PROJETO – RECOLHA DE PRODUTOS EM ARMAZÉNS

Alexandre Cardeira
2213107

Miguel Alberto
2211444

ABSTRACT
Neste relatório, discutimos os principais desenvolvimentos para poder resolver o problema de recolha otimizada de produtos de prateleiras de um armazém.

1. INTRODUÇÃO
Os capítulos seguintes vão focar em pontos importantes para o desenvolvimento de todo o projeto. Podemos dividir este projeto em três grandes fases: Uma fase inicial, em que é usado o algoritmo de procura A* para calcular os caminhos ideais, uma segunda fase onde são aplicados os algoritmos genéticos com base nos caminhos ideais calculados na fase anterior, e onde são calculados os percursos mais otimizados possível para cada forklift. Por último temos a fase de realização de experiências, onde são feitos testes com vários datasets com diferentes parâmetros e são, posteriormente, discutidos e comparados os resultados obtidos.

2. Estado do problema
Chamamos a um estado do problema a uma configuração de todos os elementos num dado ambiente num certo instante.

No nosso caso, o estado representa o tamanho do espaço do problema (matriz) e as posições, num dado instante, dos forklifts e dos restantes objetos imóveis (saída, produtos e prateleiras). Isto permite-nos saber, para um forklift e em cada instante, quais são os movimentos permitidos (Movimento para a esquerda, direita, cima e baixo). Resumidamente, os obstáculos de um forklift são os próprios limites do espaço (matriz), os produtos e as prateleiras.

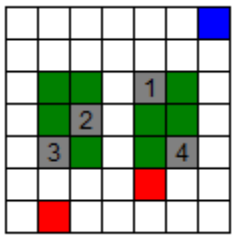

Figura 1 - Exemplo de representação de um estado do problema

Na figura 1 temos, como exemplo, uma representação gráfica de um estado inicial do nosso problema, as quadrículas vermelhas correspondem a forklifts, as quadrículas verdes a prateleiras vazias, as quadrículas cinzentas a produtos por recolher, as quadrículas brancas a espaço livre para circulação dos forklifts e a quadrícula azul corresponde à posição de saída.

3. Heurística
A heurística é uma função que estima o custo (distância no nosso caso) do caminho a partir de um certo estado para um estado objetivo.
No nosso caso, achámos que a melhor heurística seria determinar uma estimativa da distância entre a posição atual do forklift e a posição objetivo, onde esta é calculada como a soma das diferenças absolutas entre as linhas e colunas da posição de objetivo e da posição atual do forklift. Temos um exemplo disso na Figura 2.

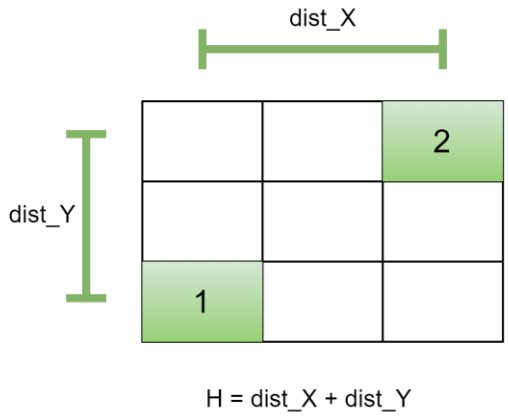

Figura 2 - Exemplo do cálculo de heurística entre duas células

É uma heurística que garante não sobrestimar o custo entre pontos, é também simples e eficiente em termos de execução devido ao facto de considerar apenas movimentos em linha reta.

4. Representação de indivíduos

Um indivíduo representa uma potencial solução para o problema. Posto isto, achámos que era fundamental esta representação focar nestes aspetos: Divisão de produtos por forklift e, por cada forklift, a ordem de recolha destes produtos.

A forma mais simples que conseguimos obter, foi através de um vetor de inteiros, preenchido com os números de todos os objetos (1:n) e com números a partir de “n+1” para representar os forklifts.

A forma estes números estão ordenados permite-nos saber os dois aspetos falados acima.

No entanto, o “último” forklift nunca é representado no vetor.

Exemplo: 5 Produtos e 2 Forklifts

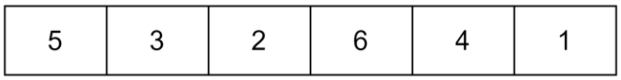

Figura 3 - Exemplo de uma representação de um indivíduo

Pegando no exemplo mostrado na Figura 1, os 5 produtos estão representados com os números de 1 a 5. Neste caso, o forklift representado neste vetor é apenas o número 6.

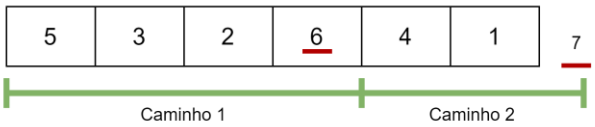

Figura 4 - Divisão do vetor em caminhos

Com isto, podemos retirar dois caminhos (Figura 2). Um primeiro caminho, em que o forklift 6 irá apanhar os objetos 5, 3 e 2 – por esta ordem, e um segundo caminho, em que o forklift que ficou de fora, irá apanhar os objetos 4 e 1 – por essa ordem:

- F6 – 5 – 3 – 2 – S
- F7 – 4 – 1 – S

De notar que no fim de apanharem todos os produtos, os forklifts devem de se dirigir para a Saída (S).

5. Função de fitness

A função de fitness tem por objetivo calcular um valor numérico representativo da qualidade de um indivíduo como solução ao problema.

Para o nosso caso, faz sentido que a qualidade de um indivíduo seja tão melhor quanto em menos tempo a recolha de todos os produtos for. Para tal, primeiramente, após calcular as distâncias percorridas por todos os forklifts, considerou-se apenas a distância maior entre todos os forklifts para o fitness, pois este é que é o último a chegar à saída.

No entanto, depois de se ter feito alguns testes, decidimos considerar também na função de fitness a soma de todas as distâncias percorridas pelos forklifts. Achámos vantajoso que a distância geral percorrida por todos os forklifts fosse o mais pequena possível.

6. Criação de população inicial

A população inicial consiste no conjunto inicial de indivíduos candidatos para a solução.

Optámos pela criação totalmente aleatória de todos os indivíduos. Em termos mais práticos, foram colocados todos os produtos e forklifts (menos um) no vetor de inteiros e depois foi executada a função "random.shuffle()" para variar todas as posições do vetor aleatoriamente.

7. Operadores genéticos desenvolvidos

7.1 Recombinação

A recombinação é um operador genético que permite, durante a criação de novos indivíduos, que estes herdem a composição de indivíduos existentes (os seus "pais") e troca, em parte, a composição de um desses indivíduos pela composição do outro. O indivíduo resultante herda, assim, informação genética de ambos os seus "pais". Criámos 2 operadores genéticos de recombinação:

recombination2
Neste operador genético são formadas 4 "partições" aleatórias com os genes de cada um dos dois indivíduos pai. Estas partições não se sobrepõem, e juntas contêm a informação genética total do indivíduo. O novo indivíduo gerado - a child - herda de um dos pais a partição 1 e 3. Do outro, a partição 2 e 4. No entanto, para prevenir repetições de genes (que para este problema não podem ser repetidos – cada gene é um produto ou forklift diferente), são trocados genes que estejam repetidos pelos genes do outro pai, da mesma posição. No final, caso ainda existam genes repetidos, um deles é substituído por um gene que ainda esteja em falta no genoma.

recombination3
Neste operador genético a child herda o genoma de um dos pais, mas depois recebe alguns dos forklifts do outro pai, na mesma posição do seu genoma. O gene desta posição é substituído pelo forklift recebido. E o forklift pré-existente, que seja idêntico ao forklift recebido, é substituído pelo gene.

7.2 Mutação

A mutação é um operador genético, geralmente utilizado com probabilidades mais baixas, que em vez de trocar genes entre 2 indivíduos, troca ou modifica apenas genes dentro do mesmo indivíduo. Criámos 2 operadores genéticos de mutação:

mutation2
Neste operador genético é escolhido ao acaso um intervalo de valores seguidos, no vetor que compõe o genoma. Os genes deste intervalo são então revertidos (se tínhamos a sequência "123", teremos "321").

mutation3
Neste operador genético são simplesmente escolhidos 2 genes, que são depois trocados (de posição) entre si.

8. Resultados – discussão

Foram realizados testes (experiments) com 2 datasets (dataset 2 e dataset 3) utilizando todos os tipos de recombinações e mutações do projeto (com diferentes probabilidades), população de 100, 200 e 400 e número de gerações de, igualmente, 100, 200 e 400. O tamanho do tournament foi de 2, 4 e 10. Os valores das barras dos gráficos correspondem à média global de cada atributo. Quanto menor o fitness melhor deverá ser o "run", uma vez que terá menos passos totais e menos passos do pior forklift contabilizados, que corresponde a um menor tempo para a resolução do problema

8.1 Dataset 2

Começando pela população e número de gerações, temos o seguinte gráfico:

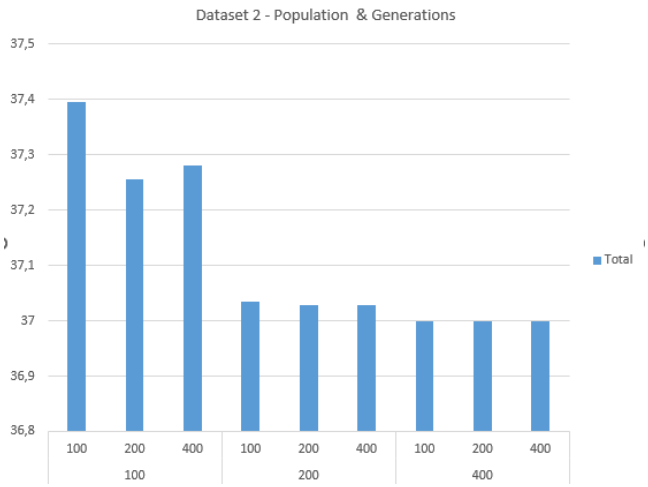

Figura 5 - Gerações (números de cima) sobre População (números de baixo)

Como seria de esperar, quanto maior a população melhores resultados são obtidos. No entanto a média total dos valores não difere muito uns dos outros.

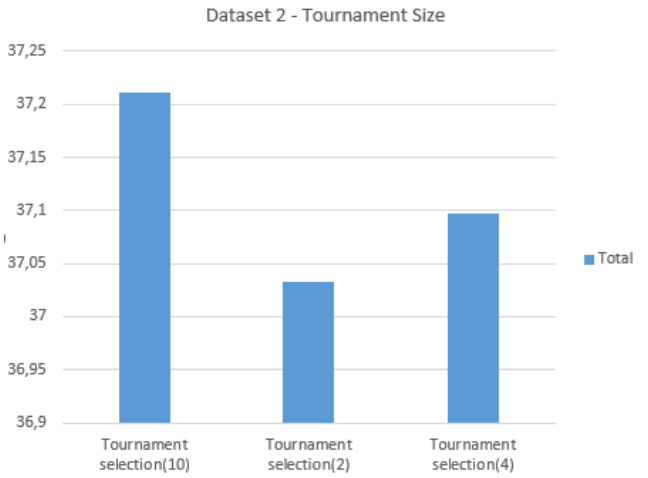

Figura 6 - Tamanho do Torneio

Por pequena diferença, o melhor tamanho de torneio foi o de 2.

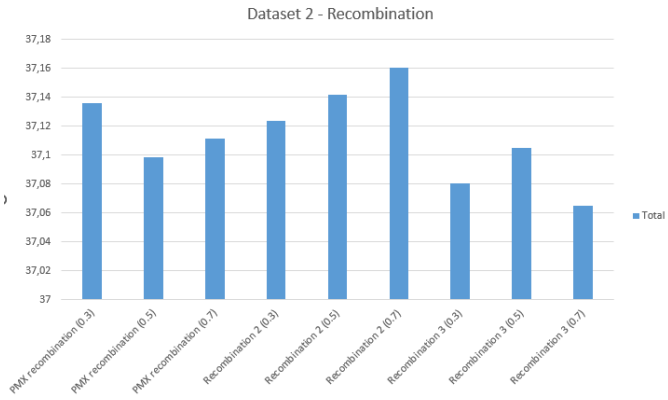

Figura 7 - Operadores de Recombinação com diferentes probabilidades de ocorrência

Olhando para os valores totais da média, os operadores genéticos de recombinação obtiveram resultados muito pouco diferentes uns dos outros, com o mais bem sucedido sendo o Recombination3.

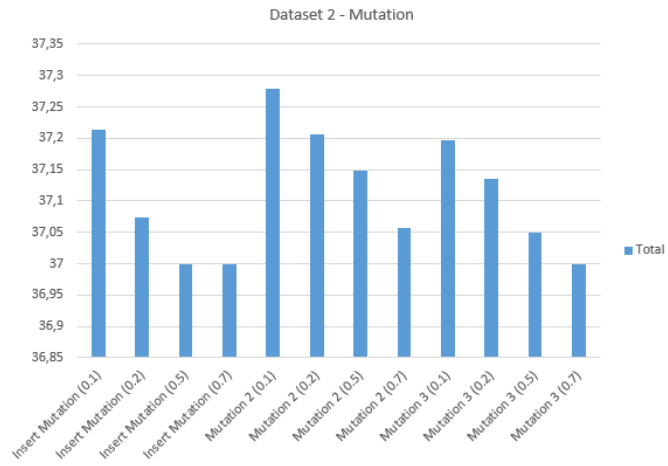

Figura 8 - Operadores de Mutação com diferentes probabilidades de ocorrência

Igualmente aos gráficos anteriores, com a média do fitness a rondar os 37, pouca diferença houve com os operadores genéticos de mutação. O Insert Mutation foi o operador com melhores resultados, e para todos os casos, a probabilidade mais elevada foi a que teve melhores resultados, apesar de ser esperado o oposto.

```
Population size: 100

Max generations: 100

Selection: Tournament selection(2)

Recombination: PMX recombination (0.7)

Mutation:Insert Mutation (0.1)

Fitness: 37
[4 5 3 2 1]

Forklifts path:
[5,4] -> [4,5] -> S
Steps: 7

[6,1] -> [4,1] -> [3,2] -> [2,4] -> S
Steps: 15
```

Figura 9 - Apresentação de uma das melhores soluções encontradas

Quase todas as experiências chegaram à mesma solução ideal - [6,1] -> [4,1] -> [3,2] -> [2,4] -> S – com 15 passos.

Podemos concluir com as experiências ao Dataset 2 que os diferentes valores dados aos "parâmetros" estudados (população, gerações, operadores genéticos, torneio) influenciam ligeiramente os resultados obtidos. Para este dataset, estes valores tiveram diferenças ligeiras provavelmente porque o próprio dataset é pequeno. É um problema com poucos produtos, poucos forklifts, e por sua vez, relativamente poucas soluções possíveis. Veremos de seguida, para um dataset mais complexo, se estes parâmetros terão maior influência.

8.2 Dataset 3

Começando pela população e número de gerações, temos o seguinte gráfico:

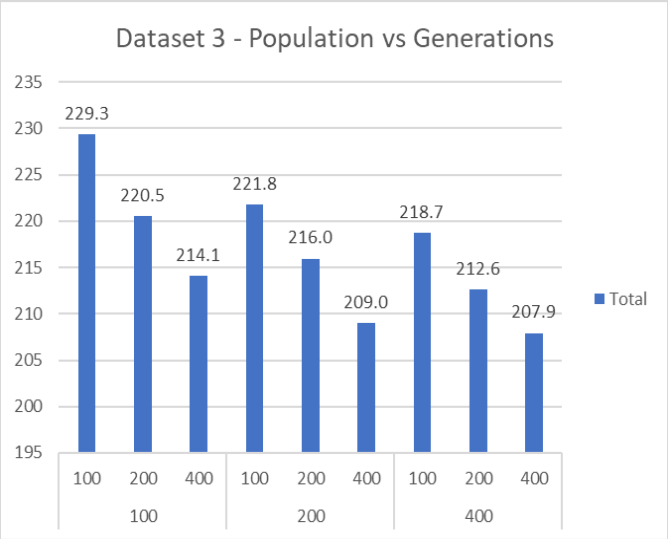

Figura 10 - Gerações (números de cima) sobre População (números de baixo)

Como seria de esperar, quanto maior a população e maior o número de gerações melhores resultados são obtidos.

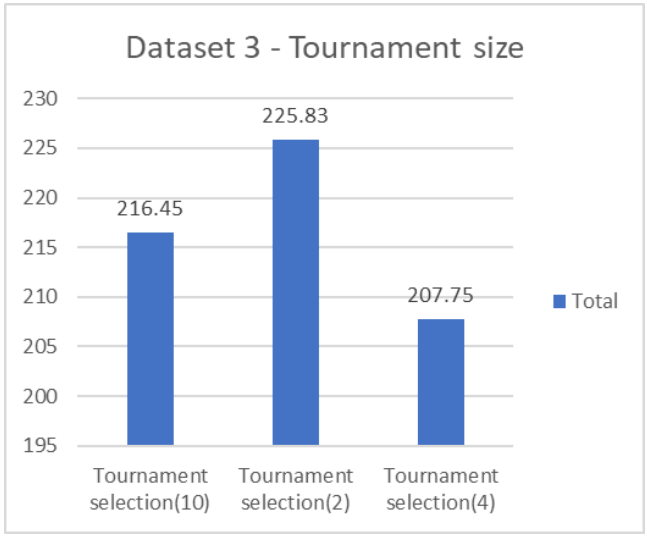

Figura 11 - Tamanho do torneio

Para este dataset, o melhor tamanho de torneio foi o de 4.

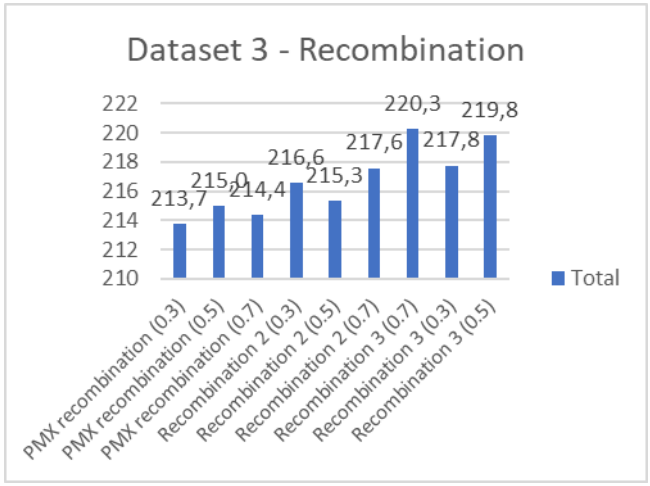

Figura 12 - Operadores de Recombinação com diferentes probabilidades de ocorrência

Para a Recombinação, as médias totais de fitness apresentam menor variação. Apesar de tudo, a Recombination_PMX foi a melhor sucedida, seguida da Recombination2. Para todos os casos, a probabilidade maior (0.7) obteve piores resultados, algo que não era esperado, pois a recombinação costuma ser utilizada com probabilidades relativamente altas. No entanto, variação dos resultados não foi tão elevada como para outros parâmetros (indo apenas de 213.7 a 219.8).

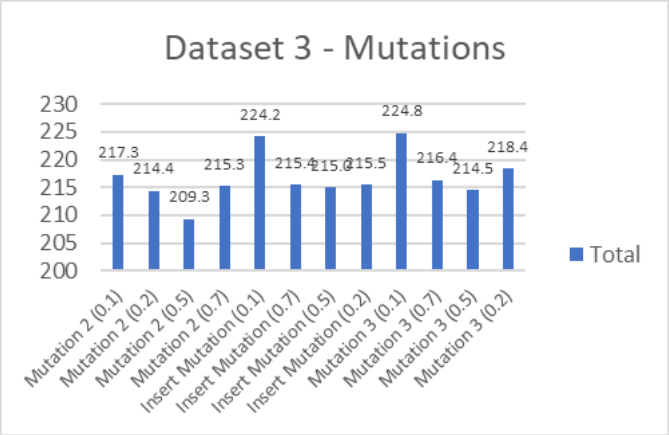

Figura 13 - Operadores de Mutação com diferentes probabilidades de ocorrência

Para a os operadores genéticos de Mutação, a Mutation2 foi a que obteve melhores resultados. Em todos os operadores, a probabilidade menor (0.1) foi a pior. Possivelmente, a Mutation2 que apenas faz a troca entre 2 produtos foi a melhor sucedida por ser menos evasiva, não prejudicando tanto cada geração e ao mesmo tempo abrindo possibilidades dos forklifts encontrarem um caminho melhor.

```
Population size: 200

Max generations: 200

Selection: Tournament selection(4)

Recombination: Recombination 2 (0.7)

Mutation:Mutation 3 (0.7)

Fitness: 160
[5, 11, 18, 20, 17, 21, 6, 2, 1, 3, 7, 9, 19, 22, 14, 13, 10, 8, 4, 12, 15, 16]

Forklifts path:
[3,13] -> [3,15] -> [9,18] -> [15,20] -> [16,15] -> [15,17] -> S
Steps: 42

[9,1] -> [4,2] -> [2,2] -> [1,0] -> [2,8] -> [4,6] -> [9,6] -> [16,3] -> S
Steps: 42

[16,10] -> [15,11] -> [14,9] -> [9,9] -> [8,11] -> [3,11] -> [11,14] -> [15,12] -> [15,14] -> S
Steps: 34
```

Figura 14 - Apresentação de uma das melhores soluções encontradas

```
Population size: 400

Max generations: 400

Selection: Tournament selection(2)

Recombination: PMX recombination (0.5)

Mutation:Mutation 2 (0.1)

Fitness: 160
[5, 11, 18, 20, 17, 21, 6, 2, 1, 3, 7, 9, 19, 22, 14, 13, 10, 8, 4, 12, 16, 15]

Forklifts path:
[3,13] -> [3,15] -> [9,18] -> [15,20] -> [16,15] -> [15,17] -> S
Steps: 42

[9,1] -> [4,2] -> [2,2] -> [1,0] -> [2,8] -> [4,6] -> [9,6] -> [16,3] -> S
Steps: 42

[16,10] -> [15,11] -> [14,9] -> [9,9] -> [8,11] -> [3,11] -> [11,14] -> [15,14] -> [15,12] -> S
Steps: 34
```

Figura 15 - Apresentação de uma das melhores soluções encontradas

O melhor caminho encontrado foi

[3,13] -> [3,15] -> [9,18] -> [15,20] -> [16,15] -> [15,17] -> S

[9,1] -> [4,2] -> [2,2] -> [1,0] -> [2,8] -> [4,6] -> [9,6] -> [16,3] -> S

[16,10] -> [15,11] -> [14,9] -> [9,9] -> [8,11] -> [3,11] -> [11,14] -> [15,14] -> [15,12] -> S

com uma variação nestes 2 últimos passos que podem alternar entre si ([15,14] e [15,12]).

Podemos concluir com as experiências ao Dataset 3 que os diferentes valores dados aos "parâmetros" estudados (população, gerações, operadores genéticos, torneio) influenciam os resultados obtidos.

8.3 Dataset 4

Para o dataset 4, na população e número de gerações, temos o seguinte gráfico:

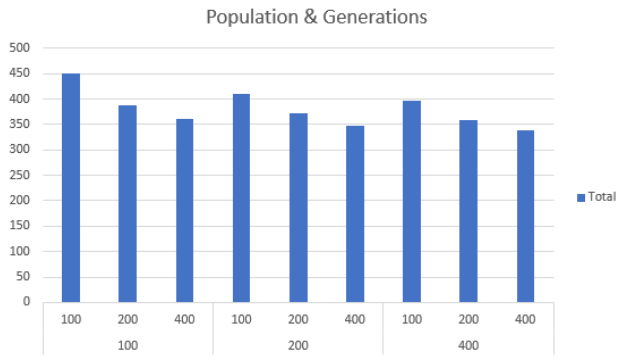

Figura 16 - Gerações (números de cima) sobre População (números de baixo)

Tal como visto no dataset anterior, e como seria de esperar, quanto maior a população e maior o número de gerações melhores resultados são obtidos.

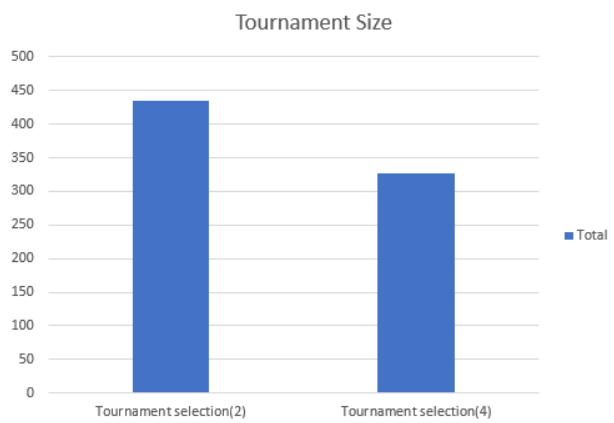

Figura 17 - Tamanho do Torneio

Para este dataset, o melhor tamanho de torneio foi também o de 4.

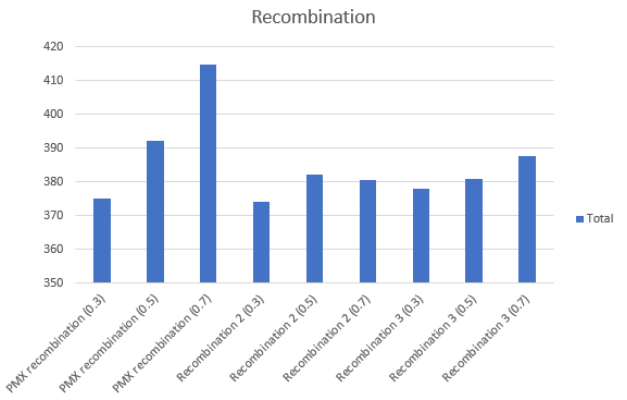

Figura 18 - Operadores de Recombinação com diferentes probabilidades de ocorrência

Na Recombinação obtivemos os melhores valores com o operador Recombination2, seguido do Recombination3 (exceptuando com a probabilidade de 0.3). Em especial para a Recombination_PMX, quanto menor a probabilidade de ocorrência melhor foi o resultado. Esta tendência também se verificou no dataset anterior.

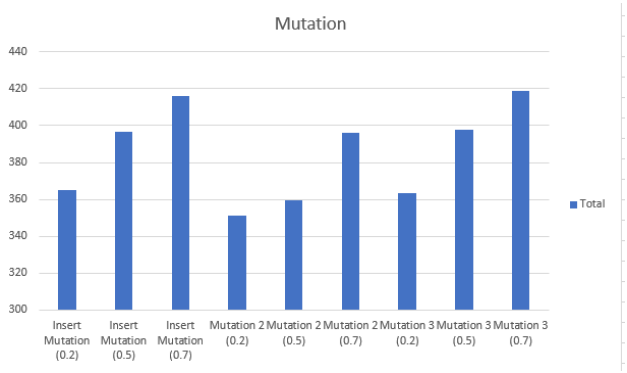

Figura 19 - Operadores de Mutação com diferentes probabilidades de ocorrência

Ao contrário do verificado nos datasets anteriores, as mutações com menor probabilidade de ocorrência revelaram-se com melhores resultados. Este comportamento já vai de acordo com o que seria esperado. A mutação melhor foi, novamente, a Mutation2.

```
Population size: 100

Max generations: 400

Selection: Tournament selection(4)

Recombination: Recombination 2 (0.5)

Mutation:Mutation 3 (0.7)

Fitness: 225
[1, 2, 9, 13, 41, 6, 3, 10, 12, 14, 31, 30, 40, 17, 21, 24, 34, 36, 35, 27, 18, 19, 22, 25, 28, 39,
29, 26, 23, 11, 7, 4, 5, 8, 15, 16, 20, 33, 38, 32, 37]

Forklifts path:
[3,4] -> [1,0] -> [2,2] -> [4,2] -> [9,6] -> S
Steps: 33

[3,13] -> [3,11] -> [2,8] -> [4,6] -> [8,11] -> [9,9] -> [15,14] -> [15,12] -> S
Steps: 42

[9,1] -> [10,0] -> [11,0] -> [14,0] -> [16,0] -> [16,5] -> [16,3] -> [15,5] -> [10,3] ->
[10,5] -> [11,3] -> [14,6] -> [15,6] -> S
Steps: 34

[16,10] -> [15,11] -> [14,9] -> [11,14] -> [4,17] -> [3,15] -> [2,17] -> [2,18] -> [3,20] ->
[9,18] -> [9,20] -> [10,20] -> [15,20] -> [17,18] -> [15,17] -> [16,15] -> S
Steps: 58
```

Figura 20 - Apresentação de uma das melhores soluções encontradas

O melhor conjunto de caminhos encontrado foi:

[3,4] -> [1,0] -> [2,2] -> [4,2] -> [9,6] -> S

Steps: 33

[3,13] -> [3,11] -> [2,8] -> [4,6] -> [8,11] -> [9,9] -> [15,14] -> [15,12] -> S

Steps: 42

[9,1] -> [10,0] -> [11,0] -> [14,0] -> [16,0] -> [16,5] -> [16,3] -> [15,5] -> [10,3] -> [10,5] -> [11,3] -> [14,6] -> [15,6] -> S

Steps: 34

[16,10] -> [15,11] -> [14,9] -> [11,14] -> [4,17] -> [3,15] -> [2,17] -> [2,18] -> [3,20] -> [9,18] -> [9,20] -> [10,20] -> [15,20] -> [17,18] -> [15,17] -> [16,15] -> S

9. CONCLUSÃO

Concluíndo, podemos afirmar que os resultados obtidos para cada problema são melhores quanto maior as suas populações (menores probabilidades em haver convergência prematura e maior a área de exploração do problema), maior o número de gerações (igualmente menores probabilidades em haver convergência prematura e mais "tempo" dado para haver convergência para a solução ideal), depende do operador genético usado e da sua probabilidade de ocorrência – tanto para recombinação como para mutação -, depende do método de seleção – o torneio. Depende ainda da população inicial, apesar de termos utilizado um método aleatório. Se tivessemos desenvolvido um método para gerar uma população mais perto da solução teria sido vantajoso. Os resultados obtidos para cada problema dependem ainda da função fitness utilizada. Apesar de não termos apresentado aqui, fizemos experiências para uma função fitness que contabilizasse apenas os steps do último forklift a se dirigir à saída (pois este iria dar o "tempo" real para a conclusão do problema) e fizemos experiências com uma função fitness onde era também contabilizado os steps totais de todos os forklifts. Este último revelou ter melhores resultados, possivelmente por impedir convergências a soluções com forklifts que acabassem muito antes dos outros, fazendo com que o último forklift fizesse consideravelmente mais steps para concluir o problema. Ou seja, ajudava a desenvolver melhores soluções, apesar de não ser propriamente indicativo da solução com menos steps.

Finalizando, concluí-se que na realização de um bom algoritmo genético deverá-se ter em conta todos os "parâmetros" mencionados em cima, e realizar experiências para conseguir testar qual a influência que cada um destes tem na solução do problema, para se poder escolher quais as melhores opções para uma rápida descoberta da solução ideal do problema.

10. Referências

Stuart J. Russell and Peter Norvig. Artificial Intelligence A Modern Approach