Universidade de São Paulo

Escola de Artes, Ciências e Humanidades

ACH2016 – Inteligência Artificial

RELATÓRIO
ANÁLISE DE ALGORITMOS DE BUSCA

Daniel Augusto de Melo Moreira – 8122477

Felipe Brigatto – 7972602

Igor Oliveira Borges – 8122442

Marcelo Werneck Gaioso – 8061963

01 de outubro de 2014

Introdução

Esse exercício programa consiste na implementação de 6 métodos de busca para resolver o problema conhecido como *travessia perigosa* que visa reduzir o tempo de travessia.

Buscas:

- Profundidade
- Largura
- Custo Uniforme
- Profundidade Interativa
- Guloso
- A*

É um quebra-cabeça lógico, no qual há pessoas do lado oeste da ponte, que anseiam chegar ao lado leste, contudo só podem atravessar duas de cada vez, cada uma possui um tempo de travessia que é dominado pelo parceiro de mais lento e é necessário retornar para trazer um artefato que é crucial para atravessar a ponte, comumente relatado como lanterna ou tocha.

A árvore completa é muito grande do problema, a depender da quantidade de pessoas modeladas, e por isto uma estratégia é ir permutando os caminhos válidos e gerando a árvore conforme explora o problema.

A seguir descrevemos formalmente o problema e três heurísticas que utilizamos para os algoritmos de busca informada, e então mostramos os resultados obtido por cada algoritmo em três casos testes propostos e os comparamos.

Descrição do Problema

Estados: A quantidade de estados do problema é dada por: $2^{n+1} - 2$.

Prova: Suponha que os estados sejam definidos apenas pelas pessoas a oeste e leste da ponte (desconsiderando a localização da lanterna), dessa forma os estados podem ser representados pela combinação do conjunto inicial de pessoas posicionadas em um lado fixado e posicionando as demais no lado oposto.

Ex.: Considerando a Instancia 1 do problema e fixando o lado a oeste, então um estado intermediário é dado por uma das possíveis combinações de 2 pessoas retiradas do conjunto inicial e posicionando-as a oeste, e as demais a leste.

Assim, podemos calcular a quantidade de estados da seguinte maneira:

$$\sum_{i=0}^{n} C_{n,i} = 2^n$$

A igualdade do lado direito é obtida, pois a somatória representa a soma de uma linha do Triangulo de Pascal.

Agora, consideremos a representação total dos estados dado pelo problema, ou seja, adicionamos a localização da lanterna. Dessa forma, dado um estado da primeira suposição geraremos dois novos estados, um com a lanterna a oeste e outro a leste. Porém, pela descrição do problema podemos inferir que a lanterna não pode estar em um lado da ponte que não há pessoas, portanto devemos remover os estados inicial e final.

Concluindo, chegamos ao cálculo final da quantidade de estados do seguinte modo:

$$2^n \times 2 - 2 = 2^{n+1} - 2$$

Obs.: Para algumas instancias do problema a quantidade de estados percorridos pelos algoritmos de busca sempre será menor que o total devido a possibilidade de alguns estados apenas serem alcançados a partir do estado final.

Definição

Estado inicial: Possui todas as pessoas do conjunto inicial e a lanterna posicionada no lado oeste da ponte e o conjunto vazio no lado leste.

Ações: Transportar i-ésima ou i-ésima e j-ésima, onde $i \neq j$, pessoa do conjunto de pessoas que possui a lanterna para o lado oposto da ponte.

Modelo de transição: As ações colocam a lanterna e as pessoas selecionadas no lado oposto.

Teste de objetivo: Verifica se não há pessoas no lado oeste (utilizada na implementação) ou se todas as pessoas estão do lado leste.

Custo do caminho: Cada transferência de pessoas possui o custo da pessoa que leva mais tempo para atravessar a ponte denotado por: $\max(custo(i), custo(j))$, assim o custo do caminho é definido pela somatória de cada transferência realizada.

Heurísticas

Demonstraremos a seguir que as heurísticas escolhidas para a resolução do problema e utilizadas pelos algoritmos guloso e A* são admissíveis, ou seja, $h(n) \leq h^*(n)$, para isso primeiro definimos as funções matemática que cada heurística representa.

A demonstração é feita na ordem invertida da definição das heurísticas, pois aumentamos a complexidade linearmente e portanto, é mais fácil demonstrar que uma é admissível em relação as anteriores do que compara-las ao custo original, o que nos obrigaria a repetir coisas já ditas.

Definição: O conjunto em ordem decrescente de tempo de travessia das pessoas a oeste da ponte é representado por "O", e o i-ésimo elemento do conjunto é representado por "O_i".

As heurísticas utilizadas são:

Heurística 1: A função de avaliação aplicada a um determinado estado é:

$h(n) = \min(O)$. A ideia da função é garantir o custo mínimo possível, porém é uma heurística que estima um valor muito longe do real.

Heurística 2: A função de avaliação aplicada a um determinado estado é:

$h(n) = \max(O)$. Essa função é derivada de um afrouxamento da regra de travessia, permitindo que todas as pessoas do lado oeste atravessem a ponte ao mesmo tempo com apenas uma única lanterna. Dessa forma, o custo é calculado pelo tempo de travessia do membro que possui o maior tempo de travessia do grupo.

Heurística 3: A função de avaliação aplicada a um determinado estado é:

$h(n) = \sum_{i=0}^{(n/2)-1} \max(O_{2i} + O_{2i+1}) + O_{n-1} \times (n\%2)$. Essa função é derivada de um afrouxamento da regra de travessia, permitindo lanternas suficientes para que vários grupos de duas pessoas, e eventualmente uma pessoa, atravessem ao mesmo tempo. Dessa forma, o custo é calculado pela somatória do tempo de travessia de cada grupo.

Ideia da prova: O custo real do caminho leva em consideração a necessidade de algumas pessoas terem de retornar para o lado oeste e portanto mesmo que o custo da volta fosse nulo ainda levaríamos mais tempo para concluir a travessia pois reduziríamos o conjunto de pessoas do lado oeste em um, quando a volta fosse necessária. Nesse caso teríamos que a solução ótima transportaria as pessoas de maior peso juntas (heurística 3), pois dessa forma amortizaria o peso da segunda mais pesada, e utilizaria os grupos de menor peso para transportarem a lanterna nas voltas.

Portanto, a solução ótima será composta pela heurística 3 adicionando os possíveis custos de transportar pessoas de volta a oeste. Então, a heurística 3 deve ser no máximo igual a solução ótima e consequentemente a heurística 3 é admissível.

As heurísticas 2 e 1 são obviamente menores ou iguais a heurística 3 e portanto são também admissíveis.

Concluindo, $h_1(n) \leq h_2(n) \leq h_3(n) \leq h^*(n)$ então todas as heurísticas são admissíveis.

RESULTADOS

Busca Largura			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
17	5	4	26
17	5	4	26
17	5	4	26
17	5	4	26
17	5	4	26
Tempo médio (ms):		4	

Busca A* (x.custo+f.h1(x) <= y.custo+f.h1(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
17	5	5	19
17	5	4	19
17	5	5	19
17	5	4	19
17	5	4	19
Tempo médio (ms):		4,4	

Busca Custo Uniforme			
Custo Total	Profundidade	Tempo (ms)*	Estados
17	5	4	26
17	5	4	26
17	5	5	26
17	5	5	26
17	5	6	26
Tempo médio (ms):		4,8	

Busca A* (x.custo+f.h2(x) < y.custo+f.h2(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
17	5	5	19
17	5	4	19
17	5	4	19
17	5	5	19
17	5	4	19
Tempo médio (ms):		4,4	

Busca A* (x.custo+f.h3(x) < y.custo+f.h3(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
17	5	4	19
17	5	5	19
17	5	6	19
17	5	6	19
17	5	6	19
Tempo médio (ms):		5,4	

Imagem 1. Tabelas de resultados de execução da instância 1. Os algoritmos Profundidade, Profundidade Iterativa e Guloso (três heurísticas) não encontraram solução para o máximo de minutos estipulado.

Busca Largura			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
104	17	164	2036
104	17	163	2036
104	17	163	2036
104	17	164	2036
104	17	159	2036
Tempo médio (ms):		162,6	

Busca A* (x.custo+f.h1(x) <= y.custo+f.h1(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
104	17	161	1987
104	17	166	1987
104	17	180	1987
104	17	196	1987
104	17	203	1987
Tempo médio (ms):		181,2	

Busca Custo Uniforme			
Custo Total	Profundidade	Tempo (ms)*	Estados
104	17	156	2034
104	17	162	2034
104	17	164	2034
104	17	163	2034
104	17	157	2034
Tempo médio (ms):		160,4	

Busca A* (x.custo+f.h2(x) < y.custo+f.h2(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
104	17	178	2036
104	17	168	2036
104	17	165	2036
104	17	177	2036
104	17	154	2036
Tempo médio (ms):		168,4	

Busca A* (x.custo+f.h3(x) < y.custo+f.h3(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
104	17	148	2036
104	17	147	2036
104	17	148	2036
104	17	159	2036
104	17	149	2036
Tempo médio (ms):		150,2	

Imagem 2. Tabelas de resultados de execução da instância 2**.** Os algoritmos Profundidade, Profundidade Iterativa e Guloso (três heurísticas) não encontraram solução para o máximo de minutos estipulado.

Busca Largura			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
182	27	11507	49136
182	27	11442	49136
182	27	11248	49136
182	27	11158	49136
182	27	10753	49136
Tempo médio (ms):		11221,6	

Busca A* (x.custo+f.h1(x) <= y.custo+f.h1(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
182	27	10459	48960
182	27	10444	48960
182	27	11945	48960
182	27	10759	48960
182	27	10375	48960
Tempo médio (ms):		10796,4	

Busca Custo Uniforme			
Custo Total	Profundidade	Tempo (ms)*	Estados
182	27	10903	49108
182	27	10676	49108
182	27	10695	49108
182	27	10658	49108
182	27	10933	49108
Tempo médio (ms):		10773	

Busca A* (x.custo+f.h2(x) < y.custo+f.h2(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
182	27	11732	48960
182	27	11008	48960
182	27	11298	48960
182	27	10324	48960
182	27	10919	48960
Tempo médio (ms):		11056,2	

Busca A* (x.custo+f.h3(x) < y.custo+f.h3(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
182	27	10502	48960
182	27	10764	48960
182	27	11151	48960
182	27	10722	48960
182	27	10420	48960
Tempo médio (ms):		10711,8	

Imagem 3. Tabelas de resultados de execução da instância 3. Os algoritmos Profundidade, Profundidade Iterativa e Guloso (três heurísticas) não encontraram solução para o máximo de minutos estipulado.

EXPERIMENTO ADICIONAL

Com finalidade de testar os algoritmos de Profundidade, Profundidade Interativa e as três heurísticas também no Guloso. Relaxamos o valor Max de aceitabilidade de uma solução recebida como parâmetro a fim de testar as buscas que não encontram solução ótima.

Max=24

Busca Profundidade			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
19	5	0	7
19	5	1	7
19	5	1	7
19	5	1	7
19	5	1	7
Tempo médio (ms)		0,8	

Busca Profundidade Iterativa			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
19	5	6	7
19	5	5	7
19	5	4	7
19	5	6	7
19	5	7	7
Tempo médio (ms)		5,6	

Busca Gulosa (f.h1(x) <= f.h1(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
20	5	6	18
20	5	6	18
20	5	5	18
20	5	6	18
20	5	6	18
Tempo médio (ms)		5,8	

Busca Gulosa (f.h2(x) < f.h2(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
24	5	4	6
24	5	4	6
24	5	5	6
24	5	3	6
24	5	4	6
Tempo médio (ms)		4	

Busca Gulosa (f.h3(x) < f.h3(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
20	5	4	8
20	5	4	8
20	5	3	8
20	5	3	8
20	5	3	8
Tempo médio (ms)		3,4	

Imagem 4. Algoritmos da Instância 1 que não encontraram soluções. O tempo máximo foi alterado para efeito de comparação entre todos os algoritmos.

Max=205

Busca Profundidade			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
134	17	3	19
134	17	2	19
134	17	3	19
134	17	2	19
134	17	2	19
Tempo médio (ms)		2,4	

Busca Gulosa (f.h1(x) <= f.h1(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
204	19	233	475
204	19	323	475
204	19	239	475
204	19	262	475
204	19	299	475
Tempo médio (ms)		271,2	

Busca Profundidade Iterativa			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
134	17	893	19
134	17	839	19
134	17	829	19
134	17	860	19
134	17	862	19
Tempo médio (ms)		856,6	

Busca Gulosa (f.h3(x) < f.h3(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
136	17	55	59
136	17	54	59
136	17	54	59
136	17	58	59
136	17	57	59
Tempo médio (ms)		55,6	

Busca Gulosa (f.h2(x) < f.h2(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
202	17	30	18
202	17	31	18
202	17	35	18
202	17	31	18
202	17	31	18
Tempo médio (ms)		31,6	

Imagem 5. Algoritmos da Instância 2 que não encontraram soluções. O tempo máximo foi alterado para efeito de comparação entre todos os algoritmos.

Max=469

Busca Profundidade			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
277	27	5	29
277	27	6	29
277	27	3	29
277	27	4	29
277	27	5	29
Tempo médio (ms)		4,6	

Busca Gulosa (f.h1(x) <= f.h1(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
318	27	6455	31740
318	27	6511	31740
318	27	6500	31740
318	27	6563	31740
318	27	6541	31740
Tempo médio (ms)		6514	

Busca Profundidade Iterativa			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
277	27	9570	29
277	27	9601	29
277	27	9522	29
277	27	9565	29
277	27	9498	29
Tempo médio (ms)		9551,2	

Busca Gulosa (f.h2(x) < f.h2(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
466	27	278	128
466	27	284	128
466	27	297	128
466	27	251	128
466	27	262	128
Tempo médio (ms)		274,4	

Busca Gulosa (f.h3(x) < f.h3(y))			
Custo Total	Profundidade	Tempo (ms)*	Estados Visitados
298	27	356	233
298	27	364	233
298	27	340	233
298	27	344	233
298	27	344	233
Tempo médio (ms)		349,6	

Imagem 6. Algoritmos da Instância 3 que não encontraram soluções. O tempo máximo foi alterado para efeito de comparação entre todos os algoritmos.

GRÁFICOS

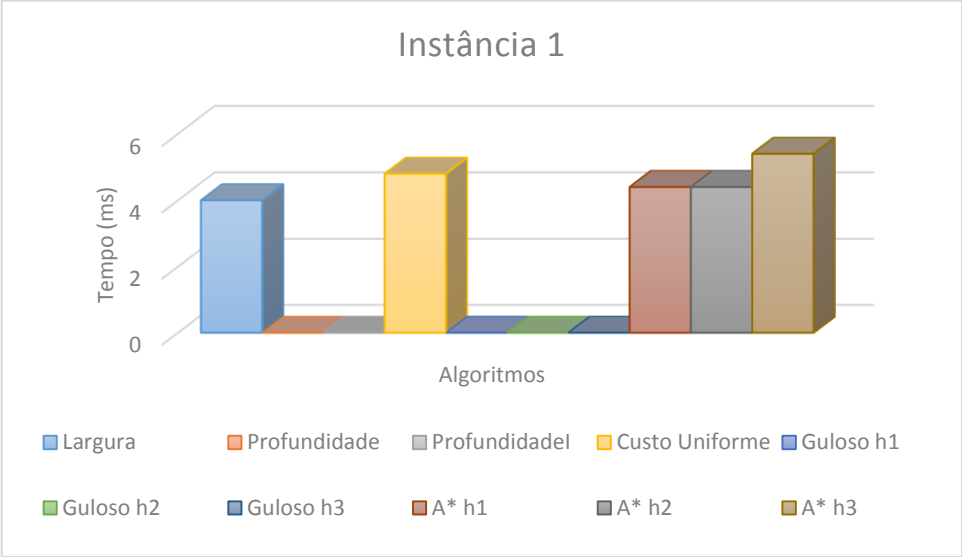


Gráfico 1. Análise da instância 1 por algoritmo.

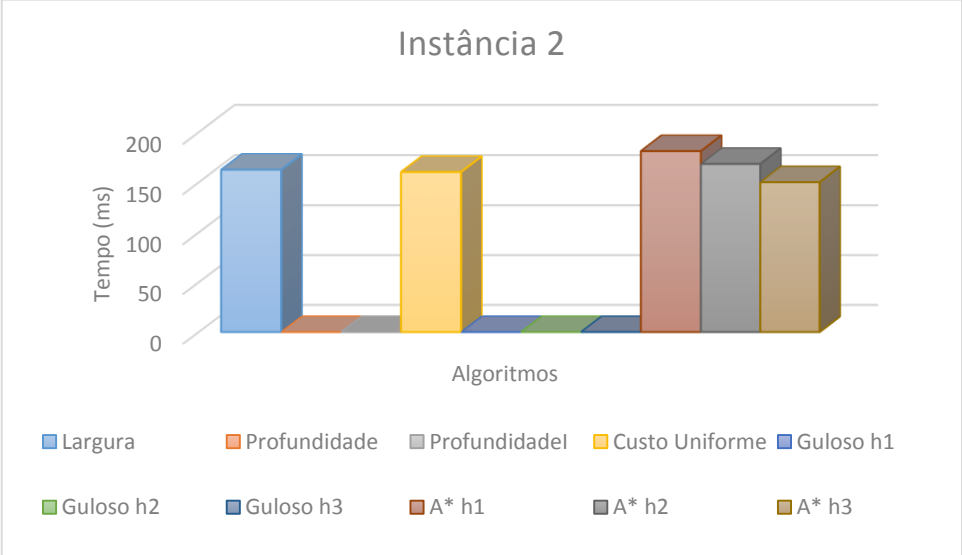


Gráfico 2. Análise da instância 2 por algoritmo.

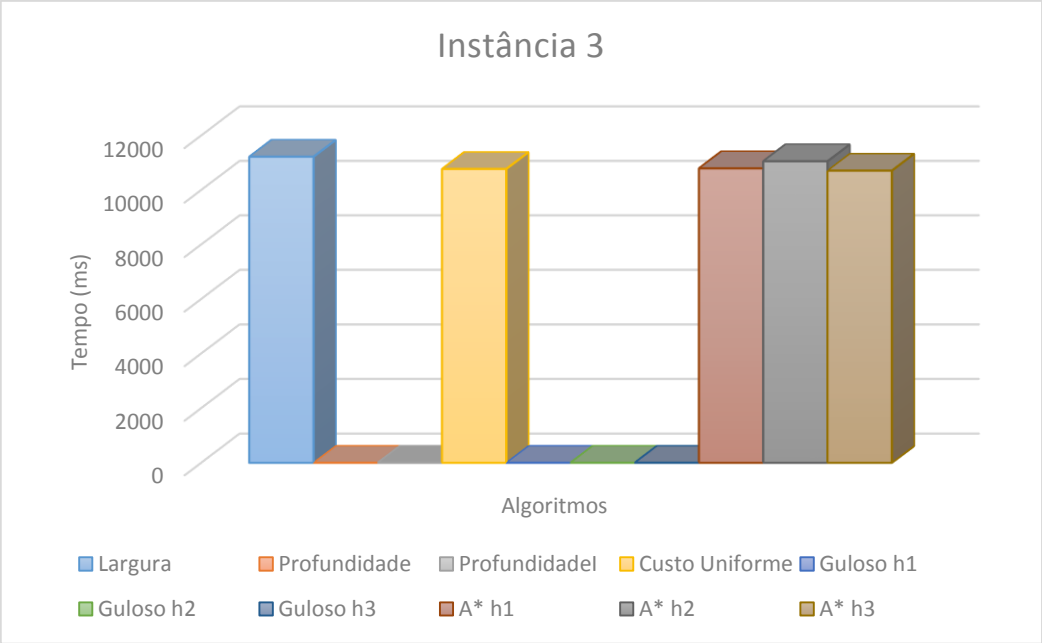


Gráfico 3. Análise da instância 3 por algoritmo.

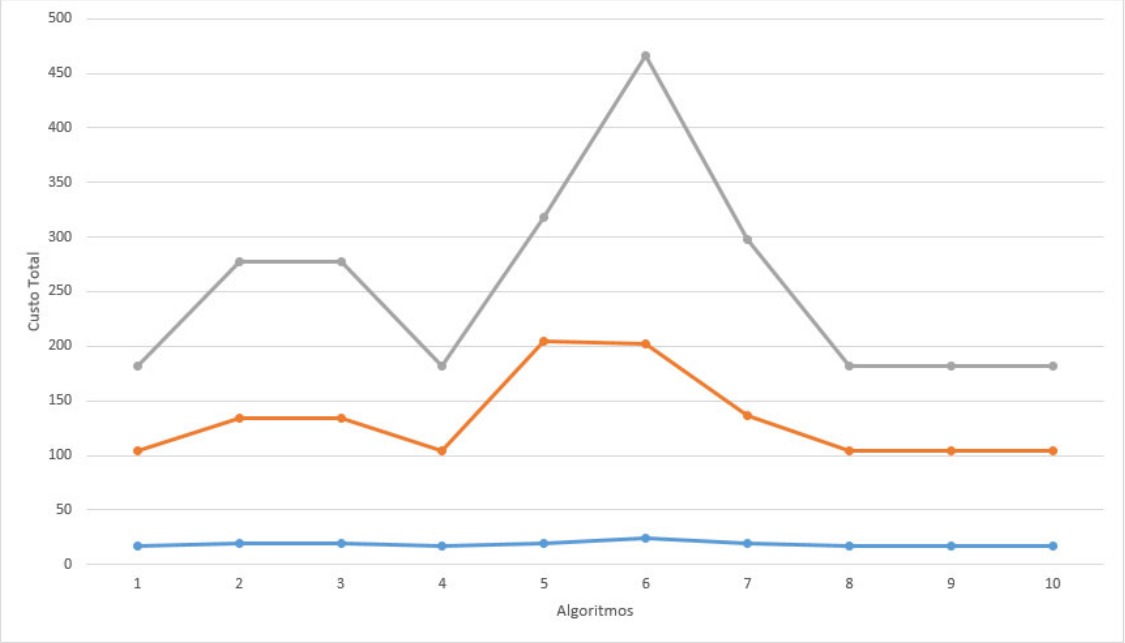
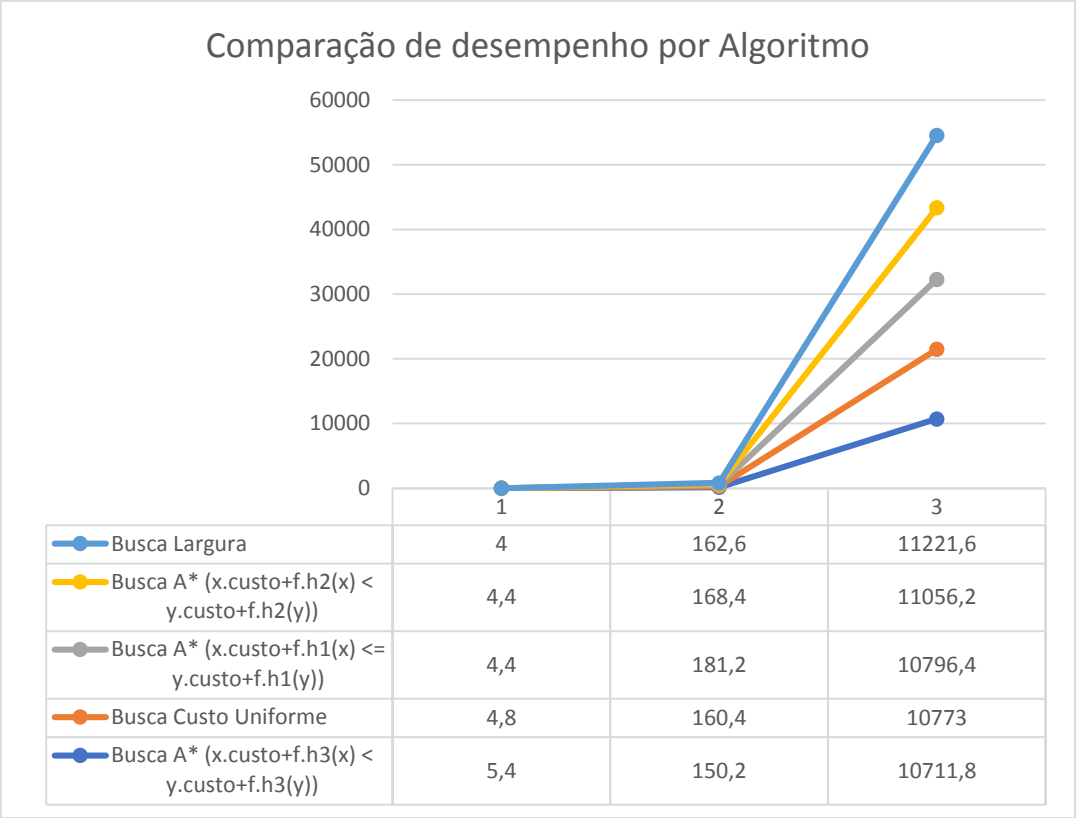


Gráfico 4. Análise do custo total por instância, relaxado o critério de aceitação da solução.



	1	2	3
Busca Largura	4	162,6	11221,6
Busca A* (x.custo+f.h2(x) < y.custo+f.h2(y))	4,4	168,4	11056,2
Busca A* (x.custo+f.h1(x) <= y.custo+f.h1(y))	4,4	181,2	10796,4
Busca Custo Uniforme	4,8	160,4	10773
Busca A* (x.custo+f.h3(x) < y.custo+f.h3(y))	5,4	150,2	10711,8

Gráfico 5. Análise dos custos computacionais nos três testes nos principais algoritmos que resolveram ambos.

CONSIDERAÇÕES FINAIS

Os algoritmos de busca cega Profundidade e Profundidade Interativa quase sempre não retornam a solução ótima, bem como na informada o algoritmo guloso também não o alcança, e mesmo nos casos testes aplicados não garantiram tal resultado conforme já esperado. Realizamos testes para estes com critério Max ampliado apenas para mostrar a satisfatibilidade dos algoritmos apresentados, uma vez que eles não encontram o ótimo, apresentamos o custo deste estado parcial que pode chegar a dobrar o obtido pelo melhor caminho, a fim de mensurar o quão distante estão da solução otimal.

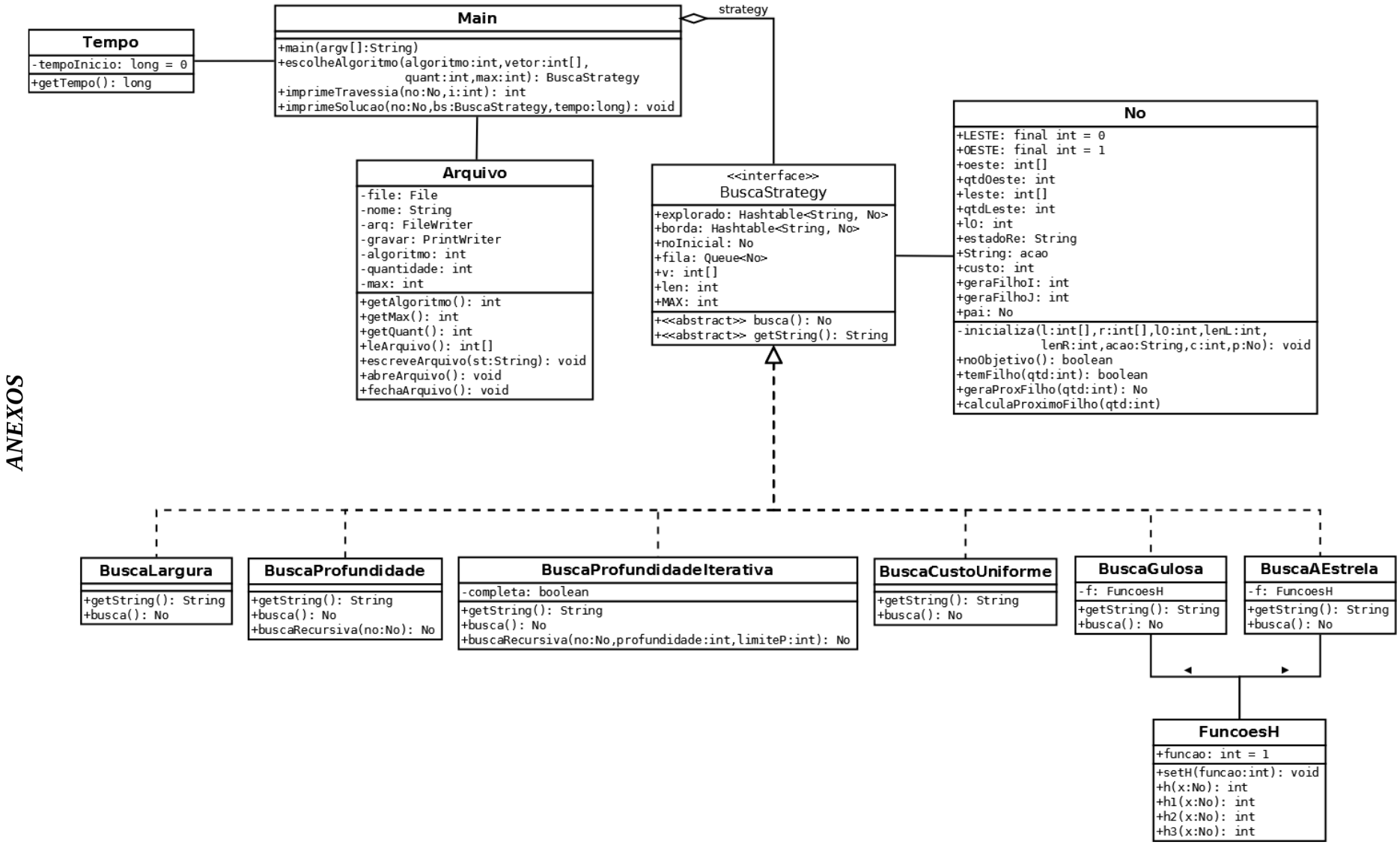
As buscas em Largura, Custo Uniforme e A* se mostraram eficiente nos testes aplicados sempre encontrando a solução ótima para o problema.

A criação e o aprimoramento de heurísticas é uma estratégia que visa melhorar a eficiência das buscas informadas, e reduzir seu custo computacional.

Em relação ao custo do caminho as três heurística apresentadas funcionam bem para a busca A* com destaque pra Heurística 3, que também é a melhor na busca Gulosa.

Problemas de busca de caminho podem até parecer simples *a priori*, mas revelam certa complexidade ao possuir um grande número de estados e de nós gerados em suas permutações, de forma que os algoritmos precisem ser adaptados para exigência de performance visando tentar achar a melhor solução possível dada uma restrição de tempo.

ANEXOS



Anexo 1. Diagrama de Classes (UML) do Exercício Programa implementado.