Relatório Projeto Inteligência Artificial 2021/2022

Grupo: al004

Aluno(s): Miguel Eleutério (99287) e Raquel Cardoso (99314)

O problema apresentado é a resolução de um *Takuzu*, também conhecido como Sudoku Binário, com recurso a técnicas de procura aprendidas durante a unidade curricular. Na solução apresentada pelo nosso grupo utilizámos o algoritmo de procura em profundidade primeiro. Mais especificamente:

Após ler o nosso tabuleiro inicial, filtramo-lo de forma a encontrar todas as posições vazias que sabemos que terão um certo número, poupando tempo de procura. Essas posições serão:

- as posições que constituem um tuplo tal que: (2, x, x), (x, 2, x) ou (x, x, 2), sendo x sempre 1 ou 0. Em qualquer uma sabemos que o valor 2 será obrigatoriamente o contrário ao x. Isto pode acontecer verticalmente ou horizontalmente.
- as posições que pertencem a uma coluna ou linha que já tem o número máximo permitido de 1 ou 0. Qualquer uma destas posições terá o valor díspar.

O nosso problema passará a ser um tabuleiro da classe *Takuzu*. Iremos aplicar a procura de profundidade ao mesmo. A função *actions* será a responsável por devolver, um a um, os valores filtrados acima, até já não existirem. De seguida, irá verificar os restantes números por preencher no tabuleiro. Ou seja, irá aplicar as restrições necessárias para a resolução correta do nosso problema e devolver as ações possíveis para cada posição, uma a uma. Essas mesmas restrições são:

- as restrições descritas acima
- nenhuma linha ou coluna podem ser iguais;

Para a verificação da última restrição é criada uma cópia do tabuleiro atual e colocamos um 1 ou um 0 na posição a avaliar. Caso o tabuleiro deixe de ser válido após a verificação, descartamos a ação, caso continue a ser válido, continuamos. No final, devolvemos a lista das ações possíveis para a posição. Se não existir nenhuma ação possível, devolvemos uma lista vazia.

Quando o nosso tabuleiro não tiver uma única posição com valor 2, podemos garantir que o problema estará resolvido.

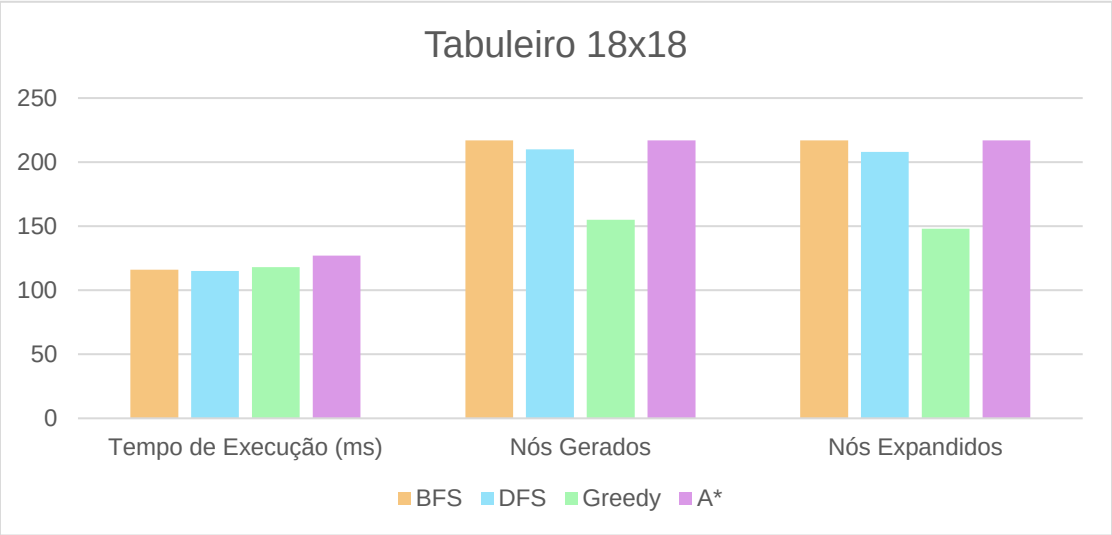

Figura 1: gráficos de barras para comparar o desempenho de diferentes algoritmos de procura num tabuleiro *Takuzu* 18x18.

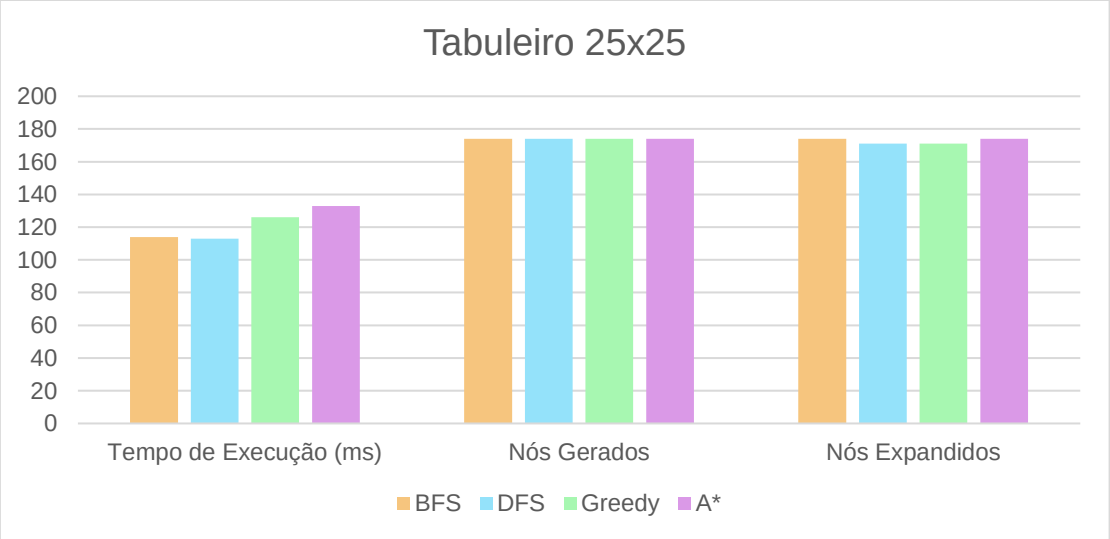

Figura 2: gráficos de barras para comparar o desempenho de diferentes algoritmos de procura num tabuleiro *Takuzu* 25x25.

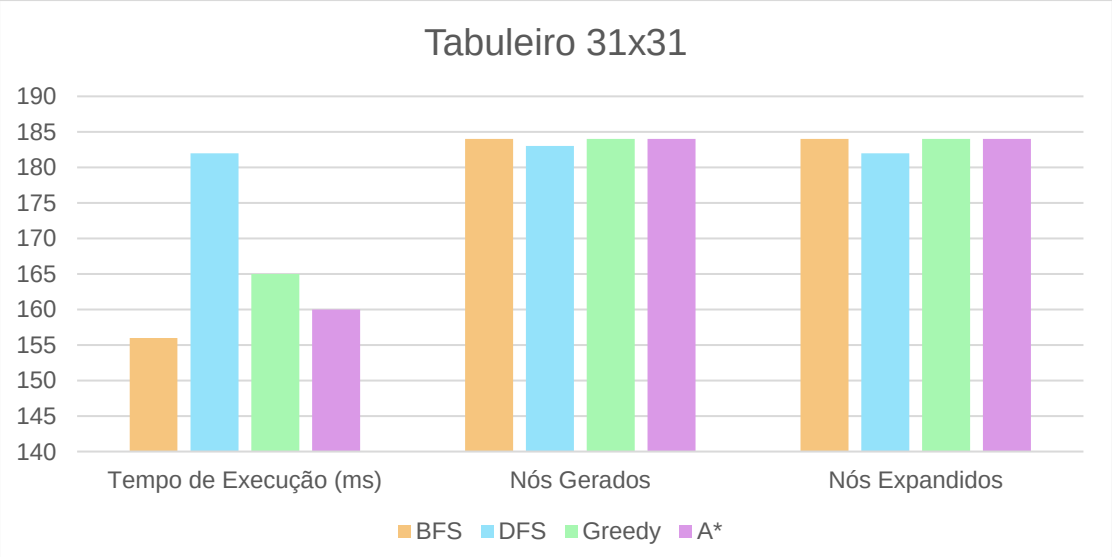

Figura 3: gráficos de barras para comparar o desempenho de diferentes algoritmos de procura num tabuleiro *Takuzu* 31x31.

1. Quanto ao tabuleiro 18x18, o algoritmo que gera e expande menos nós é o Greedy; o mais rápido, no entanto, é a DFS.
2. Quanto ao tabuleiro 25x25, todos os algoritmos geram o mesmo número de nós. Os algoritmos que expandem menos nós são a DFS e Greedy. O algoritmo mais rápido é a DFS.
3. Quanto ao tabuleiro 31x31, o algoritmo que gera e expande menos nós é a DFS; o algoritmo mais rápido, no entanto, é A*.

Podemos verificar que em tempos de tempo a DFS é a melhor, no entanto, a Greedy apresenta o menor número de nós gerados e expandidos, exceto no Tabuleiro 31x31, cuja diferença é negligenciável. No entanto, esta deve-se ao tempo de cálculo da nossa heurística e caso seja possível melhorá-lo, a Greedy seria a melhor escolha sempre.

Uma opção de heurística seria o número de posições por preencher num tabuleiro, no entanto, esta é uma heurística má, uma vez que:

1. Ao substituir uma posição vazia, serão sempre gerados no máximo dois nós, que segundo esta heurística, teriam o mesmo valor de h(n).
2. A heurística não ajuda necessariamente a encontrar uma solução, uma vez que ela considerará que a resposta estará mais perto, mesmo partindo de uma pressuposição errada.

De forma a resolver o problema 1, a heurística não pode depender se é inserida ou não uma posição, sendo então baseada no valor colocado. Logo, a heurística escolhida é a diferença entre a metade do número de posições totais do tabuleiro (número máximo de 1's e 0's) e o maior entre o número de 1's ou 0's já colocados no mesmo. Deste modo, o nosso algoritmo tenderá a preencher primeiro um certo número e apenas depois o outro, acelerando o processo.

2	2	0	1
1	0	2	1
0	2	1	0
1	2	2	2

Figura 4: Exemplo de Tabuleiro *Takuzu* inicial 4x4.

Número de 1's: 5
Números de 0's: 4
Tamanho: 16

Número máximo de 1's/0's: 16/2 = 8
MAX(número de 1's, número de 0's): 5
Heurística: 8 – 5 = 3