

Universidade Federal de São Carlos
Departamento de Computação
Inteligência Artificial

Trabalho 1

Prof. Lúcia Machado Rino
Turma A

Cristiano de Oliveira Faustino
Matheus Naoto Shimura Takata
Pedro Henrique Babolim Zago

São Carlos – SP
2013

Indicação do problema eleito

Neste trabalho, abordou-se o Sliding-Tile Puzzle. Nele, o jogador dispõe de três fichas brancas e três pretas, em um espaço com sete posições cujo objetivo é deixar todas as fichas brancas à esquerda e as pretas à direita. É possível deslizar uma ficha para um espaço vazio adjacente ou pular sobre uma ou duas fichas para uma posição vazia.

Definição de $f1(n)$ e $f2(n)$

As funções heurísticas utilizadas foram:

$f1(n)$: Considera-se, para o cálculo de $h1(n)$ a subtração das fichas pretas das brancas situadas nas quatro primeiras posições do tabuleiro, podendo assumir valores negativos se a quantidade de fichas pretas for maior. A esse resultado, é somado $g1(n)$, que considera o peso de cada jogada da raiz até o nó analisado. Esse peso tem valor igual a 1 para arrastar e pular uma peça e igual a 2 se pular duas peças.

$f2(n)$: Utiliza-se metade da soma do número de peças brancas à direita de cada peça preta w_i . Dividindo o valor da soma pela metade garante que o resultado não seja superestimado, uma vez que uma peça branca pode pular duas peças pretas com um único movimento.

O cálculo de $h2(n)$ é dado por:

$$h(n) = \frac{1}{2} \sum_i w_i$$

Ao valor de $h2(n)$, soma-se $g2(n)$, que considera o peso de cada jogada da raiz até o nó analisado, assim como $g1(n)$.

Métodos de Busca e Solução

O método de busca utilizado foi baseado no *best-first*. Dado o estado atual do jogo, verifica-se a possibilidade de jogada de cada ficha. Feito isso, verifica-se se esta nova jogada possível já foi imaginada pelo robô, procurando-a nas listas OPEN e CLOSED. Caso não esteja em nenhuma das listas, a jogada será inserida na lista OPEN, como possibilidade de solução. Caso esteja na lista CLOSED, é preciso comparar se o custo desta jogada agora é inferior ao custo do nó que está em CLOSED. Caso seja, então este nó é removido da lista CLOSED e a nova jogada é inserida na lista OPEN. Se a jogada já estiver na lista OPEN, nada acontece.

Ambiente e Instruções de Uso

O programa foi desenvolvido na IDE Visual Studio 2012 profissional na linguagem C#, utilizando o Monogame, uma implementação *open-source* do *framework* XNA da Microsoft para desenvolvimento de jogos que tem como objetivo permitir que os desenvolvedores deste *framework* possam produzir jogos para as diversas plataformas existentes hoje em dia, como Linux, Sony PlayStation 3, OUYA com o mesmo código utilizado para desenvolver os jogos para as plataformas da Microsoft.

Para executar o programa, será necessário instalar o Visual Studio 2012 em uma plataforma Windows (como o programa foi desenvolvido no Windows 7, deve-se dar preferência a esta plataforma). Além disso, é necessário instalar o Monogame, cujo instalador já está inserido no arquivo .rar. Instalado o *framework*, é necessário executar o arquivo oalinst.exe que se encontra no diretório “C:\Program Files (x86)\MonoGame\v3.0”. É importante observar que o diretório C: e Program Files (x86) são diretórios que variam de acordo com a partição e a versão do Windows instalado.

Feito isso, abra execute o arquivo GameName1.sln localizado dentro da pasta GameName1. Com o ambiente de desenvolvimento aberto, basta apertar F5 para executar o jogo. Durante a execução, é possível acompanhar cada passo que o robô executou até chegar no resultado apertando a tecla Espaço. Para fechar o programa, basta apertar a tecla de Escape(Esc). No código enviado, o programa executará a heurística do primeiro robô. Para acompanhar a heurística do segundo robô, é necessário fazer algumas mudanças no código, mais especificamente no arquivo LogicaIA.cs, sendo elas:

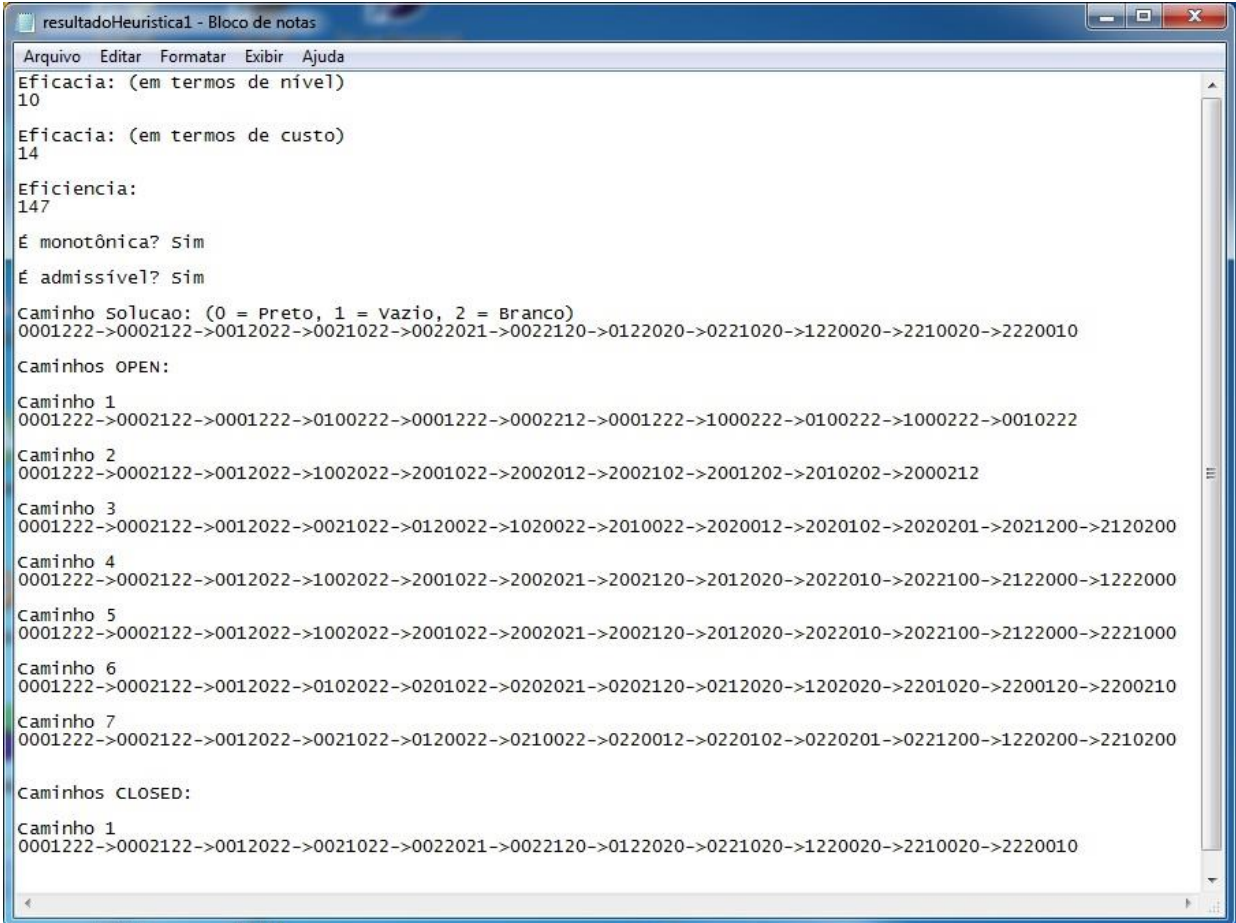
- Na linha 30, trocar “calculaHeuristica1” por “calculaHeuristica2”;
- Na linha 177, trocar “calculaHeuristica1” por “calculaHeuristica2”;
- Na linha 383, trocar “calculaHeuristica1” por “calculaHeuristica2”;

Para gerar os resultados no arquivo correto é necessário trocar, na linha 244, StreamWriter("resultadoHeuristica1.txt") por StreamWriter("resultadoHeuristica2.txt").

Resultados

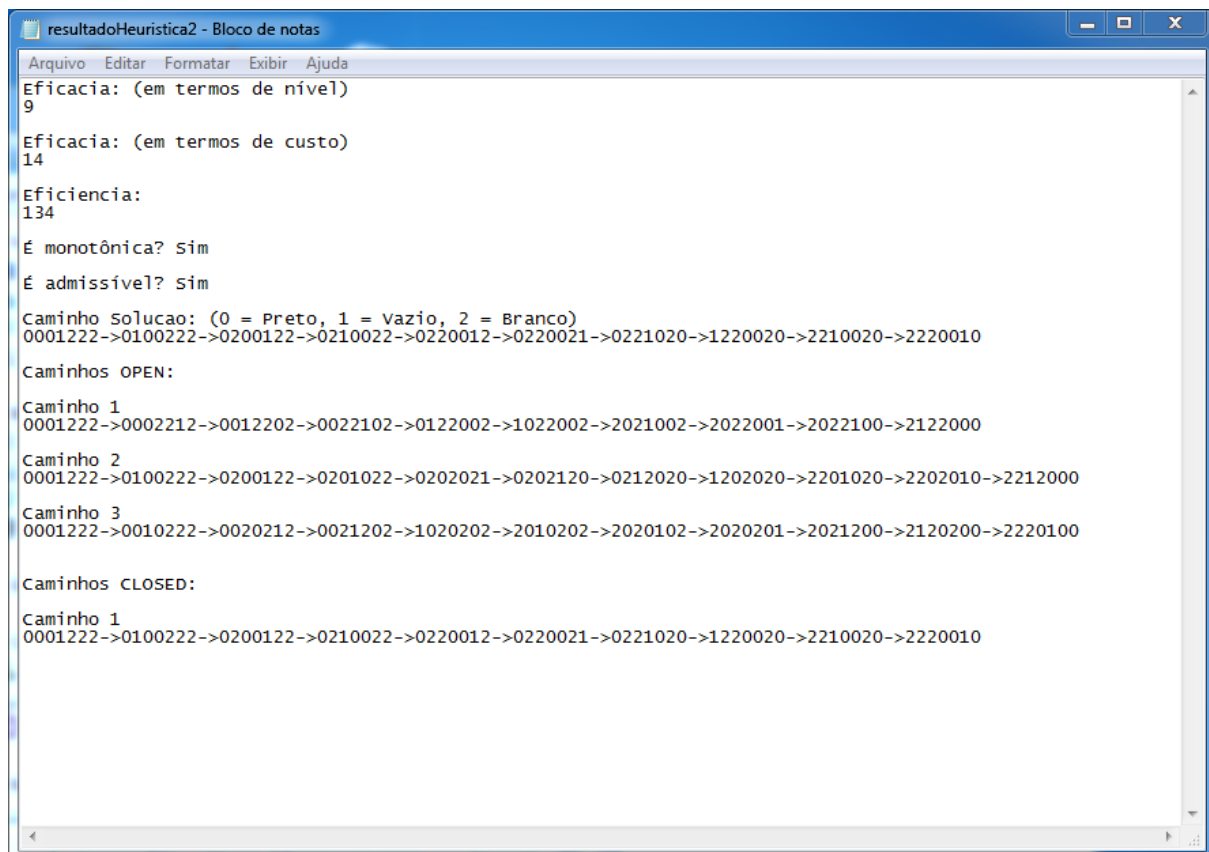
Os resultados gerados pelos robôs foram escritos em arquivo, que acompanha o pacote de execução do programa. Nas figuras, são exibidos os *dumps* de tela com o conteúdo dos arquivos correspondentes a cada heurística.

Resultado – Robô 1:



```
resultadoHeuristica1 - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda
Eficacia: (em termos de nível)
10
Eficacia: (em termos de custo)
14
Eficiencia:
147
É monotônica? Sim
É admissível? Sim
Caminho Solucao: (0 = Preto, 1 = Vazio, 2 = Branco)
0001222->0002122->0012022->0021022->0022021->0022120->0122020->0221020->1220020->2210020->2220010
Caminhos OPEN:
Caminho 1
0001222->0002122->0001222->0100222->0001222->0002212->0001222->1000222->0100222->1000222->0010222
Caminho 2
0001222->0002122->0012022->1002022->2001022->2002012->2002102->2001202->2010202->2000212
Caminho 3
0001222->0002122->0012022->0021022->0120022->1020022->2010022->2020012->2020102->2020201->2021200->2120200
Caminho 4
0001222->0002122->0012022->1002022->2001022->2002021->2002120->2012020->2022010->2022100->2122000->1222000
Caminho 5
0001222->0002122->0012022->1002022->2001022->2002021->2002120->2012020->2022010->2022100->2122000->2221000
Caminho 6
0001222->0002122->0012022->0102022->0201022->0202021->0202120->0212020->1202020->2201020->2200120->2200210
Caminho 7
0001222->0002122->0012022->0021022->0120022->0210022->0220012->0220102->0220201->0221200->1220200->2210200
Caminhos CLOSED:
Caminho 1
0001222->0002122->0012022->0021022->0022021->0022120->0122020->0221020->1220020->2210020->2220010
```

Resultado – Robô 2:



```
resultadoHeuristica2 - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda
Eficacia: (em termos de nível)
9
Eficacia: (em termos de custo)
14
Eficiencia:
134
É monotônica? Sim
É admissível? Sim
Caminho Solucao: (0 = Preto, 1 = Vazio, 2 = Branco)
0001222->0100222->0200122->0210022->0220012->0220021->0221020->1220020->2210020->2220010
Caminhos OPEN:
Caminho 1
0001222->0002212->0012202->0022102->0122002->1022002->2021002->2022001->2022100->2122000
Caminho 2
0001222->0100222->0200122->0201022->0202021->0202120->0212020->1202020->2201020->2202010->2212000
Caminho 3
0001222->0010222->0020212->0021202->1020202->2010202->2020102->2020201->2021200->2120200->2220100
Caminhos CLOSED:
Caminho 1
0001222->0100222->0200122->0210022->0220012->0220021->0221020->1220020->2210020->2220010
```

Como se pode verificar nos resultados, o único caminho que leva o nó raiz até a folha é o caminho-solução. Isto acontece pois não há, pelo menos nos primeiros níveis, um nó folha que venha a ser desconsiderado, ou seja, os nós folhas que foram fechados anteriormente sempre acabam voltando à lista OPEN, até que as folhas se tornem nós-objetivos.

Comparação entre os robôs

Analisando as propriedades dos robôs, constatou-se que ambos são monotônicos, o que implica que os mesmos são admissíveis. Do ponto de vista do peso das jogadas, ambos possuem o mesmo custo. Por outro lado, o caminho de solução encontrado pelo robô 2 é mais curto, o que caracteriza um custo menor em relação ao tamanho do caminho. Sendo assim, considera-se que o robô 2 é o mais informado.

O modelo de raciocínio do primeiro robô analisa as quatro primeiras posições e compara o número de peças brancas em relação às pretas nesse espaço. É deduzido que, quanto maior o número de peças brancas, mais promissor é o caminho.

Já o segundo robô compara a quantidade de peças brancas à direita das pretas no tabuleiro. Quanto menor for esse valor, mais próximo da configuração desejada o nó está e, portanto, é considerado o mais promissor.

De uma forma geral, os robôs apresentam níveis próximos de desempenho. Entretanto, em comparação a um raciocínio natural, nota-se que o primeiro robô possui limitações mais aparentes por restringir suas comparações às quatro primeiras posições do tabuleiro e dar ênfase à movimentação de peças também nessa região. O segundo robô se mostra mais próximo do raciocínio natural, uma vez que sua heurística abrange todo o tabuleiro e, no caso de um jogador real, seu objetivo também seria buscar por peças brancas à direita das pretas e trocá-las de posição.