Universidade do Minho
Escola de Engenharia

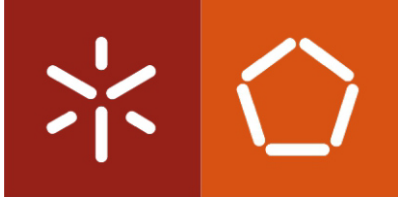
Mário Jorge Meira de Abreu

Problema de orientação de equipas (TOP) aplicado às linhas de engarrafamento móveis

Outubro de 2014

Problema de orientação de equipas (TOP) aplicado às linhas de engarrafamento móveis
Mário Jorge Meira de Abreu

UMinho | 2014



Universidade do Minho
Escola de Engenharia

Mário Jorge Meira de Abreu

Problema de orientação de equipas (TOP) aplicado às linhas de engarrafamento móveis

Dissertação de Mestrado
Mestrado em Engenharia de Sistemas

Trabalho Efetuado sob a orientação do
Professor Doutor José António Vasconcelos Oliveira

Outubro de 2014

DECLARAÇÃO

Nome: Mário Jorge Meira de Abreu

Endereço eletrónico: pg23099@alunos.uminho.pt Telefone: 939462957

Cartão do Cidadão: 13240799

Título da dissertação: Problema de orientação de equipas (TOP) aplicado a linhas de engarrafamento móvel.

Orientador:

Professor Doutor José António Vasconcelos Oliveira

Ano de conclusão: 2014

Mestrado em Engenharia de Sistemas

Universidade do Minho, _____/_____/_____

Assinatura:

AGRADECIMENTOS

A presente dissertação é o fruto de muitas horas de esforço e dedicação, e representa o investimento feito para alcançar um objetivo. No entanto, tal só foi possível com o apoio e força dados por várias pessoas, nomeadamente:

O Professor Doutor José António Oliveira que muito bem me orientou, pela sua disponibilidade, conhecimento e paciência desde o início até à conclusão deste projeto.

A minha namorada Carla, que sempre me apoiou e motivou o mais que pôde, mesmo quando as condições eram adversas.

Aos meus pais e irmã, cujo apoio incondicional foi imprescindível para hoje estar e entregar esta dissertação.

Aos meus amigos, em especial ao Fernando Araújo, que sempre se mostrou disponível para o que fosse necessário.

A todos aqueles que, apesar de não serem mencionados, me ajudaram direta ou indiretamente a alcançar este objetivo.

A todas as pessoas acima mencionadas, o meu sincero obrigado, sem elas este trabalho não teria sido tão conseguido.

RESUMO

Os custos com transportes nas empresas estão constantemente a aumentar devido à realidade económica nacional e à dependência de energias fósseis. É, portanto essencial que seja feita uma otimização constante com vista a minimizar estes custos.

Esta dissertação propõe estudar a aplicação do modelo de problemas de orientação de equipas com janelas temporais às linhas de engarrafamento móvel de uma empresa Portuguesa. Recorre-se para isso a coordenadas geográficas geradas aleatoriamente (clientes aleatórios) e à utilização de distâncias reais entre essas localizações através do serviço disponibilizado pelo *Google Maps*.

Inicialmente apresentam-se conceitos importantes sobre a cadeia de abastecimento seguido do estado da arte no âmbito de problemas de otimização de rotas.

Apresenta-se o caso de estudo, com a descrição do problema real, recursos disponíveis e assunções na modelação do problema real e a ainda o modelo matemático submetido ao NEOS Server para a obtenção de soluções ótimas. Recorre-se a utilitários da linha de comandos em conjunto com expressões regulares para filtrar e formatar a resposta fornecida pelo NEOS Server.

Apresenta-se ainda a aplicação *web*, desenvolvida em Nodejs, para gerir os clientes e para a conceção das diferentes instâncias submetidas aos testes, para a consolidação do estudo.

Apresenta-se uma análise dos resultados obtidos da aplicação do TOPTW, bem como estratégias alternativas para melhorar as soluções obtidas e para tentar obter resultados em instâncias de maior dimensão. Os resultados dos diferentes testes são apresentados numa folha de Excel, juntamente com informação pertinente extraída da análise dos resultados, nomeadamente através de tabelas comparativas gráficos e diagramas de Gantt.

Por fim, apresentam-se as conclusões do trabalho desenvolvido ao longo da dissertação e perspetivas para trabalho futuro.

Palavras-Chave: Problemas de Orientação com Janelas Temporais, Otimização, Logística, Planeamento e Controlo, Gestão da Cadeia de Abastecimento

ABSTRACT

The costs with transports in organizations are constantly increasing due to the national economic reality and to the highly dependency of fossil energies. It's crucial to do a constant optimization to minimize that costs.

This dissertation proposes a study of the team orienteering problems with time windows applied to a mobile bottling line of a Portuguese organization. To do that, we use random generated geographic coordinates and we use the actual distances between them through the use of a Google Maps service.

Initially, we present important concepts of the chain supply and the state of the art in route optimization problems. Then we present the case of study with the description of the real problem, available resources and the assumptions of the real problem and also the mathematic model submitted to NEOS Server in order to obtain the optimal solutions.

After that, we present a web application developed in Nodejs and mongoDB for random client generation, instance generation and client management.

We show the analysis of results and alternative strategies to improve the obtained solutions. The results are displayed with resource of Excel tables and comparative charts.

Finally we present the conclusions and perspectives for the future work.

KEYWORDS: Team Orienteering Problems with Time Windows, Optimization, Logistics, Planning and Control, Supply Chain Management.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE MODELOS MATEMÁTICOS

Lista de Abreviaturas, Siglas e Acrónimos

VRP	*Vehicle Routing Problems*
OP	*Orienteering Problems*
TOP	*Team Orienteering Problems*
AMPL	*A Mathematical Programming Language*
OPTW	*Orienteering Problems With Time Window*
TOPTW	*Team Orienteering Problems With Time Window*
ELE	*Empresa de Engarrafamento Móvel*
GLS	*Guided Local Search*
MILP	*Mixed integer Linear Programming*
RPC	*Remote Procedure Call*

1. INTRODUÇÃO

1.1 Motivação

Mais do que nunca, a economia mundial gira hoje em torno do conceito da mobilidade. Para além da mobilidade de ideias e de conhecimento, proporcionada pela Internet e pelas tecnologias de informação e comunicação, a mobilidade de bens e pessoas, resulta da massificação dos transportes, que veio permitir reduzir distâncias. Atualmente o transporte e a logística constituem uma importante parcela nos custos das empresas. Em 2011, o sector dos transportes e armazenagem representava 2.1% das empresas, 4.3% do pessoal ao serviço, 5.2% do volume de negócios e 6.7% do valor acrescentado do total nacional. Na Europa, o sector de transporte emprega diretamente cerca de 10 milhões de pessoas e é responsável por cerca de 5% do PIB [1]. Dada a grande fração que a logística representa nos custos das empresas, torna-se então crucial ter a logística organizacional a operar o mais eficientemente possível, por forma a reduzir custos, eventualmente supérfluos.

Nos dias de hoje, os clientes são cada vez mais exigentes e pretendem os produtos certos, no local exato e em horas e quantidades previamente acordadas. Eficiência e inteligência, são por isso as palavras de ordem para apostar em meios e redes de transporte como acréscimo de valor e competitividade. Neste sentido é importante recorrer às tecnologias de informação para tentar solucionar, o mais eficientemente possível, problemas de encaminhamento de veículos. A modelação deste tipo de problemas tenta aproximar ao máximo a realidade, sendo impostas restrições reais, como a frota disponível, capacidade da frota, possíveis janelas temporais por parte dos clientes, entre outras.

Na presente dissertação pretende-se estudar uma possível melhoria do serviço de engarrafamento móvel de uma empresa nacional. O escalonamento dos clientes aos recursos para engarrafamento pode ser visto como uma variante dos problemas de orientação (Orienteering Problem, OP), problemas de orientação de equipas (Team Orienteering Problems, TOP) ou ainda problemas de orientação de equipas com janelas temporais (Team Orienteering Problems with Time Windows, TOPTW**)** se considerarmos que os clientes ou o recurso, ou ambos, está estrito por uma janela temporal [2].

Problemas de encaminhamento de veículos (Vehicle Routing Problems, VRP) é um nome genérico atribuído à classe de problemas que visa a otimização de entrega e/ou recolha de bens recorrendo eficientemente a uma frota de veículos. Dada a sua complexidade e importância prática, os

problemas de VRP contam com uma grande variedade de estudos no sentido de melhor solucionar este tipo de problemas [3].

1.2 Objetivo

Esta dissertação tem como objetivo o estudo do comportamento de modelos de orientação de equipas aplicados ao problema real de escalonamento de clientes aos recursos de engarrafamento móvel, existente numa empresa portuguesa. Recorrer-se-á linguagem *AMPL* e ao servidor NEOS Server *Gurobi*[1] para obter soluções óptimas para a modelação do problema. No melhor caso, o gestor conseguirá obter a solução ótima para o problema. Paralelamente ao servidor NEOS Server *Gurobi*, será criada uma aplicação web para a gestão dos clientes, serviços e respetivas rotas com soluções praticáveis, contendo a ordem de visita dos clientes.

1.3 Estrutura da dissertação

Este documento divide-se em cinco capítulos: introdução, revisão da literatura, caso de estudo, análise de resultados e por fim, conclusões e trabalho futuro.

No primeiro capítulo descreveu-se a motivação para a realização desta dissertação, bem como o enquadramento do problema real e objetivo desejado. Termina-se o capitulo com a descrição da estrutura deste documento.

No Capítulo 2 é apresentada uma revisão sobre o estado da arte, através da apresentação das principais instâncias de problemas de orientação na expetativa de identificar o que melhor se enquadra com o problema real.

No Capítulo 3 é apresentada a descrição detalhada do problema real, assunções e descrição modelação do problema e modelo utilizado. É apresentada também uma aplicação parcialmente concebida para o apoio de suporte à decisão e geração de instâncias.

No quarto Capítulo é feita uma análise de resultados e são estudadas abordagens alternativas à utilização do modelo matemático TOPTW.

Por fim, no quinto Capítulo fez-se a conclusão do estudo e apresentam-se apontadores para trabalho futuro.

---

[1] Solucionador de problemas lineares modelados em AMPL.

[2] Principal minério do alumínio

[3] Substância segregada por um animal e reconhecida por animais da mesma espécie na comunicação e no reconhecimento.

2. REVISÃO DA LITERATURA

No presente capítulo será apresentada uma breve definição dos principais conceitos da cadeia de abastecimento. Será também apresentada a revisão da literatura e atual estado da arte que se entende ser interessante e adequada para o caso de estudo.

2.1 Definição de conceitos

A cadeia de abastecimento como um todo abrange desde matérias primas (que existem no solo, mar ou ar) à venda do produto acabado ao consumidor final e até à reciclagem do produto usado. O material flui desde o produto básico (como uma mina de bauxita[2] como fonte de minério de alumínio) até ao produto acabado (e.g. uma lata de refrigerante). A analogia ao fluxo da água num rio é muitas vezes utilizada para descrever organizações localizadas perto da fonte como *upstream* e aquelas perto do consumidor final como *downstream*. Cada empresa numa cadeia de abastecimento é vista como um parceiro. Em cada estágio de conversão da matéria prima em produto acabado, podem existir retornos. Por exemplo, material rejeitado da firma precedente da cadeia, ou desperdícios tal como uma lata que precisa de ser reciclada.

A cadeia de abastecimento é uma rede de parceiros que coletivamente convertem um bem básico (upstream) em produto acabado (downstream) que é valorizado pelo consumidor final, e gere os retornos em cada estágio.

Cada parceiro da cadeia é diretamente responsável por um processo que acrescenta valor ao produto.

O processo transforma inputs na forma de material ou informação em outputs na forma de bens ou serviços.

No caso da lata de refrigerante, os parceiros realizam processos tais como mineração, transporte, refinação e laminação a quente. A lata de refrigerante tem um valor superior do que a bauxita (por quilograma de alumínio). A gestão da cadeia de abastecimento envolve o planeamento e controlo de todos os processos desde a produção de matéria prima até à compra pelo consumidor final e reciclagem da lata usada. O planeamento refere-se a fazer o plano que define quanto de cada

[2] Principal minério do alumínio

produto deve ser comprado, feito, distribuído e vendido a cada dia da semana ou mês. Controlar significa manter o plano traçado, apesar dos diversos problemas que poderão aparecer no caminho até ao final do horizonte de planeamento. O objetivo é coordenar o planeamento e o controlo de cada processo para que as necessidades do cliente final sejam satisfeitas corretamente. Então, uma definição da cadeia de abastecimento válida é:

Planear e controlar todos os processos do negócio - desde o consumidor final até aos fornecedores de matéria prima – que interligam os parceiros de uma cadeia de abastecimento de maneira a servir as necessidades do cliente final.

Servir as necessidades do cliente final tem diferentes implicações, em diferentes contextos. Em ambientes sem fins lucrativos como saúde pública, servir implica uma melhoria continua, melhor que outras regiões/países, melhor valor. Em ambiente comercial, servir implica melhor do que os competidores, melhor valor para o dinheiro, etc. Em qualquer dos ambientes o foco da cadeia de abastecimento como um todo é integrar os processos dos parceiros da cadeia de abastecimento, dos quais o cliente final é o mais importante. O cliente final é quem desencadeia todo este processo, com a compra dos produtos finais. É este processo que inicia todo o fluxo da cadeia de abastecimento.

O grau de satisfação do cliente final com o produto final depende fundamentalmente da gestão de fluxos de materiais e da gestão do fluxo de informação ao longo da cadeia de abastecimento. Se uma entrega está atrasada ou o produto não está completo, o objetivo de toda a cadeia de abastecimento pode ser comprometido caso os competidores consigam executar melhor as tarefas logísticas. Logística é o elemento vital para a gestão da cadeia de abastecimento e abrange tanto planeamento de longo prazo como planeamento de curto/médio prazo e controlo. Assim, a logística pode ser definida como:

A tarefa de coordenação do fluxo de materiais e do fluxo de informação em toda a cadeia de abastecimento. [4]

2.2 Enquadramento

O termo OP surge do desporto de orientação. Na prova de orientação cada participante começa num ponto de controlo e tem que visitar tantos *check-points* quantos aqueles que conseguir e voltar ao ponto de controlo final num dado limite de tempo. Os participantes que não respeitem o tempo

imposto são desqualificados ou são penalizados na classificação final. Cada *check-point* representa um prémio para o jogador, e o objetivo principal é maximizar o prémio recolhido em todos os *check-point's*.

O problema de orientação de equipas aparece documentado pela primeira vez em 1994 com o nome *Multiple Tour Maximum Collection Problem* [5] enquanto que a definição aparece em 1996 em The Team Orienteering Problem por *Chao et al.* [6] Na verdade, os OP pertencem à classe de problemas de encaminhamento (Routing Problem).

Atualmente existem várias instâncias documentadas no âmbito da problemática dos OP, que geralmente são fruto de restrições impostas analisadas em problemas reais. Serão apresentados problemas de orientação *(Orienteering Problems, OP)*, problemas de orientação de equipas *(Team Orienteering Problems, TOP)*, problemas de orientação com janelas temporais *(Orienteering Problems with Time Windows, OPTW)* e problemas de orientação de equipas com janelas temporais (Team Orienteering Problems With Time Windows, TOPTW). Para cada um deles será apresentado a definição do problema com uma breve descrição, modelo matemático, complexidade, heurísticas e meta-heurísticas, metodologias e casos de aplicação [2].

2.3 Problemas de Orientação

2.3.1 Definição do problema

Os OP podem ser definidos com os seguintes componentes, um conjunto de N vértices em que cada um tem um prémio de visita S_i. O ponto inicial é o vértice 1, ao passo que o vértice N é o final. O tempo de viagem t_{ij} entre o vértice i e o vértice j é conhecido para todos os pares de vértices. Como existe um tempo máximo t_{max} de operação, em princípio podem não ser visitados todos os vértices. O objetivo do problema de orientação é encontrar um caminho, limitado pelo t_{max}, que visite alguns vértices, de tal forma a maximizar o prémio coletado. Assume-se que os prémios são valores positivos e que cada vértice só pode ser visitado uma única vez.

O OP pode ainda ser definido por um grafo $G = \{V, A\}$ em que $V = \{v_1, v_2 \dots v_n\}$ é o conjunto de vértices e $A = \{(v_i, v_j), (v_k, v_l,) \dots (v_m, v_n)\}$ um conjunto de arcos que interligam vértices. Os OP podem ser modelados como um problema de programação inteira em que as variáveis de decisão são: $x_{ij} = 1$ se, no caminho *p*, o vértice i for seguido da visita ao vértice j, $x_{ij} = 0$ caso contrário; $u_i =$ é a posição do vértice i no caminho *p.*

$$Max \sum_{i=2}^{N-1} \sum_{j=2}^{N} S_i x_{ij}, \quad (1)$$

$$\sum_{j=2}^{N} x_{1j} = \sum_{i=1}^{N-1} x_{jN} = 1, \quad (2)$$

$$\sum_{i=1}^{N-1} x_{ik} = \sum_{j=2}^{N} x_{kj} \leq 1; \ \forall k = 2, \dots, N-1, \quad (3)$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} t_{ij} x_{ij} \leq T_{max}, \quad (4)$$

$$2 \leq u_i \leq N; \ \forall i = 2, \dots, N, \quad (5)$$

$$u_i - u_j + 1 \leq (N-1)\left(1 - x_{ij}\right); \ \forall i, j = 2, \dots, N, \quad (6)$$

$$x_{ij} \in \{0,1\}; \ \forall i, j = 1, \dots, N. \quad (7)$$

Modelo 1 Modelo matemático OP

A função objetivo (1) pretende maximizar o prémio recolhido. A restrição (2) garante que todos os caminhos têm início no vértice 1 e terminam no vértice N. Em (3) é garantido a conectividade do caminho e que cada vértice é visitado, no máximo, uma vez. A garantia que o tempo máximo é respeitado é assegurado na restrição (4). Com as restrições (5) e (6) garante-se que não existem sub-rotas.

Assume-se que o grafo é completo e não direcionado em que as distâncias entre dois vértices é simétrica, ou seja, $t_{ij} = t_{ji}$.

2.3.2 Complexidade

Relativamente à complexidade deste tipo de problemas, estes pertencem à classe *NP-difícil (Non Polynomial hard – NP-Hard)* e não existe nenhum algoritmo exato capaz de solucionar este problema em tempo polinomial [7].

2.3.3 Abordagens de solução

Vários investigadores propõem algoritmos exatos para resolver Problemas de Orientação. Em 1990, Laporte e Martello usaram algoritmos de *branch-and-bound* para resolver problemas com menos de 20 vértices [8]. Dois anos mais tarde, *Ramesh et al.(1992)* aumentaram a fasquia para 150 vértices [9]. Leifer e Rosenwein [10] adicionaram inequações à formulação de *Laporte e Martello* para obter soluções com melhores limites superiores. Em 1998, Fischetti et al. [11] e Gendreau et al. [12] introduziram mais inequações válidas e propuseram algoritmos de *branch-and-cut* e, desta feita, conseguiram solucionar problemas com 500 vértices.

Existem também heurísticas para solucionar OP. Tsiligiridis [13] propôs um algoritmo estocástico *(S-Algorithm)* e determinístico *(D-Algorithm)*. O algoritmo estocástico essencialmente gera vários caminhos e seleciona o melhor. É utilizado o método de *Monte-Carlo* para selecionar o vértice seguinte a ser adicionado ao caminho. A probabilidade de um dado vértice ser adicionado ao caminho é baseada na distância euclidiana necessária para o visitar e no prémio de visita. O algoritmo determinístico usa uma variante do procedimento do VRP de Wren et al. [14]. Os caminhos são construídos em sectores separados da mesma região geográfica, de acordo com as regras predefinidas. Variando os sectores, são gerados 48 caminhos dos quais será selecionado o melhor. Ambos os algoritmos são complementados com *path-improvement algorithm (R-I-Algorithm)*.

Chao et al. [15] desenvolveu uma heurística de centro de gravidade recorrendo à métrica euclidiana. O primeiro passo é a construção do caminho e utiliza uma estratégia *greedy*, que iterativamente insere vértices com o prémio mais elevado e uma duração razoável. O segundo passo é um procedimento de aperfeiçoamento que usa *2-Opt* [16] e inserção de menor custo *(cheapest insertion)*. O terceiro e último passo é o centro de gravidade e consiste em, a partir de um caminho vazio, adicionar vértices de acordo com o seu ranking, usando inserção de menor custo. O ranking de cada vértice é calculado com base no rácio entre o prémio e a distância ao centro de gravidade do caminho anterior. Se não for possível inserir mais vértices, repetem-se os passos 2 e 3.

Ramesh e Brown [17] introduziram a heurística de quatro fases *(four-phase)*. A fase de inserção relaxa a restrição de tempo e a fase de melhoramento de custo usa 2-Opt e 3-Opt. A terceira fase passa por diminuir o comprimento do caminho através de remoção e inserção de um vértice. Estas três fases são repetidas até que o número máximo de vértices existentes no caminho seja atingido.

A heurística cinco passos *(five-steps)* de Chao et al. [18] só considera vértices que podem ser alcançados num determinado tempo máximo T_{max}. No passo de inicialização são criados vários

caminhos diferentes no espaço euclidiano. Cada caminho começa num vértice longe dos vértices inicial e final. A inserção de todos os outros vértices a um dos caminhos é feita através da inserção de menor custo. O melhor caminho é escolhido como a solução inicial T_{op}. Os vértices não incluídos são também atribuídos a caminhos possíveis T_{nop}. O primeiro passo de aperfeiçoamento é a troca de dois pontos *(two-point Exchange)* que, recorrendo à inserção de menor custo, consiste em tentar melhorar o T_{op} adicionando um vértice extra, dos vértices existentes em T_{nop} e removendo um vértice incluído para T_{nop}. Todos os caminhos tem que permanecer viáveis e é permitido uma pequena diminuição do prémio total. O segundo passo consiste em trocar um vértice de um caminho para outro, se possível, e se o prémio total não diminuir em demasia. O terceiro passo engloba 2-Opt. Por fim, um número específico de vértices com o rácio entre o prémio e custo de inserção baixo são removidos do caminho ótimo e o algoritmo recomeça. Esta é, até aqui, a melhor heurística mencionada.

Gendreau et al. [19] propuseram a heurística de procura tabu *(Tabu Search)* que iterativamente insere grupos *(clusters)* de vértices no caminho e remove vários vértices. Comparativamente com as heurísticas anteriores, esta heurística diminui a probabilidade de ficar preso num ótimo local e a probabilidade de incluir vértices com bom score longe do caminho ótimo. Foram desenvolvidas mais heurísticas para resolver problemas de orientação, mas nenhuma delas melhorou significativamente as já existentes.

2.3.4 Casos de aplicação

É possível encontrar vários ambientes onde OP são aplicados. Segundo Tsiligiridis [13], a primeira aplicação foi no problema do caixeiro viajante *(travelling salesperson)*, que não tinha tempo suficiente para visitar todas as cidades. O caixeiro sabe exatamente o número de vendas em cada cidade, e quer maximizar as vendas, limitando o tempo de viagem a uma período de tempo (normalmente um dia ou uma semana).

Golden et al. [7] descrevem o OP aplicado ao problema de entrega de combustível ao domicílio. O objectivo principal é maximizar o prémio recolhido satisfazendo um certo número de clientes diariamente. Nesta caso o prémio está diretamente relacionado com a urgência, que neste caso é a necessidade em receber o combustível.

Também é possível ver OP aplicados a guias turísticos móveis, Souffriau et al. [20]. Em que , para turistas que estão de visita a uma cidade ou região é muitas vezes impossível visitar todos os pontos que lhes interessa. Então, os turistas escolhem os pontos de visita que acreditam ser os mais interessantes. Muitas vezes é uma tarefa extremamente difícil fazer o planeamento para visitar todos

esses pontos de interesse no tempo disponível. Estes problemas são conhecidos na literatura como problemas de conceção de rotas para turistas *(Tourist Trip Design Problems , TTDP)*, [21].

O OP é o modelo mais simples de TTDP. Estes modelos exigem soluções de grande qualidade em poucos segundos. Nestes problemas, o tempo necessário para visitar um determinado vértice desempenha um papel importante na seleção de vértices.

2.4 Problemas de orientação de equipas

Os problemas de orientação de equipas *(TOP)*, também conhecidos como *multiple tour maximum collection problem (MTMCP)*, podem ser vistos como OP jogados por equipas compostas por mais que um elemento, em que o objetivo é maximizar o prémio total da equipa, somando o prémio recolhido por cada um dos seus elementos, no tempo limite. Tal como nos OP, também estes estão sob pena de desqualificação ou penalização no prémio final caso não respeitem o limite temporal.

2.4.1 Definição do problema

O TOP [15] pode ser definido como um grafo $G = \{V, A\}$ em que $V = \{v_1, v_2 \dots v_n\}$ é o conjunto de vértices e $A = \{(v_i, v_j), (v_k, v_l,) \dots (v_m, v_n)\}$ um conjunto de arcos que interligam vértices. Assim, os TOP podem ser modelados como um problema de programação inteira em que as variáveis de decisão são: $x_{ijp} = 1$ se, no caminho *p*, o vértice i for seguido da visita ao vértice j, $x_{ijp} = 0$ caso contrário. $Y_{ip} = 1$ se o vértice i for visitado no caminho *p*, $Y_{ip} = 0$ caso contrário; $u_{ip} =$ é a posição do vértice i no caminho *p.*

$$Max \sum_{p=1}^{p} \sum_{i=2}^{N-1} S_i Y_{ip}, \tag{8}$$

$$\sum_{p=1}^{p} \sum_{j=2}^{N} X_{1jp} = \sum_{p=1}^{p} \sum_{i=2}^{N-1} X_{iNp} = P, \tag{9}$$

$$\sum_{p=1}^{p} Y_{kp} \leq 1;\ \forall k = 2, \dots, N-1 \tag{10}$$

$$\sum_{i=1}^{N-1} x_{ikp} = \sum_{j=2}^{N} x_{kjp} = y_{kp};\ \forall k = 2, \dots, N-1; \forall p = 1, \dots, P \tag{11}$$

$$\sum_{i=1}^{N-1}\sum_{j=2}^{N} t_{ij}x_{ijp} \leq T_{\max}\ ;\ \forall p = 1, \dots, P \quad (12)$$

$$2 \leq u_{ip} \leq N;\ \forall i = 2, \dots, N;\ \forall p = 1, \dots, P \quad (13)$$

$$u_{ip} - u_{jp} + 1 \leq (N-1)\left(1 - x_{ijp}\right);\ \forall i, j = 2, \dots, N;\ \ \forall p = 1, \dots, P \quad (14)$$

$$x_{ijp}, y_{ip} \in \{0,1\};\ \forall i, j = 1, \dots, N;\ \forall p = 1, \dots, P \quad (15)$$

Modelo 2 Modelo matemático TOP

A função objetivo (8) maximiza o prémio total recolhido. A restrição (9) garante que cada um dos caminhos têm início no vértice 1 e terminam no vértice N. A restrição (10) assegura que cada vértice é visitado no máximo uma vez. Com (11) garante-se a conectividade de cada caminho. Com a restrição (12) garante-se que o tempo limite é respeitado por cada um dos caminhos da equipa. As restrições (13) e (14) são necessárias para garantir que não são criadas sub-rotas.

2.4.2 Complexidade

O TOP na sua versão mais simples, ou seja, composto por uma equipa com um membro, problema OP, tem complexidade *NP-difícil*, o que quer dizer que TOP têm, pelo menos, a mesma complexidade [6].

2.4.3 Abordagens de solução

Butt e Ryan [22] apresentaram um algoritmo exato para solucionar TOP usando geração de colunas. Este algoritmo é capaz de solucionar problemas com até 100 vértices, se cada caminho for constituído por poucos vértices.

Boussier et al.[23] mostram um algoritmo exato para resolver problemas TOP e TOPTW. Eles conjugaram vários métodos para aperfeiçoar o algoritmo. O esquema *branch-and-price* é conjugação de *geração de coluna (column generation)* com *branch-and-bound. Por forma a melhorar a performance, são utilizados várias técnicas de aceleração: "limited discrepancy search",* uma heurística de árvore de procura *(tree-search) apresentada por* Harvey e Ginsberg [24]*, "label loading"* e "*meta extensions".* Com este algoritmo, problemas com até 100 vértices, e com até cerca de 15 vértices por caminho, são otimamente solucionados em menos de duas horas de computação.

A primeira heurística para problemas TOP foi apresentada por Chao et al. [25] e é idêntica à heurística *five-steps* para problemas OP. Enquanto que em OP é escolhido apenas o melhor caminho, em TOP os melhores caminhos são escolhidos e são utilizados dois passos de reinicialização.

Em 2005 Tang e Miller-Hooks [26] desenvolveram uma heurística de procura tabu *(Tabu Search, TMH)* embutida num procedimento de memória adaptativa *(Adaptative Memory Procedure, AMP).* Na inicialização desta heurística, os parâmetros são programados para explorar um reduzido número de soluções vizinhas. No passo seguinte, o de aperfeiçoamento, procedimentos aleatórios e gananciosos *(random and greedy)* geram soluções vizinhas possíveis e impossíveis com base nos parâmetros estabelecidos. Os parâmetros garantem trocas contínuas entre pequenas e grandes estruturas vizinhas. No passo de avaliação, a melhor solução não-tabu (*non-tabu*) é selecionada e os parâmetros são ajustados com base no tamanho da vizinhança e na qualidade da solução. O *AMP* funciona de forma similar em algoritmos genéticos (*genetic algorithm*) e um descendente pode ser gerado a partir de mais que dois pais. Através da combinação de caminhos simples cria-se uma solução simples para o *Tabu Search*. Como resultado do *Tabu Search* são atualizados esses caminhos.

Archetti et al. [27] desenvolveram quatro meta-heurísticas. Duas variantes da heurística *Tabu Search*, procura de vizinhança de variação lenta *(slow variant neighbourhood search, SVN)* e procura de vizinhança de variação rápida *(fast variant neighbourhood search, FVN)*. As quatro meta-heurísticas iniciam com uma solução incumbente s, saltando depois para uma solução intermédia s'. Depois com a *Tabu Search* tenta-se melhorar a solução s'. A nova solução s'' é comparada com a solução s. Na estratégia *VNS*, a solução s'' só é aceite se o prémio for superior ao da solução *s*. Na estratégia Tabu Search, s'' é sempre aceite. Este processo é repetido até que se chegue ao critério de paragem. Tal como na heurística *five-steps* de Chao et al. [18], os vértices não incluídos são também organizados em caminhos. A *Tabu Search* usa dois movimentos, *1-move* para mover um vértice de um caminho para outro e *swap-move* para trocar dois vértices entre dois caminhos. Os vértices são inseridos utilizando sempre a inserção de menor custo. Como as soluções impossíveis também são aceites pelo algoritmo, são desenvolvidos vários procedimentos para reduzir ou eliminar a inviabilidade de um caminho. Nestes algoritmos são utilizados dois tipos de salto, o primeiro é uma série de movimentos *1-move* com vértices não incluídos, o segundo troca dois conjuntos de vértices entre os caminhos selecionados e os caminhos com vértices não incluídos. Ambos os algoritmos *Tabu Search* apenas utilizam o segundo salto. Um algoritmo apenas utiliza soluções viáveis *(TSF)*, enquanto que o outro também aceita soluções inviáveis *(TSU)*. O *VNS* também utiliza o *Tabu Search* como procura local, mas com muito menos iterações e apenas considera soluções viáveis. Sempre que a solução incumbente é

melhorada, é utilizando o *2-Opt* [16] para reduzir a duração do caminho. Para comparar diferentes soluções são utilizadas cinco funções baseadas no prémio, duração e viabilidade da solução. Archetti et al. [27] apresentam os melhores resultados para os problemas OP e TOP. A principal razão será, provavelmente, porque eles agrupam os vértices não incluídos em caminhos.

Ke et al. [28] implementou a otimização de uma colónia de formigas *(Ant Colony Optimization, ACO)* para solucionar instâncias de TOP. Em cada iteração, cada formiga construía uma solução viável, que é melhorada através de um procedimento de procura local. Subsequentemente, os trilhos de feromonas[3] são atualizados. O algoritmo pára quando o número máximo de iterações for alcançado. O procedimento de procura local utilizado consiste em encurtar o caminho, introduzindo o maior número de vértices possível. Este procedimento é feito até que o máximo local seja alcançado. Na *framework ACO* são propostos quatro métodos diferentes para alcançar soluções viáveis. Os métodos propostos são:

- Método sequencial *(ASe)*– caminhos completos são construídos um a seguir ao outro.
- Método concorrentemente-aleatório *(ARC)* – em cada iteração, um caminho é selecionado aleatoriamente e é adicionado um novo vértice.
- Método concorrentemente-determinístico *(ADC)* – a sequência de caminhos a considerar é fixa.
- Método simultâneo *(ASi)* – em cada iteração, um vértice é adicionado a cada caminho até que o limite de comprimento seja atingido em todos os caminhos.

Aparentemente o método sequencial é aquele que oferece melhores resultados tendo em conta a qualidade das soluções e o tempo de computação necessário. A qualidade das soluções do método sequencial são, no pior caso, tão boas como os resultados obtidos em Archetti et al. [27] e muito mais rápidas.

Vansteenwegen *et al.* [29] *e* [30] foram os primeiros a colocar o seu foco em conseguir obter boas soluções para o TOP em poucos segundos de tempo de computação. Inicialmente implementaram uma *framework* procura local guiada *(Guided Local Search, GLS)* e mais tarde a *framework skewed variable Neighbourhood Search, (SVNS).* Ambos os algoritmos aplicam procedimentos de intensificação e diversificação. Dois procedimentos de diversificação simplesmente removem um conjunto de vértices em cada caminho. Existe um procedimento que tenta juntar o tempo disponível espalhado por diferentes caminhos da solução corrente, num único caminho, numa nova

[3] Substância segregada por um animal e reconhecida por animais da mesma espécie na comunicação e no reconhecimento.

solução. Dois tipos de procedimentos de intensificação, um para tentar aumentar o prémio total e o segundo para tentar diminuir o tempo de viagem de um caminho. O algoritmo SVNS supera claramente o algoritmo GLS, e é computacionalmente mais rápido. O sucesso do algoritmo SVNS pode ser explicado recorrendo à combinação de uma série de fatores. Em primeiro lugar, a *framework* SVNS parece ser apropriada para este tipo de problemas. A aceitação de soluções intermédias ligeiramente piores quando se está longe da solução incumbente é uma boa estratégia para selecionar vértices que vão fazer parte da solução ótima.

Souffriau [31] criaram duas variantes do procedimento *Greedy Randomised Adaptative Search* (GRASP) com religação de caminhos *(Path Relinking)*. Através da alteração do critério de paragem é possível alternar entre uma versão mais lenta, mas com excelentes resultados, e uma versão mais rápida com resultados bastante satisfatórios. Neste algoritmo são executados quatro procedimentos em sequência até não ser possível melhorar a solução obtida durante um número fixo de iterações. No início, o procedimento de construção gera uma solução inicial. Vão sendo inseridos vértices , um por um, aos caminhos, com base no rácio entre a ganância e a aleatoriedade *(greediness and randomness)*, até que todos os caminhos fiquem completos. Devido à aleatoriedade, a cada iteração é criada uma nova solução inicial. Depois, a solução é melhorada usando pesquisa local (de vizinhos) *(local search neighbourhoods).* O procedimento de pesquisa local alterna entre a redução do tempo total da solução e o aumento do prémio total, até que a solução seja, localmente, a ótima. De seguida a religação do caminho introduz uma lista das soluções de elite como um componente de memória de longo prazo. Para além disso, são consideradas soluções num caminho virtual no espaço de pesquisa entre a solução da pesquisa local e cada solução de elite. É retornada a melhor solução encontrada nesses caminhos. Por fim, a lista de soluções de elite é atualizada. O algoritmo faz o rastreio continuo à melhor solução encontrada durante todas as iterações. A qualidade dos resultados da variante mais lenta é comparável com a qualidade obtida pelos melhores algoritmos de Archetti et al. [27] e Ke et al. [28].

Na Figura 1 são apresentadas as performances dos melhores algoritmos de TOP. A comparação é baseada em 157 instâncias de *benchmark* de Chao et al. [15]. Para cada algoritmo, é apresentado o número de vezes que esse algoritmo apresentou a melhor solução *(#Best)* e a média percentual da distância à melhor solução *(Avg gap (%))*, bem como o tempo de computação médio em segundos *(Avg CPU(seconds))*. Para cada algoritmo, apenas o melhor resultado (de várias execuções) é tido em conta.

Reference	Computer specifications	Technique	Algorithm	# best	Avg gap (%)	Avg CPU (seconds)
Tang and Miller-Hooks (2005)	DEC Alpha XP1000, 1 GB RAM, 1.5 GB swap	Tabu search	TMH	34	1.32	336.6
Archetti et al. (2007)	Intel Pentium 4, 1 GB RAM, 2.8 GHz	Tabu search	TSF	94	0.20	531.5
			TSU	69	0.49	318.0
		Variable neighbourhood search	SVN	**128**	**0.05**	906.1
			FVN	97	0.18	**63.6**
Ke et al. (2008)	PC, 3.0 GHz	Ant colony optimisation	ASe	**130**	**0.08**	252.3
			ARC	81	0.40	204.8
			ADC	80	0.35	213.8
			ASi	84	0.32	215.0
Vansteenwegen et al. (2009c)	Intel Pentium 4, 1 GB RAM, 2.8 GHz	Variable neighbourhood search	SVNS	44	0.97	**3.8**
Souffriau et al. (in press)	Intel Xeon, 4 GB RAM, 2.5 GHz	Greedy randomised adaptive search procedure with path relinking	FPR	78	0.39	**5.0**
			SPR	**131**	**0.04**	212.4

Figura 1 Performance dos melhores algoritmos TOP

Por forma a dar alguma orientação para desenvolvimento de algoritmos futuros é interessante analisar os movimentos mais comuns usados nas pesquisas locais. Os primeiros 5 movimentos aumentam o prémio total da solução: *Insert*, *TwoInsert*, *Replace*, *TwoReplace* e *Chance*. Os outros dois reduzem o tempo de viagem entre os dois vértices: *2-Opt* e *Swap*. Os algoritmos TOP eficientes alternam entre movimentos de aumento do prémio e movimentos de diminuição do tempo de viagem.

1. O movimento de inserção *(insert move)* tenta incluir um vértice extra em qualquer um dos caminhos, usando a inserção de menor custo *(cheapest insertion)*.
2. O movimento de inserção dupla *(TwoInsert)* tenta incluir dois vértices extra. Para cada combinação de dois vértices não incluídos, a posição que menos tempo consome para cada vértice é considerada.
3. O movimento de substituição *(replace)* considera para inserção todos os vértices que não estão incluídos. Se houver tempo disponível para inserção, o vértice é inserido. Se a inserção não for possível, é considerado um vértice com prémio menor que respeite a restrição temporal.
4. O movimento de substituição dupla *(TwoReplace)* considera, para inserção, todas as combinações de dois vértices não incluídos. Se for necessário, consideram-se dois vértices previamente inseridos, para exclusão de forma a ser possível a inserção.
5. A movimento de mudança *(Change)* gera vizinhos de maneira oposta ao movimento substituição *(replace)* e substituição dupla *(TwoReplace)*. Este movimento primeiro remove cinco vértices sucessivos de um caminho e depois tenta inserir vértices não incluídos, um por um, até que mais nenhum vértice possa ser inserido no caminho. A nova solução só é aceite caso o prémio total aumente.
6. *2-Opt* [16] é um movimento muito popular para reduzir o tempo de viagem entre dois vértices. O movimento remove dois vértices do caminho e insere dois novos, não incluídos

anteriormente no caminho. O caminho tem que permanecer fechado e o prémio total tem que diminuir.

7. O movimento de troca *(Swap)* consiste em trocar dois vértices pertencentes a dois caminhos diferentes, no sentido de diminuir o tempo de viagem.

A Figura 2 mostra os movimentos utilizados nas diferentes heurísticas TOP.

	Insert	*TwoInsert*	*Replace*	*TwoReplace*	*Change*	*2-Opt*	*Swap*
Tang and Miller-Hooks (2005)	✓				✓		✓
Archetti et al. (2007)	✓					✓	✓
Ke et al. (2008)	✓					✓	
Vansteenwegen et al. (2009b)	✓		✓			✓	✓
Vansteenwegen et al. (2009c)	✓	✓	✓	✓	✓	✓	✓
Souffriau et al. (in press)	✓		✓			✓	✓

Figura 2 Movimentos utilizados nas diferentes heurísticas

2.4.4 Casos de aplicação

Butt e Cavalier [32] descrevem uma aplicação do TOP em recrutamento de atletas do ensino secundário. O recrutador tem um dado número de dias para visitar as escolas. Pode ser atribuído um prémio a cada escola baseado no potencial dos atletas. Como o tempo é limitado, o objetivo é maximizar o potencial recrutado. Também Tang e Millller-Hooks [26] descrevem a aplicação do TOP em encaminhamento de técnicos para serviços ao cliente em que cada caminho TOP representa um único técnico, que têm um número limitado de horas de trabalho diário. É escalonada uma lista de clientes baseada na urgência do serviço e também a importância do cliente.

2.5 Problema de orientação com janelas temporais

Os OPTW são problemas muito semelhantes aos OP mas com restrições temporais nos nodos a visitar. Isto é, a cada nodo é imposto um "horário de funcionamento" em que não poderá ser visitado fora desse período. Embora os problemas sejam parecidos, os métodos para resolução são diferentes. Por exemplo, com o movimento 2-opt obtém-se soluções de bastante qualidade em problemas de OP, no entanto este método não pode ser aplicado para resolver, eficientemente, o OPTW devido à restrição das janelas temporais.

2.5.1 Definição do problema

Nos OPTW, a cada vértice é atribuído uma janela temporal $[O_i, C_i]$, composto pelos valores da *abertura da janela* (O_i) e do fecho da janela (C_i). A visita a cada vértice só pode ser realizada caso

se respeite estas restrições, ou seja, o início da visita tem que ser feito depois do O_i e antes do C_i. O OPTW pode ser formulado como um problema inteiro misto com as seguintes variáveis de decisão: $x_{ij} = 1$ se a visita ao vértice i é seguida da visita ao vértice j, 0 caso contrário; $s_i =$ início do serviço no vértice i; M é uma constante muito grande.

$$Max \sum_{i=2}^{N-1} \sum_{j=2}^{N} S_i x_{ij}, \tag{16}$$

$$\sum_{j=2}^{N} x_{1j} = \sum_{i=1}^{N-1} x_{iN} = 1 \tag{17}$$

$$\sum_{i=1}^{N-1} x_{ik} = \sum_{j=2}^{N} x_{kj} \leq 1 \; ; \; \forall k = 2, \dots, N-1; \tag{18}$$

$$S_i + t_{ij} - S_j \leq M\left(1 - x_{ij}\right); \; \forall i, j = 1, \dots, N, \tag{19}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} t_{ij} x_{ij} \leq \; T_{max}, \tag{20}$$

$$O_i \leq s_i; \; \forall i = 1, \dots, N, \tag{21}$$

$$s_i \leq C_i; \; \forall i = 1, \dots, N, \tag{22}$$

$$x_{ij} \in \{0,1\}; \; \forall i, j = 1, \dots, N. \tag{23}$$

Modelo 3 Modelo matemático OPTW

A função objetivo (16) maximiza o prémio total coletado. Com a restrição (17) garante-se que o caminho tem início no vértice 1 e termina no vértice *N*. Com a restrição (18) assegura-se a conectividade do caminho e que cada vértice é visitado, no máximo, uma única vez. Com a restrição (19) garante-se que a janela temporal do vértice é respeitada. A restrição (20) garante que tempo limite imposto é respeitado. As restrições (21) e (22) restringem o início de visita ao da janela temporal.

2.5.2 Complexidade

Não existe nenhuma referência quanto à complexidade deste tipo de problemas, mas no seguimento dos problemas de OP, e dado que a única alteração ao modelo é a inclusão de restrições temporais nos vértices, então também estes pertencem à classe *NP-Difícil*.

2.5.3 Abordagens de solução

Kantor e Rosenwein [33] foram os primeiros a resolver problemas de OPTW. Inicialmente descrevem uma heurística de inserção direta. O vértice com o maior rácio "prémio sobre o tempo de inserção" é inserido no caminho, sem violar as restrições da janela temporal. Depois, propõe um algoritmo de prioridade à profundidade *(depth-first)*, que constrói caminhos parciais utilizando a heurística de inserção e com início no vértice inicial. Caminhos parciais inviáveis ou que muito provavelmente não conseguiram atingir o melhor prémio total são abandonados. Righini e Salani [34] criaram um algoritmo exato, com recurso à programação dinâmica bidirecional, para solucionar otimamente problemas de OPTW. Do vértice inicial para a frente e do vértice final para trás, o estado corrente é estendido adicionando um vértice extra no final. Para a frente e para trás os estados combinam se os testes de dominância e viabilidade forem aplicados para registar apenas estados não dominados *(non-dominated states)*. A diminuição do estado do espaço de relaxamento é utilizado para reduzir o número de estados a ser explorados [35]. *Mansini et al. (2006)* desenvolveram uma heurística construtiva simples e granular *variable neighbourhoods search (VNS)* para um caso especial do OPTW, em que o vértice inicial e o vértice final são o mesmo. O *GVNS* é um aperfeiçoamento ao algoritmo *VNS* reduzindo o número de vizinhos analisados e prevenindo a inserção de arcos não promissores.

2.5.4 Casos de aplicação

Não existem muitos casos específicos de aplicação do OPTW na literatura, mas na maioria dos casos de aplicação descritos em 2.3 têm, na verdade, restrições de ao nível de janela temporal. Veja-se a aplicação do caixeiro viajante, em que os seus clientes da cidade podem não estar disponíveis para receber o caixeiro o dia todo, mas antes apenas em certas horas específicas. Ou ainda no caso da entrega de combustível ao domicílio, as pessoas são diferentes e consequentemente poderão ter disponibilidades temporais diferentes.

2.6 Problema de orientação de equipas com janelas temporais

2.6.1 Definição do problema

O TOPTW é, essencialmente, o OPTW mas com vários veículos disponíveis em simultâneo para visitar os vértices. Respeitando as mesmas regras, ou seja, cada vértice tem uma janela temporal e este apenas pode ser visitado nesse período de tempo.

Recorrendo à mesma notação utilizada anteriormente, o TOPTW pode ser formulado como sendo um problema de programação inteira mista com as seguintes variáveis de decisão: $x_{ijp} = 1$ se no caminho **c** a visita ao vértice *i* for seguida da visita ao vértice j, 0 caso contrário; $y_{ip} = 1$ se o vértice i é visitado no caminho p, 0 caso contrário; $S_{ip} =$ início do serviço no vértice *i* do caminho *c*; M um número muito grande.

$$Max \sum_{p=1}^{p} \sum_{i=2}^{N-1} S_i Y_{ip}, \tag{24}$$

$$\sum_{p=1}^{p} \sum_{j=2}^{N} X_{1jp} = \sum_{p=1}^{p} \sum_{i=1}^{N-1} X_{iNp} = P, \tag{25}$$

$$\sum_{i=1}^{N-1} x_{ikp} = \sum_{j=2}^{N} x_{kjp} = y_{kp};\ \forall k = 2, \dots, N-1; \forall p = 1, \dots, P \tag{26}$$

$$s_{ip} + t_{ij} - s_{jp} \leq M\left(1 - x_{ijp}\right);\ \forall i, j = 1, \dots, N; \forall p = 1, \dots, P \tag{27}$$

$$\sum_{p=1}^{p} y_{kp} \leq 1, \forall k = 2, \dots, N-1, \tag{28}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} t_{ij} x_{ijp} \leq T_{max};\ \forall p = 1, \dots, P, \tag{29}$$

$$O_i \leq S_{ip};\ \forall i = 1, \dots, N;\ \forall p = 1, \dots, P \tag{30}$$

$$S_{ip} \leq C_i;\ \forall i = 1, \dots, N;\ \forall p = 1, \dots, P \tag{31}$$

$$x_{ijp}, y_{ip} \in \{0,1\};\ \forall i, j = 1, \dots, N;\ \forall p = 1, \dots, P \tag{32}$$

Modelo 4 Modelo matemático TOPTW

A função objetivo (24) maximiza o prémio total recolhido. A restrição (25) garante que todos os caminhos iniciam no vértice 1 e terminam no vértice N. As restrições (26) e (27) determinam a

conectividade e o tempo de execução de cada caminho. A restrição (28) garante que cada vértice é visitado, no máximo, uma vez e a restrição (29) garante que os caminhos terminam todos no tempo disponível t_{max} . As restrições (30) e (31) controlam o início da visita para que seja dentro da janela temporal.

2.6.2 Complexidade

O TOPTW é um problema com muitas restrições e bastante difícil de solucionar. Golden [7] provou que o problema OP pertence à classe dos problemas *NP-difícil*. E é altamente improvável que o TOPTW possa ser resolvido, otimamente, em tempo polinomial. [36]

2.6.3 Abordagens de solução

Montemanni and Gambardella [37] basearam o seu algoritmo de resolução TOPTW na otimização de uma colónia de formigas. Este algoritmo supera o algoritmo de *Mansini et al. (2006)* [38] em todas as instâncias OPTW consideradas. *Tricoire et al.* [39] solucionou o Problema de Orientação multiperíodo com múltiplas janelas temporais, uma generalização do TOPTW. Neste problema cada vértice pode ter mais que uma janela temporal em determinado dia e as janelas temporais podem ser diferentes nos diferentes dias. Propuseram o algoritmo VNS e embutiram também um algoritmo exato para lidar com o problema da viabilidade dos caminhos. Os resultados mostram que obtiveram soluções de elevadíssima qualidade para problemas com cem vértices e dois caminhos em apenas um minuto de computação. A aplicação descrita por Tricoire et al. [39] é sobre facilitar o planeamento de trabalhos de dias futuros. A qualidade das soluções são piores comparativamente com o *Montemanni and Gambardella* [37], mas o tempo de computação reduz consideravelmente. Entre Vansteenwegen et al. [36], *Montemanni and Gambardella* [37] e Tricoire et al. [39] é impossível fazer uma comparação detalhada, uma vez que todos eles usaram instâncias de *benchmark* ligeiramente diferentes, contudo conclui-se que a solução utilizada por ILS *de* Vansteenwegen et al. [36] tem a vantagem de obter soluções rapidamente, enquanto que as soluções de *Montemanni and Gambardella (2009) e* Tricoire et al. [39] são de melhor qualidade.

Os problemas de encaminhamento de veículos seletivos com janelas temporais *(Selective Vehicle Routing Problem with Time Windows, SVRPTW)* é uma generalização de TOPTW. No *SVRPTW* foram adicionadas mais 2 restrições. A primeira restrição trata de adicionar capacidade aos veículos, enquanto que cada cliente tem uma necessidade definida. A segunda restrição limita a distância dos

caminhos. Importante notar aqui que impor restrições de distância é diferente de impor restrições de tempo. Na prática, todos as visitas a vértices vão requerer tempo de visita (ou tempo de serviço). As limitações de tempo não só limitam o tempo de viagem, como também o tempo de visita. O tempo de visita não é considerado quando se limita apenas a distância. SVRPTW corresponde a um VRPTW em que nem todos os vértices podem ser visitados. Boussier et al.[23] conseguiram solucionar instâncias de 100 vértices com até 10 caminhos, recorrendo a um algoritmo exato de *branch-and-price*.

2.6.4 Casos de aplicação

Pode-se utilizar o TOPTW para modelar uma grande variedade de problemas de decisão incluindo planeamento de viagens de turísticas ou a prestação de serviços a clientes com horários de funcionamento bem definidos e espalhados geograficamente, recorrendo a uma frota de veículos. Entre outras aplicações, o TOPTW pode ser aplicado para resolver o problema de afetação de cirurgias hospitalares às salas de operações e equipas [40]. Outro problema onde o TOPTW pode ser aplicado é na recolha de resíduos municipal, onde existem pontos de recolha, veículos à disposição para efetuar a recolha, bem como uma calendarização para a rota do veículo [41].

2.7 Variantes de problemas de orientação

Os OP podem ser formulados como um problema do caixeiro viajante com recompensas *(Travelling salesman problem with profits, TSPWP)* [42][43] ou ainda como um caso especial de *Resource Constrained TSP (RCTSP) (Pekny and Miller, 1990)*[44].

O problema de TSPWP pode ser visto como um problema de TSP de duplo critério com dois objetivos opostos: maximizar as recompensas recolhidas e minimizar os custos da viagem. Quando o custo da viagem é uma restrição, o problema passa a pertencer à classe dos TOP. Classificações e técnicas de solução do TSPWP podem ser encontradas em Feillet *et al.* [42]. Nos problemas de *RCTSP* é dado um conjunto de vértices e a viagem entre dois vértices incorre num custo de viagem e no consumo de um recurso. O objetivo é encontrar um caminho de custo mínimo que visite todos os vértices enquanto que consumo do recurso é limitado a um certo valor. Este problema pode ser convertido num problema de orientação, se não for estritamente necessário visitar todos os vértices, o custo de viagem entre dois vértices for substituído por um prémio de valor negativo e se o custo de viagem for considerado como recurso consumível.

Outra variante dos problemas de orientação são os problemas de orientação generalizada *(Generalised Orienteering Problem, GOP)* [45]–[47], problemas de orientação multiobjectivo [48], problemas de orientação com recompensas estocásticas [49], problemas e orientação com vértices compulsórios *(Compulsory vertices for the OP)*, problemas de orientação com tempo dependente [50] e problema de orientação de equipas com capacidade *(Capacitated TOP)* [51]. A diferença entre OP e GOP é a função objetivo. Em problemas de orientação o prémio total é alcançado através da soma dos prémios de cada vértice visitado, enquanto que em GOP o prémio total é uma função mais complicada (não linear) dos vértices não visitados. Neste caso, uma certa combinação de vértices pode gerar uma solução com o melhor prémio total, ou mais baixo, do que a soma dos vértices individualmente. Por exemplo, no sector turístico pode haver certas atrações com variações temáticas, e de maneira a ser verdadeiramente apreciadas, todos elas devem ser visitadas. Também pode acontecer que um ponto turístico de baixo valor, veja o seu valor acrescentado quando visitado em conjunto com outro ponto. Por exemplo, a visita a um artesanato pode ser mais apreciado se visitado depois de um museu relacionado. Schilde et al. [48] modelam diferentes interesses que um turista terá em diferentes categorias de atração como um problema de orientação multiobjectivo. Além disso, apresentam uma solução para lidar com problemas de orientação com dois objetivos.

Nos problemas de orientação com recompensas estocásticas *(OPSP)*, cada vértice tem um prémio e o objetivo é maximizar a probabilidade e recolher um nível previamente estabelecido, dado um tempo limite. Ilhan et al. [49] desenvolveu um algoritmo exato e um algoritmo genético biobjetivo para lidar com problemas OPSP.

Problemas de orientação com vértices compulsórios foi inicialmente mencionado em [12]. Neste tipo de instância, um determinado conjunto de vértices tem que ser, obrigatoriamente visitado. No caso de uma empresa que utilize OP para o planeamento diário, os vértices obrigatórios podem ser os clientes com marcação para hoje, e a visita não pode ser adiada. Em planeamento turístico, esses vértices podem ser pontos de atração forte, que devem ser incluídas em todas as visitas personalizadas. [12] desenvolveram diferentes classes de inequações válidas e usou branch-and-cut para solucionar, otimamente, problemas com 100 vértices em que alguns deles eram compulsórios.

Fomin e Lingas [50] apresentam os Problemas de Orientação com tempo dependente (time-dependent OP). Neste caso o tempo de viagem entre dois vértices depende do tempo de saída do primeiro vértice. Uma aplicação deste problema são os robots que necessitam de interceptar o maior número de alvos em movimento possível, num determinado tempo limite. Em cada momento, a localização dos alvos é conhecido, bem como os tempos de viagem (time-dependent).

Problemas de orientação de equipas com capacidade (CTOP), inicialmente apresentado em Archetti et al. [51], é uma variante do TOP onde, a cada vértice é associada um valor de exigência e o total de exigências em cada caminho não pode exceder uma certa capacidade. Na maioria dos problemas reais, a capacidade dos veículos é uma questão a ter em conta.

Também os problemas encaminhamento de arcos *(arc Routing)* são considerados uma variante de OP onde o prémio está associada a cada arco. Gendreau et al. [52] apresentam o *Ring Network Design Problem, RNDP* com uma aplicação em comunicações. *Feillet et al. (2005b)* [53] propuseram um algoritmo *branch-and-price* para o *Profitable Arc Tour Problem, PATP.* A aplicação considerada foi o transporte de cargas na indústria automóvel. Aráoz et al. [54] apresentam um problema semelhante chamado *Prize-Collecting Rural Postman Problem*, *PCRPP*. Apresentaram a aplicação de recolha de ecopontos. Quando a recolha é planeada por uma empresa publica, todos os arcos de uma área devem ser servidos. Contudo, quando planeada por uma empresa privada, esta deve apenas selecionar e servir as ruas mais vantajosas, ou seja, aquelas que oferecem os prémios mais elevados.

Nos problemas RNDP, PCRPP e PATP o objetivo é maximizar a diferença entre o prémio total recolhido e o tempo de viagem.

Archetti et al. [55] descrevem o undirected capacitated arc routing problem with profits que corresponde a um capacitated arc TOP. O objetivo é determinar um caminho para cada veículo disponível, sem infringir as restrições de capacidade e de tempo limite de cada veículo. Eles consideram uma aplicação onde as operadoras podem selecionar os potenciais clientes e transportar os seus bens. Li e Hu formularam o team orienteering problem with capacity constraint and time windows, TOPCTW (uma derivação do TOPTW em que cada cliente tem uma necessidade e o veículo de serviço tem limitação de capacidade) e obtiveram soluções exatas utilizando um solver de programação linear [56]. Outra potencial aplicação é a conceção de viagens de bicicleta personalizadas, baseada nos pessoais do ciclista e, dado um ponto inicial, um ponto final e um determinado tempo, é criada uma rota personalizada usando uma seleção de arcos que mais se enquadrem com o perfil do ciclista.

Mais recentemente, Van Der Merwe et al. [57] apresentaram o *The Cooperative Orienteering Problem With Time Windows, COPTW*, que requer a utilização de múltiplos veículos para, cooperativamente, recolher os prémios associados aos diferentes vértices. A demonstração do *COPTW* é aplicada a um cenário de incêndio florestal em *South Hobart, Tasmânia Austrália*. Os testes computacionais indicam que é viável a aplicação do COPTW a problemas reais.

3. CASO DE ESTUDO

Neste capítulo são apresentados os principais aspetos que caracterizam o problema de escalonamento de recursos de engarrafamento móvel aos clientes. São também descritas as metodologias e ferramentas utilizadas na resolução do problema. É apresentada uma descrição detalhada do problema real, descriminando os recursos disponíveis e eventuais limitações por parte dos clientes e/ou nos veículos prestadores de serviço, nomeadamente as janelas temporais, das quais se tentará transportar para o modelo matemático.

A empresa têm clientes a requisitar os seus serviços ao longo do ano, podendo haver um pico de procura com a aproximação da época das vindimas de modo a libertar as cubas onde está armazenado o vinho.

3.1 Descrição do problema real

Hoje em dia, para os pequenos e médios produtores torna-se muitas vezes incomportável investir em equipamento de engarrafamento e rolhamento devido ao elevado custo inicial e subsequentes custos de manutenção. A estes custos, acrescem ainda os custos com tratamento da vinha, fertilizantes, entre outras despesas, o que encarece os custos de produção. Por outro lado, a incerteza das condições climatéricas não permite assegurar o nível de receitas. Por estas razões, os produtores procuram cada vez mais soluções viáveis para fazer o serviço de engarrafamento, nomeadamente entidades com formação, *know-how* e capacidade técnica.

A empresa de engarrafamento móvel tem ao seu dispor uma frota de veículos para prestar o serviço ao seu cliente, como será apresentado em 3.2. Atualmente, os pedidos para o serviço de engarrafamento é efetuado pelo cliente com, pelo menos, 30 dias de antecedência. Não há informação quanto ao período de tempo que decorre desde a solicitação do serviço, por parte do cliente, até à confirmação da data, no entanto a resposta tem que ser dada atempadamente ao cliente para que este tenha tempo de fazer o planeamento e, eventualmente, convocar os recursos e mão de obra necessários para ajudarem no serviço. A empresa tem bons clientes, que de maneira alguma poderá deixar de servir em detrimento de um outro cliente menos importante. Todos os clientes são importantes, mas uns são mais importantes do que outros, existindo assim o conceito de cliente

prioritário. Existe também o conceito de janela temporal, ou seja, o cliente informa, antecipadamente, a empresa de um intervalo de datas disponíveis para a qual quer ver o serviço realizado. Os pedidos dos clientes são expressos em litros e é portanto possível antecipar os tempos de processamento necessários. Os veículos têm custos de manutenção e outros custos diretamente relacionados com a distância percorrida pelo mesmo, nomeadamente combustível. Por forma a garantir a sustentabilidade da empresa os pedidos de serviço aos clientes apenas são considerados se forem superiores a uma quantidade mínima. Na modelação assume-se que a quantidade mínima é de 1000 litros. O serviço de engarrafamento exige um tempo de preparação de 20 minutos por tipo de vinho. No final é necessário também 20 minutos para lavar o veículo e deixá-lo em condições de ser utilizado pelo próximo serviço. Os clientes estão localizados em Portugal Continental com maior incidência nas regiões vinhateiras. É de todo o interesse da empresa conseguir agregar, da melhor forma possível, os pedidos de visita de maneira a conseguir reduzir os custos associados às deslocações aos clientes na prestação dos diferentes serviços.

3.2 Recursos disponíveis

Para dar resposta aos vinicultores, a empresa tem ao seu dispor recursos, unidades móveis, em constante atualização para garantir a máxima qualidade e segurança na prestação dos diferentes serviços.

Desses recursos fazem parte as seguintes:

- Unidade Móvel de **Desalcoolização e Concentração de Vinhos** com capacidade de 100Hl por dia. A Figura 3 ilustra a unidade móvel de desalcoolização e concentração de vinhos.

Figura 3 Desalcoolização e concentração de vinhos

- Unidade Móvel de **Filtração Tangencial (SIN)** com capacidade de 3000L por dia. A Figura 4 ilustra a unidade móvel de filtração tangencial (sin).

Figura 4 Filtração tangencial (SIN)

- Unidade Móvel de **Estabilização tartárica** com capacidade de 600Hl por dia. A Figura 5 ilustra a unidade móvel de estabilização tartárica.

Figura 5 Estabilização tartárica

- Unidade Móvel de **Esterilização a Frio por DMDC** com capacidade de 1200 a 7200 litros por Hora. A Figura 6 ilustra a unidade móvel de esterilização a frio por DMDC.

Figura 6 Esterilização a frio por DMDC

- Três tipos de unidades móveis de engarrafamento, a UME01, UME02 e UME03 com capacidades de engarrafar 2000 garrafas por hora, 3500 garrafas por hora e 2500 garrafas por hora, respetivamente. As figuras 7 a 9 apresentam as diferentes unidades móveis de engarrafamento da empresa.

Figura 7 Unidade Móvel de engarrafamento 1

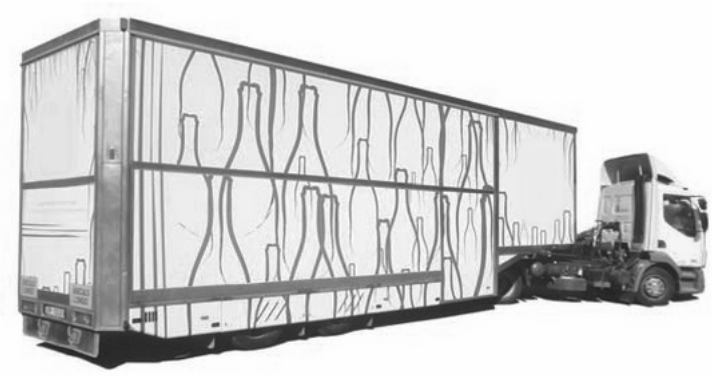


Figura 8 Unidade Móvel de engarrafamento 2

Figura 9 Unidade Móvel de engarrafamento 3

- Duas Unidades Móveis de **Acabamento,** UMA01 e UMA02, com capacidades de 1500 e 3000 garrafas por Hora. As Figuras 10 e 11 ilustram as unidades móveis de acabamento da empresa.

Figura 10 Unidade Móvel de Acabamento 1

Figura 11 Unidade Móvel de acabamento 2

- Unidade Móvel de **renovação da barricas** com capacidade de renovar entre 12 a 15 barricas por dia. A Figura 12 mostra a unidade móvel de renovação de barricas da empresa.

Figura 12 Unidade Móvel de renovação de barricas

3.3 Modelação

Por questões de dimensão e relevância, este estudo incide unicamente no serviço de engarrafamento, não sendo considerados os restantes serviços que a empresa disponibiliza. Assumiu-se também que a unidade móvel é do tipo UME02, ilustrada na Figura 8, que tem capacidade de resposta de 2500 garrafas por hora.

Os pedidos dos clientes são expressos em litros, pelo que terá que ser convertido em número de garrafas, uma vez que a capacidade de resposta dos recursos está expresso nessas unidades. Considera-se que as garrafas são de 0.75cl e que os tempos de preparação são considerados e calculados consoante o número de vinhos diferentes que o cliente quer engarrafar. Em seguida, será calculado o tempo de serviço necessário para realizar as tarefas para satisfazer um cliente. Assume-se que o lucro resultante de um serviço realizado está também diretamente associado ao número de litros a processar.

O horizonte de planeamento adotado é variável, pois não existem entraves com questões logísticas como, por exemplo, os motoristas não terem obrigatoriamente que voltar ao ponto de partida com o veículo no final de cada dia. O que acontece frequentemente é o serviço ficar por terminar no final de um dia de trabalho, sendo o motorista obrigado a deixar o atrelado da unidade móvel,

retirando-se do cliente apenas com o trator. Assumiu-se que o horário laboral é de oito horas de trabalho mais uma hora extraordinária por dia, que resulta em nove horas diárias.

Este modelo está preparado para lidar com diferentes disponibilidades impostas pelos diferentes clientes. A disponibilidade de cada cliente depende de diversos factores, entre eles a necessidade do cliente encontrar colaboradores para ajudar na realização de trabalhos associados ao engarrafamento que geralmente exige mão de obra. Esta noção de disponibilidade é transportada para o modelo através das restrições das janelas temporais.

Nos problemas TOP, dado um número de vértices com prémio de visita associado, o objetivo principal é determinar, num determinado espaço de tempo, um caminho para visitar os vértices de forma a maximizar o prémio total recolhido. Assume-se que o prémio total é a soma dos prémios dos vértices visitados. A fórmula (33) calcula o prémio de visita que é baseada, essencialmente, na prioridade do cliente e quantidade de litros a processar. A fórmula é a seguinte:

$$premio_i = litros_i * prioridade_i \quad (33)$$

Existem situações onde a empresa tem que optar por visitar um cliente em detrimento de outro, quando não tem meios suficientes para satisfazer todos os pedidos. Por forma a tentar transportar essas situações para o modelo, definiu-se uma escala de prioridades de clientes que terá impacto no prémio recolhido. A Tabela 1 mostra a escala de prioridades dos clientes e o respetivo peso no prémio de visita.

Prioridade	Peso
Cliente de prioridade alta	3
Cliente de prioridade moderada	2
Cliente normal	1

Tabela 1 Escala de prioridades de clientes

O prémio de visitar um cliente está diretamente relacionado com o número de litros a processar nesse mesmo cliente e consoante a sua prioridade. Ou seja, se um determinado cliente com prioridade normal tem 1000 litros para processar, o prémio por visitar este cliente é de 1000. Se o cliente tem os mesmos 1000 litros para processar e prioridade moderada, o prémio de visita será de 2000. Já se o cliente tiver 1000 litros para processar e prioridade alta, então o prémio de visita é de 3000.

O tempo de processamento é diretamente proporcional ao número de litros a processar e à quantidade das diferentes variedades de vinho. Assume-se que a capacidade de processamento do veículo é de 2500 garrafas por hora e ainda que é preciso preparar o veículo sempre que haja a necessidade para processar um tipo de vinho diferente, essa preparação demora 20 minutos. A fórmula (34) serve para calcular o tempo de processamento, em horas é:

$$tempoProc_i = \frac{\left(\frac{Litros_i}{0.75}\right)}{2500} + (nVinhos * .33) \tag{34}$$

Para clientes diferentes existem localizações diferentes e as distâncias entre os clientes, previamente calculadas, são armazenadas numa matriz quadrada *distancias*. Assim, para saber se o tempo de viagem entre a origem i e o destino j recorre-se a:

$$distancias_{ij} \tag{35}$$

3.4 Modelo matemático

Para a resolução do problema da empresa de engarrafamento móvel, optou-se por adaptar a modelação TOPTW anteriormente abordada em 2.6 com as alterações necessárias.

As variáveis relevantes no modelo são:

- n – Número de vértices da rede ;
- $carros$ – número de veículos disponíveis para resolver o modelo ;
- $premio_i$ – prémio por visitar $i,\ i = 1..n$;
- $distancias_{ij}$ – tabela com as distâncias entre os nodos i,j com $i = 1..n$ e $j = 1..n$;
- $tpreparacao_i$ – tempo de preparação necessário para engarrafar determinado tipo de vinho em $i, i = 1..n$;
- $tproc_i$ – tempo de processamento em $i,\ i = 1..n$. Este tempo inclui o tempo de preparação e tempo de lavagem final ;
- $pontos_{ij}$ tabela com os dados relevantes de cada nodo, $i = 1..n, j = 1..5$;
- ji_i tempo de abertura da janela temporal no ponto $i,\ i = 1..n$;
- jf_i tempo de fecho da janela temporal no ponto $i,\ i = 1..n$;

As variáveis de decisão do modelo são:

- $x_{c,i,j} = \begin{cases} 0, & caso\ o\ veiculo\ \boldsymbol{c}\ não\ visite\ o\ nodo\ j \\ 1, & caso\ o\ veículo\ \boldsymbol{c}\ visite\ o\ nodo\ j\ depois\ de\ visitar\ o\ nodo\ i \end{cases}$,

$c = 1..carros, i \neq j, i = 1..N-1, j = 2..N$

- $t_i \geq 0,\ i = 1..n$

$$Max \sum_{c=1}^{carros} \sum_{i=1}^{N-1} \sum_{j=2}^{N} x_{c,i,j} * premio_j \quad (36)$$

$$\sum_{j=2}^{N} X_{c1j} = 1, \qquad c = 1..carros \quad (37)$$

$$\sum_{i=1}^{N-1} X_{cij} = \sum_{i=2}^{N} X_{cji}, \quad c = 1..carros, j = 2..N-1 \quad (38)$$

$$\sum_{i=2}^{N-1} X_{cin} = 1, \qquad c = 1..carros \quad (39)$$

$$\sum_{c=1}^{carros} \sum_{i=1}^{N-1} x_{cij} \leq 1, \qquad i \neq j, j = 2..N-1 \quad (40)$$

$$x_{cij} + x_{cji} \leq 1, \qquad i \neq j, c = 1..carros, i = 2..n-1, j = 2..n-1 \quad (41)$$

$$t_1 = 0 \quad (42)$$

$$t_i \geq ji_i, \qquad i = 2..n-1 \quad (43)$$

$$t_i \leq jf_i, \qquad i = 2..n \quad (44)$$

$$t_j \geq \left(t_i + tproc_i + distancias_{ij} x_{cij}\right) - \left(10000 * \left(1 - x_{cij}\right)\right) \quad (45)$$

Modelo 5 Modelo matemático utilizado para solucionar o problema da empresa

O modelo pode ser descrito como um conjunto de n vértices a visitar por um conjunto de veículos. A função objetivo (36) pretende maximizar o prémio total associado aos vértices visitados. A restrição (37) garante que cada veículo só sai uma única vez do vértice inicial e a restrição (42) garante que o tempo de partida no vértice inicial é no início da janela temporal total. A restrição (38) garante o fluxo, ou seja, um veículo ao entrar num vértice sai obrigatoriamente deste, garantindo ainda que as

restrições temporais (43) e (44) são respeitadas e é garantido que o último vértice é visitado com a restrição (39). A restrição (40) garante que para todos os arcos com chegada a um vértice, no máximo apenas um será percorrido, impedindo assim que o mesmo vértice seja visitado por mais que um veículo, sendo também garantido com a restrição (41) que um veículo não regresse ao vértice anterior, evitando que volte atrás. Por fim, uma última restrição (45) garante que o tempo de chegada a um vértice respeita a janela temporal, obrigando a que a chegada para os pontos visitados ocorra depois do tempo de chegada no vértice anterior, o tempo de processamento nesse vértice e a distância efetuada até ao novo vértice. Para situações em que o arco não é percorrido pelo carro é acrescentada uma componente representativa de $-\infty$ permitindo que o tempo seja maior que "menos infinito".

De acordo com o nosso melhor conhecimento esta é a primeira formulação para o problema de engarrafamento móvel, utilizando uma modelação baseada no TOP.

3.5 NEOS Server

O servidor *NEOS (Network-Enabled Optimization System)* é um projeto da universidade de Wisconsin – Madison. É um serviço *web* disponibilizado gratuitamente com o intuito de resolver problemas de otimização. Os solucionadores *(solvers)* disponibilizados pela plataforma representam o estado da arte na área de otimização. O servidor *NEOS* oferece uma abordagem simples para a resolução de uma vasta variedade de problemas de otimização e fornece várias interfaces para aceder aos solucionadores. Depois de selecionar o solver e fornecer uma descrição do problema de otimização o servidor NEOS determina a informação adicional exigida pelo solver, liga o problema de otimização com o solver, e após concluído, retorna uma solução.

3.6 Gurobi

O problema objeto de estudo nesta dissertação enquadra-se no âmbito de problemas de programação inteira mista, modelado em AMPL. Embora existam solucionadores que conseguem dar respostas mais detalhadas, optou-se pelo *Gurobi* por entender-se ser um solucionador com tempos de resposta bastante aceitáveis. A Tabela 2 mostra os solucionadores disponíveis no NEOS Server e quais permitem entrada de modelo em AMPL.

Solucionadores	Usa AMPL
Cbc	Sim
feaspump	Sim
Gurobi	Sim
MINTO	Sim
MOSEK	Não
Proxy	Não
Qsopt	Sim
Scip	Sim
SYMPHONY	Não
XpessMP	Sim

Tabela 2 Solucionadores NEOS Server

3.7 AMPL

A programação matemática de problemas de larga escala é mais do que aplicar um algoritmo para maximizar ou minimizar uma função objetivo. Antes de uma rotina de otimização poder ser invocada é indispensável formular um modelo base e gerar estruturas de dados computacionais necessárias. AMPL *(A Mathematic Programming Language)* é uma linguagem, criada por Robert Fourer em 1985, que torna esta tarefa mais fácil e menos propícia a erros. AMPL é bastante semelhante à notação algébrica simbólica. É uma linguagem de programação para descrever produção, distribuição, calendarização e muitos outros tipos de problemas geralmente conhecidos por problemas de otimização de larga escala ou problemas matemáticos.

Os modelos AMPL são organizados em quarto partes que são:

- **modelo:** que contém uma descrição do modelo a ser usado.
- **data:** que contém informação referente ao modelo já definido.
- **opções:** opções a ter em conta quando executado, disponível em alguns solucionadores.
- **solve:** indica ao *solucionador* que deve executar.

A escolha pela linguagem em AMPL deve-se por ser intuitiva e de fácil percepção, para além do facto de ter sido abordada nas aulas lecionadas de mestrado.

3.8 Aplicação web

Numa fase mais embrionária do projeto, o objetivo incluía também a criação de uma aplicação web para gerir os clientes, os serviços de engarrafamento móvel e as rotas a efetuar pelos diferentes veículos. Depois, tendo em conta as disponibilidades dos clientes, paralelamente à aplicação, era intenção utilizar um modelo TOPTW para enviar problemas ao NEOS Server e analisar as soluções. Optou-se por diminuir o trabalho na aplicação e aumentar com a componente do servidor NEOS Server. No entanto a aplicação está parcialmente concebida e tem alguns mecanismos que foram muito importantes para automatizar processos na resolução do problema objeto de estudo.

3.8.1 Base de dados

Para que a aplicação fizesse o que era esperado, era essencial ter uma base de dados para guardar informação pertinente dos clientes. Para a criação da base de dados optou-se por uma classe de bases de dados não relacionais, o NoSQL. Dentro dessa classe existem vários projetos *open source,* o que se pode traduzir em algumas vantagens, nomeadamente menores custos na utilização, maior comunidade para suporte técnico e uma maior compatibilidade com outros sistemas. Depois de uma pesquisa extensiva, concluiu-se que para o nosso projeto o indicado seria utilizar o mongoDB. O mongoDB é uma base de dados orientada a documentos. Para além de nunca termos trabalhado com este tipo de bases de dados, achou-se que poderia ser enriquecedor do ponto de vista da utilização de novas tecnologias. Algumas das vantagens de utilizar mongoDB são a performance e Escalabilidade. Empresas como o Facebook, Google, Amazon e LinkedIn desenvolveram as suas próprias soluções NoSQL para melhorar a performance, quando lidam com quantidades astronómicas de dados[58].

3.8.2 Node.js

O Node.js é uma plataforma construída sob o motor JavaScript do Google Chrome para construir aplicações de rede rápidas e escaláveis. Os servidores web criam *threads*[4] para processar os pedidos dos clientes *(browsers)*, sejam eles pedidos para mostrar páginas html ou outros, e o comportamento normal é ilustrado na Figura 13.

[4] Uma thread é uma sequência de instruções que vão ser executadas num programa

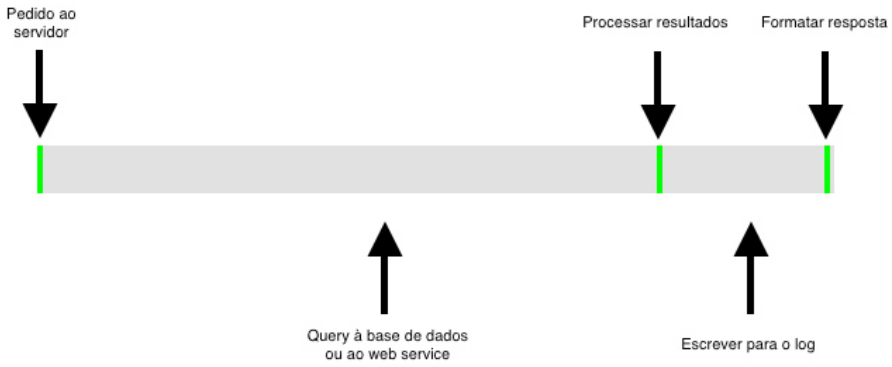


Figura 13 Ciclo de pedido ao servidor

É possível verificar que na maioria do tempo o servidor está à espera de operações I/O, sem fazer nada, enquanto que o tempo em que efetivamente trabalha é para processar resultados e formatar a resposta para posteriormente responder ao pedido. O servidor tem um limite máximo para aceitar pedidos, e este está diretamente relacionado com a memória disponível do servidor. A Figura 14 ilustra os pedidos tratados pelas diferentes *threads* do servidor.

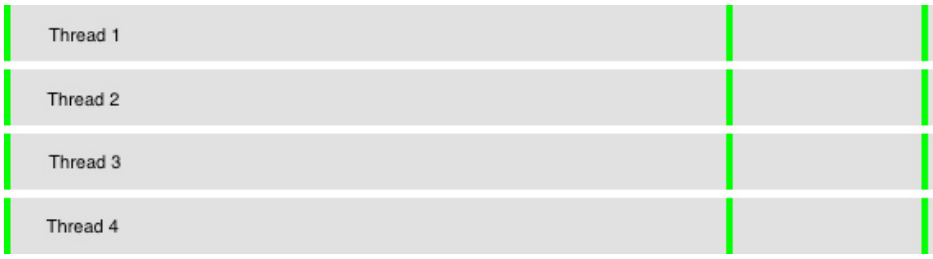


Figura 14 Pedidos em simultâneo

O objetivo do Node.js é acabar os tempos de espera longos, onde uma *thread* fica encarregue de receber os pedidos, como ilustrado na Figura 15.

Figura 15 Objetivo Node.js

O modelo conceptual do Node.js é ilustrado na Figura 16. Existe apenas uma *thread* para tratar os pedidos existentes numa fila de espera, chamada de *event loop.* Se o pedido a ser avaliado necessitar de executar operações I/O, essas operações serão delegadas para a *thread Pool*, libertando a thread *event loop* para aceitar mais pedidos enquanto que as operações I/O vão sendo processadas. Quando as operações I/O terminarem, o resultado vai para a fila de espera para voltar a ser executado pelo servidor.

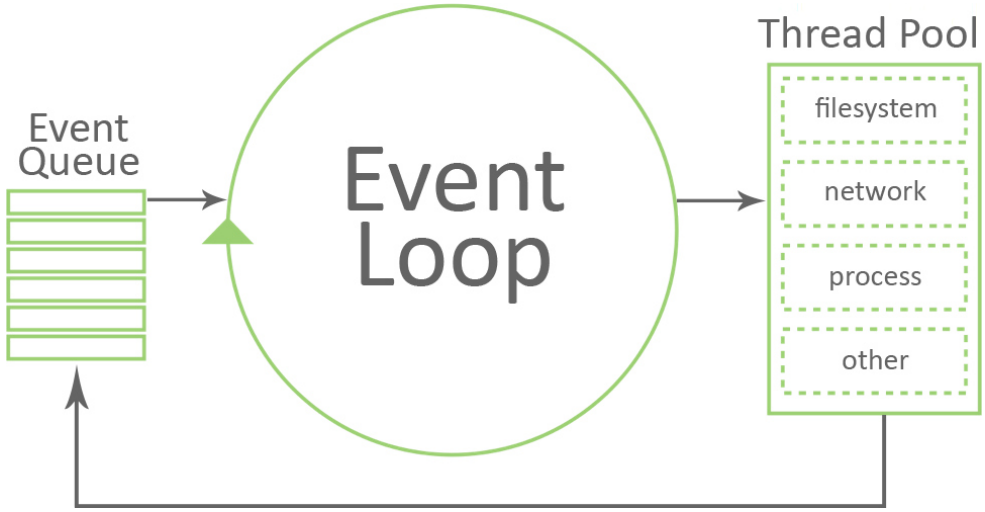


Figura 16 Metodologia Node.js

3.8.3 Desenvolvimento

Criou-se então uma aplicação web utilizando o Express.js, uma *framework* para criar aplicações web em Node.js, capaz de armazenar informação sobre os clientes da empresa de engarrafamento móvel bem como gerar instâncias de clientes. Porque foi desenvolvida em Node.js, a aplicação funciona em qualquer browser moderno que suporte JavaScript, como os browsers dos dispositivos móveis.

Requisitos para a utilização

Primeiro de tudo, é necessário ter a aplicação no computador, bem como ter o Node.js e o mongoDB instalado no computador. Pode fazer-se download da aplicação através de um clone do projeto bitBucket localizado em https://omarioabreu@bitbucket.org/omarioabreu/projecto-mes.git. Para instalar o Node.js e o mongoDB basta aceder aos sites, http://nodejs.org/ e http://mongodb.org/ respetivamente, fazer download das aplicações e instalar como um programa normal. As instruções aqui descritas são apenas para sistemas Mac OS X. No entanto, tanto o Node.js

e como o mongoDB são multiplataforma, ou seja, funcionam em sistemas mais comuns como Windows e Unix.

Instalar *nodemon*, um simples monitor para utilizar durante o desenvolvimento de aplicações Node.js, para que sempre que existam alterações na aplicação, que impliquem o reinício do servidor, o *nodemon* deteta essas alterações e reinicia automaticamente. A Figura 17 ilustra como instalar o modulo *nodemon*.

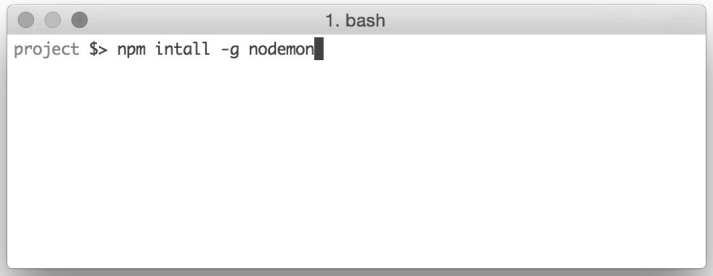


Figura 17 Instalar modulo *nodemon*

Para se iniciar a utilização da aplicação desenvolvida é necessário utilizar a linha de comandos e navegar até à pasta onde se localiza o projeto, como ilustrado na Figura 18 Localização do projeto.

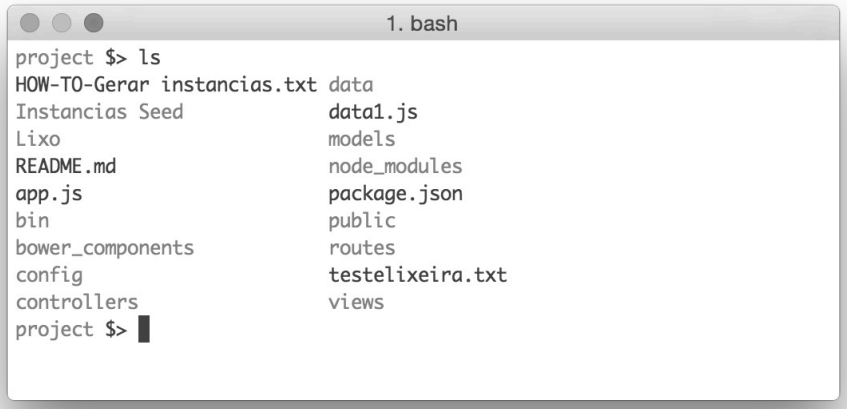


Figura 18 Localização do projeto

Para se colocar o servidor a trabalhar recorre-se ao seguinte comando:

```
npm start
```

Depois de executar o comando, o resultado visível na linha de comandos deverá ser semelhante ao que aparece ilustrado na Figura 19.

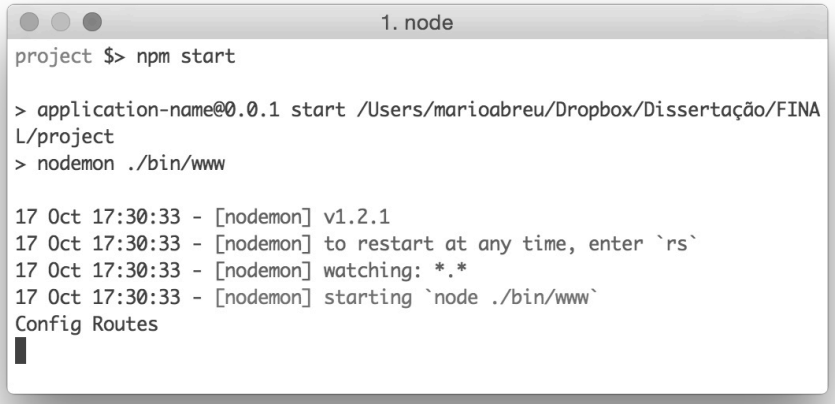


Figura 19 Servidor Resultado depois de iniciar o servidor Node.js

Agora que temos o servidor Node.js à espera de pedidos *http* falta apenas colocar o deamon[5] do mongoDB a trabalhar para conseguirmos aceder à base de dados. Para isso, abrimos uma nova janela na linha de comandos e executamos o comando seguinte:

```
mongod --dbpath data/
```

O resultado deverá ser idêntico ao que é mostrado na Figura 20 Servidor mongoDB a executar

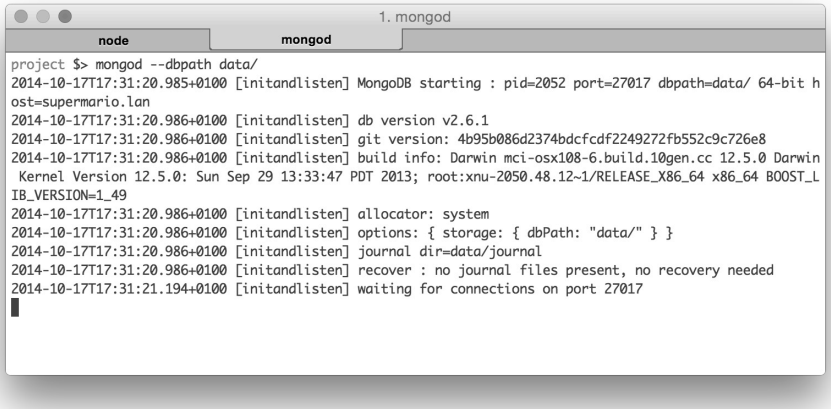


Figura 20 Servidor mongoDB a executar

[5] Deamon - *Disk And Execution MONitor* executa de forma independente em segundo plano.

Agora que temos a aplicação Node.js a funcionar, basta aceder a um browser do computador e aceder a *localhost:3000.* O resultado será a página principal da aplicação, ilustrada na Figura 21.

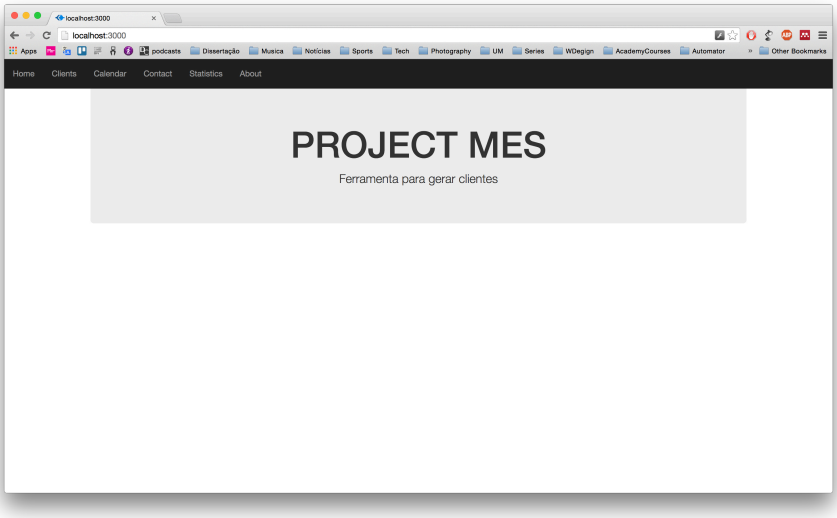


Figura 21 Página inicial da aplicação

É possível inserir clientes, bem como eliminar. Na Figura 22 está ilustrado o formulário para inserir um cliente. Através do formulário é possível fornecer informação como o nome do cliente, nome da empresa, email, número de telefone e localização, através da navegação no mapa.

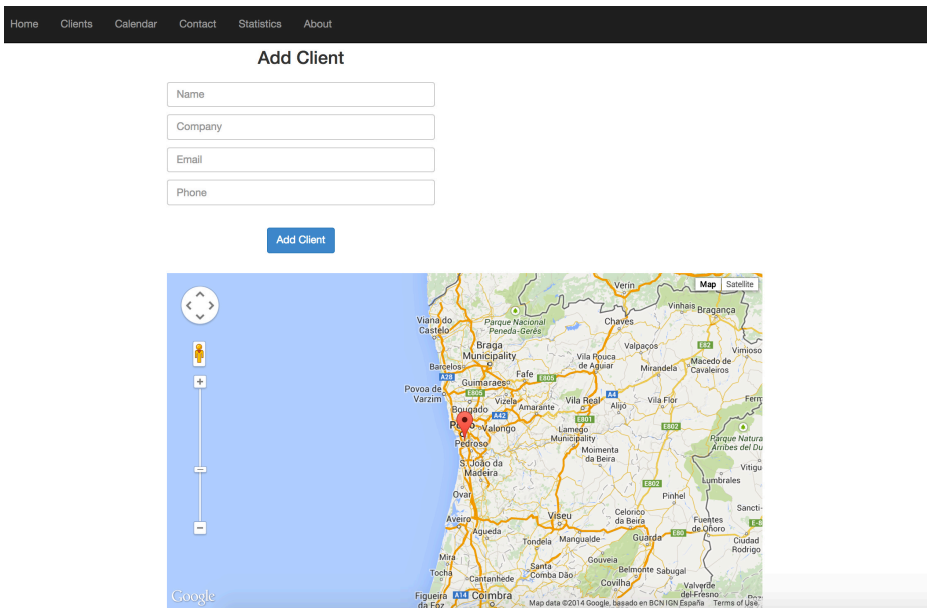


Figura 22 Formulário para inserir cliente

A Figura 23 ilustra que é possível consultar os clientes existentes na base de dados. Depois de selecionar um cliente, do lado direito podemos consultar informação mais detalhada sobre esse cliente, informação que não aparece na listagem inicial.

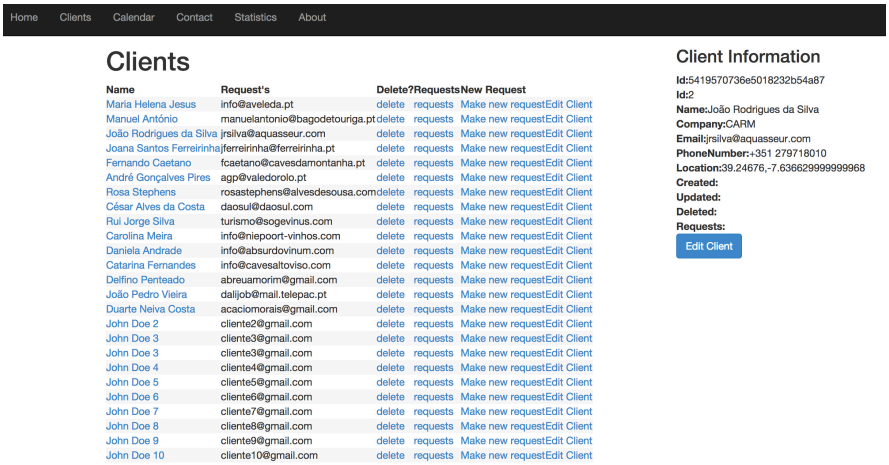


Figura 23 Lista de clientes

Podemos consultar a disposição dos clientes da empresa de engarrafamento pelo território, em que cada marcador encarnado indica a localização de um cliente, como ilustrado na Figura 24.

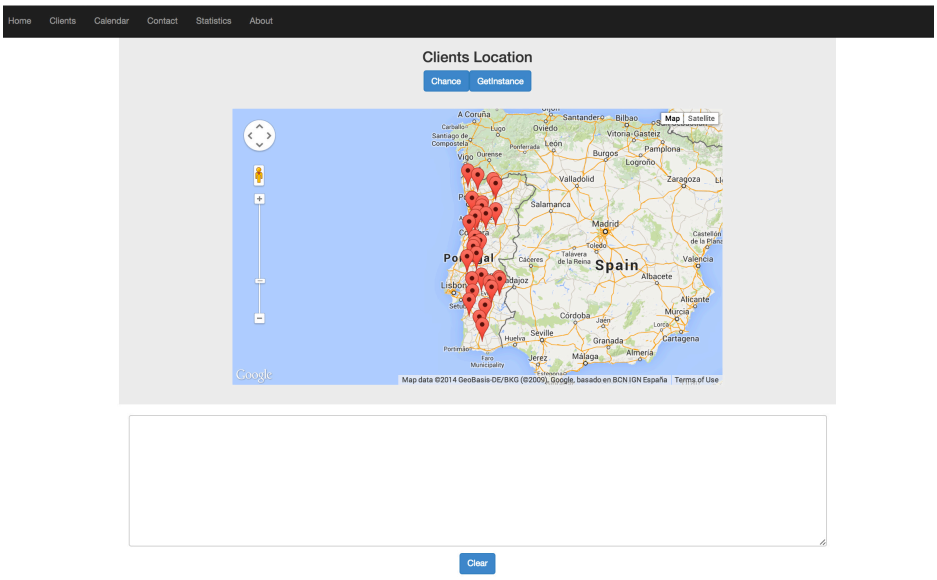


Figura 24 Disposição dos clientes no mapa

3.9 Testes

De seguida explica-se o mecanismo desenvolvido para a realização dos testes das diferentes instâncias. Essencialmente temos que seguir quatro etapas, nomeadamente gerar a localização dos

clientes, criar uma matriz quadrada de distâncias entre todos os clientes, incorporar matriz no modelo a enviar ao NEOS Server com respetivo envio e por fim filtrar e interpretar os resultados devolvidos pelo NEOS Server. Importa referir que podiam-se ter utilizado matrizes de distâncias mais simples, calculadas através da distância euclidiana, por exemplo, mas entendeu-se que seria mais realista e vantajoso adquirir conhecimento ao aprender a utilizar os mapas e trabalhar com distâncias reais.

A nomenclatura das instâncias a utilizar foi definida como sendo *xml-data{A}-{B}.xml* para os modelos a submeter ao NEOS Server, em que *{A}* representa a número de vértices da instância e *{B}* o número de veículos disponíveis. O resultado de NEOS Server é guardado também em ficheiros, seguindo uma nomenclatura semelhante, *data{A}-{B}.txt* em que {A} e {B} significam o mesmo anteriormente.

3.9.1 Etapa 1

Para a realização da primeira etapa, recorre-se à aplicação desenvolvida, que abordamos em 3.8. A aplicação está preparada para gerar instâncias de diversas dimensões. Para gerar uma instância com o mesmo número de clientes existentes na base de dados, basta carregar no botão *chance*, ilustrado na Figura 25. Ao gerar uma instância, estão-se a gerar dados completamente aleatórios, utilizando o chance.js, uma biblioteca JavaScript minimalista para geração de dados aleatórios. Os dados gerados são o nome do cliente, empresa, email, telefone, localização, necessidade, abertura da janela temporal, fecho da janela temporal e a prioridade do cliente. Uma vez geradas as localizações dos clientes, se voltarmos a carregar no botão vão ser geradas novas distâncias. Depois de gerar a informação, queremos guarda-la para criar uma instância. Para isso, carregamos no botão *getInstance* e a caixa de texto localizada abaixo do mapa ficará preenchida com texto, na estrutura de ficheiro JSON, pronta a importar para a base de dados. Teremos que selecionar e copiar todo o texto, como ilustrado na Figura 25. É tido em conta, ao gerar a instância, que o primeiro e o último vértice são exatamente o mesmo. A procura nestes vértices é zero, o número de vinhos também é nulo e a janela temporal é total.

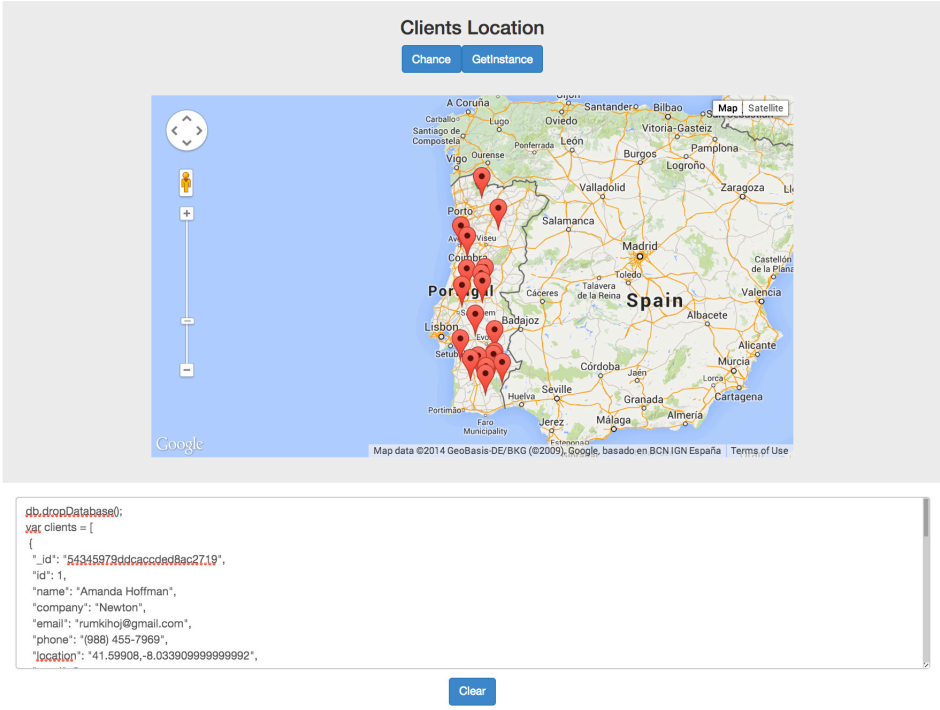


Figura 25 Copiar instância gerada

Depois de copiar o texto, temos que o guardar num ficheiro com extensão *.js*, para indicar que é um ficheiro JavaScript.

Assumindo que a informação foi guardada num ficheiro chamado *seed-i25.js*, e que temos a linha de comandos localizada na diretoria onde está o ficheiro, a Figura 26 mostra o comando a executar para importar esta instância para a base de dados.

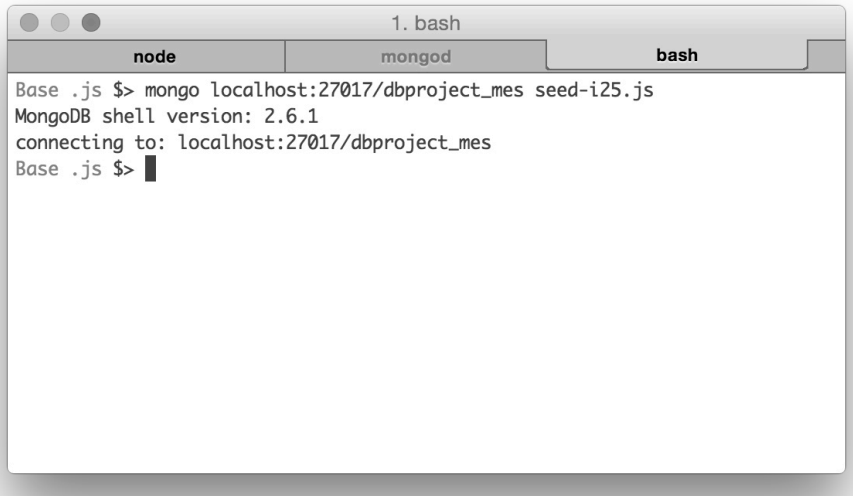


Figura 26 Importar informação de clientes para a base de dados

Neste ponto, os únicos clientes existentes na base de dados são os que acabaram de ser importados, já que os que lá existiam foram substituídos, propositadamente.

3.9.2 Etapa 2

Nesta etapa recorrer-se à API[6] do Google Maps JavaScript para utilizar o serviço de matriz de distâncias. O serviço matriz de distâncias permite-nos saber as distâncias entre várias origens e vários destinos. No nosso caso, as origens e os destinos são coordenadas geográficas, mas poderiam ser endereços. As origens e os destinos são parâmetros obrigatórios para a utilização do serviço. Os parâmetros opcionais são:

- mode – calcula as direções consoante o tipo de transporte *driving*, *walking* ou *bicycling*. Por defeito a opção é *driving*;
- language – língua em que os resultados são devolvidos;
- avoid – introduz restrições à rota tais como, evitar portagens, autoestradas ou ferries;
- units – especificar o sistema de unidades a utilizar. Por defeito é utilizado o sistema métrico;
- departure_time – especificar o tempo de partida, em segundos, depois da meia noite de 1 de janeiro de 1970;

Existem diferentes restrições de utilização deste serviço. Para os utilizadores da API gratuita, as limitações são:

- 100 elementos por query;
- 100 elementos por cada 10 segundos;
- 2500 elementos por períodos de 24 horas;
- máximo de 25 origens e/ou destinos por query;

Já para os utilizadores da Google Maps API for Work, as restrições são:

- 625 elementos por query;
- 1 000 elementos por cada 10 segundos;
- 100 000 elementos por períodos de 24 horas;

[6] Interface de programação de aplicações

Dadas estas restrições impostas pela Google e tendo em conta de que se necessita de testar instâncias com matrizes de dimensão superior a 10 por 10, foi necessário utilizar o serviço disponibilizado pela Google por partes e de forma organizada.

A obtenção da matriz terá que ser feita parcialmente, em mais ou menos iterações, dependendo do tamanho da matriz. Se a matriz for até um máximo de 25 destinos, então conseguimos receber uma linha da matriz de cada vez. Se a matriz for maior, teremos que receber a linha parcialmente. Admitindo que a Figura 27 é uma matriz completa 100 por 100, pensou-se em criar um primeiro botão para obter a área representada a azul, um segundo para obter a vermelha, um terceiro para obter a área a amarelo e por fim, o ultimo botão par obter a área a verde.

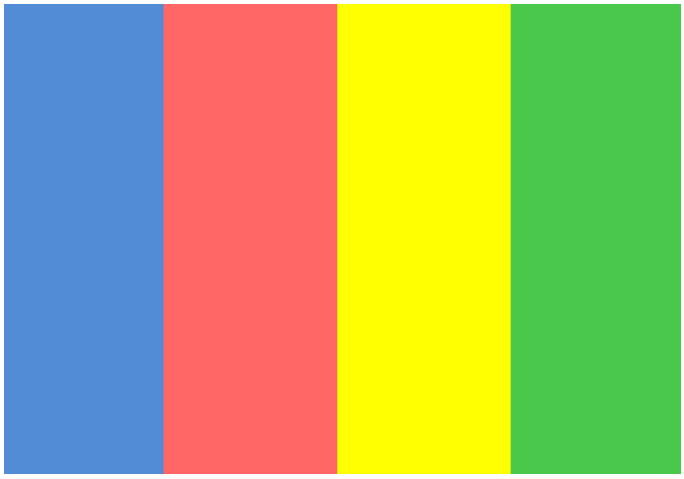


Figura 27 Matriz da distâncias abstrata

A Figura 28 mostra a interface para obter a informação dos clientes e a matriz de distâncias. Consegue ver-se uma etiqueta que indica o índice da linha da matriz a obter, juntamente com dois botões para incrementar e decrementara o índice. Existem também quatro botões para se conseguir obter matrizes até um máximo de 100 origens e 100 destinos. O primeiro botão, *get Distances 25* é destina-se à obtenção dos valores dos primeiros 25 índices de colunas da matriz, indicado a azul na Figura 27. O segundo botão para os índices maiores que 25 e menores do que 50 para obter os valores da área vermelha, o terceiro para os índices entre 50 e 75, representado pela cor amarela e por fim, o último para índices maiores que 75 representado pela área verde.

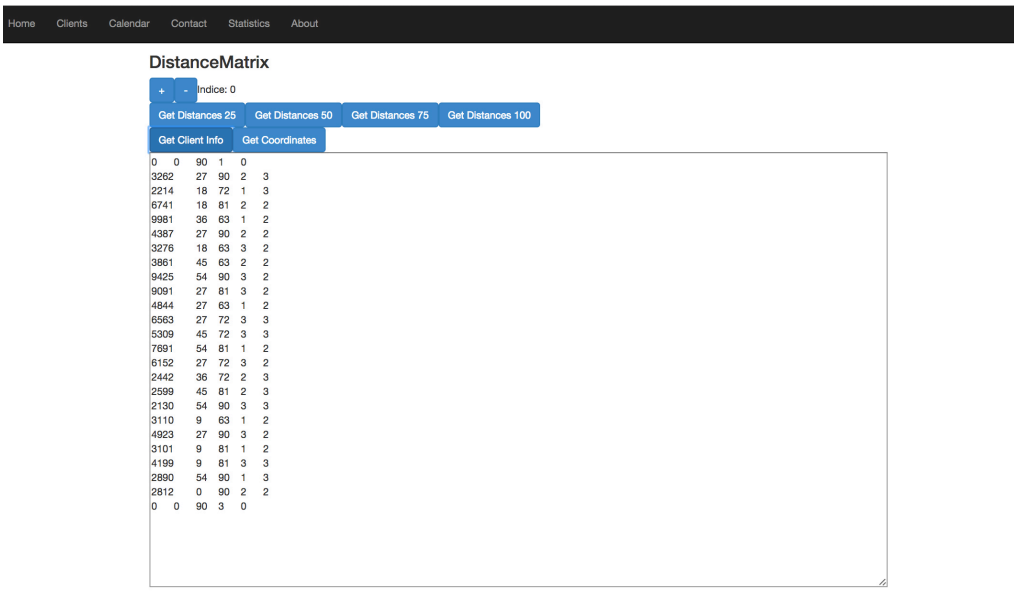


Figura 28 Interface para obter informação dos clientes e matriz de distâncias

Para se obter a matriz de distâncias para a instância 25-3, utiliza-se apenas o botão *get Distances 25*, já que a matriz é composta por 25 origens e 25 destinos. Inicialmente o índice é zero e podemos, portanto, iniciar o processo da obtenção da matriz ordenadamente pressionando o botão *get Distances 25*, como ilustrado na Figura 29, obtendo-se as distâncias da origem localizada no índice 0 para todos os 25 destinos.

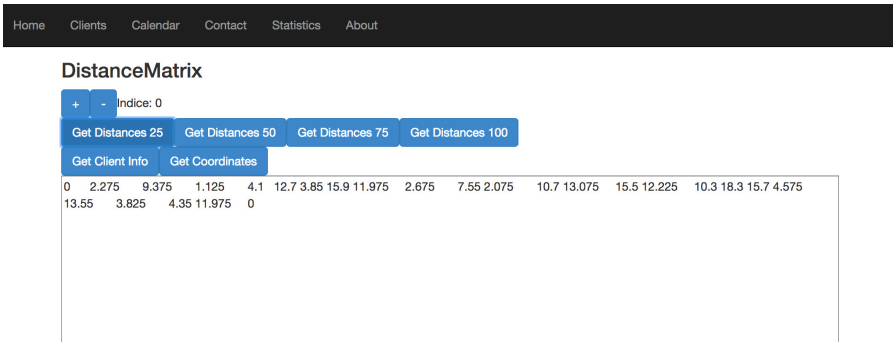


Figura 29 Início do processo de obtenção da matriz de distâncias

Depois de se receber as distâncias, incrementa-se o índice das origens para 1 através do botão com o carácter + (mais) e volta-se a pressionar o botão *get Distances 25* para obter as distâncias desde a origem índice 1 para todos os 25 destinos. Repetir este processo até chegar ao último índice das origens. O botão – (menos) está disponível para voltar a índices anteriores se, por alguma razão, não se conseguir obter as distâncias. A matriz de distâncias obtida já está convertida em horas, assumindo que a velocidade média do veículo é de 40km/hora.

Existem mais dois botões para exportar a informação dos clientes, nomeadamente as coordenadas dos clientes, necessidade de cada cliente expressa em litros, a abertura e fecho da janela temporal, prioridade do cliente e, por fim, o número de vinhos a engarrafar. Obter a informação numa caixa de texto, como ilustrado na Figura 28, é prático para poder rapidamente ser importado para o modelo a enviar ao NEOS Server e também para incluir no ficheiro auxiliar de Excel.

3.9.3 Etapa 3

Depois de obtida a matriz de distâncias entre todas as coordenadas geográficas, clientes e ponto de partida, a etapa seguinte consiste em a introduzir no modelo a enviar ao NEOS Server. Existem duas formas de enviar trabalhos para o *Gurobi NEOS Server*, através da interface web ou através de uma chamada de procedimento remoto *(Remote Procedure Call, rpc).* Uma *rpc* consiste em enviar um ficheiro por linha de comandos, escrito em *xml*, contendo o modelo, dados e comandos a executar pelo servidor. Na página web do NEOS Server são disponibilizados vários aplicativos (cliente), escritos em diferentes linguagens de programação, para realizar as *rpc's*. Ao longo desta dissertação, a configuração mais utilizada foi através das chamadas de procedimento remoto. No entanto também foi utilizada a interface *web*, numa fase mais embrionária do estudo realizado.

Estrutura das instâncias

Para que o ficheiro seja reconhecido no servidor, este tem que respeitar uma estrutura bem definida. A estrutura do ficheiro tem que ser a seguinte:

```
<document>
    <category>milp</category>
    <solver>Gurobi</solver>
    <inputType>AMPL</inputType>
    <email>projectmes2014@gmail.com</email>
    <model><![CDATA[ - colocar o modelo AMPL AQUI - ]]></model>
    <data><![CDATA[ - colocar os dados aqui -]]></data>
```

<***commands***><*![CDATA[- colocar aqui os comandos necessários -]]*></***commands***>

<***comments***><*![CDATA[]]*></***comments***>

</***document***>

O ficheiro é composto por diversas *tags*, com diferentes significados:

- <***document***> - que indica o início do documento.
- <***category***> - indica o tipo de ficheiro, no nosso caso é *milp.*
- <***solver***> - indica o solver que vamos utilizar para resolver o modelo, no nosso caso *Gurobi*.
- <***inputType***> - indica a linguagem de programação em que o ficheiro está escrito, no nosso caso é em *AMPL*.
- <***email***> - opcional, indica o email para onde se quer que envie uma cópia dos resultados.
- <***model***> - indica o modelo a ser executado.
- <***data***> - informação relevante para o modelo.
- <***commands***> - comandos importantes para a execução do modelo, nomeadamente o ***solve***;
- <***comments***> - Opcional, pode-se colocar os comentários referentes ao modelo.

A instância xml-data25-3.xml é disponibilizada no anexo 1.

Remote procedure calls

Para fazer as *rpc*, primeiro de tudo é preciso descarregar o aplicativo localizado em http://www.neos-server.org/neos/Installation.html. No caso de estudo, utilizou-se o aplicativo escrito em Python. Uma vez descarregado, podemos executa-lo através da linha de comandos. Se o aplicativo e o ficheiro *xml* estiverem localizados na mesma diretoria, e assumindo que o ficheiro *xml* tem o nome *xml-dat25-3.xml*, a rpc é feita como mostra na Figura 30.

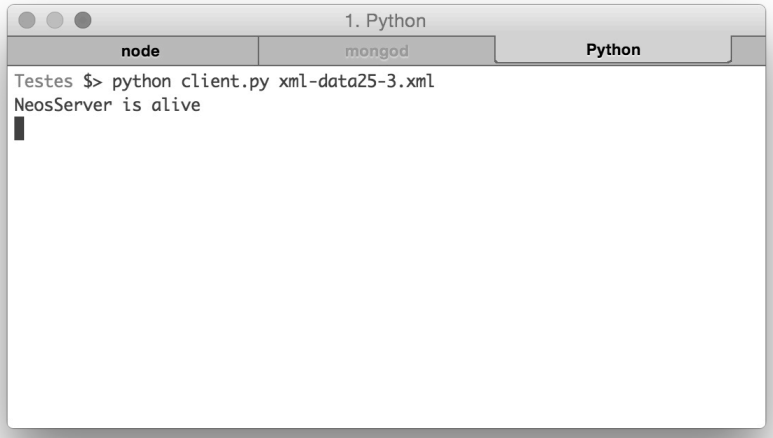


Figura 30 Remote procedure call

No caso de se ter configurado previamente a tag *email* no ficheiro *xml*, quando o servidor responder, mesmo que o computador a partir de onde fez a *rpc* já não possua ligação à Internet ou, por alguma razão, tenha fechado a linha de comandos, o resultado da *rpc* é enviado para o email. Na linha de comandos podemos direcionar o resultado diretamente para um ficheiro de texto, como é ilustrado na Figura 31.

Figura 31 Remote procedure call com redirecionamento do output

O resultado do comando Python cliente.py xml-data25-3.xml é guardado no ficheiro data25-3.txt. O ficheiro data25-3.txt será reescrito com o resultado se já existir ou criado, caso contrário.

3.9.4 Etapa 4

Nesta fase será feita a interpretação dos resultados devolvidos pelo NEOS Server, na Etapa 3. É importante perceber, nomeadamente, a ordem pela qual os vértices foram visitados, e por qual veículo. Conhecendo o padrão bem definido do ficheiro devolvido pelo NEOS Server é possível, através de expressões regulares[7] e aplicativos da linha de comandos, filtrar o resultado de forma a facilitar a leitura e interpretação. Depois de filtrar a informação relevante, vamos catalogar essa informação num ficheiro auxiliar em Excel, onde pensamos que seja vantajoso para uma melhor organização e visão global das soluções.

É importante referir que a dimensão dos ficheiros devolvidos pelo NEOS Server pode, facilmente atingir milhares de linhas e que se não fossem desenvolvidos estes mecanismos e ferramenta ilustradas nas figuras 32, 33 e 34 a interpretação manual e destes ficheiros seria um processo muito penoso e demorado. Acredita-se assim que a ferramenta é útil e poderá ser reutilizada futuramente, com o objetivo de sistematizar a interpretação destes ficheiros e ainda que, graças a ela, foram poupadas horas de trabalho no caso de estudo.

No início, é importante saber o instante de tempo de início de trabalho de cada vértice, então criou-se um comando que, tendo como input o resultado do NEOS Server, devolve uma lista com estes instantes de tempo, como ilustrado na Figura 32.

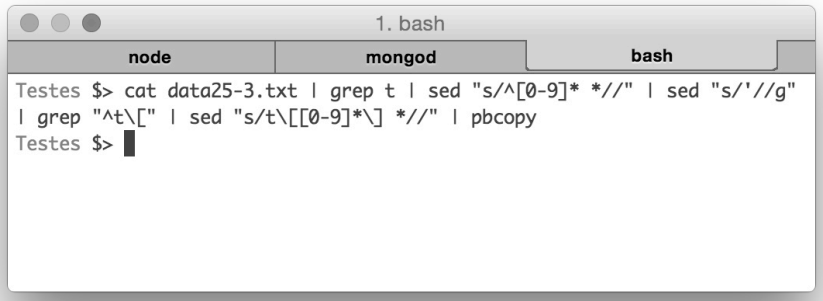


Figura 32 Extração do instante de tempo de início de trabalho de todos os vértices

O resultado da execução do comando da Figura 32 é guardado diretamente para o clipboard[8], pronto a colar no ficheiro auxiliar, em Excel.

[7] Regras para representar padrões cadeias de caracteres.
[8] Área de transferência de dados, normalmente utilizada em operações de copiar, cortar e colar.

Depois houve a necessidade de saber quais os vértices visitados, e quais os veículos que os visitavam. Para tal, desenvolveu-se um comando para extrair do resultado do NEOS Server todos os pares de vértices, caminhos percorridos entre os vértices, ilustrado na Figura 33.

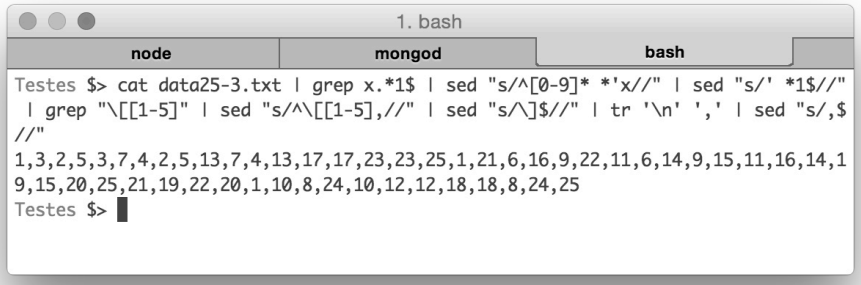
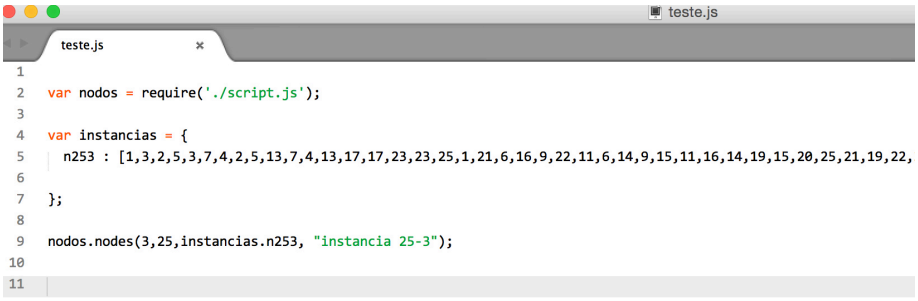


Figura 33 Caminhos percorridos pela solução do NEOS Server

A leitura do resultado da Figura 33 é feita aos pares, ou seja, a solução inicia-se no vértice 1 e daí viaja para o vértice 3. De seguida temos que encontrar o próximo par de vértices que inicia no vértice 3, ou seja, o quinto elemento da lista, lendo-se de forma semelhante. Do vértice 3 viaja para o vértice 7 e assim sucessivamente até encontrar o vértice final, que no caso desta instância é o vértice 25. O número de vezes que o vértice 1 aparece significa o número de veículos utilizados na instância. Na Figura 33 o vértice 1 aparece 3 vezes, o que significa que foram utilizados 3 veículos.

Percebendo este mecanismo, desenvolveu-se uma ferramenta em JavaScript que, tendo esta lista como *input* devolve uma lista com a ordem dos vértices visitados. A ferramenta consiste num ficheiro *JavaScript,* disponibilizado em anexo, que será invocado por um outro ficheiro e será aplicado a uma ou várias listas de pares de vértices. Cada lista corresponde unicamente a uma instância. Assim, na Figura 34 é ilustrado a preparação do ficheiro para a utilização da ferramenta, carregando o script na linha 2. Na linha 4 é definido um objeto, composto por pares chave-valor, para guardar os pares de vértices das diferentes instâncias. Na linha 5 define-se a o par chave-valor para a instância 25-3, em que a chave é *n253* e o valor é a lista devolvida pelo comando executado na Figura 33. Por fim, na linha 9 executa-se o script. Para a execução correta do script são necessários quatro parâmetros. Os parâmetros são o número de veículos da instância, o número de vértices a visitar, a lista de pares de vértices e, por fim, uma etiqueta de texto para identificação dos resultados. No exemplo da Figura 34 usam-se 3 veículos, 25 nodos, a lista de pares *instancias.n253* e uma etiqueta de texto "instância 25-3".



.

Figura 34 Preparação para utilização da ferramenta

O ficheiro tem que ser guardado com extensão *.js* para indicar que é um ficheiro *JavaScript*. No nosso caso, o ficheiro chama-se teste.js e Figura 35 ilustra a sua execução.

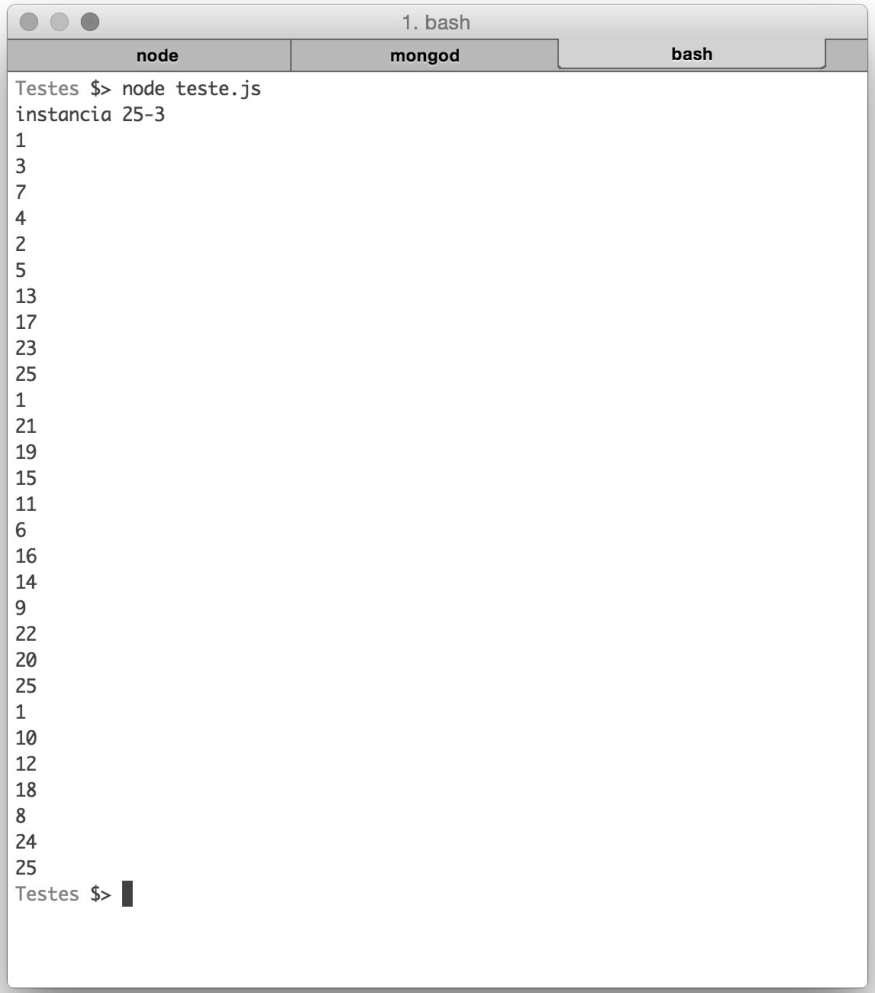


Figura 35 Resultado da ferramenta desenvolvida

Com este resultado, é bastante percetível identificar as rotas dos diferentes veículos, estão ilustradas na Figura 36.

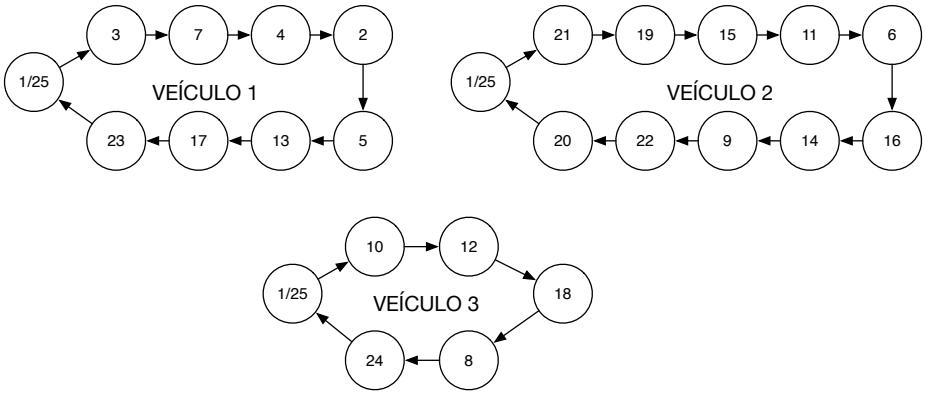


Figura 36 Rotas efetuadas pelos diferentes veículos

É necessário repetir todas as etapas para cada instância gerada, só assim será possível a obtenção dos resultados. Apenas a utilização da ferramenta ilustrada na Figura 34 pode ser executada apenas uma vez, desde que o objeto *instancias* contenha os pares chave-valor com um nome alusivo à instância e os resultados do comando ilustrado na Figura 33, para a chave e valor, respetivamente.

A informação extraída irá ser importada para um ficheiro auxiliar em Excel, onde se possa organizar a informação e criar outro tipo de estruturas de dados para tirar conclusões. Na secção seguinte será apresentado o ficheiro auxiliar utilizado.

3.10Ficheiro auxiliar Excel

Decidiu-se que seria útil agregar a informação das diferentes instâncias num só ficheiro para ser mais fácil efetuar comparações e tirar conclusões. A Figura 37 ilustra a informação relativa aos clientes.

#	1	2	3	4	5	6	7	8	9	10
1	41.45642	-7.80415	0	0	90	1	0	0.00	0	90
2	41.36863	-8.5525	3262	27	90	2	3	2.73	6524	63
3	39.41634	-7.93274	2214	18	72	1	3	2.17	2214	54
4	41.63718	-7.74616	6741	18	81	2	2	4.26	13482	63
5	40.51428	-8.10048	9981	36	63	1	2	5.98	9981	27
6	38.2037	-8.6459	4387	27	90	2	2	3.00	8774	63
7	40.58022	-8.01053	3276	18	63	3	2	2.41	9828	45
8	37.62492	-7.48272	3861	45	63	2	2	2.72	7722	18
9	38.6999	-7.56948	9425	54	90	3	2	5.69	28275	36
10	41.37678	-8.6605	9091	27	81	3	2	5.51	27273	54
11	39.69254	-8.62616	4844	27	63	1	2	3.24	4844	36
12	41.34663	-8.39185	6563	27	72	3	3	4.49	19689	45
13	38.76151	-8.37393	5309	45	72	3	3	3.82	15927	27
14	38.61088	-7.9408	7691	54	81	1	2	4.76	7691	27
15	37.45694	-7.88845	6152	27	72	3	2	3.94	18456	45
16	38.40889	-8.40719	2442	36	72	2	3	2.29	4884	36
17	39.04432	-7.82454	2599	45	81	2	3	2.38	5198	36
18	37.34364	-7.6132	2130	54	90	3	3	2.13	6390	36
19	37.32083	-8.28855	3110	9	63	1	2	2.32	3110	54
20	40.63013	-8.63038	4923	27	90	3	2	3.29	14769	63
21	37.86148	-8.41385	3101	9	81	1	2	2.31	3101	72
22	40.64239	-7.9464	4199	9	81	3	3	3.23	12597	72
23	40.5107	-8.19128	2890	54	90	1	3	2.53	2890	36
24	38.67971	-7.55922	2812	0	90	2	2	2.16	5624	90
25	41.45642	-7.80415	0	0	90	3	0	0.00	0	90

Figura 37 Informação relativa a clientes

A coluna 1 e 2 indicam a latitude e longitude, respetivamente. A coluna 3 indica a necessidade do cliente, expressa em litros. A coluna 4 e 5 indicam a abertura e fecho da janela temporal, respetivamente. A coluna 6 indica a prioridade do cliente, como abordado anteriormente na Tabela 1. Na coluna 8 indica-se o tempo necessário para servir o cliente, em horas, com base na quantidade de litros a processar indicado na coluna 3 e na quantidade de vinhos a processar referido na coluna 7 como indicado em (34). A coluna 9 indica o prémio a ser coletado pela visita ao cliente. E a coluna 10 indica a disponibilidade do cliente, em horas.

Encontra-se também neste documento a matriz de distâncias entre as localizações dos clientes a visitar, obtida no Etapa 2 da fase de testes. A Figura 38 ilustra a matriz de distâncias para a instância data25-3.txt

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0.00	2.28	9.38	1.13	4.10	12.70	3.85	15.90	11.98	2.68	7.55	2.08	10.70	13.08	15.50	12.23	10.30	18.30	15.70	4.58	13.55	3.83	4.35	11.98	0.00
2	2.28	0.00	7.53	2.75	4.33	10.85	4.08	14.08	10.15	0.38	5.70	0.58	8.85	11.25	13.65	10.40	8.48	16.45	13.85	2.73	11.70	4.05	4.18	10.15	2.28
3	9.35	7.50	0.00	9.83	5.55	5.00	5.83	6.73	3.38	7.55	2.53	7.55	2.83	2.93	7.80	3.55	1.40	7.88	8.00	5.10	5.85	6.25	5.43	3.38	9.35
4	1.13	2.75	9.85	0.00	4.80	13.18	4.55	16.40	12.70	3.15	8.03	2.55	11.20	13.58	15.98	12.73	10.80	18.80	16.18	5.05	14.03	4.53	5.08	12.70	1.13
5	4.13	4.35	5.58	4.65	0.00	8.88	0.40	12.10	8.18	4.38	3.75	4.38	6.90	9.28	11.70	8.43	6.50	14.50	11.88	2.40	9.75	0.60	0.28	8.20	4.13
6	12.58	10.73	5.00	13.05	8.78	0.00	9.05	3.93	4.80	10.78	5.43	10.75	2.48	3.45	3.53	1.05	4.25	6.33	3.73	8.30	1.43	9.25	8.65	4.30	12.58
7	3.90	4.10	6.40	4.40	0.40	9.18	0.00	12.38	8.83	4.15	4.03	4.13	7.18	9.55	11.98	8.73	6.78	14.78	12.18	2.18	10.03	0.35	0.65	8.85	3.90
8	15.75	13.93	6.73	16.25	11.95	3.93	12.23	0.00	4.18	13.98	8.63	13.95	4.98	3.85	1.40	4.60	5.50	2.03	3.23	11.50	2.73	12.45	11.85	4.10	15.75
9	12.00	10.13	3.40	12.50	8.15	4.80	8.83	4.20	0.00	10.18	5.85	10.15	2.78	1.18	4.75	3.25	1.45	5.35	5.28	7.70	4.10	8.68	8.05	0.08	12.00
10	2.68	0.38	7.40	3.15	4.20	10.73	3.95	13.93	10.03	0.00	5.58	1.15	8.73	11.10	13.53	10.28	8.35	16.33	13.73	2.60	11.58	3.93	4.05	10.03	2.68
11	7.53	5.70	2.53	8.03	3.73	5.55	4.00	8.75	5.85	5.75	0.00	5.73	3.55	5.93	8.35	5.10	3.48	11.15	8.55	3.28	6.40	4.23	3.63	5.85	7.53
12	2.08	0.58	7.55	2.55	4.35	10.88	4.10	14.08	10.18	1.13	5.73	0.00	8.88	11.28	13.68	10.43	8.50	16.48	13.88	2.75	11.73	4.08	4.20	10.18	2.08
13	10.68	8.83	2.83	11.15	6.88	2.48	7.15	4.98	2.78	8.88	3.53	8.85	0.00	1.43	5.28	1.38	2.23	6.13	5.48	6.40	3.33	7.35	6.75	2.30	10.68
14	12.93	11.08	2.95	13.40	9.13	3.45	9.40	3.85	1.20	11.13	5.78	11.13	1.43	0.00	4.40	1.90	1.73	5.03	4.93	8.68	4.30	9.60	9.00	1.10	12.93
15	15.38	13.53	7.80	15.85	11.58	3.53	11.85	1.40	4.75	13.58	8.23	13.55	5.28	4.40	0.00	4.20	7.05	1.38	2.03	11.10	2.33	12.05	11.45	4.65	15.38
16	12.10	10.28	3.55	12.58	8.30	1.05	8.58	4.60	3.25	10.30	4.98	10.30	1.38	1.78	4.20	0.00	2.63	7.00	4.38	7.85	2.25	8.80	8.20	2.65	12.10
17	10.28	8.45	1.40	10.78	6.48	4.25	6.75	5.50	1.45	8.50	3.48	8.48	2.25	1.73	7.05	2.63	0.00	6.68	7.25	6.03	5.10	6.98	6.38	1.48	10.28
18	18.15	16.33	7.90	18.65	14.35	6.33	14.63	2.03	5.35	16.35	11.03	16.35	8.08	5.03	1.38	7.00	6.68	0.00	3.13	13.90	5.18	14.85	14.25	5.28	18.15
19	15.58	13.73	7.98	16.05	11.75	3.73	12.03	3.23	5.25	13.78	8.43	13.75	5.48	4.93	2.03	4.40	7.25	3.15	0.00	11.30	2.20	12.25	11.65	5.18	15.58
20	4.55	2.70	5.20	5.03	2.35	8.50	2.10	11.73	7.80	2.75	3.38	2.73	6.53	8.90	11.33	8.05	6.13	14.13	11.50	0.00	9.35	2.08	2.20	7.83	4.55
21	13.43	11.58	5.85	13.90	9.60	1.43	9.90	2.75	5.65	11.63	6.28	11.60	3.33	4.30	2.33	2.25	5.10	3.90	2.20	9.15	0.00	10.10	9.50	4.00	13.43
22	3.83	4.03	6.23	4.35	0.60	9.38	0.35	12.58	8.65	4.08	4.23	4.08	7.38	9.75	12.18	8.93	7.00	14.98	12.38	2.10	10.23	0.00	0.85	8.65	3.83
23	4.38	4.13	5.45	4.90	0.28	8.78	0.65	12.00	8.08	4.18	3.65	4.15	6.80	9.18	11.58	8.33	6.40	14.40	11.78	2.20	9.63	0.88	0.00	8.10	4.38
24	12.00	10.13	3.40	12.53	8.18	4.38	8.83	4.13	0.08	10.18	5.85	10.15	2.38	1.10	4.68	2.85	1.48	5.28	5.20	7.70	4.00	8.70	8.05	0.00	12.00
25	0.00	2.28	9.38	1.13	4.10	12.70	3.85	15.90	11.98	2.68	7.55	2.08	10.70	13.08	15.50	12.23	10.30	18.30	15.70	4.58	13.55	3.83	4.35	11.98	0.00

Figura 38 Matriz de distâncias da instância 25-3

Depois de adaptar a matriz de distâncias ao modelo, enviar os dados ao NEOS Server, receber os resultados e filtrar a informação pertinente do ficheiro de resultados, incorporou-se a informação para o ficheiro. Uma das informações que interessa importar é o instante, em horas, em que o vértice é visitado, como ilustrado na Figura 39.

	Instante de início de
1	0
2	39.8082
3	18
4	32.803
5	46.8629
6	47.9595
7	25.9958
8	60.2808
9	62.0385
10	27
11	39.1661
12	33.6585
13	68.1785
14	56.0767
15	27
16	52.0093
17	74.3425
18	54.6238
19	22.6563
20	82.1644
21	13.55
22	76.8349
23	83.0937
24	75.8403
25	90

Figura 39 Instante de tempo de início de trabalho

Analisando a tabela ilustrada na Figura 39 torna-se evidente que esta instância utiliza mais do que um veículo, uma vez que os vértices 10 e 15 iniciam o trabalho no mesmo instante de tempo.

Criou-se também uma tabela, para cada instância, com a solução alcançada através do NEOS Server. Através da interpretação do ficheiro de resposta do NEOS Server foi possível criar a tabela com mais informação, como ilustra a Figura 40.

1	2	3	4	5	6	7	8	9	10	11
1	1	0	90	0.00	0.00	0.00	0.00	0	0.00	90
1	3	18	72	18.00	20.17	2.17	9.38	2214	8.63	54
1	7	18	63	26.00	28.41	2.41	5.83	9828	20.18	45
1	4	18	81	32.80	37.06	4.26	4.40	13482	28.40	63
1	2	27	90	39.81	42.54	2.73	2.75	6524	37.06	63
1	5	36	63	46.86	52.84	5.98	4.33	9981	42.54	27
1	13	45	72	68.18	72.00	3.82	6.90	15927	61.28	27
1	17	45	81	74.34	76.72	2.38	2.23	5198	72.12	36
1	23	54	90	83.09	85.62	2.53	6.38	2890	76.72	36
1	25	0	90	90.00	90.00	0.00	4.38	0	85.63	90
2	1	0	90	0.00	0.00	0.00	0.00	0	0.00	90
2	21	9	81	13.55	15.86	2.31	13.55	3101	0.00	72
2	19	9	63	22.66	24.98	2.32	2.20	3110	20.46	54
2	15	27	72	27.00	30.94	3.94	2.03	18456	24.98	45
2	11	27	63	39.17	42.41	3.24	8.23	4844	30.95	36
2	6	27	90	47.96	50.96	3.00	5.55	8774	42.41	63
2	16	36	72	52.01	54.30	2.29	1.05	4884	50.96	36
2	14	54	81	56.08	60.84	4.76	1.78	7691	54.31	27
2	9	54	90	62.04	67.73	5.69	1.20	28275	60.84	36
2	22	9	81	76.83	80.06	3.23	8.68	12597	68.16	72
2	20	27	90	82.16	85.45	3.29	2.10	14769	80.06	63
2	25	0	90	90.00	90.00	0.00	4.55	0	85.45	90
3	1	0	90	0.00	0.00	0.00	0.00	0	0.00	90
3	10	27	81	27.00	32.51	5.51	2.68	27273	24.33	54
3	12	27	72	33.66	38.15	4.49	1.15	19689	32.51	45
3	18	54	90	54.62	56.75	2.13	16.48	6390	38.15	36
3	8	45	63	60.28	63.00	2.72	2.03	7722	58.26	18
3	24	0	90	75.84	78.00	2.16	4.10	5624	71.74	90
3	25	0	90	90.00	90.00	0.00	12.00	0	78.00	90
Nodos Visitados				25						

Figura 40 Solução para instância 25-3

De maneira a compactar a tabela ilustrada na Figura 40, alteraram-se nomes dos cabeçalhos para números. Na coluna 1 indica-se qual o veículo que vai operar na rota. Utiliza-se repetidamente o número identificador do veículo para mais tarde simplificar o calculo dos valores totais por veículo, como por exemplo o prémio total recolhido. Na coluna 2 indica-se o vértice que foi visitado. Nas colunas 3 e 4 indica-se, em horas, a abertura e fecho de janela temporal, respetivamente. Os valores apresentados nas colunas 5 e 6 são o instante de início e fim de trabalho, em horas, respectivamente. Já a coluna 7 indica o tempo necessário para realizar o trabalho. A coluna 8 indica o tempo de viagem entre o vértice anterior e o vértice atual. Já a colunas 9 indica o prémio recolhido por visitar o vértice. Na coluna 10 indica-se o instante de tempo, em horas, em que o veículo iniciou a viagem para chegar ao vértice. Por fim, na coluna 11 indica-se a disponibilidade total do cliente, ou seja, o tamanho da janela temporal, em horas. Existe ainda um indicador que notifica quantos nodos foram visitados pelo conjunto de veículos utilizados. Na instância 25-3 foram visitados todos os vértices.

O ficheiro Excel têm vários separadores, um por cada instância. Por cada instância, efetuaram-se testes com diferentes números de veículos existindo uma tabela, como a indicada na Figura 40, para cada teste com diferente número de veículos.

Com base na Figura 40 é possível aferir mais informação relativamente ao problema, de um ponto de vista mais global. Entende-se por utilização do veículo todo o tempo em que este se encontra em serviço num cliente ou em viagem. A Figura 41 mostra o prémio recolhido, a taxa de utilização agregado por veículo, taxa de utilização em viagem e taxa de utilização em serviço efetivo. É Indicado ainda o total de prémio recolhido, bem como a taxa de utilização média dos veículos.

Veículo	Prémio	Tx Utilização	Viagem	Serviço
1	66044	80.92%	52%	29%
2	106501	94.41%	57%	38%
3	66698	61.59%	43%	19%
Total	239243	78.97%	50%	29%

Figura 41 Prémio total e taxas de utilização

Os valores apresentados na tabela ilustrada pela Figura 41 são referentes à instância 25-3, e é possível verificar que os veículos 1 e 2 são os que têm as taxas de utilização maiores, com o veiculo 2 a estar muito perto dos 100%. Já o veículo 3 teve apenas 61.59% de utilização. Embora o veículo 1 tenha tido maior taxa de utilização, recolheu menos prémios que o veiculo 3. O veículo 2 foi o veículo que mais prémios recolheu, com um total de 106501. O prémio total recolhido é de 239243, já a utilização média dos veículos é de 78,97%.

Com estes indicadores poderemos ter uma melhor perceção dos resultados. Queremos ver os valores do prémio recolhido o mais alto possível, bem como a taxa de utilização dos veículos o mais próximo dos 100% possível.

Criou-se ainda um diagrama de *Gantt* por cada rota efetuada pelos diferentes veículos. O diagrama de Gantt é utilizado para ilustrar o progresso das diferentes etapas. Neste diagrama encontra-se os registos do tempo de viagem para chegar ao vértice e o tempo de trabalho nesse mesmo cliente, representados a vermelho e verde, respetivamente. Estes diagramas são gráficos criados em Excel, com o tipo de gráfico barras empilhadas. Para a criação do diagrama recorreu-se informação apresentada na Figura 40, nomeadamente a informação das colunas 7,8 e 10, tempo necessário, tempo de viagem e início de viagem, respetivamente.

A informação sobre o início de viagem não aparece indicada no gráfico, colocada propositadamente em transparente, caso contrário não se poderia chamar um diagrama de Gantt.

Uma vez que os gráficos são criados em Excel, qualquer alteração feita na tabela ilustrada na Figura 40 o diagrama de *Gantt* também sofre alterações, já que este é dinâmico. Na Figura 42 ilustra-se o diagrama de *Gantt* para a instância 1-3.

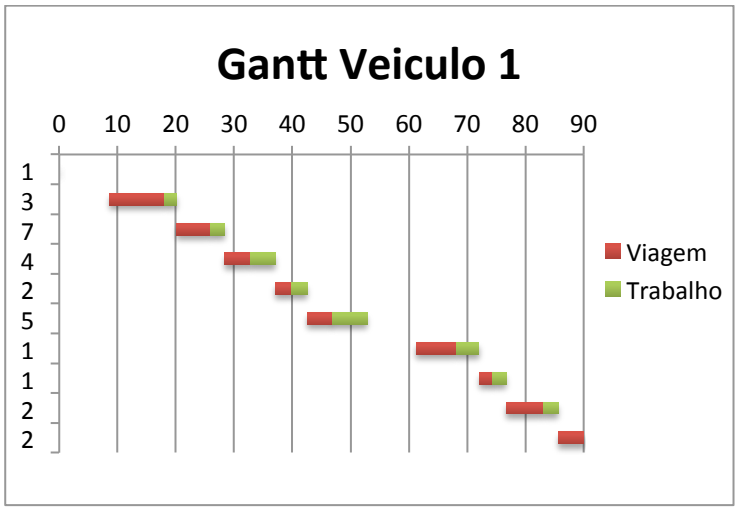


Figura 42 Diagrama de *Gantt* do veículo 1 da instância 1-3

Analogamente à criação do diagrama de *Gantt* ilustrado na Figura 42, criou-se um diagrama de disponibilidades para indicar a disponibilidade dos clientes que o veículo visitou. Para a criação deste gráfico, recorreu-se às colunas 3 e 11, que indica o instante de abertura de janela e o comprimento total da janela temporal. A Figura 43 ilustra a disponibilidade do veículo 1 da instância 25-3.

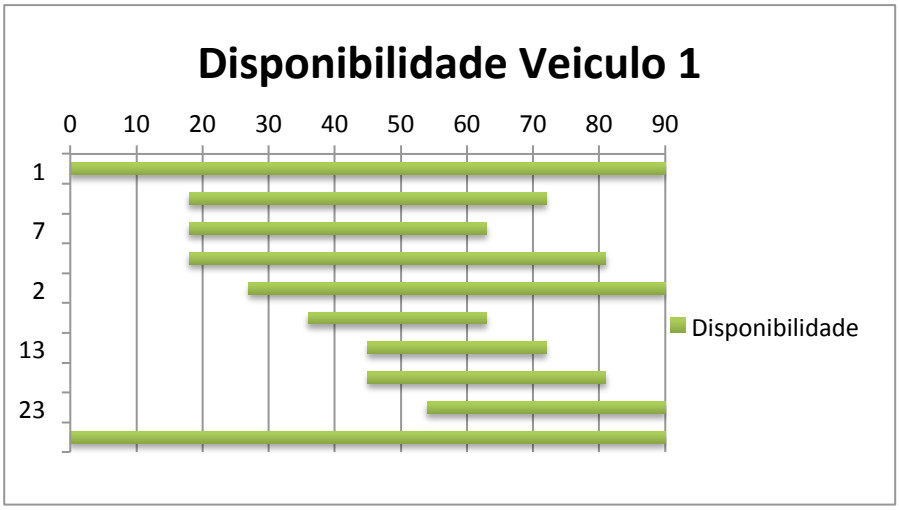


Figura 43 Diagrama de disponibilidades do veículo 1 da instância 1-3

No diagrama de *Gantt* ilustrado na Figura 42 é possível ver que, no vértice 3, embora a viagem se tenha iniciado por volta do instante 10, o trabalho só é iniciado quando a janela temporal o permite, no caso do vértice 3, a partir do instante 18. Sobrepondo os dois diagramas seria possível verificar que o tempo de trabalho, assinalados a verde na Figura 42, era abrangido pela disponibilidade, assinalada também a verde na Figura 43.

Verifica-se ainda que no final do trabalho no vértice 5 existe tempo em que não que não é utilizado em viagem ou em serviço, embora a janela temporal do vértice seguinte, o vértice 13 já permitisse que o trabalho fosse iniciado. No entanto os tempos de serviço respeitam o modelo, não havendo nenhuma violação de restrições temporais.

Por forma a sumariar a taxa de utilização dos veículos e a melhor identificar em que é gasto esse tempo, a Figura 44 ilustra a taxa de utilização dos diferentes veículos da instância.

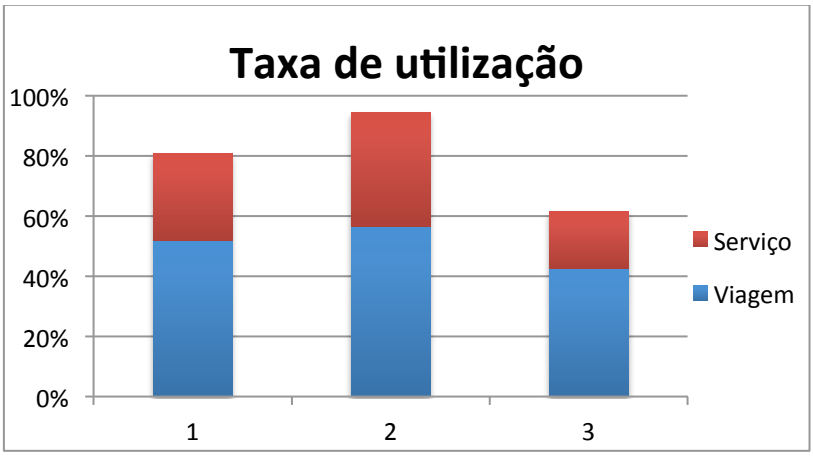


Figura 44 Taxa de utilização

É possível verificar que a maior quota de tempo de utilização é em viagens, mais evidentes no veículo 1 e no veículo 3, embora no veículo 2 a diferença também não passe despercebida.

Na Figura 45 é ilustrada, de forma global, a taxa de utilização dos três veículos utilizados na instância.

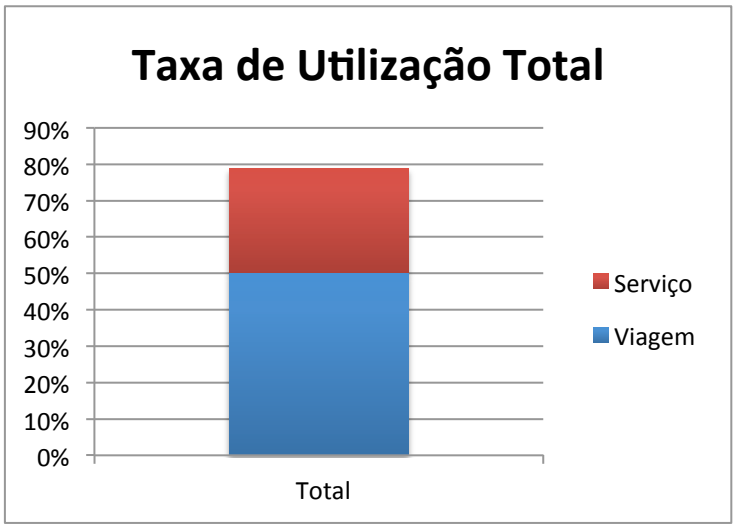


Figura 45 Taxa de utilização total

Como se constatou anteriormente, o tempo de viagem é significativamente superior ao tempo que efetivamente é utilizado em serviço efetivo. Mais precisamente, os veículos passaram cerca de 50% do tempo em viagem e apenas 29% do tempo em serviço. Enquanto que cerca de 21% do horizonte de planeamento ficou por preencher.

4. RESULTADOS

Neste capítulo apresentam-se os resultados obtidos da aplicação do TOPTW às diferentes instâncias estudadas, bem como estratégias alternativas e consequentes resultados na tentativa de melhorar a qualidade e eficiência das soluções.

4.1 Resultados TOPTW

Foram criadas 18 instâncias diferentes, para verificar o comportamento do modelo baseado no TOPTW aplicado às linhas de engarrafamento móveis com um horizonte de planeamento de 10 dias úteis. Para além da aleatoriedade dos dados dos clientes criados, como indicado na secção 3.9.1 da fase de Testes, foram concebidas instâncias com diferente número de vértices. E para cada conjunto de vértices gerados, criaram-se 5 instâncias, com diferente número de veículos, a variar entre 1 veículo e 5 veículos. As 46 e 47 ilustram as tabelas com os resultados obtidos nas diferentes instâncias para um mesmo horizonte de planeamento. A coluna (1) indica a instância, na coluna (2) indica-se o número de clientes da instância e a coluna (3) o número de veículos utilizados para a resolução do problema. Na coluna (4) são indicados o número de clientes visitados. Na coluna (5) e (6) são indicados os valores do prémio total recolhido e taxa média de utilização do(s) veículo(s). Nas dez colunas seguintes são indicadas as taxas de utilização e o prémio recolhido por veículo.

O número mínimo adotado para a criação de instâncias foi de 15 vértices e, para este número de clientes, localizações e disponibilidade, foi possível encontrar solução para todos os testes, com os diferentes veículos.

(1)	(2)	(3)	(4)	(5)	(6)	Taxa de Utilização					Prémio recolhido				
						1	2	3	4	5	1	2	3	4	5
data15-1	15	1	14	86689	0.95	0.95	N/A	N/A	N/A	N/A	86689	N/A	N/A	N/A	N/A
data15-2	15	2	15	101815	0.62	0.56	0.68	N/A	N/A	N/A	48760	53055	N/A	N/A	N/A
data15-3	15	3	15	101815	0.52	0.46	0.33	0.77	N/A	N/A	31280	31779	38756	N/A	N/A
data15-4	15	4	15	101815	0.38	0.29	0.29	0.61	0.47	N/A	21575	21575	27920	31584	N/A
data15-5	15	5	15	101815	0.29	0.07	0.43	0.07	0.49	0.13	14052	34487	14052	35535	2615
data16-1	16	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data16-2	16	2	16	141710	0.67	0.54	0.79	N/A	N/A	N/A	56375	85335	N/A	N/A	N/A
data16-3	16	3	16	141710	0.49	0.40	0.32	0.74	N/A	N/A	23783	12414	105513	N/A	N/A
data16-4	16	4	16	141710	0.37	0.36	0.40	0.28	0.43	N/A	8849	43038	38145	51678	N/A
data16-5	16	5	16	141710	0.35	0.12	0.29	0.12	0.46	0.37	7008	41893	34047	45711	13051
data17-1	17	1	11	180575	0.95	0.95	N/A	N/A	N/A	N/A	180575	N/A	N/A	N/A	N/A
data17-2	17	2	17	220446	0.90	0.94	0.86	N/A	N/A	N/A	116797	103649	N/A	N/A	N/A
data17-3	17	3	17	220446	0.68	0.80	0.37	0.86	N/A	N/A	93283	7096	120067	N/A	N/A
data17-4	17	4	17	220446	0.45	0.57	0.39	0.50	0.34	N/A	39458	35486	86192	59310	N/A
data17-5	17	5	17	220446	0.41	0.29	0.29	0.29	0.65	0.71	22754	50112	3683	51262	92635
data18-1	18	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data18-2	18	2	18	159465	0.81	0.94	0.68	N/A	N/A	N/A	77314	82151	N/A	N/A	N/A
data18-3	18	3	18	159465	0.56	0.56	0.66	0.46	N/A	N/A	51996	67982	39487	N/A	N/A
data18-4	18	4	18	159465	0.50	0.76	0.54	0.44	0.28	N/A	53380	47141	46465	12479	N/A
data18-5	18	5	18	159465	0.36	0.54	0.43	0.54	0.19	0.36	46549	40984	16407	23050	32475
data19-1	19	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data19-2	19	2	19	188982	0.82	0.85	0.79	N/A	N/A	N/A	114578	74404	N/A	N/A	N/A
data19-3	19	3	19	188982	0.63	0.34	0.81	0.75	N/A	N/A	36795	92306	59881	N/A	N/A
data19-4	19	4	19	188982	0.49	0.32	0.53	0.53	0.59	N/A	29519	23817	68196	67450	N/A
data19-5	19	5	19	188982	0.43	0.12	0.70	0.12	0.47	0.55	17552	64863	29519	30107	46941
data20-1	20	1	12	144565	1	1.00	N/A	N/A	N/A	N/A	144565	N/A	N/A	N/A	N/A
data20-2	20	2	KILL	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
data20-3	20	3	20	215596	0.71	0.67	0.63	0.83	N/A	N/A	71664	53305	90627	N/A	N/A
data20-4	20	4	20	215596	0.63	0.48	0.52	0.67	0.84	N/A	45596	20664	87893	61443	N/A
data20-5	20	5	20	215596	0.57	0.63	0.36	0.63	0.77	0.39	56993	35172	53578	60979	8874
data21-1	21	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data21-2	21	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data21-3	21	3	21	240079	0.67	0.59	0.69	0.74	N/A	N/A	89929	64372	85778	N/A	N/A
data21-4	21	4	21	240079	0.59	0.34	0.71	0.58	0.75	N/A	59407	64506	51777	64389	N/A
data21-5	21	5	21	240079	0.47	0.16	0.33	0.16	0.80	0.66	4247	47426	47844	91320	49242
data22-1	22	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data22-2	22	2	22	265418	0.93	0.88	0.97	N/A	N/A	N/A	129550	135868	N/A	N/A	N/A
data22-3	22	3	22	265418	0.72	0.67	0.89	0.61	N/A	N/A	77430	114371	73617	N/A	N/A
data22-4	22	4	22	265418	0.55	0.77	0.40	0.75	0.27	N/A	87793	74397	71610	31618	N/A
data22-5	22	5	22	265418	0.48	0.71	0.88	0.71	0.33	0.25	77438	105107	36334	26007	20532
data23-1	23	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data23-2	23	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data23-3	23	3	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data23-4	23	4	23	241701	0.55	0.39	0.72	0.72	0.35	N/A	62996	91701	58981	28023	N/A
data23-5	23	5	23	241701	0.51	0.53	0.75	0.53	0.59	0.58	41471	52896	8040	71129	68165

Figura 46 Tabela de resultados

(1)	(2)	(3)	(4)	(5)	(6)	Taxa de Utilização					Prémio recolhido				
						1	2	3	4	5	1	2	3	4	5
data24-1	24	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data24-2	24	2	24	297699	0.95	0.92	0.98	N/A	N/A	N/A	103693	194006	N/A	N/A	N/A
data24-3	24	3	24	297699	0.76	0.63	0.90	0.75	N/A	N/A	79124	111273	107302	N/A	N/A
data24-4	24	4	24	297699	0.60	0.49	0.38	0.73	0.83	N/A	63621	52594	76113	105371	N/A
data24-5	24	5	24	297699	0.49	0.15	0.30	0.15	0.81	0.66	27600	28901	81834	74052	85312
data25-1	25	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data25-2	25	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data25-3	25	3	25	239243	0.79	0.81	0.94	0.62	N/A	N/A	66044	106501	66698	N/A	N/A
data25-4	25	4	25	239243	0.67	0.61	0.80	0.61	0.66	N/A	80666	80408	23429	54740	N/A
data25-5	25	5	25	239243	0.54	0.52	0.23	0.52	0.63	0.78	43561	24750	22106	70794	78032
data26-1	26	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data26-2	26	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data26-3	26	3	26	293507	0.81	0.90	0.62	0.90	N/A	N/A	107890	109966	75651	N/A	N/A
data26-4	26	4	26	293507	0.69	0.82	0.72	0.36	0.87	N/A	62175	49393	56008	125931	N/A
data26-5	26	5	26	293507	0.62	0.84	0.44	0.84	0.65	0.71	94569	81519	17761	56708	42950
data27-1	27	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data27-2	27	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data27-3	27	3	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data27-4	27	4	27	265153	0.73	0.87	0.51	0.73	0.83	N/A	95570	34057	63458	72068	N/A
data27-5	27	5	27	265153	0.65	0.59	0.53	0.59	0.49	0.92	58361	26853	34684	31092	114163
data28-1	28	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data28-2	28	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data28-3	28	3	28	332861	0.84	0.89	0.73	0.92	N/A	N/A	108675	144674	79512	N/A	N/A
data28-4	28	4	28	332861	0.67	0.86	0.40	0.92	0.48	N/A	159629	37705	100924	34603	N/A
data28-5	28	5	28	332861	0.61	0.55	0.44	0.55	0.73	0.91	65195	30154	50080	106684	80748
data29-1	29	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data29-2	29	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data29-3	29	3	29	290818	0.88	0.94	0.88	0.82	N/A	N/A	92653	97123	101042	N/A	N/A
data29-4	29	4	29	290818	0.72	0.79	0.83	0.53	0.75	N/A	85036	75957	39231	90594	N/A
data29-5	29	5	29	290818	0.62	0.50	0.60	0.50	0.53	0.94	45749	33852	38338	41795	131084
data30-1	30	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data30-2	30	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data30-3	30	3	30	388645	0.88	0.97	0.80	0.88	N/A	N/A	108846	145489	134310	N/A	N/A
data30-4	30	4	30	388645	0.77	0.81	0.99	0.61	0.67	N/A	116770	126400	61824	83651	N/A
data30-5	30	5	30	388645	0.33	0.33	0.22	0.33	0.47	0.38	75922	116025	33543	85305	77850
data40-1	40	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data40-2	40	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data40-3	40	3	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data40-4	40	4	40	439745	0.89	0.93	0.96	0.76	0.92	N/A	84439	75089	150866	129351	N/A
data40-5	40	5	40	439745	0.74	0.86	0.56	0.86	0.91	0.65	67657	105659	65737	117864	82828
data50-1	50	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data50-2	50	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data50-3	50	3	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data50-4	50	4	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data50-5	50	5	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL

Figura 47 Tabela de resultados

Existem limitações de utilização do NEOS Server que não permitem obter resultados em algumas instâncias. As limitações são o tempo limite de processamento, fixado em 8 horas e 3 GB memória disponíveis. Os processos de resolução das instâncias que violem uma dessas limitações são forçados a terminar. Estas instâncias são representadas, nas 46 e 47, com a linha preenchida com a palavra KILL. O valor N/A (Não Aplicável) indicam que não podem ser obtidos valores para a taxa de utilização e prémio recolhido, uma vez que esses veículos não são considerados na instância.

Na instância mais pequena, com 15 vértices e 1 veículo, é possível verificar que a utilização foi 95%, mas ficou um cliente por visitar e consequentemente o prémio total foi inferior ao somatório do prémio de todos os vértices. Esta é a essência dos OPTW/TOPTW, recolher o maior prémio possível no

contexto onde a procura é superior à oferta, tendo em conta a disponibilidade dos clientes, e o horizonte de planeamento. Trata-se de uma tarefa que é dificultada à medida que o problema cresce. Utilizando dois veículos, verifica-se que já é possível visitar todos os vértices, e consequentemente atingir o prémio máximo possível, com uma taxa média de utilização de 62%. Conseguindo-se visitar todos os vértices, a instância torna-se numa instância fácil de resolver para o modelo, uma vez que a procura torna-se perfeitamente alcançável pela oferta. O TOPTW não tenta minimizar distâncias percorridas mas antes maximizar a recolha de prémios. Assim, a partir daqui, incrementando o número de veículos disponíveis para resolver o problema, vamos ter como consequência uma baixa na taxa de utilização média dos veículos, com 52%, 38% e 29% utilizando 3, 4 e 5 veículos, respetivamente. Analisando a solução apresentada pela utilização de 5 veículos torna-se evidente que este modelo não está preparado para balancear a utilização dos veículos, pois a utilização dos veículos são de 7%, 43%, 7%, 49% e 13% para os veículos 1, 2, 3, 4 e 5, respetivamente. Verifica-se assim uma disparidade muito grande nas taxas de utilização dos diferentes veículos. Teriam de ser criadas restrições no modelo para tentar balancear estas taxas.

Para as instâncias com 17 vértices, conseguiu-se solução para todas elas, embora na primeira solução os vértices não podem ser todos visitados, à semelhança do que aconteceu com as instâncias de 15 vértices. A taxa de utilização para as duas primeiras (1 e 2 veículos) mantem-se a superior a 90%, enquanto que com 3 veículos a taxa de utilização média baixou significativamente para os 68%. Com 4 e 5 veículos os valores já são inferiores a 50%, com 45% e 41%, respetivamente. A taxa média de utilização com 1 e 2 veículos é cerca de 90%, e entende-se que quando adaptado à realidade, estas soluções tem grande probabilidade de falhar, uma vez que neste problema real é possível a ocorrência de imprevistos. Assim, no nosso entender, as soluções ideais (mais prudentes e conservadoras) são soluções a rondar os 80%. Assim continuamos a ter uma taxa de utilização de boa qualidade, e mesmo que os imprevistos aconteçam, temos mais margem de manobra e de conseguir garantir que os trabalhos são realizados nos prazos estabelecidos.

Nas instâncias com 20 vértices, conseguiu-se chegar a uma solução com apenas um veículo, em que o prémio total recolhido é de 144565, e a taxa de utilização é de 99.99%. Nesta solução são visitados apenas 12 dos 20 vértices disponíveis. Pela taxa de utilização verifica-se que, com os recursos disponíveis, é o máximo que se consegue. No entanto, com este mesmo conjunto de vértices e 2 veículos, não foi possível encontrar uma solução ótima, por ter atingido um dos limites do NEOS Server. Com 3, 4 e 5 veículos já foi possível encontrar uma solução, que visita todos os vértices. Em

todas elas recolhe-se o prémio máximo possível e a taxa de utilização média não é menos do que 50%, com 71%, 63% e 57% para 3, 4 e 5 veículos, respetivamente.

A partir da instância com 23 vértices, já não se consegue obter resposta para as instâncias com 1, 2 e 3 veículos, o que evidencia já a dificuldade em se obter soluções utilizando métodos de otimização (solução exata) em instâncias de pequena dimensão. Para as instâncias que utilizam 4 e 5 veículos, consegue-se obter solução. Estas soluções são eficazes, pois visitam todos os vértices, mas são pouco eficientes uma vez que a taxa de utilização média é de 55% e 51% para utilização com 4 e 5 veículos respetivamente. Seria necessário mais tempo de processamento e/ou mais memória disponível NEOS Server para encontrar a solução ótima utilizando menos do que 4 veículos.

Analogamente, nas instâncias com 40 vértices, conseguiu-se solução quando se utilizou 4 e 5 veículos na resolução. Neste caso as taxas de utilização já são mais elevadas, com 89% e 74%, com 4 e 5 veículos, respetivamente, fruto de mais encomendas presentes nas instâncias. Estas taxas traduzem um sistema muito congestionado, pelo que a implementação no problema real possa ser crítica.

Na instância com 50 vértices já não se consegue encontrar nenhuma solução, utilizando até 5 veículos, pois todos os trabalhos foram forçados a encerrar por violar as limitações do servidor.

Uma vez que estes problemas pertencem à classe de problemas *NP-difícil,* à medida que a dimensão da instância do problema cresce, cresce exponencialmente o tempo necessário para obter uma solução ótima. A utilização de um modelo (matemático) compacto para resolução no NEOS Server está condicionada a instâncias de pequena ou média dimensão. A obtenção da solução ótima passa por se utilizar outras metodologias mais avançadas como é o caso dos métodos baseados em geração de colunas, com outros recursos de computação. Para se obter soluções válidas para o problema, uma alternativa consiste em se recorrer ao uso de heurísticas ou meta-heurísticas. Estes métodos são reconhecidos por conseguirem obter soluções bastante aceitáveis em tempo útil, mesmo para se conseguir solucionar problemas de grande dimensão.

Após a análise dos resultados para as instâncias correspondentes a um planeamento de 10 dias úteis, e dada a dificuldade em obter-se resultados à medida que o número de vértices aumentava, decidiu-se aumentar o horizonte de planeamento para o dobro do tempo, para 20 dias úteis, e verificar o impacto desta alteração.

Adicionou-se uma variável à nomenclatura das instâncias anteriormente apresentada, para diferenciar instâncias de horizontes de planeamento diferentes. Assim, a nomenclatura é *xml-data{A}-{B}-{C}*, em que *{A}* e *{B}* representam, como anteriormente definido, o número de vértices e número de

veículos disponíveis, respetivamente. A variável *{C}* representa o número de horas consideradas para o horizonte de planeamento. Os ficheiros para guardar os resultados destas instâncias seguem a nomenclatura semelhante.

A Figura 48 ilustra a tabela com os resultados obtidos no NEOS Server para o novo horizonte de planeamento.

(1)	(2)	(3)	(4)	(5)	(6)	Taxa de Utilização					Prémio recolhido				
						1	2	3	4	5	1	2	3	4	5
data50-1-180	50	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data50-2-180	50	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data50-3-180	50	3	50	402001	0.83	0.73	0.79	0.97	N/A	N/A	116434	140854	144713	N/A	N/A
data50-4-180	50	4	50	402001	0.69	0.76	0.77	0.56	0.67	N/A	113669	112449	86332	89552	N/A
data50-5-180	50	5	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data100-1-180	100	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data100-2-180	100	2	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data100-3-180	100	3	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data100-4-180	100	4	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
data100-5-180	100	5	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL

Figura 48 Resultados para um horizonte de planeamento de 20 dias

Como é possível verificar, na instância com 50 vértices é agora possível encontrar soluções para as instâncias com 3 e 4 veículos, em que todos os vértices são visitados e as taxas de utilização são de 83% e 69%, respetivamente. Já nas instâncias com 100 vértices, embora seja para um horizonte de planeamento maior, não foi possível encontrar nenhuma solução para as referidas instâncias, ou seja, todas elas violaram pelo menos uma das restrições de utilização do NEOS Server.

Torna-se evidente que aumentar o horizonte de planeamento não foi uma boa opção e que, na maioria dos casos, o NEOS Server não consegue dar resposta. Quando consegue dar resposta, todos os vértices são visitados, o que quer dizer que a instância se tornou numa instância fácil.

Nesta altura, para se tentar obter soluções ótimas para instâncias de maior seria necessário recorrer a métodos de geração de colunas como *branch-and-cut, branch-and-price* e *branch-and-cut-and-price*.

4.2 Abordagem VRP

Por forma a tentar melhorar a qualidade das soluções, tentou-se uma abordagem baseada no conceito do VRP *(vehicle routing problem)* para forçar o modelo a escolher de entre todas as soluções que visitam todos os vértices a solução menor distância percorrida, o que necessariamente tem impacto nos custos operacionais do problema real. Por forma a podermos compara as soluções, utilizou-se uma instância das já definidas anteriormente, a instâncias data17. A alteração é feita no modelo ao nível da função objetivo. A formula (46) ilustra a função objetivo do TOPTW e a formula (47) ilustra a função objetivo alterada, baseada no conceito do VRP.

$$Max \sum_{c=1}^{carros} \sum_{i=1}^{N-1} \sum_{j=2}^{N} x_{c,i,j} * premio_j \quad (46)$$

$$Max \sum_{c=1}^{carros} \sum_{i=1}^{N-1} \sum_{j=2}^{N} x_{c,i,j} * (premio_j - distancias_{ij}) \quad (47)$$

Com diferentes funções objetivo, é óbvio que as soluções vão dar resultados diferentes para o valor da função objetivo. Porém, a comparação será efetuada ao nível dos prémios recolhidos (que devem manter-se) e das distâncias percorridas que deverão ser diferentes caso haja mais que uma solução alternativa para recolher a totalidade dos prémios.

A Figura 49 ilustra uma tabela que compara os resultados obtidos pelos diferentes modelos.

#	TOPTW		VRP		Poupança
	F.O.	KMS	F.O.	KMS	
17-1	180575	1584	180535.5	1579	5
17-2	220446	3885	220392.6	2137	1748
17-3	220446	4687	220391.4	2186	2501
17-4	220446	3857	220393.1	2117	1740
17-5	220446	4699	220388.6	2117	2582

Figura 49 Comparação entre abordagem VRP e TOPTW

Analisando a tabela ilustrada na Figura 49 torna-se evidente que o modelo baseado em VRP consegue visitar os mesmos vértices percorrendo menos quilómetros. Nesta instância, utilizando apenas um veículo, a poupança não é muito significante, cerca de 0,32%. Conseguiu-se uma poupança de 5 quilómetros, visitando os mesmos vértices, alterando apenas a ordem de visita. No entanto, quando o número de veículos aumenta, é possível visitar todos os vértices, a poupança já é bastante significativa. As poupanças são de 45%, 53.36%, 45.11% e 33.67% para a utilização de 2, 3, 4 e 5 veículos. Quando são utilizados 2 veículos, a poupança chega mesmo a ser mais de metade. Ou seja, percorrendo mesmo de metade dos quilómetros, conseguimos visitar todos os vértices. Na Figura 50 mostra o gráfico das distâncias percorridas pelos dois modelos.

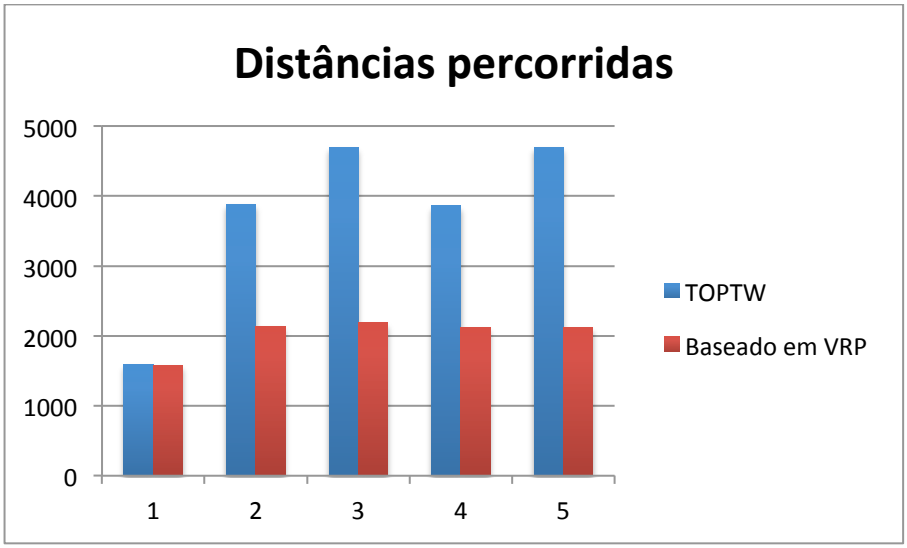


Figura 50 Comparação de distâncias percorridas

4.3 Abordagens de resolução

4.3.1 Clusters

Analisando as soluções devolvidas pelo NEOS Server é possível concluir que, devido à complexidade do problema, o modelo poucas vezes consegue dar respostas quando é forçado a escolher vértices para visitar, sendo na sua maioria das vezes interrompido pelo NEOS Server, devido às suas limitações. Assim, tentou-se uma abordagem diferente na tentativa de obter soluções, mesmo para instâncias maiores.

A abordagem passa pela criação de *clusters*, em que cada *cluster* agrupa vértices de diferentes zonas do país. Os *clusters* foram definidos atendendo à configuração geográfica, resultando em três

zonas, a zona Norte, zona Centro e zona Sul. Para criar esta divisão traçaram-se duas linhas a dividir o país em três. Um linha passa pela cidade de Coimbra e outra pela cidade de Évora. A Figura 51 ilustra as linhas de divisão.

Figura 51 Divisão por clusters

Utilizou-se na instância de 50 vértices e, aplicando esta divisão, o resultado foram 3 instâncias com 22 vértices na zona Norte, 16 vértices na zona Centro e 14 vértices na zona Sul. Na Figura 51 é possível verificar a localização dos diferentes clientes da instância. Na instância original, o vértice de partida localiza-se no Norte e assume-se que a viagem para visitar os vértices do centro e do sul é iniciada igualmente no Norte do país. Assim, as instâncias da zona Centro e zona Sul vão ficar com 18 e 16 vértices, respetivamente. Inicialmente utilizou-se apenas um veículo por zona.

Após enviar os trabalhos com as diferentes instâncias ao NEOS Server, apenas se conseguiu obter resposta para a zona Sul. Os resultados são ilustrados na Figura 52.

(1)	(2)	(3)	(4)	(5)	(6)	Taxa de Utilização					Prémio recolhido				
						1	2	3	4	5	1	2	3	4	5
Norte	22	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
Centro	18	1	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL	KILL
Sul	14	1	13	131921	0.99	0.99	N/A	N/A	N/A	N/A	131921	N/A	N/A	N/A	N/A

Figura 52 Resultados por clusters (1ª iteração)

Analisando a tabela ilustrada na Figura 52 é possível verificar que o modelo foi obrigado a escolher vértices, e não visitou um deles.

Como não se conseguiram resultados para os dois primeiros *clusters*, incrementou-se o número de veículos em cada *cluster*. Depois de enviados os trabalhos ao NEOS Server. A Figura 53 ilustra a tabela de resultados para a 2ª iteração.

(1)	(2)	(3)	(4)	(5)	(6)	Taxa de Utilização					Prémio recolhido				
						1	2	3	4	5	1	2	3	4	5
Norte	22	2	22	194453	0.73	0.77	0.69	N/A	N/A	N/A	105172	89281	N/A	N/A	N/A
Centro	18	2	18	211601	0.65	0.56	0.74	N/A	N/A	N/A	78470	133131	N/A	N/A	N/A
Sul	14	1	13	131921	0.99	0.99	N/A	N/A	N/A	N/A	131921	N/A	N/A	N/A	N/A
TOTAL	54	5	53	537975	0.79	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Figura 53 Resultados por clusters (2ª iteração)

Analisando a tabela ilustrada na Figura 53 verifica-se que já é possível obter resposta. Na instância da zona Norte, as taxas de utilização são de 77% e 69%% para os veículos 1 e 2, respetivamente e a taxa de utilização média é de 73%. São taxas de utilização aceitáveis e, caso exista algum atraso em algum dos serviços, há uma margem de manobra considerável para evitar que algum dos serviços não seja feito.

Na instância da zona Centro, as taxas de utilização são um pouco mais baixas do que as taxas de utilização alcançadas na zona Norte, com 56% e 74% para o veículo 1 e 2, respetivamente. A taxa de utilização média é de 65%. Há semelhança do que acontece com a instância do norte, são taxas de utilização aceitáveis.

Em outras instâncias, a taxa de utilização é bastante inflacionada pela taxa de utilização em viagem, ora veja-se as taxas de utilização da instância data25, para 3, 4 e 5 veículos, ilustrada na Figura 54.

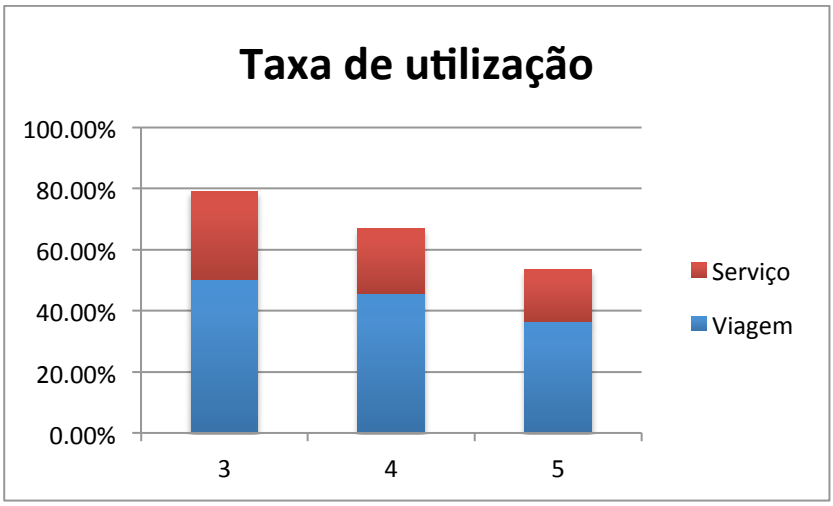


Figura 54 Taxas de utilização data25

Esta inflação acontece porque o modelo não pondera as distâncias percorridas, apenas os prémios a recolher.

Ao utilizar a estratégia de *clusters*, dividindo o problema em três zonas geográficas, era espectável que a taxa de utilização em viagem baixasse, dado que os vértices a tratar em cada instância estão mais próximos uns dos outros. A Figura 55 ilustra a taxa de utilização dividida por serviço e viagem.

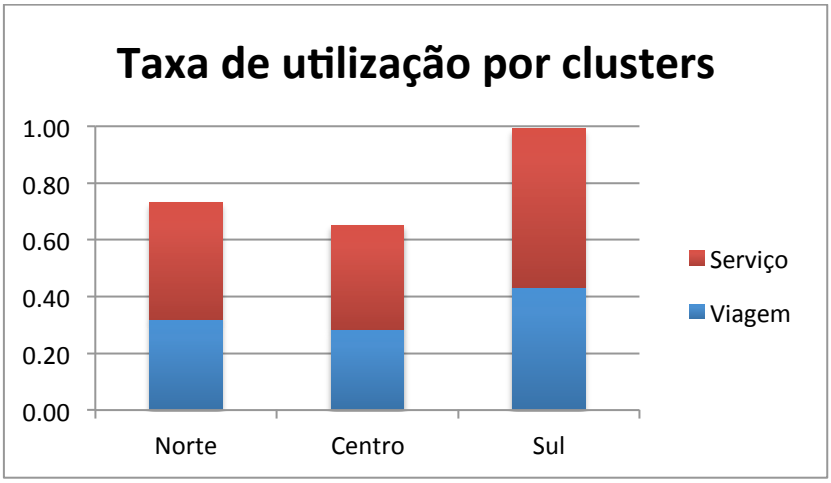


Figura 55 Taxa de utilização (clusters)

Analisando o Figura 55 verifica-se que, efetivamente, o tempo de viagem passou a ser a parcela mais pequena da taxa de utilização. Em todos os *clusters*, a taxa de utilização em serviço é superior à taxa de utilização em viagem.

A Figura 56 ilustra o diagrama de *Gantt* do cluster da zona Norte para o veículo 1.

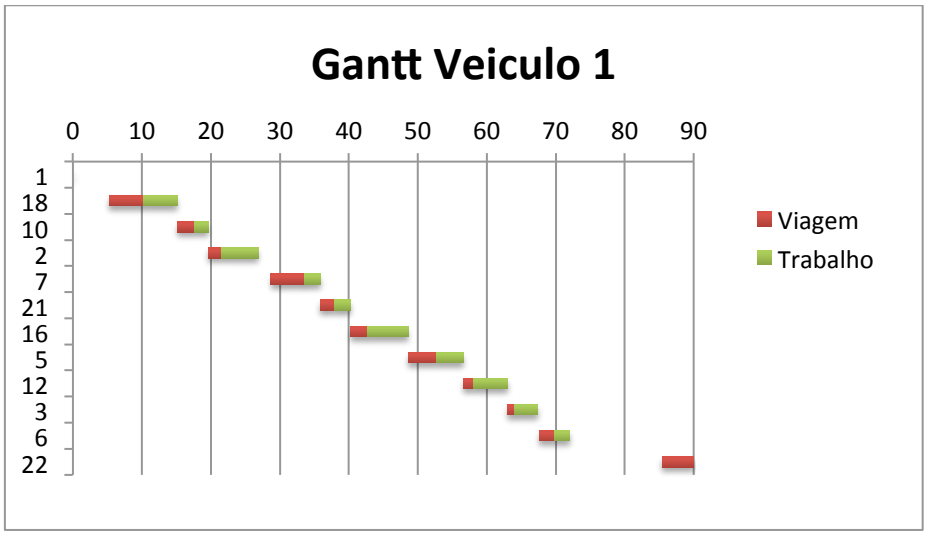


Figura 56 Diagrama de Gantt do cluster Norte - veículo 1

A Figura 57 mostra o diagrama de *Gantt* do cluster da zona Norte para o veículo 2.

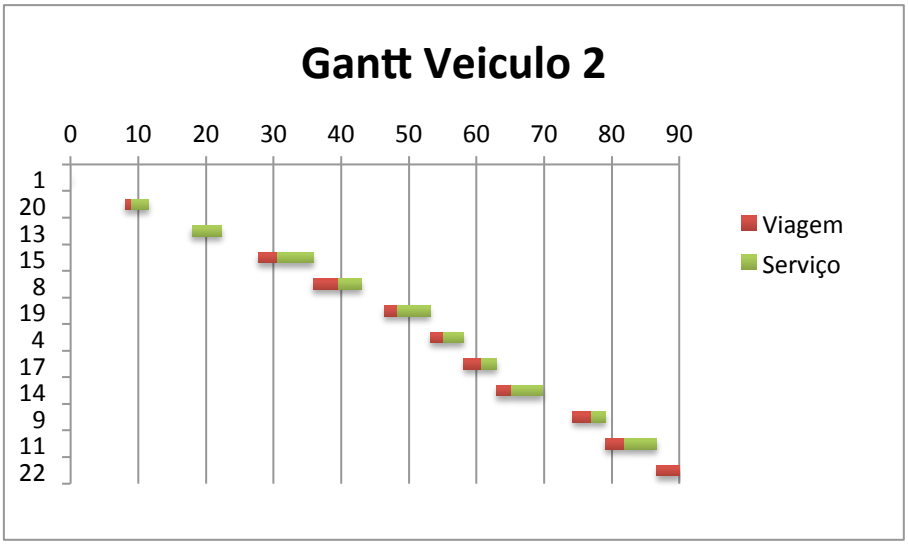


Figura 57 Diagrama de Gantt do cluster Norte - veículo 2

A Figura 58 mostra o diagrama de *Gantt* do cluster da zona Centro para o veículo 1.

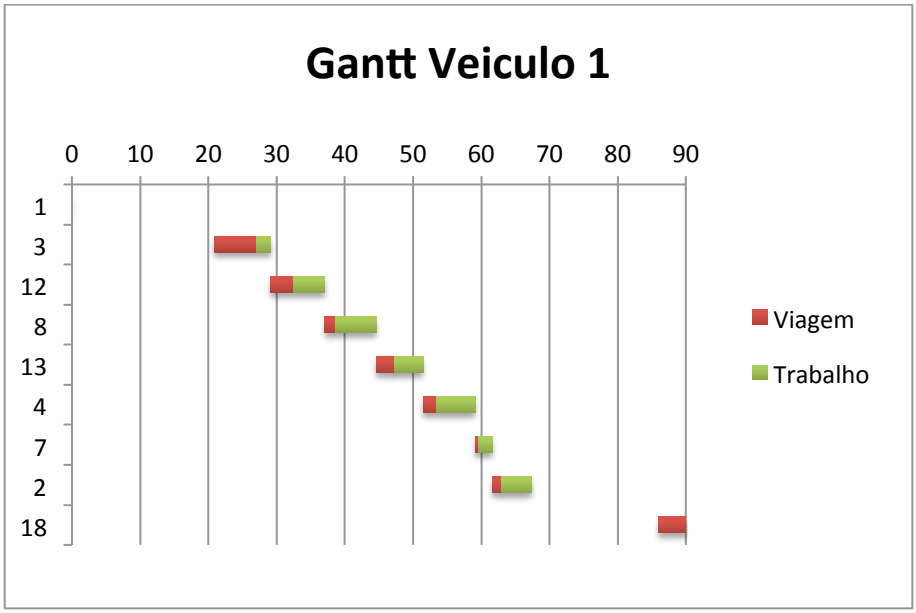


Figura 58 Diagrama de Gantt cluster Centro - veículo 1

A Figura 59 ilustra o diagrama de *Gantt* do cluster da zona Centro para o veículo 2.

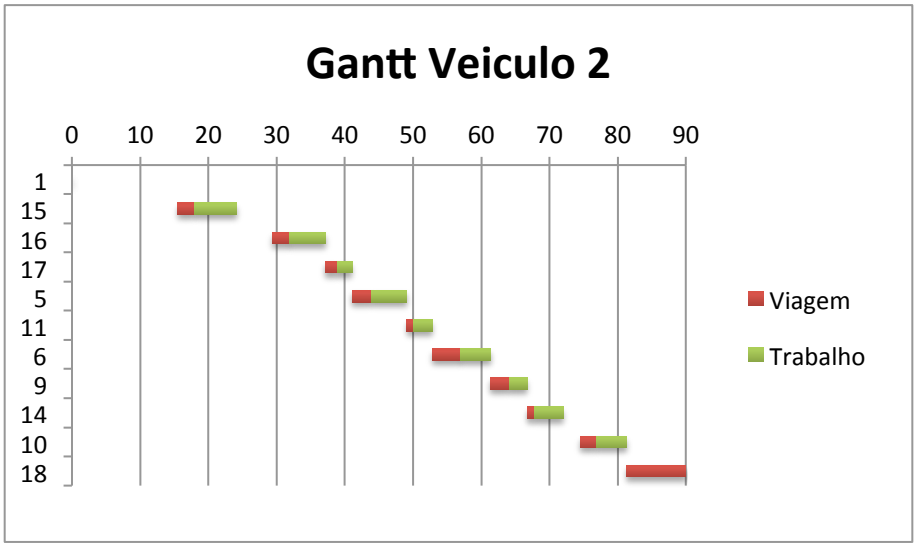


Figura 59 Diagrama de Gantt cluster Centro - veículo 2

Por fim, a Figura 60 ilustra o diagrama de *Gantt* do cluster da zona Sul.

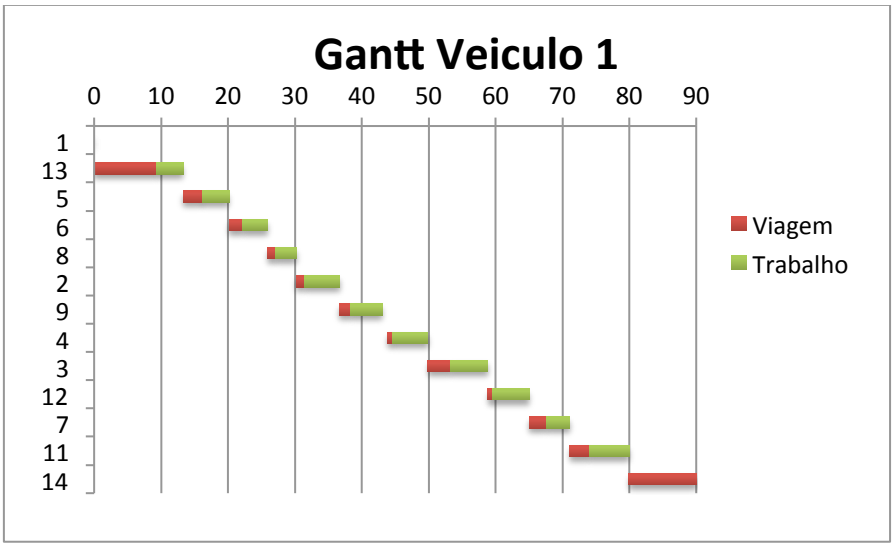


Figura 60 Diagrama de Gantt cluster Sul veículo 1

Uma vez que o vértice de origem está localizado no Norte, é possível verificar pelos diagramas e *Gantt* da zona Centro e Zona Sul, que a viagem inicial e a viagem final tem uma duração bastante superior às viagens intermédias entre os clientes.

Aplicando esta estratégia, foram necessários 5 veículos para apresentar resultados e, mais uma vez, apenas foram obtidas respostas nas instâncias do Norte e Centro devido a estas instância se terem tornado em instâncias fáceis, já que todos os vértices foram visitados.

Tendo presente as limitações do NEOS Server torna-se necessário recorrer a outras metodologias para obter soluções, mesmo que estas não sejam as ótimas. As alternativas são recorrer

a Algoritmos Construtivos, Procura Local ou Meta-Heurísticas. A qualidade das soluções apresentadas pelas meta-heurísticas geralmente são de qualidade superior, no entanto estas implicam mais tempo de resposta e maior custo computacional quando comparadas com as alternativas de procura local e algoritmos heurísticos construtivos. Pelo que há problemas que não podem recorrer a meta-heurística dada a rapidez de resposta que estas exigem. A procura local são interações sobre os algoritmos heurísticos construtivos, e geralmente são o meio termo entre a qualidade das soluções e o esforço computacional. Os algoritmos heurísticos construtivos apresentam boas soluções, aproximadas do ótimo, dando resposta a problemas complexos ou ainda em problemas cujo espaço temporal é curto para a obtenção de soluções exatas.

A Figura 61 ilustra o comportamento destas três abordagens heurísticas no que diz respeito à qualidade da solução obtida e ao tempo de computação utilizado.

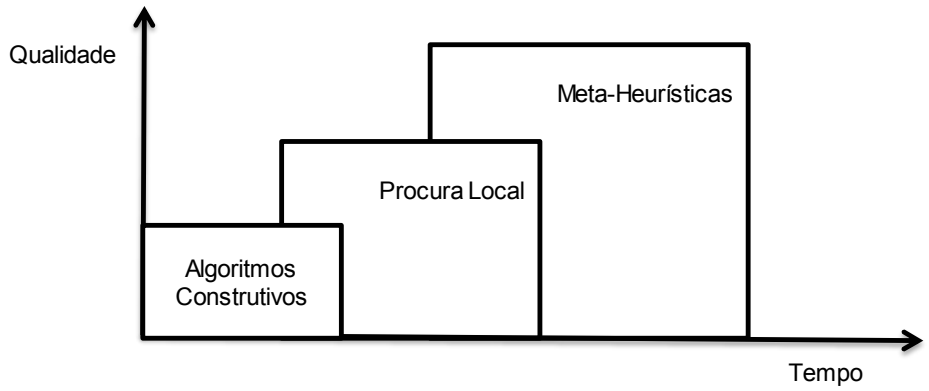


Figura 61 Gráfico Qualidade / Tempo das abordagens para resolução do problema

O ideal seria conseguir continuadamente soluções de elevada qualidade, muito próximas do óptimo, em curtos espaços de tempo. Entendendo estes problemas, percebe-se que há medida a qualidade das soluções aumenta, esse aumento tem um custo, e esse custo é o tempo e o esforço computacional.

4.3.2 Heurística construtiva

Uma outra alternativa aos *clusters* é a utilização de uma heurística construtiva. Uma heurística visa obter soluções em tempo útil, seguindo um algoritmo construtivo. Os vértices são ordenados por preferência de visita, ou seja, por ranking. O ranking utilizado é uma fórmula ponderada, que é alterada consoante três atributos do vértice. Os atributos que se consideram são o prémio a recolher, o valor da abertura da janela e o tamanho da janela. A Formula 1 ilustra o ranking que se considerou.

$$ranking = 0.3 * premio + 0.5 * (90 - startW) + 0.2 * (endW - startW)$$

Formula 1 Fórmula de ranking

A fórmula de ranking pondera 30% para o prémio, 50% para o início da janela, penalizando aqueles que comecem mais tarde e por fim, 20% para o tamanho da janela temporal. A Figura 62 ilustra o organograma do algoritmo da heurística construtiva criada.

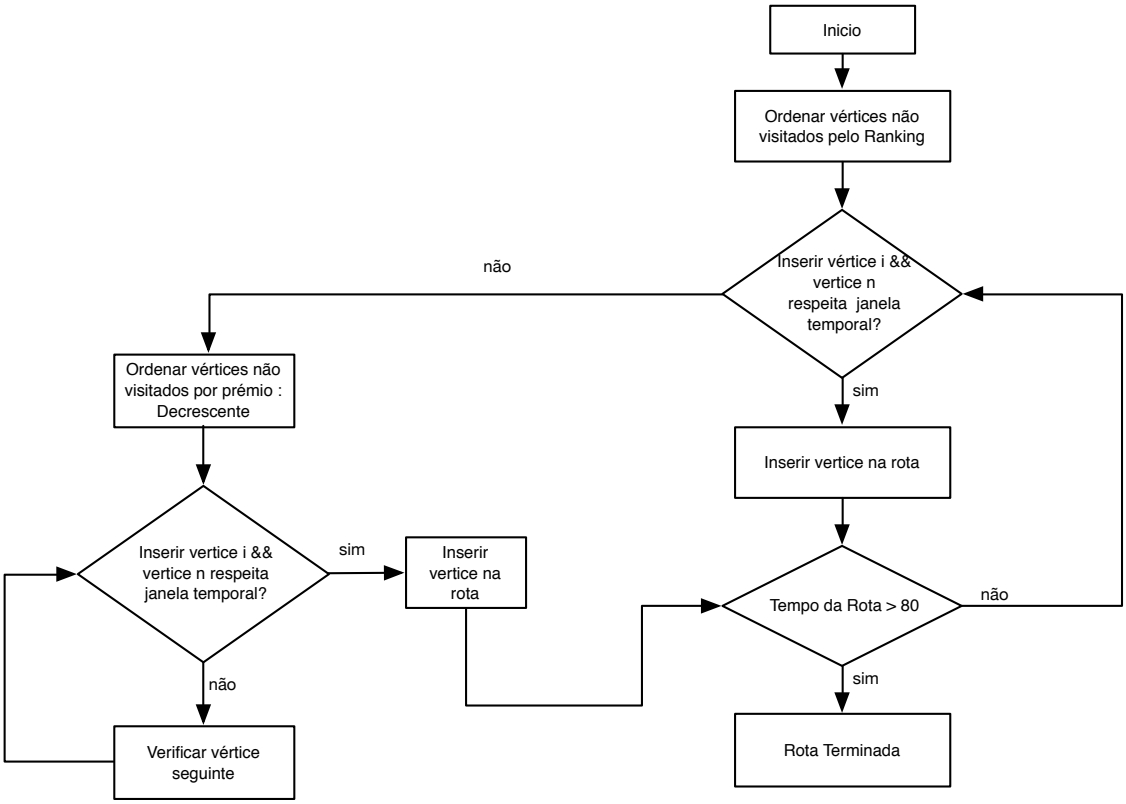


Figura 62 Organograma do algoritmo da heurística construtiva

Inicialmente ordenam-se os vértices pelo *ranking* descrito na Formula 1. Depois de ter os vértices ordenados, vamos iniciar a criação de rotas. Todas as rotas se iniciam no vértice 1 e terminam no vértice *N.* Verificar se o vértice com maior ranking, ainda não visitado, depois de inserido na rota respeita as janelas temporais, ou seja, se respeita a janela temporal do vértice a inserir e se a chegada ao vértice N respeita o horizonte de planeamento. Se sim, então insere-se o vértice na rota, e verificamos se o tempo da rota é maior do que 80, para respeitar as taxas de utilização que se consideram aceitáveis. Caso seja superior a 80, termina-se a rota, caso contrário, tenta-se inserir o vértice seguinte da lista ordenada pelo ranking. Se o vértice não respeitar as janelas temporais,

ordenam-se por ordem decrescente de prémio todos os vértices que ainda não foram visitados. E tenta-se inserir, na rota, o vértice com mais prémio. Se o vértice respeitar os janelas temporais, inserir vértice na rota e verificar se o tempo da rota já ultrapassa as 80 horas. Se já ultrapassar, terminar a rota, caso contrário tentar inserir o próximo vértice com maior ranking.

Depois de ordenar os vértices pelo *ranking* descrito na Formula 1, aplicou-se o algoritmo para obter rotas para os diferentes veículos.

A Figura 63 Resultados heurística construtiva ilustra os resultados de aplicar a heurística à instância data50-5.

Veículo	Prémio	Tx Utilização
1	103876	90.82%
2	80913	89.25%
3	121202	89.87%
4	85622	82.17%
5	57914	74.88%
Total	449527	85.40%
Visitados	34	

Figura 63 Resultados heurística construtiva

Analisando a tabela ilustrada na Figura 63 é possível verificar que, com 5 veículos, foram visitados 34 vértices. O prémio total recolhido é de 449527, ao passo que a taxa de utilização média é de 85%.

A Figura 64 mostra o diagrama de *Gantt* resultante da heurística construtiva referente ao veículo 1.

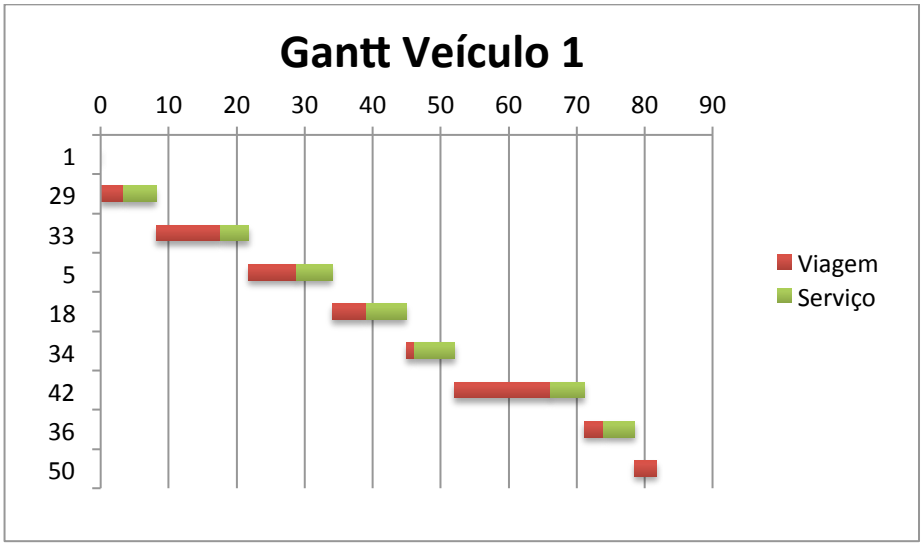


Figura 64 Diagrama de Gantt veículo 1

A Figura 65 ilustra o diagrama de *Gantt* da rota do veículo 2.

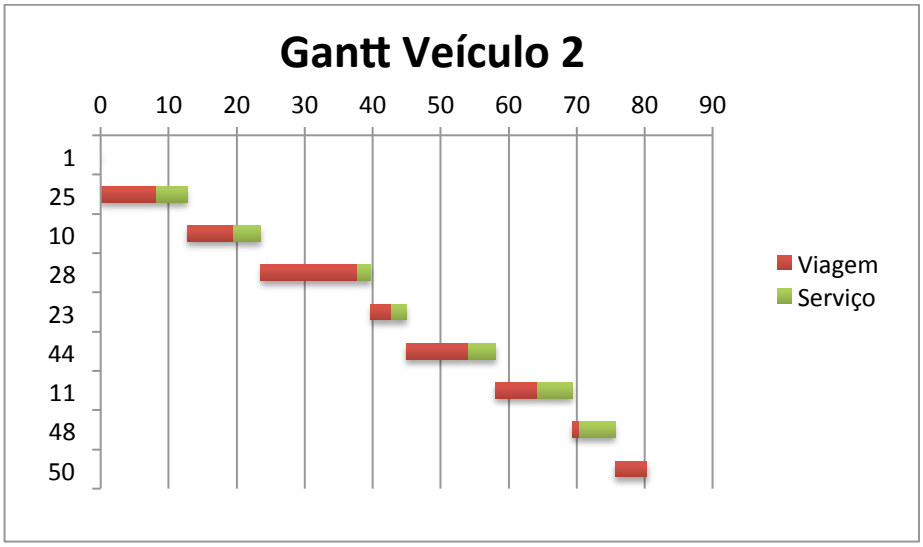


Figura 65 Diagrama de Gantt veículo 2

A Figura 66 mostra o diagrama de *Gantt* para o veículo 3.

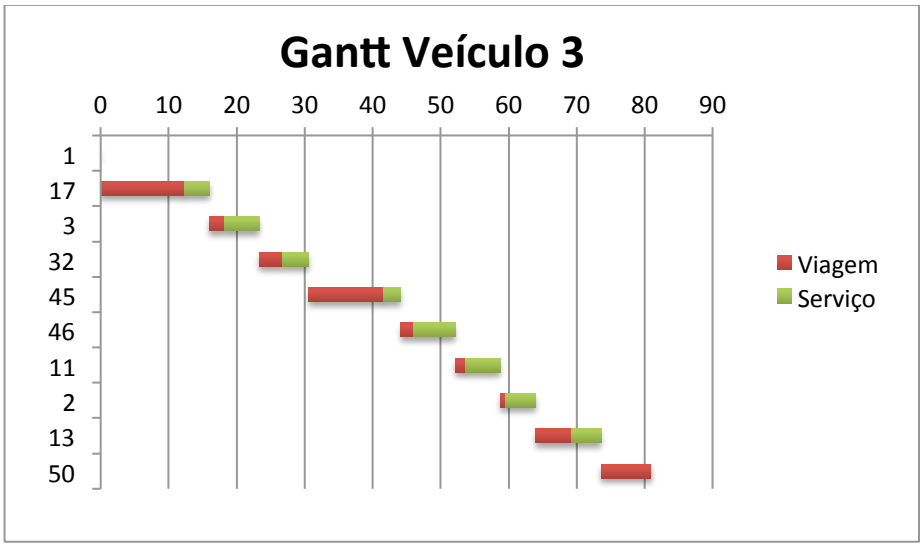


Figura 66 Diagrama de Gantt veículo 3

A Figura 67 ilustra o diagrama de *Gantt* da rota do veículo 4.

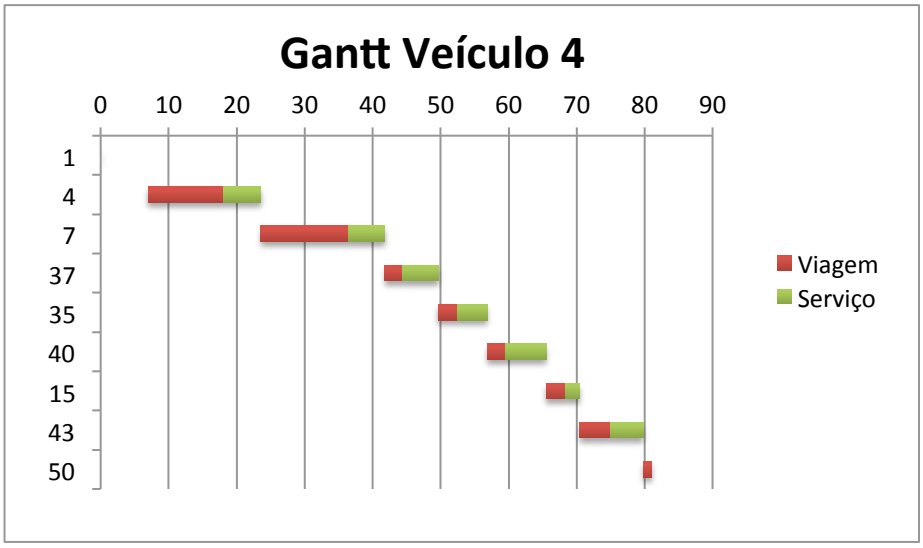


Figura 67 Diagrama de Gantt veículo 4

Por fim, a Figura 68 mostra o Diagrama de *Gantt* da rota do veículo 5.

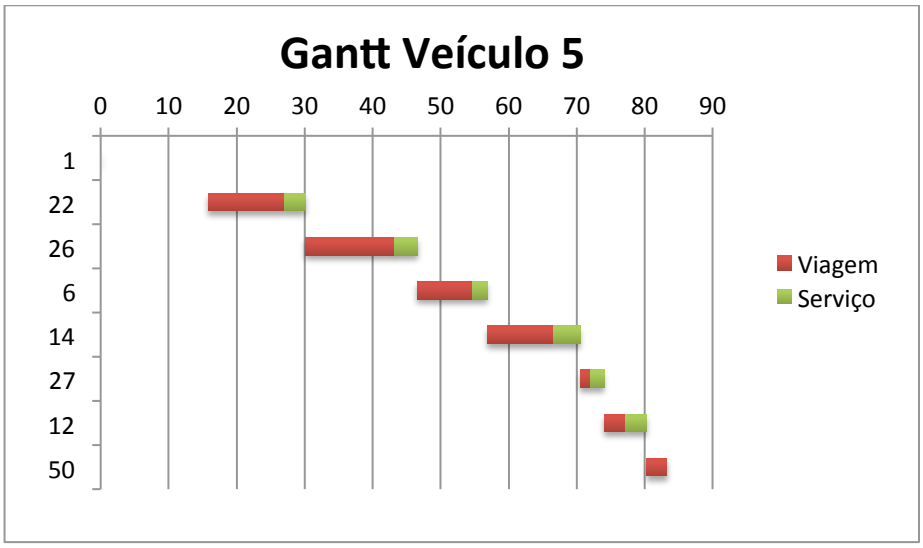


Figura 68 Diagrama de Gantt veículo 5

Analisando os diagramas de *Gantt* é possível verificar que a heurística está a gastar muito do seu tempo em viagens.

A Figura 69 mostra a taxa de utilização dividida pelo tempo de serviço e tempo de viagem. Nesta figura torna-se evidente que as taxas de utilização em tempos de viagens são exageradamente altas.

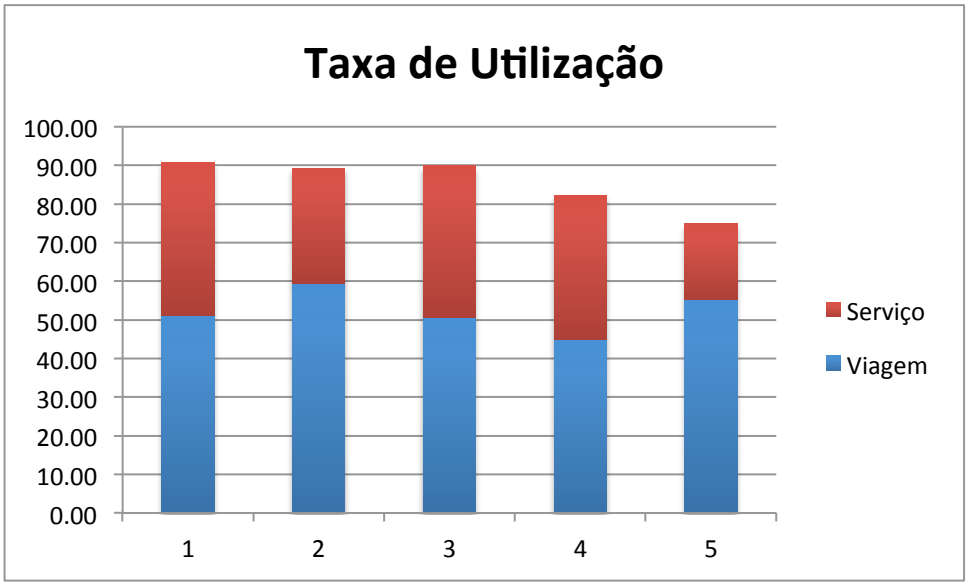


Figura 69 Taxa de utilização heurística construtiva

A Figura 70 ilustra um gráfico com a taxa de utilização global, dividida pelo tempo de viagem e pelo tempo de Serviço.

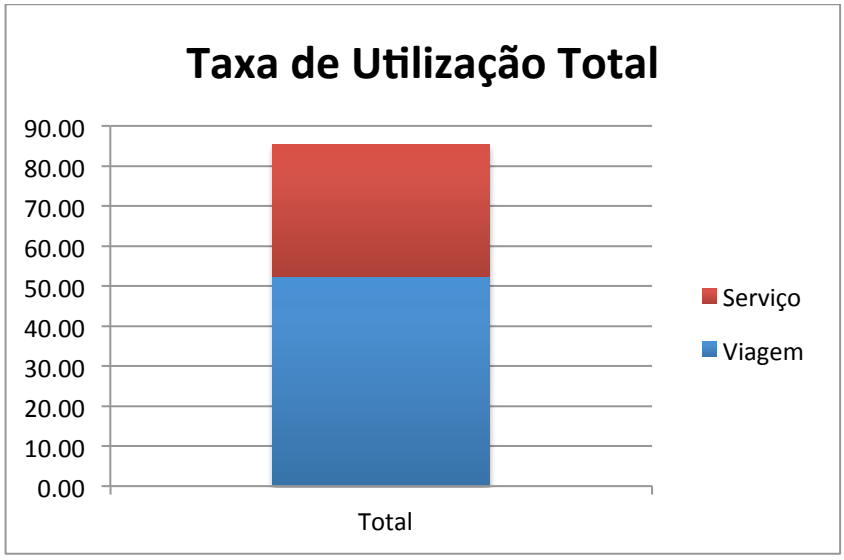


Figura 70 Taxa de utilização total

Analisando o gráfico da Figura 70 conclui-se que, cerca de 47% da taxa de utilização é gasto apenas em viagens. Um valor bastante elevado. O tempo de serviço é de cerca de 30%, resultado de não se visitarem todos os clientes, mas apenas 34. O restante valor, cerca de 23% é o valor em que os veículos se encontram parados.

Relativamente à heurística construtiva, esta está a construir as rotas em série, podendo as últimas rotas a serem definidas não terem hipótese de escolher bons vértices para iniciar logo que possível, já que estes vértices podem ter sidos já escolhidos a iniciar mais tarde no horizonte de

planeamento, veja-se os diagramas de Gantt ilustrados nas Figuras 67 e 68, em que os serviços se iniciam apenas no após 18 e 27 horas, respetivamente. Se a construção das rotas fosse feita em paralelo, talvez todas elas tivessem hipótese de se iniciar no instante zero com boas soluções.

4.3.3 Comparação de abordagens

Relativamente às duas abordagens aplicadas, a abordagem por *clusters* é, claramente, a que apresenta os melhores resultados para a instância em questão, conseguindo visitar todos os vértices e, consequentemente, recolhendo maior prémio comparativamente com a abordagem da heurística construtiva, como indica nos gráficos ilustrados nas figuras 71 e 72. No entanto, tal só é possível porque as instâncias se tornaram em instâncias suficientemente fáceis e o NEOS Server conseguiu solucionar o problema. No entanto, se a procura aumentar significativamente, ou seja, se o número de vértices aumentar o NEOS Server deixará ter capacidade computacional e não conseguirá apresentar uma resposta. Veja-se que para instâncias com 50 vértices, mesmo recorrendo a 5 veículos não foram obtidas soluções ótimas. Nessa altura, a heurística construtiva será uma melhor opção, já que é melhor obter uma solução que não seja ótima, do que não obter solução nenhuma.

Poder-se-ia ainda melhorar a heurística se, por exemplo, se aplicasse a heurística depois da divisão por *clusters*, o que provavelmente implicaria uma diminuição considerável nos quilómetros percorridos pelas rotas.

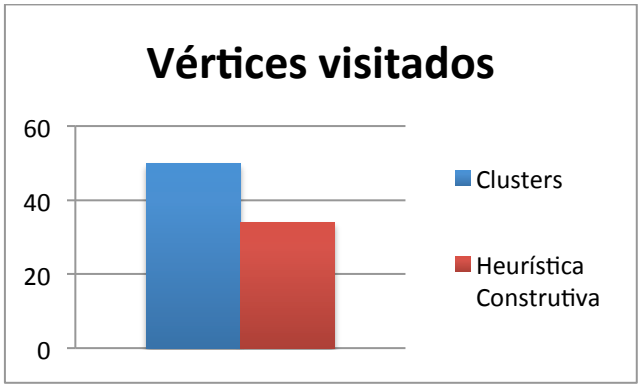


Figura 71 Vértices visitados pelas duas abordagens

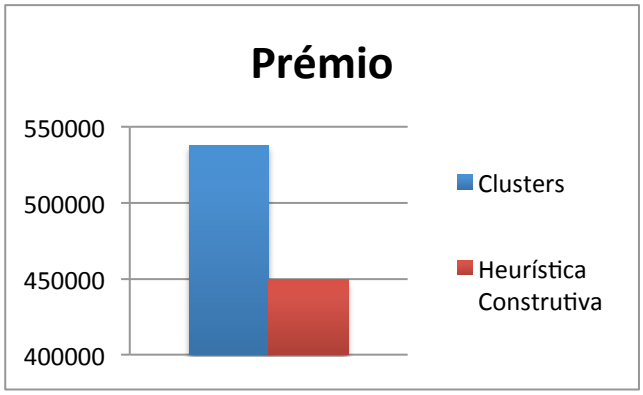


Figura 72 Prémio recolhido pelas duas abordagens

A Figura 73 mostra os índices de utilização das duas abordagens.

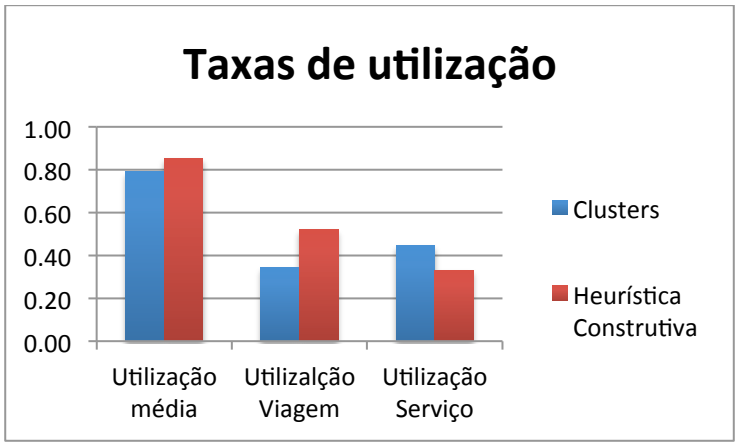


Figura 73 Taxas de utilização pelas duas abordagens

Devido à divisão por zona geográfica, a abordagem por *cluster*s apresenta taxas de utilização mais eficientes do que a abordagem da heurística construtiva. Embora a taxa de utilização média da apresentada pela heurística construtiva seja mais alta, esta é bastante influenciada pela elevada taxa de utilização em viagens, 52%. Já na abordagem por *clusters* este indicador é bastante mais reduzido, 34%. A taxa de utilização é, sem surpresa, mais elevado na abordagem por *clusters*, uma vez que com esta abordagem fica apenas um vértice por visitar, enquanto que com heurística construtiva ficam 16 vértices por visitar. As taxas de utilização em serviço são de 45% e 33% para as abordagens por *clusters* e heurística construtiva, respetivamente.

Poderia-se-ia desenvolver uma nova heurística construtiva utilizando a inserção de menor custo, tendo em conta a proximidade dos vértices, evitando assim que se fizessem grandes deslocações.

5. CONCLUSÕES E TRABALHO FUTURO

5.1 Conclusão

Nesta dissertação fez-se o estudo do comportamento dos modelos baseados no problema de orientação de equipas aplicados a um problema real de escalonamento de clientes aos recursos de engarrafamento móvel de uma empresa que opera em Portugal.

Apresentaram-se conceitos importantes sobre a gestão da cadeia de abastecimento, bem como uma definição válida da mesma. É indispensável compreender estes conceitos para maximizar o grau de satisfação de um cliente com o produto final. O grau de satisfação do cliente é um excelente indicador de coordenação de fluxos de informação e materiais ao longo de toda a cadeia de abastecimento.

A revisão da literatura realizada no âmbito deste trabalho incidiu sobre o atual estado da arte dos problemas de orientação. Este estudo incidiu essencialmente sobre os problemas de orientação, problemas de orientação com janelas temporais, problemas de orientação de equipas e problemas de orientação de equipas com janelas temporais.

Na descrição do problema real apresentaram-se os serviços e os recursos disponíveis da empresa e decidiu-se focar o estudo no serviço de engarrafamento móvel, por questão de dimensão do problema e por se entender ser o serviço mais relevante para a empresa.

De acordo com a modelação realizada, desenvolveu-se um modelo matemático baseado em programação linear inteira mista, baseado no problema de orientação de equipas com janelas temporais. A escolha recaiu sobre este modelo por se entender ser o indicado para este problema, dadas as restrições de disponibilidade dos clientes e porque, devido à grande afluência, o modelo seria forçado a escolher os melhores clientes. O modelo matemático desenvolvido foi implementado em AMPL e submetido ao NEOS Server. Desenvolveu-se o modelo em AMPL por ser uma linguagem relativamente simples e intuitiva, mas também por ser muito semelhante à notação matemática. O NEOS Server é um projeto disponibilizado pela universidade de Wisconsin Madison que disponibiliza gratuitamente serviços com o intuito de resolver problemas de otimização de grande dimensão. Para além da existir uma grande comunidade a trabalhar com o NEOS Server, os servidores têm uma

capacidade de processamento, muito superior à de um computador pessoal, e podem ser enviados vários pedidos em simultâneo.

Foi criada uma aplicação web para a gestão dos clientes da empresa, em concreto para dar apoio ao planeamento da prestação do serviço aos clientes. A aplicação é desenvolvida numa plataforma JavaScript (*Nodejs)*. Nodejs representa o desenvolvimento da aplicação do lado do servidor, e propício para fazer aplicações web e com grande capacidade de escalabilidade, dado a sua metodologia. Recorreu-se a uma base de dados orientada a documentos (*mongoDB)*, que inicialmente não requerer um modelo conceptual de base de dados e a notação é muito semelhante à notação de JavaScript. Através da aplicação web obtêm-se as distâncias reais entre as localizações dos clientes, previamente geradas com recurso a uma biblioteca JavaScript (*Chance*.js) Chance.js é uma biblioteca para gerar dados aleatórios, dados como nome, email, telefone, moradas, coordenadas e entendeu-se que faria sentido popular devidamente a base de dados para se trabalhar com informação semelhante à real.

Na interação com o NEOS Server Gurobi optou-se pela utilização de um cliente *rpc* (remote procedure call), disponibilizado pelo NEOS Server. Neste procedimento cria-se um ficheiro *xml* com o modelo AMPL, a informação referente aos clientes e a matriz de distâncias e envia-se esta informação ao NEOS Server Gurobi via linha de comandos. A resposta é guardada no computador que invocou a rpc, num ficheiro de texto.

Dada a dimensão do ficheiro gerado pelo NEOS Server, desenvolveram-se ferramentas de linha de comandos, que juntamente com expressões regulares permitem extrair, automaticamente, a informação pertinente da resposta guardada no ficheiro de texto. Esta informação depois de extraída, é colocada num ficheiro Excel para criação de gráficos e tabelas com os indicadores mais importantes para se proceder à analise dos resultados.

Foram criadas 18 instâncias com diferentes números de vértices. E para cada uma dessas instâncias, aplicou-se o modelo TOPTW desenvolvido com um número diferente de veículos, a variar entre 1 e 5 veículos, obtendo-se assim o resultado para 90 instâncias diferentes. A análise de resultados permitiu verificar que o modelo baseado no problema de orientação de equipas com janelas temporais (TOPTW) sendo um problema NP-difícil enfrenta dificuldades na solução de instâncias de grande dimensão. A natureza combinatória que resulta da seleção de clientes aliada à dificuldade da definição das rotas faz com que o recurso ao NEOS Server não seja bem sucedida quando as instâncias são de grande dimensão, sendo os processos interrompidos pelo NEOS Server devido à limitação das 8 horas de processamento ou 3GB de memória.

Quando se trata de instâncias fáceis, isto é, quando a solução visita todos os vértices, verificou-se que o modelo não tem em conta as distâncias percorridas, e por vezes apresenta soluções com elevado número de quilómetros percorridos.

Tentaram-se abordagens diferentes com vista a obter respostas quando o NEOS Server não as consegue dar. Tentou-se uma abordagem com base em VRP, onde se alterou a função objetivo do modelo estudado para este ter em conta as distâncias percorridas. Na instância comparativa, houve uma poupança em quilómetros percorridos superior a 50%, embora as instâncias para quais não se conseguia resposta, continuem sem resposta. Utilizou-se uma abordagem que visa à criação de *clusters*, dividindo a instância em três zonas geográficas, (Norte Centro e Sul), e aplicou-se o modelo TOPTW a cada uma delas. Como o problema é dividido em problemas mais pequenos, é mais fácil a obtenção de resposta. Com esta abordagem já é possível obter resposta mesmo para a instância de 50 vértices, para a qual não era possível até então, no entanto se a procura aumentar, o problema continuará sem resposta devido ao elevado esforço computacional e consequente rejeição do NEOS Server. Criou-se ainda uma heurística construtiva na tentativa de obter resposta mesmo para instâncias maiores e, caso a instância aumente, a heurística consegue dar resposta. Os resultados obtidos poderiam ser melhorados se a construção das rotas fosse feita em série, ou se fossem aplicados métodos de inserção de menor custo tendo em conta as distâncias entre os clientes.

Desenvolveu-se uma aplicação web preparada para gerir os clientes, os serviços e as rotas a efetuar pelos diferentes recursos da empresa. A aplicação foi parcialmente concebida, e foi muito importante no desenvolvimento da dissertação, nomeadamente na geração das diferentes instâncias.
Foi bastante empolgante trabalhar com tecnologias recentes, como é o caso do Node.js e mongoDB, embora se tenha noção que, por causa de se ter optado por estas tecnologias, se tenha consumido mais tempo na aprendizagem. Também foi cativante a utilização da API JavaScript do Google Maps, bem como o recurso à *rpc* através linha de comandos, permitindo uma diferente abordagem à forma como se interage com o NEOS Server.

5.2 Trabalho futuro

A aplicação web pode ser aumentada, pelo que seria interessante incluir mais funcionalidades. A aplicação está desenvolvida para uma empresa de engarrafamento móvel, mas pode facilmente ser adaptada para uma outra área em que exista a necessidade de prestar serviços ao cliente com disponibilidades previamente conhecidas. A aplicação pode ainda ser utilizada para gerar instâncias

para outros projetos, com a matriz de distâncias preenchida com distâncias reais, basta para isso conhecer as localizações a colocar.

6. BIBLIOGRAFIA

[1] M. E. Transportes, "A – Enquadramento dos setores e potencial estratégico da I & D," 2014.

[2] P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden, "The orienteering problem: A survey," *Eur. J. Oper. Res.*, vol. 209, no. 1, pp. 1–10, Feb. 2011.

[3] J. Ferreira, J. A. Oliveira, G. A. B. Pereira, L. Dias, F. Vieira, J. Macedo, T. Carção, T. Leite, and D. Murta, "Developing tools for the team orienteering problem A simple genetic algorithm," no. 1996, pp. 134–140, 2006.

[4] A. Harrison and R. Van Hoek, "Logistics Management and Strategy.pdf." p. 51, 2008.

[5] S. E. Butt and T. M. Cavalier, "A heuristic for the multiple tour maximum collection problem," *Comput. Oper. Res.*, vol. 21, no. 1, pp. 101–111, Jan. 1994.

[6] I.-M. Chao, B. L. Golden, and E. a. Wasil, "The team orienteering problem," *Eur. J. Oper. Res.*, vol. 88, no. 3, pp. 464–474, Feb. 1996.

[7] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Nav. Res. Logist.*, vol. 34, no. 3, pp. 307–318, Jun. 1987.

[8] G. Laporte and S. Martello, "The selective travelling salesman problem," *Discret. Appl. Math.*, vol. 26, no. 2–3, pp. 193–207, Mar. 1990.

[9] R. Ramesh, Y.-S. Yoon, and M. H. Karwan, "An Optimal Algorithm for the Orienteering Tour Problem," *ORSA J. Comput.*, vol. 4, no. 2, pp. 155–165, May 1992.

[10] A. C. Leifer and M. B. Rosenwein, "Strong linear programming relaxations for the orienteering problem," *Eur. J. Oper. Res.*, vol. 73, no. 3, pp. 517–523, Mar. 1994.

[11] M. Fischetti, Juan José Salazar González, and P. Toth, "Solving the Orienteering Problem through Branch-and-Cut." 1998.

[12] M. Gendreau, G. Laporte, and F. Semet, "A branch-and-cut algorithm for the undirected selective traveling salesman problem," *Networks*, vol. 32, no. 4, pp. 263–273, Dec. 1998.

[13] Theodore Tsiligiridis, "Heuristic Methods Applied to Orienteering," *Palgrave Macmillan Journals*, 1984. [Online]. Available: http://www.inf.unibz.it/dis/teaching/SDB/papers/batch3a.pdf. [Accessed: 15-Apr-2014].

[14] A. Wren, A. Holliday, and A. Hollidayt, "Computer Scheduling of Vehicles from One or More Depots to a Number of Delivery Points," vol. 23, no. 3, pp. 333–344, 2014.

[15] I.-M. Chao, B. L. Golden, and E. a. Wasil, “Theory and Methodology - The team orienteering problem,” *Eur. J. Oper. Res.*, vol. 88, no. 3, pp. 464–474, Feb. 1996.

[16] S. Lin, “Computer Solutions to the Travelling Salesman Problem.” 1965.

[17] R. Ramesh and K. M. Brown, “An efficient four-phase heuristic for the generalized orienteering problem,” *Comput. Oper. Res.*, vol. 18, no. 2, pp. 151–165, Jan. 1991.

[18] I.-M. Chao, B. L. Golden, and E. a. Wasil, “Theory and Methodology A fast and effective heuristic for the orienteering problem,” vol. 2217, no. 95, 1996.

[19] M. Gendreau, G. Laporte, and F. Semet, “A tabu search heuristic for the undirected selective travelling salesman problem,” *Eur. J. Oper. Res.*, vol. 106, no. 2–3, pp. 539–545, Apr. 1998.

[20] W. Souffriau, P. Vansteenwegen, J. Vertommen, G. Vanden Berghe, and D. Van Oudheusden, “a Personalized Tourist Trip Design Algorithm for Mobile Tourist Guides,” *Appl. Artif. Intell.*, vol. 22, no. 10, pp. 964–985, Oct. 2008.

[21] P. Vansteenwegen and D. Van Oudheusden, “The Mobile Tourist Guide: An OR Opportunity,” *OR Insight*, vol. 20, no. 3, pp. 21–27, Jul. 2007.

[22] S. E. Butt and D. M. Ryan, “An optimal solution procedure for the multiple tour maximum collection problem using column generation,” *Comput. Oper. Res.*, vol. 26, no. 4, pp. 427–441, Apr. 1999.

[23] S. Boussier, D. Feillet, and M. Gendreau, “An exact algorithm for team orienteering problems,” *4or*, vol. 5, no. 3, pp. 211–230, Jul. 2006.

[24] W. D. Harvey and M. L. Ginsberg, “Limited Discrepancy Search,” *Morgan Kaufmann*, pp. 607 – 613, 1995.

[25] I.-M. Chao, B. L. Golden, and E. A. Wasil, “Theory and Methodology A fast and effective heuristic for the orienteering problem,” vol. 2217, no. 95, 1996.

[26] H. Tang and E. Miller-Hooks, “A TABU search heuristic for the team orienteering problem,” *Comput. Oper. Res.*, vol. 32, no. 6, pp. 1379–1407, Jun. 2005.

[27] C. Archetti, A. Hertz, and M. G. Speranza, “Metaheuristics for the team orienteering problem,” *J. Heuristics*, vol. 13, no. 1, pp. 49–76, Dec. 2007.

[28] L. Ke, C. Archetti, and Z. Feng, “Ants can solve the team orienteering problem,” *Comput. Ind. Eng.*, vol. 54, no. 3, pp. 648–665, Apr. 2008.

[29] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden, “A guided local search metaheuristic for the team orienteering problem,” *Eur. J. Oper. Res.*, vol. 196, no. 1, pp. 118–127, Jul. 2009.

[30] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden, “Metaheuristics for tourist trip planning.” 2009.

[31] W. Souffriau, P. Vansteenwegen, G. Vanden Berghe, and D. Van Oudheusden, “A Path Relinking approach for the Team Orienteering Problem,” *Comput. Oper. Res.*, vol. 37, no. 11, pp. 1853–1859, Nov. 2010.

[32] S. E. Butt and T. M. Cavalier, “A heuristic for the multiple tour maximum collection problem,” *Comput. Oper. Res.*, vol. 21, no. 1, pp. 101–111, Jan. 1994.

[33] M. G. Kantor and M. B. Rosenwein, “The Orienteering Problem with Time Windows,” vol. 43, no. 6, pp. 629–635, 2012.

[34] G. Righini and M. Salani, “Decremental state space relaxation strategies and initialization heuristics for solving the Orienteering Problem with Time Windows with dynamic programming,” *Comput. Oper. Res.*, vol. 36, no. 4, pp. 1191–1203, Apr. 2009.

[35] G. Righini and M. Salani, “New Dynamic Programming Algorithms for the Resource Constrained Elementary Shortest Path Problem,” 2008.

[36] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden, “Iterated local search for the team orienteering problem with time windows,” *Comput. Oper. Res.*, vol. 36, no. 12, pp. 3281–3290, Dec. 2009.

[37] R. Montemanni and L. M. Gambardella, “AN ANT COLONY SYSTEM FOR TEAM ORIENTEERING PROBLEMS WITH TIME WINDOWS,” *Found. Comput. Decis. Sci.*, 2009.

[38] R. Mansini, M. Pelizzari, and R. Wolfer, “A granular variable neighbourhood search heuristic for the tour orienteering problem with time windows,” 2006.

[39] F. Tricoire, M. Romauch, K. F. Doerner, and R. F. Hartl, “Heuristics for the multi-period orienteering problem with multiple time windows,” *Comput. Oper. Res.*, vol. 37, no. 2, pp. 351–367, Feb. 2010.

[40] J. Mota, Gabriel and Abreu, Mário and Quintas, Artur and Ferreira, João and Dias, LuisS. and Pereira, GuilhermeA.B. and Oliveira, “A Genetic Algorithm for the TOPdTW at Operating Rooms,” *Comput. Sci. Its Appl. – ICCSA 2013*, vol. 7971, pp. 304–317, 2013.

[41] J. A. Oliveira, G. Mota, J. Ferreira, M. Figueiredo, L. Dias, and G. Pereira, “A DECISION SUPPORT SYSTEM FOR WASTE COLLECTION MODELED AS TOPTW VARIANT,” 2013.

[42] D. Feillet, P. Dejax, and M. Gendreau, “Traveling Salesman Problems with Profits,” *Transp. Sci.*, vol. 39, no. 2, pp. 188–205, May 2005.

[43] J.-F. Bérubé, M. Gendreau, and J.-Y. Potvin, “An exact -constraint method for bi-objective combinatorial optimization problems: Application to the Traveling Salesman Problem with Profits,” *Eur. J. Oper. Res.*, vol. 194, no. 1, pp. 39–50, Apr. 2009.

[44] J. F. Pekny and D. L. Miller, “An Exact Parallel Algorithm for the Resource Constrained Traveling Salesman Problem with Application to Scheduling with an Aggregate Deadline,” in *Proceedings of the 1990 ACM Annual Conference on Cooperation*, 1990, pp. 208–214.

[45] Q. Wang, X. Sun, B. L. Golden, and J. Jia, “Using artificial neural networks to solve the orienteering problem,” *Ann. Oper. Res.*, vol. 61, no. 1, pp. 111–120, Dec. 1995.

[46] Z. W. Geem, C. Tseng, and Y. Park, “Harmony Search for Generalized Orienteering Problem : Best Touring in China,” pp. 741–750, 2005.

[47] X. Wang, B. L. Golden, and E. A. Wasil, “Using a Genetic Algorithm to Solve the Generalized Orienteering Problem,” vol. 43, 2008.

[48] M. Schilde, K. F. Doerner, R. F. Hartl, and G. Kiechle, “Metaheuristics for the bi-objective orienteering problem,” *Swarm Intell.*, vol. 3, no. 3, pp. 179–201, May 2009.

[49] T. İlhan, S. M. R. Iravani, and M. S. Daskin, “The orienteering problem with stochastic profits,” *IIE Trans.*, vol. 40, no. 4, pp. 406–421, Feb. 2008.

[50] F. V Fomin and A. Lingas, “Approximation Algorithms for Time-Dependent Orienteering,” pp. 508–515, 2001.

[51] C. Archetti, D. Feillet, a Hertz, and M. G. Speranza, “The capacitated team orienteering and profitable tour problems,” *J. Oper. Res. Soc.*, vol. 60, no. 6, pp. 831–842, May 2008.

[52] M. Gendreau, M. Labb, and G. Laporte, “Efficient heuristics for the design of ring networks,” vol. 4, pp. 177–188, 1995.

[53] D. Feillet, P. Dejax, and M. Gendreau, “The Profitable Arc Tour Problem: Solution with a Branch-and-Price Algorithm,” *Transp. Sci.*, vol. 39, no. 4, pp. 539–552, 2005.

[54] J. Aráoz, E. Fernández, and O. Meza, “Solving the Prize-collecting Rural Postman Problem,” *Eur. J. Oper. Res.*, vol. 196, no. 3, pp. 886–896, Aug. 2009.

[55] C. Archetti, D. Feillet, A. Hertz, and M. G. Speranza, “The undirected capacitated arc routing problem with profits,” *Comput. Oper. Res.*, vol. 37, no. 11, pp. 1860–1869, Nov. 2010.

[56] Z. Li and X. Hu, “The Team Orienteering Problem with Capacity Constraint and Time Window The Team Orienteering Problem with Capacity Con- straint and Time Window,” no. Isora, pp. 157–163, 2011.

[57] M. Van Der Merwe, J. P. Minas, M. Ozlen, and J. W. Hearne, “The cooperative orienteering problem with time windows,” 2014.

[58] B. F. Lóscio, H. R. De Oliveira, J. César, and D. S. Pontes, “NoSQL no desenvolvimento de aplicações Web colaborativas,” vol. d.

7. Anexo I - Ficheiro xml com modelo a enviar ao NEOS Server via *RPC*

```
<document>
<category>milp</category>
<solver>Gurobi</solver>
<inputType>AMPL</inputType>
<priority>long</priority>
<email>projectmes2014@gmail.com</email>
<model><![CDATA[
param n;
param carros;
param premio{i in 1..n};
param distancias{i in 1..n, j in 1..n};
param tproc{i in 1..n};
param pontos{i in 1..n, j in 1..5};
param ji{i in 1..n};
param js{i in 1..n};
var x{c in 1..carros, i in 1..n-1, j in 2..n : i!=j} binary;
var t{i in 1..n} >=0;

maximize profit: sum{c in 1..carros, i in 1..n-1, j in 2..n : i!=j} x[c,i,j]*premio[j];

subject to inicio {c in 1..carros}: sum{j in 2..n} x[c,1,j] = 1;
subject to fluxo {c in 1..carros, j in 2..n-1}: sum{i in 1..n-1 : i!=j} x[c,i,j] = sum{i in 2..n : i!=j}
x[c,j,i];
subject to fim {c in 1..carros}: sum{i in 2..n-1} x[c,i,n] = 1;
subject to nodo {j in 2..n-1} : sum{c in 1..carros, i in 1..n-1 : i!=j} x[c,i,j] <= 1;
subject to voltar {c in 1..carros, i in 2..n-1, j in 2..n-1 : i!=j}: x[c,i,j] + x[c,j,i] <= 1;
subject to tempini {i in 1..n}: t[1] = 0;
subject to tempo {c in 1..carros, i in 1..n-1, j in 2..n : i!=j}: t[j] >= t[i] +tproc[i] +
distancias[i,j]*x[c,i,j] - (10000 * (1- x[c,i,j]));
subject to twjs {i in 2..n}: t[i] <= js[i] - tproc[i];
subject to twji {i in 2..n-1}: t[i] >= ji[i] ; ]]></model>

<data><![CDATA[
param n := 25;
param carros := 3;
param pontos : 1 2 3 4 5:=
1  0    0   90  1 0
2  3262 27  90  2 3
3  2214 18  72  1 3
4  6741 18  81  2 2
5  9981 36  63  1 2
6  4387 27  90  2 2
7  3276 18  63  3 2
8  3861 45  63  2 2
9  9425 54  90  3 2
10 9091 27  81  3 2
11 4844 27  63  1 2
12 6563 27  72  3 3
13 5309 45  72  3 3
14 7691 54  81  1 2
15 6152 27  72  3 2
16 2442 36  72  2 3
17 2599 45  81  2 3
18 2130 54  90  3 3
19 3110 9   63  1 2
20 4923 27  90  3 2
21 3101 9   81  1 2
22 4199 9   81  3 3
23 2890 54  90  1 3
24 2812 0   90  2 2
25 0    0   90  3 0;

param distancias : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25:=
1  0 2.275 9.375 1.125 4.1 12.7  3.85  15.9  11.975  2.675 7.55  2.075 10.7  13.075  15.5  12.225  10.3  18.3
15.7  4.575 13.55 3.825 4.35  11.975  0
2  2.275 0 7.525 2.75  4.325 10.85 4.075 14.075  10.15 0.375 5.7 0.575 8.85  11.25 13.65 10.4  8.475 16.45
13.85 2.725 11.7  4.05  4.175 10.15 2.275
```

```
3  9.35  7.5 0 9.825 5.55  5 5.825 6.725 3.375 7.55  2.525 7.55  2.825 2.925 7.8 3.55  1.4 7.875 8 5.1 5.85
6.25  5.425 3.375 9.35
4  1.125 2.75  9.85  0 4.8 13.175  4.55  16.4  12.7  3.15  8.025 2.55  11.2  13.575  15.975  12.725  10.8
18.8  16.175  5.05  14.025  4.525 5.075 12.7  1.125
5  4.125 4.35  5.575 4.65  0 8.875 0.4 12.1  8.175 4.375 3.75  4.375 6.9 9.275 11.7  8.425 6.5 14.5  11.875
2.4 9.75  0.6 0.275 8.2 4.125
6  12.575  10.725  5 13.05 8.775 0 9.05  3.925 4.8 10.775  5.425 10.75 2.475 3.45  3.525 1.05  4.25  6.325
3.725 8.3 1.425 9.25  8.65  4.3 12.575
7  3.9 4.1 6.4 4.4 0.4 9.175 0 12.375  8.825 4.15  4.025 4.125 7.175 9.55  11.975  8.725 6.775 14.775  12.175
2.175 10.025  0.35  0.65  8.85  3.9
8  15.75 13.925  6.725 16.25 11.95 3.925 12.225  0 4.175 13.975  8.625 13.95 4.975 3.85  1.4 4.6 5.5 2.025
3.225 11.5  2.725 12.45 11.85 4.1 15.75
9  12  10.125  3.4 12.5  8.15  4.8 8.825 4.2 0 10.175  5.85  10.15 2.775 1.175 4.75  3.25  1.45  5.35  5.275
7.7 4.1 8.675 8.05  0.075 12
10 2.675 0.375 7.4 3.15  4.2 10.725  3.95  13.925  10.025  0 5.575 1.15  8.725 11.1  13.525  10.275  8.35
16.325  13.725  2.6 11.575  3.925 4.05  10.025  2.675
11 7.525 5.7 2.525 8.025 3.725 5.55  4 8.75  5.85  5.75  0 5.725 3.55  5.925 8.35  5.1 3.475 11.15 8.55
3.275 6.4 4.225 3.625 5.85  7.525
12 2.075 0.575 7.55  2.55  4.35  10.875  4.1 14.075  10.175  1.125 5.725 0 8.875 11.275  13.675  10.425  8.5
16.475  13.875  2.75  11.725  4.075 4.2 10.175  2.075
13 10.675  8.825 2.825 11.15 6.875 2.475 7.15  4.975 2.775 8.875 3.525 8.85  0 1.425 5.275 1.375 2.225 6.125
5.475 6.4 3.325 7.35  6.75  2.3 10.675
14 12.925  11.075  2.95  13.4  9.125 3.45  9.4 3.85  1.2 11.125  5.775 11.125  1.425 0 4.4 1.9 1.725 5.025
4.925 8.675 4.3 9.6 9 1.1 12.925
15 15.375  13.525  7.8 15.85 11.575  3.525 11.85 1.4 4.75  13.575  8.225 13.55 5.275 4.4 0 4.2 7.05  1.375
2.025 11.1  2.325 12.05 11.45 4.65  15.375
16 12.1  10.275  3.55  12.575  8.3 1.05  8.575 4.6 3.25  10.3  4.975 10.3  1.375 1.775 4.2 0 2.625 7 4.375
7.85  2.25  8.8 8.2 2.65  12.1
17 10.275  8.45  1.4 10.775  6.475 4.25  6.75  5.5 1.45  8.5 3.475 8.475 2.25  1.725 7.05  2.625 0 6.675 7.25
6.025 5.1 6.975 6.375 1.475 10.275
18 18.15 16.325  7.9 18.65 14.35 6.325 14.625  2.025 5.35  16.35 11.025  16.35 8.075 5.025 1.375 7 6.675 0
3.125 13.9  5.175 14.85 14.25 5.275 18.15
19 15.575  13.725  7.975 16.05 11.75 3.725 12.025  3.225 5.25  13.775  8.425 13.75 5.475 4.925 2.025 4.4 7.25
3.15  0 11.3  2.2 12.25 11.65 5.175 15.575
20 4.55  2.7 5.2 5.025 2.35  8.5 2.1 11.725  7.8 2.75  3.375 2.725 6.525 8.9 11.325  8.05  6.125 14.125  11.5
0 9.35  2.075 2.2 7.825 4.55
21 13.425 11.575  5.85  13.9  9.6 1.425 9.9 2.75  5.65  11.625  6.275 11.6  3.325 4.3 2.325 2.25  5.1 3.9 2.2
9.15  0 10.1  9.5 4 13.425
22 3.825  4.025 6.225 4.35  0.6 9.375 0.35  12.575  8.65  4.075 4.225 4.075 7.375 9.75  12.175  8.925 7
14.975  12.375  2.1 10.225  0 0.85  8.65  3.825
23 4.375  4.125 5.45  4.9 0.275 8.775 0.65  12  8.075 4.175 3.65  4.15  6.8 9.175 11.575  8.325 6.4 14.4
11.775  2.2 9.625 0.875 0 8.1 4.375
24 12 10.125  3.4 12.525  8.175 4.375 8.825 4.125 0.075 10.175  5.85  10.15 2.375 1.1 4.675 2.85  1.475 5.275
5.2 7.7 4 8.7 8.05  0 12
25 0  2.275 9.375 1.125 4.1 12.7  3.85  15.9  11.975  2.675 7.55  2.075 10.7  13.075  15.5  12.225  10.3
18.3  15.7  4.575 13.55 3.825 4.35  11.975  0;
]]></data>

<commands><![CDATA[for{k in 1..n}{
  let tproc[k] := (pontos[k,1]/0.75)/2500 + (pontos[k, 5]*.33);
  let premio[k] := pontos[k,1] * pontos[k,4];
 let ji[k] := pontos[k,2];
 let js[k] := pontos[k,3];
}

solve;
display _varname, _var;]]></commands>

<comments><![CDATA[Instancia 25-3]]></comments>

</document>
```

8. Anexo II - Script desenvolvido para devolver ordem de execução

```
exports.nodes = function(v, nodes, a, s){

  indexAux = 0;
  index = 0;
  n = a[0];
  aux = a;
  res = [  ];
  stringRes = "";

  console.log(s);

  while(n != nodes && v){

    res.push(n)
    console.log(n);

    if (n>aux[index+1]){
        index = aux.indexOf(aux[index+1]);
    }

    else{
        index = aux.lastIndexOf(aux[index+1]);
    }

    n = aux[index];

    if (n == nodes){
      console.log(nodes);
      aux = aux.slice(1,a.length);
      aux = aux.slice(aux.indexOf(1),a.length);
      index = a.indexOf(1);
      n = a[index];
      v--;
    }
  }
}
```