Relatório Takuzu – IA P4 2021/2022

Grupo: al017

Alunos: Francisco Lopes (ist 99220) e Nuno Martins (ist 99292)

1. Descrição do Problema e da Solução proposta

O objetivo deste relatório é apresentar uma solução para o problema Takuzu, também denominado Puzzle Binário, no qual existe uma grelha quadrada N x N, em que cada célula pode conter os números 0 ou 1, seguindo um conjunto de restrições das quais: existe um número igual de 1s e 0s em cada linha e coluna (ou mais um para N ímpar), não existem mais que dois números iguais em posições adjacentes e em que todas as linhas e colunas são diferentes.

A nossa solução formaliza o problema Takuzu como um problema de procura, em que cada estado é representado por uma matriz N x N (state.board), sendo o estado inicial um tabuleiro válido dado como input em que algumas das suas células já se encontram preenchidas. Em cada estado, existem um conjunto de ações que são passíveis de ser realizadas, correspondendo a um tuplo '(row,col,value)', em que 'row' é o número da linha, 'col' é o número da coluna e 'value' é o valor (0 ou 1) a colocar nessa posição, posição essa que tem de ser obrigatoriamente vazia e que na nossa solução representamos por value=2.

Na obtenção das ações, tivemos como prioridades as jogadas óbvias, isto é: se a posição atual é vazia e as adjacentes são iguais, então a atual tem de ser o valor contrário às adjacentes (por contrário, significa que se for 0 colocamos 1, se for 1 colocamos 0); se a posição atual está preenchida e se uma das adjacentes tiver o mesmo valor e a outra adjacente for vazia, então essa vazia tem de ter valor contrário à atual; se uma linha ou coluna tiver um total de N//2 ou N//2+1 (se N ímpar) valores 0s ou 1s preenchidos, então qualquer posição vazia nessa linha/coluna deve ser preenchida pelo valor contrário. Após estarem esgotadas estas ações óbvias, simplesmente escolhemos as restantes posições vazias, por ordem crescente de linhas e colunas.

Após estar escolhida uma ação, é criado um novo estado (state), copiando a grelha antiga para uma nova grelha e inserido na posição escolhida na ação, o valor da mesma. De forma a guiar eficientemente a procura, foi criada uma simples heurística que retorna o número de linhas e colunas que se encontram com posições para preencher.

Por último, em cada estado é verificado se foi atingido o estado objetivo, isto é, se o tabuleiro (state.board) não tem posições vazias e se respeita as restrições impostas e descritas no final do primeiro parágrafo deste relatório. Se de facto for o estado objetivo, então a procura foi terminada com sucesso e é impresso o tabuleiro preenchido (como uma string) para o standard output.

2. Teste da Solução e Análise dos Resultados obtidos

Usando os testes de input e ouput fornecidos pelo corpo docente da cadeira de Inteligência Artificial, e após aplicar a nossa solução proposta para resolver o problema, foram obtidos os seguintes resultados:

	Tempo de Execução (s)				Número de Nós Gerados				Número de Nós Expandidos			
Input Test	BFS	DFS	Greedy	A*	BFS	DFS	Greedy	A*	BFS	DFS	Greedy	A*
1	0,0013	0,0013	0,0014	0,0014	7	7	7	7	7	7	7	7
2	0,018	0,0018	0,0019	0,0019	7	7	7	7	7	7	7	7
3	0,0214	0,0152	0,0165	0,0210	65	49	51	62	62	42	45	57
4	0,0145	0,0132	0,0142	0,0142	37	35	35	35	35	32	32	32
5	0,1186	0,0313	0,0324	0,1056	234	81	76	203	211	58	57	176
7	0,0608	0,0584	0,0610	0,0614	69	69	69	69	69	69	69	69
8	0,0179	0,0181	0,0184	0,0185	19	19	19	19	19	19	19	19
9	0,1794	0,1801	0,1835	0,1865	139	139	139	139	139	139	139	139
10	0,2981	0,2940	0,3035	0,3045	184	184	184	184	184	184	184	184
11	0,3113	0,3070	0,3246	0,3198	180	180	180	180	180	180	180	180
12	0,4002	0,4019	0,4419	0,4111	166	166	166	166	166	166	166	166
13	0,6388	0,6389	0,6521	0,6554	180	180	180	180	180	180	180	180

Com a análise da tabela acima, podemos verificar que todos os métodos são bastante parecidos em termos de tempo de execução, excepto no caso do Input Test 3 e 5, no qual a DFS e a Greedy Search são bastante superiores. Relativamente ao número de nós gerados, a BFS e A* são relativamente piores no Input Test 5, sendo no geral bastante parecidos aos outros métodos nos restantes testes. No caso do número de nós expandidos, novamente a BFS e A* são piores no Input Test 5, e DFS é claramente superior no Input Test 3 comparativamente aos restantes métodos.

Esta diferença poderá eventualmente ser explicada pelo facto da heurística usada não ser a mais ideal (heurística usada nestes resultados foi o número de linhas e colunas por preencher, isto é, que contêm no mínimo 1 posição livre), mas nestes testes dados como input, concluimos que a DFS é a melhor opção para procura da solução do problema.