Computação Natural
Trabalho prático 1 - Programação Genética

Universidade Federal de Minas Gerais (UFMG) – Belo Horizonte, MG – Brasil

pedrobrum@dcc.ufmg.br

1. Introdução

Um dos problemas mais conhecidos que pode ser resolvido através de programação genética é a regressão simbólica. Esse problema consiste em descobrir qual a função que melhor se ajusta a um conjunto de amostras representadas por uma tupla $\langle X, Y \rangle$ onde $X \in \mathbb{R}^{m \times n}$ e $Y \in \mathbb{R}^m$.

Neste trabalho foi implementado um algoritmo genético para resolver o problema da regressão simbólica. Os indivíduos são representados por árvore, compostas pos nós terminais e operadores. Os nós terminais das árvores podem ser constantes ou variáveis do problema, e os operadores podem ser as operações matemáticas básicas (+, -, $\div$, $\times$).

O critério de avaliação utilizado para medir a qualidade de um indivíduo foi a raiz quadrada do erro quadrático médio normalizada (NRMSE). Além de considerar as operações de mutação e cruzamento, também devemos considerar o uso de operadores elitistas. Esses operadores. Esses operadores possibilitam a adição do indivíduo gerado à população apenas se ele for melhor (em fitness) que todos os pais. A solução desenvolvida foi executada utilizando 3 conjuntos de dados, sendo 2 sintéticos e 1 real.

Na seção 2 é apresentada uma descrição sobre a implementação do programa, incluindo detalhes da representação, fitness e operadores utilizados. Na seção 5 é apresentada uma análise do impactos dos parâmetros do algoritmo no resultado obtido.

2. Modelagem

2.1. Indivíduo

O primeiro fator que devemos definir em programação genéticas é a representação de um indivíduo. Um indivíduo representa uma solução válidade para o problema e nesse trabalho ele é representado por uma árvore, composta por nós terminais e operadores.

Cada nó terminal da árvore pode ser uma váriavel ou um número aleátorio entre 0.1, 0.2, 0.3, 0.4 e 0.5. Os nós operadores são operações matemáticas e foram definidos de acordo com o conjunto de dados utilizado. Com uma árvore podemos construir uma função matemática a partir do caminhamento em ordem e gerar um conjunto de saídas dado um conjunto de entradas.

2.2. O valor da fitness

Como mencionado anteriormente, neste trabalho os indivíduos de uma população são avaliados utilizando a métrica NRMSE, que pode ser calculada por:

$$NRMSE(Ind) = \sqrt{\frac{\sum_N^{i=1}(y_i - EVAL(Ind, x_i))^2}{\sum_N^{i=1}(y_i - \overline{Y})^2}}$$

onde N é o número de intâncias fornecidas, Ind é o indivíduo sendo avaliado, $EVAL(Ind, x_i)$ avalia o indivíduo Ind na *i-ésima* instância de X, y_i é a saída esperada para entra x_i e $\overline{Y}$ é a média do vetor de saídas Y.

Logo, a patir de uma expressão matemática gerada pelo caminhamento em ordem em árvore (indivíduo), calculamos a $fitness$ do indivíduo. Se para algum X o valor da expressão foi inválido o resultado é definido como zero.

2.3. Geração da população inicial

A população inicial é gerada utilizando os métodos *Ramped half-and-half*. Esse método combinas os métodos *full* e *grow* para aumentar a diversidade inicialmente. Metade dos indivíduos da população são gerados utilizando o método *full* e a outra métade com o método *grow*. No método *grow* o nó de uma árvore é escolhido elementos em ambos os conjuntos de terminais e funções, considerando uma profundidae máxima. No método *full* o nó de uma árvore é escolhido considerando elementos apenas do conjunto de funções até que se chegue na altura máxima. Observe que mesmo definido uma profundidade o método *grow* pode gerar árvores com profundidades menores, pois se o conteúdo de um nó é escolhido de forma aleatória. Assim, o método *grow* pode gerar árvores irregulares. Por outro lado o método *full* cria árvores balanceadas.

Se a profundidade da árvore é n, então serão criados um mesmo número de indivíduos com profundidade 2 até n para cada um dos métodos. Se não for possível gerar a toda a população inicial de forma balanceada em relação aos métodos *full* e *grow* e as profundidates de 2 até N, parte dos indivíduos são gerados de forma aleátoria.

Por exemplo, considere que o tamanho da população seja 50 e que a profundidade máxima da árvore seja 4, serão criados um mesmo número de indivíduos com profundidade 2, 3 e 4 (nesse caso, 16 indivíduos), 8 deles inicializados utilizando *grow* e 8 *full*. Note que dessa forma são gerados 48 indivíduos. Os 2 indivíduos restantes são criados escolhendo o método e a profudidade máxima aleatoriamente.

O uso do método *Ramped half-and-half* pode introduzir introns na população inicial, oferecendo um efeito protetor contra o efeito destruitivo do cruzamento.

2.4. Mutação e cruzamento

Tanto na operação de mutação como na operação de cruzamento os indivíduos são selecionados por torneio, ou seja, são selecionados um conjunto de n indivíduos aleatoriamente e se escolhe o melhor indivíduo em termos de *fitness* do conjunto.

Na operação de cruzamento são utilizado dois indivíduos para gerar um único filho, e portanto, realiza-se torneio duas vezes. A partir desses dois pais geramos um novo indivíduo trocando subárvores escolhidas aleatoriamente: um nó é aleatoriamente escolhido da primeira árvore e outro nó é aleatoriamente escolhido da segunda árvore. Depois troca-se o nó da primeira árvore pelo nó proveniente da segunda árvore. Assim, o filho gerado pela operação de cruzamento consiste na árvore gerada trocando-se um único

nó do primeiro pai selecionado por torneio. Na operação de mutação realiza-se o torneio apenas uma vez e o filho é gerado através do cruzamento entre o único pai seleciondo e um indivíduo gerado aleatoriamente.

Tanto no cruzamento como na mutação definiu-se que a profundidade máxima para selecionar o nó que será trocado é 3. Isso é interessante para resolver esse problema porque limita o tamanho dos indivíduos, impedindo que eles cresçam incontrolavelmente.

Também foi implementada as versões elitistas do cruzamento e da mutação. Nas versões elitistas as operações acontecem da mesma forma que nas versões não-elitistas. A única diferença é que o filho só é adicionado a próxima geração se ele for melhor que os pais. Caso contrário escolhe-se o melhor pai em termos de $fitness$.

Além disso, se as probabilidade de mutação e cruzamento não somem 1, a reprodução ocorre com uma probabilidade de 1 - (p_c + p_m). A operação de reprodução consistem em selecionar um indivíduo por torneio e o adicionar na nova geração.

3. Implementação

A implementação da solução proposta foi realizada utilizando a linguagem $python$. Foram criados objetos que representam os principais envolvidos no GP. Cada objeto foi implementado como uma classe e possui atributos e métodos.

3.1. Tree

A classe $Tree$ representa um indivíduo no GP.

Os principais campos de $Tree$ são:

- $primitives$: representa o conjunto de possíveis valores para um nó.
- $depth$: profundidade da árvore que representa o indivíduo.
- $size$: número máximo de nós do indivíduo.
- $root$: raiz da árvore.

Os principais métodos de $Tree$ são:

- $\_full$: gera um indivíduo de forma aleatoria para a população inicial utilizando o método $full$.
- $\_grow$: gera um indivíduo de forma aleatoria para a população inicial utilizando o método $grow$.
- $build\_program$: realiza o caminhamento em ordem da árvore que representa um indivíduo e retorna uma expressão matemática.
- $node$: escolhe um nó de uma árvore aleatoriamente.

4. Arquivos, compilação e execução

4.1. Arquivos

São partes do código-fonte do programa os seguintes arquivos:

- $main.py$: arquivo principal de execução. Nele estão implementados os métodos de geração da população inicial e de evolução dos indivíduos.
- $gp.py$: arquivo onde estão implementadas as classes $Tree$ e $Node$, seus respectivos métodos e os componentes principais da evolução como: seleção por torneio, mutação, cruzamento, reprodução e cálculo de $fitness$.

4.2. Compilação

A compilação do programa pode ser realizada da seguinte forma:

```
$ python main.py -r trainfile.csv -t testfile.csv
-p 500 -g 30 -m 0.05 -c 0.9 -s 2 -l 1 -o results -b synth1
```

4.3. Execução

Os parâmetros são opcionais, e assumirão o valor padrão se não forem informados.

1. População inicial (inteiro). Padrão: 500
2. Número de gerações (inteiro). Padrão: 30
3. Tamanho máximo de um indivíduo (inteiro). Padrão: 4
4. Probabilidade de cruzamento (float). Padrão: 0.9
5. Probabilidade de mutação (float). Padrão: 0.05
6. Tamanho do torneio (inteiro). Padrão: 2
7. Operadores elitistas (inteiro). Padrão 1
8. Número de repetições (inteiro). Padrão: 30
9. Base de dados (string). Padrão: concrete

5. Experimentos

5.1. Metodologia

Para realizar um estudo dos parâmetros do GP implementado foi escolhido a primeira base de dados sintética $synth1$. Essa base possui 3 atributos, 60 instâncias no conjunto de treino e 600 instâncias no conjunto de teste. Os parâmetros estudados foram: tamanho da população, número de gerações, probabilidade de mutação (p_m), probabilidade de cruzamento (p_c), tamanho do torneio. Além disso, também foi realizado um estudo do impacto de usar somente operadores elitistas. Depois de definir todos os parâmetros, o GP também foi executado para a segunda base sintética $synth2$ e para a base real $concrete$.

Para todos os experimentos realizados o tamanho máximo do indivíduo foi fixado como 4. Como na operação de cruzamento e mutação limitamos a escolha do ponto de troca até a profundidade 3, a altura máxima que um indivíduo pode ter durante a execução do algoritmo é 7. Na operação de cruzamento, se o trocarmos um nó da profundidade 3 por uma subárvore de tamanho 4, geramos um indivíduo de tamanho 7. Alguns teste foram feitos considerando diferentes tamanhos máximo de indivíduo, mas os melhores resultados foram obtidos como tamanho máximo 4. Isso estimula a evolução de indivíduos mais simples.

Para a base $synth1$ os nós terminais de uma árvore podem ser as váriaveis do problema ou os números 0.1, 0.2, 0.3, 0.4 e 0.5. As função escolhidas foram as operação matemáticas +, -, $\div$ e $\times$.

Como o algoritmo baseado em GP é um método estocástico, a avaliação experimental foi realizada com repetições. Para cada escolha de parâmetros o algoritmo foi executado 30 vezes.

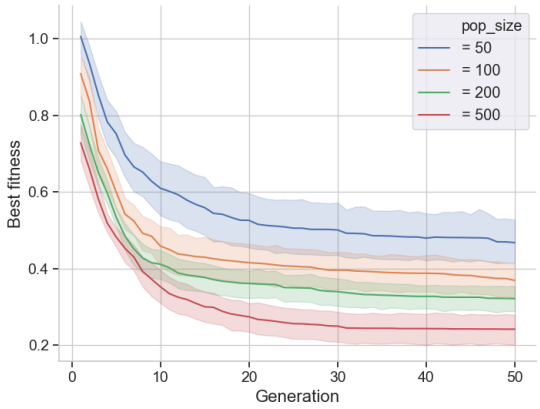

Figura 1: Valor da melhor $fitness$ de uma geração ao longo da evolução considerando população de diferentes tamanhos.

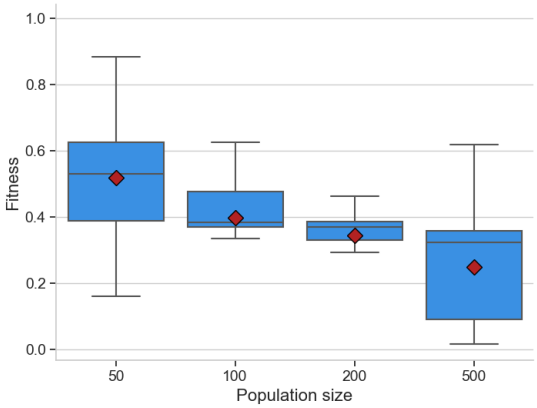

Figura 2: Valores médios de $fitness$ para diferentes tamanhos de população.

5.2. Experimentos

Abaixo estão descritos os experimentos realizados, bem como discussões sobre o resultado.

5.2.1. Tamanho da população

Para gerar os resultados para diferentes tamanhos de população, foram mantidos o elitismo, torneio de tamanho 2, p_c = 0.9, p_m = 0.05 e 50 gerações. Foram gerados resultados para tamanhos de população iguais a 50, 100, 200 e 500.

Observando o gráfico da figura 1 pode-se concluir que o algoritmo evolui a solução de forma que os resultados caminahm para um solução melhor. Além disso, podemos notar que o tamanho da população, e fato, incluencia na evolução da melhor solução. Quanto maior a população melhor é o valor da $fitness$ e mais rápida é a convergência. Quando a população tem tamanho 500 a melhor solução é encontrada aproximadamente na geração 20.

Na figura 2 temos um *box plot* apresentando os resultados finais para cada tamanho de população. O losango vermelho representa o valor médio de $fitness$ obtido pela execução solução utilizando o conjunto de teste. Podemos notar que o melhor resultado foi obtido com população de tamanho 500. Além disso, podemos notar que o resultado quando utilizamos tamanho 100 apresenta um resultado muito próximo de quando utilizamos tamanho 200.

5.2.2. Número de gerações

Para gerar os resultados considerando diferentes números de gerações, foram mantidos o elitismo, torneio de tamanho 2, p_c = 0.9, p_m = 0.05 e população de tamanho 100. Foram gerados resultados considerando 50, 100 e 500 gerações.

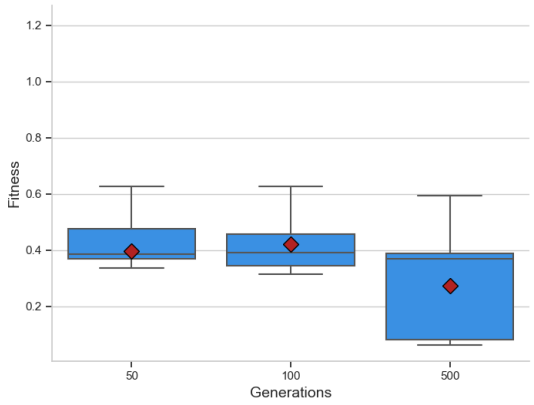

Figura 3: Valores médios de $fitness$ para diferentes números de geração.

Na figura 3 temos um *box plot* apresentando os resultados finais para cada número de gerações. O losango vermelho representa o valor médio de $fitness$ obtido pela execução solução utilizando o conjunto de teste. Podemos notar que o melhor resultado foi obtido com 100 gerações. Além disso, podemos notar que o resultado quando utilizamos 100 gerações apresenta um resultado muito próximo de quando utilizamos 500 gerações. Provavelmente, isso ocorre porque a melhor solução já é alcançada na geração 100 e não consegue evoluir nas próximas gerações. A partir desses resultados e observando o gráfico da figura x, podemos concluir que existe uma relação entre o tamanho da população e o número de gerações.

No *box plot* da figura x, temos que o resultado final médio quando utilizamos 100 geração é aproximadamente 0.4. Quando observamos o gráfico da figura x, temos que para população de tamanho 500 a melhor solução é encontrada, aproximadamente, na geração 20, e quando olhamos para o resultado final temos um valor de $fitness$ médio de aproximadamente 0.2.

Dessa forma, podemos obter um resultado muito melhor utilizando uma população maior e o menor número de gerações. Isso é interessante, visto quanto maior o número de gerações maior é o tempo de execução do algoritmo baseado em GP. A partir disso, para todos os outros experimentos fixamos o tamanho da população como 500 e o número de gerações como 30.

5.2.3. Probabilidade de mutação e cruzamento

Tabela 1: Probabilidade de mutação e cruzamento

p_c	p_m	$Fitness$
0.3	0.6	0.2167
0.2	0.7	0.2461
0.1	0.8	0.2364
0.05	0.9	0.2003

Para gerar os resultados utilizando diferentes probabilidades de cruzamento e mutação, foram mantidos elitismo, 30 gerações, população de tamanho 500 e torneio de

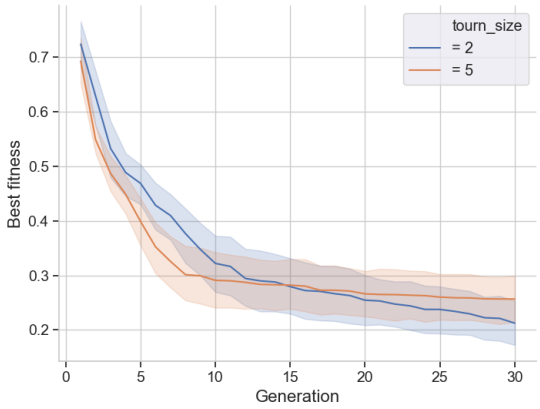

Figura 4: Valor da melhor $fitness$ de uma geração ao longo da evolução considerando torneios de diferentes tamanhos.

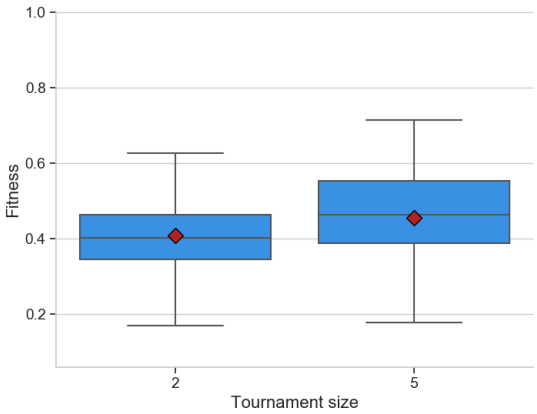

Figura 5: Valores médios de $fitness$ para diferentes tamanhos de torneio.

tamanho 2. Na tabela 1 estão os valores final de $fitness$ considerando diferentes valores para probabilidade de mutação e cruzamento.

Quanto menor é a probabilidade de cruzamento e maior é a probabilidade de mutação maior é a velocidade de convergência, e portanto, o número de indivíduos com uma mesma $fitness$ cresce mais rápido. Além disso, observou-se que com o aumento da probabilidade de mutação e o redução da probabilidade de cruzamento o número de filhos gerados que são melhores que a $fitness$ média dos pais diminui. Isso ocorre, porque a operação de mutação não consegue melhorar tanto a solução quando a operação de cruzamento. Dessa forma, o melhor seria utilizar um probabilidade maior para o cruzamento (0.9) e uma probabilidade menor para a mutação (0.05). Pode-se observar que a combinação que gerou o melhor resultado foi p_m = 0.05 e p_c = 0.9. A partir disso definiu-se que esses seriam os valores padrão para mutação e cruzamento.

5.2.4. Tamanho do torneio

Para gerar os resultados para diferentes tamanhos de torneio, foram mantidos elitismo, p_c = 0.9, p_m = 0.05, 30 gerações, e população de tamanho 500. Foram gerados resultados para tamanhos de torneio 2 e 5.

Observando o gráfico da figura 4 podemos observar que o tamanho do torneio influencia diretamento no valor da melhor $fitness$ ao longo das gerações. Além disso, pode-se notar que a curva que representa o tamanho de torneio 5 é mais acentuada que a curva que representa tamanho de torneio 2. Isso reflete o fato de que quanto maior o tamanho do torneio maior é a pressão seletiva, por conseguinte maior é a velocidade de convegência. Outro fato importante é que o valor da melhor $fitness$ é menor quando consideramos torneio de tamanho 2 do que quando consideramos torneio de tamanho 5. Isso pode ser observado no boxplot da figura 5. No uso de torneio de tamanho 2 a $fitness$ média da solução é aproximadamente 0.4 e no uso de torneio de tamanho 5 esse valor é aproximadamente 0.5. Dessa forma, decidiu-se manter torneio de tamanho 2, tanto

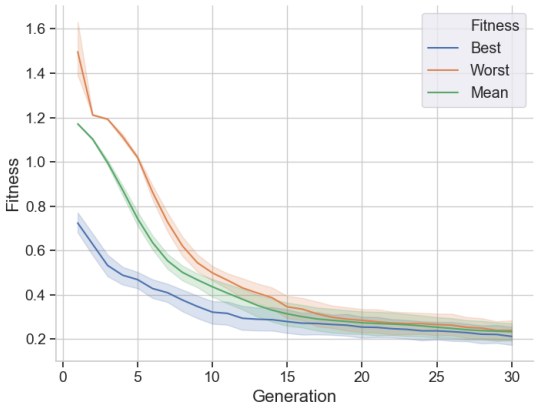

Figura 6: Valores da melhor $fitness$, da pior $fitness$ e da $fitness$ média ao longo da evolução dos indivíduos.

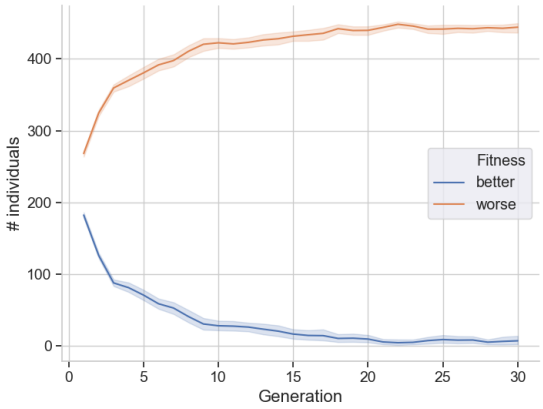

Figura 7: Número de indivíduos gerados a partir do cruzamento que são melhores e piores que a $fitness$ dos pais.

por resultar em uma $fitness$ melhor como por oferecer uma velocidade de convergência menor.

6. Resultados finais

Para cada uma das bases de dados foram gerados os resultados finais considerando os valores parâmetros escolhidos. O conjunto de valores de parâmetros que foram utilizados para gerar os resultados desta seção foi definido no algoritmo como o padrão:

1. População inicial 500
2. Número de gerações: 30
3. Tamanho máximo de um indivíduo: 4
4. Probabilidade de cruzamento: 0.9
5. Probabilidade de mutação: 0.05
6. Tamanho do torneio: 2
7. Operadores elitistas: 1

6.1. Bases sintéticas

6.1.1. Synth1

Pelo gráfico da figura 6 podemos observar que os valores de melhor $fitness$, pior $fitness$ e $fitness$ média convergem para o mesmo valor que é atingido aproximadamente na geração 30. Além disso, podemos notar que ao longo da evolução o valor de $fitness$ sempre cai. Isso ocorre devido ao uso de operadores elitistas, que fazem com que os melhores indivíduos persistem nas próximas gerações.

No gráfico da figura 7 podemos observar o número de indivíduos gerados pelo cruzamento que são melhores e piores que a $fitness$ média dos pais. Pode-se notar que o número de indivíduos gerados que são melhores diminuem com o número de geração e que o número de indivíduos gerados que são piores aumentam com o número de gerações. Além disso, pode-se observar que o número de indivíduos melhores converge para 0 e o número de indivíduos melhores converge para o tamanho da população.

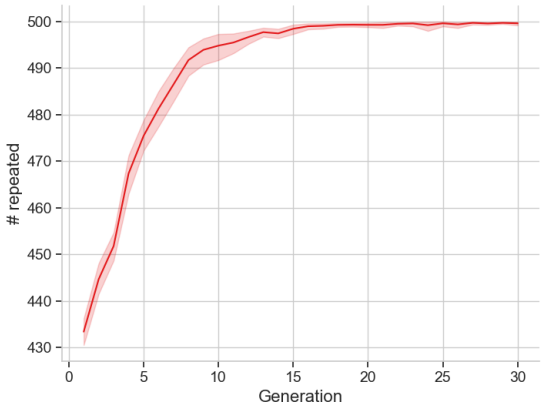

Figura 8: Número de indivíduos repetidos ao longo da evolução.

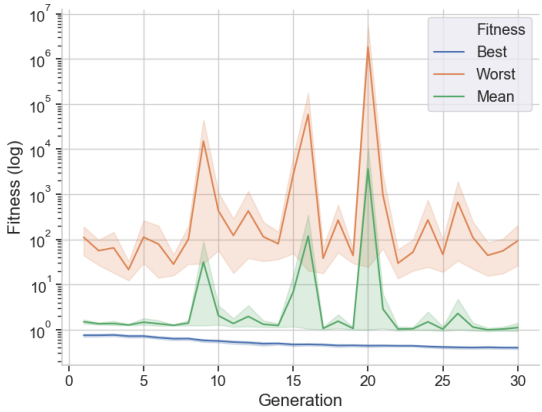

Figura 9: Valores da melhor $fitness$, da pior $fitness$ e da $fitness$ média ao longo da evolução sem o uso de operadores elitistas.

No gráfico da figura 8 temos o número de indivíduos da população que são repetidos em termos de $fitness$, ou seja, se dois indivíduos possuem a mesma $fitness$ eles são considerados iguais. Pode-se notar que o número de indivíduos repetidos aumenta ao longo da evolução até convergir para o tamanho da população. O número de indíviduos repetidos e o número de indvíduos gerados que são melhores e piores que a $fitness$ média dos pais pode ser uma forma de se medir $bloating$.

6.1.2. Operadores elitistas

Para gerar os resultados sem o uso de operatores elitistas, foram mantidos todos os outros parâmetros.

Pelo gráfico da figura 9 podemos observar que os piores valores de $fitness$ e o valor médio de $fitness$ são bem maiores que a melhor $fitness$ ao longo da evolucção. Além disso, diferente do que ocorre quando usamos os operadores elitistas, os valores da pior $fitness$ e da $fitness$ média é variam muito ao durante a execução do GP e não

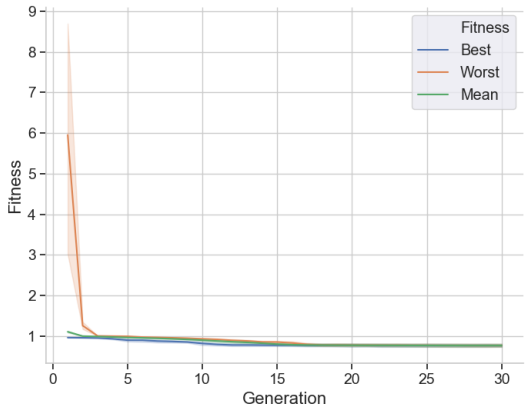

Figura 10: Valores da melhor $fitness$, da pior $fitness$ e da $fitness$ média ao longo da evolução dos indivíduos.

convergem para o mesmo valor que a melhor $fitness$.

6.1.3. Synth2

Para a segunda base de dados sintética foram gerados os resultados quando utilizamos somente operadores elitistas na evolução.

Os resultados podem ser vistos no gráfico da figura 10. Podemos concluir que a melhor $fitness$, a pior $fitness$ e a $fitness$ média apresentam o mesmo comportamento observados na primeira base de dados sintética ($synth1$) e que esses valores convergem para o mesmo valor. Porém, todos valores de $fitness$ apresentam um valor maior em relação aos valores observados na $synth1$. Provavelmente, isso ocorre porque as funções utilizadas não conseguem formar uma função matemática que representa o conjunto de dados de forma tão eficiente quanto na $synth1$. No entando, quando olhamos para o valor da melhor $fitnes$ podemos notar que apresenta um valor menor do que 1 ao longo da evolução. Avaliando as soluções com o conjunto de teste, temos que a média do valor final da $fitness$ é 0.8126, o que pode ser considerado um bom resultado.

6.2. Base real

Para a base de dados real foram gerados resultados utilizando o conjuntos de valores padrão definido, variando o tamanho do torneio e sem o uso de operadores elitistas.

No gráfico da figura 11 podemos observar os valores da melhor $fitness$, pior $fitness$ e $fitness$ média ao longo da evolução. Da mesma forma que a base $synth2$ temos que esses valores apresentam o mesmo comportamento observados na primeira base de dados sintética ($synth1$) e que esses valores convergem para o mesmo valor. Avaliando as soluções com o conjunto de teste, temos que a média do valor final da $fitness$ é 0.7748, o que pode ser considerado um bom resultado.

Na figura 12 temos o gráfico com o número de indivíduos gerados pelo cruzamento que são melhores que a $fitness$ média dos pais e que são piores que a $fitness$ média dos pais ao longo da evolução. Na figura 13 temos o gráfico com o número de indivíduos

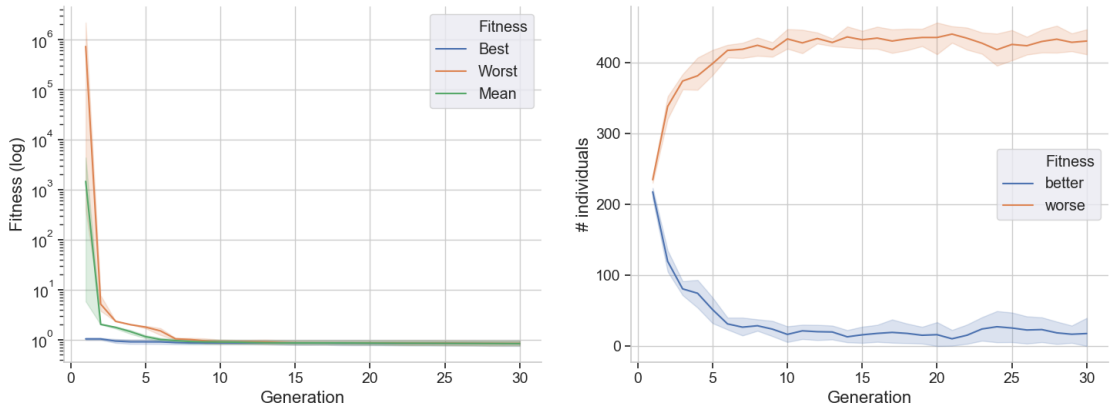

Figura 11: Valores da melhor $fitness$, daFigura 12: Número de indivíduos gerados a pior $fitness$ e da $fitness$ média ao longopartir do cruzamento que são melhores e pida evolução dos indivíduos para a base real. ores que a $fitness$ dos pais para a base real.

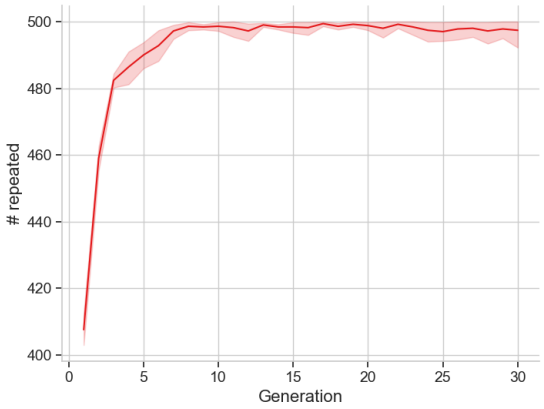

Figura 13: Número de indivíduos repetidos ao longo da evolução para a base real.

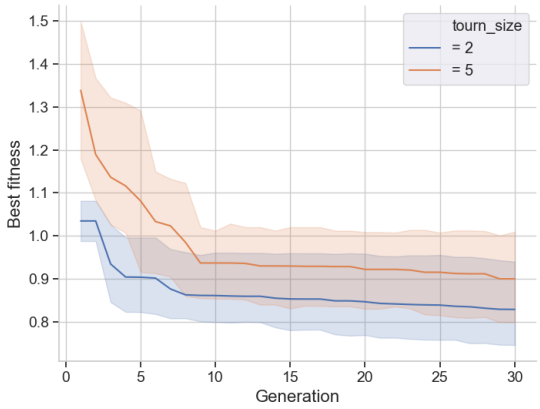

Figura 14: Valor da melhor $fitness$ de uma geração ao longo da evolução considerando torneios de diferentes tamanhos para a base real.

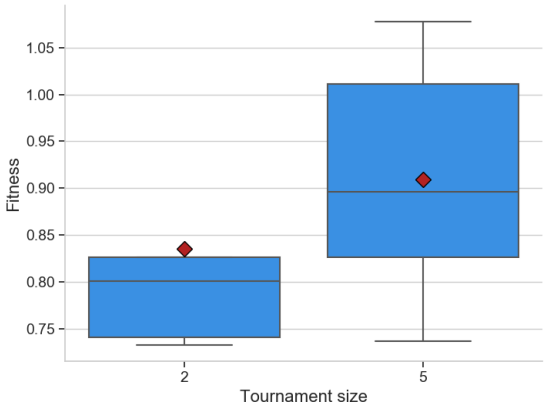

Figura 15: Valores médios de $fitness$ para diferentes tamanhos de torneio para a base real.

repetidos ao longo da evolução. Como esperado, a base real apresentou resultados bem similares aos gerados para as bases sintéticas.

6.2.1. Tamanho do torneio

Observando o gráfico da figura 14 podemos observar que o tamanho do torneio também influencia diretamento no valor da melhor $fitness$ ao longo das gerações para a base dados real. Além disso, pode-se notar que a curva que representa o tamanho de torneio 5 é mais acentuada que a curva que representa tamanho de torneio 2, representando uma maior pressão seletiva. Outro fato importante é que o valor da melhor $fitness$ é menor quando consideramos torneio de tamanho 2 do que quando consideramos torneio de tamanho 5. Isso pode ser observado no boxplot da figura 15. No uso de torneio de tamanho 2 a $fitness$ média da solução é aproximadamente 0.85 e no uso de torneio de tamanho 5 esse valor é aproximadamente 0.90.

6.2.2. Operadores elitistas

Os operadores elitistas também impactam muito nos valores de $fitness$ ao longo das gerações. Como observado nas bases de dados sintéticas os valores de $fitness$ média e pior $fitness$ variam muito e não convergem para o mesmo valor que a melhor $fitness$.

7. Conclusão

Neste trabalho implementamos um algoritmo baseado em GP para resolver o problema de regressão simbólica. Os parâmetros do algoritmo implementado são tamanho máximo do indivíduo, tamanho da população, número de geração, probabilidade de mutação, probabilidade de cruzamento, tamanho do torneioe e uso de operadores elitistas. Para escolher os melhores parâmetros do GP foi realizado um estudo considerando cada um deles individualmente. A escolha desses parêmtros não é uma tarefa fácil, visto que eles podem

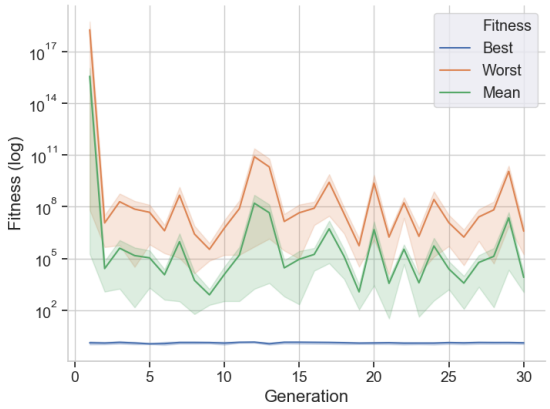

Figura 16: Valores da melhor $fitness$, da pior $fitness$ e da $fitness$ média ao longo da evolução sem o uso de operadores elitistas para a base real.

alterar drasticamente o resultado. A realização de testes foi fundamental para entender a efeito de cada um dos parâmentros nos valores finais de $fitness$, pressão seletiva e o número de indivíduos repetidos ao longo da evolução. Com a escolha dos parâmetros foram gerados os resultados para cada uma das três bases disponíveis ($synth1$, $synth2$, $concrete$). Os melhores resultados foram obtidos na base sintética $synth1$. Apesar de ser difícil determinar a combinação ótima de parâmentros as outras bases de dados também apresentaram bons resultados na média.

Referências

[1] R. Poli, W.B. Langdoo, N.F. McPhee, J.R. Koza. *A Field Guide to Genetic Programming*. (2008) 1-250.