Universidade do Porto

Faculdade de Engenharia

FEUP

Pesquisa aplicada à resolução do jogo Hopeless

Relatório Final

Inteligência Artificial

GRUPO A5_1:

- João Carlos Eusébio Almeida - up201306301
- João Gabriel Marques Costa - up201304197
- Nuno Martins Marques Pinto - up201307878

1. Objectivo

Este projeto, desenvolvido no âmbito da unidade curricular Inteligência Artificial, consiste na implementação do jogo solitário Hopeless, implementação essa feita em JavaScript e linguagens web.

O programa final deve, através de diversos algoritmos, apresentar um tabuleiro com diversas cores (dependendo da dificuldade) e a sua solução ótima, ou próximo dela. É também possível a comparação de diferentes algoritmos na determinação da solução de modo a tirar conclusões a nível de eficiência (qualidade/tempo).

Por fim, o objetivo deste projeto é a obtenção de soluções para diferentes configurações do jogo.

2. Especificação

2.1. Algoritmos

Algoritmo A*

É um algoritmo de busca (*path finding*) que toma um problema e retorna a sua solução, após considerar várias. É extremamente útil devido à sua precisão, flexibilidade e pode ser utilizado nos mais variados contextos. Pode também ser usado para encontrar o caminho mais curto e utilizar a heurística para se "auto-guiar".

Neste trabalho, a utilização deste algoritmo insere-se na otimização das jogadas de modo a obter o maior número de pontos. Dado que o algoritmo é minimizante e que se pretende maximizar os pontos, estes mesmo foram definidos como negativos na implementação. Desta forma, irá efetivamente funcionar como maximizante.

A fórmula para a estimativa do custo para o node final é:

$$f(n) = g(n) + h(n)$$

onde n é o último node no path, $g(n)$ é o custo desde o node inicial até n, e $h(n)$ é a heurística que estima o menor custo do caminho de n até ao node pretendido que, neste caso, é quando não existe mais nenhuma jogada possível.

Best-First Search

É um algoritmo de busca que explora o grafo ao expandir o node mais promissor de acordo com uma regra especificada. Também é associado a uma busca com uma heurística que tenta prever o caminho até ao fim do path, sendo que os que estão mais perto são expandidos primeiro.

O algoritmo A* acima referido é um exemplo de um algoritmo Best-First Search, que são muitas vezes usados para *path finding*.

Em comparação com o algoritmo A*, podemos concluir que o A* é mais ótimo. Apesar de nem sempre encontrar uma heurística que seja admissível, tem em conta a distância já percorrida (*G(n)*) assim como a distância a percorrer (*H(n)*) o que faz o algoritmo mais eficiente.

Por sua vez, o Best-First Search tem na sua fórmula, f(n) = h(n) , somente a estimativa do custo do *node* atual até ao final.

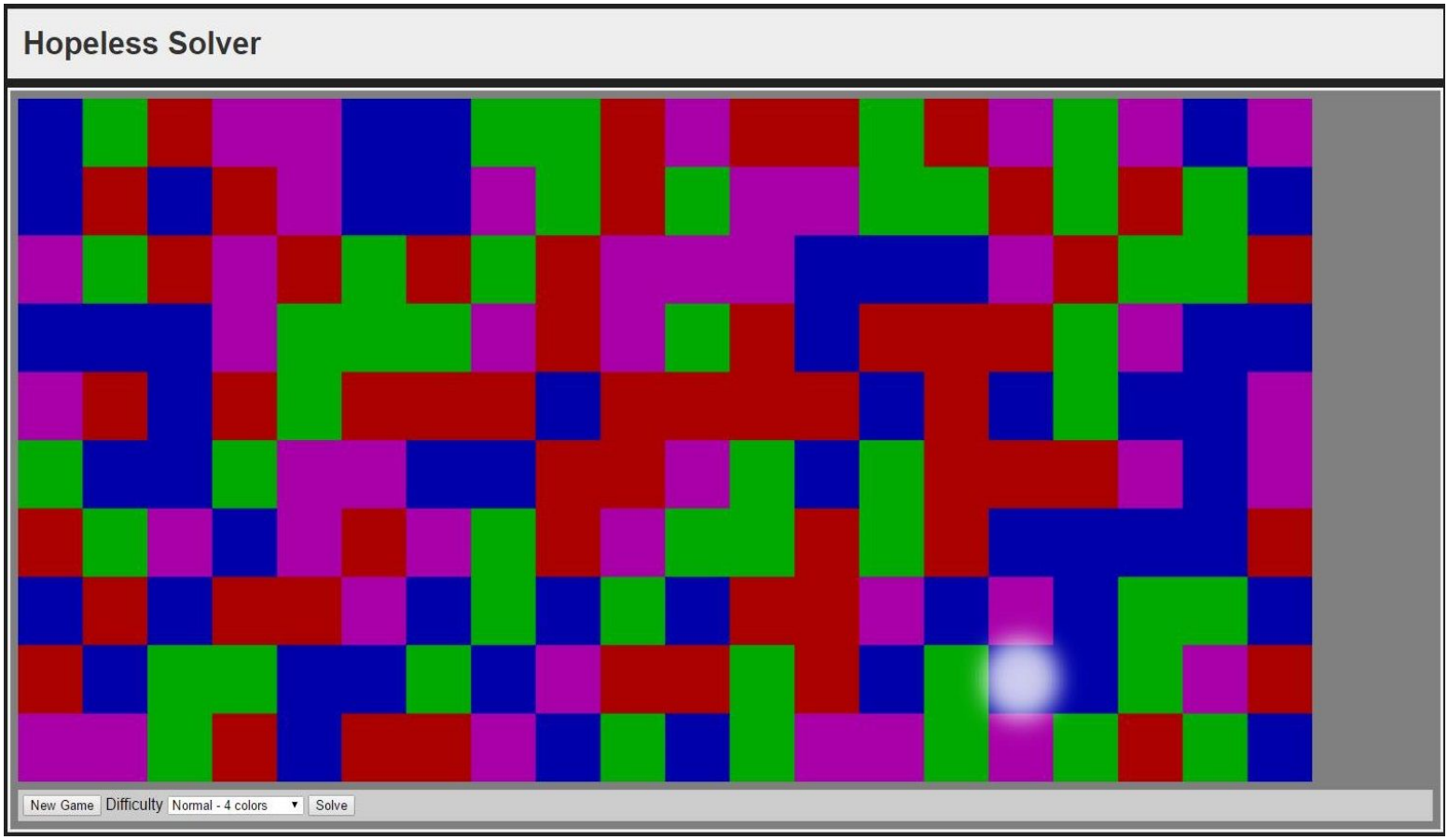
2.2. Heurísticas

A principal dificuldade foi encontrar heurísticas admissíveis para os algoritmos, ou seja que não os sobre-estime. Posto isto, a única heurística utilizada para encontrar a solução do tabuleiro é baseada no máximo de pontos possível a obter em cada jogada.

É importante dizer que são usados pontos negativos uma vez que o algoritmo tem como objectivo minimizar distâncias, neste caso a distância são os pontos possíveis obter.

3. Interface

Visto a aplicação ter sido desenvolvida fazendo uso de tecnologias web, para ser corrida é só necessário um *web browser* com *Javascript* ativado e abrir o ficheiro *"index.html"* nesse mesmo browser.



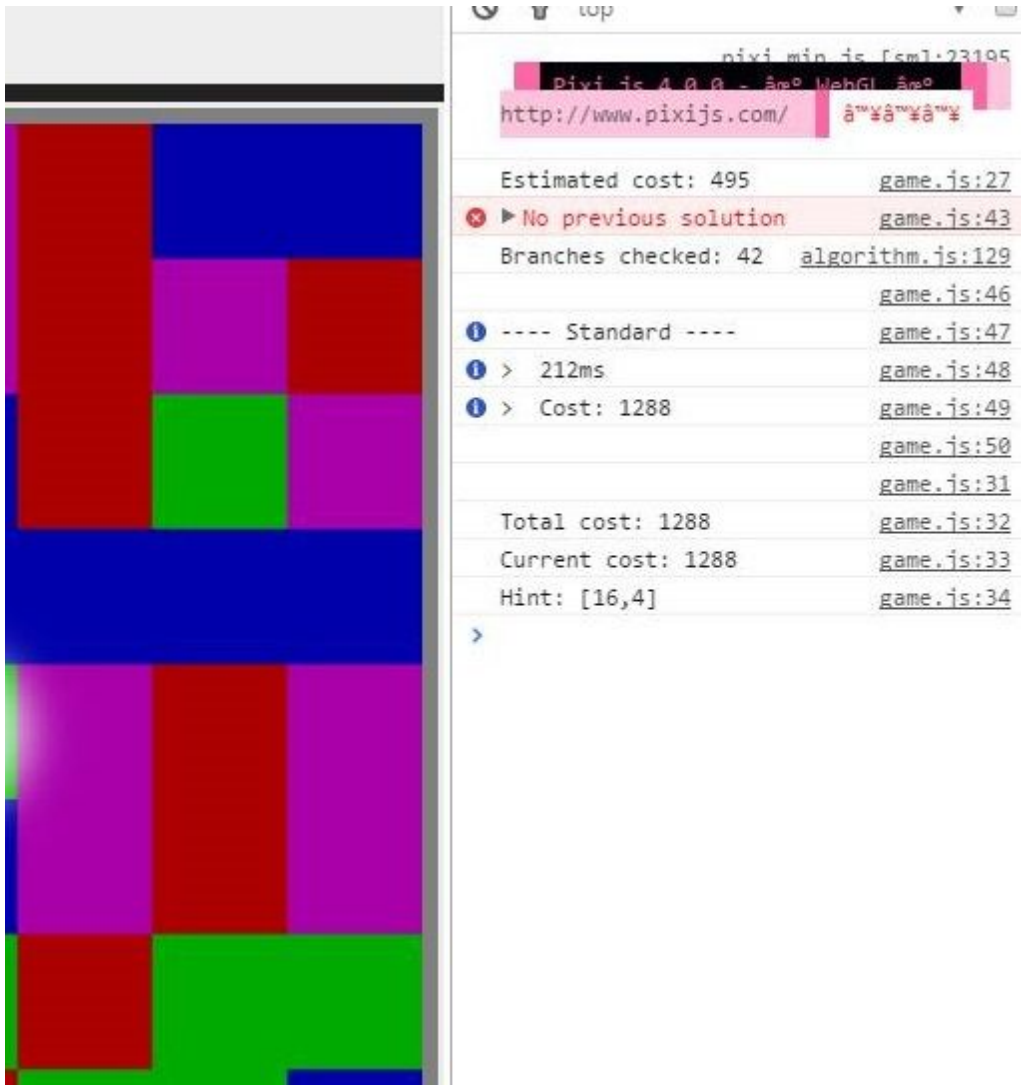
O grupo optou por uma interface simples de modo a facilitar o uso e para ser intuitivo. Como se pode ver pela a imagem, observa-se que o jogo tem um menu com três opções:

- *New Game:* Este botão tem como funcionalidade recomeçar o jogo, ou seja criar um novo tabuleiro
- *Difficulty:* Este botão permite ao utilizador escolher a dificuldade do nível. A dificuldade varia com o número de cores, neste jogo o nível mais fácil tem 1 cor e o mais díficil 6.

- *Solve:* Este botão tem como função resolver e encontrar a solução para o tabuleiro apresentado.

Por último, aquele *highlight* no tabuleiro tem como função representar as peças que devem ser clicadas com base no algoritmo utilizado.

4. Resultados



Como podemos observar pela imagem, o algoritmo A* é bastante eficiente pois encontra uma solução ótima para um tabuleiro (o mesmo que em 3.) relativamente complexo de forma quase instantânea.

5. Conclusões

Após a realização deste projeto, e tendo em conta o seu propósito sem dúvida que este contribuiu para aumentar o conhecimento de todos os elementos do grupo: houve uma clara aquisição de novas ideias e de novos conceitos, também se evidenciou uma compreensão notável sobre a necessidade indispensável de estudar cuidadosamente os algoritmos existentes de modo a implementá-los de forma correta para que a aplicação seja rápida e eficaz.

Contudo só o algoritmo A* é que está completamente funcional, uma vez que houver diversos problemas na implementação do BFS (sendo uma delas encontrar uma heurística que permita a resolução do problema em tempo real).

Por fim, o grupo está satisfeito com o trabalho realizado e percebeu a utilidade dos algoritmos assim como o seu potencial uso em diversas áreas.

6. Recursos

6.1. Bibliografia

- https://web.fe.up.pt/~eol/IA/1516/TRABALHOS/pesqsol_hopeless.html
- http://greenfelt.net/hopeless
- https://web.fe.up.pt/~eol/IA/1516/ia_.html
- http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html
- Heuristics: Intelligent Search Strategies for Computer Problem Solving. Addison-Wesley, 1984.

6.2. Software

- Linux
- Emacs

6.3. Percentagem de cada elemento do grupo

- João Almeida: 32,5%
- João Costa: 35%
- Nuno Pinto: 32,5%