inicial e que estados são finais : estados de vitória, estados de derrota ou estados de empate. Isso, depende de cada jogo e varia de problema para problema. A técnica que apresentarei aqui é genérica e poderá ser usada para problemas cujo espaço de estados é razoavelmente pequeno e se trata de um quebra-cabeça [=)], ou seja, não é um jogo competitivo. Jogos competitivos são jogos com mais de 1 jogador cujas decisões se sobrepõe. Problemas com jogos competitivos exigem técnicas mais sofisticadas, mas entendendo o que vou apresentar aqui tu não terás problemas.

Agora, que já temos definido o modo de representar uma configuração do problema, o grafo de estados, os estados finais e iniciais. Podemos usar o BFS, abaixo vai o pseudo-código:

```
1  Algoritmo BFS_STATE_GRAPH (s,T)
2  {
3       input: O estado inicial s e o conjunto de estados finais T.
4       output: A menor seqüencia de estados que te leva a um estado final.
5
6       ans=INF;
7       pans=NIL;
8       Q.push_back(s);
9       d[s]=0;
10      pi[s]=NIL;
11
12      while(!Q.empty()){
13          u=Q.front(); Q.pop();
14          if(u pertence a T,ou seja, é estado final)
15              if(ans>d[u]){
16                  pans=u;
17                  ans=d[u];
18              }
19
20          du=d[u]+1;
21          vis[u]=1;//Vetor que controla se um estado foi visitado
22
23          if(du>ans) continue;
24
25          for(cada v, um estado que se se chega a partir de u com 1 jogada)
26              if(!vis[v] ou d[v]>du){
27                  d[v]=du;
28                  pi[v]=u;
29                  Q.push_back(v);
```

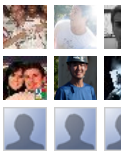
Quem sou eu

Arquivo do blog

► 2013 (5)
▼ 2009 (9)
► Setembro (
▼ Maio (7)
Usando BFS troco
Como obter seqüênci
Grafo de es
O algoritmo
O algoritmo
Uma aplica booleana
Uma peque

Seguidores

Participar
Google Friend Co
Membros (17)



Já é um membro?

Google+ Follower

+1 4

Pesquisar este blo

```
                }
        }

        return [ans,pans,pi];
}
```

Em pi temos o caminho armazenado, se você conhece o BFS sabe como reconstruir a sequencia de jogadas, ans é o numero mínimo de jogadas, pans é o estado final onde termina o caminho (util para reconstrução). O legal desse algoritmo é que ele pode ser facilmente adaptado nos casos de grafos de estados com pesos para usar um DIJKSTRA, isso fica como exercício. Para completar a exposição vou apresentar uma versão do DFS para o espaço de estados, no caso a backtracking.

```
Algoritmo  DFS_STATE_GRAPH (s,T, n)
{
    input: O estado inicial s e o conjunto de estados finais T,
n é o numero de cliques até esse estado.
    output: sim se for possivel atingir algum elemento de T, não caso contrário.

    if(s está em T) return sim;

    vis[s]=1;
    for(cada v, estado que se pode chegar a partir de s com uma jogada) {
        if(!vis[v])
            if(DFS_STATE_GRAPH(v,T,n+1) == sim)
                return sim;
    }

    return não;
}
```

Você pode adaptar facilmente essa DFS para encontrar a rota mais curta e isso também fica como exercício. Como você deve ter visto o grafo de estado é uma ferramenta poderosa para resolver problemas envolvendo quebra-cabeças. Uma aplicação do BFS_STATE_GRAPH é no problema TRAFEGO do SPOJ BR e no problema SETEMARES. Grafo de estados parece ser uma constante em problemas da final mundial, em 2007 a primeira questão pode ser resolvida usando-se essa abordagem. Pois é, ai fica o algoritmo e a referência de problemas. Boa Sorte!!!

Postado por Péricles Lopes Machado às 10:09  Recomende isto no Google

6 comentários:

Thiago- 24 de maio de 2009 06:27

Sempre joguei xadrez e jogos de estratégia pra pc(alias foi um dos fatores que me fez escolher ciencia da computacao rs), as AI de xadrez atuais alem de usarem books de jogadas famosas usam o algoritmo de minmax(arvores binárias com profundidade) q parece um caso restrito desse tipo de grafos pois a complexidade do xadrez é muito alta devido a quantidade de jogadas possíveis. Mas usando grafo de estados deve dar pra implementar uma AI que só empata ou ganha em um jogo da velha com uma relativa facilidade(9! jogadas sao possíveis em um jogo da velha acredito eu). Continue postando, tenho certeza que vai servir de um ótimo material de referencia para mim quando for resolver esses problemas dificeis que mal tenho coragem de ler =]

Responder

Péricles Lopes Machado 24 de maio de 2009 09:48

Se quiser pode tentar o last year of marienbad é um bom começo. Além disso, a implementação não é complicada, no caso dos setemares. No trafego, o maior problema é a representação dos estados... Na verdade nesses problemas envolvendo grafos de estados, uma boa representação de cada estado é vital para um bom desempenho.

Responder

Péricles Lopes Machado 24 de maio de 2009 09:53

Implementar uma IA para jogar jogo da velha ou resolver um sudoku é um grande exercício para você se aprofundar no mundo dos algoritmos. Pode-se dizer que esses problemas são como ritos de passagens, rituais que separam os meninos dos homens[=P], no caso da programação. IA é fascinante e algoritmos são demais, o poder que eles oferecem é bem legal, o modo como ele expandi sua capacidade de analise de determinado problema.

Responder

Péricles Lopes Machado 24 de maio de 2009 10:06

No caso desses jogos de estratégia, a abordagem passa pelo grafo de estados, mas usasse heurísticas para limitar as arestas e o numero de vértices, no caso do xadrez creio que é comum podar o grafo para vértices que estão a mais de X jogados do estado atual. O mesmo deve acontecer nesses games de estratégia. No meu TCC estou pensando em implementar uma AI para jogar damas ou xadrez.

Responder

Thiago- 25 de maio de 2009 07:44

Usando heurísticas deve ser bom mesmo, acredito que jogos onde o MINMAX não tem um desempenho mto bom(um com muitos jogadores por exemplo), eu gosto bastante de AI e ciências cognitivas(mesmo conhecendo pouco)... acredito que independentemente do projeto que vc pretende fazer(simples ou não), vc aprende muita coisa =]

Se vc pretende criar o jogo de xadrez(a parte gráfica) recomendo usar a biblioteca SDL, eu tinha feito um até com os eventos de mouse mas acabou perdido dentre minhas formatações frequentes... é bem simples de se usar pra fazer um jogo 2d! Eu fiz logo que entrei na faculdade, foi interessante q melhorei bastante minha visão e conhecimento de algoritmos.

Um algoritmo pra ver se o rei está em cheque, roque, an peasant(typo?) sao algoritmos legais de se fazer ^^

Tb pretendo fazer algo relacionado a AI, é uma área nova q tem mta coisa pra se pensar e desenvolver =]

Responder

Renato Almeida 11 de outubro de 2009 06:37

Muito bom o tutorial, me ajudou a resolver o problema CUBO, do spoj :]
Valeu!

Responder

Digite seu comentário...

Comentar como: Conta do Googl

Publicar Visualizar

Postagem mais recente Início Postagem mais antiga

Assinar: Postar comentários (Atom)