Same Game Solver

Procura e Planeamento - 2019/2020

Grupo 011
João Palet - 86447
Simão Nunes - 86512

1 Introdução

Com este relatório pretendemos expor os detalhes da nossa proposta de solução ao projeto da unidade curricular de Procura e Planeamento. Foi-nos proposto a elaboração de um programa capaz de resolver uma variedade de tabuleiros do jogo Same Game de forma a implementar diferentes estratégias de procura estudadas na cadeira.

Nas seguintes secções deste relatório, explicitamos as abordagens utilizadas, estruturas desenvolvidas e alternativas implementadas.

2 Abordagem ao Problema

De forma a combater a explosão de combinatória gerada pelo problema em questão, implementámos uma solução que nos permite analisar as diferentes possibilidades através de uma procura em árvore. Para isso, é necessário que a partir de um tabuleiro inicial se gerem sucessores de uma forma sequencial até encontrarmos um estado que satisfaça o interesse do jogador, que neste caso é atingir a maior pontuação possível.

Ao longo de toda a execução da procura, o nosso programa é capaz de manter um registo do estado que representa a maior pontuação obtida até então. Desta forma, possibilitamos que outros ramos da árvore sejam explorados, sem comprometer a melhor solução encontrada até ao momento.

Esta abordagem possuí duas condições de paragem. Assim, o melhor estado é retornado quando:

- Não restam mais nós para explorar
- A duração da procura atinge um limite de tempo pré-definido

3 Estruturas de Dados

Para descrever um estado da procura, optámos por usar uma estrutura com os seguintes campos:

- Tabuleiro
- Pontuação obtida na última jogada
- Pontuação total acumulada até ao momento
- Todas as jogadas efetuadas até ao momento

O tabuleiro é representado por uma lista de listas, a sua representação externa. Para calcular o custo de geração de um estado, é calculado o inverso da pontuação obtida na jogada que conduziu ao estado em questão. Deste modo, estados que são obtidos a partir de jogadas com pontuações elevadas têm um custo de geração mais baixo. Sempre que um novo estado é gerado, este é testado contra o melhor estado descoberto pela procura até até ao momento. Se o novo estado tiver atingido uma pontuação mais alta, passa a ser ele o melhor estado. Esgotado o tempo limite, é selecionado o melhor estado e é retornada a lista das jogadas efetuadas para o atingir.

Para a representação interna de uma jogada, criámos uma estrutura ponto com dois campos: a linha e a coluna onde a jogada foi feita.

4 Heurísticas

Utilizámos um conjunto de heurísticas que acreditamos que possam maximizar a pontuação total em cada momento de exploração de nós. Assim, ao longo da procura, utilizamos as seguintes heurísticas:

- Heurística de maior grupo

Esta heurística beneficia tabuleiros cujo seu maior grupo de peças seja o maior possível e que, portanto, poderão produzir uma maior pontuação assim que essa jogada for efetuada.

- Heurística do menor número de peças isoladas

Uma vez que estas peças não são capazes de produzir jogadas no tabuleiro, utilizamos esta heurística para favorecer tabuleiros que contenham uma maior variedade de jogadas possíveis.

- Uma combinação das duas heurísticas anteriores

De forma a potencializar a solução obtida, elaborámos uma heurística mista que permite combinar as duas abordagens anteriores.

Cada uma destas heurísticas foi testada devidamente e os resultados estão detalhados na secção 7.

5 Estratégias de Corte

De modo a atingirmos uma maior performance da procura, implementamos duas estratégias de corte que permitem reduzir a explosão combinatória inerente ao problema:

- Agrupar numa só jogada as peças adjacentes que produzem tabuleiros semelhantes

Visto que é indiferente em que posição do grupo é efetuada a jogada, dado que os tabuleiros resultantes serão os mesmos, apenas são consideradas as jogadas na posição mais acima e mais à esquerda, respetivamente, de cada grupo.

- Atribuir um fator de ramificação no momento de exploração de sucessores

Após gerar todos os sucessores de um estado, já tento em conta a primeira estratégia de corte mencionada, estes são ordenados do melhor para o pior tendo em conta o valor heurístico de cada sucessor e a pontuação acumulada até ao momento[1]. Desta lista ordenada, apenas os X primeiros sucessores são considerados, onde X é o fator de ramificação máximo definido no início da procura.

6 Abordagem Alternativa

Como abordagem alternativa, implementámos uma estratégia híbrida *A** - *Sondagem Iterativa* de modo a tentar encontrar um bom compromisso entre tempo de execução e memória utilizada. O algoritmo começa por realizar uma procura *A** a partir da raiz da árvore por considerarmos que as jogadas iniciais podem ser vir a ser decisivas para atingir uma boa pontuação no jogo. Após um certo de limite de tempo, a procura é retomada com uma *Sondagem Iterativa* a partir do melhor nó, segundo um critério de pontuação e valor heurístico[2], atingido pela procura *A**. Ao fim do tempo limite imposto, a procura por *Sondagem Iterativa* retorna o estado com maior pontuação que encontrou, tal como explicado em 2.

7 Resultados Experimentais

Para os próximos resultados, todas as procuras terminam dentro de um tempo limite de 5 minutos, retornando as jogadas que maximizam a pontuação atingida até ao momento. Usámos o tabuleiro de dimensões 15x10 com 3 cores presente no enunciado por ser complexo o suficiente para que não seja possível explorar todo o espaço de estados dentro do tempo limite.

[1] O ficheiro submetido tem uma gralha onde apenas a pontuação jogada usada para atingir o nó é considerada, ao invés da pontuação acumulada até ele.

[2] O ficheiro submetido tem uma gralha onde apenas a pontuação do tabuleiro é considerada, ao invés de uma combinação da pontuação acumulada e da heurística do estado.

Devido às duas gralhas mencionadas nas secções anteriores, os resultados experimentais não foram de acordo ao esperado caso as procuras tivessem sido implementadas como previsto. Muito por causa disso, a estratégia híbrida $A*$ - *Sondagem Iterativa* não se destacou em relação às restantes como esperado, apresentando resultados bastante abaixo da *Sondagem Iterativa*.

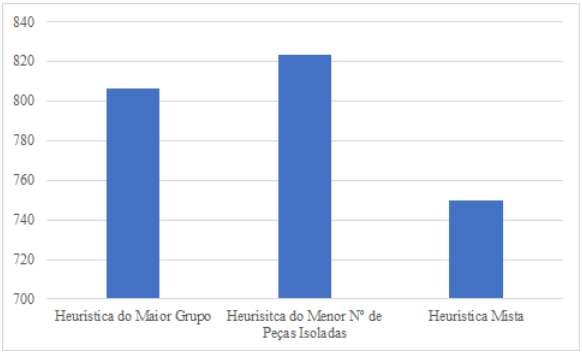

Figure 1: Pontuação Atingida por cada heurística usada na procura $A*$.

Como podemos observar na figura 1, a heurística do menor número de peças isoladas foi a que obteve o melhor desempenho em relação à pontuação final. A heurística do maior grupo de peças adjacentes também atingiu valores elevados. É curioso notar que a heurística mista não obteve uma boa performance apesar de ser uma combinação de duas heurísticas com um bom desempenho individual.

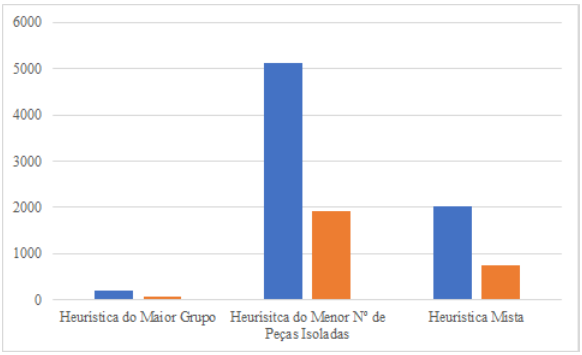

Figure 2: Nós gerados (a azul) e nós expandidos (a laranja) em cada heurística na procura $A*$.

Em relação ao número de nós gerados/explorados, observamos que a heurística do menor número de peças isoladas apresenta o pior desempenho. A heurística do maior grupo apresentou números de nós expandidos e gerados drasticamente menores que as outras, caso para o qual não conseguimos apresentar uma justificação.

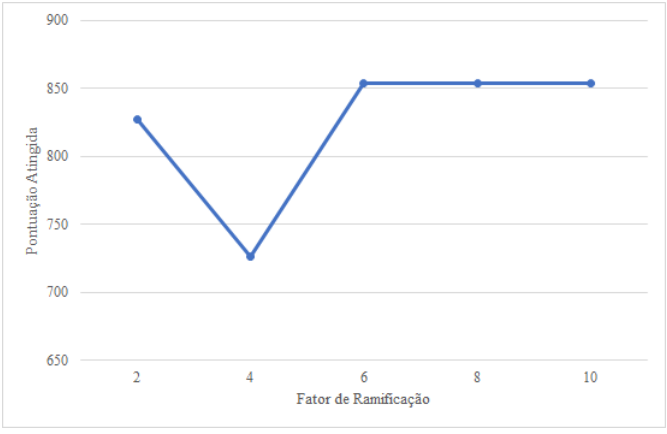

Figure 3: Pontuação Atingida em função do Fator de Ramificação. A procura usada foi $A*$ e a heurística foi a mista.

De acordo com a figura 3, concluímos que quando o fator de ramificação assume valores superiores a 5, não afeta a pontuação obtida.

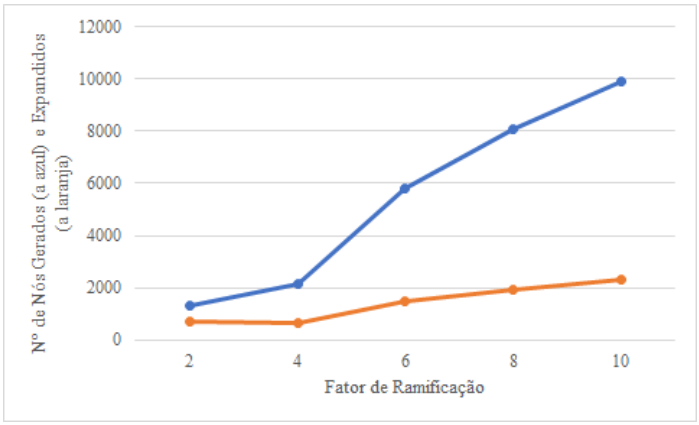

Figure 4: Nº de Nós Expandidos em função do Fator de Ramificação. A procura usada foi $A*$ e a heurística foi a mista.

Na figura 4, podemos observar um aumento do número de nós gerados/expandidos à medida que elevamos o fator de ramificação. Este resultado está de acordo com aquilo que esperávamos, visto que fatores de ramificação mais elevados vão permitir que, no momento de exploração da procura, mais nós sejam visitados.

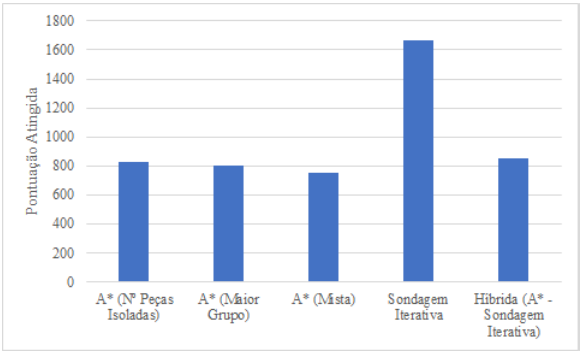

Figure 5: Pontuação Atingida por cada uma das estratégias de procura usadas. Para as procuras onde a aleatoriedade é um fator a considerar, calculámos a pontuação média entre 10 procuras realizadas.

A última figura demonstra o elevado desempenho da sondagem iterativa na procura por um estado com a maior pontuação possível. Esta obteve cerca do dobro da pontuação das restantes, sendo que estas revelam resultados muito semelhantes entre si.

8 Conclusão

Apesar de no ficheiro submetido termos considerado a estratégia híbrida como a melhor abordagem, devido ao erros de implementação mencionados, os resultados obtidos por esta foram amplamente superados pela a estratégia de *Sondagem Iterativa*.

9 Melhorias Futuras

Em relação à estratégia de procura híbrida $A*$ - *Sondagem Iterativa*, uma melhoria possível a implementar seria usar a estratégia $A*$ até a memória usada se aproximar do total de memória disponível. Só então se procederia à realização de sondagens iterativas que procurassem soluções nas sub-árvores abaixo, fazendo uso da baixa exigência de memória por parte da sondagem visto que esta apenas mantém em memória o caminho a ser considerado. Outra melhoria, complementar à anterior, seria considerar todos os nós da fronteira resultante da procura $A*$ em vez de considerar apenas o melhor. Assim, quando as sondagens varressem toda a sub-árvore abaixo do melhor nó

encontrado pela *A**, prosseguiria para a sub-árvore abaixo do segundo melhor nó e assim sucessivamente até esgotar o tempo disponível.

Seria interessante implementar um mecanismo de eliminação de linhas e colunas inválidas no tabuleiro visto que estas já não apresentam informação relevante para a execução da procura e podem, portanto, ser descartadas.

Seria também interessante implementar uma heurística alternativa que calcula a média dos tamanhos de todos os grupos de peças, sendo com isso possível calcular a pontuação "esperada" até esgotar as peças do tabuleiro. Sendo G o número de grupos no tabuleiro e $\overline{N}$ o número médio de peças por grupo, esta heurística consistiria então em:

$$\frac{1}{G * (\overline{N} - 2)^2}$$

Assim, no momento de exploração de estados, são beneficiados os tabuleiros que possuem, em média, grupo maiores.