P.PORTO

ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO

Licenciatura em Engenharia Informática

Inteligência Artificial

Trabalho Prático

Diogo Magalhães 8120336
Cristiana Monteiro 8150489

Ano letivo 2018/2019

Índice

1 – Introdução

1.1 Propósito do projeto

O grupo teve consenso comum na medida em que consideramos valioso o desafio de desenvolver este projeto lançado pelo Sr. Professor Davide Carneiro, com intuito de consolidar a matéria desenvolvida no âmbito da unidade curricular de Inteligência Artificial. Atravez do exercício de superar as adversidades inerentes ao processo de desenvolver um algoritmo de natureza evolutiva e a superar, paradigmas do mesmo. Em suma, podemos afirmar com grande grau de certeza esta estratégia será uma bom veiculo para assimilar os conceitos básicos dos métodos baseados em computação evolutiva

1.2 Algoritmos Genéticos

Os métodos baseados em computação evolutiva pertencem a uma classe de algoritmos de busca e otimização inspirados no desenvolvimento orgânico da natureza na evolução das espécies como defende Darwin. Métodos evolutivos são meramente uma alternativa com a finalidade de transpor os obstáculos apresentados por métodos tradicionais, porem são característicos por não garantir uma solução exata.

1.3 Ferramentas utilizadas

Para o desenvolvimento do trabalho pratico de Inteligência Artificial os nossos portáteis munidos de Linux com os seguintes softwares:

-Netbeans para a composição do código necessário para implementar os métodos evolutivos do jogo SuperMario clássico.

Usufruímos também da maquina virtual cedida pelo Sr. Professor onde receberia, através de pedidos tipo “post”

Por ultimo, recorremos ao software Postman para testar inicialmente com um array estático de comandos elegidos mais ou menos de forma arbitraria.

1.4 Recursos Fornecidos

Primeiramente analisamos o Enunciado do trabalho pratico enunciado no Moodle: https://moodle.estg.ipp.pt pelo Sr. Professor Davide Carneiro munido de um enunciado de como instalar e testar o projeto com recurso a Postman juntamente com o código necessário para submeter para leaderboard(site criado e hospedado pelo sr professor para submissão dos nossos resultados do projeto desenvolvido).

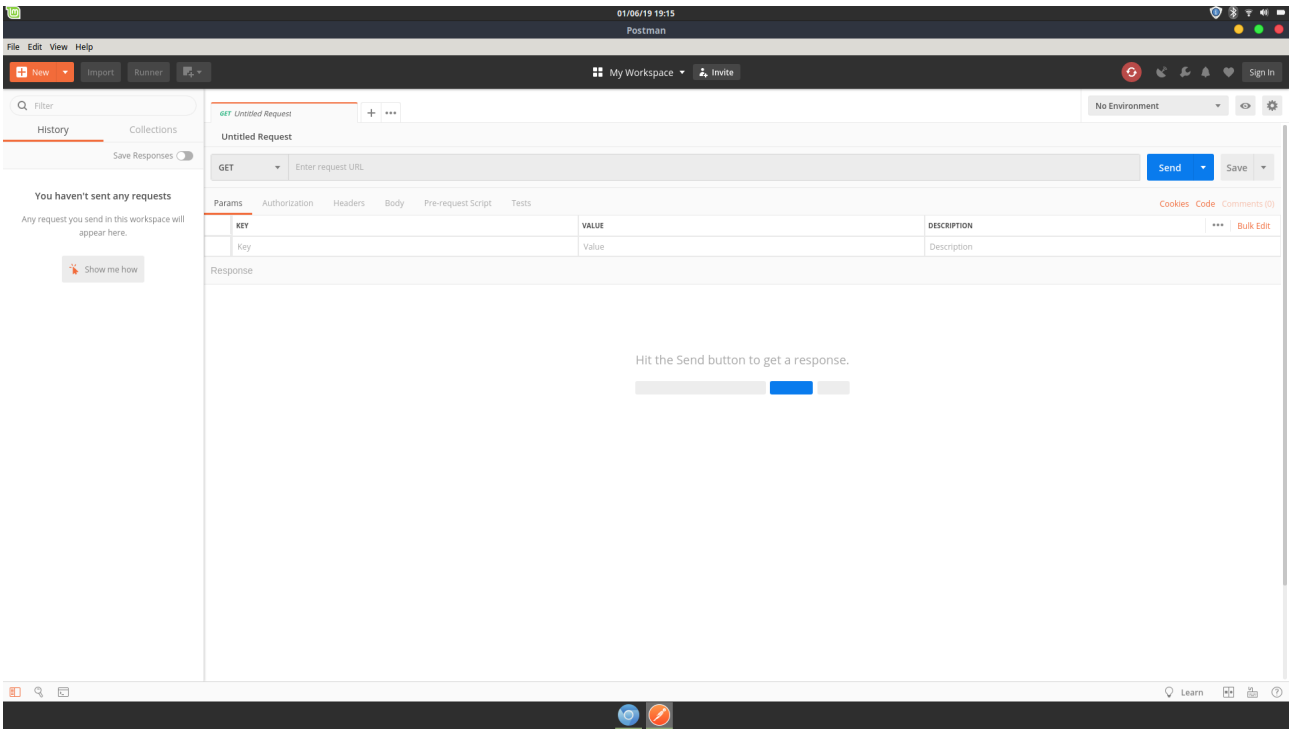
Fora também cedido uma pasta de Recursos “TP.IA.2018.Recursos V3 (com a pasta jeneticNetbeans, com interfaces já totalmente preparadas com a estrutura total do projeto o que simplifica e faz com que todos os projetos tenham um esqueleto uniforme.

É nos também cedido uma pasta com documentação “documentação V2”, marioengine V3 com marioengine.py/mario-server.py.

Adicionalmente é nos facultado uma Maquina Virtual criada pelo Sr. professor com tudo para iniciar o projeto descomplicadamente e inclusivamente fora cedido um ficheiro tutorial de “Instruções instalação para Linux”.

Finalmente é nos dedicado um local no Moodle para submeter o projeto para avaliação.

Vizualizacao do software Postman para primeiro teste com array estatico



2- Definição de Termos Genéticos

2.1- Geração

Resultado do ciclo apresentado na imagem anterior.

2.2- População

Conjunto de cromossomas que compõe uma geração.

2.3- Cromossoma

Conjunto de genes (neste caso comandos), que formam apenas um cromossoma.

2.4- Função de fitness

Função que permite avaliar quais os melhores cromossomas a cada geração;

2.5- Seleção

Processo de escolha, entre os melhores cromossomas, para participarem no processo de reprodução.

2.6- Reprodução

Processo pelo qual os cromossomas com melhores valores para a função de fitness passam para através da aplicação dos operadores genéticos consigam obter nova geração.

2.7- Critérios de paragem

Carateriza-se pela obtenção de uma solução pré-definida, ou seja, mediante os três critérios de paragem desenvolvidos neste trabalho: nível máximo de iterações que equivale ao número máximo de gerações que o algoritmo poderá ter, duração máxima em minutos que o algoritmo pode demorar ou caso haja pouca evolução. Mediante o valor dado pelo utilizador à variável “deltaMin”, a partir da 6º geração caso não haja diferenças significativas entre a média da geração atual e a média das últimas 5 gerações, então o algoritmo genético já encontrou uma solução ótima.

3 – De que forma é que o Algoritmo Genético pode ser configurado

O algoritmo evolutivo desenvolvido pode ser configurado nos seguintes parâmetros:

Nome	Parâmetro	Parâmetros aceites
Tamanho da população	populationSize;	Int positive
Aleatoridade de tamanho	randomSize	Int positive
Máximo de Iterações	maxIterations	Int positive
Tamanho da mutação	mutationSize;	Int positive
Taxa de evolução	deltaMin	Double positive
Fator de mutação	mutationFactor	Int positive
Numero de melhores casos	numBestCases	Int positive
Tamanho do cromossoma	chromosomeSize	Int positive
Versão	versão	Int positive
Visualização do jogo	render	“true” or ”false”
Modo do jogo	modoDeJogo	“forever” or ”level”
IP da maquina virtual	ip	Ipv4 ip virtual machine

4- Qual a estratégia de seleção utilizada

A estratégia passou por criar inicialmente uma população de 10 com cromossomas. O valor elegido para melhor performance nos teste foi de cromossomas de 500 genes divididos por grupos de 50 genes iguais. Para este efeito utilizamos uma variável “buttomSize” que define quantos genes seguidos terão o mesmo valor. Portanto o cromossoma será dividido pelo seu tamanho (neste configuração ótima serão 500) , pelo tamanho do botão pressionado “buttonSize” pelo que podemos deduzir que seriam cromossomas de 100 botões (500/50). Após esta parte desenvolvida continuamos por desenvolver a parte da hereditariedade. Para isso, geramos a variável de “bestNumCases” onde configuramos o numero de cromossomas que serão mantidos na geração seguinte. Os cromossomas serão considerados melhores tendo em conta o valor do seu “fitness” e pelos algoritmos definidos pelos user (como por exemplo: quantos cromossomas serão mutados, cruzados ou aleatório). Importante relembrar que apenas sofrerão estas transformações os cromossomas elegidos com melhores.

Próximo passo será escolher de que forma iremos mutar. Através do resultado do metodo “getReasonFinish” onde terá 3 possíveis resultados (“win”, “death”, “no_more_commands”); Tendo em conta o resultados, o cromossoma sofrera uma mutação diferente.

- Caso resultado seja “win” apenas fazemos print de uma mensagem a congratular o user.
- Caso resultado seja “death” a mutação do cromossoma será implementada pelo metodo “mutateIfDeath” onde este ira ter como ponto de partida onde o avatar “Mário” morreu, e a partir daí ira gerar valor aleatórios de comandos para ele tentar prosseguir com sucesso mais um pouco no jogo.

-Caso resultado seja “no_more_commands” ira ser implementado o método “addRandomChromossomes” que fará com que sejam incrementados mais 50 comandos gerados aleatoriamente ao cromossoma.

Importante frisar que existe também em comentário no algoritmo desenvolvido o protótipo de solução caso o avatar morra devido a terminar o tempo de jogo. A solução seria idêntica ao método “addRandomChromossomes” mas em vez de incrementar 50 comandos, seriam incrementados 500. A razão pela qual colocamos em comentário, é devido a otimização do algoritmo. Uma vez que esta comparação iria ser executada em todas as interacoes do programa em todos os cromossomas de todas as gerações e o resultado seria sempre irrelevante por 2 razoes. Primeira, com o limite de comandos que utilizamos (“3” e ”4”) que correspondem a “correr rápido” e a “saltar”, respetivamente o “Mário” nunca iria perder por falta de tempo mas sim por falta de comandos ou morte. Segunda razão e que neste modo de jogo “Forever” como é avaliado como melhor a distancia que ele percorre e não o “score”, podemos deduzir que o “Mário” ira tentar progredir o mais rápido possível em termos de distancia, e uma vez que o tempo permite grande margem de erro em termos de gestão de tempo, será valido deduzir que este caso nunca se ira verificar.

5- De que forma é implementado cada um dos operadores genéticos

5.1- Hereditariedade

Tendo em conta os casos elegidos como melhores pelo algoritmo do user, verificamos que no final de cada geração iremos invocar o métodos “selection” cuja função será de ordenar os mais aptos e devolver um array da dimensão da variável “bestNumCases”.

Depois verificamos que teremos um array “historic” que terá a função de guardar os cromossomas elegidos de todas as gerações geradas. Seguidamente adicionara a lista da futura geração com o método “heredity”. Este método será o ponto de partida para a próxima geração começar a enviar post requests ao servidor.

5.2- Mutação

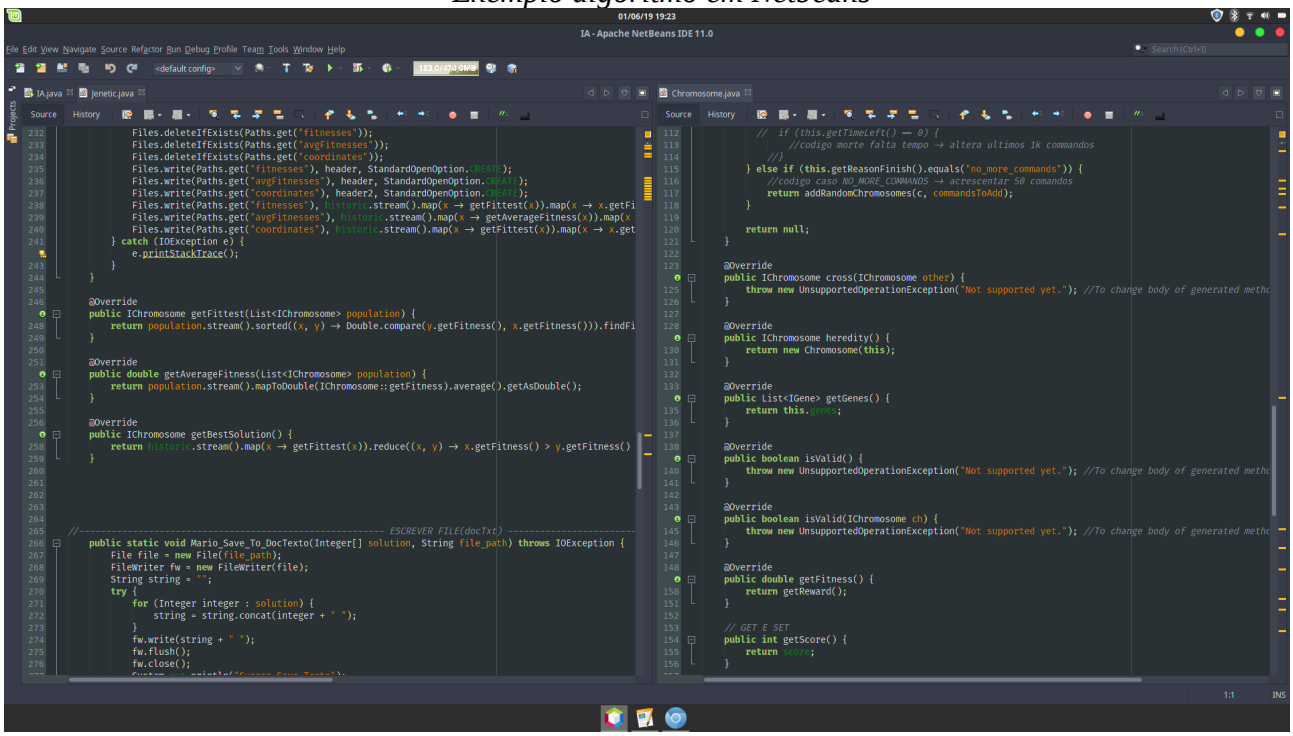
Este método é responsável por garantir a mutação do cromossoma tendo em conta a razão pela qual perdeu. Desta forma otimizamos a evolução e o progresso no jogo.

Começamos por criar uma copia dos valores do cromossoma a mutar num construtor do mesmo vazio de forma a evitar trabalhar por referencia. Seguidamente utilizamos o método “getReasonFinish” e tendo em conta o seu retorno, implementamos uma maneira forma de mutação a esse cromossoma.

5.3- Random

Com este método criamos cromossomas de dimensão definidas pelo usar, que seguidamente serão preenchidas com valores aleatórios balizados num range valido de inteiros(que correspondem as acções do “Mário”)

Exemplo algoritmo em Netbeans



6- Modo de jogo abordado pelo grupo “PandaPanda” foi o modo “Forever”.

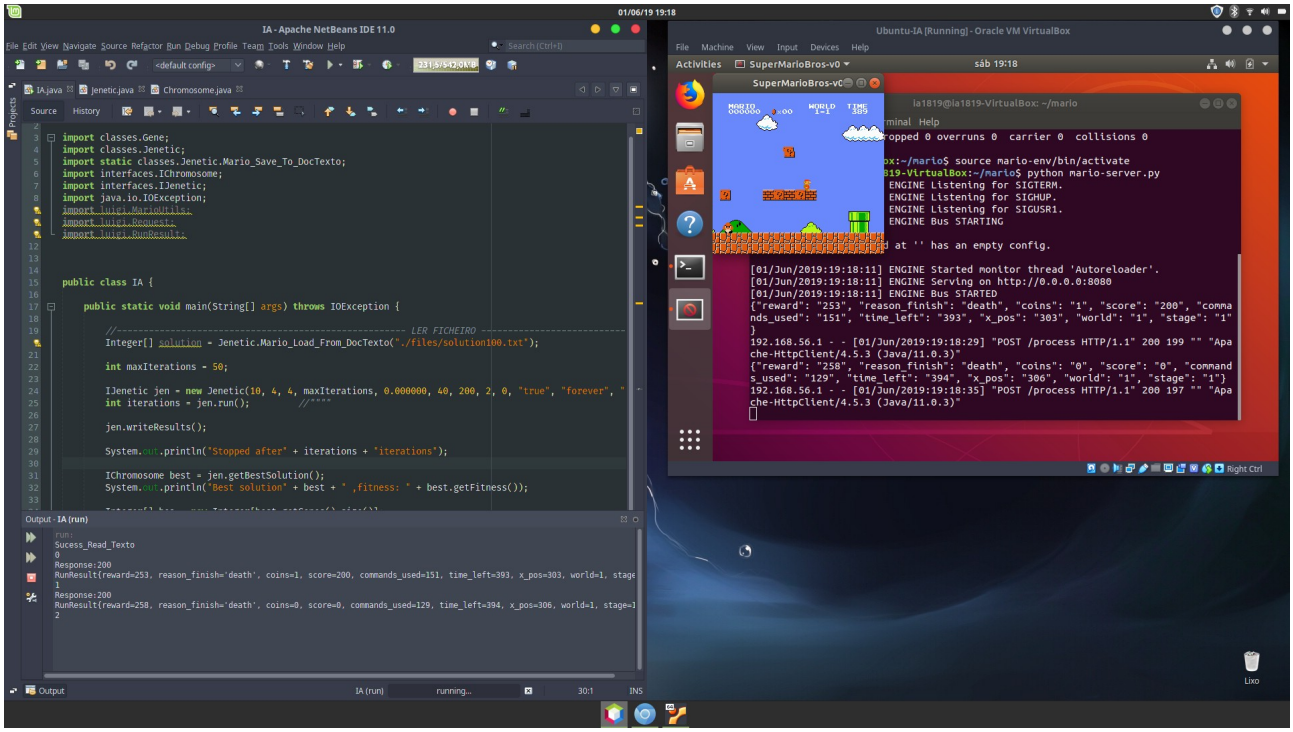
6.1- Quais as estratégias implementadas para lidar com os problemas encontrados

Primeiramente inicializamos a população com o valor 10 e geramos valor aleatórios validos para os mesmos.

Seguidamente escolhemos gerar inicialmente 10 gerações, depois 50, e finalmente 250 para analisar a performance e evolução da progresso no jogo do “Mário”. Com isto verificamos que a evolução tem algumas fases de estagnação de evolução em algumas gerações, o que e típico de algoritmos de natureza evolutiva. Como ficamos satisfeitos em termos gerais com a evolução, decidimos que a configuração se adequava aos nossos padrões. Verificamos também que quantos maior o numero da geração, maior tempo era consumido em termos de processamento, isto e, obter resultados de, por exemplo, geracao 3 é mais rápido a gerar os dados do que a geração 30.

Finalmente começamos por definir que os cromossomas elegidos são os que têem o valor mais alto no parâmetro “reward” pois desta forma conseguimos satisfazer o resultado do modo “Forever”.

Demonstração *de algoritmo em running time com o servidor*



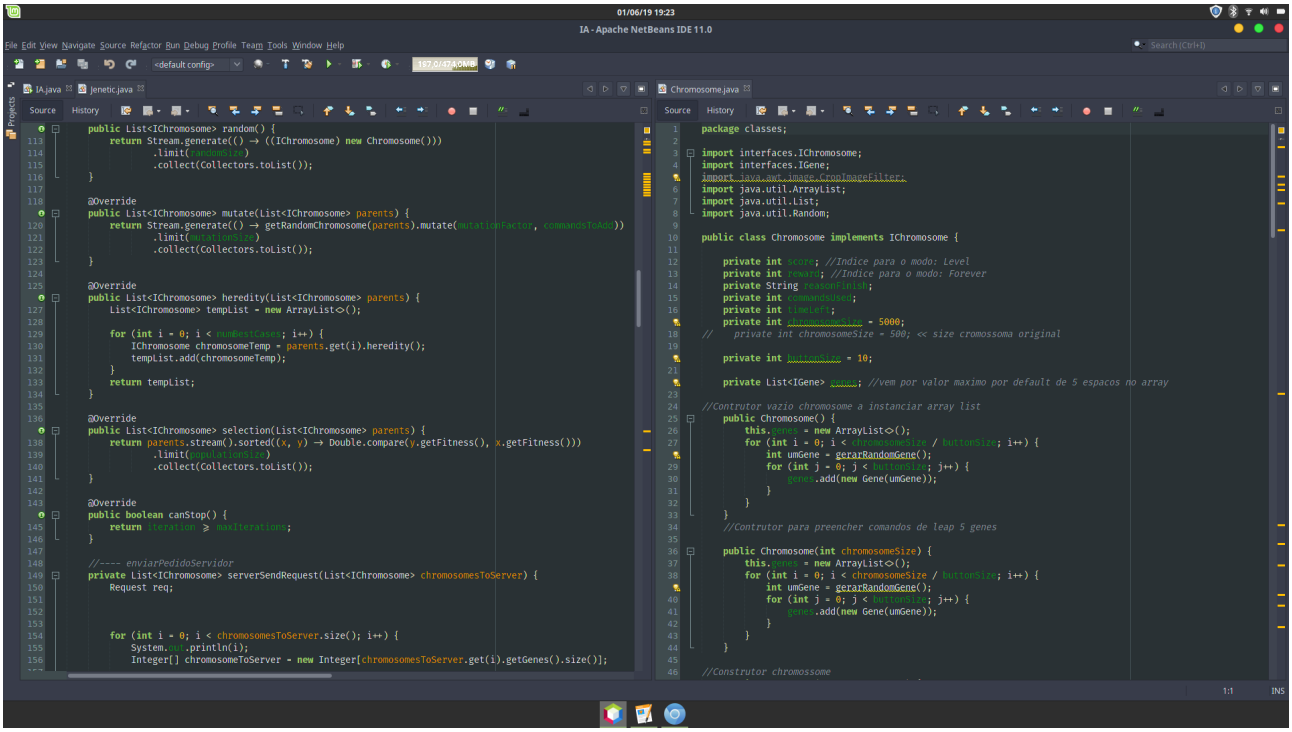
6.2- De que forma e calculado o fitness

Neste modo de jogo podemos definir que o fitness e a valor devolvido pelo metodos “getFitness”. Elegemos os de maior valor.

6.3- De que forma é a informação devolvida pelo servidor utilizada no processo de evolução do Algoritmo Genético

As respostas do servidor ficam numa lista temporaria, que sofrera alteracoes no final de cada geração. Desta for a a lista pode ser ordenada pelo melhor fitness para o pior.

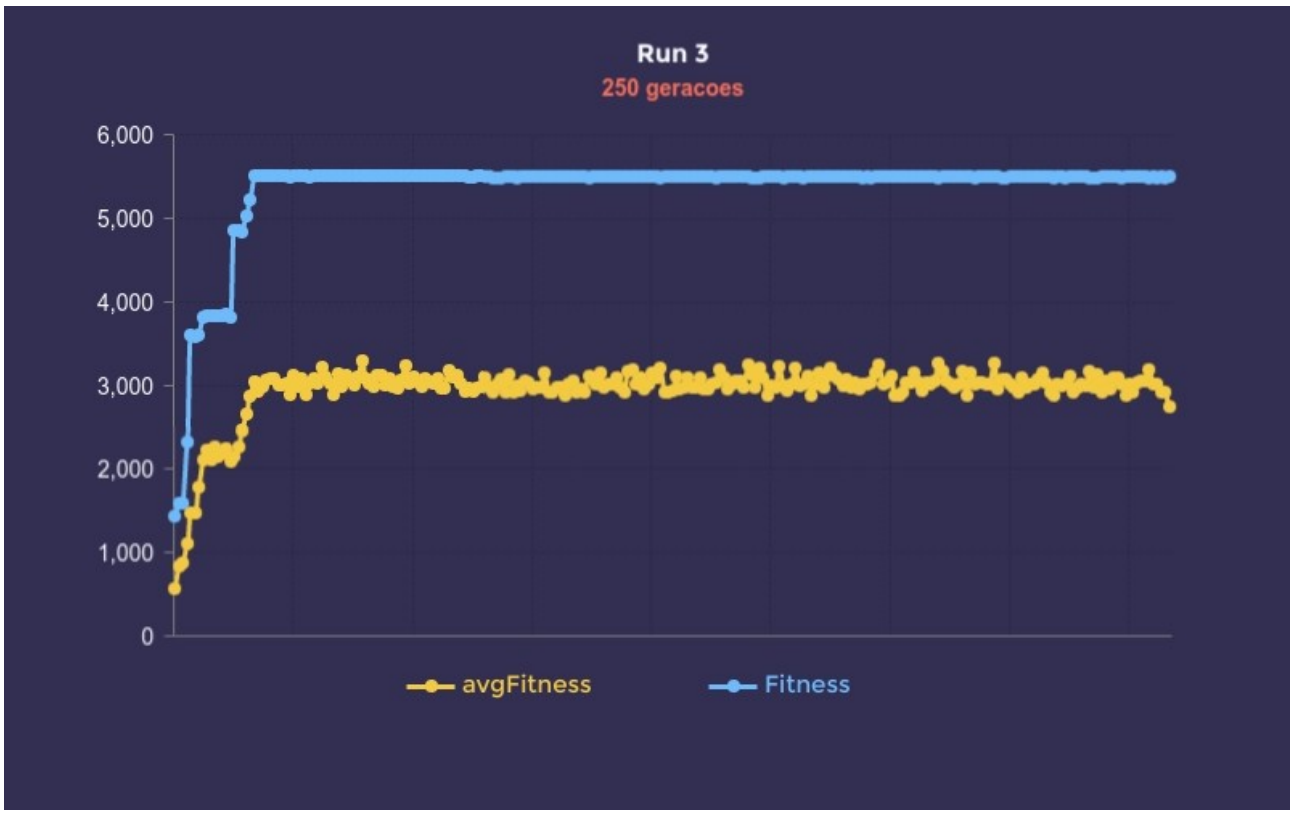
Exemplo do algoritmo em Netbeans



6.4- Qual a configuração com o qual obtiveram um melhor score

A melhor configuracao para o modo “Forever” que obtivemos foi a final. Esta tera os seguintes parâmetros:

Parametro	Valor
`populationSize`	`10`
`randomSize`	4
`mutationSize`	4
`maxIterations`	250
`deltaMin`	0.00
`mutationFactor`	40
`commandsToAdd`	200
`numBestCases`	2
`versao`	0
`render`	“false”
`modoDeJogo`	“forever
ip	Ip_current_vmbox


Run 3
250 geracoes
6,000
5,000
4,000
3,000
2,000
1,000
0
avgFitness
Fitness

7- Conclusão e outras decisões que o grupo considerar importante

Podemos, após a conclusão da elaboração do projeto, afirmar que assimilamos os conceitos básicos de algoritmos evolutivos clássicos. Compreendemos, agora, de forma menos abstrata de que forma os parâmetros têem consequência no comportamento evolutivo e determinar a qualidade dos mesmos. Compreendemos também que encontrar os valores apropriados (configurações) para estes parâmetros é uma tarefa computacionalmente custosa devido a não existência de uma metodologia eficiente que ajude na definição dos mesmos. Desta forma, fomos propondo algumas abordagens ao longo do desenvolvimento do algoritmo.

Em suma, compreendemos melhor os paradigmas das metodologias evolutivas e os problemas inerentes do mesmo depois de desenvolvido este projeto e da superação dos obstáculos que nos fomos deparando no progresso do projeto.