Universidade de Coimbra

Faculdade de Ciências e Tecnologia

Departamento de Engenharia Informática

Introdução à Inteligência Artificial

2010/2011 – 2º Semestre

Trabalho Prático Nº2: Curva Braquistócrona

Meta Final

Realizado por:

Diogo Filipe Coimbra Pinheiro	Nº 2007183552	dfcp@student.dei.uc.pt
Pedro Miguel Ricardo Geadas	Nº 2006131902	pmrg@student.dei.uc.pt

Índice:

Introdução

Neste trabalho foi-nos pedido para desenvolver um algoritmo evolucionário de maneira a resolver o problema da curva braquistócrona. (http://pt.wikipedia.org/wiki/Braquist%C3%B3crona)

A representação escolhida, os operadores genéticos, o mecanismo de selecção, de atribuição de fittnes, são pois componentes essenciais que tivemos bastante em conta visto o objectivo final – a obtenção da melhor curva. Deste modo, iremos no presente relatório analisar os resultados obtidos através do teste de várias configurações de valores, quer em relação aos próprios operadores genéticos, quer em relação ao número de gerações e de população, quer ao número de pontos utilizado. Estes valores podem facilmente ser alterados através do GUI desenvolvido, que era facultativo, mas que apesar disso se tornou essencial ao facilitar a tarefa de testes propriamente dita.

Além de tabelas e gráficos, que irão tentar ilustrar o pretendido da melhor maneira possível, este trabalho contemplará ainda um pequeno manual do utilizador e outro do programador, onde serão tratados assuntos pertinentes.

Manual do Utilizador

Parâmetros de entrada e operadores genéticos

Nesta secção iremos falar de como utilizar o programa e em que consiste cada um dos parâmetros de entrada. Os operadores evolutivos (genéticos), tal como o nome indica, simulam num computador o que tem vindo a ser feito pela Natureza ao longo dos tempos e que de outro modo seria impossível de observar: a evolução! E claro é que esta está directa e condicionalmente relacionada com a genética. Assim sendo, foram desenvolvidos operadores genéticos que pretendem simular mecanismos que ocorrem na própria Natureza e que promovem a dita evolução. São exemplo disso os "Crossovers" e as "Mutações".

Para que o utilizador fique esclarecido quando utilizar o programa, segue um breve resumo de cada um dos campos passíveis de ser alterados: (todos os campos estão protegidos contra dados inválidos)

Características da população:

1. **Número de Gerações:** Este parâmetro tem que ser um inteiro e representa o número de gerações que irão ser criadas.

2. **Tamanho da População:** Tem que ser um número inteiro par e representa o número de indivíduos (neste caso, conjuntos de pontos) da nossa população.

3. **Número de Pontos:** Número de pontos por indivíduo. (inteiro)

4. **Tipo de Representação:** Representação de cada indivíduo na população.

1- Partições iguais: cada ponto dos nossos indivíduos distam, em X, de uma igual distância entre si, enquanto que o Y correspondente é aleatório;

2- Partições aleatórias: cada ponto consiste numa coordenada X e Y aleatórias, não existindo restrições ao nível da distância que separa cada X, neste caso.

5. **Percentagem de Elitismo:** Define a percentagem de indivíduos (dos que obtiveram melhor fitness) que irão ser preservados para a próxima geração. (0 a 1)

De relevar que todos os pontos têm que estar contidos entre o ponto inicial e o final, sendo considerados inválidos caso tal não aconteça. O nosso programa garante a validade dos mesmos por defeito.

Métodos de Selecção:

1. **Modo de Selecção:** Modo como iremos seleccionar os pais em cada geração. Estes métodos irão, de uma maneira ou outra, tentar dar primazia a indivíduos com melhor fitness. (1 ou 2)

   1- Selecção por Torneio: selecciona N filhos aleatoriamente da população e no final escolhe o melhor desses;

   2- Selecção por Roleta: faz a escolha com base em probabilidades directamente proporcionais com a qualidade do fitness dos indivíduos.

2. **Tamanho do Torneio:** Este parâmetro irá influenciar a escolha no caso da selecção por Torneio e diz-nos quantos indivíduos irão ser seleccionados, antes de ser escolhido o melhor. (inteiro)

Operadores genéticos:

1. **Probabilidade de Crossover:** Probabilidade com que irá ocorrer uma recombinação dos genes entre indivíduos. (0 a 1)

2. **Número de pontos de Crossover:** Número de pontos de corte do gene dos indivíduos quando estes fazem recombinação. (inteiro)

3. **Tipo de Mutação:** Existem 2 tipos de mutação:

   1- *Mutação normal:* quando ocorre, é escolhido aleatoriamente um ponto e a sua coordenada x ou y é alterada por outra escolhida aleatoriamente e que mantenha o indivíduo válido.

   2- *2- Mutação gaussiana:* (perturbações gaussianas) igual ao anterior, mas escolhe a nova coordenada de entre um intervalo próximo ao valor da coordenada onde vai ocorrer a mutação.

4. **Probabilidade de Mutação:** Probabilidade com que irão ocorrer mutações nos nossos indivíduos. (0 a 1)

Target:

1. **Target:** O target corresponde à especificação do ponto inicial e final, sobre os quais iremos desenhar a nossa curva.

Quando o programa se encontra em execução, são impressos diversos valores para o ecrã, como por exemplo a média de fitness do melhor/pior indivíduos de cada geração, o melhor e o pior indivíduo de cada geração, o desvio padrão, bem como a média final. Todos os valores são também guardados em ficheiros.

Manual do Programador

Algoritmos e métodos

Por detrás da interface gráfica, encontra-se um poderoso motor evolutivo composto por diversos métodos que desempenham um papel muito importante para o correcto funcionamento do mesmo. Nesta secção pretendemos esclarecer a função de cada método criado. Todo o trabalho está escrito em Python.

1. **Ficheiro BrachFitness.py:** É aqui que está definida o método que irá medir o fitness dos nossos indivíduos. Faz também a verificação da validade de um indivíduo.
2. **Ficheiro GUI.py:** Interface GUI e chamada da função principal.
3. **Ficheiro SGA_BC.py:** Os restantes métodos estão aqui definidos:

- Métodos que que geram os indivíduos, consoante o tipo de representação escolhida:

1- **create_indiv_same_spacement(Target, num_pontos);**

2- **create_indiv_random_spacement(Target, num_pontos);**

Ambos recebem um Target (ponto inicial e final) e o Número de Pontos como parâmetros.

- Métodos responsáveis pela escolha do tipo de Selecção:

1 **- tournament_selection(population, size_tournament);**

2 **- roulette_selection(population, total);**

Ambos recebem a população como parâmetro. No caso da selecção por torneio, recebe também o "Tamanho do Torneio" enquanto que a roleta recebe o fitness "total" da população, para efeitos de cálculo de probabilidades. (O "Tamanho do Torneio" corresponde ao número de indivíduos escolhidos aleatoriamente antes de se escolher o melhor deles; o "total" corresponde à soma dos fitnesses individuais de cada indivíduo, por população.)

- Método que escolhe os X melhores indivíduos de uma população, ou elites,

elitism_selection(parents, offspring, size_elite), que irão ser preservados para a próxima geração. Deste modo, impede-se que se eliminem todos os bons candidatos de uma geração, que poderiam levar gerações para aparecer novamente.

- Métodos que introduzem mutações:

  1 – mutation(indiv, alphabet);

  2 - mutation_gaussian(indiv, alphabet);

  3 - mutation_gaussian_xy(indiv, alphabet);

O primeiro apenas introduz mutação em Y (gera um valor aleatório entre o mínimo e o máximo permitidos); o segundo introduz uma perturbação gaussiana também apenas em Y; o terceiro, que apenas é utilizada no caso das partições serem aleatórias, já que pode haver variação no espaçamento entre X's introduz perturbações gaussianas em X ou em Y, aleatoriamente.

- Métodos responsáveis pelas operações de Crossover entre indivíduos:

  1- **crossover_random_spacement(parent_1, parent_2, n_crossover),** caso se trate de partições aleatórias;

  2- *2-* **crossover_same_spacement(parent_1, parent_2, n_crossover),** caso se trate de partições iguais.

Ambos recebem dois indivíduos (os pais) e o número de pontos de corte que irão ser feitos. (detalhado na Secção Problemas e Soluções deste relatório)

- O nosso método principal, onde tudo é então chamado consoante os parâmetros fornecidos pelo Utilizador: **sga()**: (os parâmetros foram omitidos porque são todos os que já aqui falamos). Começa por criar a população, chamando **create_population(size_pop, Target, num_pontos, spacement),** que irá escolher o tipo de indivíduos a ser criado, bem como o seu tamanho da população, chamando um dos dois métodos de criação falados no primeiro ponto. De seguida, mede o fitness de cada indivíduo da população utilizando o método do ficheiro apresentado acima (1) e, consoante o tipo de selecção definido, escolhe uma população que irá ser submetida (possivelmente) a recombinações e mutações. Após efectuar estas operações, é medido novamente o fitness dos indivíduos e definida a próxima população, que será a junção de X elites da população anterior, com Z indivíduos da nova população (offspring), onde Z= Nº indivíduos – X, para que o tamanho seja sempre o mesmo. Repete o processo 30 vezes, de modo a que no final imprima a estatística pretendida para o trabalho.

Todo o código está devidamente comentado, de maneira a que seja mais perceptível o que acontece em cada método.

Todos os dados são guardados em ficheiros.

Problemas e soluções

Ao longo do desenvolvimento do projecto, foram surgindo alguns problemas ao nível da implementação de alguns dos operadores presentes no mesmo, os quais causaram ainda alguma dor de cabeça. Não obstante, todos foram devidamente analisados e solucionados pelo que pensamos ter sido a melhor solução para o problema em causa.

Representação:

Ao nível da representação, deparámo-nos com alguns problemas. “Que representações usar? Qual será a que melhores resultados produz?”, foram algumas das questões que nos colocámos. Decidimos então usar:

1- Representação em que apenas os y's eram sujeitos a mudança e em que a distância entre x's consecutivos era *a mesma*.

2- Representação em que tanto x's como y's eram gerados *aleatoriamente*.

Foi com esta segunda implementação que os problemas surgiram. Para a primeira, apenas tínhamos a preocupação de arranjar Y's contidos entre o intervalo máximo e mínimo, já que os X's estavam garantidamente na posição correcta tanto na fase de geração de indivíduos, tanto na fase de crossover e mutação. Para a segunda, cuidados extra tiveram de ser tidos em conta. Para além das preocupações de validade consideradas para a primeira, houve também a preocupação de manter sempre os X's ordenados no final e de não haver X's iguais. Para tal foram geradas as coordenas x e y independentemente uma da outra (em arrays diferentes, 'xx' e 'yy'), as coordenadas em 'xx' foram posteriormente ordenadas e foi finalmente feita a junção dos dois para um array de pontos final, composto por um elemento de 'xx' e outro de 'yy' alternadamente, até não existirem elementos em qualquer dos arrays auxiliares. Feito isto, temos então um array do género [Xi,Yi, x1, y1, (…), xn, yn, Xf, Yf], que representa então um indivíduo.

Selecção:

Os dois mecanismos de selecção desenvolvidos foram, como já foi também referido, a selecção por Torneio, mais eficiente computacionalmente, e o mecanismo de Roleta. Quanto à selecção por Torneio, não houve qualquer problema. O mesmo não podemos dizer em relação à Roleta, tendo sido contudo facilmente resolvido. O problema tinha que ver com a função de avaliação de fitness, uma vez que esta nos retornava um valor tanto mais pequeno quanto melhor fosse o fitness. Para o resolver tivemos de inverter as probabilidades, de modo a que a valores mais pequenos (e portanto, com melhor fitness) correspondesse uma maior probabilidade. Como foi também referido, este método de selecção, menos eficiente que o anterior uma vez que tem que efectuar vários cálculos, calcula as probabilidades que cada indivíduo tem para ser escolhido com base no seu fitness individual, dividido pelo total de fitness da população onde o mesmo se encontra inserido. De outro modo, se dividíssemos apenas o nosso fitness individual pelo total, iríamos ter que a maiores valores de fitness(e portanto piores) corresponderia uma maior probabilidade. Não iremos alargar esta explicação, convém apenas reter o problema encontrado e a solução adoptada.

Crossover:

Aqui encontrámos talvez o maior problema com que tivemos de lidar. Para o caso em que usámos a representação 1 (x's igualmente espaçados), não houve qualquer problema. Independentemente do número de pontos de corte, os pedaços que iam sendo juntados estavam garantidamente ordenados em x e assim sendo, mantinha o nosso indivíduo válido. O mesmo já não aconteceu no caso em que gerámos as coordenadas x's aleatoriamente para todos os indivíduos da população. Pontos de corte iguais para ambos os pais não significava necessariamente que a junção destes iriam gerar um indivíduo válido. Este problema poderia ser resolvido de várias maneiras: ignorar esse indivíduo(e gerar outro) , mudar os pontos de crossover(o que mesmo assim poderia não garantir a validade) ou assegurar que, à medida que o crossover ia sendo processado, os pontos que iam sendo juntados não invalidavam o indivíduo. Foi a última solução que a que resolvemos implementar, por diversas razões: *eficiência, efectividade, "preservação" de indivíduos,* entre outras.

Eficiência e efectividade: quanto maior o número de pontos, maior a probabilidade do crossover gerar indivíduos inválidos. Desprezar o indivíduo e gerar outro até que eventualmente um deixasse de ser inválido, poderia consumir bastante tempo de computação e até poderia acontecer o caso em que não era possível fazer a junção sem invalidar o mesmo. Apesar de eficaz (teoricamente), este método era bastante ineficiente. A nossa abordagem, à medida que vamos avançando no crossover, vai comparando o último elemento do novo indivíduo (offspring) com os pontos a ser juntados (e que foram guardados num array auxiliar). Caso o valor comparado do array auxiliar seja maior, podemos adicionar no final do offspring porque não viola nenhuma regra de validação. No entanto, quanto o valor que temos para comparar no array auxiliar é menor, temos de percorrer o array (do final para o início, no nosso caso) e ir comparando com todos os pontos, até encontrar a posição correcta para o ponto a ser testado. Quando este array auxiliar fica vazio, avançamos no ponto de corte e temos a garantia de que todos os pontos adicionados até aí estão na posição correcta.

Preservação de indivíduos: Como não eliminamos o indivíduo, e garantimos sempre que o mesmo será válido, existe uma menor interferência da nossa parte no funcionamento do algoritmo, tornando-o mais fluido e "autónomo". Se por acaso uma determinada operação de crossover fosse potencialmente geradora de um excelente indivíduo, e este fosse inválido e descartado, poderia nunca mais voltar a aparecer e comprometer assim o resultado final.

Para que o leitor melhor compreenda o mecanismo de crossover e alguns conceitos introduzidos, como "pontos de corte", iremos dar um pequeno exemplo de seguida:

Target: [0,100, 120, 30]

2 Pontos de Corte: [2, 4]

parent_1: [0, 100, 20, 50,| 70, 20, 90, 25,| 95, 28, 120,30]

parent_2: [0, 100, 40, 60,| 60, 70, 80, 25,| 85, 27, 120, 30]

[1 2 3 4 5]

offspring _1 = [parent_1[0:2] + parent_2[2:4] + parent_1[4:]] = [0, 100, 20, 50, 60, 70, 80, 25, 95, 28, 120, 30]

offspring _2 = [parent_2[0:2] + parent_1[2:4] + parent_2[4:]] = [0, 100, 40, 60, 70, 20, 90, 25, 85, 27, 120, 30]

Repare no filho gerado (offspring_2). Facilmente se constata que não é válido pois existe um X fora do sítio, isto é, não está ordenado. Repare também que caso os X's estivessem igualmente espaçados entre si nunca iria surgir este problema, já que os pontos de corte são iguais para ambos os pais.

Mutação:

A mutação consiste em alterar o valor de uma das coordenadas de um indivíduo (neste caso, obviamente). Para o caso da representação ser a 1 (partições iguais), apenas ocorre mutação em Y de modo a não tornar o indivíduo inválido, quer seja uma mutação normal ou uma perturbação gaussiana. No caso de a representação escolhida ser a 2 (partições aleatórias) a mutação pode ocorrer em X ou Y, e contempla também os dois tipos de mutação.

Resolvidos os problemas de implementação, estávamos em condições de proceder para a fase de testes, de onde iremos tentar extrair algumas conclusões.

Análise de Resultados e Proposições

Após vários testes, podemos então retirar algumas conclusões acerca da parametrização e indicar para que gamas de valores se obtiveram os melhores resultados. Note que todo o processo teve que ser repetido várias vezes de modo a que a estatística gerada tivesse significado. O número recomendado foi de 30 repetições, no mínimo, e foi esse o valor escolhido por nós. Obviamente que quanto maior é este número, mais precisa e concludente será a estatística. No entanto, para a segunda tabela que iremos apresentar, em que o número de indivíduos é de 600 e o número de gerações é de 350, alguns casos de teste demoraram a terminar por vezes mais de 1 hora, o que nos levou a optar então pelas 30 repetições.

Posto isto, passemos então à análise dos dados recolhidos.

Nº Gerações-50 Nº de Individuos-80 Tamanho Torneio/Roleta-5 Prob. Crossover-50% Pontos Crossover-1 Prob. Mutação-10% Elitismo-20%

Número de Pontos	Partições Iguais				Partições Aleatórias			
	Mutação Normal		Mutação Gaussiana		Mutação Normal		Mutação Gaussiana	
	Torneio	Roleta	Torneio	Roleta	Torneio	Roleta	Torneio	Roleta
10	**1) 5.650356**	**1) 6.333169**	**1) 5.263642**	**1) 5.294858**	**1) 5.865535**	**1) 6.480553**	**1) 6.105794**	**1) 5.381194**
	2) 5.448356	2) 5.370094	2) 5.263209	2) 5.291667	2) 5.653714	2) 5.299487	2) 6.104410	2) 5.379518
	3) 7.559210	3) 8.265574	3) 5.267257	3) 5.298439	3) 7.846709	3) 8.404217	3) 6.107502	3) 5.382597
	4) 5.313606	4) 5.282631	4) 5.260608	4) 5.260635	4) 5.583863	4) 5.273459	4) 5.918042	4) 5.347913
	5) 10.630837	5) 11.306366	5) 5.301225	5) 5.829070	5) 12.176857	5) 12.457627	5) 6.789513	5) 5.630124
20	**1) 6.338374**	**1) 7.955889**	**1) 6.898353**	**1) 6.595817**	**1) 6.292291**	**1) 7.517957**	**1) 6.719678**	**1) 6.604071**
	2) 6.171100	2) 6.929141	2) 6.894061	2) 6.579834	2) 6.012618	2) 6.055464	2) 6.716910	2) 6.594727
	3) 8.050306	3) 10.030271	3) 6.902625	3) 6.607137	3) 8.674736	3) 9.900032	3) 6.722424	3) 6.609751
	4) 5.833451	4) 6.808634	4) 5.964474	4) 5.440684	4) 5.895076	4) 5.933492	4) 6.477067	4) 6.206861
	5) 13.350812	5) 20.530903	5) 10.598033	5) 11.070351	5) 18.433579	5) 16.073425	5) 8.857086	5) 8.563557
30	**1) 6.850661**	**1) 8.856477**	**1) 8.537169**	**1) 8.455409**	**1) 7.681360**	**1) 8.691598**	**1) 9.046746**	**1) 8.382458**
	2) 6.535299	2) 7.671576	2) 8.523108	2) 8.432194	2) 7.445574	2) 7.876268	2) 9.033444	2) 8.365783
	3) 9.266471	3) 11.326148	3) 8.544252	3) 8.469553	3) 9.891122	3) 10.303482	3) 9.052298	3) 8.392537
	4) 5.960517	4) 7.240049	4) 6.378289	4) 6.816072	4) 6.685033	4) 6.760183	4) 7.507708	4) 7.293618
	5) 27.517009	5) 21.060554	5) 15.580074	5) 14.480377	5) 17.329466	5) 16.257088	5) 13.897944	5) 11.890573

Tabela 1:

Legenda:
1- Aptidão média da população
2- Aptidão média do melhor individuo
3- Aptidão média do pior individuo
4- A melhor
5- A pior

Como se pode observar, na Tabela 1 estão os resultados dos testes realizados com os parâmetros acima enumerados. Pretendemos aqui observar, para cada tipo de representação, padrões que nos levem a poder concluir acerca das melhores condições para resolver o problema descrito, obtendo o melhor resultado. Na Tabela 2 encontramos a mesma experiência, mas aumentando o número de gerações e de indivíduos. Desse modo, pretendemos verificar se as observações e proposições se mantêm válidas ou não.

De realçar que a experimentação se trata de isso mesmo: experimentar. Cada experiência pode resultar em diferentes resultados, mesmo para os mesmos parâmetros, devido a inúmeros factores. Com o aumento do número de indivíduos e de gerações, pretende-se também atenuar o efeito que possíveis “más” gerações possam ter na experiência.

Observação 1: Quanto mais pontos existem por indivíduo, maiores são os valores de fitness médios registados;

Proposição 1: Como os pontos a "trabalhar" são mais, para o mesmo número de repetições, era expectável que assim fosse.

Observação 2: Mutações normais combinadas com o método de selecção por Torneio, para ambos os tipos de Representação, geram *melhores* resultados do que com o método de selecção por Roleta no entanto:

Observação 3: Mutações gaussianas combinadas com o método de selecção por Torneio, para ambos os tipos de Representação, geram *piores* resultados do que com o método de selecção por Roleta;

Proposição 2: Mutações normais podem introduzir grandes alterações no fitness total de um indivíduo, uma vez que os novos valores gerados são aleatórios para o intervalo em causa. Perturbações gaussianas alteram apenas um pouco o indivíduo, com a modificação a ser feita com base no valor anterior da coordenada em causa e introduzindo apenas uma pequena alteração no valor (cerca de [-3,3] no nosso caso). Assim sendo, aparenta que a Roleta, que selecciona com base nas probabilidades de cada indivíduo, tende a escolher os melhores indivíduos que vão sendo gerados promovendo uma boa evolução da população, no caso da Mutação ser Gaussiana. Por sua vez o Torneio, como escolhe ao acaso, acabará eventualmente por não escolher os bons indivíduos introduzidos por recombinações/mutações favoráveis, ou escolhe-os menos vezes. Uma possível solução para melhorar aqui o Torneio seria aumentar o número de indivíduos escolhidos (Número de Torneio) antes de ser escolhido o melhor. Para Mutações normais, ocorre o inverso. O método da Roleta tende a escolher sempre o melhor, uma vez que as discrepâncias são maiores entre indivíduos, o que leva a uma estagnação na evolução, conduzindo a curva para máximos locais e não promove a diversidade. Já o Torneio introduz mais diversidade, porque escolhe ao acaso, o que leva o algoritmo a explorar outros espaços de resultados e a obter melhores valores.

Observação 4: Observaram-se melhores resultados com o uso de Partições Iguais em oposição a Partições Aleatórias, com a excepção de um caso, em que o número de Pontos era 20.

Proposição 3: Esta situação é bastante interessante de ser analisada. Obtiveram-se sempre melhores resultados para o tipo de Representação 1 excepto quando o número de pontos é 20, onde a Representação do tipo 2 é melhor em todos os casos testados. Daqui se pode constatar claramente a influência que diferentes combinações de parâmetros têm no resultado final, e que conclusões tiradas para determinados valores podem não se verificar para outros. Através da observação da Tabela 2, iremos tentar ver se existe algum tipo de convergência de valores, ou se as diferenças se irão manter.

É relevante acrescentar que o número de pontos de Crossover, taxa de mutação, tamanho do Torneio e probabilidade de Crossover, foram escolhidos com base em alguns testes também realizados e os quais irão ser apresentados mais à frente. Como também já foi referido, não existem valores óptimos universais. Bons resultados para determinadas condições, não significa melhores resultados noutras.

Nº Gerações-350 Nº de Individuos-600 Tamanho Torneio/Roleta-5 Prob. Crossover-50% Pontos Crossover-1 Prob. Mutação-10% Elitismo-20%

Número de Pontos	Partições Iguais				Partições Aleatórias			
	Mutação Normal		Mutação Gaussiana		Mutação Normal		Mutação Gaussiana	
	Torneio	Roleta	Torneio	Roleta	Torneio	Roleta	Torneio	Roleta
10	**1) 5.694337**	**1) 7.002664**	**1) 5.261904**	**1) 5.279010**	**1) 5.679155**	**1) 8.032447**	**1) 5.267445**	**1) 5.227118**
	2) 5.261693	2) 5.279951	2) 5.260576	2) 5.260583	2) 5.226800	2) 5.292462	2) 5.267406	2) 5.227086
	3) 9.178973	3) 9.056940	3) 5.278463	3) 5.305665	3) 9.166849	3) 10.671725	3) 5.268175	3) 5.227224
	4) 5.260606	4) 5.260795	4) 5.260571	4) 5.260571	4) 5.226073	4) 5.241756	4) 5.267081	4) 5.225754
	5) 12.131658	5) 14.425071	5) 5.322905	5) 5.373673	5) 12.412144	5) 13.976494	5) 5.273967	5) 5.242226
20	**1) 5.612305**	**1) 8.280280**	**1) 5.259420**	**1) 5.328533**	**1) 6.710186**	**1) 12.848447**	**1) 5.402437**	**1) 5.886743**
	2) 5.519927	2) 5.524032	2) 5.259081	2) 5.327755	2) 6.415323	2) 6.181995	2) 5.402133	2) 5.884147
	3) 7.164041	3) 11.223203	3) 5.262400	3) 5.329424	3) 9.679317	3) 18.047039	3) 5.406620	3) 5.890434
	4) 5.270022	4) 5.360171	4) 5.229163	4) 5.229514	4) 6.407229	4) 5.925085	4) 5.373717	4) 5.826677
	5) 8.928348	5) 24.029905	5) 5.719444	5) 6.159814	5) 17.990902	5) 25.575709	5) 5.479221	5) 6.154202
30	**1) 6.061773**	**1) 7.642962**	**1) 5.355387**	**1) 5.980736**	**1) 7.259872**	**1) 16.084275**	**1) 6.825824**	**1) 7.119982**
	2) 5.881446	2) 6.330226	2) 5.354859	2) 5.978864	2) 7.108366	2) 7.119080	2) 6.825699	2) 7.119919
	3) 8.339437	3) 9.457566	3) 5.359548	3) 5.981865	3) 9.454392	3) 23.198072	3) 6.827233	3) 7.120224
	4) 5.629688	4) 6.084331	4) 5.227722	4) 5.400943	4) 7.074635	4) 7.035247	4) 6.820084	4) 7.108136
	5) 12.654731	5) 30.908708	5) 6.045680	5) 7.368375	5) 12.813335	5) 32.329592	5) 6.925877	5) 7.138261

Tabela 2:

Observação 1: Quanto mais pontos existem por indivíduo, maiores são os valores de fitness médios registados;

Proposição 1: Aumentando o tamanho da população e o número de gerações, pode verificar-se que a tendência se mantém. No entanto, a diferença entre os resultados obtidos diminui significativamente, o que não nos permite concluir que a observação 1 seja verdadeira. Se aumentássemos ainda mais estes parâmetros, esta diferença podia ser ainda mais atenuada e, talvez, até se poderiam vir a obter melhores resultados com mais pontos. É de resto este o comportamento esperado. Quantos mais pontos se tem, mais precisa irá tender a ser a curva. Este comportamento é aliás observado no caso da Mutação Gaussiana/Torneio/Partições Iguais. Apesar dos valores serem bastante próximos, quer para 10, 20 e 30 pontos, existe uma melhoria com 20 pontos em relação aos 10, o que ilustra o que acabámos de dizer.

Observação 2: Mutações normais combinadas com o método de selecção por Torneio, para ambos os tipos de Representação, geram *melhores* resultados do que com o método de selecção por Roleta no entanto:

Observação 3: Mutações gaussianas combinadas com o método de selecção por Torneio, para ambos os tipos de Representação, geram *piores* resultados do que com o método de selecção por Roleta;

Proposição 2: Para um grande número de indivíduos também se verifica o que foi constatado na *Observação 2.* Não obstante, a *Observação 3* já não se verifica. Como a população é muito grande, o método da Roleta aparenta deixar de ter vantagens sobre o Torneio e é até neste último que se verificam os melhores valores. Apesar disso ambos são bastante semelhantes, o que indica que o valor óptimo se deve situar naquela gama. Observa-se pois uma convergência de valores, independentemente do método de selecção usado, no caso de utilizarmos perturbações Gaussianas. No entanto, tais observações permitem-nos concluir que, caso optemos pelo tipo de Mutação Normal, devemos utilizar o método do Torneio, pois iremos obter melhores resultados.

Observação 4: Melhorias de resultados, na generalidade, com o uso de Partições Iguais em oposição a Partições Aleatórias.

Proposição 3**:** Aumentando o número de indivíduos e gerações fez com que os valores obtidos fossem bastante semelhantes entre representações. À excepção da combinação Mutação Normal/Roleta, para 10 e 20 pontos os resultados observados encontram-se na mesma gama, com uma ligeira vantagem para a Representação 1 (Partições iguais). Com 30 pontos, os resultados observados com partições iguais são significativamente melhores. Justificação: Como na Representação 2 ambas as coordenadas são aleatórias (na fase de criação dos indivíduos), seriam necessárias mais gerações para fazer evoluir a curva até um valor próximo do óptimo. Usando a Representação 1, onde só mutámos os Y's, com o mesmo número de gerações a população sofre uma melhor e maior evolução, o que se traduz nos resultados obtidos. Assim sendo, para partições aleatórias seriam necessárias mais gerações e um maior número de indivíduos para que se consiga obter uma boa curva, quando comparada com a Representação Normal e para um valor de 30 pontos. Os outros, como referimos, geram resultados bastante próximos

Da observação das tabelas, verificamos também que os resultados são tanto melhores, quanto menor for a diferença da média do melhor e pior indivíduos, sendo nestes casos que o desvio padrão é também menor.

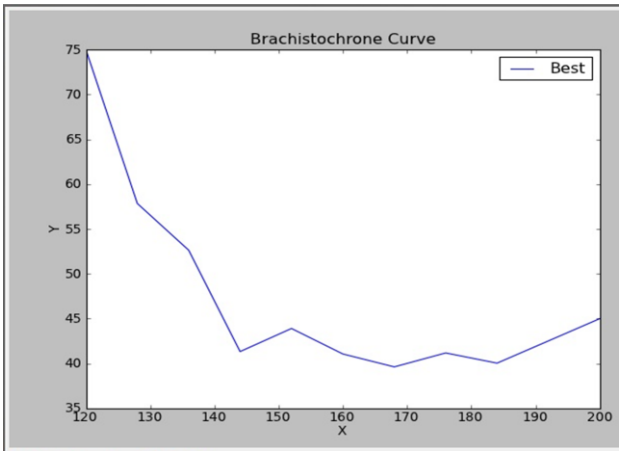
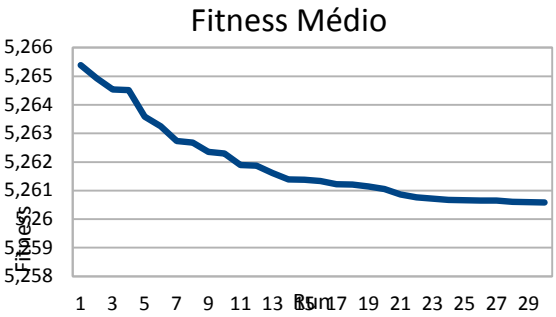
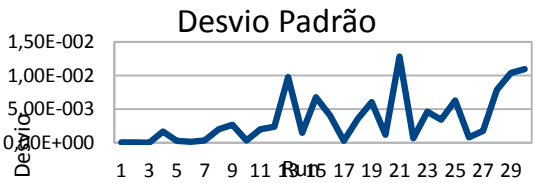


Ilustração 1: Representação 1 /Torneio/Gaussiana/10 pontos





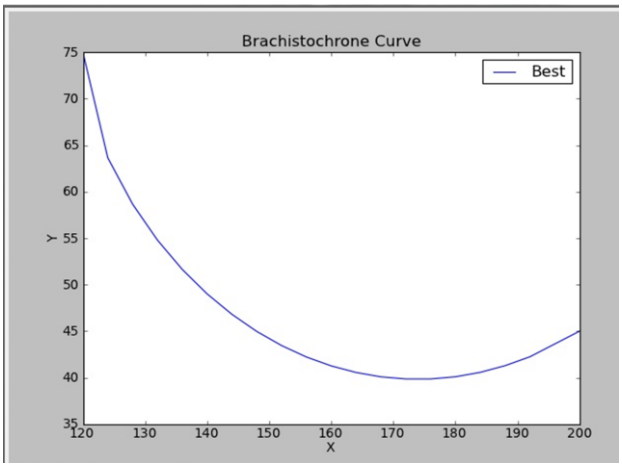
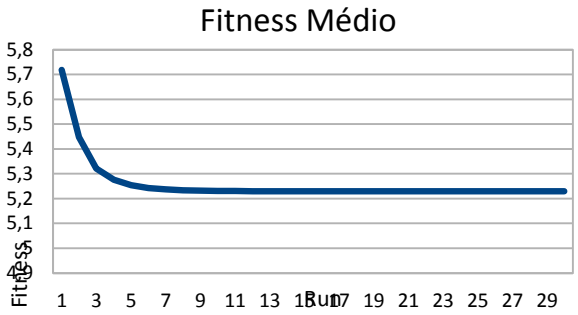
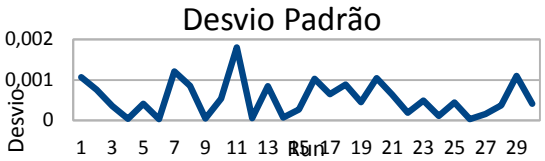


Ilustração 2: Representação 1 /Torneio/Gaussiana/20 pontos





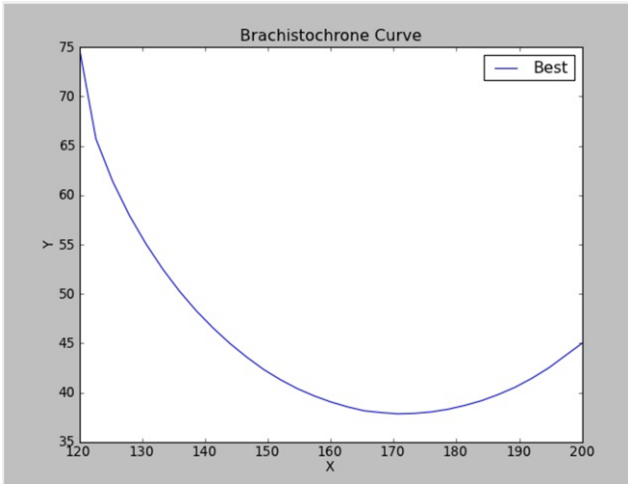
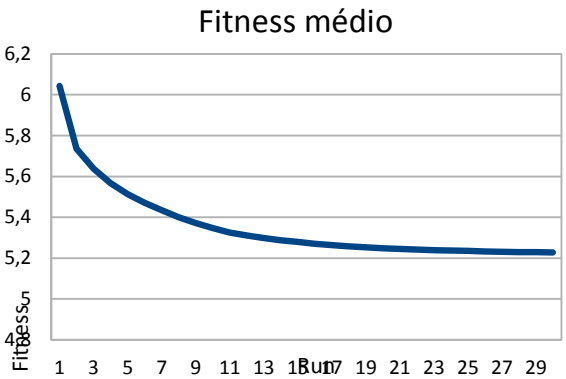
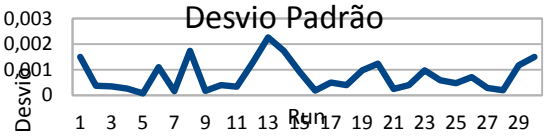


Ilustração 3: Representação 1 /Torneio/Gaussiana/30 pontos





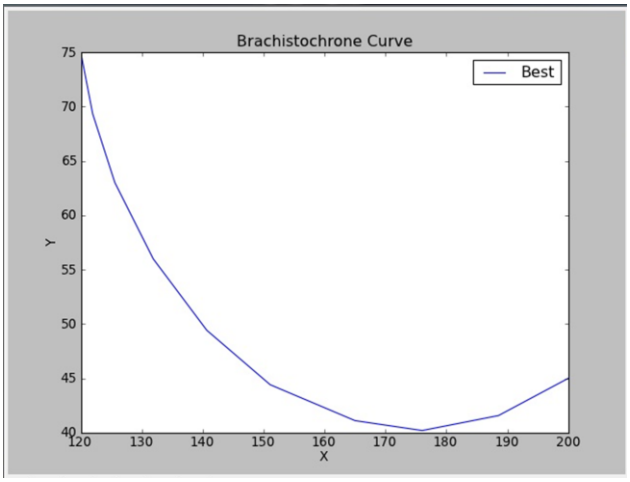
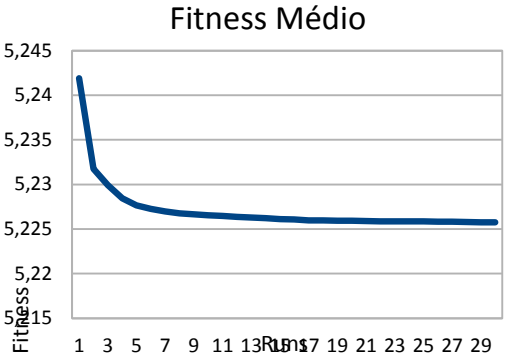
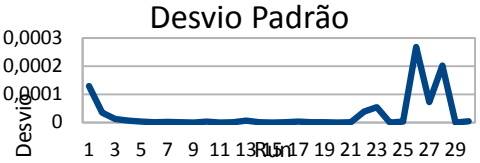


Ilustração 4: Representação 2 /Roleta/Gaussiana/10 pontos





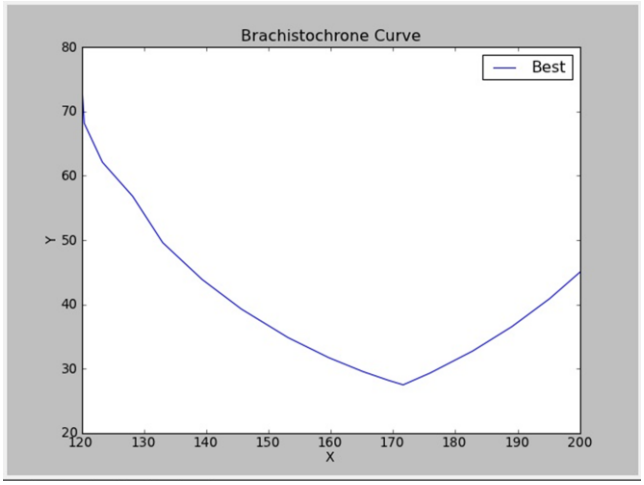
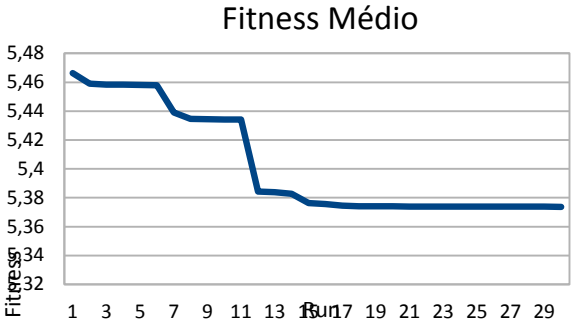
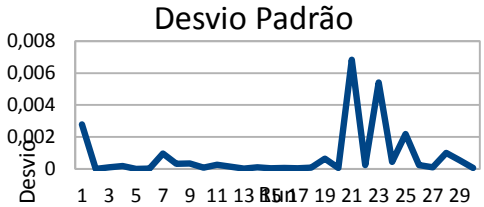


Ilustração 5: Representação 2 /Torneio/Gaussiana/20 pontos





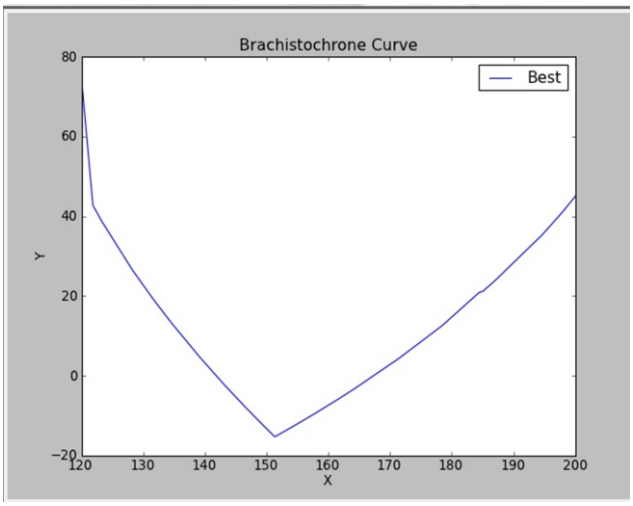
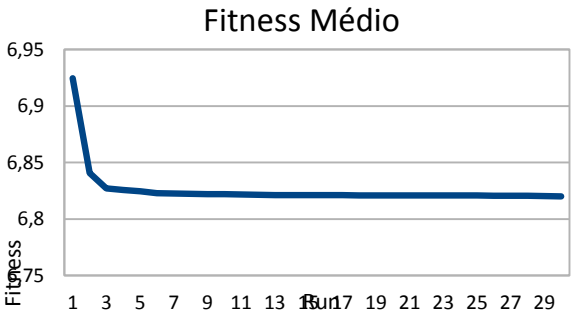
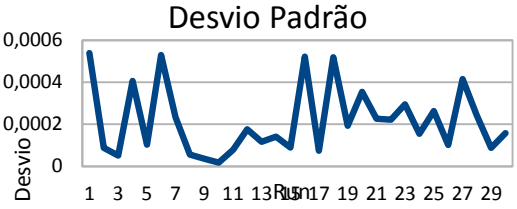


Ilustração 6: Representação 2 /Torneio/Gaussiana/30 pontos





Durante a experimentação guardamos os valores de desvio padrão e da média de Fitness por Run, isto é, para cada repetição das 30 que realizámos por experiência. Iremos aqui apresentar alguns dos gráficos das melhores curvas obtidas por Número de Pontos/Tipo de Representação e da evolução do respectivo Fitness e desvio-padrão. É passível de ser observado nos gráficos a convergência já mencionada para um dado valor, e a sua dificuldade em evoluir mais a partir daí.

Na situação em que utilizámos 10 pontos, o desvio padrão foi superior ao dos outros casos, significando isso que existia uma maior divergência entre os diferentes indivíduos. Os valores nos 2 últimos casos são bastante inferiores. Parece então que quanto menor for a discrepância entre indivíduos, isto é, quanto menor o desvio padrão, melhor tende a ser a representação final e a evolução do fitness entre Runs da mesma experiência. Era expectável que assim fosse.

O facto da curva da primeira Ilustração não ser tão boa quanto as outras, vem também apoiar a ideia que já tinha sido considerada expectável: quanto mais pontos, melhor a representação da curva final.

Também podemos ver, pela análise dos gráficos de fitness, que quanto mais estes demoram a estabilizar num valor, desenhando uma curva que parece logarítmica, maior o desvio padrão. Os dados parecem demonstrar então que quanto menor o desvio padrão entre indivíduos, melhor tende a ser a curva final, e nota-se uma evolução mais rápida ao nível do Fitness da população.

No caso em que simulamos 30 pontos, o fitness é pior porque havia mais pontos para tratar, no entanto com a Representação Normal a curva desenhada assemelha-se mais ao esperado. Tal ocorreu, como já foi referido ao longo do relatório, muito provavelmente pelo rácio entre quantidade de pontos/gerações/nº de indivíduos. De maneira a gerar um gráfico com melhor fitness e com mais pontos, será então necessário aumentar o número de gerações e/ou indivíduos, para que obtenhamos resultados dentro da mesma gama de valores.

Outros Parâmetros Importantes

Probabilidade de CrossOver

Nº Gerações-40 Nº de Individuos-600 Tipo Mutação-Gaussiana Torneio Tamanho Torneio/Roleta-5 Pontos Crossover-1
Prob. Mutação-10% Elitismo-20%

Probabilidade de Crossover	Partições Iguais			Partições Aleatórias		
	10 Pontos	20 Pontos	30 Pontos	10 Pontos	20 Pontos	30 Pontos
20%	5.264828	5.831937	6.833001	5.437042	6.968674	7.706695
50%	5.261525	5.550527	6.103128	5.282211	6.369887	7.590522
80%	5.264749	5.778847	6.429249	5.383145	6.343079	7.743167

O objectivo deste teste foi descobrir o valor ideal para a percentagem de Crossover. Para tal, testámos com um tamanho de indivíduos de 600 que está representado na tabela acima. Foi também feito o teste com uma população de 80, no entanto omitimos a tabela deste último visto que os valores eram idênticos aos obtidos no anterior. Ao analisarmos a tabela, verificamos que o valor da probabilidade de Crossover é melhor em valores que rondam o 50%. Sabendo que quanto mais baixo este valor menor seria a probabilidade de cruzamentos entre indivíduos, e vice-versa, concluiu-se que para este caso este valor intermédio seria o que mais se justificaria utilizar. Como tal, passou a ser o valor usado em todos os testes realizados posteriormente.

O número de pontos de Crossover usado foi sempre 1, porque foi aquele com que obtivemos melhores resultados também.

Percentagem de Elites

Nº Gerações-40 Nº de Individuos-600 Tipo Mutação-Gaussiana Torneio Prob. Crossover-50% Pontos Crossover-1
Prob. Mutação-10% Tamanho Torneio-5

Percentagem de Elites	Partições Iguais			Partições Aleatórias		
	10 Pontos	20 Pontos	30 Pontos	10 Pontos	20 Pontos	30 Pontos
0%	5.263595	6.166882	6.444462	5.280253	6.617176	6.961125
20%	5.261433	6.256123	6.410012	5.270045	6.117740	6.403153
50%	5.263332	5.580373	6.614411	5.275683	5.883810	7.596998
100%	14.707795	27.590929	40.370802	14.734302	27.914621	40.641110

No que diz respeito ao valor ideal da percentagem de elitismo, ao analisar a tabela, concluímos que ronda os 20%. No entanto, percentagens até 50% produzem também resultados bastante satisfatórios. Qualquer outro valor acima do atrás indicado é desaconselhado visto que a quantidade de indivíduos que são pre-

servados de uma geração para a outra é bastante alta, levando a uma fraca rotatividade destes e consequentemente a uma permanência de bastantes indivíduos considerados fracos durante várias gerações. De realçar também que esta percentagem não deve ser nula pois impede que indivíduos considerados bons de sobreviver ao longo das gerações.

Tamanho do Torneio

Nº Gerações-40 Nº de Individuos-600 Tipo Mutação-Gaussiana Torneio Prob. Crossover-50% Pontos Crossover-1
Prob. Mutação-10% Elitismo-20%

Tamanho Torneio	Partições Iguais			Partições Aleatórias		
	10 Pontos	20 Pontos	30 Pontos	10 Pontos	20 Pontos	30 Pontos
1	5.282740	6.208755	6.900945	5.267277	6.166654	6.418729
3	5.262321	5.536184	6.665391	5.274143	6.574738	6.282089
5	5.261075	5.457343	6.630266	5.248324	5.936736	7.152460
7	5.271048	5.778062	6.808637	5.242962	5.797586	7.353520
9	5.263628	5.931357	6.693606	5.307354	6.588226	7.321439

Em relação ao tamanho do torneio, o valor por nós escolhido foi o de 5 indivíduos, apesar de todos os outros acima enumerados poderem também ser aceites pelos bons resultados que obtivemos deles. Cremos que este valor é o mais admissível pois permite uma escolha adequada dos melhores 'pais' para gerar novos indivíduos através do crossover. O facto de não ser um valor muito alto evita que sejam sempre escolhidos os mesmos indivíduos e ao mesmo tempo, por também não ser muito baixo, evita a questão de ser logo seleccionado o primeiro individuo. Como se pode observar, para 10 pontos a diferença que existe não é muita. Seria de esperar que um valor muito alto prejudicasse neste caso, pois iria tender para a escolha dos mesmos indivíduos ao longo do tempo. No caso da Representação Aleatório tal facto é mais visível, devido a tudo o que foi exposto na análise das tabelas em relação a esta questão. Para o caso dos 30 pontos nesta Representação, ainda existe uma diferença considerável. Não obstante, como para as restantes combinações, o 5 aparenta ser ligeiramente melhor, daí termos optado por esse. Para a Representação Normal, a diferença registada é menor, sendo também o 5 que obtém melhores resultados.

Percentagem de Mutação

Nº Gerações-40 Nº de Pontos-20 Tipo Mutação-Gaussiana Torneio Prob. Crossover-50% Pontos Crossover-1
Tamanho Torneio-5 Percentagem de Elites-20%

Probabilidade de Mutação	Partições Iguais				Partições Aleatórias			
	Mutação Normal		Mutação Gaussiana		Mutação Normal		Mutação Gaussiana	
	80 Indivíduos	600 Indivíduos	80 Indivíduos	600 Indivíduos	80 Indivíduos	600 Indivíduos	80 Indivíduos	600 Indivíduos
10%	6.498905	6.571330	6.729091	5.633875	7.636428	7.114648	7.694377	6.233102
50%	14.161360	12.890829	5.967000	5.516692	14.591231	13.833640	7.107102	5.980578
100%	20.094006	19.499468	5.703198	5.468643	19.653657	19.533923	6.906929	6.203473

No que trata da escolha da probabilidade de mutação obtivemos valores bastante curiosos durante os testes. Se pretendermos obter o melhor valor no geral facilmente concluímos que este ronda os 10%, no entanto, olhando mais pormenorizadamente, deparámo-nos com valores antagónicos entre o tipo de mutação normal e gaussiana. À medida que aumentávamos a percentagem de mutação, utilizando uma mutação normal, a aptidão geral era cada vez menos satisfatória contudo, utilizando uma mutação gaussiana esta assumiu valores bastante bons e directamente proporcionais à percentagem da mesma. Justificação: Como já foi referido acima, uma Mutação Normal introduz uma grande perturbação no gene modificado. Ora, como esta mutação é bastante agressiva para o indivíduo, taxas mais altas fazem com que seja impossível para o nosso algoritmo de estabilizar em torno de um valor, uma vez que o largo número de mutações altera constantemente os nossos indivíduos de uma maneira brusca. Caso se usem Perturbações Gaussianas, o mesmo não se verifica porque é benéfico que estas mutações ocorram mais vezes, dado que apenas introduzem uma pequena alteração no indivíduo. Assim sendo, muitas mutações irão funcionar como um calibrador gradual, proporcionando assim melhores resultados.

Esforço dos elementos:

Pedro Geadas - cerca de 40 horas em casa + PL's

Diogo Pinheiro – cerca de 20 horas em casa + PL's