

Faculdade de Engenharia da Universidade do Porto



Gestão de Rotas na Recolha de Resíduos

João Carlos de Antão Bilber

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Orientador na FEUP: José Soeiro Ferreira (Prof.)
Orientadora no INESC TEC: Ana Maria Rodrigues (M.^a)

Junho de 2012

© João Carlos Antão Bilber, 2012

Resumo

Os problemas de planeamento de rotas são estudados com frequência pela comunidade científica, contudo são raras as validações dos modelos desenvolvidos em casos reais.

A presente tese abrange um estudo detalhado do problema de planeamento de rotas na recolha de resíduos, referente à região do Alto Minho. Este problema é novo e consiste em determinar as rotas mais económicas que passem obrigatoriamente por todos os arcos com procura, satisfazendo no geral as restrições de capacidade dos veículos e de número de descargas por aterro.

Devido à complexidade deste problema, um método aproximado é proposto para sua resolução, que é uma meta-heurística baseada na colónia de formigas.

O método baseado na colónia de formigas já demonstrou ser um algoritmo poderoso e com soluções muito próximas do óptimo para problemas com procura nos arcos, contudo é a primeira vez que é feita uma adaptação desta meta-heurística a este problema e é até mesmo a primeira vez que se tenta resolver este problema com um método aproximado.

Propõe-se uma nova meta-heurística para resolução deste problema, pois terão que haver modificações em vários componentes desenvolvidos para o modelo original de forma a adaptá-los às nossas particularidades. Deste modo para além da introdução de alterações nos algoritmos da colónia de formigas, é adaptada uma heurística para criação da população inicial, e é criada uma heurística própria para gestão das descargas nos aterros.

De forma a melhorar o valor da população inicial e das populações em cada iteração da meta-heurística, são aplicados dois algoritmos de melhoria local.

O algoritmo desenvolvido é testado numa instância real representativa de um circuito de recolha de Resíduos Sólidos Urbanos, e também num grupo de instâncias teóricas.

Abstract

The arc routing problems are usually studied by the scientific community, however only a few are validated in real-life situations.

This thesis presents a detailed study of a waste collection routing problem in the Alto Minho region. This problem consists in finding the cheaper routes that service all the edges demand, subjected to main restrictions like vehicle capacity and limited number of visits for waste disposal.

Due to this problem complexity, an approximate method is proposed to solve it, which is a meta-heuristic inspired on the ant colony optimization.

The Ant based method has proven to be a powerful algorithm and with near optimal solutions for arc routing problems, however this is the first time that an Ant based meta-heuristic is proposed to solve this problem, and is even the first time that this problem is tried to solve with an approximate method.

A new meta-heuristic is proposed to solve this problem, because changes need to be made in various components developed for the standard model, in order to extend them to our specificities. Thus besides the modifications in the Ant main algorithms, is adapted a heuristic for construct the initial population, and is introduced a new method for manage the waste disposals.

In order to improve the solutions quality achieved in the initial population and in the population after each iteration of the meta-heuristic, are applied two local search procedures.

The developed algorithm is tested in a real instance that is an image of a waste collection circuit, and in a group of theoretical instances.

Agradecimentos

À minha família, amigos e todos os que me apoiaram e orientaram nesta etapa.

Índice

Resumo	v
Abstract	vii
Agradecimentos	ix
Índice	xi
Lista de figuras	xiii
Lista de tabelas	xv
Abreviaturas e símbolos	xvii
Capítulo 1	1
Introdução	1
1.1 - Motivação	1
1.2 - Contexto e Objectivos	2
1.3 - Estrutura da dissertação	6
Capítulo 2	7
Estado da Arte	7
2.1 - Algoritmos e Complexidade computacional	7
2.2 - Teoria dos grafos	10
2.3 - Problemas de optimização de rotas	12
2.3.1 - Introdução	12
2.3.2 - Problemas com procura nos vértices	14
2.3.3 - Problemas com procura nos arcos	15
2.3.4 - Métodos de Resolução	17
2.4 - O Sistema de Recolha de RSU	19
2.4.1 - Caracterização do sistema	19
2.4.2 - Aplicações descritas na literatura	21
Capítulo 3	25
Definição do modelo matemático	25

3.1 - Definições e notação	25
3.2 - Modelação em programação linear inteira mista	28
3.2.1 - Entradas	28
3.2.2 - Saídas	28
3.2.3 - Objectivo	29
3.2.4 - Restrições	29
Capítulo 4.....	33
Método de resolução	33
4.1 - Introdução ao algoritmo da colónia de formigas	33
4.1.1 - Inspiração biológica.....	33
4.1.2 - Primeiras heurísticas ACO	35
4.2 - Meta-heurística ACO-LML	37
4.2.1 - Inicialização.....	38
4.2.2 - Geração da população inicial	39
4.2.3 - Algoritmo ACO	42
4.2.4 - Melhoria das soluções	45
4.2.5 - Divisão em rotas viáveis	47
Capítulo 5.....	49
Resultados computacionais	49
5.1 - Instâncias e parâmetros.....	49
5.2 - População Inicial	53
5.3 - Meta-heurística	54
5.3.1 - Instâncias “GDB”	54
5.3.2 - Instância “AltoMinho”	56
Capítulo 6.....	59
Conclusões.....	59
Anexo A	61
Referências	63

Lista de figuras

Figura 1.1 - Evolução da quantidade de RSU produzidos por dia (fontes: [3, 4])	2
Figura 1.2 - Sistema de recolha de Resíduos Sólidos Urbanos	4
Figura 1.3 - Sustentabilidade ambiental dos concelhos em 2010 (fonte: [3])	4
Figura 2.1 - Evolução da função $f(n)$ com o aumento da entrada n	8
Figura 2.2 - Relação entre conjuntos de classes de tempo polinomial	9
Figura 2.3 - Representação do problema das pontes de Königsberg num multigrafo	10
Figura 2.4 - Grafo de uma rede	11
Figura 2.5 - Classificação das Meta-heurísticas [19]	18
Figura 2.6 - Quantidade de RSU produzidos por dia no concelho de Monção (fonte [4]).....	19
Figura 2.7 - Recolha porta-a-porta	20
Figura 3.1 - Procura nos arcos no caso de estudo	26
Figura 3.2 - Exemplo do MCARP-LML com 2 aterros.....	27
Figura 4.1 - Comportamentos <i>Swarm Intelligence</i> na natureza [29]	34
Figura 4.2 - Pseudocódigo da meta-heurística ACO de Dorigo, Caro e Gambardella [35].....	36
Figura 4.3 - Pseudocódigo da meta-heurística ACO-LML	38
Figura 4.4 - Visualização no Google Earth do grafo do caso real	39
Figura 4.5 - Pseudocódigo do algoritmo PSRC-LML	39
Figura 4.6 - Pseudocódigo do algoritmo “Construcao_solucoes”	42
Figura 4.7 - Algoritmo de melhoria global - divisão de zonas	45
Figura 4.8 - Algoritmo de melhoria global - exemplo do movimento.....	46
Figura 4.9 - Algoritmo de melhoria local - exemplo do movimento	46
Figura 4.10 - Algoritmo de divisão em rotas viáveis	48
Figura 5.1 - Comportamento da meta-heurística em 50 iterações	57
Figura 5.2 - Comportamento da meta-heurística em 100 iterações	57
Figura 5.3 - Comportamento da meta-heurística em 200 iterações	57
Figura 5.4 - Mapa da viagem efectuada com solução óptima determinada pelo CPLEX.....	58
Figura 5.5 - Mapa da viagem efectuada com solução melhor da meta-heurística ACO-LML....	58

Lista de tabelas

Tabela 1.1 - Lista dos principais constituintes dos RSU e respectivas densidades [5]	3
Tabela 2.1 - Aumento do processamento computacional com evolução dos computadores[7] .	8
Tabela 3.1 - Variáveis do modelo MCARP-LML	28
Tabela 4.1 - Principais notações utilizadas.....	37
Tabela 5.1 - Principais características das instâncias testadas	50
Tabela 5.2 - Parâmetros usados nos algoritmos	51
Tabela 5.3 - Resultados da determinação do número de elementos da população	51
Tabela 5.4 - Resultados da heurística PSRC-LML	53
Tabela 5.5 - Resultados obtidos nas instâncias GDB	55
Tabela 5.6 - Resultados obtidos na instância “AltoMinho”.....	56

Abreviaturas e símbolos

Lista de abreviaturas (ordenadas por ordem alfabética)

ACO	Ant Colony Optimization
ACO-LML	Designação do algoritmo desenvolvido
CARP	<i>Capacitated Arc Routing Problem</i>
CCPP	<i>Capacitated Chinese postmand problem</i>
CPP	<i>Chinese postman problem</i>
CVRP	<i>Capacitated Vehicle Routing Problem</i>
MCARP	<i>Mixed capacitated arc routing problem</i>
MCARP-LML	<i>Mixed capacitated arc routing problem with Limited Multi-Landfills</i>
PERSU	Plano Estratégico para os Resíduos Sólidos Urbanos
PSRC	<i>Path-Scanning with Random Criterium</i>
PSRC-LML	Heurística desenvolvida para criação da população inicial
RPP	<i>Rural Postman Problem</i>
RSU	Resíduo sólido urbano
SA	<i>Simulated annealing</i>
SI	<i>Swarm Intelligence</i>
SARP	<i>Sectoring arc routing problem</i>
TSP	<i>Travelling Salesmen Problem</i>
VRP	<i>Vehicle Routing Problem</i>

Capítulo 1

Introdução

1.1 - Motivação

A sustentabilidade económica e financeira de serviços públicos essenciais aos habitantes tais como água, electricidade ou resíduos sólidos urbanos é um tema que é frequentemente debatido na opinião pública.

O termo “resíduo sólido urbano” (RSU) define-se como um conjunto de componentes de origem doméstica, ou de outras origens tais como comercial ou industrial mas de natureza ou composição igual aos resíduos domésticos.

A avaliação da gestão dos serviços públicos é feita com base no cumprimento dos objectivos sociais ou ambientais, sem que para isso comprometa a sustentabilidade económica e financeira da entidade pública.

Actualmente, o sector do RSU encontra-se estruturado e regulamentado com vista ao cumprimento de metas europeias de reciclagem e valorização, estando em vigor a segunda versão do Plano Estratégico para os Resíduos Sólidos Urbanos (PERSU II), regulamentado pelo Portaria n.º187/2007, de 12 de Fevereiro. Este plano pretende afirmar-se com um método de gestão eficaz com vista à sustentabilidade ambiental, económico e social do sistema.

Segundo um estudo feito sobre os tarifários de gestão de resíduos para o ERSAR (Entidade Reguladora dos Serviços de Águas e Resíduos, antiga IRAR)[1], que é a entidade nacional responsável pela regulação do serviço público de gestão de resíduos urbanos, foi verificado que as autarquias suportam um défice tarifário anual médio de cerca de 70% dos custos dos serviços de remoção, transferência e tratamento de resíduos urbanos prestados.

Uma das medidas apontadas para a viabilidade do sistema de gestão de RSU, é a optimização dos custos operativos. Como os custos de recolha e transporte de RSU correspondem a valores entre os 60% a 80% do custo total de gestão [2], o presente estudo irá incidir na optimização das rotas na recolha de resíduos sólidos urbanos, com vista à minimização dos custos destas operações.

1.2 - Contexto e Objectivos

Num passado recente, os resíduos urbanos eram transportados directamente para lixeiras a céu aberto situadas nos arredores dos centros populacionais. A perspectiva de melhoria das condições de vida levou à mudança das pessoas para as cidades, o que fez aumentar rapidamente a densidade populacional das áreas urbanas e consequentemente o número de lixeiras. A falta de um plano estratégico de gestão dos RSU levou a que situações como estas passassem de simples problemas a graves problemas de saúde pública.

A produção de resíduos sólidos urbanos tem evoluído significativamente nas últimas décadas, e está relacionada com o aumento da qualidade de vida dos cidadãos. Como se pode observar no gráfico da figura 1.1, em média cada português produziu por cada dia de 2010, 1,4Kg de RSU. É fácil de entender que numa pequena cidade com dez mil habitantes, se produzam diariamente 14 toneladas de resíduos. Também se pode observar na mesma figura que os concelhos com maior densidade populacional produzem mais RSU por dia que os centros de menor dimensão.

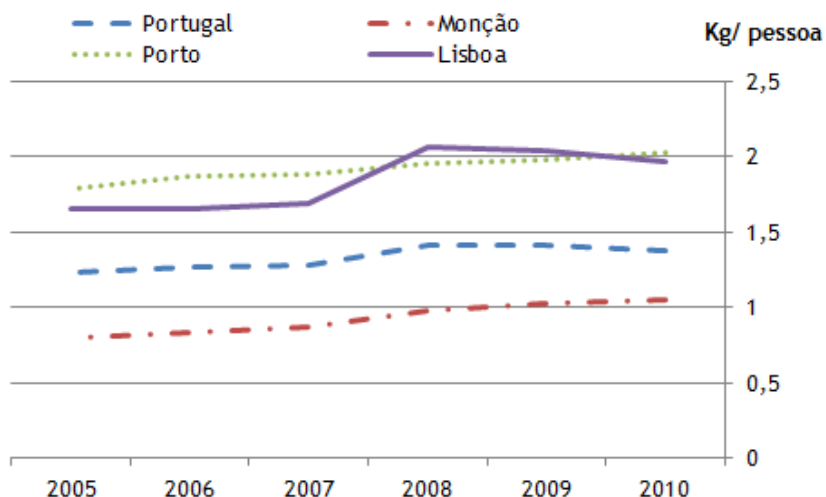


Figura 1.1 - Evolução da quantidade de RSU produzidos por dia (fontes: [3, 4])

A remoção de resíduos sólidos urbanos compreende duas fases distintas:

- Deposição de resíduos: Compreende a separação dos resíduos pelo produtor e colocação nos recipientes específicos atribuídos a cada tipo de resíduo;
- Recolha e Transporte: Operação de transferência dos resíduos alojados nos contentores para os aterros ou estações de transferência, através de viaturas próprias.

A enorme quantidade de resíduos sólidos urbanos produzidos diariamente, são hoje em dia depositados pelos produtores em contentores próprios para resíduos indiferenciados ou em ecopontos. Os ecopontos são conjuntos de três ou mais contentores para recolha selectiva de embalagens usadas.

Tipicamente, os RSU são constituídos por vários tipos de material, sendo difícil a atribuição de uma relação exacta entre o volume ocupado e o seu peso. Na tabela 1.1 são apresentadas as categorias mais comuns que constituem os RSU e a respectiva densidade, sem compressão.

Tabela 1.1 - Lista dos principais constituintes dos RSU e respectivas densidades [5]

Categoria	Densidade (t/m ³)
Matéria orgânica	0,343
Papel e cartão	0,100
Vidro	0,174
Metais	0,900
Plásticos	0,013

Segundo dados de 2009 [6], a densidade determinada para os típicos RSU não compactados foi de 0,131 t/m³, e para os RSU compactados foi de 0,262 t/m³. Estes valores poderão variar substancialmente entre concelhos, pois este rácio está dependente da quantidade e natureza dos resíduos colocados nos contentores de resíduos indiferenciados. Por outro lado os camiões de recolha mais recentes conseguem uma capacidade de compactação de 6:1.

Por questões de espaço, de saúde pública ou outras razões, é necessária a recolha periódica destes mesmos resíduos. Deste modo é necessário estabelecer uma frequência mínima de recolha de RSU por semana que seja aceitável para os habitantes e viável para as entidades públicas.

O Sistema de Gestão dos RSU tal como o sistema de águas, são frequentemente distribuídos consoante o tipo de actividades desenvolvidas, em dois grupos distintos chamados de alta e baixa.

Considera-se o sistema em baixa, o processo que inclui a recolha dos RSU, em contentores situados em locais estratégicos ou recolha porta-a-porta, e o transporte desses resíduos recolhidos até centros de tratamento ou estações de transferência. Os municípios ou associações de municípios são responsáveis pelo sistema, mas a gestão/ operação do sistema pode ser delegada a empresas privadas.

O sistema em alta refere-se ao tratamento dos RSU, através da sua separação, valorização, reciclagem ou deposição em aterro ou local apropriado. As empresas multimunicipais ou intermunicipais são responsáveis pelo sistema e a gestão é feita por empresas concessionárias de capitais maioritariamente públicos.

A figura 1.2 mostra como se processa o fluxo de recolha de resíduos sólidos urbanos e a separação entre o sistema em alta e em baixa.

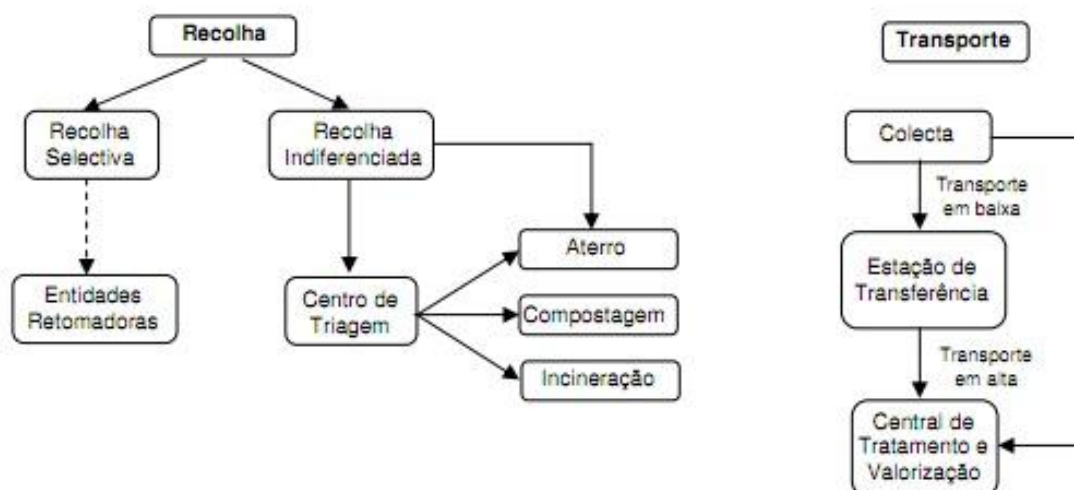


Figura 1.2 - Sistema de recolha de Resíduos Sólidos Urbanos

Do ponto de vista da sustentabilidade ambiental, um dos objectivos que o Plano Estratégico para os Resíduos Sólidos Urbanos (PERSU II) aponta é a redução da quantidade de RSU que vão para aterro, através de um aumento da reciclagem e da valorização orgânica e energética. Como se pode ver na figura 1.3, os RSU que vão para aterro continuam a atingir percentagens que ultrapassam em muito os objectivos propostos, à excepção dos grandes centros urbanos, que mostram já estar em conformidade com as metas europeias.

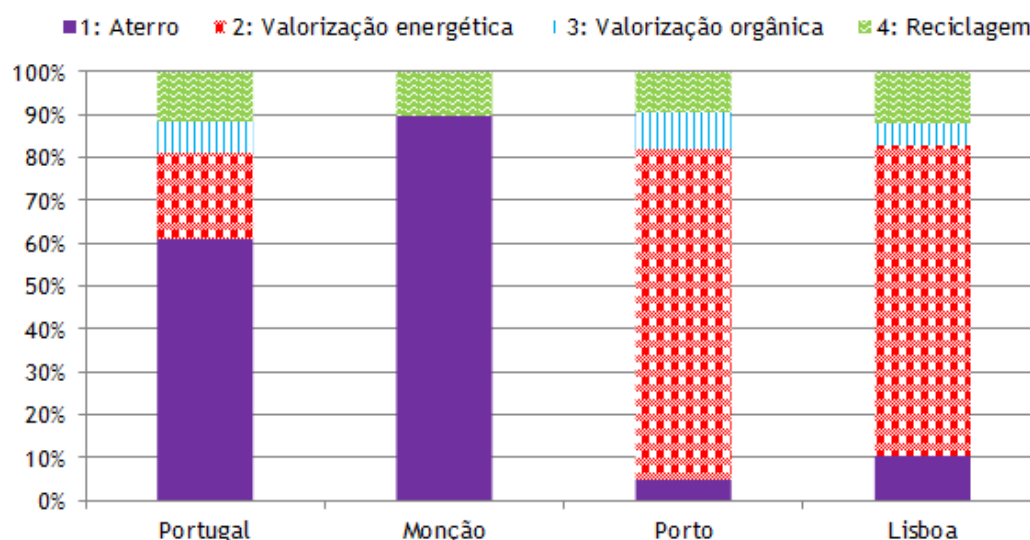


Figura 1.3 - Sustentabilidade ambiental dos concelhos em 2010 (fonte: [3])

Para se tornar economicamente sustentável o sistema de gestão de resíduos sólidos urbanos, também é necessária uma análise profunda das causas inerentes ao elevado défice tarifário.

O princípio do utilizador pagador é um sistema tarifário que teoricamente é mais justo, contudo é de difícil aplicação nos RSU. Em Portugal é grande a diversidade de sistemas tarifários domésticos referentes à recolha de RSU, sendo o mais comum a indexação de tarifas à factura de saneamento ou abastecimento de água. Neste caso, alguns municípios preferem cobrar um valor fixo mensal, e outras preferem um preço composto baseado num componente

fixo e noutro variável. Os outros sistemas existentes baseiam-se em variáveis como a frequência da remoção, as características rurais ou urbanas do local servido, o sistema de remoção ou a área da habitação.

O PERSU II propõe as seguintes linhas gerais com vista à sustentabilidade económica e financeira:

- Optimização dos sistemas “em alta” e “em baixa”;
- “*Outsourcing* de actividades”;
- Tarifas reais suportadas pelo utilizador final;
- Regulação ambiental, económica e financeira;
- Avaliação económica de soluções alternativas.

A presente dissertação irá enquadrar-se na optimização do sistema “em baixa” de recolha de resíduos sólidos urbanos. Como grande parte dos custos do sistema são referentes aos custos de transporte, será elaborado um estudo completo de optimização de rotas de recolha de RSU.

Muitos dos artigos que analisam o problema da optimização de rotas na recolha de resíduos, demonstram resultados satisfatórios do ponto de vista teórico. Como em regra não é feita a aplicação desses resultados em casos realmente reais, por um lado levantam-se dúvidas quanto ao real desempenho dos métodos desenvolvidos, e por outro lado os algoritmos são incompatíveis com a realidade da recolha dos RSU visto serem analisadas versões simplificadas dos problemas reais.

Deste modo, o estudo a realizar neste trabalho será de um caso prático e real da recolha de resíduos sólidos, referente a um concelho do Alto Minho de Portugal.

O objectivo do trabalho é a optimização de rotas na recolha de resíduos, isto é, pretende-se encontrar as melhores rotas de recolha de RSU que satisfaçam os requisitos específicos deste tipo de problemas. Como irá ser referido no estado da arte, a recolha de resíduos é um problema complexo que está sujeito a diversos casos particulares que irão dificultar o estudo.

Deste modo, a dissertação irá desenvolver-se pelas seguintes fases:

1. Caracterização do problema específico da RSU: Estudo detalhado do sistema de gestão dos RSU num concelho de Portugal;
2. Definição do Modelo Matemático: Análise detalhada do modelo que foi recentemente desenvolvido para este problema;
3. Método de resolução: Implementação e desenvolvimento de um método de resolução para este problema de optimização combinatória. O método de resolução será um método aproximado (meta-heurística);
4. Resultados computacionais: Desenvolvimento de aplicação informática que possa validar o algoritmo aproximado;
5. Testes e Análise de resultados: Realização de testes e análise comparativa das soluções resultantes.

As cinco etapas enunciadas correspondem, genericamente, às contribuições da presente tese.

1.3 - Estrutura da dissertação

Nesta secção é apresentada a estrutura e organização deste documento, bem como o conteúdo apresentado em cada um dos capítulos que o constituem.

Estruturalmente, a dissertação está dividida em seis capítulos.

No Capítulo 1, é feita a introdução com a apresentação do tema da tese, justificação da sua relevância, explicação da abordagem a adoptar e da organização prevista da dissertação.

O Capítulo 2 contém uma revisão bibliográfica, onde são abordados temas ou definições importantes para este trabalho, e são também analisados estudos de problemas semelhantes.

No Capítulo 3 é apresentada o modelo matemático e a respectiva formulação do problema abordado.

No Capítulo 4 são definidos os métodos de resolução elaborados para o problema de optimização combinatória.

No Capítulo 5 são apresentados os resultados computacionais que foram obtidos pelos algoritmos utilizados.

E por fim, no Capítulo 6 são apresentadas as conclusões e referências para desenvolvimento futuro.

Capítulo 2

Estado da Arte

2.1 - Algoritmos e Complexidade computacional

Para se compreenderem alguns termos e métodos usado ao longo deste documento, e que justificam certas escolhas na resolução de problemas de optimização combinatória, faz-se nesta secção uma breve introdução à complexidade computacional.

O termo Complexidade Computacional envolve a classificação de problemas, de acordo com facilidade com que estes são resolvidos. A complexidade computacional de um algoritmo é medida pela quantidade de passos necessários à resolução de uma instância num problema.

O tempo polinomial pode ser interpretado como a linha divisória que separa a classe computacional de problemas que podem ser resolvidos eficientemente. A definição tempo polinomial de um algoritmo refere-se ao tempo de execução de um algoritmo que é função polinomial do tamanho da entrada.

A notação assintótica ou Big-O descreve o comportamento dos limites que uma função $f(n)$ tem, em relação à taxa de crescimento n . Portanto funções diferentes com semelhante taxa de crescimento são representadas utilizando a mesma notação assintótica. Deste modo é possível comparar algoritmos e distingui-los quanto à sua eficiência. A figura 2.1 representa um gráfico de evolução da função $f(n)$ com o aumento da entrada n . Foram representadas algumas das classes de funções mais utilizadas nos algoritmos, onde se pode observar que qualquer algoritmo de ordem factorial $O(n!)$ é muito pouco eficiente pois tem um crescimento ainda superior ao exponencial. Contudo a ordem factorial está presente em alguns dos algoritmos exactos para a resolução de problemas de optimização combinatória, tal como o problema do caixeiro-viajante, abordado nas secções seguintes.

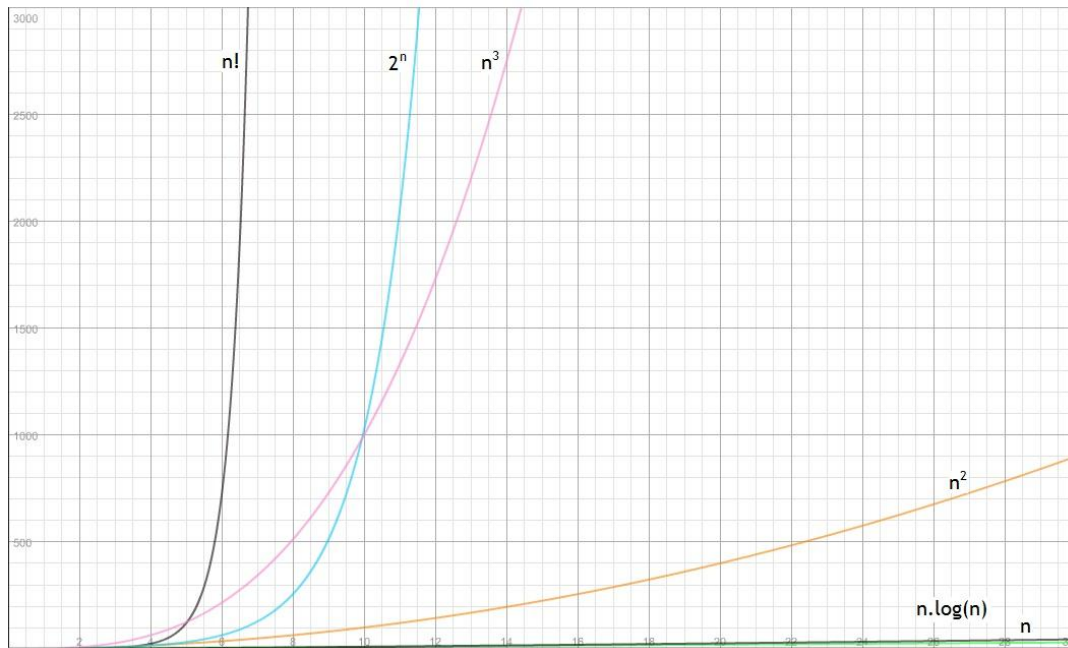


Figura 2.1 - Evolução da função $f(n)$ com o aumento da entrada n

A tabela seguinte mostra que a evolução da capacidade de processamento de dados dos computadores, não irá trazer grandes vantagens na resolução de problemas de complexidade computacional superior à de ordem polinomial. Conclui-se deste modo é preciso esperar muito tempo para que a tecnologia evolua ao ponto de resolver problemas de elevada complexidade.

Tabela 2.1 - Aumento do processamento computacional com evolução dos computadores[7]

Função	Tamanho da Instância resolvida num dia	
	Computador	Computador B
	A	10X mais rápido que A
n	10^{12}	10^{13}
$n \log n$	$0,948 \times 10^{11}$	$0,87 \times 10^{12}$
n^2	10^6	$3,16 \times 10^6$
n^3	10^4	$2,15 \times 10^4$
$10^8 n^4$	10	18
2^n	40	43
10^n	12	13
$n^{\log n}$	79	95
$n!$	14	15

Houve a necessidade de se agruparem conjuntos de problemas relacionados do ponto de vista complexo e computacional. Surgiram assim várias classes de tempo polinomial, mas somente duas irão ser mencionadas de seguida.

- A classe P engloba um conjunto de problemas de decisão que podem ser resolvidos por um algoritmo em tempo polinomial.
- A classe NP engloba um conjunto de problemas de decisão que para os quais se desconhece um algoritmo de resolução em tempo polinomial, mas cuja solução pode ser verificada por um algoritmo em tempo polinomial.

Surgiram duas subclasses de problemas referentes ao NP que correspondem aos problemas mais difíceis de resolver:

- A subclasse NP-difícil (NP-Hard) representa os problemas P que podem ser reduzidos de problemas da classe NP em tempo polinomial;
- A subclasse NP-completa (NP-complete) compreende os problemas que pertencem simultaneamente às classes NP e NP-difícil.

Em matemática computacional, há uma questão que ainda ninguém conseguiu provar, que é se a classe P é igual à NP o que torna todos os problemas NP e NP-completo resolúveis em tempo polinomial.

A figura 2.2 mostra a relação entre conjuntos de classes de tempo polinomial, para as duas relações de igualdade de P e NP.

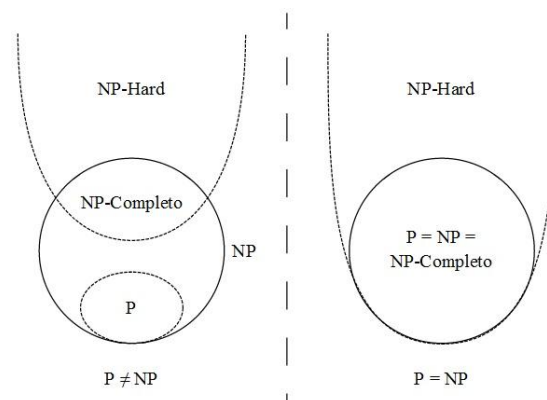


Figura 2.2 - Relação entre conjuntos de classes de tempo polinomial

2.2 - Teoria dos grafos

Os grafos são um dos objectos de estudo em matemática discreta, e podem-se definir como o par ordenado $G = \langle V, E \rangle$, onde $V(G)$ é um conjunto de vértices e $E(G)$ um conjunto de arestas. Cada aresta liga um par de vértices.

A teoria dos grafos surgiu quando o matemático suíço, Leonhard Euler, em 1736 se questionou se alguém poderia dar um passeio atravessando as 7 pontes de Königsberg de uma só vez e voltar ao mesmo ponto de partida. Euler estudou o problema, e para isso representou-o num grafo, sendo a terra os vértices e as pontes como arcos, tal como exemplificado na figura 2.3.

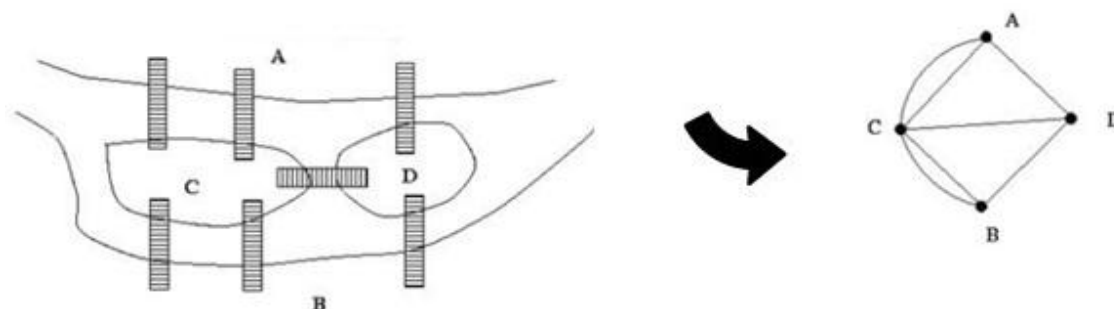


Figura 2.3 - Representação do problema das pontes de Königsberg num multigrafo

Se houver mais do que uma aresta a ligar o mesmo par de vértices, dá-se o nome de multigrafo.

Se os arcos tiverem direcção, o grafo é chamado de grafo orientado ou dígrafo. Contudo se existir no mesmo grafo arcos com e sem direcção, é chamado de grafo misto.

Um grafo é considerado conexo se tiver um caminho entre dois vértices. O grau de um vértice é o número de arestas que incidem nele.

Euler conseguiu provar que era impossível fazer o circuito das pontes e provou-o pelo teorema que herdou o nome do matemático. O Teorema de Euler diz que um multigrafo conexo tem um circuito de Euler se e só se todos os vértices tiverem grau par. Entende-se por circuito de Euler um passeio sem repetir as arestas que começa e acaba no mesmo vértice, e que contém todas as arestas do grafo.

Do mesmo modo foi estudado um problema pelo matemático irlandês Willian Hamilton, em 1857, algo semelhante ao que Euler se tinha questionado, mas em vez do circuito ser de passagem uma única vez por todas as pontes, Hamilton pretendia encontrar um circuito que passasse uma única vez por todos os vértices (Ciclo de Hamilton).

Os conceitos de circuitos de Euler e ciclos de Hamilton são similares naquilo que proíbem voltar a utilizar: nos circuitos de Euler as arestas, nos ciclos de Hamilton os vértices. No entanto, é muito mais difícil determinar quais são os grafos conexos que admitem um ciclo de Hamilton do que determinar os grafos conexos que admitem um circuito de Euler.

As redes podem ser definidas como grafos com um ou mais pesos associados os vértices e /ou arestas conforme mostra a figura 2.4. Este número é frequentemente referido como peso da ligação. Essa classificação é dada de acordo com a necessidade, ou não, da indicação do fluxo entre os vértices. Na prática este número pode representar:

- Custos;
- Distâncias,
- Capacidades;
- Tempos;
- etc.

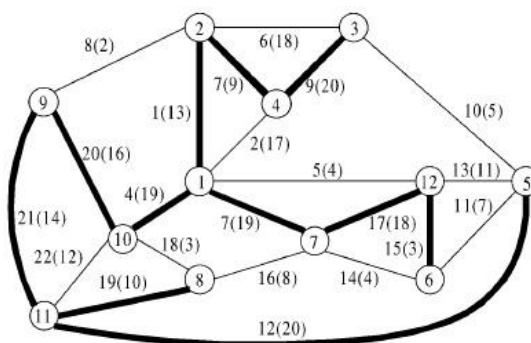


Figura 2.4 - Grafo de uma rede

Tal como Euler e Hamilton tentaram resolver os problemas de planeamento de rotas através de grafos, faz todo o sentido aplicar o mesmo princípio ao problema da recolha de recolha dos resíduos sólidos urbanos.

2.3 - Problemas de otimização de rotas

2.3.1 - Introdução

Os problemas de planeamento de rotas são constituídos pelos seguintes principais componentes:

- Rede de estradas: Pode ser representada por um grafo não orientado no caso de redes em grande escala (ex.: ligações entre cidades), ou por um grafo orientado (dígrafo) ou misto, no caso de redes em pequena escala (ex.: cidades). Os vértices dos grafos correspondem a armazéns, clientes, intercepções de ruas e os arcos a estradas, distâncias, custos ou tempos de viagem;
- Clientes: São as entidades que requerem o serviço, que pode ser de entrega ou recolha. Os clientes estão associados aos vértices (NRP) ou aos arcos (ARP). O serviço pode ter que ser cumprido numa determinada janela temporal;
- Veículos: Existe um conjunto de viaturas (frota), para cumprir a procura total. As viaturas têm normalmente uma determinada capacidade e um custo operativo associado. A frota pode englobar viaturas iguais o que facilita a formulação do problema;
- Armazéns e Locais particulares: Os armazéns são o local inicial de onde partem os veículos. Os locais particulares podem ou não existir, e são entrepostos ou no caso da recolha de resíduos os aterros ou estações de transferência. Ambos os casos são representados em vértices com condições específicas;
- Condutores: Podem ser considerados os horários de trabalho;
- Restrições operacionais: As restrições dividem-se em locais e globais. As locais podem ser a capacidade do veículo ou a distância/ duração máxima da rota. As globais podem ser o número máximo de rotas por veículo, número máximo de viaturas da frota, o balanceamento de carga;
- Objectivo a otimizar: Neste tipo de problemas, o objectivo normalmente é de minimização de custos, distâncias, tempos, etc.

Geralmente os problemas de determinação de rotas óptimas são classificados em dois grandes grupos: problemas com procuras nos vértices (*Node Routing Problems* - NRP) e problemas com procura nos arcos (*Arc Routing Problems* - ARP)[8].

Quando se incluem em problemas de planeamento de rotas, restrições de capacidade dos veículos, conduzem a problemas de roteamento de veículos (*Vehicle Routing Problems* - VRP) e problemas com procura nos arcos e restrições de capacidade (*Capacitated Arc Routing Problems* - CARP) [9].

De facto, um problema de procura nos vértices pode ser transformado num problema de procura nos arcos. O CVRP pode ser transformado num CARP, bastando para isso dividir cada nó em dois vértices ligados por um arco, e atribuindo a procura do vértice ao arco [9].

Na secção 23.2 são referidos alguns modelos mais comuns de problemas com procura nos vértices e na secção 2.3.3 são referidos alguns modelos mais comuns de problemas com procura nos arcos.

Na secção 2.3.4 são mencionados métodos de resolução para problemas de optimização combinatória.

2.3.2 - Problemas com procura nos vértices

O problema do caixeiro-viajante (*Travelling Salesmen Problem* - TSP) é um dos problemas mais investigados em optimização combinatória, e é um exemplo típico de um problema com procura nos vértices.

O problema do TSP consiste em encontrar o caminho mais curto para um caixeiro-viajante visitar todas as cidades e de seguida voltar ao ponto de partida. Representando o problema num grafo, pode ser dito que o problema consiste em analisar todos circuitos Hamiltonianos existentes para $(n+1)$ pontos se ele tiver que visita n cidades.

As aplicações reais de um modelo semelhante ao caixeiro-viajante:

- Um robot que efectua furos numa placa de circuito impresso;
- Um determinado percurso turístico;
- Um técnico de reparações que tem de visitar várias empresas.

Se é adicionado ao problema do TSP uma restrição de capacidade ao veículo, e a procura total excede a capacidade de uma viatura obrigando a mais do que uma viagem, o problema transforma-se num conhecido como VRP (*Vehicle Routing Problem*). O VRP tem como objectivo determinar um conjunto de rotas que um ou mais veículos têm de cumprir de modo a que satisfaçam determinados pontos de procura ou oferta, com o menor custo total. Dada a versatilidade deste modelo, há várias situações onde o VRP pode ser aplicado, quer na distribuição, quer nos transportes.

Certos autores diferenciam o CVRP (*Capacitated Vehicle Routing Problem*) do VRP, pois consideram o VRP igual ao TSP. Neste documento será considerado o VRP igual ao CVRP.

Em problemas VRP, será necessário o cálculo das rotas e a atribuição das viaturas a essas rotas, tendo em consideração as capacidades das viaturas. O modelo matemático deve assim respeitar um conjunto de restrições tais como:

- Cada rota deve começar e acabar no armazém;
- Cada cliente deve ser visitado uma só vez;
- A capacidade da rota não deve exceder a capacidade do veículo;
- Frota de veículos com capacidade idêntica;
- A procura total tem que ser satisfeita.

Os problemas VRP têm normalmente como função objectivo a minimização de custos totais ou a distância percorrida. Contudo poderão ser considerados outros objectivos tais como a minimização de tempos de operação.

Quando se pretende aplicar determinados modelos a casos práticos reais, surge por vezes a necessidade de se efectuarem adaptações ao modelo original do VRP de modo a obter soluções viáveis. Foi então que surgiram variantes do VRP que respondem a essas particularidades, e que normalmente consideram novas restrições.

Uma das variantes mais utilizadas no âmbito da recolha de RSU é o VRPTW (VRP *with time windows*). Neste caso, os clientes têm que ser satisfeitos numa certa janela temporal e segundo certos autores [10], este modelo é o que se aplica à recolha de resíduos provenientes do comércio/serviços.

Em suma, no VRPTW será necessário o cálculo das rotas, tendo em consideração as capacidades das viaturas e os horários de serviço no cliente.

2.3.3 - Problemas com procura nos arcos

Quando a procura a satisfazer reside nos arcos de uma rede, o equivalente ao TSP é o *Chinese postman problem* (CPP) [11].

O CPP é NP-difícil, embora tenha instâncias para as quais se conhecem algoritmos polinomiais [8].

Este problema consiste em cobrir, ao mínimo custo, todos os arcos de uma rede, começando e acabando num nó de origem. Se a procura gerar mais do que uma rota devido às restrições de capacidade, o resultado é denominado de *Capacitated Chinese postman problem* (CCPP).

O CCPP tem muitas aplicações reais, tais como distribuição de correio, manutenção da estrada, i.e. remoção de neve e aplicação de sal, limpeza, recolha de RSU, transporte de crianças, contagens electricidade, recolha moedas parcometros, etc.

Quando numa rede é necessário percorrer somente um conjunto de arcos de uma rede, este problema é conhecido como o *Rural Postman Problem* (RPP), que é um caso particular do *Chinese Postman Problem* (CPP). Segundo Eiselt, são poucos os contextos de aplicação prática onde é necessário servir todos os arcos duma rede, por isso grande parte dos casos reais de planeamento de rotas são normalmente modelados como RPP [12].

Um problema semelhante ao RPP foi introduzido em 1981 por Golden e Wong [9], denominado por CARP (*Capacitated Arc Route Problem*). O CARP introduz no modelo uma restrição que é comum a grande parte dos problemas reais, a restrição de capacidade.

O CARP é considerado uma generalização do CCPP e tem como função objectivo minimizar os custos das rotas sendo uma das restrições o número de viaturas. Caracteriza-se por ser:

- Grafo não orientado;
- Cada rota começa e acaba no armazém;
- Cada arco ter um custo e um peso maiores que zero;
- Cada arco é servido por uma única viatura;
- A procura total de todos os arcos percorridos por qualquer viatura não ultrapasse a carga máxima do veículo.

O CARP é NP-difícil mesmo para grafos orientados e não orientados.

Uma das aplicações do CARP num problema real é a recolha de resíduos, embora seja muito simplista como modelo de redes de ruas reais encontradas por exemplo na recolha de resíduos sólidos urbanos [13].

Mais recentemente foram introduzidas melhorias à formulação inicial do CARP, que traduzem melhor a realidade das ruas numa rede. Esta versão foi denominada ECARP (*Extended version of CARP*), e acrescenta os seguintes quatro suplementos ao CARP [14]:

- Multigrafo misto (MCARP) com dois tipos de ligações: Um grafo misto permite ao modelo ter dois tipos de ruas não necessárias e três tipos de ruas necessárias. Uma rua é não-necessária ou com um arco (rua sentido único) ou com dois arcos de sentido oposto (rua de dois sentidos). Uma rua necessária pode ser uma rua de dois sentidos onde é possível a recolha em zigzag (neste caso aplica-se um arco sem sentido), uma rua de dois sentidos com recolha separada de cada um dos lados e uma rua de sentido único;
- Dois custos diferentes por arco: Quando um veículo exerce actividade num determinado sentido de uma rua tem um certo custo. Se no regresso passar no sentido oposto dessa mesma rua é lógico que exista outro custo bastante inferior;
- Inversão de sentido de marcha proibida (*prohibited turns*) e mudanças de direcção (*turn penalties*): Estas regras de trânsito são comuns em todas as cidades e de extrema importância na criação de rotas;
- Distância máxima de rota: permite ao veículo chegar aos arcos definidos.

Deste modo surgiu o MCARPTC (*mixed capacitated arc routing problema with turn constraints*), consiste em encontrar um conjunto de rotas a um custo mínimo que satisfaça o seguinte:

- Cada rota inicia e pára no armazém;
- Cada arco é servido por uma rota;
- A procura total dos arcos que é servido por cada veículo não ultrapassa a capacidade total do veículo;
- As regras do trânsito são cumpridas.

Foi construído recentemente um modelo específico para a recolha de resíduos sólidos, que foi denominado de MCARP-LML (*Mixed Capacitated Arc Routing Problem with Limited Multi-Landfills*) [15]. Este modelo para além de considerar o problema num grafo misto, acrescenta uma novidade importante que é o facto de serem considerados vários pontos de descarga, em que cada um deles tem um número limite de descargas.

Os problemas de sectorização consistem em dividir uma grande região em regiões mais pequenas (sectores ou distritos), para facilitar a gestão de algumas actividades. Para modelar actividades associadas a ruas de grandes áreas urbanas, foi introduzido o SARP (*sectoring arc routing problem*) [16].

O SARP é definido num multigrafo misto, tendo como objectivo dividir os arcos necessários para criar um determinado conjunto de sectores, em que cada sector fica atribuído uma viatura, e resolve-se o problema de cada sector como um problema MCARP.

Segundo a autora [16], o SARP é mais realista que o MCARP porque a gestão do RSU é fácil com sectores pois: os indicadores produtivos são individuais por área, os motoristas têm que memorizar uma parte mais pequena das ruas, o balanceamento de cargas entre várias equipas pode ser melhor distribuído.

2.3.4 - Métodos de Resolução

Muitos problemas de optimização com importância prática e teórica consistem na procura de uma configuração óptima de um conjunto de variáveis para atingir alguns objectivos. Estes problemas dividem-se naturalmente em duas categorias [17]:

- Problemas com soluções que estão formuladas com variáveis reais;
- Problemas com soluções que estão formuladas com variáveis inteiras.

Na última categoria encontra-se uma classe de problemas chamados de problemas de optimização combinatória.

Os problemas de optimização de rotas são problemas típicos de Optimização combinatórias.

Devido à importância prática do problema de Optimização combinatória, muitos algoritmos foram desenvolvidos, e podem ser classificados de completos ou aproximados.

Os algoritmos completos encontram a solução óptima, e são métodos de potencial aplicação em problemas com solução em tempo polinomial.

Os algoritmos aproximados (heurísticas) sacrificam a solução óptima em função de obter uma solução tanto perto quanto possível do óptimo mas num tempo aceitável.

A aplicação de algoritmos complexos ou algoritmos aproximados na resolução de um determinado problema de optimização combinatória está relacionada, naturalmente, com a complexidade computacional do problema.

Até agora não foi encontrado um algoritmo polinomial exacto para qualquer algoritmo NP e portanto considera-se que $P \neq NP$. Deste modo, até que alguém consiga provar que $P=NP$, são aconselhadas heurísticas para a resolução de problemas NP [18].

Conforme visto nas secções anteriores, grande parte dos algoritmos de resolução dos problemas de rotas são NP-difícil. Deste modo, será justificável a utilização de métodos aproximados para obtenção de soluções aceitáveis.

Como um método heurístico devolve um resultado que se espera bom, mas não necessariamente óptimo, por vezes há a necessidade de avaliar o quão perto do óptimo está essa mesma solução, sendo a relaxação uma possível resposta para essa questão.

Num problema de minimização complexo, os limites superiores podem ser determinados por heurísticas, e os limites inferiores podem ser gerados por programação linear ou por relaxação Lagrangeana.

Os métodos heurísticos dividem-se em duas grandes classes: métodos de construção e métodos de melhoria (*local search*). Para resolução de um problema de rotas, utilizam-se numa primeira fase, heurísticas de construção. Estas heurísticas criam a partir do nada, rotas que satisfaçam os critérios definidos, até completarem a procura total. Numa segunda fase, são aplicadas heurísticas de melhoria, que, como o próprio nome indica, analisam possibilidades de optimização das rotas actuais. Para isso, procuram e comparam com os vizinhos uma solução que melhore a solução actual.

As meta-heurísticas são estratégias de alto nível que combinam técnicas de baixo nível para exploração do espaço de procura.

Uma das características mais importantes que caracteriza os algoritmos das meta-heurísticas é o número de soluções utilizadas ao mesmo tempo. Deste modo, as meta-heurísticas de populações (*population-based*) verificam em cada iteração um conjunto de soluções ou indivíduos. As meta-heurísticas EC (*Evolutionary computation*) e ACO (*Ant colony optimization*) são um exemplo de métodos baseados em populações. As meta-heurísticas individuais ou de trajetória (*trajectory methods*) focam-se em encontrar soluções individuais. São os casos dos métodos *Basic Local Search*, Arrefecimento simulado (*Simulated Annealing*), pesquisa Tabu (*Tabu Search*), GRASP (*Greedy Randomized Adaptive Search Procedure*), VNS (*Variable Neighborhood Search*), GLS (*Guided Local Search*) e ILD (*Iterated Local Search*).

Existem outros métodos de classificação das meta-heurísticas, tais como serem inspiradas na natureza, utilizarem ou não memória durante as iterações, etc.

A figura 2.5 mostra o enquadramento de alguma meta-heurísticas em função das classificações mais importantes.

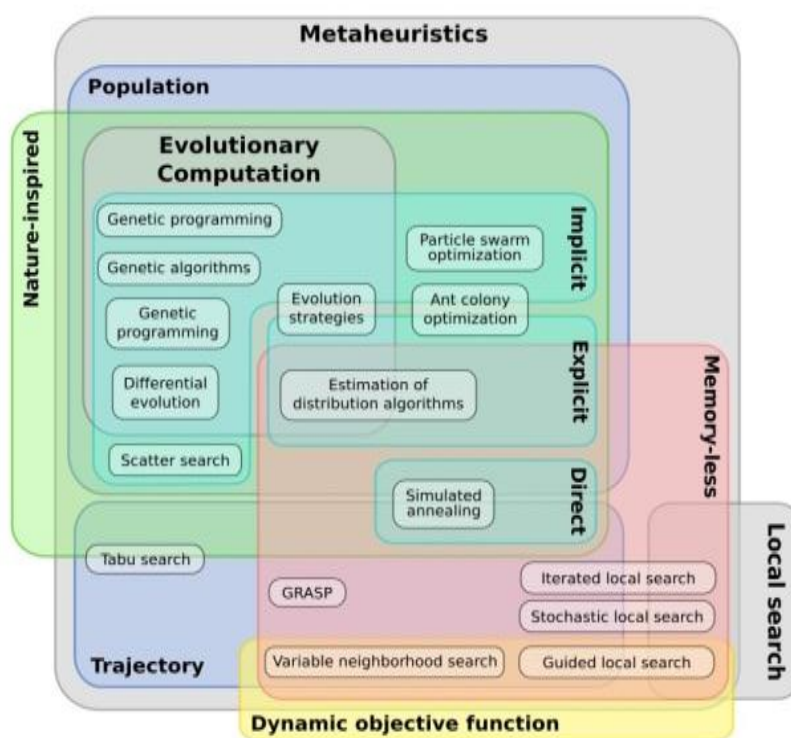


Figura 2.5 - Classificação das Meta-heurísticas [19]

2.4 - O Sistema de Recolha de RSU

2.4.1 - Caracterização do sistema

Foi escolhido para este trabalho o caso real da recolha de resíduos sólidos urbanos no concelho de Monção.

O concelho de Monção situa-se no norte de Portugal, pertence ao distrito de Viana do Castelo e à sub-região do Alto Minho. A norte tem o rio Minho que faz fronteira com a Galiza (Espanha). A oeste tem o concelho de Valença, a sudoeste o concelho de Paredes de Coura, a sul o concelho de Arcos de Valdevez e a este o concelho de Melgaço.

O concelho de Monção é composto por 33 freguesias e tem uma base territorial com cerca de 212 quilómetros quadrados. Em 2010 tinha 19257 pessoas residentes o que correspondia a uma densidade populacional de 91 pessoas/Km [4].

A figura 2.6 mostra a evolução da quantidade de RSU produzidos por dia no concelho de Monção, em função do destino final.

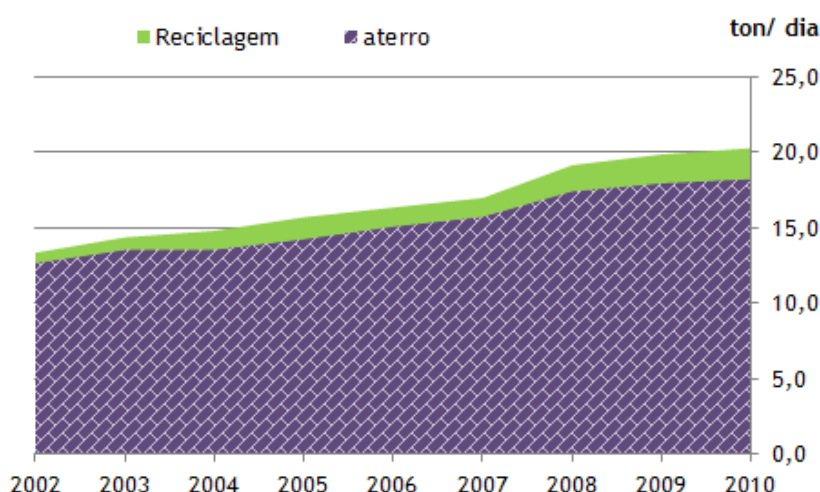


Figura 2.6 - Quantidade de RSU produzidos por dia no concelho de Monção (fonte [4])

Actualmente estão instalados no concelho de Monção cerca de 1120 contentores para deposição indiferenciada de resíduos, que correspondem a uma capacidade global de 779,4m³. Os contentores existentes são de 5000 litros no caso dos enterrados, e 120 l, 800l ou 1100 litros no caso de contentores normais. Os contentores de 800L correspondem ao tipo de contentores mais utilizado e atingem os 95%.

No concelho de Monção, a responsabilidade pela gestão do sistema “em baixa” de resíduos urbanos indiferenciados é do município, contudo a actividade está contratada à empresa privada Rumoflex [20]. O sistema multimunicipal Valorminho é responsável pelo sistema “em alta” de resíduos sólidos em seis concelhos, incluindo Monção.

A recolha de RSU no concelho de Monção tem uma periodicidade diária no centro urbano e nas freguesias limítrofes. Nas restantes freguesias é feita a recolha duas vezes por semana, e em determinados períodos de grande afluência de emigrantes e turistas a recolha é feita três vezes por semana.

Os resíduos recolhidos são depositados em dois locais definidos pela Valorminho, que actualmente são um aterro sanitário do Vale do Minho em S. Pedro da Torre - Valença e uma estação de transferência em Messegães - Monção.

As viaturas utilizadas na recolha e transporte de RSU são normalmente camiões com capacidade de 16m³ e com caixa compactadora de modo a poderem transportar mais quantidade de resíduos. Cada viatura é normalmente constituída por uma equipa de 3 elementos sendo um o condutor e ou outros dois cantoneiros.

A Valorminho impõe actualmente um número de descargas máximas por dia no aterro de Messegães.

Existem vários estudos referentes à optimização de RSU, que apresentam soluções adoptando tanto modelos de procura nos vértices como modelos de procura nos arcos, embora os últimos sejam em número notoriamente inferior.

Segundo um estudo de optimização de rotas [10] feito para a empresa Waste Management (WM), que é líder nos Estados Unidos da América na gestão de resíduos, foram criados três grupos distintos com diferentes perfis consoante o tipo de recolha:

- Recolha doméstica: Recolha porta-a-porta em habitações conforme figura 2.7. É elevado o número de recolhas por dia;
- Recolha Comercial: A recolha de resíduos resultante da actividade comercial envolve recolha de resíduos em contentores, restaurantes e pequenos edifícios de escritórios;
- Recolha Industrial: Grandes quantidades de resíduos que normalmente exigem uma viagem única por cliente.



Figura 2.7 - Recolha porta-a-porta

O modelo de recolha porta-a-porta de resíduos indiferenciados é pouco frequente em Portugal, sendo que o mais comum é a instalação de contentores enterrados (3 a 5 m³) em zonas urbanas, e contentores superficiais de 1m³ em zonas rurais.

Deste modo, se for analisada a recolha porta-a-porta, em que para cada rua existem vários contentores de pequenas capacidades, o modelo de problema com procura nos arcos é o que melhor se aplica pois pode-se considerar que o camião tem que percorrer a rua do lado direito da via, sendo que o peso do arco é a soma das capacidades previstas dos contentores.

Por outro lado, quando a localização é discreta dada a distância que os separa, e até podem ser recolhidos por diferentes camiões, pode considerar-se um problema distinto do anterior, com a procura nos vértices.

2.4.2 - Aplicações descritas na literatura

Problemas com procura nos vértices

Para o estudo de optimização de rotas na recolha de resíduos na cidade de Irbid (Jordânia)[21] foi aplicado o modelo do caixeiro-viajante (TSP) que é um problema típico de procura nos vértices. Para a resolução do problema, foram utilizados algoritmos genéticos reais em comparação com outros métodos de resolução. Foi concluído que os algoritmos genéticos reais produziram a solução melhor, pois resultou na rota de comprimento menor, sendo por isso um método a aconselhar no planeamento de rotas na recolha de resíduos.

A optimização de rotas na cidade de Atenas também foi objecto de análise [2]. Neste caso também foi adoptado o modelo do caixeiro-viajante (TSP), e como métodos de resolução foram utilizados dois, o algoritmo *Ant-Colony System* (ACS) e outro resultante de um programa informático, o ArcGIS Network Analyst. Os resultados mostram que ambos os métodos resultaram em rotas mais eficientes comparativamente com a rota implementada pelo município de Atenas, sendo a heurística *Ant-Colony System* a que forneceu a melhor rota. Contudo mostrou necessitar de muitas iterações para chegar a um resultado aceitável, o que implicou uma grande demora a calcular. Foi aconselhado pelos autores a utilizar como complemento à meta-heurística ACS, uma heurística que melhore a solução inicial, tal como a 2-opt.

Um modelo VRP simples foi utilizado num estudo optimização de circuitos de recolha de resíduos domésticos em zonas urbanas [22]. O objectivo do autor foi transformar um problema de múltiplo caixeiro-viajante (m veículos) em m problemas de caixeiro-viajante. Deste modo, foi adoptado uma combinação de dois algoritmos. O primeiro algoritmo aplicado foi programação linear binária, que teve como objectivo a distribuição de cargas por cada veículo. O segundo algoritmo serviu para a construção das rotas e neste caso foram aplicados 2 métodos diferentes para posterior comparação, que foi a meta-heurística do arrefecimento simulado (*simulated-annealing*) e a programação linear binária.

No caso da recolha do RSU nos Estados Unidos da América, foi realizado para a empresa Waste Management um estudo de planeamento de rotas para os resíduos provenientes de comércio/serviços [23]. Neste caso foi utilizado o modelo VRPTW simples, com o objectivo de minimizar os tempos de viagem. Foi implementado um algoritmo de duas fases, tendo como objectivos a primeira fase criar uma solução inicial, e a segunda fase melhorar a solução inicial. Na primeira fase foi aplicado o algoritmo *K-means-variant-clustering* juntamente o algoritmo *Greedy-Insertion*. Na segunda fase foi aplicado o algoritmo *Solomon's insertion* juntamente com a meta-heurística do Arrefecimento Simulado (*simulated-annealing*). Neste caso, o algoritmo mostrou ser bastante competitivo e do ponto de vista prático, atingiu os objectivos a que se propôs.

Em continuação do trabalho anterior, os mesmos autores criaram um algoritmo para o VRPTW mas considerando múltiplos aterros e com paragens para almoço dos trabalhadores [10]. Neste caso testaram dois algoritmos e compararam os seus resultados. O primeiro algoritmo testado foi uma versão evoluída do algoritmo *Solomon's insertion*. O segundo algoritmo testado foi o *Clustering-based waste collection* VRPTW. Após se obter uma solução inicial em ambos os algoritmos, foi aplicada a meta-heurística do Arrefecimento Simulado (*simulated-annealing* SA) para análise das potencialidades destes algoritmos na solução final.

Foi concluído que o algoritmo Clustering-based waste collection VRPTW trouxe melhores resultados juntamente com a meta-heurística SA.

Em [24] e tendo por base o mesmo modelo e os resultados do estudo anterior, foi aplicado um algoritmo genético multiobjectivo como método de resolução. Apesar do algoritmo não contemplar explicitamente o balanceamento de carga entre rotas e a compactação das rotas, no geral os resultados são superiores ao do estudo anterior.

Do mesmo modo, em [25] também foi utilizado o mesmo modelo e resultados do estudo [10]. O algoritmo proposto é constituído por duas fases. Numa primeira fase é construída a solução inicial e são definidos os conjuntos vizinhos. Numa segunda fase são aplicadas e comparadas 3 meta-heurísticas diferentes, a TS (*Tabu search*), VNS (*variable neighbourhood search*) e uma terceira baseada na VNS onde a pesquisa é feita via TS, apelidada de VNTS (*variable neighbourhood tabu search*). Os resultados apresentados mostram que a VNS é a meta-heurística que produz melhores resultados, superado todos os resultados de estudos anteriores.

Problemas com procura nos arcos

Uma versão do problema do carteiro chinês com restrições de capacidade (CCPP) foi aplicada ao contexto dos carteiros do Canadá [11]. Neste caso, não foram incluídas restrições de capacidade, mas sim limites de tempo, que impõem máximos no tempo dispensado em cada rota. O modelo está preparado para que cada rota tenha um tempo limite diferente.

Uma adaptação do CCPP foi usada na optimização da recolha de resíduos em Bruxelas [26]. Neste caso optaram por dividir o problema em duas fases, como objectivo de diminuir os custos individuais de transporte de cada fase, pois argumentam que se conseguem melhores resultados com um modelo multicritério em vez de um modelo com critério único. Numa primeira fase é determinado o terminal de recolha e avaliado a necessidade de um terminal intermédio entre as rotas de recolha e a incineração. Na segunda fase é avaliada a hipótese de redução das quantidades de estações de transferência. Foi utilizado o algoritmo *Branch and bound* para resolução dos problemas.

O CARP foi adoptado por Mourão ao problema da recolha de RSU [8]. Neste caso, foram acrescentadas restrições especiais que o fazem diferir de problemas CARP clássicos e tem características específicas, tais como a existência de um posto de serviço e a representação numa rede orientada. Face a estas diferenças, foi denominado o problema de PRRS (Problema de recolha de resíduos sólidos). Neste caso, foram determinados os limites inferiores do problema por relaxamento, e foram criadas três heurísticas construtivas com três fases, cada uma para comparação de resultados.

Um modelo baseado no multigrafo misto MCARP foi utilizado no estudo da recolha de resíduos no município de Castrovillari em Itália [27]. Foi utilizada a heurística de construção *cluster-first route second*. Ghiani concluiu que o método aplicado alcançou bons resultados mas referiu que poderia ter soluções ainda melhores se fosse aplicada uma heurística de melhoramento.

Na recolha de resíduos sólidos urbanos no município de Sant Boi de Llobregat em Barcelona [28], foi construído um modelo para o MCARPTC (*mixed capacitated arc routing problem with turn constraints*).

Neste caso, para a formulação do modelo foi transformado o problema de arcos (ARP) num problema de procura nos vértices. Deste modo foi considerado que o *generalized vehicle routing* (GVRP) é um modelo equivalente a MCARPTC, mas com procura nos nós. Para a resolução do problema foram utilizadas duas heurísticas ACO (*Ant Colony Optimization*) diferentes para comparação de resultados. Ambas as heurísticas construídas em duas fases, sendo a primeira de construção de rotas e pesquisa local, e a segunda fase para actualização de feromonas que informam e ajudam na construção de novas rotas. Os 2 algoritmos variam somente na heurística de construção, em que um usou a heurística NN (*nearest neighbor*) e o outro a heurística de 2 etapas NI (*nearest insertion*). Foi demonstrado que a heurística NI supera os resultados da NN em todas as vertentes e é uma mais-valia na optimização de rotas na recolha de resíduos.

Capítulo 3

Definição do modelo matemático

3.1 - Definições e notação

Os problemas de planeamento de rotas na recolha de resíduos são complexos de caracterizar devido às especificidades que cada um tem. Conforme já descrito nos capítulos anteriores, este trabalho pretende representar um estudo real de recolha de RSU.

Apesar da região do Alto Minho se enquadrar num meio menos urbano e sem grandes restrições de tráfego, entendeu-se que seria mais lógico e funcional representar o problema num grafo misto. Os vértices do grafo representam o conjunto de estaleiros, aterros, cruzamentos ou entroncamentos de ruas, e as ligações do grafo representam as ruas.

As ligações dum grafo misto traduzem uma visão mais real da rede de estradas pois são consideradas as ligações não direccionais chamadas de arestas, e as ligações direccionais chamadas de arcos. As arestas representam ruas de dois sentidos onde é possível o trânsito em qualquer sentido, e nas quais é possível se efectuar a recolha em *zig-zag*, isto é, a recolha simultânea dos dois lados da rua numa passagem única. Os arcos representam ruas de sentido único ou avenidas de dois sentidos mas que têm a obrigatoriedade de manter o sentido, não sendo possível a recolha em *zig-zag*. As arestas podem-se representar por dois arcos de sentido oposto.

As ruas que têm contentores de lixo são classificadas de ruas com procura ou obrigatórias, sendo a procura da ligação igual à soma da procura dos vários contentores.

Conforme se pode ver na figura 3.1, de uma forma geral foram criados vértices nos cruzamentos ou entroncamentos de ruas identificados pelo prefixo V, e arcos ou arestas fazendo as ligações dos vértices. As ligações com contentores, logo com procura, foram representadas a cor vermelho escuro e os arcos facultativos a cor amarelo claro. Os contentores estão identificados com um ícone de cor verde.

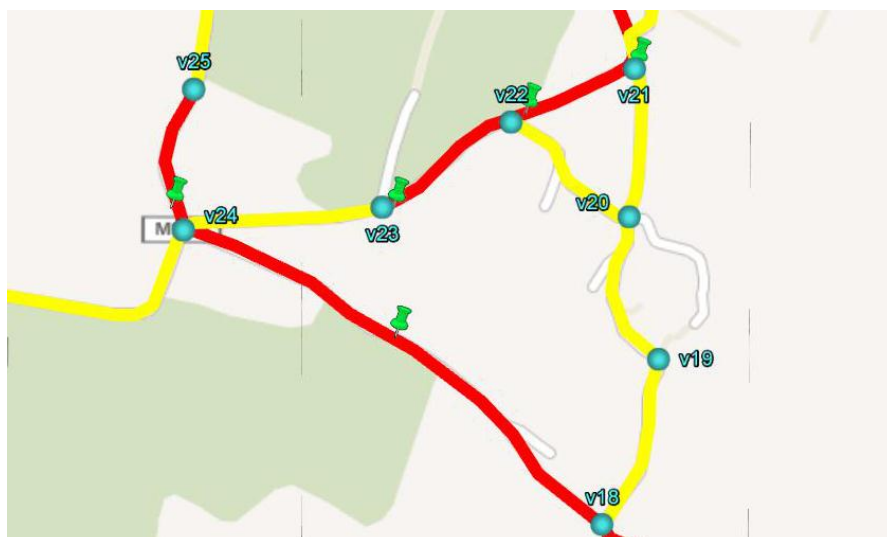


Figura 3.1 - Procura nos arcos no caso de estudo

De um modo geral, o problema de determinação de rotas no caso de estudo é classificado como um problema dum grafo misto com procura nos arcos com restrições de capacidade MCARP, embora tenha outras características específicas que o fazem diferenciar do modelo original. Uma das particularidades deste caso é o facto de existir por razões de espaço, um limite máximo de descargas diárias por aterro.

Foi desenvolvido recentemente por Rodrigues e Ferreira [15] um modelo denominado de MCARP-LML (*Mixed Capacitated Arc Routing Problem with Limited Multi-Landfills*). Este modelo representa fielmente a realidade do problema de rotas no Alto Minho e deste modo serviu como referência ao desenvolvimento da meta-heurística.

No MCARP-LML é atribuído a cada veículo um circuito que se realiza num turno de trabalho. O circuito começa com a partida do veículo do estaleiro, e a termina com a chegada ao estaleiro, vazio. Cada circuito é composto por várias viagens ou rotas, que efectuem a recolha ou serviço nos arcos com procura, tendo em consideração a capacidade do veículo, e começando e acabando num estaleiro ou aterro. Portanto a primeira viagem de um circuito começa no estaleiro e termina num qualquer aterro, e a última viagem do circuito é sempre entre um aterro e o estaleiro pois o veículo deve regressar vazio. Cada rua percorrida por um veículo sem efectuar recolha, tem um custo de passagem atribuído.

O objectivo do problema é encontrar um grupo de T viagens, que consiga ao menor custo passar uma única vez em todas as ligações obrigatórias e efectuar a recolha, satisfazendo a capacidade de carga dos veículos.

Designa-se por:

$G = (V^*, L)$ é um grafo misto composto pelos vértices V^* e L ligações.

D conjunto de vértices de estaleiros/armazéns sendo sempre o vértice 1.

B conjuntos de vértices dos aterros (considera-se que b representa um aterro específico, sendo $b=2,3,..B$)

$V' = D \cup B$ representa o conjunto dos seguintes vértices especiais

$V^* = V' \cup V$

$L = A \cup E$ onde E é o conjunto de arestas não direccionadas e A o conjunto de arcos direccionados.

$R = A_R \cup E_R$ representa o conjunto das ligações que têm procura, e que têm que ser servidas, com $A_R \in A$ e $E_R \in E$.

T é o número máximo de viagens permitido necessário para satisfazer toda a procura

Q é a capacidade de cada veículo

d_{ij} é o custo de passagem no arco $(i,j) \in L$

c_{ij} é o custo de serviço no arco $(i,j) \in R$

q_{ij} é a procura do arco $(i,j) \in R$

Q_T é a procura total, calculada pela expressão $Q_T = \sum_{(i,j) \in R} q_{ij}$

l_i é limite máximo de descargas no aterro $(i) \in B$

Na figura 3.2 está exemplificado um grafo de um circuito do MCARP-LML, onde o vértice 1 é o estaleiro e os vértices 2 e 3 são aterros. No aterro nº2 foram efectuadas duas descargas e atingiu o número máximo de descargas (l_2). A descarga seguinte já foi feita no aterro nº3. Este circuito foi percorrido em 4 viagens (vermelho, azul, laranja e verde). As ligações obrigatórias estão representadas a traço contínuo, e as ligações facultativas a traço interrompido. As arestas estão representadas com setas bidireccionais e os arcos com setas unidireccionais.

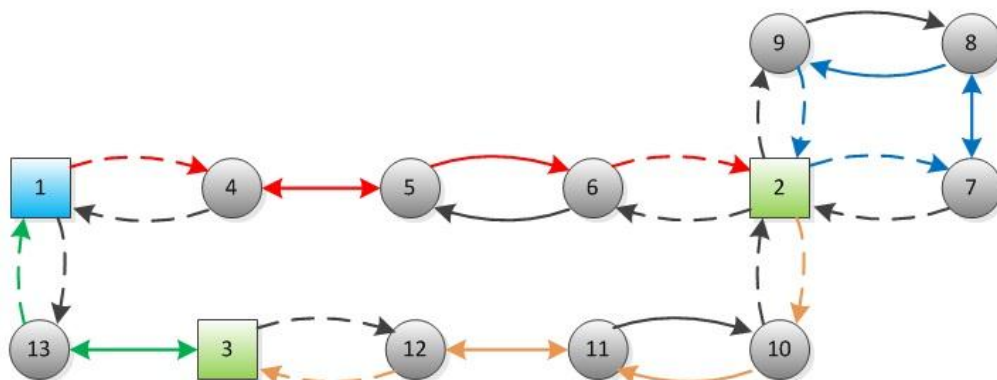


Figura 3.2 - Exemplo do MCARP-LML com 2 aterros

3.2 - Modelação em programação linear inteira mista

O MCARP-LML foi modelado como um problema de programação linear inteira mista, isto porque existem simultaneamente variáveis inteiras e variáveis binárias, conforme se pode ver na tabela 3.1.

Tabela 3.1 - Variáveis do modelo MCARP-LML

Variável	Tipo
$x_{ijb}^t = \begin{cases} 1, & \text{se } (i,j) \in R \text{ é servida pela viagem } t \text{ com descarga no aterro } b \\ 0, & \text{restantes casos} \end{cases}$	Binária
y_{ij}^t é o número de vezes que a ligação $(i,j) \in R$ é servida pela viagem t	Inteira
z_{ij}^t é a quantidade de lixo recolhido na viagem t após percorrer a ligação (i,j)	Real

Na recolha de resíduos sólidos, a cada veículo é atribuída uma zona ou circuito, na qual terá que servir em várias viagens T a procura total. Apesar do número de viagens ser um dado do problema, também pode ser calculado este minorante, através do quociente entre o somatório da procura total do circuito sobre a capacidade do veículo, adicionando mais um que representa o regresso ao estaleiro.

3.2.1 - Entradas

- a) Um grafo misto conexo é $G = (V^*, L)$, composto pelos vértices V^* e arcos L
- b) Cada ligação $(i,j) \in L$ tem um custo de passagem atribuído d_{ij} .
- c) Cada ligação $(i,j) \in R$ tem um custo de serviço c_{ij} .
- d) Cada ligação $(i,j) \in R$ tem uma procura q_{ij} .
- e) Cada aterro $b \in B$ tem um limite de visitas l_b .
- f) T viagens estão disponíveis.
- g) A capacidade dos veículos é Q .

3.2.2 - Saídas

- a) Conjunto óptimo de viagens L .
- b) Valor da função objectivo
- c) Os aterros visitados por cada viagem L .

3.2.3 - Objectivo

A função-objectivo do MCARP-LML é minimizar os custos total de todas as T viagens, que se repartem em custos de serviço e custos de passagem. Uma viagem corresponde a uma rota entre dois vértices pertencentes a V' (vértices de aterros ou estaleiro).

$$\text{Min} \sum_{t=1}^T \left(\sum_{(i,j) \in R} \sum_{b \in B} x_{ijb}^t \cdot c_{ij} + \sum_{(i,j) \in L} y_{ij}^t \cdot d_{ij} \right) \quad (3.1)$$

3.2.4 - Restrições

$$\sum_{t=1}^T \left(\sum_{(i,j) \in L} y_{ij}^t + \sum_{b \in V^*} \sum_{(i,j) \in R} x_{ijb}^t \right) = \sum_{t=1}^T \left(\sum_{(j,i) \in L} y_{ji}^t + \sum_{b \in V^*} \sum_{(j,i) \in R} x_{jib}^t \right), \quad \forall i \in V^* \quad (3.2)$$

$$\sum_{(i,j) \in L} y_{ij}^t + \sum_{b \in B} \sum_{(i,j) \in R} x_{ijb}^t = \sum_{(j,i) \in L} y_{ji}^t + \sum_{b \in B} \sum_{(j,i) \in R} x_{jib}^t, \quad \forall i \in V; t = 1, \dots, T \quad (3.3)$$

$$\sum_{t=1}^T \sum_{b \in B} x_{ijb}^t = 1, \quad \forall (i,j) \in A_R \quad (3.4)$$

$$\sum_{t=1}^T \sum_{b \in B} (x_{ijb}^t + x_{jib}^t) = 1, \quad \forall (i,j) \in E_R \quad (3.5)$$

$$\sum_{b \in V': (i,b) \in L} y_{ib}^t = 1, \quad t = 1, \dots, T \quad (3.6)$$

$$\sum_{b \in V': (b,j) \in L} y_{bj}^t = 1, \quad t = 1, \dots, T \quad (3.7)$$

$$\sum_{t=1}^T \sum_{b \in B} x_{ijb}^t = 0, \quad \forall (i,j) \in \bar{R} \quad (3.8)$$

$$x_{ij1}^t = 0, \quad \forall (i,j) \in R; t = 1, \dots, T \quad (3.9)$$

$$\sum_{(i,b) \in L} y_{ib}^t = \sum_{(b,j) \in L} y_{bj}^{t+1}, \quad t = 1, \dots, T-1; \forall b \in B \quad (3.10)$$

$$\sum_{t=1}^T \sum_{(b,j) \in L} y_{bj}^t \leq l_b, \quad \forall b \in B \quad (3.11)$$

$$\sum_{(i,b) \in L} y_{ib}^t + \sum_{(b,j) \in L} y_{bj}^t \leq 2, \quad t = 1, \dots, T; \forall b \in V' \quad (3.12)$$

$$\sum_{(i,j) \in L} z_{ij}^t - \sum_{(j,i) \in L} z_{ji}^t = \sum_{(i,j) \in R} \sum_{b \in B} q_{ij} \cdot x_{ijb}^t, \quad \forall i \in V; t = 1, \dots, T \quad (3.13)$$

$$\sum_{(i,b) \in L} z_{ib}^t = \sum_{(i,j) \in R} q_{ij} \cdot x_{ijb}^t, \quad t = 1, \dots, T; \forall b \in B \quad (3.14)$$

$$\sum_{(b,j) \in L} z_{bj}^t = 0, \quad t = 1, \dots, T; \forall b \in B \quad (3.15)$$

$$z_{ij}^t \leq Q \left(y_{ij}^t + \sum_{b \in B} x_{ijb}^t \right), \quad \forall (i,j) \in L; t = 1, \dots, T \quad (3.16)$$

$$x_{ijb}^t \in \{0,1\}, \quad \forall (i,j) \in L; \forall b \in B; t = 1, \dots, T \quad (3.17)$$

$$z_{ij}^t \geq 0, \quad \forall (i,j) \in L; t = 1, \dots, T \quad (3.18)$$

$$y_{ij}^t \geq 0, \quad integer, \forall (i,j) \in L, t = 1, \dots, T \quad (3.19)$$

Condições:

- Eqs. (3.2)-(3.3) Para se garantir a continuidade das viagens ao longo dos vértices do grafo.
- Eqs. (3.4)-(3.5) É feita a recolha em cada aresta ou arco com procura uma única vez.
- Eqs. (3.6)-(3.7) Garante que cada viagem termina e começa num aterro ou no estaleiro.
- Eq. (3.8) Uma ligação facultativa não é servida.
- Eq. (3.9) Todas as viagens com serviço depositam os resíduos sempre em aterros e nunca no estaleiro (vértice 1).
- Eq. (3.10) Quando o veículo descarrega a carga numa determinada viagem num aterro, a próxima viagem tem que começar nesse aterro
- Eq. (3.11) O número de visitas ou descargas permitidas em cada aterro é limitado.
- Eq. (3.12) Cada viagem começa e acaba num aterro ou estaleiro.
- Eqs. (3.13)-(3.14) Restrições de conservação do fluxo, isto é, garantem que após a recolha de resíduos num arco ou numa viagem, é feita a actualização local ou global da carga recolhida com os valores da procura.
- Eq. (3.15) O valor da quantidade de RSU recolhido é 0 quando o veículo sai do aterro.
- Eq. (3.16) Finalmente garante-se que a quantidade de RSU recolhido não poder excede a capacidade do veículo.
- Eqs. (3.17) -(3.19) Declarações das variáveis

Capítulo 4

Método de resolução

O objectivo principal desta investigação foi a aplicação de uma meta-heurística para a resolução de um problema real que foi modelado como um MCARP-LML. Tanto quanto se sabe, não há até ao momento qualquer trabalho desenvolvido neste modelo. Deste modo foi adoptada a meta-heurística da Colónia de Formigas (*Ant Colony Optimization* - ACO).

4.1 - Introdução ao algoritmo da colónia de formigas

4.1.1 - Inspiração biológica

O termo *Swarm Intelligence* (SI) descreve-se como sendo uma propriedade que um determinado grupo de agentes tem, em se comportarem de forma auto-organizada no seu ambiente de forma a cumprir determinados objectivos. Esta forma de interacção pode ser observada em sistemas artificiais ou em sistemas naturais tais como as colónias de formigas, cardumes de peixes, bactérias, etc, onde geralmente as espécies são pouco inteligentes mas conseguem resolver colectivamente problema complexos.

“Quer se esteja a falar de formigas, abelhas, pombos ou renas, os ingredientes de um comportamento colectivo inteligente são um controlo descentralizado, resposta a estímulos locais, princípios básicos simples levam a uma estratégia perspicaz de lidar com a complexidade”[29].

A figura 4.1 contém duas imagens exemplificativas de comportamentos SI na natureza. Na imagem da esquerda observa-se a organização coordenada de um cardume de peixes de forma a estarem mais protegidos de predadores. A imagem do lado direito mostra o comportamento SI que um bando de aves toma, para defesa dos ataques da ave de rapina (predador).

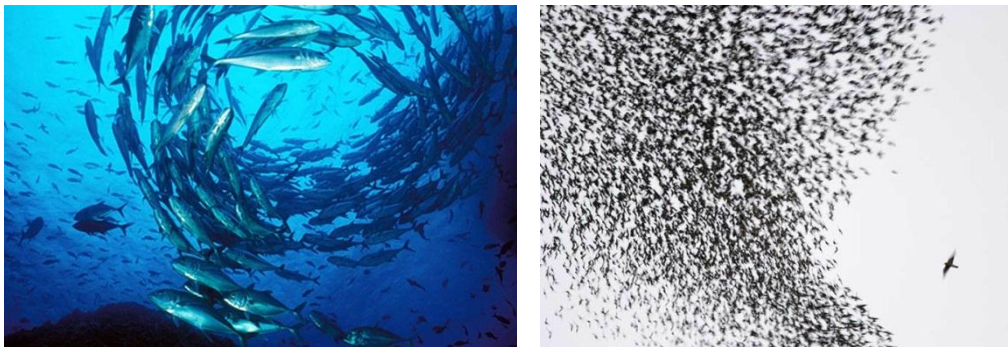


Figura 4.1 - Comportamentos *Swarm Intelligence* na natureza [29]

As colônias de formigas são um exemplo de um grupo de insectos auto-organizados que apresentam um comportamento coordenado nas suas funções vitais, tais como a recolha de alimento, a construção do ninho ou a defesa da colónia.

A Estigmergia é uma forma de comunicação através da modificação física de certas propriedades do meio ambiente reconhecidas pelos membros do grupo. Em certas sociedades de formigas é possível observar-se a estigmergia, pois ao procurarem alimento estabelecem uma comunicação indirecta através da deposição de uma substância química no chão denominada feromona. A substância libertada nos trilhos é reconhecida pelas formigas seguintes que decidem numa forma probabilística os percursos a seguir, isto é, quanto maior a quantidade de feromonas mais será a probabilidade de uma formiga escolher esse percurso. Este feedback positivo ajuda as formigas a determinarem o caminho mínimo entre o ninho e a fonte de alimento.

A evaporação de feromonas acontece naturalmente com a decorrer do tempo e é a forma natural de feedback negativo do trilho. Para que um caminho continue válido e com bons resultados é necessário haver uma constante deposição de feromonas, pois se isso não acontece é porque ou o caminho é claramente desvantajoso ou o destino final já não tem alimento.

Os algoritmos *Ant Colony Optimization* (ACO) são inspirados nestes comportamentos organizacionais de *swarm intelligence* para a resolução de problemas de optimização combinatória. Para isso são criadas formigas artificiais com critérios de estigmergia próprios que de uma forma simples irão resolver problemas complexos.

4.1.2 - Primeiras heurísticas ACO

No geral os algoritmos ACO são métodos estocásticos de busca com memória, baseados na construção repetitiva de soluções. A população é constituída por formigas artificiais ou agentes que individualmente constroem uma solução possível para um determinado problema. Cada decisão é feita de forma aleatória onde está associada a cada hipótese de escolha uma probabilidade que advém da combinação entre a memória estigmergética (feromonas) e uma informação heurística.

A informação memorizada pode ser actualizada de três modos diferentes:

- On-line, isto é, deposição de feromonas após a formiga percorrer o arco;
- On-line com atraso: deposição de feromonas após a formiga percorrer o caminho completo;
- Off-line: a deposição de feromonas é actualizada no final de todas as formigas terminarem.

O algoritmo original foi publicado em 1992 e denominado de Ant-System [30]. Este método foi desenvolvido inicialmente para o TSP e já foi incluída uma rotina de evaporação de feromonas. Quanto à actualização de feromonas, todas as formigas contribuem para a actualização da memória e foram propostas três variantes que diferem na forma de actualização das feromonas.

- Ant-quantity: Actualização de feromonas *On-line* com valores de feromonas dependente da distância do arco
- Ant-density: Actualização de feromonas *On-line* com valores de feromonas independente da distância do arco
- Ant-cycle: Actualização de feromonas *On-line com atraso*, com valores de feromonas dependente do valor da distância total

O algoritmo Ant-Cycle obteve resultados substancialmente melhores que os outros dois, principalmente em problemas mais complexos.

Surgiram mais tarde outros algoritmos baseados nas colónias de formigas que são variantes do Ant System, onde se realçam os seguintes:

-Max-Min Ant System [31]-[32]: Este algoritmo difere do Ant System em dois aspectos principais: a actualização de feromonas é feita só pela formiga com a melhor solução e os valores de feromonas não podem ultrapassar os limites definidos máximo e mínimo.

- Ant Colony System [33]-[34]: Este algoritmo vem acrescentar uma rotina de actualização local de feromonas que traz diversidade na procura de novos caminhos. A actualização de feromonas geral é feita em cada iteração somente pela melhor solução.

A primeira meta-heurística ACO foi inicialmente proposta por Dorigo, Caro e Gambardella [35], e foi caracterizada como sendo constituída por dois componentes principais que actuam de forma local:

- Criação de formigas e actividades;
- Evaporação de feromonas: Processo de decremento automático do rasto de feromonas com o tempo.

Foi definido um terceiro componente desta meta-heurística que é opcional e actual de forma global, denominado de *Daemon Actions*. Nesta rotina poderão ser efectuadas actualizações de feromonas off-line e poderão existir sub-rotinas de optimização local, que melhoram a solução obtida.

A figura 4.2 mostra uma descrição dos passos desta meta-heurística.

```
procedure ACO_meta-heuristic()
  while (termination_criterion_not_satisfied)
    schedule_activities
      ant_generation_and_activity();
      pheromone_evaporation();
      daemon_actions(); {optional}
    end schedule_activities
  end while
end procedure
```

Figura 4.2 - Pseudocódigo da meta-heurística ACO de Dorigo, Caro e Gambardella [35]

4.2 - Meta-heurística ACO-LML

Foi adaptado ao nosso problema uma meta-heurística elaborada para o CARP [36]. Visto se terem sido feitas várias alterações de modo a adaptá-la ao MCARP-LML e ao caso real em estudo, decidiu-se denominar a nosso método de ACO-LML.

A meta-heurística ACO-LML consiste em três fases principais: inicialização, geração da população inicial e algoritmo ACO. Foram ainda criados dois algoritmos importantes que são executados na segunda e terceira fases: melhoria da solução e divisão em viagens possíveis. Estes algoritmos serão explicados em pormenor nos seguintes secções.

Tabela 4.1 - Principais notações utilizadas

Notação	Descrição
f	Número de formigas (população)
f_e	Número de formigas elitistas ($f_e < f$)
μ	Formiga actual ou ranking
i_t	Iteração actual
m_{it}	Número máximo de iterações
a_i	Último arco a ser servido
a_j	Arco com procura ainda por servir
q	Número aleatório entre 0 e 1
q_{heu}	Probabilidade atribuída à escolha do critério 1
q_{fer}	Probabilidade atribuída à escolha do critério 2
$P_{a_i a_j}^{\mu}$	Probabilidade de uma formiga μ se deslocar entre o arco a_i e o arco a_j
$W_{a_i a_j}$	Caminho mínimo entre o arco a_i e o arco a_j
$n_{a_i a_j}$	Valor da informação heurística (desejabilidade) entre o arco a_i e o arco a_j
$\tau_{a_i a_j}$	Valor memorizado de feromona entre o arco a_i e o arco a_j
n_s	Número de iterações sem melhoria para evaporação feromonas
ρ	Coeficiente de evaporação de feromonas
M_d	Caminho mínimo de tamanho máximo, $M_d = \max(W_{a_i a_j})$
S^{μ}	Valor actual da solução para a formiga μ
n_{ap}	Nº de formigas que usam somente a memória (feromonas) para decisão

É apresentada na figura 4.3 uma notação em pseudocódigo para a meta-heurística ACO-LML:

```
procedure ACO-LML
  //Inicialização
  Inicializa_parametros();
  Leitura_grafo();
  Caminhos_minimos();

  //Geração da população inicial
  Algoritmo_PSRC-LML();

  //Algoritmo ACO
  Actualizacao_feromonas();
  for  $i_t=1$  to  $m_{it}$  do
    Construcão_solucoes();
    Evaporacao_feromonas()
    Actualizacao_feromonas();
  end for
end procedure
```

Figura 4.3 - Pseudocódigo da meta-heurística ACO-LML

4.2.1 - Inicialização

Nesta fase são inicializados determinado parâmetros tais como o número de iterações, o número de formigas, a taxa de evaporação, etc.

Foi criada uma rotina para leitura do grafo. Esta rotina após ler o grafo de um ficheiro de texto, analisa o número de arcos obrigatórios.

A aplicação desenvolvida conta também com uma rotina extra implementada especificamente para o caso prático. Após a leitura do grafo, é criado automaticamente um ficheiro de texto passível de importação no Google Earth com o mapa real do grafo, como se pode ver na figura 4.4. Os arcos com procura estão representados a cor vermelho escuro e os facultativos a cor amarelo claro.

De forma a otimizar o desempenho da meta-heurística, foram criadas também nesta fase duas matrizes que contêm as rotas e os custos dos caminhos mínimos entre todos os vértices do grafo, pois durante a execução dos algoritmos há a necessidade de por diversas vezes se saber essa informação. Esta rotina foi elaborada com base no algoritmo de Dijkstra.

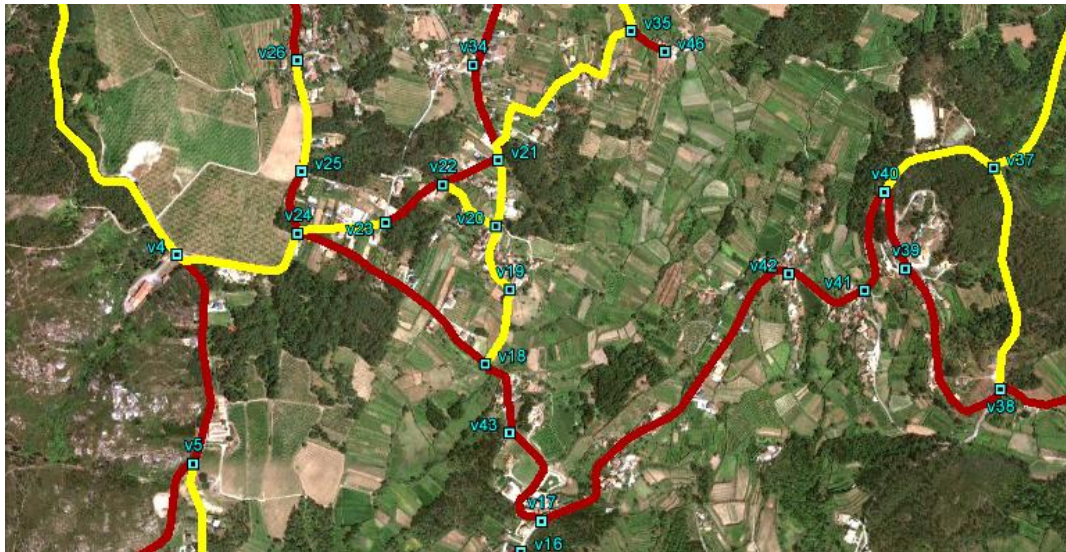


Figura 4.4 - Visualização no Google Earth do grafo do caso real

4.2.2 - Geração da população inicial

Uma grande parte das meta-heurísticas necessita de um solução inicial ou de um conjunto de soluções iniciais como ponto de partida. As meta-heurísticas baseadas nas colônias de formigas são um dos casos em que isso se aplica, pois necessitam de uma população de soluções iniciais (formigas) para determinarem os valores iniciais de feromonas.

O algoritmo desenvolvido para criação do conjunto de f soluções iniciais foi denominado de PSRC-LML e está representado na figura 4.5 em pseudocódigo:

```

procedure Algoritmo_PSRC-LML
  for  $\mu=1$  to  $f$  do
    //Criação da rota "gigante" através da heurística PSRC
    while  $S'' < R$ 
      Selecciona um arco obrigatório  $i$  ainda não servido
      Adiciona arco  $i$  e arcos de passagem até chegar a  $i$ 
      Remove procura nesse arco. Se aresta remove procura no sentido oposto
    end while

    Verifica_soluções();
    Melhoria_local();
    Divisão_rotas_viaveis();
    Verifica_clones();
  end for
  Ordena_soluções();
  Inicializa matriz de feromonas();
end procedure
  
```

Figura 4.5 - Pseudocódigo do algoritmo PSRC-LML

A heurística Path-Scanning (PS) [37] foi desenvolvida para problemas modelados como CARP, e caracteriza-se por ser um método do tipo “guloso” em que é construída a viagem através da adição de um arco com procura de cada vez, e escolhendo os arcos mais promissores, até a capacidade do veículo ser atingida.

Foram definidos pelo autor cinco critérios de escolha do arco mais promissor:

1. Minimizar o rácio C_{ij}/q_{ij}
2. Maximizar o rácio C_{ij}/q_{ij}
3. Minimizar o custo do vértice v_j para o nó v_0
4. Maximizar o custo do vértice v_j para o nó v_0
5. Se o veículo está com a capacidade actual a menos de metade segue o critério 4, se não escolhe o critério 3,

onde v_i é o último vértice visitado na viagem actual e $[v_i, v_j]$ é o um arco ainda por servir.

O problema é resolvido 5 vezes utilizando os cinco critérios, e a solução final corresponde à melhor solução encontrada das cinco.

Se todos os arcos incidentes ao último vértice v_i não tiverem procura, são escolhidos os arcos com procura que ficam à distância mínima do arco actual. Caso exista mais do que um arco com procura à distância mínima de v_i , é utilizado novamente um dos 5 critérios de selecção.

Mais recentemente foi modificada a heurística Path-Scanning por Belenger [13], de forma a gerar mais e melhores resultados. Foram propostos dois métodos de escolha para o próximo arco a servir nos casos em que existam vários arcos com procura à mesma distância mínima do arco servido. Um dos métodos caracteriza-se pela escolha aleatória de critérios (*Path Scanning with Random Criterion* - PSRC) e o outro pela escolha aleatória de ligações (*Path Scanning with Random Link* - PSRL). Estes métodos traduzem-se por gerar mais soluções que o anterior, e consequentemente, aumentar a probabilidade de obter melhores soluções, pois a quantidade de soluções resultantes é função do número de iterações. Os resultados demonstraram que nos grafos testados o PSRC obteve claramente melhores resultados que o PSRL.

Na primeira fase da nossa heurística PSRC-LML é ignorada a capacidade dos veículos para a construção de uma rota “gigante”, isto é, uma rota que tem início no estaleiro, passa por todos os arcos com procura e termina no último arco com procura adicionado. Foi utilizada a heurística PSRC para esta etapa.

Para cada solução obtida, há uma validação de resultados e depois é executada uma rotina de melhoria local, descrita em pormenor no ponto 4.2.4. De seguida é dividida a rota “gigante” em rotas que não ultrapassem a capacidade dos veículos, conforme procedimento descrito no ponto 4.2.5.

Foi aplicada uma rotina de detecção de soluções idênticas chamada de “verifica_clones”, para evitar ter soluções iniciais semelhantes, pois uma população de várias soluções iniciais iguais reduz substancialmente o desempenho da meta-heurística. Uma solução é considerada “clonada” se tiver as seguintes características:

- Igual custo total das viagens;
- Número de vértices das viagens igual;
- O arco do meio for igual.

Se for detectada uma solução “clonada” esta é considerada inválida. Em certos grafos é quase impossível determinar uma população sem “clones”. Deste modo, foi implementado um limite máximo de tentativas de $10*f$ para encontrar uma solução nova. Caso seja atingido este limite, na construção das próximas viagens será atribuída uma probabilidade de 5% à escolha aleatória do próximo arco com procura, sendo possível deste modo conseguir uma população diversa.

No final da heurística PSRC-LML e após estar criada uma população de soluções iniciais é feita uma inicialização da matriz de feromonas, aplicado um valor igual entre todos os arcos com procura, pois como há ligações entre arcos com procura que não são consideradas nas soluções iniciais, se tivessem um valor nulo de feromonas poderiam nunca ser visitadas. Utilizou-se a seguinte fórmula para inicializar a matriz de feromonas:

$$\tau_0 = \frac{f}{S_{pior}} \quad (4.1)$$

4.2.3 - Algoritmo ACO

Os algoritmos ACO são constituídos em regra por três componentes principais: a construção de novas soluções, a actualização de feromonas e a evaporação de feromonas. Conforme descrito figura 4.3 também foram utilizados na nossa meta-heurística os três componentes referidos e cada um deles vai ser descrito de seguida.

Construção de novas soluções

Esta fase é das mais importantes no algoritmo ACO pois determina como é que as formigas irão escolher os próximos arcos com procura a serem servidos. Na figura 4.6 descrito como este algoritmo está estruturado:

```

procedure Construcão_solucoes
  for  $\mu=1$  to  $f$  do
    while  $S^\mu < R$ 
      Gera número aleatório  $q$ 
      //Escolha de um critério
      if  $\mu > n_{ap}$  And  $q > (1-q_{heu})$ 
        Selecciona um arco obrigatório  $i$  ainda não servido conforme critério 1
        //Fórmula 4.2
      elseif  $q < q_{Fer}$ 
        Selecciona um arco obrigatório  $i$  ainda não servido conforme critério 2
        //Fórmula 4.4
      else
        Selecciona um arco obrigatório  $i$  ainda não servido conforme critério 3
        //Fórmula 4.5
      end if
      Selecciona um arco obrigatório  $i$  ainda não servido conforme critério
      Adiciona arco  $i$  e arcos de passagem até chegar a  $i$ 
      Remove procura nesse arco. Se aresta remove procura no sentido oposto
    end while
    Verifica_soluções();
    Melhoria_local();
    Divisão_rotas_viaveis();
  end for
  Ordena_soluções();
end procedure

```

Figura 4.6 - Pseudocódigo do algoritmo “Construcão_solucoes”

A construção da solução é feita segundo um método de três critérios proposto por Santos, Rodrigues e Current [36], pois demonstrou ter melhor desempenho que os restantes métodos ensaiados. A escolha entre os vários critérios é aleatória e atribuiu-se uma probabilidade a cada critério. A excepção é no critério 1 pois não basta a probabilidade ser favorável ao critério, também é necessário que a formiga actual seja superior ao parâmetro n_{ap} , em que este parâmetro representa o número de formigas que usam somente a memória (feromonas) para decisão do próximo arco com procura a escolher.

A escolha do próximo arco a servir no critério 1 é com base na informação da heurística local $n_{a_i a_j}$, também chamada de desejabilidade. A desejabilidade diminui com o aumento da distância entre os arcos a_i e a_j , sendo portanto seleccionado o arco que tiver maior desejabilidade, ou no caso de haver mais do que um arco com procura com valores iguais de desejabilidade máxima é escolhido um deles aleatoriamente:

$$P_{a_i a_j}^{\mu} = \begin{cases} 1, & \text{se } j = \arg \max \{n_{a_i a_j}\} \\ 0, & \text{restantes casos} \end{cases}, \quad (4.2)$$

onde a desejabilidade é determinada pela equação 4.3:

$$n_{a_i a_j} = \frac{M_d - w_{a_i a_j} + 1}{M_d + 1}. \quad (4.3)$$

Se a probabilidade sair favorável ao critério 2, o próximo arco a ser servido é o que tem maior concentração de feromonas, ou no caso de haver mais do que um arco com procura com valores iguais de feromonas é escolhido um deles aleatoriamente:

$$P_{a_i a_j}^{\mu} = \begin{cases} 1, & \text{se } j = \arg \max \{\tau_{a_i a_j}\} \\ 0, & \text{restantes casos} \end{cases}. \quad (4.4)$$

O critério 3 é o que torna possível a procura de novas soluções, pois é atribuída uma probabilidade a todos os arcos com procura ainda por servir que é proporcional à quantidade de feromonas:

$$P_{a_i a_j}^{\mu} = \frac{\tau_{a_i a_j}}{\sum_{a_s \in N_{a_i}^{\mu}} \tau_{a_i a_s}}. \quad (4.5)$$

A população de formigas (f) está dividida em dois grupos: as formigas elitistas (f_e) e as formigas não elitistas. As formigas elitistas representam as melhores soluções da população, e em cada iteração só actualizam as suas soluções se houver uma melhoria dos resultados. As formigas não elitistas mudam as suas soluções em cada iteração com o valor da última iteração, independentemente da qualidade do resultado. Este fenómeno faz com a matriz de feromonas fique relativamente dispersa de modo a que se possa encontrar uma solução ainda melhor.

Actualização de feromonas

As feromonas são atribuídas às ligações “virtuais” entre arcos com procura, e às ligações iniciais entre o estaleiro e os arcos com procura mais próximos. Foi ponderada a colocação de feromonas nos arcos com procura ou até mesmo em todos os arcos. Contudo verificou-se que o algoritmo era pouco eficiente e foi colocada de parte esta hipótese.

Ao contrário de Santos, Rodrigues e Current [36] e por este modelo ser representado num grafo misto, decidiu-se que as feromonas só seriam depositadas no sentido do arco percorrido pela formiga, e não em ambos os sentidos.

Foi utilizada a fórmula de actualização de feromonas de Lacomme, Prins e Tanguy [38], em que a quantidade de feromona é inversamente proporcional ao custo da solução, e proporcional ao inverso do ranking da formiga no conjunto de soluções:

$$\Delta\tau_{a_i a_j}^\mu = \frac{\mu * \left(\frac{M_d - 1}{f - 1}\right) + \frac{f - M_d}{f - 1}}{S^\mu} . \quad (4.6)$$

Esta equação também considera propriedades do grafo tais como a máxima distância entre dois arcos com procura.

Evaporação de feromonas

A evaporação de feromonas é normalmente formulada pela seguinte expressão:

$$\tau_{a_i a_j} \leftarrow (1 - \rho) * \tau_{a_i a_j} . \quad (4.7)$$

Contudo para que certos caminhos não fiquem desactivados permanentemente, foi decidido estabelecer-se um limite mínimo igual a τ_0 , no qual a partir desse ponto não há decréscimo do valor de feromonas.

4.2.4 - Melhoria das soluções

Os algoritmos de pesquisa local são frequentemente utilizados nas meta-heurísticas pois melhoram a qualidade das soluções [14]. Conforme referido no estado da arte, muitos autores apontaram como medidas de melhoria das suas soluções a implementação de métodos de melhoria local.

Foram implementadas dois algoritmos de melhoria das soluções, que são executados após a construção de cada solução, tanto para a fase inicial de construção da população inicial, bem como após o algoritmo ACO. O primeiro algoritmo é um método de melhoria global, isto é, procura otimizar troços da viagem, e foi elaborado a pensar no caso real em estudo. O segundo algoritmo é um método simples de pesquisa local.

O primeiro algoritmo foi inspirado na análise das soluções do grafo do caso real. Por este grafo ser constituído por um número grande de ruas sem procura, o circuito final resultante por vezes engloba o agrupamento de arcos sem procura que só trazem o aumento dos custos.

Este método começa por dividir a solução final em dois tipos de grupos classificados de “zonas de recolha” e “zonas de passagem”. Uma zona de passagem inicia-se sempre que sejam detectadas três passagens entre arcos sem haver recolha, e termina quando for detectada uma passagem com recolha. A figura 4.7 mostra um exemplo dessa divisão.

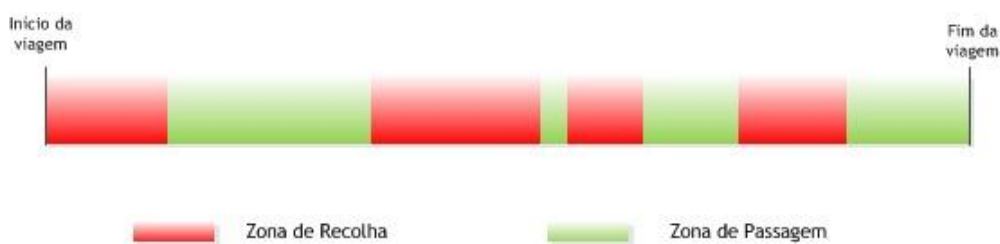


Figura 4.7 - Algoritmo de melhoria global - divisão de zonas

O objectivo deste algoritmo é aumentar a dimensão das zonas de recolha e reduzir as zonas de passagem. O passo seguinte é encontrar a zona de recolha mais pequena da viagem, e tentar enquadrá-la noutra zona de recolha, e depois refazer as ligações mínimas necessárias.

São identificados os vértices de início e fim da zona de recolha a mover. De seguida procuram-se nas restantes zonas de recolha da viagem, pela existência desses vértices ou pela existência de vértices com ligações directas aos vértices da zona de recolha a mover. Face aos resultados encontrados, é escolhida a ligação de menor custo. Depois são refeitas as ligações da viagem. A figura 4.8 exemplifica essa mudança.

O algoritmo termina quando não forem detectadas melhorias nos enquadramentos.

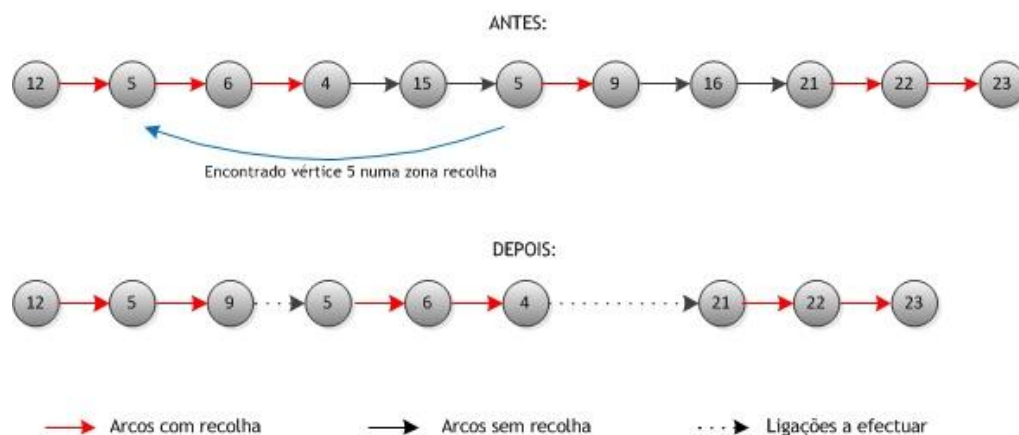


Figura 4.8 - Algoritmo de melhoria global - exemplo do movimento

No segundo algoritmo implementado, é feita uma pesquisa na viagem por arcos repetidos em que um deles tenha recolha. Depois é feita uma troca do momento da recolha desse arco para o arco de passagem. Numa primeira análise os custos são iguais após a troca, mas de seguida é executado o primeiro algoritmo que analisa se existe a possibilidade de melhoramento da solução. A figura 4.9 seguinte mostra um exemplo deste algoritmo.

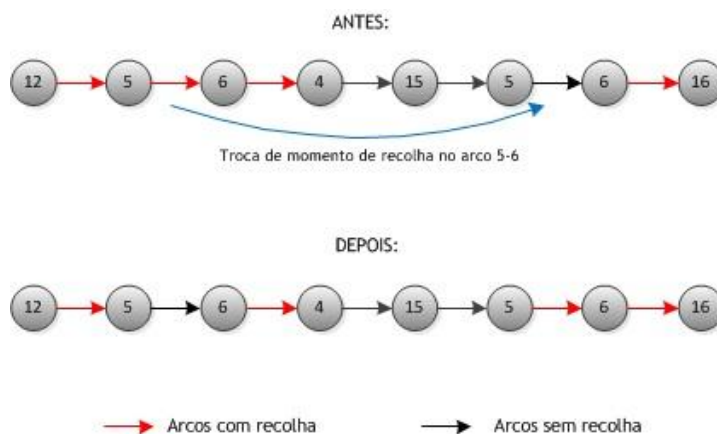


Figura 4.9 - Algoritmo de melhoria local - exemplo do movimento

4.2.5 - Divisão em rotas viáveis

Para tornar o único circuito gigante em várias viagens praticáveis de modo a cumprir as restrições de capacidade dos veículos, e garantindo os limites de descargas nos aterros conforme os requisitos do modelo, foram analisadas duas opções: A adaptação de heurísticas existentes e a criação de nova heurística.

Foi analisada a heurística de Ulusoy [39] que é um algoritmo de divisão de rotas criado para o CARP. De um modo geral este método consiste nas seguintes quatro etapas:

- 1) Encontrar uma rota “gigante”
- 2) Dividir a rota “gigante” em subrotas correspondentes a rotas com viabilidade de capacidade dos veículos
- 3) Selecção do conjunto de rotas de menor custo
- 4) Repetir os passos anteriores até ser atingido um objectivo

A etapa 1 do algoritmo de Ulusoy já estava criada pelos algoritmos de criação da rota “gigante”. Foram então desenvolvidas as etapas 2 a 4 deste método.

A divisão de rotas em sub-rotas faz-se através da criação de um grafo auxiliar com o número de vértices igual ao número de arcos com procura, e onde se cria cada arco com procura com uma ligação de chegada para o estaleiro e de partida para o estaleiro. Os restantes arcos desse grafo auxiliar são criados em função do somatório dos arcos com procura consecutivos até que atinjam a capacidade limite dos veículos. Após o grafo auxiliar estar criado, é escolhido a divisão da rota “gigante” através do caminho mínimo entre o primeiro e o último vértice do grafo auxiliar.

Como este método foi desenvolvido para o CARP, tentou-se fazer a adaptação deste algoritmo MCARP-LML, pois o nosso modelo difere do CARP no seguinte:

- A partida é do Estaleiro
- A chegada ao Estaleiro é com a carga vazia
- Existem múltiplos aterros
- Cada aterro tem uma capacidade limitada

Apesar do método de Ulusoy apresentar bons resultados no modelo CARP, a adaptação ao MCARP-LML demonstrou ser complicada e inviável. Deste modo optou-se pela criação de um método novo de partição da rota global que satisfizesse o nosso modelo.

Cada viagem tem que ter pelo menos uma descarga, pois um veículo nunca pode chegar com carga ao estaleiro. A última descarga é a primeira a ser considerada neste método, e o aterro é seleccionado aleatoriamente de acordo com as respectivas disponibilidades, pois foi a solução que demonstrou melhores resultados.

As restantes descargas são feitas tendo em consideração a margem económica de capacidade. A margem económica corresponde à diferença entre a capacidade dos veículos nas várias rotas menos a procura total. Sabendo que a carga limite do veículo é Q , e a margem económica é ME , o veículo vai efectuar a descarga ao aterro quando a capacidade estiver entre ME e Q . A escolha do momento exacto da descarga é conforme o custo do caminho mínimo entre o arco servido e os aterros com disponibilidade mais o regresso até ao próximo arco servido.

Como se pode observar na figura 4.10, tem que ser feita a descarga entre os vértices 4 e 6. Estão disponíveis dois aterros e a descarga é efectuada escolhendo o caminho mínimo entre as quatro rotas possíveis.

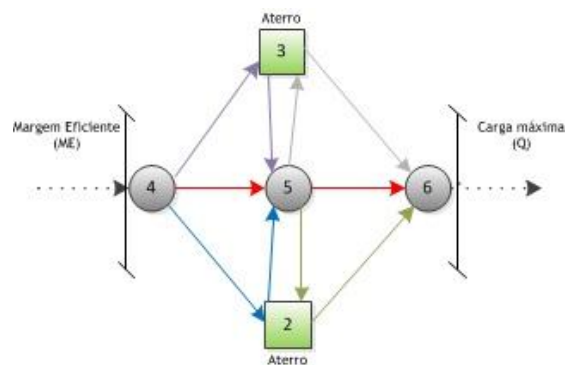


Figura 4.10 - Algoritmo de divisão em rotas viáveis

Capítulo 5

Resultados computacionais

5.1 - Instâncias e parâmetros

Foi utilizado o IBM ILOG CPLEX versão 12.4 para obtenção de soluções através de um método exacto. Deste modo foi possível comparar os resultados obtidos entre o método exacto e os métodos aproximados desenvolvidos. O CPLEX conseguiu determinar soluções óptimas para as instâncias mais simples, contudo nas restantes instâncias de maior complexidade o CPLEX devolveu apenas a melhor solução que encontrou antes de atingir a capacidade limite de memória do computador.

Todas as heurísticas foram desenvolvidas em Visual Basic 2010. Todos os métodos foram executadas num computador com processador Intel Core I3-2100 3,1GHz com 4GB de memória RAM e sistema operativo Windows 7 Pro SP1.

Testaram-se dois grupos de instâncias que estão disponíveis para descarregar no seguinte endereço: http://www.inescporto.pt/~amr/Limited_Multi_Landfills/. O primeiro grupo é bastante referenciado na literatura e surge de uma adaptação das instâncias *gdb* ao MCARP-LML [15], que se caracterizam por serem grafos somente com arestas ou arcos bidireccionais. O segundo grupo contém uma instância referente ao grafo real do caso estudo.

A tabela 5.1 mostra as principais características dos grafos ensaiadas, e foi utilizada a seguinte simbologia para as tabelas:

- ID: Identificação da instância
- T: número de viagens
- #V*: número total de vértices (com aterros, estaleiros e vértices normais)
- #R: número de arcos/arestas obrigatórias
- #F: número de arcos/arestas facultativas
- Q: Capacidade do veículo
- #B: Número de aterros
- $[\text{Max } l_i]_{i=1, \dots, \#B}$: Número máximo de descargas no aterro *i*

Tabela 5.1 - Principais características das instâncias testadas

ID	T	#V*	#R	#F	Q	#B	[Max l_i] _{i=1,...,#B}
GDB ₁ _LML	6	15	22	6	5	2	[3,3]
GDB ₂ _LML	7	15	26	6	5	2	[4,4]
GDB ₃ _LML	6	15	22	6	5	2	[3,3]
GDB ₄ _LML	5	14	19	6	5	2	[3,3]
GDB ₅ _LML	7	16	26	6	5	2	[3,3]
GDB ₆ _LML	6	15	22	6	5	2	[3,3]
GDB ₇ _LML	6	15	22	6	5	2	[3,3]
GDB ₈ _LML	11	30	46	6	27	2	[5,5]
GDB ₉ _LML	11	30	51	6	27	2	[5,5]
GDB ₁₀ _LML	5	15	25	6	10	2	[2,2]
GDB ₁₁ _LML	6	25	45	6	50	2	[3,3]
GDB ₁₂ _LML	8	16	23	6	35	2	[4,4]
GDB ₁₃ _LML	7	13	28	6	41	2	[4,4]
GDB ₁₄ _LML	6	10	21	6	21	2	[3,3]
GDB ₁₅ _LML	5	10	21	6	37	2	[2,2]
GDB ₁₅ _LML1	5	10	21	6	37	2	[1,4]
GDB ₁₅ _LML2	5	10	21	6	37	3	[2,2,2]
GDB ₁₅ _LML3	5	10	21	6	37	4	[1,1,1,2]
GDB ₁₆ _LML	6	11	28	6	24	2	[3,3]
GDB ₁₇ _LML	6	11	28	6	41	2	[3,3]
GDB ₁₈ _LML	6	12	36	6	37	2	[3,3]
GDB ₁₉ _LML	4	11	11	6	27	2	[2,2]
GDB ₂₀ _LML	5	14	22	6	27	2	[2,2]
GDB ₂₁ _LML	7	14	33	6	27	2	[3,3]
GDB ₂₂ _LML	9	14	44	6	27	2	[5,5]
GDB ₂₃ _LML	11	14	55	6	27	2	[6,6]
AltoMinho	3	49	31	31	18	2	[1,3]

Numa primeira fase houve um estudo do software GLPK que é uma aplicação para resolução exacta de problemas em programação linear. Este software apresenta uma linguagem de modelação denominada “GNU Math Prog” que é bastante intuitiva para a programação das formulações matemáticas. Contudo quando são executadas as instâncias referidas anteriormente, o software rapidamente atinge o seu limite de ruptura.

Deste modo, todas as instâncias foram executadas num software bem mais poderoso e relativamente conhecido, que é o Cplex. Estes testes pretenderam para além da determinação dos valores das soluções e dos respectivos tempos despendidos, a avaliação do comportamento e desempenho que uma aplicação que resolve problemas de forma exacta tem.

Conforme referido na secção anterior, a meta-heurística desenvolvida contém vários parâmetros passíveis de ajuste, de modo a atingir os melhores resultados. Os parâmetros utilizados estão representados na seguinte tabela 5.2.

Tabela 5.2 - Parâmetros usados nos algoritmos

Notação	Descrição	Valor
f	Número de formigas (População)	#R
f_e	Número de formigas elitistas ($f_e < f$)	#R/3
ρ	Coeficiente de evaporação de feromonas	0,25
n_s	Número de iterações sem melhoria para evaporar feromonas	10
q_{heu}	Probabilidade atribuída à escolha do critério 1	0,1
q_{fer}	Probabilidade atribuída à escolha do critério 2	0,1
n_{ap}	Nº de formigas que usam momentaneamente a memória (feromonas) para decisão	#R/2
m_{it}	Número máximo de iterações	200

Após várias análises ao comportamento da meta-heurística em função dos parâmetros escolhidos, pode-se concluir que com os valores definidos na tabela anterior os resultados melhoraram.

Um dos factores parâmetros que tem grande influência nos resultados é o tamanho da população (f). Uma população grande aumenta a probabilidade de encontrar uma solução boa ou ótima, contudo traz maior dispersão de resultados e torna a meta-heurística mais lenta. Uma população pequena torna as soluções mais próximas do ótimo, contudo não conseguem evoluir o esperado. Encontrar uma população em número equilibrado é um factor de sucesso dos algoritmos. Os artigos que inspiraram este trabalho [14, 36] consideraram uma população inicial de 60 e 30 elementos respectivamente. Foi considerada por nós uma abordagem diferente, isto é, definiu-se a população variável. Pretendeu-se deste modo encontrar uma relação entre a complexidade dos grafos e o tamanho da população, e a relação mais lógica foi considerar o número de indivíduos da população uma função do número de arcos obrigatório, isto é, $f = K * R$.

Foram feitos testes para $k=0,5$ e $k=1$, e $k=2$ e os resultados estão na tabela 5.3. Pode-se concluir que a escolha acertada para k é o valor 1 pois atinge o mesmo valor com k igual a dois, mas com um tempo consumido substancialmente menor. Com base nestes dados foram decididos optar-se por $k=1$, isto é, $f=R$.

Tabela 5.3 - Resultados da determinação do número de elementos da população

k	f	Heurística PSRC-LML		Meta-heurística	
		Custo	Tempo (s)	Custo	Tempo (s)
0,5	16	13186,00	2,64	13145,00	52,79
1	31	13186,00	5,58	13044,00	106,85
2	62	13186,00	9,86	13044,00	190,82

Quando ao factor de evaporação, foram feitos testes com valores de ρ superiores 0 e verificou-se que o aumento da taxa de evaporação conduziu a soluções de pior qualidade. Deste modo a actualização periódica de feromonas em cada iteração não foi considerada. Contudo foi verificado que se passadas várias iterações sem melhoria, ao ser feito nesse momento a evaporação de feromonas os resultados iriam melhorar. Neste sentido chegou-se a conclusão que as soluções obtidas pela meta-heurística melhoram para n_s igual a 10 e ρ igual a 0,25.

As probabilidades atribuídas a q_{heu} e a q_{fer} foram iguais à que Santos, Rodrigues e Current [36] utilizaram no seu algoritmo. Também neste caso se fizeram testes para encontrar a melhor relação entre as probabilidades e os resultados, e conclui-se que os melhores resultados surgiram quando se atribui probabilidades de 0,1 a q_{heu} e a q_{fer} .

Quanto ao número formigas que usam somente as feromonas para decisão (n_{ap}), optou-se por escolher um parâmetro variável em função da quantidade de arcos requerido tal com f . O valor que gerou melhores resultados foi com $n_{ap} = \frac{R}{2}$.

O aumento do número máximo de iterações pode melhorar o resultado, mas é certo que o tempo despendido para o cálculo também cresce proporcionalmente. Considerou-se que um valor máximo de 200 iterações com sendo o valor razoável para encontrar soluções boas para as instâncias.

Em certas tabelas de resultados é apresentado o parâmetro desvio médio. O desvio médio representa a afastamento médio percentual obtido entre as soluções dos algoritmos aproximados desenvolvidos e a melhor solução encontrada pelo CPLEX. Este indicador calcula-se segundo a equação 5.1:

$$Desvio\ médio = \frac{\frac{\sum_{\mu=1}^N |S_{heurística}^{\mu} - S_{CPLEX}|}{N}}{S_{heurística}} * 100 \quad , \quad (5.1)$$

onde N é o número de elementos da população e S^{μ} é o valor actual da solução para a formiga μ .

5.2 - População Inicial

Este algoritmo serve de ponto de partida para uma meta-heurística ACO-LML, e tem como principal objectivo criar uma população inicial de soluções. O desempenho deste método mede-se pela proximidade dos resultados obtidos em relação às soluções determinadas pelo CPLEX, e pela variedade de soluções da população.

Os resultados resumidos da eficácia deste algoritmo estão presentes na tabela 5.4:

Tabela 5.4 - Resultados da heurística PSRC-LML

Instâncias	Desvio médio PSRC-LML (%)			Diversidade média (%)	Tempo médio (s)
	f	f_e	Min(f)		
GDB	12,47	7,98	4,08	100,00	0,40
AltoMinho	12,01	5,17	1,64	100,00	7,42

Pode-se verificar que a população inicial de formigas (f) está em média 12% acima dos valores obtidos pelo CPLEX. Este valor considera-se aceitável face à quantidade de elementos da população. Verifica-se que para a população de formigas elitistas o desvio médio diminui cerca de 50% para as instâncias “GDB”, e diminui 232% para a instância “AltoMinho”.

O valor mínimo da heurística PSRC-LML ficou 4% acima da melhor solução encontrada pelo CPLEX para as instâncias “GDB” e para a instância “AltoMinho” o valor situou-se já muito próximo da solução obtida pelo método exacto.

Quanto à diversidade média que representa a variação do número de diferentes soluções, esta rotina demonstrou ser totalmente eficaz, isto é, gerar a totalidade das soluções diferentes, comprometendo ligeiramente o tempo despendido para esse cálculo.

Em geral os tempos gastos a determinar as soluções são relativamente baixos, à excepção do instância “AltoMinho” que demorou cerca de sete segundo a determinar as soluções, isto devido à natureza do grafo, visto ter muitos vértices e também por ser difícil encontrar rotas diferentes.

5.3 - Meta-heurística

A meta-heurística implementada demonstra no geral um melhoramento das soluções face ao algoritmo de construção da população inicial, o PSRC-LML. Foi implementada uma meta-heurística dedicada ao MCARP-LML, e tanto quanto se sabe, não há até ao momento qualquer trabalho desenvolvido para a resolução deste problema, o que torna difícil a avaliação deste método aproximado.

A meta-heurística foi executada 3 vezes para cada instância, e os resultados são apresentados de seguida.

5.3.1 - Instâncias “GDB”

Os resultados obtidos nas instâncias “GDB” estão resumidos na tabela 5.5.

O tempo médio despendido para calcular as melhores rotas ficou em cerca de 68 segundos, sendo a instância “GDB₁₉_LML” a mais rápida demorando três segundos, e a instância “GDB₉_LML” a mais lenta demorando seis minutos.

A meta-heurística apresenta um desvio médio de 3% face aos valores obtidos pelo CPLEX, o que revela ser bastante eficiente. Aliás, para cinco instâncias foram alcançados as soluções óptimas nas três experiências, o que um factor bastante optimista.

Também um dado surpreendentemente positivo é o facto da meta-heurística ACO-LML ter descoberto duas soluções melhores que as determinadas pelo CPLEX. São o caso das instâncias “GDB₈_LML” e “GDB₉_LML”.

Pode-se observar que os resultados em média melhoraram até 11% face às soluções iniciais, contudo há 6 instâncias que não demonstram melhorias.

No anexo A é possível ver os resultados detalhados destas experiências.

Tabela 5.5 - Resultados obtidos nas instâncias GDB

ID	f	f _e	CPLEX		PSRC-LML		ACO-LML		Desvio médio ACO-LML
			Custo	Tempo	Custo	Tempo	Custo	Tempo	%
GDB ₁ _LML	22	7	<u>310</u>	677,1	365	0,2	327	20,4	5,10
GDB ₂ _LML	26	8	<u>347</u>	5147,4	393	0,3	372	29,9	6,64
GDB ₃ _LML	22	7	<u>286</u>	28,7	353	0,3	316	20,4	9,59
GDB ₄ _LML	19	6	<u>293</u>	185,4	314	0,3	298	13,1	1,68
GDB ₅ _LML	26	8	376	2206,3	430	0,3	397	31,8	5,21
GDB ₆ _LML	22	7	<u>317</u>	1666,9	324	0,3	<u>317</u>	16,9	0,00
GDB ₇ _LML	22	7	<u>329</u>	555,3	376	0,1	345	19,0	4,73
GDB ₈ _LML	46	15	388	30593,5	396	1,5	396	267,3	2,02
GDB ₉ _LML	51	17	378	27343,6	387	1,7	387	360,9	2,24
GDB ₁₀ _LML	25	8	<u>296</u>	1,1	325	0,3	301	25,4	1,77
GDB ₁₁ _LML	45	15	<u>414</u>	2746,1	439	0,3	439	190,8	5,69
GDB ₁₂ _LML	23	7	472	1871,2	550	0,2	487	21,2	3,15
GDB ₁₃ _LML	28	9	565	2811,8	596	0,1	573	35,4	1,34
GDB ₁₄ _LML	21	7	<u>123</u>	4,0	125	0,1	<u>123</u>	14,2	0,00
GDB ₁₅ _LML	21	7	<u>78</u>	1,9	84	0,1	<u>78</u>	15,5	0,00
GDB ₁₅ _LML1	21	7	<u>78</u>	0,3	84	0,1	<u>78</u>	15,3	0,00
GDB ₁₅ _LML2	21	7	<u>76</u>	0,6	79	0,2	<u>77</u>	24,4	1,72
GDB ₁₅ _LML3	21	7	<u>70</u>	0,6	76	0,1	73	24,3	3,67
GDB ₁₆ _LML	28	9	<u>149</u>	4,3	161	0,3	154	33,6	3,46
GDB ₁₇ _LML	28	9	<u>115</u>	2,0	126	0,3	119	37,1	3,09
GDB ₁₈ _LML	36	12	<u>187</u>	2,8	191	0,3	190	73,1	1,58
GDB ₁₉ _LML	11	3	<u>77</u>	1,0	<u>77</u>	0,1	<u>77</u>	3,3	0,00
GDB ₂₀ _LML	22	7	<u>150</u>	0,9	168	0,2	153	21,2	2,17
GDB ₂₁ _LML	33	11	<u>184</u>	1078,9	192	0,4	190	57,1	2,99
GDB ₂₂ _LML	44	14	232	1671,4	245	0,7	245	147,9	5,18
GDB ₂₃ _LML	55	18	276	1471,0	287	1,1	287	246,1	3,94
Média				3079,8		0,4		67,9	2,96

Notas: Tempo em segundos; Soluções a sublinhado representam soluções ótimas.

5.3.2 - Instância “AltoMinho”

O objectivo principal desta investigação foi a aplicação de uma meta-heurística para a resolução de um problema real, que neste caso é identificado pela instância “AltoMinho”.

Esta instância representa um mapa de um sector de recolha periódica, onde estão fielmente identificadas as ruas obrigatórias e as ruas facultativas.

O grafo deste caso real é muito diferente dos apresentados nas instâncias “GDB”, pois assemelha-se aos grafos identificados no RPP, visto o número de arcos/arestas de passagem ser elevado, o que dificulta a eficácia dos algoritmos.

Os resultados obtidos estão resumidos na tabela 5.6.

Tabela 5.6 - Resultados obtidos na instância “AltoMinho”

Média									Desvio médio
ID	f	f _e	CPLEX		-LML		ACO-LML		ACO-LML
			Custo	Tempo	Custo	Tempo	Custo	Tempo	%
AltoMinho	31	10	12973	7,1	13441	7.4	13067	74.7	0.72

Notas: Tempo em segundos; Soluções a sublinhado representam soluções óptimas.

A meta-heurística demora em média menos de oitenta segundos para devolver uma solução para esta instância. É possível atingir-se um custo total de 13035, o que representa um custo inferior a 0,5% em relação à solução óptima determinada pelo CPLEX.

No anexo A estão disponíveis os resultados detalhados das várias experiências.

Nas três figuras seguintes é possível analisar o comportamento da meta-heurística ao longo das várias iterações. A figura 5.1 mostra uma representação gráfica do ranking das formigas para as soluções inicial e após 50 iterações. Verifica-se que a solução inicial devolve uma população toda diferente de 31 formigas, e que ao fim de 50 iterações já há quatro soluções de custo inferior à melhor solução inicial.

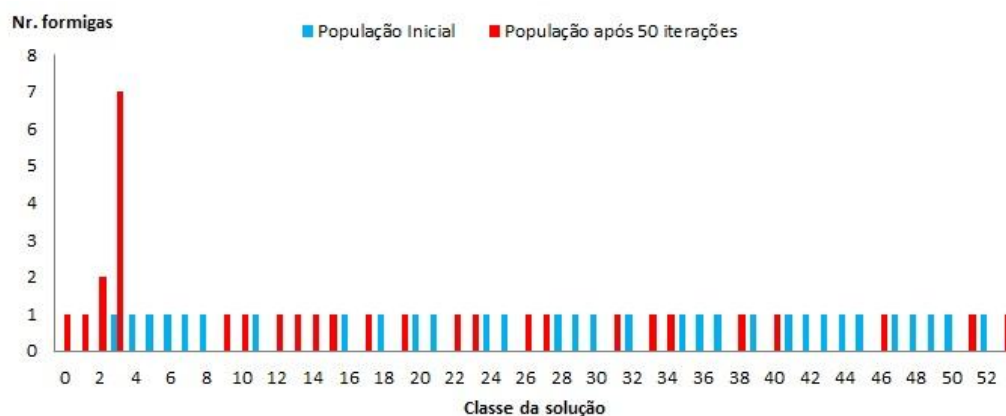


Figura 5.1 - Comportamento da meta-heurística em 50 iterações

A figura 5.2 mostra a evolução após 100 iterações, onde já se encontraram 8 soluções melhores que a solução inicial.

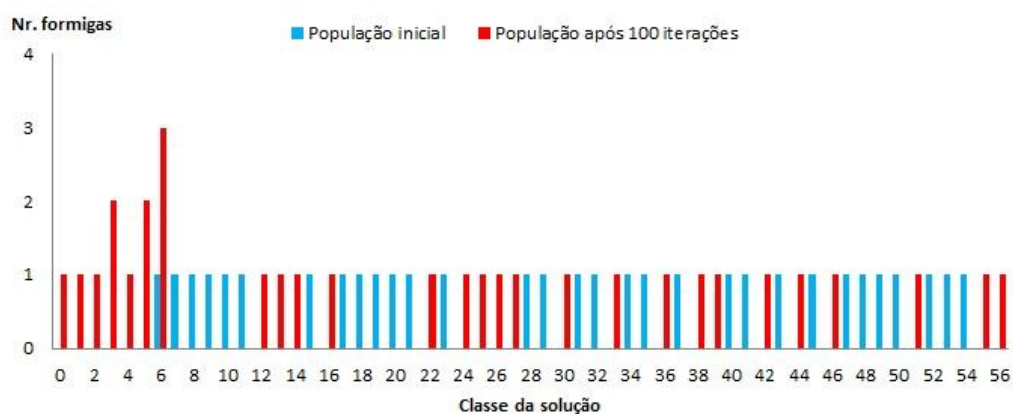


Figura 5.2 - Comportamento da meta-heurística em 100 iterações

Ao final do ciclo de iterações, que neste caso é 200, foram descobertas 10 soluções melhores que a solução inicial, e comprovando deste modo a eficácia do algoritmo, representado na figura 5.3.

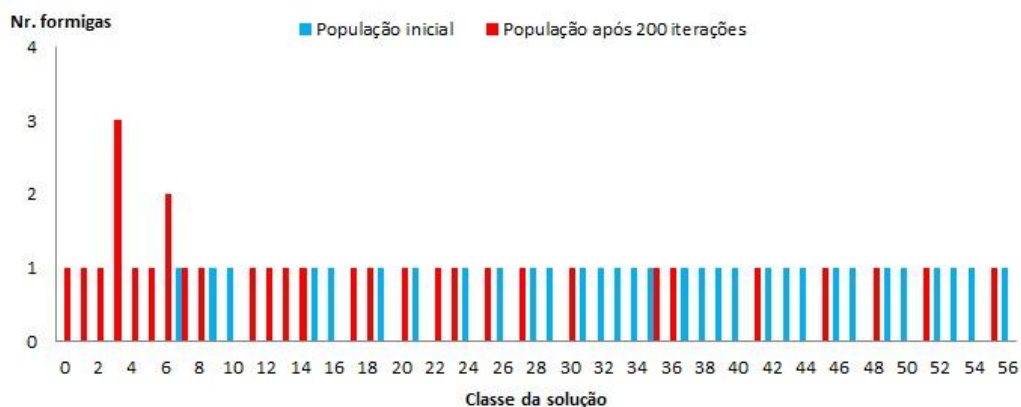


Figura 5.3 - Comportamento da meta-heurística em 200 iterações

Nesta instância foram consideradas 31 formigas e 10 formigas elitistas. O comportamento das formigas não elitistas é visível nos gráficos anteriores, pois estas formigas actualizam as suas soluções em cada iteração, independentemente da qualidade das soluções. Este processo de actualização tem a desvantagem de criar soluções muito díspares do óptimo, mas que podem ajudar o algoritmo a descobrir novos caminhos.

Como este trabalho foi desenvolvido com base no caso real, foi desenvolvida para esta instância uma rotina extra que exporta a melhor solução para um ficheiro passível de importação no Google Earth. É atribuída uma cor a cada viagem deste modo pode-se analisar graficamente cada pormenor da solução obtida.

Nas figuras 5.4 e 5.5 estão representadas na forma de mapa a solução do modelo exacto, e a solução determinada pela meta-heurística. Pode-se observar que as diferenças são mínimas, e só um algoritmo de melhoria local poderá atingir esse resultado. As diferentes cores representam diferentes viagens.

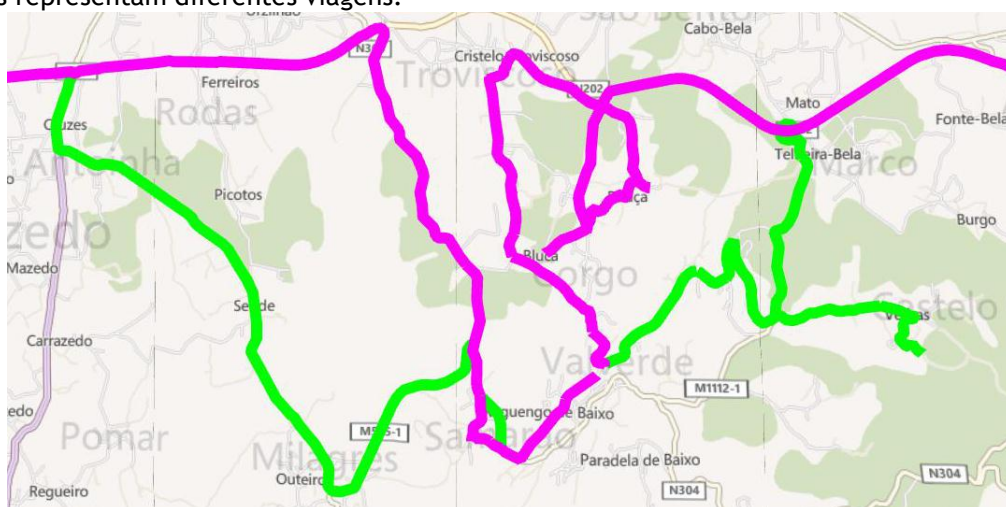


Figura 5.4 - Mapa da viagem efectuada com solução óptima determinada pelo CPLEX

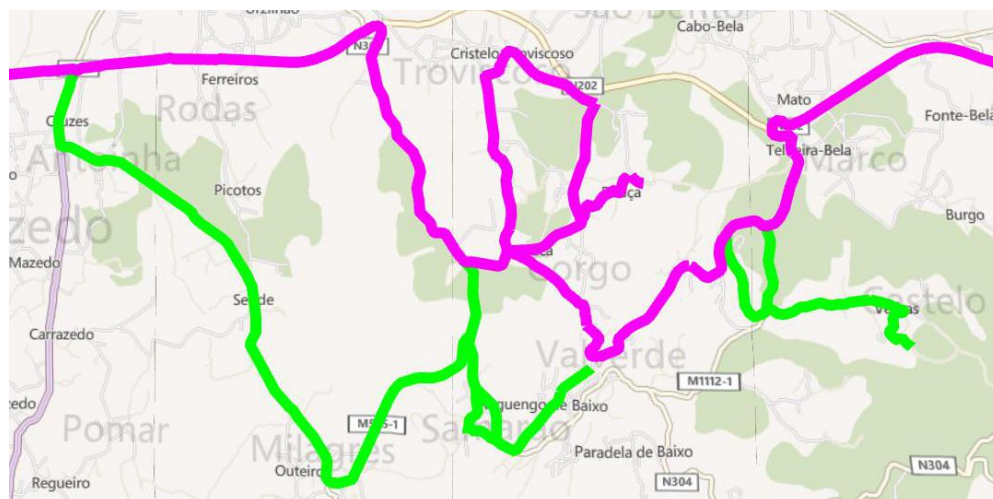


Figura 5.5 - Mapa da viagem efectuada com solução melhor da meta-heurística ACO-LML

Capítulo 6

Conclusões

A presente tese foi dedicada ao planeamento de rotas na recolha de resíduos. Este problema foi modelado como um MCARP-LML, que é um MCARP com vários aterros com limitações de descargas. Este é um problema actual e ainda em estudo, que se manifesta na região do Alto Minho.

Como se trata de um problema de difícil resolução, foi proposto um método aproximado que solucionasse o MCARP-LML, com resultados aceitáveis. Este método foi denominado de ACO-LML e caracterizou-se por ser uma meta-heurística com inspiração nas heurísticas da colónia de formigas (ACO).

A meta-heurística desenvolvida acrescentou um grande contributo ao planeamento eficaz de rotas na recolha de resíduos, pois não há até ao momento nenhum método aproximado desenvolvido para resolução deste problema real.

Os resultados computacionais foram avaliados em 26 instâncias teóricas e numa instância real denominada de “AltoMinho”, que é o resultado de um levantamento de contentores de RSU numa determinada zona populacional.

Não havendo dados comparativos de outros métodos aproximados de resolução do MCARP-LML, a única comparação possível foi com os dados que o método exacto obteve.

Os resultados dos métodos exactos demonstraram claramente as desvantagens que estes métodos têm nas resoluções de problemas de optimização combinatória, embora nas instâncias pequenas estes métodos são bastante eficientes e devolvem uma solução óptima em tempo reduzido.

Os resultados foram bastante positivos sendo esta a primeira abordagem para resolução do MCARP-LML, pois as soluções determinadas estão em média 3% superiores aos valores das soluções determinados pelo CPLEX. Face à quantidade de parâmetros passíveis de ajuste, e visto as soluções resultantes terem um factor probabilístico associado, foi de relativa dificuldade encontrar uma afinação ideal da meta-heurística.

Foi possível obter uma solução muito próxima do óptimo para o caso real em estudo, o que torna a meta-heurística ACO-LML adequada ao planeamento de rotas reais, e que se prevê como sendo eficaz noutras instâncias do Alto Minho.

No pior caso foram consumidos 6 minutos para finalização da meta-heurística, o que nos parece aceitável. Foi verificado que quando o algoritmo está em execução, só é utilizado um processador/núcleo na capacidade máxima. Em desenvolvimentos futuros poderá ser reduzido substancialmente este tempo, pois é aconselhável o desenvolvimento de uma aplicação preparada para o processamento em *multithreading*, onde são aproveitadas ao máximo as capacidades de multiprocessamento que existem nos novos processadores, e certamente irá resultar na diminuição dos tempos de processamento da meta-heurística.

Verificou-se que os algoritmos de melhoria local são de extrema dificuldade de implementação e de teste, mas foram notórias as melhorias da qualidade das soluções. Contudo foram ainda insuficientes para se atingirem os resultados iguais ao do método exacto.

Outro elemento influente no resultado final foi a dificuldade tida na determinação das rotas mais económicas para se efectuarem as descargas nos aterros, pois estarmos perante um problema que por tem vários aterros com capacidades limitadas. Sugere-se deste modo a implementação de um algoritmo de melhoria local integrado com a divisão das rotas e respectivas descargas.

Visto não haver até ao momento um levantamento das restantes instâncias reais no concelho, espera-se que quando estiverem disponíveis se possam verificar os resultados na meta-heurística ACO-LML. Também será interessante ser feita uma comparação de resultados entre as rotas que estão actualmente a ser realizadas pelos camiões de recolha de RSU e as rotas determinadas pela meta-heurística, a fim de se avaliar o verdadeiro sucesso deste algoritmo.

Além do desenvolvimento de mais algoritmos de melhoria local, deverá melhorar-se o desempenho desta meta-heurística através do reajuste dos parâmetros definidos, de modo a que se verifiquem melhorias dos resultados face às soluções iniciais. Caso não seja possível atingir essas mudanças, deverá ser criado um novo critério na escolha do próximo arco a servir, de forma que sejam encontradas novas e melhores soluções.

Anexo A

ID	f f _e		Experiência 1				Experiência 2				Experiência 3			
			PSRC-LML		ACO-LML		PSRC-LML		ACO-LML		PSRC-LML		ACO-LML	
			Custo	Tempo	Custo	Tempo	Custo	Tempo	Custo	Tempo	Custo	Tempo	Custo	Tempo
GDB ₁ _LML	22	7	360	0,25	314	20,3	366	0,24	332	20,0	368	0,25	334	20,9
GDB ₂ _LML	26	8	397	0,28	371	29,1	381	0,26	373	29,7	402	0,29	371	30,9
GDB ₃ _LML	22	7	347	0,30	315	20,1	361	0,29	315	20,7	351	0,3	319	20,4
GDB ₄ _LML	19	6	311	0,32	299	14,5	307	0,31	299	12,2	325	0,3	296	12,7
GDB ₅ _LML	26	8	424	0,33	401	30,9	434	0,32	391	32,1	432	0,32	398	32,6
GDB ₆ _LML	22	7	321	0,27	<u>317</u>	17,3	321	0,25	<u>317</u>	16,0	331	0,28	<u>317</u>	17,2
GDB ₇ _LML	22	7	371	0,14	338	19,2	393	0,13	355	18,2	365	0,15	343	19,8
GDB ₈ _LML	46	15	406	1,60	406	270,2	385	1,42	385	262,6	397	1,61	397	269,0
GDB ₉ _LML	51	17	397	1,83	397	354,4	377	1,6	377	375,0	386	1,8	386	353,4
GDB ₁₀ _LML	25	8	308	0,28	299	26,1	325	0,24	306	25,1	343	0,28	299	25,1
GDB ₁₁ _LML	45	15	437	0,28	437	186,1	443	0,25	443	196,7	437	0,26	437	189,6
GDB ₁₂ _LML	23	7	549	0,15	482	21,1	562	0,12	504	19,4	540	0,18	476	23,2
GDB ₁₃ _LML	28	9	593	0,15	580	35,3	595	0,12	569	35,3	601	0,12	569	35,7
GDB ₁₄ _LML	21	7	125	0,16	<u>123</u>	14,2	125	0,14	<u>123</u>	14,2	125	0,14	<u>123</u>	14,1
GDB ₁₅ _LML	21	7	84	0,13	<u>78</u>	14,5	84	0,12	<u>78</u>	14,7	84	0,12	<u>78</u>	17,3
GDB ₁₅ _LML1	21	7	84	0,15	<u>78</u>	15,5	84	0,14	<u>78</u>	14,8	84	0,14	<u>78</u>	15,6
GDB ₁₅ _LML2	21	7	80	0,16	78	23,1	79	0,15	77	24,2	79	0,15	77	26,1
GDB ₁₅ _LML3	21	7	75	0,15	73	24,7	76	0,14	72	26,2	77	0,15	73	22,1
GDB ₁₆ _LML	28	9	161	0,28	157	33,6	157	0,25	155	32,9	165	0,29	151	34,2
GDB ₁₇ _LML	28	9	126	0,31	119	38,9	125	0,28	118	35,3	126	0,29	119	37,1
GDB ₁₈ _LML	36	12	<u>187</u>	0,35	<u>187</u>	71,2	191	0,32	191	71,6	194	0,32	192	76,5
GDB ₁₉ _LML	11	3	<u>77</u>	0,12	<u>77</u>	3,3	<u>77</u>	0,12	<u>77</u>	3,1	<u>77</u>	0,1	<u>77</u>	3,6
GDB ₂₀ _LML	22	7	168	0,21	156	20,7	170	0,2	<u>150</u>	19,1	166	0,25	154	23,8
GDB ₂₁ _LML	33	11	194	0,43	190	58,4	188	0,41	188	54,3	194	0,42	191	58,5
GDB ₂₂ _LML	44	14	245	0,69	245	149,9	242	0,6	242	142,1	248	0,7	247	151,6
GDB ₂₃ _LML	55	18	288	1,12	288	246,4	286	1,02	286	239,9	288	1,13	288	252,0
AltoMinho	31	10	13459	7,42	13035	77,8	13406	7,39	13076	72,9	13459	7,43	13090	73,6

Notas: Tempo em segundos; Soluções a sublinhado representam soluções óptimas; Novas soluções óptimas identificadas a negrito

Referências

- [1] CESUR, "Análise dos tarifários de resíduos urbanos, em Portugal," Instituto Superior Técnico 2007.
- [2] N. V. Karadimas, N. Doukas, M. Kolokathi, and G. Defteraiou, "Routing optimization heuristics algorithms for urban solid waste transportation management," *WSEAS Transactions on Computers*, vol. 7, pp. 2022-2031, 2008.
- [3] ERSAR, "Dados ao consumidor," 2012.
- [4] INE. (2012). *Dados Estatísticos*. Available: http://www.ine.pt/xportal/xmain?xpid=INE&xpgid=ine_main
- [5] E. Victoria. (2004). Waste wise events toolkit. Available: <http://www.ecorecycle.vic.gov.au/>
- [6] S. Victoria, "Guide to Best Practice for Waste Management in Multi-unit Developments," *resourcesmart.vic.gov.au*, 2010.
- [7] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization algorithms and complexity*. Englewood Cliffs: Prentice-Hall, 1982.
- [8] M. C. Mourão, "Optimização de Rotas na Recolha de Resíduos Urbanos : Modelos e Algoritmos," Doutoramento em Matemática Aplicada à Economia e Gestão, Instituto Superior de Economia e Gestão, 1997.
- [9] B. L. Golden and R. T. Wong, "Capacitated Arc Routing Problems," *Networks*, vol. 11, pp. 305-315, 1981.
- [10] B.-I. Kim, S. Kim, and S. Sahoo, "Waste collection vehicle routing problem with time windows," *Computers & Operations Research*, vol. 33, pp. 3624-3642, 2006.
- [11] S. Roy and J. M. Rousseau, "The capacitated Canadian postman problem," *INFOR*, vol. 27, pp. 58-73, 1989.
- [12] H. A. Eiselt, M. Gendreau, and G. Laporte, "Arc routing problems. II. The rural postman problem," *Operations Research*, vol. 43, pp. 399-414, 1995.
- [13] J. M. Belenguer, E. Benavent, P. Lacomme, and C. Prins, "Lower and upper bounds for the mixed capacitated arc routing problem," *Computers and Operations Research*, vol. 33, pp. 3363-3383, 2006.
- [14] P. Lacomme, C. Prins, and W. Ramdane-Cherif, "Competitive memetic algorithms for arc routing problems," *Annals of Operations Research*, vol. 131, pp. 159-185, 2004.
- [15] A. Rodrigues and J. S. Ferreira, "A mixed capacitated arc routing problem with limited multi-landfills for waste collection," *INESC TEC* p. 11, 2012 (submitted for publication).
- [16] M. C. Mourao, A. C. Nunes, and C. Prins, "Heuristic methods for the sectoring arc routing problem," *European Journal of Operational Research*, vol. 196, pp. 856-68, 2009.
- [17] C. Blum and A. Roli, "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison," *ACM Computing Surveys*, vol. 35, pp. 268-308, 2003.
- [18] C. R. Reeves, *Modern heuristic techniques for combinatorial problems*. London: McGraw-Hill Book, 1995.
- [19] J. Dréo. (2007). *Classification of metaheuristics*. Available: <http://metah.nojhan.net/post/2007/10/12/Classification-of-metaheuristics>

- [20] C. M. Monção. (2012, 16/02/2012). *Portal municipal de Monção - Recolha de lixo*. Available: http://www.cm-moncao.pt/portal/page/moncao/portal_municipal/SERVICOS_MUNICIPAIS/servicos_banos/recolhadelixo
- [21] I. von Poser and A. R. Awad, "Optimal Routing for Solid Waste Collection in Cities by using Real Genetic Algorithm," in *Information and Communication Technologies, 2006. ICTTA '06. 2nd*, 2006, pp. 221-226.
- [22] M. Oliveira, "Optimização de circuitos de recolha de lixos domésticos em zonas urbanas," Doutor em Ciências de Engenharia, Tese de doutoramento, Porto, FEUP, 2008.
- [23] S. Sahoo, K. Seongbae, K. Byung-In, B. Kraas, and A. Popov, Jr., "Routing optimization for waste management," *Interfaces*, vol. 35, pp. 24-36, 2005.
- [24] B. M. Ombuki-Berman, A. Runka, and F. T. Hanshar, "Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms," in *3rd IASTED International Conference on Computational Intelligence, CI 2007, July 2, 2007 - July 4, 2007*, Banff, AB, Canada, 2007, pp. 91-97.
- [25] A. M. Benjamin and J. E. Beasley, "Metaheuristics for the waste collection vehicle routing problem with time windows," *Computers & Operations Research*, vol. 37, pp. 2270-2280, 2010.
- [26] T. Kulcar, "Optimizing solid waste collection in Brussels," *European Journal of Operational Research*, vol. 90, pp. 71-77, 1996.
- [27] G. Ghiani, F. Guerriero, G. Improta, and R. Musmanno, "Waste collection in Southern Italy: solution of a real-life arc routing problem," *International Transactions in Operational Research*, vol. 12, pp. 135-44, 2005.
- [28] J. Bautista, E. Fernandez, and J. Pereira, "Solving an urban waste collection problem using ants heuristics," *Computers and Operations Research*, vol. 35, pp. 3020-3033, 2008.
- [29] P. Miller. (2007, July 2007) The Genius of Swarms. *National Geographic Magazine*. Available: <http://ngm.nationalgeographic.com/2007/07/swarms/miller-text>
- [30] A. Coloni, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," in *Toward a Practice of Autonomous Systems. Proceedings of the First European Conference on Artificial Life, 11-13 Dec. 1991*, Cambridge, MA, USA, 1992, pp. 134-42.
- [31] T. H. Stützle, Holger, "Improving the Ant System: A Detailed Report on the MAX-MIN Ant System," *FG Intellektik; FB Informatik; TU Darmstadt, Germany, Tech Rep. AIDA-96-12*, 1996.
- [32] T. Stutzle and H. H. Hoos, "MAX-MIN Ant System," *Future Generation Computer Systems*, vol. 16, pp. 889-914, 2000.
- [33] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *BioSystems*, vol. 43, pp. 73-81, 1997.
- [34] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 53-66, 1997.
- [35] M. Dorigo, G. Di Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artificial Life*, vol. 5, pp. 137-72, 1999.
- [36] L. Santos, J. Coutinho-Rodrigues, and J. R. Current, "An improved ant colony optimization based algorithm for the capacitated arc routing problem," *Transportation Research Part B: Methodological*, vol. 44, pp. 246-66, 2010.
- [37] B. L. Golden, J. S. DeArmon, and E. K. Baker, "Computational experiments with algorithms for a class of routing problems," *Computers & Operations Research*, vol. 10, pp. 47-59, 1983.
- [38] P. Lacomme, C. Prins, and A. Tanguy, "First competitive ant colony scheme for the CARP," in *Ant Colony Optimization and Swarm Intelligence. 4th International Workshop, ANTS 2004. Proceedings, 5-8 Sept. 2004*, Berlin, Germany, 2004, pp. 426-7.
- [39] G. Ulusoy, "The fleet size and mix problem for capacitated arc routing," *European Journal of Operational Research*, vol. 22, pp. 329-37, 1985.