

RELATÓRIO PARA PROJETO 1 DE INTELIGÊNCIA ARTIFICIAL

Grupo 87

David Sousa Batista 86405

Lucas Lobo Fell 86464

Introdução

O objetivo deste projeto é comparar a eficiência de diferentes métodos de procura da resolução de um problema, tendo sido escolhido uma variante do jogo “Solitaire” como exemplo. Foram escolhidos três algoritmos, dois dos quais que recorrem a estratégias de procura informadas. A procura não informada é realizada pelo algoritmo *DFS*, sendo que as procuras informadas são realizadas pelos algoritmos *Greedy* e *A**.

O algoritmo *Greedy* utiliza apenas a função de heurística (que a partir de aqui será denominada por $h(n)$), sendo que o algoritmo *A** utiliza tanto a função $h(n)$ como a função do custo do percurso (que a partir daqui será denominada por $g(n)$).

O $h(n)$ escolhido foi $\max(1, state.number_of_pegs - state.number_of_actions)$ em que *state* representa o estado atual, *number_of_pegs* representa o número de peças no tabuleiro, e *number_of_actions* representa o número de ações que são possíveis realizar também no tabuleiro atual. Foi feito o máximo entre “1” e a expressão apresentada pois este valor em princípio não deverá ser 0 ou inferior. O valor da função $h(n)$ também nunca deverá exceder o custo real de completar o tabuleiro, o que nunca acontecerá, pois, a parcela da direita terá no *state.number_of_pegs*.

O $g(n)$ escolhido foi $1/h(n)$ em que o estado utilizado é o do estado anterior à próxima ação.

Tabuleiro 5x5

Algoritmo	Nós expandidos	Nós gerados	Time (s)
DFS	12	13	0.06
Greedy	12	14	0.07
A*	10	12	0.06

O primeiro tabuleiro é um tabuleiro simples e com muitos espaços vazios, o que faz com que não existam muitas formas de ser resolvido. Qualquer um dos algoritmos chega à solução rapidamente e sem qualquer dificuldade, havendo uma ligeira diferença devido às funções utilizadas pelos algoritmos *Greedy* e *A** de forma a ordenar e a escolher o nó por onde continuar a procura em cada iteração.

Neste caso pode-se ver que $h(n)$ piora ligeiramente o desempenho, como se pode observar no algoritmo *Greedy*. Por outro lado, o *A** é mais eficiente o que significa que $g(n)$ melhora a procura.

O tempo levado a processar qualquer um dos algoritmos neste tabuleiro vai se acordo com o esperado.

Tabuleiro 4x4

Algoritmo	Nós expandidos	Nós gerados	Time (s)
DFS	5984	5985	0.74
Greedy	79	81	0.08
A*	88	90	0.10

A partir do segundo tabuleiro já é possível verificar significativamente o aumento de desempenho que a função $h(n)$ proporciona, utilizada tanto no *Greedy* como no *A**. Neste tabuleiro a função $g(n)$ tem um impacto negativo atrasando o algoritmo *A** face ao *Greedy*.

Os algoritmos de procura informada geram cerca de 70 vezes menos nós reduzindo apenas cerca de 8 vezes o tempo de processamento. Isto deve-se a uma maior simplicidade da função que implementa o *DFS* em

comparação com a função que implementa o *Greedy* e o *A**, e também aos cálculos adicionais realizados pelas funções $h(n)$ e $g(n)$.

Tabuleiro 4x5

<i>Algoritmo</i>	Nós expandidos	Nós gerados	Time (s)
<i>DFS</i>	53636	53637	8.24
<i>Greedy</i>	2081	2083	0.94
<i>A*</i>	1911	1913	0.71

Desta vez existe uma redução em cerca de 25 vezes o número de nós e uma redução em cerca de 10 vezes o tempo de execução entre as procura não informadas e as procura informadas. Mais uma vez a função $h(n)$ é aquela que cria uma otimização maior, sendo que $g(n)$ acelera apenas ligeiramente a execução. Podemos ver que em tabuleiros maiores a execução destes algoritmos se torna mais eficiente. Neste tabuleiro os algoritmos são cerca de 3.5 vezes mais eficientes em termos de redução de nós comparativamente à redução do tempo ($8x$ tempo / $70x$ nós para $10x$ tempo / $25x$ nós).

Tabuleiro 4x6

<i>Algoritmo</i>	Nós expandidos	Nós gerados	Time (s)
<i>DFS</i>	NA	NA	> 1800
<i>Greedy</i>	320	322	0.95
<i>A*</i>	37	39	0.09

Neste último tabuleiro é onde se pode observar a maior melhoria do algoritmo *A** face ao *Greedy*, que corre cerca de 10 vezes mais rápido com 8 vezes menos nós. Assim pode-se esperar que o algoritmo *A** seja mais eficiente em tabuleiros maiores e mais densos como é o caso deste. Não foi possível correr o algoritmo *DFS* neste tabuleiro pois após 30 minutos ainda não tinha descoberto a solução. Podemos, portanto, observar também que os algoritmos de procura informada foram muito superiores ao algoritmo *DFS*. Podemos concluir que o algoritmo *A** neste tabuleiro é mais de 20 mil vezes mais rápido que o algoritmo *DFS*.

Este é o tabuleiro onde as funções $h(n)$ e $g(n)$ tiveram mais impacto e mostraram uma melhoria mais acentuada.

Conclusão

As funções $h(n)$ e $g(n)$ dependem muito do tabuleiro onde estão a ser executadas pelo que nem sempre proporcionam uma otimização tão acentuada. Isto pode querer dizer que neste jogo não existe uma heurística perfeita para ser utilizada e por este motivo a que foi escolhida nem sempre funcionar tão bem, ou também pode querer dizer que a heurística escolhida poderia ser melhor que a atual. Entre a função $h(n)$ e a função $g(n)$ nota-se claramente que a $h(n)$ aumenta mais o desempenho do programa. Isto pode querer dizer que neste jogo é difícil calcular um custo de caminho pois qualquer jogada remove uma peça do tabuleiro, no entanto também é possível que simplesmente a função $g(n)$ escolhida não tenha sido a melhor.