

ESTUDO COMPARATIVO DOS MÉTODOS DE BUSCA POR LARGURA, PROFUNDIDADE, A* HAMMING E MANHATTAN PARA SOLUÇÃO DO 8-PUZZLE

COMPARATIVE STUDY OF THE METHODS BFS, DFS, A HAMMING AND MANHATTAN FOR THE 8-PUZZLE SOLUTION*

Paulo Roberto Quícoli - pauloquicoli@gmail.com

Raul Peres de Góes - raulperesgoes@gmail.com

Faculdade de Tecnologia de Taquaritinga (FATEC) – SP – Brasil

RESUMO

Dentro da área de computação observa-se muitas sub-áreas entre elas destaca-se a de inteligência artificial. Dentro dessa, nota-se os métodos de busca inteligentes. O objetivo deste estudo é comparar o desempenho de quatro desses, sendo eles: busca por largura, profundidade, A* Hamming e A* Manhattan para a solução do 8-puzzle. Para implementação foi utilizado a linguagem Java. Os resultados obtidos indicaram que os métodos de busca por largura e ambos A* sempre encontraram a melhor solução. O método A* Manhattan sempre obteve os melhores resultados, seguido do A* Hamming. A busca por profundidade somente encontrou a solução para o primeiro caso testado, sendo não significativo para os demais testes. Através do estudo concluiu-se que os métodos baseados em heurísticas são mais complexos para implementar, porém apresentam performance significativamente melhor que os demais métodos. A busca por largura, apesar de não ser otimizada se mostrou relativamente eficiente para solução de problemas mais simples.
Palavras-chave: Busca. Largura. Profundidade. Hamming. Manhattan.

ABSTRACT

In the area of computing it could be observed many sub-areas in which the artificial intelligence has called attention. In that, it could be noticed the intelligent methods of search. The aim is of this article is to evaluate the performance of four algorithms the breadth-first search, depth-first search, A* Hamming and A* Manhattan for solving the 8-puzzle. Java was the language used to implement them. The results pointed that the methods breadth-first search and both A* always found the best solution. The method A* Manhattan always had the best performance fallowed by A* Hamming. The depth-first search only found the solution for the first case of use. Finally, it was concluded that the methods based on heuristics are more complex to implement, however they presented a significantly better performance

compared to the others. Although the breadth-first search is not optimized, it showed a good performance when solving simpler problems.

Keywords: Search. Breadth. Depth. Hamming. Manhattan.

COMO REFERENCIAR ESTE ARTIGO:

QUÍCOLI, P.R.; GÓES, R.P. Estudo comparativo dos métodos de busca por largura, profundidade, A* Hamming e Manhattan para solução do 8-puzzle. **In: III SIMTEC – Simpósio de Tecnologia da FATEC Taquaritinga**. Disponível em: <www.fatectq.edu.br> 10 p. Outubro de 2015.

1 INTRODUÇÃO

Neste artigo é proposta uma análise de desempenho de quatro algoritmos de busca inteligentes sendo eles, busca por largura, profundidade, distância Hamming e Manhattan para o 8-puzzle. Esse é um jogo de quebra-cabeça com 9 posições e 8 números, sendo que o objetivo é organizar os números em uma sequência pré-definida. Os números só podem interagir com o espaço vazio da mesa e destaca-se o fato de não ser permitido movimentos diagonais.

Todos os algoritmos propostos têm o mesmo princípio de funcionamento, que segundo Russell (2004) resume-se em considerar todas as possibilidades de movimento a partir de um estado inicial definido. Portanto, todos os estados intermediários são os possíveis caminhos a serem percorridos entre o estado inicial e o estado final.

A busca por largura é uma solução inteligente, porém não otimizada, para o problema do 8-puzzle. Segundo Luger (2004) essa explora cada nível do espaço amostral, ou seja, a busca somente segue para próximo nível após ter explorado todas as possibilidades do nível anterior. Esse método também pode ser chamado de busca por amplitude ou extensão.

A busca em profundidade segundo Luger (2004) explora todos os filhos de um estado e somente após estes testes os irmãos daquele primeiro estado é testado. Ou seja, este método explora o nó mais profundo (último nível) da árvore e no caso de não encontrar a solução a busca retorna para o nó mais raso da árvore para explorar todos os filhos deste e assim por diante.

Segundo Jensen (2013) os algoritmos inteligentes, A*, são baseados em heurísticas que são basicamente um conjunto de regras que gerenciam a busca. Para a solução do 8-

puzzle foi utilizado dois métodos de busca por distância, Hamming e Manhattan. O primeiro considera os números fora da posição correta e para realizar o movimento considera como melhor cenário a solução com menos números fora de posição somado da profundidade da árvore. O segundo além de considerar os números fora de posição, também calcula a quantidade de movimentos necessários para move-los para a posição correta somado da profundidade da árvore. Nesse caso também se considera como a melhor opção a situação que precisará da menor quantidade de movimentos possíveis para ser resolvida.

2 DEFINIÇÃO DO PROBLEMA

O objetivo do algoritmo é solucionar o 8-puzzle. A premissa para a solução é que todo número pode trocar posição exclusivamente com o espaço vazio (representado por 0). Nota-se que através dos métodos de busca a repetição de um estado de jogo já testado não deve acontecer.

- **Estado inicial**: Pode ser qualquer combinação de números entre 0-8, onde 0 representa vazio.
- **Estado final:** O estado final geralmente é caracterizado pela sequência [1,2,3,4,5,6,7,8,0], mas pode ser diferente desta desde que não seja igual ao estado inicial.
- **Espaço amostral:** é uma matrix 3x3.
- **Ações**

1. Movimentar. Um movimento é considerado válido:

a. Quando um dos números trocado é o zero.

b. Não é permitido movimentos na diagonal.

c. Quando a nova posição assumida pelo número está dentro do limite amostral.

2. Imprimir a solução do problema.

- **Custo funcional do caminho:** Pode ser considerado o número de interações bem como o custo das heurísticas para os métodos AStar.
- **Objetivo:** É alcançado quando o estado dos números na mesa corresponde ao estado final desejado.

3 IMPLEMENTAÇÃO

3.1 Lógica

Todos os métodos dependem de um movimento válido. Para demonstração de um movimento válido foi usado as tabelas 1 e 2, onde elas representam os endereços da mesa e os números em cada posição, respectivamente. O número da esquerda representa as linhas e os da direita as colunas, na figura 1.

Figura 1 – Representação das linhas e colunas

0,0	0,1	0,2
1,0	1,1	1,2
2,0	2,1	2,2

Tabela 1 – Representação de um cenário real de jogo.

0	1	2
3	4	5
6	7	8

De acordo com a situação acima, tem-se:

Tabela 2 – Relação Figura 1 x Tabela 1

Linha	Coluna	Valor
0	0	0
0	1	1
0	2	2
1	0	3
1	1	4
1	2	5
2	0	6
2	1	7
2	2	8

A lógica da implementação para validação de um movimento é baseada em um laço duplo de repetição. A primeira estrutura representa a linha e a segunda as colunas para aquela linha. A validação do movimento é feita através da identificação da posição do número 0. Uma vez encontrado adiciona-se e subtrai-se 1 no valor da linha para testar se é possível movimentar para cima e para baixo. O mesmo teste é realizado para a coluna, porém nesse

caso é testado a possibilidade de mover-se para direita e esquerda. O movimento é valido se o valor adicionado for menor que 3 ou se o valor subtraído for no mínimo 0.

Para cada possível movimento é criado um objeto do tipo Mesa que é preenchido com os seguintes atributos:

- Posição da mesa: integer[9] (próximo movimento)
- Profundidade: integer
- Pai: Mesa (mesa corrente)
- Proximo: Mesa (utilizado nos métodos inteligentes para identificação da melhor opção)
- Solucao: integer[9]
- Descricao: string (descrição da direção do movimento)

Após a criação do objeto é realizado um teste para certificar-se se o mesmo foi testado anteriormente e caso não aquele é adicionado à lista de nós não explorados.

3.2 Heurísticas

O método AStar foi implementado utilizando-se 2 heurísticas diferentes: a distância de Hamming, o qual é considerado a quantidade de números fora da posição correta. O segundo a distância Manhattan, o qual considera, para cada número fora de posição, quantos movimentos serão necessários para posicioná-lo corretamente.

Em ambos métodos é adicionado a profundidade (nível) do movimento. Então tem-se:

- Hamming, por exemplo:
  - h(n): 1+1+0+1+0+0+0+1+1+3(p) = 8, onde:
  - Número fora da posição: +1
  - Número posicionado corretamente: +0
  - p: Profundidade

- Manhattan, por exemplo:
  - h(n) = cm + p, onde:
  - cm: custo movimento (quantos movimentos são necessários para posionar corretamente o número n)
  - p: Profundidade

O custo movimento é calculado através da seguinte formula:

- cm = [|(LinhaCorreta - LinhaAtual)|] + [|(ColunaCorreta - ColunaAtual)|]

4 RESULTADOS

Os jogos foram testados e avaliados segundo os parâmetros: tempo de processamento em segundos, tamanho das listas, quantidade de movimentos necessários para solucionar o jogo e quantidade de interações.

Foram testados 3 estados iniciais diferentes:

1.
1	2	5
3	0	4
6	7	8

2.
1	2	0
3	4	5
6	7	8

3.
3	2	0
6	1	5
7	4	8

Foram testados 2 estados finais diferentes:

1.
0	1	2
3	4	5
6	7	8

2.
1	2	3
4	5	6
7	8	0

4.1. Teste jogo 1

Resultados para solução 1:

Tabela 3 – Resultado para jogo 1 solução 1

	Movimentos	Tempo(ms)	Lista Explorada	Lista Não Explorada	Interações
Largura	2	11	7	7	7
Profundidade	2	9	2	3	2
AStar Hamming	2	8	2	3	2
AStar Manhattan	2	9	2	3	2

O primeiro jogo para a primeira solução (tabela 3) é a situação mais simples possível. Todos os algoritmos foram capazes de resolver o jogo com a quantidade otimizada de movimentos. Nota-se que a busca por largura precisou de quase 4 vezes mais interações para solucionar o problema enquanto a busca por profundidade teve comportamento otimizado se equiparando com os métodos AStar. Pela simplicidade do jogo a diferença obtida no tempo não é significativa.

Resultados para solução 2:

Tabela 4 – Resultados para jogo 1 estado final 2

	Movimentos	Tempo(ms)	Lista Explorada	Lista Não Explorada	Interações
Largura	22	265542	76041	23639	76041
Profundidade	43300	154771	47421	30414	47421
AStar Hamming	22	3808	8279	4561	8279
AStar Manhattan	22	121	1291	783	1291

O contexto do primeiro estado inicial para o segundo estado final (tabela 4) é muito mais complexo. Ainda assim, as buscas por largura, AStar Hamming e AStar Manhattan obtiveram sucesso ao solucionar o jogo em 22 movimentos (solução ótima). Comparando-se tempo nota-se uma diferença significativa da busca por largura em relação aos métodos AStar sendo aquele aproximadamente 6x mais lento que o Hamming e 2000x o Manhattan. A busca por largura ainda precisou de 9x mais interações que o Hamming e de aproximadamente 70x que o Manhattan.

Comparando-se os dois métodos AStar, conclui-se que o Manhattan foi aproximadamente 30x mais rápido e que o mesmo precisou de 6x menos interações para

solucionar o problema. Nesse cenário os resultados obtidos pela busca por profundidade foram não significativos.

4.2 Teste jogo 2

Resultados para solução 1:

Tabela 5 – Resultados para jogo 2 e solução 1

	Movimentos	Tempo(ms)	Lista Explorada	Lista Não Explorada	Interações
Largura	4	17	78	144	78
Profundidade	52640	729848	132553	35895	132553
AStar Hamming	4	7	4	6	4
AStar Manhattan	4	8	4	6	4

O segundo teste para a primeira solução (tabela 5) é resolvido com 4 movimentos. Os algoritmos de busca por largura e ambos A* resolveram o jogo, porém como esperado a busca por largura precisou de mais interações. Considerando tempo os algoritmos A* tiveram melhor desempenho sendo 2x mais rápido que a busca por largura, porém pela simplicidade do teste o tempo demorado para resolver não é significativo. Os resultados obtidos pelo teste de profundidade foram inconclusivos.

Resultados para solução 2:

Tabela 6 – Resultados para jogo 2 e solução 2

	Movimentos	Tempo(ms)	Lista Explorada	Lista Não Explorada	Interações
Largura	22	361598	96672	22252	96672
Profundidade	20218	1125625	165202	15352	165202
AStar Hamming	22	5391	10131	5415	10131
AStar Manhattan	22	458	2270	1303	2270

O segundo jogo para a segunda solução (tabela 6) foi resolvido com 22 para a busca com largura, e ambos A*, o que confirma o fato de que essa é a solução ótima. Porém, como esperado, a mesma precisou de aproximadamente 10x mais interações que o A* Hamming e 20x mais que o A* Manhattan. Avaliando-se tempo observa-se uma diferença significativa de

performance comparando os métodos, sendo o A* Manhattan aproximadamente 11x mais rápido que o A* Hamming e 72x mais rápido que a busca por largura. O método de busca por profundidade não foi capaz de encontrar a solução do problema.

4.3 Teste jogo 3

Resultados para solução 1:

Tabela 7 – Resultados para jogo 3 e solução 1

	Movimentos	Tempo(ms)	Lista Explorada	Lista Não Explorada	Interações
Largura	6	46	507	893	507
Profundidade	62994	419761	79609	40950	79609
AStar Hamming	6	8	6	7	6
AStar Manhattan	6	11	6	11	6

Os métodos A* apresentaram comportamento semelhante, tabela 7. Ambos resolveram o puzzle com 6 movimentos, porém ambos com 6 interações o que indica que a heurística funcionou com 100% de precisão. A busca por largura também resolveu o jogo, porém precisou de cerca de 80x mais interações chegando a 507, e consequentemente demorou-se 4x mais em termos de tempo. Apesar da diferença em tempo, na prática essa é mínima. O método de busca de profundidade não foi capaz de resolver o problema.

Resultados para solução 2:

Tabela 8 – Resultados para jogo 3 e solução 2

	Movimentos	Tempo(ms)	Lista Explorada	Lista Não Explorada	Interações
Largura	22	299372	91138	21321	91138
Profundidade	9740	6307	10116	7565	10116
AStar Hamming	22	3608	8307	4602	8307
AStar Manhattan	22	302	2035	1209	2035

Para o terceiro jogo e segunda solução (tabela 8) observou-se que o método A* Manhattan obteve melhor desempenho sendo 10x mais rápido que o A* Hamming e 90x mais rápido que a busca por largura. Aquele método ainda precisou de 4x menos interações que o A* Hamming para resolver o puzzle e 45x menos interações que a busca por largura. A busca por profundidade novamente não foi capaz de solucionar o problema.

5 CONCLUSÃO

A busca por largura pode ser considerada um método simples, mas eficaz. Entretanto, pode-se também concluir que para problemas mais complexos essa solução não apresentaria um resultado de performance, em termos de tempo, satisfatório, apesar de possivelmente encontrar a solução. O uso de heurísticas permitiu um desempenho muito melhor em termos de tempo sendo que o A* Manhattan em todos os cenários apresentou melhor desempenho. Em todos os cenários testados os métodos de busca por largura e A* encontraram a solução ótima. A busca por profundidade solucionou apenas um cenário dos seis propostos, porém com desempenho semelhante ao dos A*.

REFERÊNCIAS

LUGER, GEORGE F. Inteligência Artificial. Estruturas e estratégias para a solução de problemas complexos. 4.ed. Porto Alegre: Editora Bookman, 2004.

JENSEN, R. 2013. Lecture 4: Lecture 6: Informed search. [Lecture to BEng Software Engeneering Year 2]. Aberystwyth University, October 2013.

RUSSELL, STUART; NORVIG, PETER. Inteligência Artificial. 3.ed. Rio de Janeiro: Editora Campus, 2004.