Trabalho Prático Nº2: D31 The Last Search

Alexandre Serra, João Fernandes e João Dinis

Abril 2020

Abstract

Os algoritmos de procura cega são algoritmos de pouca complexidade teórica, porém grande complexidade analítica. Este trabalho foca-se na análise e comparação de três algoritmos cegos, com características semelhantes, porém como se verificou, comportamentos bastante distintos. Foram documentados mais de 600 testes, para 7 ambientes com características distintas, que testam as capacidades dos algoritmos ao limite. Num cômputo geral, observou-se que com um escalonador de temperatura adaptativo e uma temperatura alta de $T_0 = 100000$, o algoritmo Recristalização Simulada foi o que teve, na maioria dos ambientes, melhor desempenho.

Keywords - Inteligência Artificial, Agentes de Procura, Procura Cega, Pesquisa Aleatória, Trepa Colinas, Recristalização Simulada

Conteúdos

1 Introdução

Este trabalho foca-se em testar a performance de algoritmos de procura cega em diversos ambientes com níveis de dificuldade distintos. Em particular, são testados os algoritmos de Procura Aleatória, Trepa Colinas e Recristalização Simulada.

2 Implementação

Foi nos fornecido um pacote que incluía o ambiente Unity pré-configurado, reservando-nos unicamente a concepção do código de três algoritmos. Para os escrevermos, apenas tivemos de passar o pseudo-código que nos foi dado para a linguagem C#. O algoritmo de Pesquisa Aleatória já veio implementado junto ao pacote, deixando para nosso trato os algoritmos Trepa Colinas e Recristalização Simulada. Nas seguintes sub-secções, passaremos a detalhar o seu funcionamento.

2.1 Pesquisa Aleatória

O algoritmo de Pesquisa Aleatória é um algoritmo cego de procura estocástica que, por definição, escolhe de forma aleatória o próximo nó a ser analisado e expandido.

A implementação usada, que nos foi devidamente fornecida, segue uma abordagem ligeiramente diferente da original: em vez de escolher de forma aleatória o próximo nó, gera uma lista de nós totalmente independente da anterior, ou seja, uma solução completamente nova.

O algoritmo é relativamente simples: começa por gerar uma solução nova, e determina o seu custo. Se esse custo for inferior ao melhor custo até ora, então essa passa a ser a melhor solução, e o melhor custo é atualizado. Se não o for, então descarta essa solução e continua até ao número de iterações máximo.

À primeira vista, este parece um péssimo algoritmo, porém para situações em que não existe informação que nos possa auxiliar e espaços de procura grandes, este algoritmo pode surpreender, pois a probabilidade de escolher um estado

prévio é baixa. Como é evidente, este algoritmo não é completo nem discriminador.

2.2 Trepa Colinas

O algoritmo Trepa Colinas é um algoritmo cego de procura heurística. Este assenta sob um princípio de algoritmo guloso: ir melhorando progressivamente a solução, fazendo em cada passo a melhor decisão local, podendo esta não ser a melhor decisão global.

A heurística considerada neste problema é o custo da solução, que corresponde à soma das distâncias entre as caixas, pelo caminho correspondente.

De forma sumária, o algoritmo começa por gerar uma solução aleatória, por onde começará a sua procura. De seguida, gera uma solução vizinha que, na prática, troca dois elementos aleatoriamente da lista. Com isto, compara a heurística (custo) das duas soluções e, se a nova solução tiver melhor custo do que o obtido até então, essa passa a ser a solução atual e o melhor custo é atualizado. Se não, descarta-se esta solução e repete-se o processo.

O maior problema deste algoritmo é que, devido à sua natureza gulosa, dependendo do caminho inicial gerado aleatoriamente, o algoritmo poderá ser levado a crer que está perante um máximo global quando não o está. Este algoritmo não é completo nem discriminador.

2.3 Recristalização Simulada

O algoritmo de Recristalização Simulada é um algoritmo cego de procura estocástica que é em tudo semelhante ao Trepa Colinas, mas que tenta evitar o maior problema desse, que é cair em máximos locais. Para fazer isto, mesmo que o valor heurístico da nova solução seja pior que o da melhor até então, a primeira pode, mesmo assim, vir a ser escolhida. Esta possibilidade depende de dois fatores: por um lado, quanto pior for a nova solução comparativamente com a atual, menor será a probabilidade de ser escolhido. Por outro, quanto maior for o resultado da função de escalonamento da temperatura, também a probabilidade será menor.

Fatores importantes no desempenho do algoritmo são, então, a temperatura inicial e a função de escalonamento de temperatura, merecendo especial atenção a última. Depois de uma pesquisa aprofundada, selecionamos 4, que foram:

- **Boltzman Annealling** - $T_k = T_0/ln(k)$ - Considerado o escalonador "clássico", é baseado na distribuição de Boltzman [2]
- **Fast Annealling** - $T_k = T_0/k$ - Baseado na distribuição de Cauchy [8]
- **Very Fast Annealling** - $T_k = T_0/k^(1/D)$ - Defendido por Lester Ingber num dos seus muitos trabalhos na área da recristalização simulada [3]
- **Adaptative Annealling** - Se a nova solução for melhor que a atual: $T_k = \alpha * T_{k-1}$ se não, mantém a temperatura - Concebido mais uma vez por Lester Ingber no seu trabalho [4]

Para além destas referências, ainda nos foi útil os trabalhos [5] [7], que fazem uma comparação aprofundada de vários métodos de escalonamento de temperatura, embora em contextos diferentes do âmbito deste trabalho.

3 Metodologia

Parâmetros gerais	**Valor**
Seed Aleatória	0, 500, 1000, 1500, 2000
Gerador de números pseudo-aleatórios	Marsaglia's Xorshift 128 [6]
Número de iterações	100, 500, 1000, 1500, 2000, 10 000 ou 50 000
Parâmetro de função de escalonamento	**Valor**
Temperatura inicial	3, 5, 10, 15, 25, 50, 75, 100, 1 000, 10 000, 100 000
Parâmetros de Very Fast Annealling	**Valor**
D	3, 6, 9
Parâmetros de Adaptative Annealling	**Valor**
α	0.25, 0.50, 0.75

Tabela 1: Parâmetros experimentais

A Tabela 1 lista os principais parâmetros usados nos nossos testes. Cada teste que fizemos, fizemos para 5 seeds diferentes, de modo a ter resultados minimamente sofríveis: temos noção que para um estudo mais aprofundado é insuficiente, mas para o âmbito deste trabalho achámos aceitável.

Relativamente ao número de iterações, fomos incrementando de modo a poder observar as melhorias dos algoritmos com cada vez mais repetições. Foi feito para cada algoritmo de cada mapa um teste com 100 000 iterações e com a seed que obteve melhor resultado (menor custo). Este teste serviu essencialmente para testarmos, num âmbito mais exaustivo, o comportamento do algoritmo e, em particular, se com um elevado número de iterações este poderia obter uma solução ainda melhor.

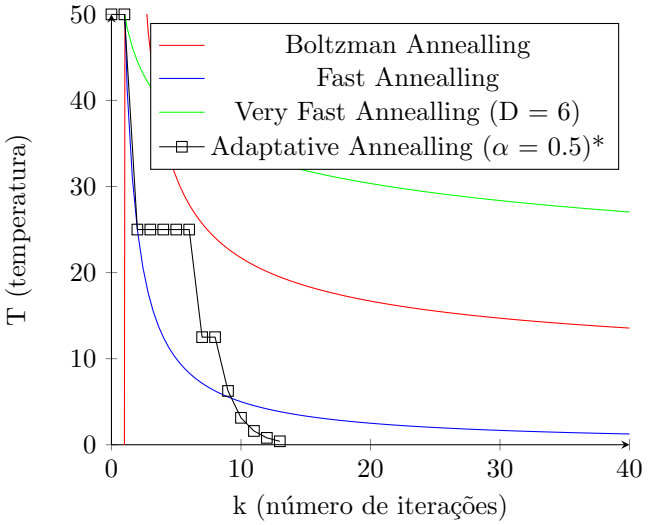
Quanto à temperatura inicial nas funções de escalonamento, estas dependeram muito dos requisitos de cada algoritmo. Como podemos ver no Gráfico 1, para uma temperatura inicial de 50, estas apresentam comportamentos completamente distintos. Portanto, a parametrização teve de ser específica em relação à função escolhida.

Sumariamente o que fizemos para a pesquisa aleatória e o trepa colinas foi o seguinte: começar com um número de iterações de 100, testar para as nossas 5 random seeds e apontar a melhor solução, o número de iterações aquando o algoritmo encontrou a melhor solução e o caminho correspondente. De seguida, repetir o processo para 500, 1000, 1500 e 2000 iterações. No final, a seed que teve melhor resultado é usada para fazer o que chamamos *force test*, em que testamos com 100 000 iterações para ver se o algoritmo é capaz de encontrar ainda melhor solução.

Para os testes de recristalização simulada, fizemos o sugerido pelos docentes: testamos os 4 algoritmos no mapa mais difícil, que era o de nome Pacman, e testamos com várias configurações. Depois utilizámos a que achámos, por

experimentação, a melhor função com os seus melhores parâmetros nos restantes mapas. Os resultados são apresentados na secção seguinte.

Gráfico 1: Comparação entre diferentes funções de escalonamento



4 Resultados

Nesta secção iremos apresentar os resultados individualizados por mapa, e fazer uma breve análise dos mesmos. De notar que, embora tenhamos feito testes para o primeiro mapa (NoObstacles), decidimos não os incluir nesta análise pelo facto de o mapa ser demasiado simples, e os resultados serem inconclusivos.

4.1 ObstaclesSmall

Este mapa apresentava um ambiente simples com 6 caixas à volta do centro. O desafio do algoritmo de procura era encontrar o caminho que permitia à bola dar uma volta em torno do centro do mapa. Devido à simplicidade do mapa, o algoritmo de pesquisa aleatória consegue atingir, na maioria dos casos, a solução ótima em menos de 1000 iterações. O algortimo Trepa Colinas demonstrou algumas limitações pois, por diversas vezes, entrou num "loop", não permitindo atingir a solução ótima. Por outro lado, quando conseguia atingir a solução ótima, fazia-o em menos de 20 iterações. O algoritmo de Recristalização Simulada apresentou resultados piores que os outros dois algoritmos. Apenas atingiu a solução ótima por uma vez e quando o fez demorou mais de 100 iterações.

Parâmetro	Valor
Melhor solução Pesquisa Aleatória	25
Iteração quando encontrada melhor solução	61
Média de melhores soluções (1000 it.)	25.4
Média de iteração quando encontrada melhor solução (1000 it.)	480.8
Melhor solução Trepa Colinas	25
Iteração quando encontrada melhor solução	4
Média de melhores soluções (1000 it.)	27.4
Média de iteração quando encontrada melhor solução (1000 it.)	13
Melhor solução Recristalização Simulada - Adaptative	25
Iteração quando encontrada melhor solução	105
Média de melhores soluções (1000 it.)	28.4
Média de iteração quando encontrada melhor solução (1000 it.)	50

Tabela 2: Resultados para o mapa ObstaclesSmall

4.2 ObstaclesSmallManyBox

Este mapa, ao contrário dos anteriores, já opôs perante um cenário mais adverso: 12 caixas distribuídas por um cenário com várias paredes. E pelos resultados, este é um excelente exemplo do potencial da Recistalização Simulada (usando Very Fast Annealling): embora a média de soluções não tenha sido melhor que a do Trepa Colinas, encontrou a mesma melhor solução, numa iteração já muito avançada, algo impensável para o algoritmo Trepa Colinas.

Parâmetro	Valor
Melhor solução Pesquisa Aleatória	44
Iteração quando encontrada melhor solução	17266
Média de melhores soluções (5000 it.)	47.2
Média de iteração quando encontrada melhor solução (5000 it.)	2688.2
Melhor solução Trepa Colinas	30
Iteração quando encontrada melhor solução	271
Média de melhores soluções (5000 it.)	32.6
Média de iteração quando encontrada melhor solução (5000 it.)	311.2
Melhor solução Recristalização Simulada - Very Fast	30
Iteração quando encontrada melhor solução	4784
Média de melhores soluções (5000 it.)	35.6
Média de iteração quando encontrada melhor solução (5000 it.)	4887.6

Tabela 3: Resultados para o mapa ObstaclesSmallManyBox

4.3 ReturnTo2b

Este mapa contem 9 caixas num ambiente com algumas paredes. O número de caixas permite que o algoritmo de pesquisa aleatória ainda consiga por vezes

encontrar soluções interessantes para um número de iterações elevado. No entanto quando comparado com os outros dois algoritmos notamos que fica aquém. Os algoritmos Trepa-Colinas e Recristalização Simulada demonstram resultados idênticos, ainda que, o Trepa-Colinas apresenta melhor média e consegue encontrar o melhor resultado mais vezes que a Recristalização Simulada.

Parâmetro	**Valor**
Melhor solução Pesquisa Aleatória	67
Iteração quando encontrada melhor solução	2220
Média de melhores soluções (5000 it.)	69.8
Média de iteração quando encontrada melhor solução (5000 it.)	6104.6
Melhor solução Trepa Colinas	63
Iteração quando encontrada melhor solução	45
Média de melhores soluções (5000 it.)	68.2
Média de iteração quando encontrada melhor solução (5000 it.)	96.8
Melhor solução Recristalização Simulada - Adaptative	63
Iteração quando encontrada melhor solução	115
Média de melhores soluções (5000 it.)	69.8
Média de iteração quando encontrada melhor solução (5000 it.)	811.4

Tabela 4: Resultados para o mapa ReturnTo2b

4.4 ReturnTo2bHarder

Este mapa é idêntico ao anterior, no entanto apresenta mais caixas (17), o que aumenta a dificuldade em encontrar a melhor solução. Ao contrário do mapa anterior, o algoritmo de Pesquisa Aleatória não se provou ser vantajoso em caso algum. O algoritmo Trepa-Colinas continuou a apresentar resultados bons, sendo batido apenas pelo algoritmo de Recristalização Simulada, que conseguiu apresentar as melhores médias, bem como a melhor solução.

4.5 Bomberman

Este mapa apresentava um ambiente mais complexo que os anteriores, com 19 caixas e uma grande densidade de paredes. O algoritmo de pesquisa aleatória apresentou os piores resultados, demorando sempre um número elevado de iterações a atingir a melhor solução. O algoritmo Trepa Colinas e Recristalização Simulada apresentaram resultados bastantes similares, sendo que o primeiro conseguiu atingir a melhor solução mas o segundo apresentou melhor média das melhores soluções. O número de iterações para atingir a melhor solução foi semelhante nos dois algoritmo, andando maioritariamente em valores entre 1000 e 2000.

Parâmetro	Valor
Melhor solução Pesquisa Aleatória	119
Iteração quando encontrada melhor solução	1181
Média de melhores soluções (5000 it.)	129.8
Média de iteração quando encontrada melhor solução (5000 it.)	1871.6
Melhor solução Trepa Colinas	81
Iteração quando encontrada melhor solução	225
Média de melhores soluções (5000 it.)	89.2
Média de iteração quando encontrada melhor solução (5000 it.)	985.6
Melhor solução Recristalização Simulada - Adaptative	77
Iteração quando encontrada melhor solução	1093
Média de melhores soluções (5000 it. e $= 0.5$)	87.6
Média de iteração quando encontrada melhor solução (5000 it. e $= 0.5$)	644.6
Média de melhores soluções (5000 it. e $= 0.98$)	86.6
Média de iteração quando encontrada melhor solução (5000 it. e $= 0.98$)	2098.2

Tabela 5: Resultados para o mapa ReturnTo2bHarder

Parâmetro	Valor
Melhor solução Pesquisa Aleatória	116
Iteração quando encontrada melhor solução	15444
Média de melhores soluções (100.000 it.)	121.2
Média de iteração quando encontrada melhor solução (100.000 it.)	27939.4
Melhor solução Trepa Colinas	75
Iteração quando encontrada melhor solução	1197
Média de melhores soluções (100.000 it.)	87.8
Média de iteração quando encontrada melhor solução (100.000 it.)	1344.2
Melhor solução Recristalização Simulada - Adaptative	77
Iteração quando encontrada melhor solução	1210
Média de melhores soluções (100.000 it.)	85.6
Média de iteração quando encontrada melhor solução (100.000 it.)	1651.2

Tabela 6: Resultados para o mapa Bomberman

4.6 Pacman

Este mapa, que consideramos o mais difícil, foi o que utilizámos para testar exaustivamente os escalonadores de temperatura, de modo a poder escolher depois o melhor para usar nos demais mapas. Como se pode observar na Tabela 7, os resultados entre escalonadores são muito semelhantes, para as suas melhores configurações (que encontramos). Tivemos então de escolher um, cujo critério foi escolher o com o melhor média de número de iterações. Esse é, pois, o Adaptativo, que tem características diferentes dos outros, pelo seu decréscimo acentuado.

Comparativamente com o Trepa Colinas, o Adapativo, em termos de média de soluções, foi o único melhor. Porém achamos seguro dizer que qualquer um dos outros também teve um bom desempenho.

Parâmetro	Valor
Melhor solução Pesquisa Aleatória	360
Iteração quando encontrada melhor solução	846
Média de melhores soluções (5000 it.)	384
Média de iteração quando encontrada melhor solução (5000 it.)	2205
Melhor solução Trepa Colinas	210
Iteração quando encontrada melhor solução	3757
Média de melhores soluções (5000 it.)	232.6
Média de iteração quando encontrada melhor solução (5000 it.)	1841.6
Melhor solução Recristalização Simulada - Boltzman ($T_0 = 5$)	222
Iteração quando encontrada melhor solução	3193
Média de melhores soluções (5000 it.)	234.2
Média de iteração quando encontrada melhor solução (5000 it.)	2944
Melhor solução Recristalização Simulada - Fast ($T_0 = 25$)	228
Iteração quando encontrada melhor solução	1116
Média de melhores soluções (5000 it.)	236.8
Média de iteração quando encontrada melhor solução (5000 it.)	1895.6
Melhor solução Recristalização Simulada - Very Fast ($T_0 = 5$, D = 6)	215
Iteração quando encontrada melhor solução	4879
Média de melhores soluções (5000 it.)	234.4
Média de iteração quando encontrada melhor solução (5000 it.)	4872
Melhor solução Recristalização Simulada - Adaptative ($T_0 = 100\ 000$, $\alpha = 0.5$)	222
Iteração quando encontrada melhor solução	1009
Média de melhores soluções (10 000 it.)	229
Média de iteração quando encontrada melhor solução (10 000 it.)	1539.8

Tabela 7: Resultados para o mapa Pacman

5 Discussão

Nesta secção iremos discutir os resultados obtidos da experimentação, bem como tentar perceber qual é o algoritmo que nos consegue dar mais garantias de encontrar a melhor solução para o problema.

Na secção de implementação foi feita uma descrição do funcionamento de cada algoritmo e uma pequena análise ao seu comportamento esperado. Através dos resultados obtidos podemos verificar se a hipótese teórica vai de encontro à prática.

5.1 Pesquisa Aleatória

Este algoritmo teve o comportamento esperado. Num ambiente com um número reduzido de caixas consegue encontrar uma solução boa, no entanto precisa de um elevado número de iterações. Já quando o ambiente é mais complicado o seu comportamento não melhora muito e fica muito aquém da melhor solução. Basta só reparar que quando temos, por exemplo, 9 caixas, temos um número de possibilidades igual a 362 880. Logo, torna-se difícil encontrar a melhor solução com um número de iterações menor que 5000.

Concluímos então que este algoritmo é muito dependente no fator sorte (i.e. probabilidade). Se tivermos sorte, este poderá atingir uma solução ótima ou sub-

ótima, mas com o aumento do espaço de procura a probabilidade de encontrar tal solução é cada vez menor. É um algoritmo simples de implementar e portanto, pode ser sempre uma hipótese, especialmente quando as nossas informações sobre o problema são perto de nulas.

5.2 Trepa Colinas

Em 6 dos 7 ambientes foi capaz de encontrar a melhor solução (quando comparado com os outros dois algoritmos), no entanto em termos médios apenas obteve a melhor solução para 3 dos 7 mapas, ainda que precise de menos iterações para encontrar a melhor solução. É um algoritmo que por vezes encontra a solução muito cedo e não consegue melhorar, independentemente do número de iterações, o que vai ao encontro da ideia que o algoritmo é guloso e não consegue sair, por vezes, dos máximos locais encontrados.

5.3 Recristalização Simulada

Este é um algoritmo que tem na sua génese o Trepa Colinas. No entanto através da função de escalonamento de temperatura é possível sair dos máximos locais do Trepa Colinas. A melhor função encontrada (dentro das testadas) foi a Adaptative Annealing, onde a temperatura é variada apenas nos casos em que a nova solução não é melhor do que a atual. Foi capaz de encontrar a melhor solução num mapa, sendo que as suas médias foram as melhores para 4 dos 7 mapas. Apesar do seu comportamento ter sido bom, sofre de alguns problemas: apresenta, por vezes, resultados inesperados por decidir usar uma solução pior, comparada com a das últimas iterações.

Tem, então, papel determinante no sucesso do algoritmo a temperatura inicial e a função de escalonamento: se o primeiro for demasiado baixo, o comportamento será equivalente ao do Trepa Colinas. Por outro lado, se este for demasiado alto, a aleatoriedade do algoritmo terá demasiado protagonismo, o que não poderá, ou não, ser vantajoso. Isto tem de jogar em consonância com o método de escalonamento, que tem de permitir descer a temperatura de forma gradual.

5.4 Como é que sai da subsection anterior?

Depois de analisados todos os algoritmos, podemos concluir que, a escolha do algoritmo depende do contexto prático.

Se tivermos um espaço de procura grande, o algoritmo de Pesquisa Aleatória pode ser um tiro no escuro. Isto será vantajoso se pouca informação tivermos sobre o algoritmo. No entanto, raras são as ocasiões em que isso acontece, o que inviabiliza este algoritmo.

No caso de termos informações sobre o problema, a decisão entre o Trepa Colinas e a Recristalização Simulada entra em cena. Se por um lado o Trepa Colinas ganha pela sua facilidade de implementação e rapidez em atingir uma

solução ótima ou sub-ótima, este não consegue fazer a distinção entre os dois tipos.

O algoritmo de Recristalização Simulada, se bem parametrizado, é o melhor algoritmo. O problema é exatamente esse: a sua boa parametrização. Para a atingir, um estudo prévio do ambiente deve ser feito, o que nem sempre é fácil, nem visível sem experimentação. É, portanto, o algoritmo com mais potencial de obter uma solução ótima. No nosso caso, a função de escalonamento Adaptative Annealing com um = 0.5 e um temperatura inicial = 100 000 foi a que teve melhor desempenho mas, poderá existir outra função ainda melhor, o que abre a possibilidade a uma análise mais aprofundada do problema.

Em baixo deixamos um gráfico que, começando em "Quer melhor algoritmo?", de forma simplificada poderá ajudar o leitor a decidir qual o algoritmo que mais se adequa às suas necessidades.

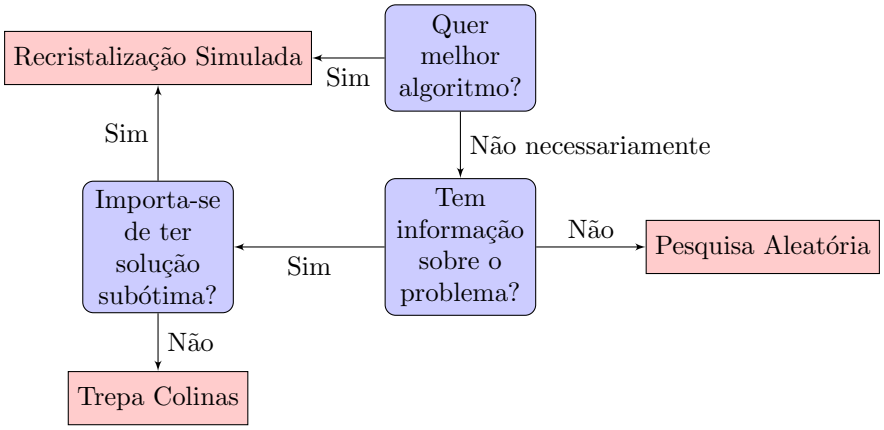


Gráfico 2: Gráfico auxiliar para decisão de algoritmo

6 Conclusões

Os algoritmos analisados, de procura cega e todos com alguma componente estocástica, são algoritmos de grande potencial. Todos têm uma fraqueza: na Pesquisa Aleatória a sua extrema aleatoriedade, no Trepa Colinas a sua aptidão para cair em máximos locais e na Recristalização Simulada a sua apetência para perder uma solução melhor em detrimento de outra pior.

A sua utilidade está, portanto, inerente ao contexto em que é usada e também, no caso particular da Recristalização Simulada, da sua parametrização.

Nos nossos testes o algoritmo que obteve melhores resultados foi a Recristalização Simulada. Porém, para atingir um valor melhor que o dos outros algoritmos, tivemos de fazer muita experimentação de modo a chegar a uma boa parametrização, enquanto os outros algoritmos, em particular o Trepa Colinas, têm resultados instantâneos e é possível inferir o seu comportamento de forma muito mais célere.

7 Reconhecimentos

Com este trabalho foi-nos possível perceber de forma aprofundada vários algoritmos de procura cega e os fundamentos de agentes de procura. Portanto, queríamos agradecer aos nossos professores das aulas práticas, João Nuno Correia e Nuno António Lourenço pelo apoio ao trabalho, quer nas aulas, quer no horário de atendimento, e ao professor regente da cadeira de Introdução à Inteligência Artificial Fernando Penousal Machado, que nos introduziu aos conceitos de agentes de procura.

Por fim queríamos agradecer à Universidade de Coimbra pela oportunidade de ensino e, também, pela forma exemplar como tem lidado com esta situação anormal.

8 Material Suplementar

Em anexo com este relatório encontra-se um ficheiro do tipo Excel, que contém documentados todos os testes efetuados.

A este juntam-se um conjunto de ficheiros de código em C# que constituem as implementações algorítmicas usadas na realização deste trabalho.

Referências

[1] Ernesto Costa and Anabela Simões. Inteligência artificial - fundamentos e aplicações. 02, 09 2008.

[2] Donald Geman. Stochastic relaxation, gibbs distributions,and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:721 – 741, 01 1984.

[3] A Ingber and Lester Ingber. Very fast simulated re-annealing. *Mathematical and Computer Modelling*, 12, 02 2002.

[4] Lester Ingber. Adaptive simulated annealing (asa): Lessons learned. *Control Cybernetics*, 25:33–54, 01 1996.

[5] Walid Mahdi, Seyyid Ahmed Medjahed, and Mohammed Ouali. Performance analysis of simulated annealing cooling schedules in the context of dense image matching. *Computación y Sistemas*, 21:493–501, 10 2017.

[6] George Marsaglia. Xorshift rngs. *Journal of Statistical Software*, 08, 01 2003.

[7] Yaghout Nourani and Bjarne Andresen. A comparison of simulated annealing cooling strategies. *J. Phys. A: Math. Gen*, 31:8373–8385, 10 1998.

[8] Harold Szu and Ralph Hartley. Fast simulated annealing. *Physics Letters A*, 122:157–162, 06 1987.