FRENCH SOLITAIRE

ARTIFICIAL INTELLIGENCE

	PROCURA EM PROFUNDIDADE PRIMEIRO	PROCURA GANANCIOSA	PROCURA A*
Tempo de execução	0.00115871s	0.01324868s	0.04985308s
Número de nós expandidos	30	93	42
Número de nós gerados	38	101	52

Tabela 1: Tabuleiro 5x5 (linhas x colunas).

	PROCURA EM PROFUNDIDADE PRIMEIRO	PROCURA GANANCIOSA	PROCURA A*
Tempo de execução	0.14178323s	0.02150702s	0.08331990s
Número de nós expandidos	6076	72	136
Número de nós gerados	6094	117	229

Tabela 2: Tabuleiro 4x4 (linhas x colunas).

	PROCURA EM PROFUNDIDADE PRIMEIRO	PROCURA GANANCIOSA	PROCURA A*
Tempo de execução	1.517438173s	0.15112662s	0.50763177s
Número de nós expandidos	53946	757	757
Número de nós gerados	53974	1715	1709

Tabela 3: Tabuleiro 4x5 (linhas x colunas)

	PROCURA EM PROFUNDIDADE PRIMEIRO	PROCURA GANANCIOSA	PROCURA A*
Tempo de execução	4m49167783s	2.13414907s	1.42363953s
Número de nós expandidos	7262371	6642	1431
Número de nós gerados	7262431	18700	2842

Tabela 4: Tabuleiro 4x6 (linhas x colunas).

Após uma análise cuidada das tabelas acima, é possível concluir que quanto maior for o número de peças em jogo, maior será o número de nós gerados e expandidos. Por exemplo, olhando para os resultados do algoritmo de procura em profundidade na **Tabela 1** verifica-se que o tempo é menor quando existe um menor número de nós gerados e expandidos, causados por uma diversidade de jogadas possíveis mais pequena enquanto que na **Tabela 2**, quando existe um tabuleiro com maior flexibilidade de jogadas, maior será o tempo de execução, pois haverão mais nós gerados e alguns destes sucessivamente expandidos.

Comparativamente aos algoritmos estudados, observa-se que, embora todos encontrem soluções, efetuam-no com eficiências distintas, principalmente quando a complexidade de um tabuleiro aumenta. Em relação às diferenças entre o Greedy e o A*, ambos apresentam uma boa performance, em termos de eficiência em tempo útil, apesar de o algoritmo A* ter vantagem relativamente ao Greedy. Quando os casos se tornam mais complexos, a eficiência da DFS baixa muito quando comparada com a do algoritmo Greedy e A*, tornando-se quase impossível a execução com o tabuleiro da **Tabela 4** em tempo útil. Pelos valores recolhidos, considera-se o algoritmo de procura gananciosa como o mais eficiente. Seguidamente o de procura A* e por fim o algoritmo de procura em profundidade primeiro. É possível comprovar isto consultando as **Tabelas 1**, **2** e **3**. Na última tabela, **Tabela 4**, verifica-se uma maior eficiência do algoritmo de procura A* explicada pela menor quantidade nós gerados e expandidos.

Para realizar este projeto optou-se por utilizar uma heurística que avalia quantas peças existem no tabuleiro e quantas jogadas é possível concretizar de acordo com um certo estado desse mesmo tabuleiro. Baseia-se, portanto, numa relação de adição entre o número de peças que existem num dado estado do tabuleiro e o número de jogadas possíveis para um dado estado do tabuleiro, tendo por objetivo restringir as posições que não têm peças ou que estão bloqueadas de forma minimizar esse valor para atingir a solução do jogo, caso esta exista.