Introdução à Inteligência Artificial

Trabalho Prático nº2 – Problema de Otimização

```
Nome do Ficheiro: file6.txt

Pretende alterar algum valor?
1 - Populacao 100
2 - Percentagem de mutacao 0.010000
3 - Percentagem de recombinacao 0.700000
4 - Tsize: 2
5 - Maximo de gen 2500
6 - Comecar
Digite a sua opcao: 6


Tipo de Recombinacao:
1 - Recombinacao de 1 corte de ponto
2 - Recombinacao de 2 corte de ponto
Digite a sua opcao: 1


Tipo de Mutacao:
1 - Mutacao Binaria
2 - Mutacao de Inversao
Digite a sua opcao: 1
```

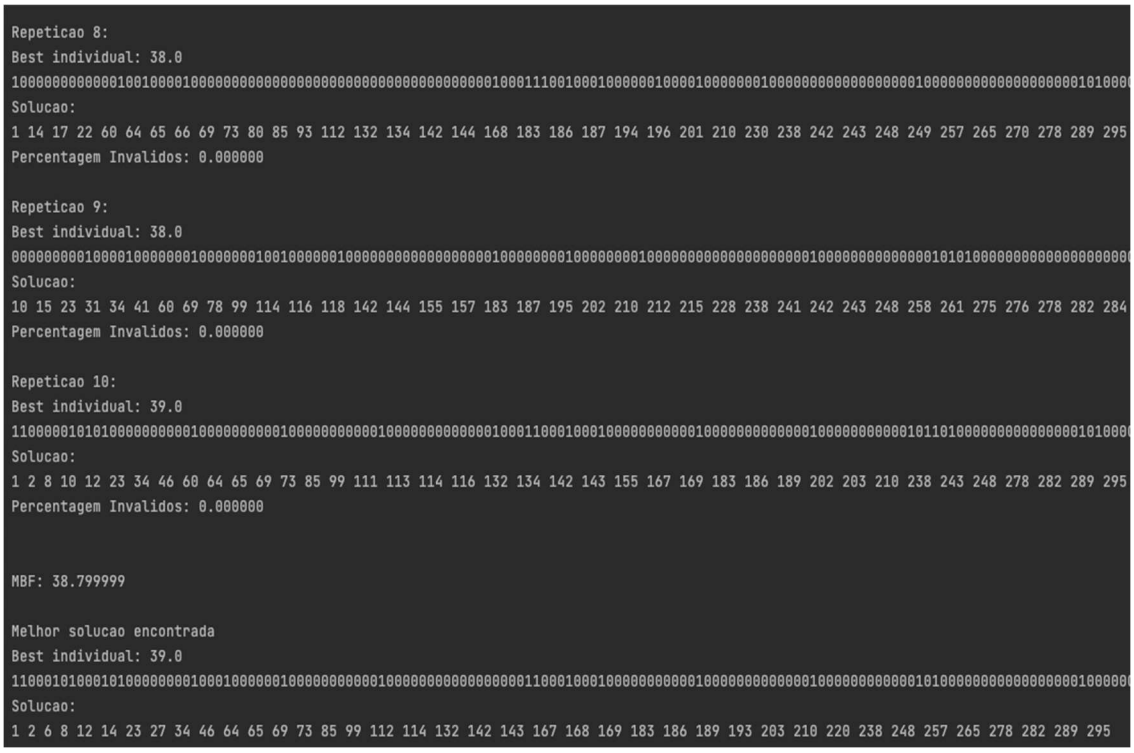
Trabalho Realizado por:
Daniel Fernandes (LEI-PL) - a2020116565@isec.pt
Hugo Jorge (LEI-PL) - a2020116988@isec.pt

Introdução

O seguinte trabalho no âmbito da unidade curricular de Introdução à Inteligência Artificial teve como objetivo desenvolver, analisar e implementar algoritmos de pesquisa local e global. No contexto do mesmo, o trabalho foi implementado na linguagem c.

A implementação consiste em 3 programas. Um programa para o algoritmo de pesquisa local, um para o algoritmo de pesquisa global e por fim um para o algoritmo híbrido que junta os dois algoritmos referidos.

Após a implementação foram realizados testes de modo a analisar os resultados obtidos.



```
Repeticao 8:
Best individual: 38.0
10000000000001001000010000000000000000000000000000000000000001000111001000100000010000100000001000000000000000000100000000000000000001010000
Solucao:
1 14 17 22 60 64 65 66 69 73 80 85 93 112 132 134 142 144 168 183 186 187 194 196 201 210 230 238 242 243 248 249 257 265 270 278 289 295
Percentagem Invalidos: 0.000000

Repeticao 9:
Best individual: 38.0
00000000010000100000000100000001001000000100000000000000000010000000010000000100000000000000000010000000000000101010000000000000000000
Solucao:
10 15 23 31 34 41 60 69 78 99 114 116 118 142 144 155 157 183 187 195 202 210 212 215 228 238 241 242 243 248 258 261 275 276 278 282 284
Percentagem Invalidos: 0.000000

Repeticao 10:
Best individual: 39.0
11000001010100000000010000000001000000000001000000000000010001100010001000000000010000000000001000000000001011010000000000000001010000
Solucao:
1 2 8 10 12 23 34 46 60 64 65 69 73 85 99 111 113 114 116 132 134 142 143 155 167 169 183 186 189 202 203 210 238 243 248 278 282 289 295
Percentagem Invalidos: 0.000000


MBF: 38.799999

Melhor solucao encontrada
Best individual: 39.0
11000101000101000000001000100000100000000001000000000000000011000100010000000000100000000000010000000000010100000000000000001000000
Solucao:
1 2 6 8 12 14 23 27 34 46 64 65 69 73 85 99 112 114 132 142 143 167 168 169 183 186 189 193 203 210 220 238 248 257 265 278 282 289 295
```

1. O Problema

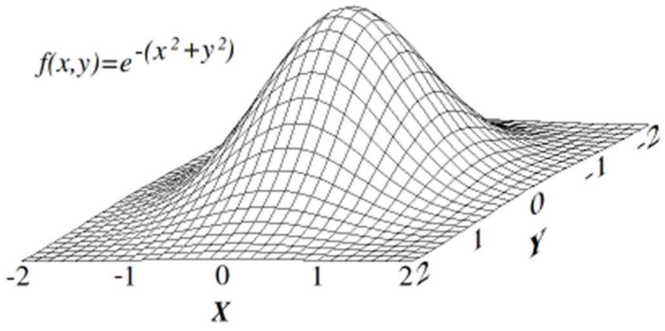
O problema proposto no âmbito do trabalho prático foi um problema de otimização. O problema em si é a pesquisa num grafo de um subconjunto sem qualquer ligação entre si. Para tal foi nos fornecido várias instâncias deste problema. O algoritmo apenas pode lidar com grafos que contenham até 500 vértices. Em cada instância as linhas iniciadas pelo caracter 'c' corresponde a ser linhas de comentário. Após as linhas de comentário surge uma linha iniciada por 'p edge' onde são indicados os vértices e as arestas do grafo.

As arestas são o número de ligações existentes, também correspondente ao número de linhas do ficheiro com ligações entre dois vértices. Cada uma dessas linhas é iniciada com o caracter 'e'.

2. Algoritmo de Pesquisa Local

2.1. Trepa-Colinas first-choice

O trepa-colinas é um algoritmo matemático de otimização que pertence aos algoritmos de pesquisa local. É um algoritmo iterativo que começa com uma solução arbitrária. Este método procura uma melhor solução alterando a solução aleatoriamente e verificando se é melhor que a anterior. Se isso acontecer, guarda essa solução. No algoritmo implementado, é também guardado para análise a média da melhor avaliação (MBF - Mean best fitness), que é a média obtida a partir das melhores soluções encontradas em cada uma das repetições.



2.2. Reparações exploradas

Na resolução do problema aplicamos dois tipos de reparação. A reparação aleatória e a reparação imediata.

Estes métodos são aplicados se for encontrada na solução alguma ligação entre os vértices da solução.

Para a reparação aleatória utilizamos uma função de avaliação da solução e caso a função não seja válida é eliminado um dos vértices aleatoriamente. Isto repete-se até a solução ser válida.

Na reparação imediata o processo é mais imediato pois quando a função de avaliação encontrar uma ligação elimina um dos vértices que tem ligação. Para decidir qual o vértice a ser eliminado é calculado um número aleatório entre 0 e 1. Conforme o número gerado é eliminado o vértice. A função apenas termina quando verificar que toda a solução é válida.

Esta reparação é mais eficaz e rápida pois elimina onde existem ligações, enquanto a reparação aleatória pode remover vértices que não tinham qualquer ligação.

2.3. Vizinhanças exploradas

No algoritmo implementado optamos por duas vizinhanças. As vizinhanças implementadas colocam mais vértices na solução do que retiram porque como as reparações removem vértices da solução, as vizinhanças têm de fazer o contrário para ser possível chegar a uma melhor solução.

A vizinhança 1 apenas adiciona 1 vértice à solução aleatoriamente enquanto a vizinhança 2 retira 1 vértice e adiciona 2 vértices.

2.4. Opções tomadas

Na implementação deste algoritmo optamos por reparar as soluções inválidas para obter mais facilmente soluções ótimas. Sendo algo que tem algum custo em termos de performance, mas nada de relevante. Sendo assim, optamos pelas vizinhanças colocarem mais vértices do que tiram, pois não é possível descobrir qual o número de vértices ideal para a solução inicial.

Caso não o fizéssemos o que referimos a solução iria ficar parada em um número de vértices fixo.

2.5. Análise de testes

Trepa-Colinas com Vizinhança 1

NUM VERT | NUM ARES		100 it	1000 it	5000 it	10000 it
11 | 20	Melhor	5	5	5	5
file1.txt	MBF	3.8	3.96	3.766	3.87
28 | 210	Melhor	7	7	7	7
file2.txt	MBF	6.466	6.733	6.2	6.2
64 | 704	Melhor	12	12	12	12
file3.txt	MBF	9.46	10.40	10.33	10.20
79 | 156	Melhor	34	35	35	36
file4.txt	MBF	28.799	29.63	29.97	30.77
200 | 1534	Melhor	17	17	18	18
file5.txt	MBF	15.87	16.03	16.26	16.3
300 | 10933	Melhor	20	28	30	31
file6.txt	MBF	15.9	23.66	24.7	24.83

Analisando a tabela de dados obtidos, podemos verificar que a partir de um certo número total de vértices, o melhor resultado obtido e o MBF, aumentam proporcionalmente com o número de iterações. Para os números de vértices total mais baixos, o algoritmo implementado atinge a melhor solução com qualquer iteração testada, contudo, verificamos que o MBF se aproxima mais da solução ótima nas 1000 iterações.

Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual

NUM VERT | NUM ARES		100 it	1000 it	5000 it	10000 it
11 | 20	Melhor	4.99	5	5	5
file1.txt	MBF	5	5	5	5
28 | 210	Melhor	7	7	7	7
file2.txt	MBF	6.87	7	7	7
64 | 704	Melhor	12	12	12	12
file3.txt	MBF	10.77	12	12	12
79 | 156	Melhor	35	39	39	39
file4.txt	MBF	30.76	37.79	38.76	38.80
200 | 1534	Melhor	18	18	18	18
file5.txt	MBF	16.56	17.96	18	18
300 | 10933	Melhor	25	37	39	39
file6.txt	MBF	16.6	33	38.06	38.299

Analisando a tabela de dados obtidos, podemos verificar que para um baixo número total de vértices, os resultados obtidos para o melhor valor como para o MBF, não variam consoante o número de iterações testadas, mostrando que sempre chegaram à solução ótima. Em exceção nas 100 iterações, onde podemos verificar ainda presença de alguns resultados obtidos que não chegaram ao ótimo e assim variaram o MBF.

Para um número maior de total de vértices, verificamos que se mantém a proporcionalidade entre a melhor solução encontrada e o número de vezes em que o algoritmo a conseguiu obter, com o aumento das iterações.

Trepa-Colinas com Vizinhança 2

NUM VERT | NUM ARES		100 it	1000 it	5000 it	10000 it
11 | 20	Melhor	5	5	5	5
file1.txt	MBF	4,03	4.26	4.26	4.46
28 | 210	Melhor	6	6	6	6
file2.txt	MBF	4.33	5.97	6	6
64 | 704	Melhor	9	11	11	12
file3.txt	MBF	6.93	9.73	10.4	10.4
79 | 156	Melhor	30	35	36	36
file4.txt	MBF	23.6	31.03	32.97	33.07
200 | 1534	Melhor	16	18	18	18
file5.txt	MBF	14.27	16.5	16.70	16.73
300 | 10933	Melhor	13	22	28	30
file6.txt	MBF	10.3	17.03	23.67	25.43

Analisando a tabela de dados obtidos, verificamos que em todas as instâncias, o algoritmo melhora tanto a solução encontrada como o número de vezes que a consegue encontrar, com o aumento do número de iterações. No entanto, verificamos que nos ficheiros *file2.txt, file4.txt* e *file6.txt*, não consegue atingir o valor ótimo fornecido no estudo.

Trepa-Colinas com Vizinhança 1 com Reparação aleatória

NUM VERT | NUM ARES		100 it	1000 it	5000 it	10000 it
11 | 20	Melhor	5	5	5	5
file1.txt	MBF	4	4,03	4	4.1
28 | 210	Melhor	7	7	7	7
file2.txt	MBF	6,03	6.43	6.46	6.83
64 | 704	Melhor	12	12	12	12
file3.txt	MBF	8.83	10.06	9.77	10.2
79 | 156	Melhor	34	37	36	38
file4.txt	MBF	25.4	28.23	29	27.03
200 | 1534	Melhor	17	17	18	18
file5.txt	MBF	14.46	15.46	15.8	15.8
300 | 10933	Melhor	18	29	30	28
file6.txt	MBF	12.83	21.6	20.2	20.76

Analisando a tabela de dados obtidos, constatamos que, embora o melhor resultado se aproxime da solução ótima, a variação do MBF não segue um padrão definido com a variação das iterações de instância para instância, como nos algoritmos anteriores. Isto deve-se ao facto de a reparação aleatória poder, ou não, reparar uma solução que se iria aproximar do ótimo, mantendo-se assim num ótimo local.

2.6. Conclusão

Ao efetuar uma análise comparativa entre as diferentes variações do algoritmo, concluímos que gerando uma vizinhança de 2, origina uma variação maior de vértices a reparar e portanto, um maior afastamento do máximo global, originando uma maior variação e dificuldade na obtenção da solução ótima.

Com a reparação aleatória, as soluções obtidas para as quais se poderia chegar ao máximo global, podem ser facilmente “destruídas” ao remover um vértice válido. Com isso o algoritmo pode nunca chegar a uma solução ótima.

Por fim, ao aceitar soluções de custo igual ao já obtido, permite ao algoritmo percorrer mais facilmente todo o conjunto de soluções válidas sem ficar “preso” num máximo local o que o leva a obter melhores soluções.

Assim, ao gerar a vizinhança de 1, com aceitação de custo da solução obtida com valor igual ou superior e com um método de reparação imediata, consegue-se obter a melhor variação do algoritmo de pesquisa utilizado.

3. Algoritmo de Pesquisa Global

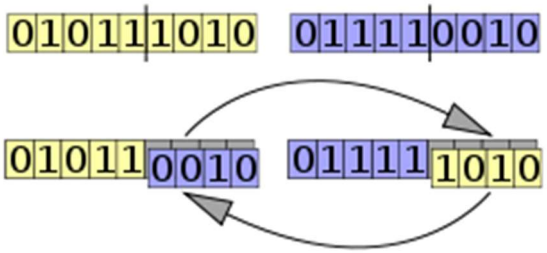
3.1. Evolutivo

Os algoritmos evolucionários são algoritmos de pesquisa global inspirados na evolução biológica. Um conjunto inicial de soluções candidatas é gerado e atualizado iterativamente. A esse conjunto nós chamamos de população. E são aplicados operadores genéticos à população a cada iteração.

3.2. Operadores de Recombinação explorados

No algoritmo do trabalho prático implementámos 2 operadores de recombinação. A recombinação de 1 ponto de corte e a recombinação de 2 pontos de cortes. Na recombinação de 1 ponto de corte é calculado um número aleatório que corresponde ao índice da solução onde será feito o corte e combinamos duas soluções "progenitoras" que vão gerar duas novas soluções da população.

Na recombinação com dois pontos de corte é idêntico à anterior, a diferença é que são gerados dois pontos de corte aleatoriamente. Estes operadores apenas são realizados com uma probabilidade que pode ser alterada no início da execução do algoritmo. E em cada execução apenas é executado um dos operadores, podendo escolher também no início da execução qual o operador pretendido.



3.3. Operadores de Mutação explorados

No algoritmo evolucionário que implementamos são usados dois operadores de mutação.

A mutação binária consiste em inverter alguns vértices da solução com uma taxa de probabilidade. A inversão referida em cima significa retirar vértices ou colocar vértices na solução.

A mutação por inversão consiste em trocar um vértice por o vértice a seguir ao mesmo. Aplica-se também com uma taxa de probabilidade.

Normalmente as probabilidades dos operadores de mutação são muito pequenas de modo a não alterar muito as soluções.

3.4. Opções tomadas

Neste algoritmo optamos como no anterior em reparar as soluções inválidas. Usando os métodos de reparação referidos no algoritmo do trepa-colinas. De modo a tentar obter melhores soluções. Também optamos por fazer uma pequena interface com o utilizador de modo a ser possível modificar os parâmetros em runtime antes de o algoritmo começar.

3.5. Análise de Testes

file1.txt | 11 | 20									
		Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação binária + Reparação2 (Aleatória)		Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação por inversão + Reparação1 (Imediata)	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (ger = 2500)	pr = 0.3	5,0	5,0	5,0	5,0	5,0	5,0	5,0	4,9
pm = 0.01	pr = 0.5	5,0	5,0	5,0	5,0	5,0	5,0	5,0	4,8
tsize = 2	pr = 0.7	5,0	5,0	5,0	5,0	5,0	5,0	5,0	4,9
pop = 100 (ger = 2500)	pm = 0.0	5,0	4,8	5,0	4,8	5,0	5,0	5,0	4,9
pop = 100	pm = 0.001	5,0	5,0	5,0	5,0	5,0	5,0	5,0	5,0
pr = 0.7	pm = 0.01	5,0	5,0	5,0	5,0	5,0	5,0	5,0	5,0
tsize = 2	pm = 0.05	5,0	5,0	5,0	5,0	5,0	5,0	5,0	5,0
pr = 0.3	pop = 10 (ger = 25K)	5,0	5,0	5,0	5,0	5,0	5,0	5,0	4,8
pm = melhor valor obtido (0.001)	pop = 50 (ger = 5K)	5,0	5,0	5,0	5,0	5,0	5,0	5,0	4,4
tsize = 2	pop = 100 (ger = 2.5K)	5,0	5,0	5,0	5,0	5,0	5,0	5,0	4,8

Analisando a tabela de dados obtidos pode se concluir que como a instância tem poucos vértices e ligações o algoritmo chega facilmente à solução ótima mesmo variando os parâmetros, os operadores de recombinação e de mutação. No então destaca-se que com a taxa de mutação a 0 ou com a mutação por inversão o algoritmo tem variância no MBF.

file2.txt | 28 | 210									
		Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação binária + Reparação2 (Aleatória)		Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação por inversão + Reparação1 (Imediata)	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (ger = 2500)	pr = 0.3	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
pm = 0.01	pr = 0.5	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
tsize = 2	pr = 0.7	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
pop = 100 (ger = 2500)	pm = 0.0	7,0	6,8	7,0	6,9	7,0	7,0	7,0	6,8
pop = 100	pm = 0.001	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
pr = 0.7	pm = 0.01	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
tsize = 2	pm = 0.05	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
pr = 0.3	pop = 10 (ger = 25K)	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
pm = melhor valor obtido (0.001)	pop = 50 (ger = 5K)	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0
tsize = 2	pop = 100 (ger = 2.5K)	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0

Analisando a tabela de dados obtidos pode se concluir que como na tabela anterior devido à instância ter poucos vértices e arestas o algoritmo consegue obter em todas as execuções soluções ótimas excepto quando a taxa de mutação é de 0.

Quanto às reparações não é possível distinguir diferenças entre elas nesta instância.

file3.txt | 64 | 704									
		Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação binária + Reparação2 (Aleatória)		Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação por inversão + Reparação1 (Imediata)	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (ger = 2500)	pr = 0.3	12,0	12,0	12,0	12,0	12,0	12,0	12,0	9,5
pm = 0.01	pr = 0.5	12,0	12,0	12,0	12,0	12,0	12,0	12,0	9,4
tsize = 2	pr = 0.7	12,0	12,0	12,0	12,0	12,0	12,0	10,0	9,5
pop = 100 (ger = 2500)	pm = 0.0	9,0	8,9	8,0	6,6	10,0	9,0	11,0	9,0
pop = 100	pm = 0.001	12,0	12,0	12,0	12,0	12,0	12,0	12,0	10,1
pr = 0.7	pm = 0.01	12,0	12,0	12,0	12,0	12,0	12,0	10,0	9,0
tsize = 2	pm = 0.05	11,0	11,0	9,0	8,8	12,0	11,1	10,0	9,9
pr = 0.7	pop = 10 (ger = 25K)	12,0	12,0	12,0	12,0	12,0	12,0	10,0	8,7
pm = melhor valor obtido (0.001)	pop = 50 (ger = 5K)	12,0	12,0	12,0	12,0	12,0	12,0	10,0	9,3
tsize = 2	pop = 100 (ger = 2.5K)	12,0	12,0	12,0	12,0	12,0	12,0	10,0	9,1

A partir da tabela de dados obtida é possível analisar que a taxa de recombinação e o tamanho da população não tem uma grande influência nos resultados. Porém destaca-se que a taxa de mutação influencia drasticamente os resultados.

Pode-se concluir que com uma taxa de mutação elevada ou nula o algoritmo afasta-se das soluções ótimas. Pode-se também concluir que os operadores de recombinação não influenciam os resultados, porém nos operadores de mutação o caso é diferente. A mutação por inversão afasta-se da solução ótima.

Em relação às reparações apenas com taxas de mutação elevadas ou nulas é possível distinguir que a reparação imediata obtém melhores resultados.

file4.txt | 79 | 156									
		Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação binária + Reparação2 (Aleatória)		Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação por inversão + Reparação1 (Imediata)	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (ger = 2500)	pr = 0.3	39,0	38,5	39,0	38,2	39,0	38,6	36,0	33,5
pm = 0.01	pr = 0.5	39,0	38,7	38,0	37,7	39,0	38,0	38,0	34,4
tsize = 2	pr = 0.7	39,0	38,5	39,0	38,1	39,0	38,3	37,0	34,2
pop = 100 (ger = 2500)	pm = 0.0	31,0	29,5	23,0	19,9	34,0	29,1	30,0	28,7
pop = 100	pm = 0.001	39,0	36,5	37,0	32,5	38,0	36,9	37,0	34,6
pr = 0.7	pm = 0.01	39,0	38,5	39,0	37,9	39,0	38,4	38,0	36,1
tsize = 2	pm = 0.05	34,0	32,7	24,0	22,0	33,0	32,1	39,0	35,0
pr = 0.3	pop = 10 (ger = 25K)	39,0	38,5	37,0	35,5	39,0	38,1	33,0	29,2
pm = melhor valor obtido (0.001)	pop = 50 (ger = 5K)	39,0	37,6	36,0	33,0	38,0	37,0	32,0	30,0
tsize = 2	pop = 100 (ger = 2.5K)	38,0	36,6	36,0	32,9	39,0	38,4	36,0	33,7

Analisando a tabela de resultados obtidos podemos concluir que a taxa de recombinação não influencia a obtenção de soluções ótimas. Em relação à taxa de mutação, a taxa de mutação como em casos anteriores quando é elevada ou nula influência negativamente o algoritmo. Ao aumentarmos a quantidade de gerações apesar de diminuirmos o tamanho da população conclui-se que obtém se melhores resultados.

Quanto aos operadores de recombinação não se obteve diferenças notórias entre os dois operadores. No entanto, nos operadores de mutação obteve-se piores resultados com a mutação por inversão. Em relação às reparações é possível distinguir melhorias na reparação imediata.

file5.txt | 200 | 1534									
		Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação binária + Reparação2 (Aleatória)		Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação por inversão + Reparação1 (Imediata)	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (ger = 2500)	pr = 0.3	18,0	18,0	16,0	15,6	18,0	18,0	18,0	17,9
pm = 0.01	pr = 0.5	18,0	18,0	17,0	15,3	18,0	18,0	18,0	18,0
tsize = 2	pr = 0.7	18,0	18,0	16,0	15,0	18,0	18,0	18,0	17,9
pop = 100 (ger = 2500)	pm = 0.0	17,0	16,5	15,0	13,0	17,0	16,7	17,0	16,4
pop = 100	pm = 0.001	18,0	17,8	18,0	17,1	18,0	17,9	18,0	17,9
pr = 0.7	pm = 0.01	18,0	18,0	16,0	15,0	18,0	18,0	18,0	18,0
tsize = 2	pm = 0.05	18,0	18,0	16,0	16,0	18,0	17,8	17,0	15,8
pr = 0.3	pop = 10 (ger = 25K)	18,0	18,0	18,0	18,0	18,0	18,0	18,0	16,8
pm = melhor valor obtido (0.001)	pop = 50 (ger = 5K)	18,0	18,0	18,0	17,3	18,0	18,0	18,0	17,2
tsize = 2	pop = 100 (ger = 2.5K)	18,0	17,9	18,0	17,5	18,0	17,8	18,0	17,2

A partir da tabela de dados obtida é possível analisar que a taxa de recombinação não influencia significativamente os resultados enquanto a taxa de mutação influencia negativamente com valores elevados e nulos. A diminuição da população aumentando o número de gerações melhorou os resultados obtidos.

Nos operadores de recombinação não se encontra uma diferença significativa entre ambas por isso concluímos que nenhum caso em particular obtém resultados melhores que o outro. Em relação aos operadores de mutação podemos concluir como em instâncias anteriores que a mutação por inversão influencia negativamente os resultados. Nas reparações, a reparação imediata obtém melhores soluções do que a reparação aleatória.

file6.txt | 300 | 10933									
		Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação binária + Reparação2 (Aleatória)		Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)		Recombinação de 1 ponto de corte + Mutação por inversão + Reparação1 (Imediata)	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF	Best	MBF	Best	MBF
pop = 100 (ger = 2500)	pr = 0.3	23,0	22,1	14,0	12,9	23,0	22,2	16,0	12,8
pm = 0.01	pr = 0.5	22,0	21,7	14,0	13,0	23,0	21,7	12,0	10,8
tsize = 2	pr = 0.7	23,0	21,6	14,0	12,6	25,0	21,9	14,0	12,0
pop = 100 (ger = 2500)	pm = 0.0	11,0	9,8	14,0	10,6	13,0	11,3	12,0	10,5
pop = 100	pm = 0.001	36,0	32,9	34,0	28,4	36,0	34,5	13,0	11,8
pr = 0.7	pm = 0.01	23,0	21,3	13,0	12,3	23,0	21,7	14,0	12,1
tsize = 2	pm = 0.05	18,0	17,3	13,0	11,7	17,0	17,0	14,0	12,1
pr = 0.3	pop = 10 (ger = 25K)	39,0	37,8	36,0	32,5	39,0	37,5	11,0	9,6
pm = melhor valor obtido (0.001)	pop = 50 (ger = 5K)	38,0	35,9	33,0	29,6	38,0	35,6	14,0	10.7
tsize = 2	pop = 100 (ger = 2.5K)	38,0	35.2	33,0	29,4	38,0	35,4	14,0	10,9

Analisando a tabela de resultados obtidos podemos concluir que a taxa de recombinação de 0.3 melhora a obtenção em média de resultados. Já a taxa de mutação pode-se concluir que a taxa de 0.001 melhora significativamente comparando com as outras taxas. Ao diminuir a população e aumentando o número de gerações, o algoritmo obtém melhores soluções.

Nos operadores de recombinação, o operador de 2 pontos de corte obtém melhores resultados, mas com uma diferença pouco significativa. Nos operadores de mutação destaca-se a mutação por inversão na negativa pois os resultados ficam muito aquém dos resultados da mutação binária.

Em relação às reparações, a imediata obtém melhores resultados na maioria dos casos.

3.6. Conclusão

Ao efetuar uma análise comparativa entre os resultados obtidos das diferentes instâncias pode-se concluir que não se verificou nenhuma diferença significativa entre os operadores de recombinação. No entanto, nos operadores de mutação houve uma diferença considerável. Em média, a mutação binária obteve melhores proveitos em todas as instâncias. No contexto das reparações, a reparação imediata obteve melhores resultados que a reparação aleatória e mostrou-se computacionalmente mais eficiente.

Nas variações das percentagens de recombinação e mutação, em alguns casos a percentagem de 0.3 obteve melhores resultados, mas não muito significantes enquanto na percentagem de mutação, a percentagem de 0.001 na maioria dos casos obteve resultados superiores em que em alguns se destacou consideravelmente. Ao diminuir a

população e aumentando o número de gerações, em alguns casos surgiram proveitos superiores.

4. Algoritmo Híbrido

4.1. Os Algoritmos Híbridos

Muitos dos problemas complexos podem ser resolvidos utilizando um conjunto de vários algoritmos. O algoritmo híbrido em que nos baseamos implementa um algoritmo de pesquisa local e um algoritmo de pesquisa global. Para tal foram implementados os dois algoritmos referidos anteriormente num só. O algoritmo sem si é o algoritmo evolutivo com recombinação de um ponto de corte, mutação binária e reparação imediata, complementado com a utilização do algoritmo trepa-colinas em alguns sítios da sua execução.

4.2. Algoritmo Híbrido 1

No algoritmo híbrido 1 o algoritmo de pesquisa local é usado para melhorar as soluções da população inicial, ou seja, após serem criadas as soluções iniciais é chamado o trepa-colinas com uma probabilidade de ser chamado.

4.3. Algoritmo Híbrido 2

No algoritmo híbrido 2 o algoritmo de pesquisa local é usado para refinar as soluções da última população, ou seja, antes de imprimir a melhor solução do algoritmo evolutivo é chamado o trepa-colinas para melhorar as soluções da última população.

4.4. Opções tomadas

Neste algoritmo optamos por não usar a pesquisa local durante o percorrer das gerações porque ficaria muito pesado computacionalmente.

4.5. Análise de testes

Com a conclusão dos testes dos algoritmos híbridos escolhidos, foram feitas tabelas resumo com os melhores resultados obtidos para cada uma das instâncias, e qual o método utilizado.

file1.txt | 11 | 20					
		Algoritmo híbrido 1		Algoritmo híbrido 2	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF
pop = 10 (gen = 25k) pr = 0.3	PROBGERAVIZ = 1	5,0	5,0	5,0	5,0
pm = 0.001 tsize = 2	PROBGERAVIZ = 0.8	5,0	5,0	5,0	5,0

Melhor método	Best	MBF	Método utilizado
Pesquisa Local	5	3,87	Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual
Evolutivo	5	5	Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)
Abordagem Híbrida 1	5	5	Ambas as probabilidades
Abordagem Híbrida 2	5	5	Ambas as probabilidades

file2.txt | 28 | 210					
		Algoritmo híbrido 1		Algoritmo híbrido 2	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF
pop = 10 (gen = 25k) pr = 0.3	PROBGERAVIZ = 1	7,0	7,0	7,0	7,0
pm = 0.001 tsize = 2	PROBGERAVIZ = 0.8	7,0	7,0	7,0	7,0

Melhor método	Best	MBF	Método utilizado
Pesquisa Local	7	7	Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual
Evolutivo	7	7	Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)
Abordagem Híbrida 1	7	7	Ambas as probabilidades
Abordagem Híbrida 2	7	7	Ambas as probabilidades

file3.txt | 64 | 704					
		Algoritmo híbrido 1		Algoritmo híbrido 2	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF
pop = 10 (gen = 25k) pr = 0.3	PROBGERAVIZ = 1	12,0	12,0	12,0	12,0
pm = 0.001 tsize = 2	PROBGERAVIZ = 0.8	12,0	12,0	12,0	12,0

Melhor método	Best	MBF	Método utilizado
Pesquisa Local	12	12	Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual
Evolutivo	12	12	Recombinação de 2 ponto de corte + Mutação binária + Reparação1 (Imediata)
Abordagem Híbrida 1	12	12	Ambas as probabilidades
Abordagem Híbrida 2	12	12	Ambas as probabilidades

file4.txt | 79 | 156					
		Algoritmo híbrido 1		Algoritmo híbrido 2	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF
pop = 10 (gen = 25k) pr = 0.3	PROBGERAVIZ = 1	39,0	39,0	39,0	39,0
pm = 0.01 tsize = 2	PROBGERAVIZ = 0.8	39,0	39,0	39,0	39,0

Melhor método	Best	MBF	Método utilizado
Pesquisa Local	39	38,8	Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual
Evolutivo	39	38,7	Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)
Abordagem Híbrida 1	39	39	Ambas as probabilidades
Abordagem Híbrida 2	39	39	Ambas as probabilidades

file5.txt | 200 | 1534					
		Algoritmo híbrido 1		Algoritmo híbrido 2	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF
pop = 10 (gen = 25k) pr = 0.7	PROBGERAVIZ = 1	18,0	18,0	18,0	18,0
pm = 0.01 tsize = 2	PROBGERAVIZ = 0.8	18,0	18,0	18,0	18,0

Melhor método	Best	MBF	Método utilizado
Pesquisa Local	18	18	Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual
Evolutivo	18	18	Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)
Abordagem Híbrida 1	18	18	Ambas as probabilidades
Abordagem Híbrida 2	18	18	Ambas as probabilidades

Analisando as tabelas de resultados obtidos da execução dos algoritmos híbridos, verificamos que desde o *file1.txt* ao *file5.txt*, ambos algoritmos atingem o máximo global, nas duas variações da probabilidade de gerar vizinho.

file6.txt | 300 | 10933					
		Algoritmo híbrido 1		Algoritmo híbrido 2	
Parâmetros Fixos	Parâmetros a variar	Best	MBF	Best	MBF
pop = 10 (gen = 25k) pr = 0.3	PROBGERAVIZ = 1	39,0	39,0	39,0	38,8
pm = 0.001 tsize = 2	PROBGERAVIZ = 0.8	39,0	39,0	39,0	38,7

Melhor método	Best	MBF	Método utilizado
Pesquisa Local	39	38,3	Trepa-Colinas com Vizinhança 1 e aceitando soluções de custo igual
Evolutivo	39	37,8	Recombinação de 1 ponto de corte + Mutação binária + Reparação1 (Imediata)
Abordagem Híbrida 1	39	39	Ambas as probabilidades
Abordagem Híbrida 2	39	38,8	Algoritmo híbrido 2 com PROBGERAVIZ = 1

No *file6.txt* pode-se verificar que quando o trepa-colinas é executado para refinar a solução obtida do algoritmo híbrido, nem sempre as soluções chegaram ao máximo global embora tenha conseguido atingir o máximo. Para além disso, verifica-se que com 100% de probabilidade de gerar vizinho, este tem um resultado ligeiramente melhor que a probabilidade de 80%.

4.6. Conclusão

Na generalidade das instâncias, o facto dos algoritmos híbridos chegarem às soluções ótimas referenciadas no material fornecido, deve-se ao facto de uma vez que na execução do primeiro algoritmo este já se encontrar próximo da solução ótima, a execução do segundo algoritmo irá refinar e alcançar a solução obtida pelo primeiro sendo muito mais fácil concretizar a chegada a esse valor.

5. Conclusões Gerais

Após a realização e análise dos testes em cima mencionados, verificamos que com o algoritmo híbrido, ou seja, a combinação de um algoritmo de pesquisa local com um algoritmo de pesquisa global obtém-se valores superiores a usar os algoritmos individualmente. Também verificamos que a função de reparação é crucial para a resolução do problema. Pois inicialmente experimentamos penalizar as soluções inválidas, porém nas instâncias com mais arestas e vértices ocorria que algumas vezes os algoritmos não obtinham uma única solução válida.

No algoritmo de pesquisa local trepa-colinas percebemos que quanto mais iterações realizar mais se aproxima da solução ótima e que aceitando soluções de custo igual também otimiza a aproximação à solução ótima.