Algoritmo Genético para Emparelhamento Diversificado

Lucas Lima de Melo
INF0510 - Otimização Combinatória
Prof.Marcus Ritt

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

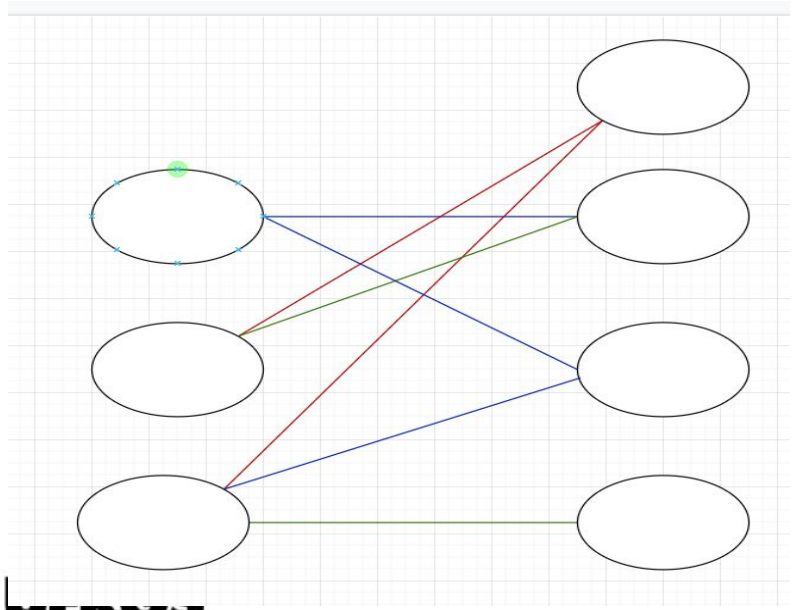
.inf
INSTITUTO DE INFORMÁTICA
UFRGS

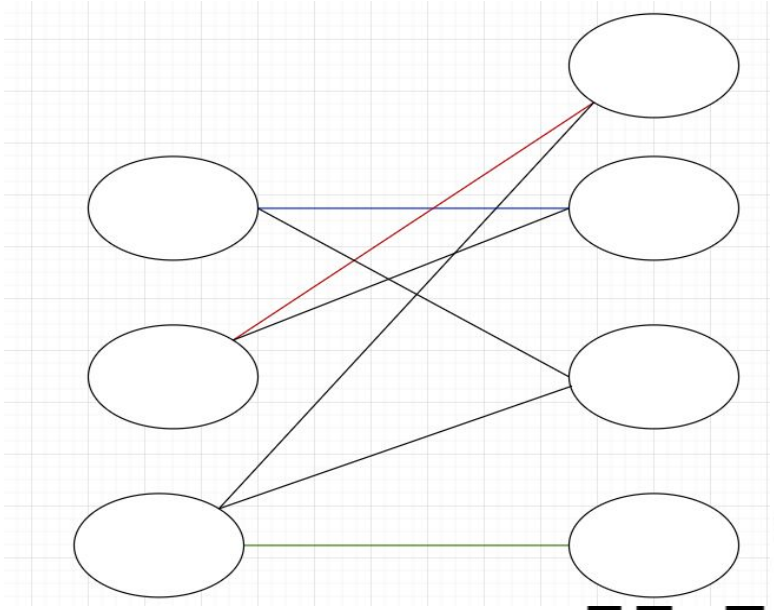
Introdução

O objetivo deste trabalho é implementar a metaheurística algoritmo genético para resolver o problema do Emparelhamento Diversificado.

O problema do Emparelhamento Diversificado é definido como: Dado um grafo não direcionado G=(V,A) onde cada aresta $a \in A$ possui um tipo ta, deseja-se encontrar um emparelhamento,ou seja um conjunto de arestas tal que não existem duas arestas adjacentes neste conjunto, que maximize o número de arestas nele contido,no qual todas arestas possuem tipos diferentes.

Exemplo:





Formulação

Variáveis:

- $x_e \in \{0, 1\}, \quad \forall e \in A$, onde

$$x_e = \begin{cases} 1, & \text{Caso a aresta e faça parte do emparelhamento} \\ 0, & \text{Caso contrario} \end{cases}$$

- $z_{ek} \in \{0, 1\}, \quad \forall e, k \mid e \in A, k \in C$, onde

$$z_{ek} = \begin{cases} 1, & \text{Caso } x_e \wedge y_{ek} \text{ é verdade} \\ 0, & \text{Caso contrario} \end{cases}$$

Constante:

- $y_{ek} \in \{0, 1\}, \quad \forall e, k \mid e \in A, k \in C$, onde

$$y_{ek} = \begin{cases} 1, & \text{Caso k seja a cor da aresta e} \\ 0, & \text{Caso contrario} \end{cases}$$

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

.inf UFRGS

Formulação

Função Objetivo:

$$\text{Max.} \quad \left(\sum_{e \in A} x_e \right)$$

Restrições:

$$\sum_{e \in A \ \wedge \ e\#v} x_e \leq 1, \quad \forall v \in V, \quad \text{e\#v representa que a aresta e é incidente em v.} \tag{1}$$

$$y_{ek} = 0, \quad \forall e, k \mid e \in A, k \in C \wedge k \neq t_e \tag{2}$$

$$\sum_{k \in C} y_{ek} = 1, \quad \forall e \in A \tag{3}$$

$$\sum_{e \in A} z_{ek} \leq 1, \quad \forall k \in C \tag{4}$$

C representa o conjunto de cores(tipos) unícas.
A restrição (1) garante que as arestas não sejam incidentes.
A restrição (2) garante que y_{ek} seja 0 quando a cor k for diferente da cor de entrada t_e da aresta e.
A restrição (3) garante que cada aresta possua exatamente 1 cor.
A restrição (4) garante que no máximo 1 aresta de cor k faça parte do emparelhamento.

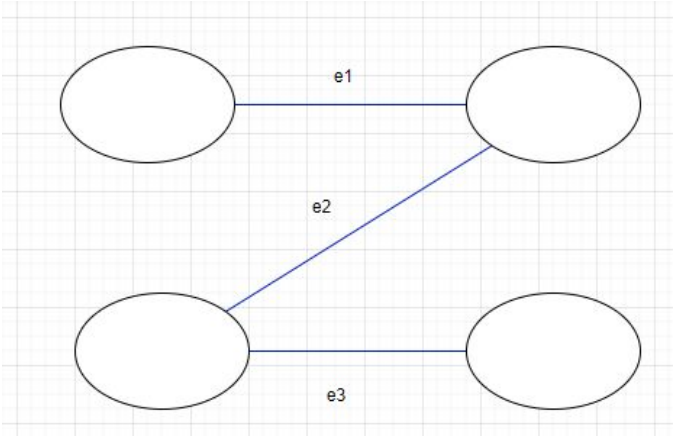
Algoritmo Genético:Parâmetros

- Population_Size:Representa a quantidade de indivíduos na população.
- New_p_Size:Representa a quantidade de novos indivíduos que serão geradosa cada iteração do algoritmo.
- Mutation_Rate:Representa a taxa de mutação, valor entre 0 e 1.
- Max_Non_Improving_Gen:Representa um dos critérios de parada do programa,que irá terminá-lo caso 'Max Non Improving Gen' gerações forem geradas sem melhora na solução final.
- Max_time:Tempo limite: o algoritmo executará por no máximo Max_time segundos.

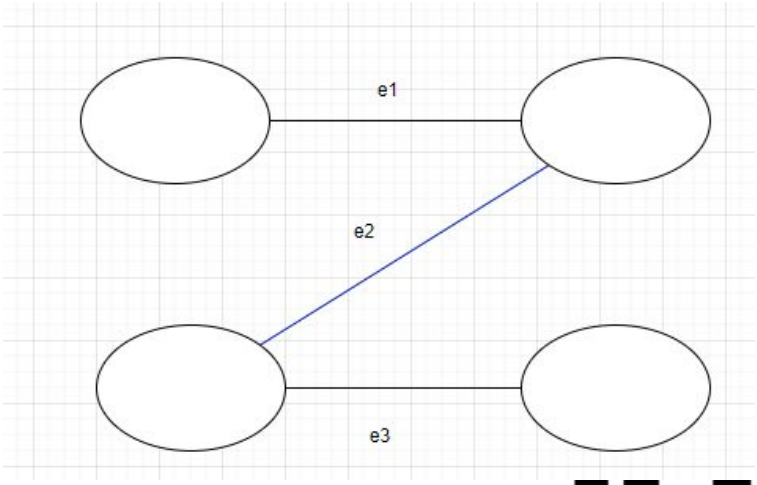
Algoritmo Genético:População Inicial

A população inicial é gerada aleatoriamente.São gerados 'Population_Size' vetores binários aleatórios,como provavelmente estes vetores não serão factíveis, uma aresta é escolhida aleatoriamente para fazer parte do emparelhamento e a solução é refactibilizada a partir desta aresta.A refactibilização é feita percorrendo a lista de genes.Caso o gene seja 1, ele é testado conforme as condições do emparelhamento diversificado, levando em consideração a aresta escolhida anteriormente e setado para 0 caso não possa participar do emparelhamento.Assim no final,1 indica que a aresta faz parte do emparelhamento e 0 caso contrário

Supor que gerou vetor(1,1,1)



Após a refactibilização:(0,1,0),se aresta e2 for escolhida



UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

UFRGS

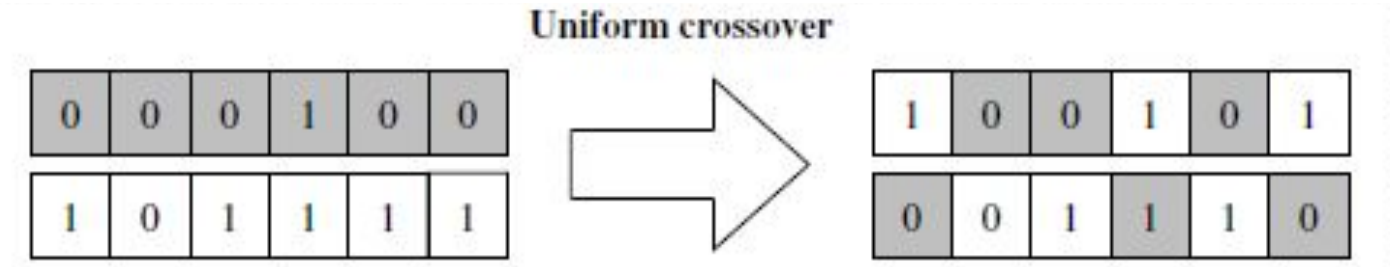
Algoritmo Genético:Método de Seleção de Indivíduos para Crossover

- O Método de Seleção de Indivíduos para Crossover é feito através de torneio, onde k indivíduos são selecionados para o torneio e é retornado o indivíduo com maior fitness.

- Ocorrem duas execuções do método para escolher dois indivíduos que serão, posteriormente, recombinados.

Algoritmo Genético:Crossover

- O crossover é feito através do método de Uniform-Crossover.Dado os indivíduos p1 e p2,cada bit do indivíduo p1 tem 50 \% de chance de ser trocado com o bit correspondente do indivíduo p2, gerando assim os dois novos indivíduos que serão retornados.
- Também é necessário executar uma função que refactibilize o indivíduo, pois o processo pode levar a um indivíduo infactível.
- A refactibilização é feita percorrendo a lista de genes.Caso o gene seja 1, ele é testado conforme as condições do emparelhamento diversificado,e setado para 0 caso ele não possa participar do emparelhamento.
- Complexidade:melhor caso=O(N), pior caso=O(N^2)



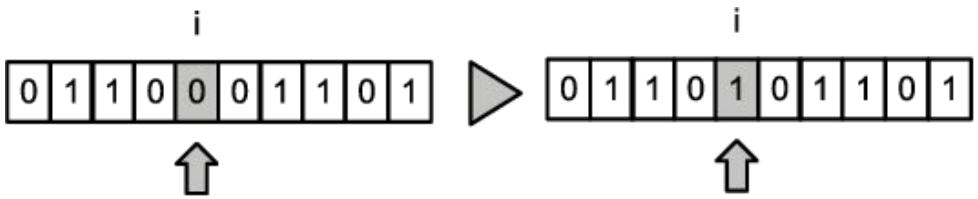
UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Algoritmo Genético:Mutação

- A mutação é feita através de uma variação do método de Bit-Flip-Mutation, onde k bits são escolhidos e se o valor do bit for 0, seu valor se torna 1.

- A refactibilização possui a mesma lógica descrita no crossover.



UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

.inf UFRGS

Algoritmo Genético:Seleção de Novos Indivíduos

- A Seleção de indivíduos da nova População é feita retirando quantos indivíduos novos foram gerados na nova população.

- Como New_p_Size novos indivíduos são gerados por geração através de crossover e mutação, os New_p_Size indivíduos com piores fitness irão sair da população agregada(população atual + novos indivíduos gerados).

Algoritmo Genético:Pseudocódigo

Algorithm 1 Algoritmo Genético

```
1:  populacao ← GeraPopulacaoInicial(Population_Size)
2:  melhor_solucao ← Torneio(populacao, tamanho(populacao))
3:  geracoes_sem_melhora ← 0
4:  while Not CriteriosDeParada() do
5:     nova_populacao ← ∅
6:     while tamanho(nova_populacao) < New_p_Size do
7:        p1 ← Torneio(populacao, k)
8:        p2 ← Torneio(populacao, k)
9:        filho1, filho2 ← Crossover(p1, p2)
10:       filho1 ← Mutacao(filho1, Mutation_Rate)
11:       filho2 ← Mutacao(filho2, Mutation_Rate)
12:       nova_populacao ← nova_populacao ∪ filho1 ∪ filho2
13:    end while
14:    for cromossomo ∈ nova_populacao do
15:       cromossomo ← Refactibiliza(cromossomo)
16:       populacao ← populacao ∪ cromossomo
17:    end for
18:    populacao ← RemovePiores(populacao, New_p_Size)
19:    nova_solucao ← Torneio(populacao, tamanho(populacao))
20:    if nova_solucao ≤ melhor_solucao then
21:       geracoes_sem_melhora ← geracoes_sem_melhora + 1
22:    else
23:       melhor_solucao ← nova_solucao
24:       geracoes_sem_melhora ← 0
25:    end if
26: end while
27: return melhor_solucao
```

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

.inf UFRGS

Algoritmo Genético:Critérios de Parada

O algoritmo para sua execução e retorna a melhor solução encontrada até o momento, caso:

- Max_time seja atingido.
- Max_Non_Improving_Gen seja atingido.

Algoritmo Genético:Cálculo do Fitness de um Cromossomo

- O cálculo do fitness de um cromossomo é feito simplesmente percorrendo a lista com os os genes do cromossomo contando o número de genes iguais a "1", ou seja contando quantas arestas estão presente no emparelhamento representado por esse cromossomo.

Teste de Parâmetros e Operadores

- Nesta seção serão testados os parâmetros e operadores do algoritmo genético nas instâncias RM01 e RM02.

- Os testes foram realizados usando a configuração inicial Population_Size=40,Mutation_Rate=0.9, Max_Non_Improving_Gen=30,Max_Time=1800(seg) e New_p_Size=6.
- Todos os testes foram realizados ao menos utilizando 5 seeds diferentes

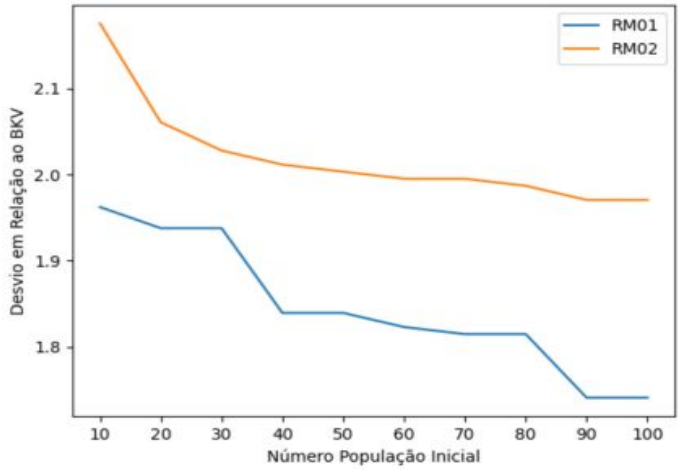
- Para cada teste foram analisados o desvio em relação ao BKV e o tempo de execução médio.O desvio em relação ao BKV foi calculado da seguinte maneira, onde S é a média das soluções encontrados e BKV é o melhor valor conhecido.

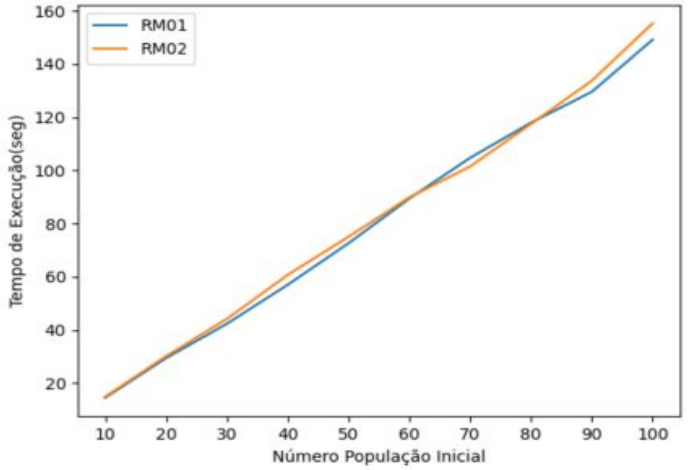
$$\frac{BKV - S}{BKV} \cdot 100$$

- Portanto quanto menor for o desvio, melhor é a média das soluções.

Teste do Parâmetro Population_Size

Para este teste foram avaliados somente o quanto a população inicial influencia na solução inicial, sem os operadores de crossover e mutação.





- Pequena melhora no desvio com aumento da população inicial.
- Aumento quase linear do tempo de execução

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Teste do Operador Crossover

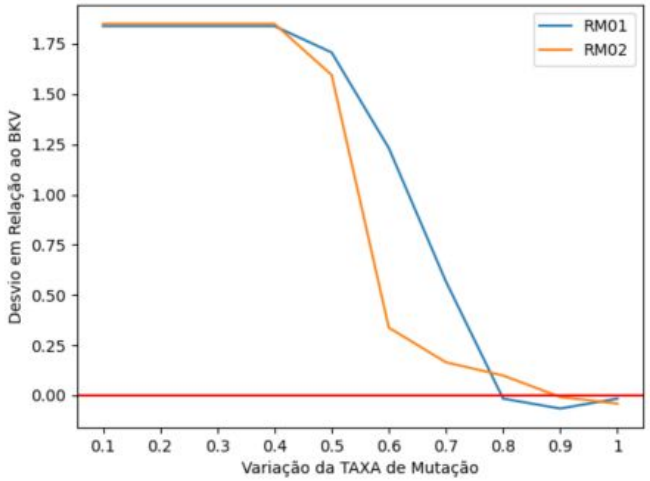
Instância	Valor Médio da Solução Inicial (SI)	Tempo Médio de Execução do SI (segundos)	Valor Médio da Solução com Crossover (SC)	Tempo Médio de Execução do SC (segundos)	Desvio da Solução SC em Relação ao SI(%)	Aumento do Tempo SC em Relação ao SI (%)
RM01	2391	56.5	2391	193.8	0	243.00
RM02	2387	60.4	2387	196.8	0	225.82

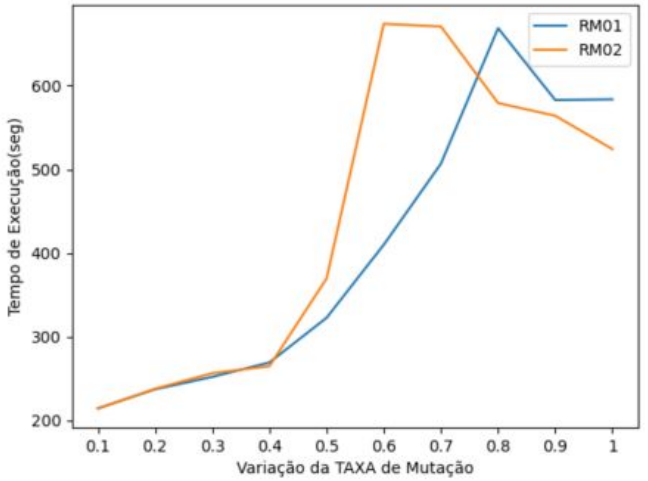
- Apenas o operador crossover não consegue produzir soluções melhores.
- O tempo de execução aumenta consideravelmente.

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Teste do Parâmetro Mutation_Rate



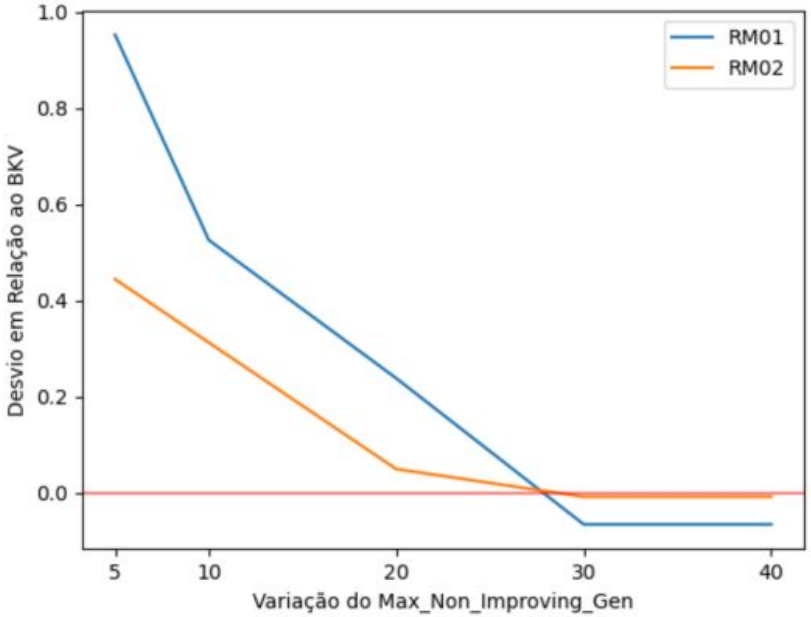


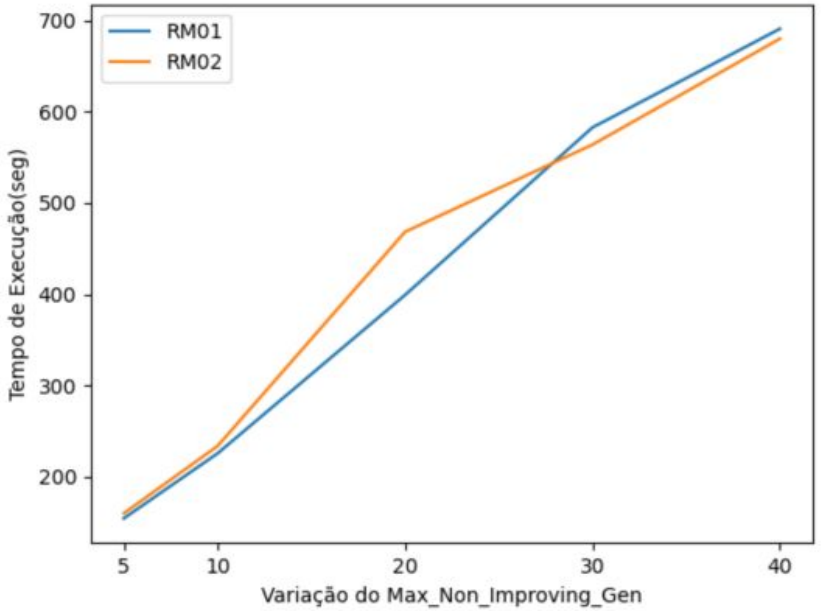
- Desvio começa a diminuir apenas com taxas de mutação maior ou igual à 0.5.
- Melhores soluções entres taxas de 0.8 e 1.
- Alto impacto da taxa de mutação.
- Tempo de execução aumenta com taxas entre 0.1 e 0.8, porém diminui com taxas de 0.9 e 1 em relação ao pico máximo do tempo.

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

.inf UFRGS

Teste do Parâmetro Max_non_gen



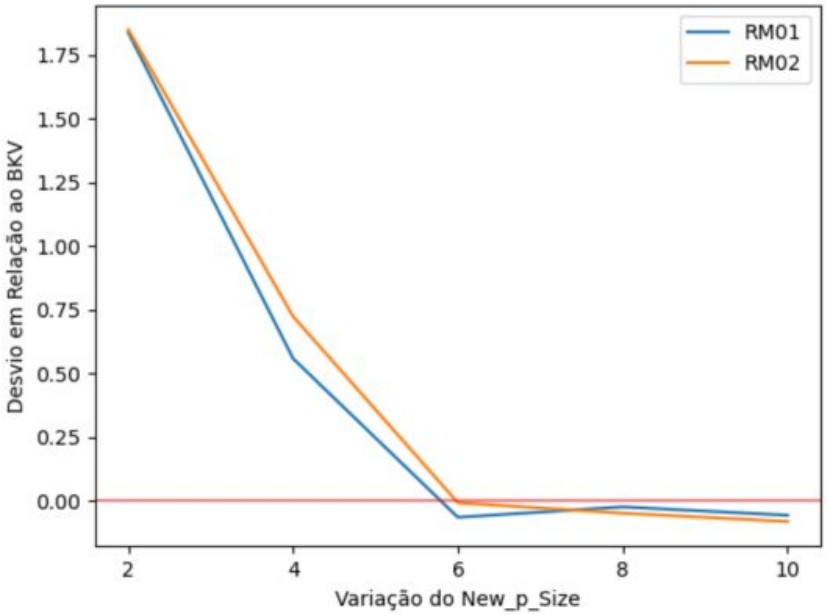


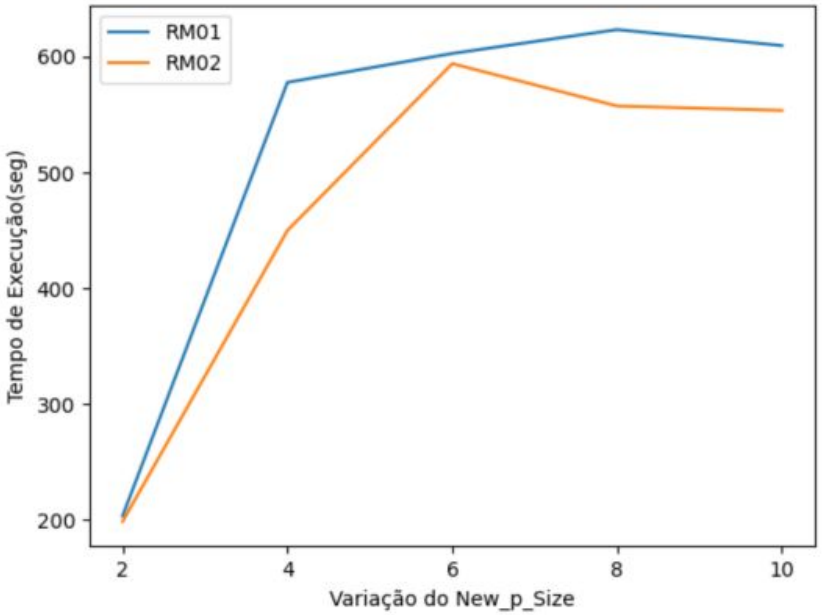
- Desvio diminui com o aumento do número de gerações sem melhora, e estabiliza no valor 30.
- Média de soluções melhores que o BKV para valores maiores que 30
- O tempo de execução aumenta de forma quase linear com o aumento do parâmetro.

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Teste do Parâmetro New_p_Size



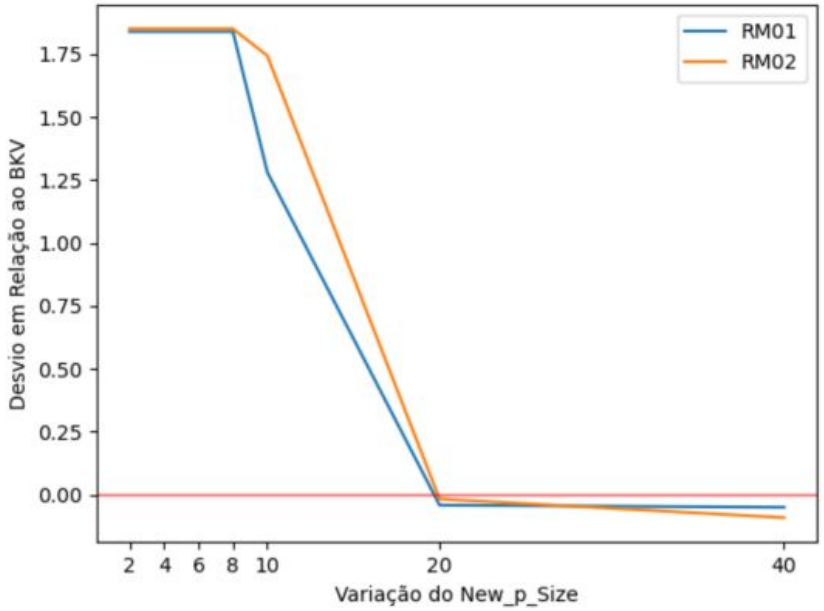


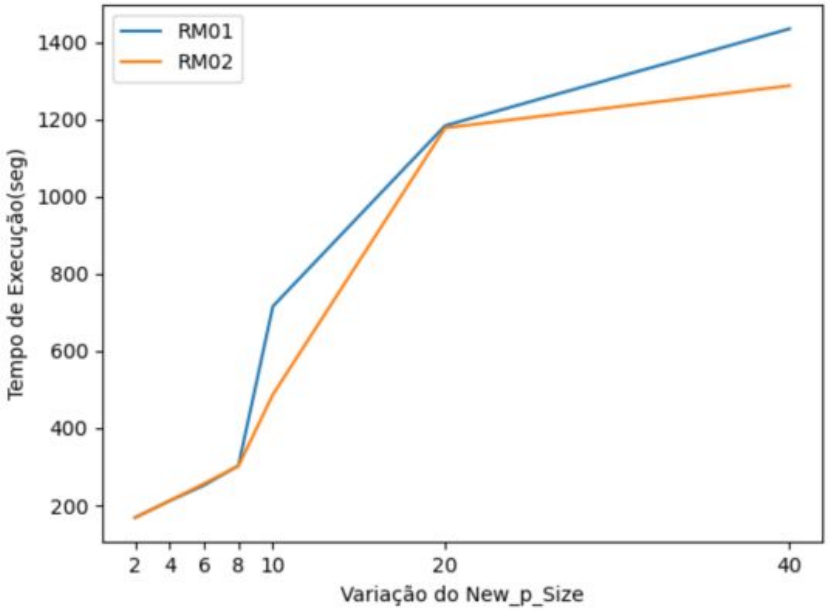
- Desvio diminui com aumento do número de indivíduos gerados, porém valores maiores que 6 possuem valores parecidos, com o valor 10 possuindo as melhores soluções.
- O tempo de execução aumenta até o valor 6,após isso os valores possuem tempos similares.

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Teste do Parâmetro new_p_Size:Mutação Baixa





- Utilizado taxa de mutação=0.3
- A partir de 10 novos indivíduos, o desvio começa a diminuir, com uma queda brusca com valores maiores que 20 indivíduos.
- O tempo de execução é aproximadamente o dobro para soluções melhores que o BKV comparado com a taxa de mutação=0.9

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Teste de Instâncias:Algoritmo Genérico:BKV

Instância	Média do Tempo de Execução (segundos)	Valor Médio da Solução Final (SF)	Desvio da Solução em Relação ao BKV (%)	Melhor Valor Obtido na Solução Final (SF)	Desvio do Melhor Valor em Relação ao BKV (%)
RM01	609	2437	-0.041	2439	-0.123
RM02	553	2434	-0.082	2437	-0.205
RM03	656	2436	0	2436	0
RM04	590	2440	-0.041	2445	-0.246
RM05	647	2436	0	2437	-0.041
RM06	502	2435	-0.164	2439	-0.329
RM07	612	2439	-0.082	2440	-0.123
RM08	540	2435	-0.082	2436	-0.123
RM09	560	2436	0	2437	-0.041
RM10	471	2432	-0.123	2437	-0.329
RM11	575	2434	0	2434	0
RM12	615	2440	0	2441	-0.040

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Teste de Instâncias:Algoritmo Genérico:Solução Inicial

Instância	Valor Médio da Solução Inicial (SI)	Valor Médio da Solução Final (SF)	Desvio de SF em Relação ao SI (%)
RM01	2391	2437	-1.923
RM02	2387	2434	-1.968
RM03	2388	2436	-2.010
RM04	2391	2440	-2.049
RM05	2390	2436	-1.923
RM06	2390	2435	-1.882
RM07	2388	2439	-2.135
RM08	2388	2435	-1.968
RM09	2389	2436	-1.967
RM10	2390	2432	-1.757
RM11	2391	2434	-1.798
RM12	2390	2440	-2.092

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf UFRGS

Conclusão

- Opção Viável
- Pelo menos médias iguais ao BKV para todas 12 instâncias
- Médias maiores que o BKV para 7 das 12 instâncias
- Obteve melhores resultados quando se comportou como uma busca aleatória com elementos e operadores do algoritmo genético
- É possível encontrar valores melhores que o BKV com mutação baixa, porém o tempo de execução aumenta consideravelmente
- Considera-se uma boa abordagem, pois consegue achar bons valores com múltiplas configurações.
- Melhorias podem ser implementadas para diminuir o tempo de execução e o impacto da aleatoriedade nas soluções, principalmente na função de refactibilização.

Obrigado !

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

.inf
INSTITUTO
DE INFORMÁTICA
UFRGS