

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA

GUSTAVO ALVES PACHECO

OTIMIZAÇÕES COM RESTRIÇÕES

Uberlândia-MG

2019

GUSTAVO ALVES PACHECO

OTIMIZAÇÕES COM RESTRIÇÕES

Trabalho apresentado ao curso de Engenharia de Computação da Universidade Federal de Uberlândia, como requisito parcial de avaliação da disciplina de Algoritmos Genéticos.

Prof.: Keiji Yamanaka

Uberlândia-MG

2019

SUMÁRIO

1.	INTRODUÇÃO	4
2.	OBJETIVOS	4
3.	MATERIAIS E MÉTODOS	5
4.	RESULTADOS E DISCUSSÕES	6
5.	CONCLUSÃO	11
6.	REFERÊNCIAS BIBLIOGRÁFICAS	13

1. INTRODUÇÃO

Existem situações, como algumas já tratadas anteriormente, em que é necessário estabelecer restrições em relação aos indivíduos de uma população, ou acerca da aptidão desses mesmos cromossomos. Foi o caso da ordem de manutenção dos geradores, no trabalho “Resolução de um problema de scheduling” ou da verificação se a peça de xadrez se encontrava dentro do tabuleiro durante o projeto “Problema do tour do cavalo”.

Apesar de serem restrições recorrentes, é possível modificá-las, de forma que, ao invés de trabalhar com um problema com restrição, trabalhe-se com uma otimização com penalidade. Na qual já não é mais necessário verificar constantemente a situação do indivíduo, e sim atribuir penalidades àqueles que não se enquadram nos requisitos.

Para isso, será minimizada a função:

$$f(x, y) = (x - 1)^2 + (y - 1)^2$$

Com as variáveis entre:

$$\begin{cases} -3 \leq x \leq 5 \\ -3 \leq y \leq 5 \end{cases}$$

E usando as seguintes restrições:

$$\begin{cases} x + y \leq 0.5 \\ x - y = 2 \end{cases}$$

A forma como a penalidade foi aplicada será discutida com mais detalhes na seção 3.

2. OBJETIVOS

- Adaptar o algoritmo genético atual para que trabalhe com penalidades

- Adequar a interface gráfica utilizada no projeto de ‘Parâmetros contínuos’, de forma a abranger os novos parâmetros relativos à penalização dos indivíduos.
- Verificar impacto dos parâmetros do algoritmo genético, no processo de *tuning*
- Encontrar o ponto de mínimo da função citada acima, que satisfaça as condições impostas

3. MATERIAIS E MÉTODOS

A linguagem utilizada foi o Racket, uma linguagem de uso geral, baseada em Scheme e Lisp. Como interface de desenvolvimento, foi utilizado o DrRacket.

Para confecção da interface gráfica, utilizou-se a linguagem *racket/gui*, um recurso nativo da própria linguagem Racket. O processo de plotagem foi realizado através das funções do pacote *plot*.

Como métodos, utilizou-se uma abordagem *top-down*, com auxílio de um framework SCRUM para desenvolvimento ágil. Em algumas situações, uma visão *bottom-up* foi adicionada ao projeto, na parte de incremento da própria linguagem, ao criar novos operadores e funcionalidades. O código produzido segue majoritariamente uma abordagem funcional, mas possui elementos procedurais, para facilitar a leitura de algumas funções.

Durante o desenvolvimento, a documentação de funções se mostrou fundamental. Principalmente, para que algumas estruturas de dados fossem melhor especificadas, e a execução do código fosse correta.

Duas formas de penalidade foram tratadas. A primeira, corresponde à uma restrição por inequação, na qual os genes do cromossomo devem respeitar condições do tipo menor ou igual a um outro valor. Para esse tipo de restrição, a inequação é primeiramente convertida em uma homogênea, ou seja, menor ou igual a zero, e o termo à esquerda da inequação é avaliado separadamente. Caso seja maior que zero, o mesmo é adicionado à penalidade (ou o quadrado do valor). Caso não seja, a penalidade deve assumir o valor zero (não será aplicada).

Esse valor deve ser somado (para o caso de minimização), ou subtraído (maximização), de forma a piorar a aptidão do indivíduo.

O segundo caso tratado foi o de igualdades, no qual uma função dos genes do cromossomo devia ser igual a zero. Nesse caso, o quadrado do valor da penalidade é somado/subtraído ao fitness, já que a situação ideal é quando a função retorna 0, não adicionando nenhuma penalidade.

Uma constante r_p é adicionada, também, de forma a controlar a intensidade do valor da penalidade em si. Dessa forma, a função de cálculo do fitness passa a ser:

$$f(x_1, x_2, \dots, x_n) = fitness(x_1, x_2, \dots, x_n) + r_p((\max(0, ineqPenalty(x_1, x_2, \dots, x_n)))^2 + (eqPenalty(x_1, x_2, \dots, x_n))^2)$$

4. RESULTADOS E DISCUSSÕES

Inicialmente, a interface foi adaptada para abranger os problemas com restrições de inequações e equações. Quatro novos campos foram adicionados. Um checkbox para habilitar ou desabilitar penalidades, uma caixa de texto para inserir o r_p , e outras duas caixas para inserir as penalidades (deve-se inserir a função penalidade, no qual o resultado da avaliação será adicionado/subtraído do fitness). A figura 1 mostra os novos campos da interface.

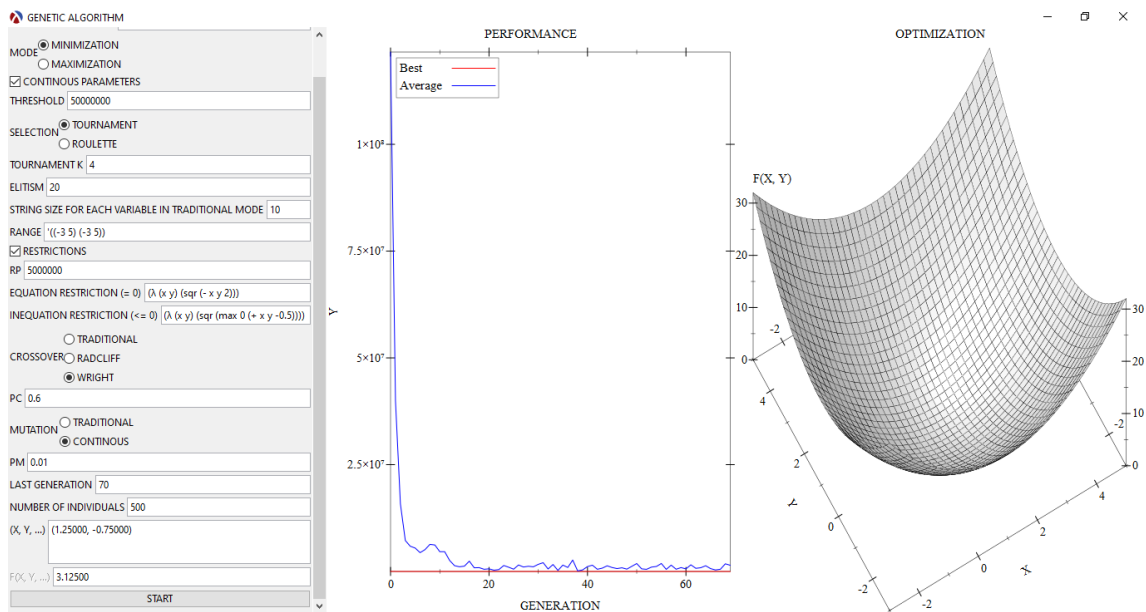
The image shows a software interface for defining constraints. It has a section titled 'RESTRICTIONS' with a checkbox. Below it, there are three input fields: 'RP' with the value '50000', 'EQUATION RESTRICTION (= 0)' with the value '(\lambda (x y) (sqr (- x y 2)))', and 'INEQUATION RESTRICTION (<= 0)' with the value '(\lambda (x y) (sqr (max 0 (+ x y -0.5))))'.

Figura 1: Adições à interface, com valores default visando desafio atual.

A seguir, a função que calculava o fitness foi modificada. Esse processo foi relativamente simples, já que a função agora passa a ser de alta ordem, recebendo os novos incrementos/decrementos na função que era aplicada anteriormente.

A partir daí, testes foram feitos para encontrar o melhor resultado para a função especificada na seção 1. O melhor resultado retornou os valores abaixo, como mostra a figura 2:

$$\begin{cases} x = 1.25 \\ y = -0.75 \end{cases}$$



MODE ☒ MINIMIZATION
☐ MAXIMIZATION

☒ CONTINUOUS PARAMETERS

THRESHOLD 50000000

SELECTION ☒ TOURNAMENT
☐ ROULETTE

TOURNAMENT K 4

ELITISM 20

STRING SIZE FOR EACH VARIABLE IN TRADITIONAL MODE 10

RANGE '((-3 5) (-3 5))'

☒ RESTRICTIONS

RP 5000000

EQUATION RESTRICTION ($= 0$) $(\lambda(x, y) (\text{sqr}(-x, y, 2)))$

INEQUATION RESTRICTION (≤ 0) $(\lambda(x, y) (\text{sqr}(\max(0, (+x, y, -0.5)))))$

☐ TRADITIONAL

CROSSOVER ☐ RADCLIFF
☒ WRIGHT

PC 0.6

MUTATION ☐ TRADITIONAL
☒ CONTINUOUS

PM 0.01

LAST GENERATION 70

NUMBER OF INDIVIDUALS 500

(X, Y, ...) (1.25000, -0.75000)

F(X, Y, ...) 3.12500

Figura 2: Melhor valor encontrado para a função desejada.

Nota-se que é necessário um valor elevado de r_p para que a função retorne um valor preciso. Caso seja mais baixo, raramente o valor fixa-se nesses. Abaixo estão imagens de outros testes realizados, com diferentes configurações. Foram omitidos os gráficos de performance e o da função, já que são semelhantes, em todas as situações. Em alguns casos, o valor da função chega a ser menor que o encontrado acima, mas as condições estabelecidas não são completamente cumpridas.

GENETIC ALGORITHM

MODE ☒ MINIMIZATION
☐ MAXIMIZATION

☒ CONTINUOUS PARAMETERS

THRESHOLD 50000000

SELECTION ☒ TOURNAMENT
☐ ROULETTE

TOURNAMENT K 4

ELITISM 20

STRING SIZE FOR EACH VARIABLE IN TRADITIONAL MODE 10

RANGE '((-3 5) (-3 5))

☒ RESTRICTIONS

RP 1

EQUATION RESTRICTION (= 0) $(\lambda (x y) (\sqrt{(- x y 2)}))$

INEQUATION RESTRICTION (≤ 0) $(\lambda (x y) (\sqrt{(\max 0 (+ x y -0.5))}))$

☐ TRADITIONAL

CROSSOVER ☐ RADCLIFF
☒ WRIGHT

PC 0.6

MUTATION ☐ TRADITIONAL
☒ CONTINUOUS

PM 0.01

LAST GENERATION 70

NUMBER OF INDIVIDUALS 500

X, Y, ... (1.16667, -0.16667)

F(X, Y, ...) 1.38891

START

Figura 3: Um valor baixo de R_p minimiza a função, mas não leva muito em conta a penalização dos indivíduos.

GENETIC ALGORITHM

MODE ☒ MINIMIZATION
☐ MAXIMIZATION

☒ CONTINUOUS PARAMETERS

THRESHOLD 50000000

SELECTION ☒ TOURNAMENT
☐ ROULETTE

TOURNAMENT K 4

ELITISM 20

STRING SIZE FOR EACH VARIABLE IN TRADITIONAL MODE 10

RANGE '((-3 5) (-3 5))

☒ RESTRICTIONS

RP 1000

EQUATION RESTRICTION (= 0) $(\lambda(x, y) (\text{sqr}(-x y^2)))$

INEQUATION RESTRICTION (≤ 0) $(\lambda(x, y) (\text{sqr}(\max(0, (+ x y -0.5)))))$

☐ TRADITIONAL

CROSSOVER ☐ RADCLIFF
☒ WRIGHT

PC 0.6

MUTATION ☐ TRADITIONAL
☒ CONTINUOUS

PM 0.01

LAST GENERATION 70


NUMBER OF INDIVIDUALS 500

(X, Y, ...) (1.24988, -0.74912)

F(X, Y, ...) 3.12188

START

Figura 4: Aumentando o valor de Rp, percebe-se que a função passa a obedecer às restrições.

 GENETIC ALGORITHM

MODE ☒ MINIMIZATION
☐ MAXIMIZATION

☒ CONTINUOUS PARAMETERS

THRESHOLD 50000000

SELECTION ☒ TOURNAMENT
☐ ROULETTE

TOURNAMENT K 4

ELITISM 20

STRING SIZE FOR EACH VARIABLE IN TRADITIONAL MODE 10

RANGE '((-3 5) (-3 5))

☒ RESTRICTIONS

RP 10000000

EQUATION RESTRICTION ($= 0$) $(\lambda(x, y) (\text{sqr}(-x, y^2)))$

INEQUATION RESTRICTION (≤ 0) $(\lambda(x, y) (\text{sqr}(\max(0, (+x, y - 0.5)))))$

☐ TRADITIONAL

CROSSOVER ☐ RADCLIFF
☒ WRIGHT

PC 0.6

MUTATION ☐ TRADITIONAL
☒ CONTINUOUS

PM 0.01

LAST GENERATION 70

NUMBER OF INDIVIDUALS 500

(X, Y, ...) (1.24175, -0.75821)

F(X, Y, ...) 3.14976

START

Figura 5: Percebe-se que o valor se aproxima do melhor resultado, à medida que Rp aumenta.

GENETIC ALGORITHM

MODE ☒ MINIMIZATION
☐ MAXIMIZATION

☒ CONTINUOUS PARAMETERS

THRESHOLD 50000000

SELECTION ☒ TOURNAMENT
☐ ROULETTE

TOURNAMENT K 4

ELITISM 20

STRING SIZE FOR EACH VARIABLE IN TRADITIONAL MODE 10

RANGE '((-3 5) (-3 5))

☐ RESTRICTIONS

RP 10000000

EQUATION RESTRICTION (= 0) $(\lambda(x, y) (\text{sqr}(-x, y^2)))$

INEQUATION RESTRICTION (≤ 0) $(\lambda(x, y) (\text{sqr}(\max(0, (+x, y - 0.5)))))$

☐ TRADITIONAL

CROSSOVER ☐ RADCLIFF
☒ WRIGHT

PC 0.8

MUTATION ☐ TRADITIONAL
☒ CONTINUOUS

PM 0.01

LAST GENERATION 70

NUMBER OF INDIVIDUALS 500

(X, Y, ...) (1.00001, 1.00000)

F(X, Y, ...) 0.00000

START

Figura 6: Otimização sem as restrições.

5. CONCLUSÃO

Apesar de já ter aparecido em problemas anteriores, as restrições nos indivíduos são essenciais para a obtenção do resultado desejado, na maioria das situações práticas, onde o AG é proveitoso. Neste projeto, uma nova alternativa, bastante interessante, foi apresentada: A de usar penalidades ao invés de restrições.

A performance do algoritmo realmente aumenta bastante, utilizando essa estratégia, e a mesma ainda permite uma variedade maior de soluções, não se restringindo a valores específicos do espaço de busca.

Com certeza um incremento fundamental ao processo que já estava sendo realizado, e um passo importante para a utilização dos algoritmos genéticos em problemas práticos.

6. REFERÊNCIAS BIBLIOGRÁFICAS

- Al-Gharaibeh, J., Qawagneh, Z., & Al-Zahawi, H. (29 de Outubro de 2015). Genetic Algorithms with Heuristic. *Research Gate*.
- Mallawaarachchi, V. (7 de Julho de 2017). *Introduction to Genetic Algorithms*. Fonte: Towards Data Science: <https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3>
- Montesanti, J. d. (s.d.). *Seleção natural*. Fonte: InfoEscola: <https://www.infoescola.com/evolucao/selecao-natural/>
- Tomassini, M. (s.d.). A Survey of Genetic Algorithms. *Annual Reviews of Computational Physics, Volume III*.
- Yamanaka, P. K. (Agosto de 2017). *ALGORITMOS GENÉTICOS: Fundamentos e Aplicações*.
- Zini, É. d., Neto, A. B., & Garbelini, E. (18 de Novembro de 2014). ALGORITMO MULTIOBJETIVO PARA OTIMIZAÇÃO DE PROBLEMAS RESTRITOS APLICADOS A INDÚSTRIA. *Congresso Nacional de Matemática Aplicada à Indústria*.