Nuno Miguel Pinto Cunha

Licenciado em Engenharia Informática

Optimização do transporte de doentes num serviço porta à porta

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática

Orientador: Francisco de Moura e Castro Ascensão de Azevedo, Professor Auxiliar, Universidade Nova de Lisboa

Coorientadora: Maria Isabel Azevedo Rodrigues Gomes, Professor Auxiliar, Universidade Nova de Lisboa

Júri:

Presidente: [Nome do presidente do júri]

Arguente: Professora Doutora Paula Alexandra Amaral

Vogal: Professor. Doutor Francisco Azevedo

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2012

Optimização do transporte de doentes num serviço porta à porta

“Qualquer homem pode alcançar o êxito, se dirigir seus pensamentos numa direcção e insistir neles até que aconteça alguma coisa”

Thomas Edison

“Seek opportunity, not security. A boat in harbor is safe, but in time its bottom will rot out.”

H. Jackson Brown Jr.

“O único homem que está isento de erros, é aquele que não arrisca acertar.”

Albert Einstein

“Suor derramado no treino é sangue poupado em batalha…”

Dale Carnegie

“If we knew what it was we were doing, it would not be called research, would it?”

Albert Einstein

Agradecimentos

Uma dissertação de mestrado é um longo caminho, recheado de desafios e incertezas que obrigam a um esforço e dedicação acrescidos, que aliados a contribuições de diversas pessoas criam o tónico necessário para a sua concepção. Deste modo o espaço que se segue é dedicado a todas as pessoas que contribuíram para que esta dissertação fosse realizada, através da sua colaboração, apoio, partilha de conhecimento e incentivo à perseverança.

Em primeiro lugar, agradeço ao meu orientador Professor Doutor Francisco Azevedo pela sua orientação, pautada por um elevado conhecimento e por uma disponibilidade e preocupação constantes. Gostaria ainda de lhe agradecer toda a paciência demonstrada na fase de reaprendizagem da linguagem Prolog e das suas especificidades, bem como todos os momentos em que deliberadamente me incutiu a pressão necessária para que este trabalho pudesse ser levado a bom porto.

Não menos importante, à minha coorientadora Professora Doutora Isabel Salema Gomes estou especialmente grato pelo seu elevado conhecimento na área dos *Vehicle Routing Problems*, em particular, do *Dial-a-Ride Problem*, e pela ligação estabelecida entre a Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa e a delegação Amadora – Sintra da Cruz Vermelha Portuguesa. Agradeço também a total disponibilidade, a motivação que me soube transmitir, a perseverança e a fantástica capacidade de conseguir sempre retirar os aspectos positivos das situações, mesmo que estas aparentem ser péssimas… Por todas as conversas, trocas de ideias e horas gastas em leituras, resumos e correcções, o meu muito obrigado!

À Professora Doutora Fernanda Barbosa e ao Professor Doutor Jorge Cruz agradeço o aval positivo que permitiu a continuidade deste trabalho e, acima de tudo, o interesse

demonstrado no mesmo. Reitero os agradecimentos aos conselhos, ideias e críticas relativas à estrutura do documento e ao modelo de Programação com Restrições do *Dial-a-Ride Problem*.

À Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa expresso todo o respeito e gratidão por todas as oportunidades que me proporcionou, bem como pela formação de excelência ministrada. Neste âmbito apresento os meus mais sinceros agradecimentos ao Departamento de Informática pela oportunidade que me foi concedida de contribuir para este *muy* nobre estabelecimento de ensino através da partilha do conhecimento que tenho vindo a adquirir ao longo destes últimos anos, na qualidade de Monitor.

À anTUNiA – Tuna de Ciências e Tecnologia da Universidade de Lisboa – gostaria de agradecer as oportunidades que me concedeu ao longo destes últimos 5 anos. As diferentes experiências, as aventuras, as responsabilidades, a praxe, o sacrifício, as viagens, os palcos… Mas acima de tudo, as valiosas lições de vida que levarei sempre comigo. A todos os anTUNiOS, os meus mais sinceros agradecimentos pela continuidade desta instituição que tive a honra de representar e que espero tê-lo feito condignamente.

Não poderia não deixar um grande e sincero agradecimento a todos os formadores do Curso de Liderança da Academia Militar que me proporcionaram uma formação impar, tornando esta experiência um marco importantíssimo na minha vida e na elaboração desta dissertação. Aos meus dezassete camaradas que faço questão de referir, um a um: Ana Brazão, António Gomes, Bárbara Oliveira, Carlos Carvalho, David Figueira, David Pereira, Eva Ferreira, Fábio Gutierres, José Santos, João Grilo, João Rato, João Santos, Marta Alves, Miguel Marques, Nelson Martins, Pedro Borges e Rui Almeida, agradeço profundamente por me fazerem acreditar novamente numa sociedade de valores onde a entreajuda, o sacrifício, o respeito, a humildade, a vontade de dar mais e melhor e a coragem fazem acontecer! Obrigado por me ajudarem a falhar, a errar e, deste modo, a aprender e a superar barreiras! Juntos superámos todos os desafios que nos colocaram (alguns nada fáceis…) e sem dúvida que superaríamos muito mais… Agora, sozinho, não será tão fácil, mas seria pior se não tivesse aprendido com todos vós! Foi uma honra e um prazer conhecer-vos e devo-vos a renovação de forças e a motivação necessárias para a fase final desta dissertação. Muito obrigado Equipa!

Às Forças Armadas Portuguesas expresso a minha gratidão e respeito pela total dedicação à pátria, pela competência, pelos valores e pelo exemplo.

Deixo também uma palavra de apreço e gratidão a todos os meus colegas que me acompanharam deste o início do curso e com quem tive a oportunidade de trabalhar, de

aprender e de partilhar as diversas batalhas com que me deparei ao longo da minha formação. A todos muito obrigado pelo companheirismo e pela entreajuda!

Por crer que a aprendizagem é um processo continuo, insaciável e que necessita de maturação para dar frutos, não podia deixar de agradecer a todos aqueles que considero terem sido os meus Professores pois, para além de uma excelência de conteúdos técnicos, proporcionaram-me lições de vida intemporais que me esforço por não esquecer. E porque esta dissertação também acaba por espelhar esses ensinamentos, deixo os meus sinceros e sentidos agradecimentos ao Professor André Gross, ao Professor Doutor Fernando Santana, ao Tenente Coronel Dias Rouco, ao Professor Doutor José Alferes, ao falecido Pe. Ivo do Vale, à Professora Lurdes Martins, ao Professor Doutor Nélson Chibeles Martins, ao Professor Nuno Ramos, ao Major Quinta, à Professora Rosário Ferreira e ao Professor Doutor Ruy Costa.

Por fim, mas não menos importante, gostaria de agradecer profundamente à minha família e à Marisa por terem estado sempre presentes, tentando acompanhar os desafios, as conquistas, as alegrias e as angústias inerentes a este trabalho. Espero um dia poder retribuir-vos…

Resumo

Com o envelhecimento da população nos países ditos desenvolvidos e o consequente aparecimento de diversas patologias, são cada vez mais as solicitações colocadas às instituições que fornecem serviços de transporte de e para as entidades de tratamento. Em Portugal, a maioria destas instituições são de natureza não-governamental e sem fins lucrativos. Dada a escassez de recursos de que dispõem, torna-se ainda mais difícil fazer face às solicitações que se lhes apresentam. Deste modo torna-se imperativo rentabilizar ao máximo os recursos disponíveis de modo a servir o maior número possível de pedidos de transporte, minimizando os custos operativos sem que a qualidade do serviço saia prejudicada.

Esta dissertação surge com o intuito de contribuir para a melhoria dos processos de criação de rotas de transportes de doentes da delegação Amadora – Sintra da Cruz Vermelha Portuguesa (CVP), de modo a que estes possam atender um maior número de pedidos de transporte com os recursos de que dispõem actualmente. Deste modo foram propostos dois modelos diferentes para o *Dial-a-Ride Problem* (DARP) utilizando uma metodologia de Programação com Restrições, com domínios distintos: grafos e finitos (inteiros), com o intuito de dar resposta a uma versão estática do DARP com janelas temporais, uma frota heterogénea e preocupações relativas à qualidade de serviço. Apesar do seu desempenho, o modelo de domínios finitos foi capaz de gerar soluções interessantes para a realidade da CVP.

Palavras-chave: Transporte porta à porta, Optimização, Multiobjectivo, Programação com Restrições; Cruz Vermelha Portuguesa

Abstract

With the aging of population from the designated “developed countries” and the resulting increase of new pathologies, the requirements imposed to transportation services to and from health treatment facilities are ever increasing. In Portugal, the majority of institutions which handles these transportation services are of non-governmental nature and thus with non-profitable services. With scarce resources, it is becoming more difficult to take care of all of the requested services. To address this issue, it is imperative to these institutions that all the available resources are carefully assigned to maximize the service’s availability satisfy all its transportation demand, whilst minimizing costs with no consequence in the quality of the original service.

This dissertation aims at contributing to the improvement of transportation routes for the transport of patients in the area of Amadora - Sintra of the Portuguese Red Cross. In this way the improvement would allow a larger number of patients to be transported with the current available resources. Thus, two different models have been proposed for the Dial-a-Ride Problem (DARP) using the methodology of Constraint Programming, with distinct domains: graphs and finite (integer). The purpose of these models is to respond to a static version of DARP, with time windows, a heterogeneous fleet and concerns about quality of service.

Despite its performance, the model of finite domains was able to generate interesting solutions to the CVP’s reality.

Keywords: Dial-a-Ride Problem, Optimization, Multi-Objective, Constraint Programming, Portuguese Red Cross

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

AG – Algoritmo Genético

AIS – Artificial Immune System

AMPL – A Mathematical Programming Language

ARC – Austrian Red Cross

ARS – Administração Regional de Saúde

CFFS – Copenhagen Fire-Fighting Service

CSP – Constraint Satisfaction Problem

CVP – Cruz Vermelha Portuguesa

DARP – Dial-a-Ride Problem

GAMS – General Algebraic Modeling System

GRASPER – *GRAph constraint Satisfaction Problem solver*

LNS – Large Neighbourhood Search

MPL – Mathematical Programming Language

SA – Simulated Annealing

VNS – Variable Neighborhood Search

VRP – Vehicle Routing Problem

PDP – Pickup and Delivery Problem

PDPTW – Pickup and Delivery Problem with Time Windows

PDVRP – Pickup and Delivery Vehicle Routing Problem

PLMO – Programação Linear Multi Objectivo

PSO – Particle Swarm Optimization

VRPTW – Vehicle Routing Problem with Time Windows

1

1. Introdução

1.1. Motivação

As preocupações com o aquecimento global, esgotamento de recursos naturais e com o impacto social causado pela congestão do trânsito e pela poluição gerada têm conduzido ao aumento dos custos das deslocações e, consequentemente, a uma mudança de mentalidades no público em geral no que diz respeito às operações de transporte, procurando melhorar a sua eficiência.

Ainda que a premissa seja válida para qualquer entidade que realize transportes, de pessoas ou de bens materiais, a verdade é que esta questão tem surgido recorrentemente nos países ocidentais, em particular na Europa e na América do Norte, devido ao fenómeno do envelhecimento da população que, por sua vez, tem conduzido a um aumento dos pedidos de transporte aos serviços ambulatórios e de saúde. Portugal tem sido também afecto por este fenómeno. Actualmente grande parte dos sistemas existentes não tem capacidade para dar resposta a todos os pedidos pois os custos de operação não lhes permite escalabilidade.

Assim, este número crescente de solicitações associado a um decréscimo de recursos, obriga que estes últimos sejam aproveitados o melhor possível. Pretende-se com esta dissertação investigar uma melhoria dos processos de definição das rotas de veículos dos transportes porta à porta efectuados por uma associação portuguesa, de forma a conseguir dar resposta ao maior número de pedidos, minimizando os custos totais do serviço sem que isso implique uma perda de qualidade do mesmo.

1.2. O Problema e Proposta de Solução

O problema de planeamento de rotas de veículos, onde se procura servir um conjunto de clientes com um custo mínimo, tem sido abordado por diferentes autores (Berbeglia et al., 2007; Berbeglia et al., 2010), dada a sua grande utilidade prática e interesse académico. Este tipo de problema dá pelo nome de *Dial-a-Ride Problem* (DARP) e diferencia-se dos restantes problemas de rotas de veículos (*Vehicle Routing Problems*) pela componente humana que lhe acresce questões relacionadas com a qualidade de serviço. Deste modo, além da minimização da distância total percorrida, directamente proporcional ao custo do serviço, que é apanágio dos *Vehicle Routing Problems* (VRP), terá também que ser considerada a maximização da qualidade de serviço que pode ser mensurável, por exemplo, de acordo com o tempo que os utentes passam a bordo das viaturas. Ao facto de ter que lidar com objectivos conflituosos acresce a natureza combinatória do problema que permite diversas soluções, algumas em certo ponto simétricas/equivalentes em termos do objectivo final.

Esta dissertação debruça-se sobre o problema enfrentado pela Cruz Vermelha Portuguesa (CVP), em particular, no que se refere aos transportes efectuados pela sua delegação Amadora – Sintra. Esta unidade disponibiliza o serviço de transporte de doentes e pessoas incapacitadas dos seus locais de residência para as entidades onde realizarão algum tipo de tratamento de saúde. Para tal dispõe de vários veículos (ambulâncias) de diferentes tipos e com diferentes características.

No sentido de resolver o problema em questão, recorrer-se-á à Programação com Restrições de modo a minimizar os custos de transporte juntamente com o "desconforto" sentido pelo utente. A qualidade dos modelos será aferida através da sua aplicação a instâncias de referência na área do *Dial-a-Ride Problem*, bem como a dados cedidos pela CVP para rotas de dias diferentes e da comparação dos resultados obtidos com os de uma solução implementada com Programação Linear Inteira Mista (Loureiro, 2010).

1.3. Contribuições

De entre as contribuições apresentadas pela dissertação "Optimização do transporte de doentes num serviço porta à porta", pode-se enumerar:

i. Análise e modelação do *Dial-a-Ride Problem*, um problema que tem vindo a ganhar cada vez mais relevância na comunidade científica dada crescente necessidade de transporte de doentes que se tem verificado na Europa e na

América do Norte. No âmbito da modelação há que destacar que os modelos propostos foram baseados em Programação com Restrições, um paradigma praticamente inexplorado no contexto do DARP.

ii. Em particular, a utilização de domínios de grafos, através do *framework* GRASPER (Viegas & Azevedo, 2007), numa das modelações propostas trata-se de uma inovação neste contexto devido à massificação da Programação Inteira na resolução do DARP e, consequentemente, dos domínios inteiros.

iii. Adicionalmente e motivados pelos resultados obtidos pelo modelo descrito acima, foi concebido um modelo de domínios finitos. Ainda que este tipo de domínio seja comum nas modelações do *Dial-a-Ride Problem* presentes na literatura, do conhecimento do autor desta dissertação, é o único modelado com recurso à Programação com Restrições que contempla diversas questões relativas à qualidade de serviço, nomeadamente tempos de espera a bordo e uma função objectivo que contempla objectivos conflituosos: minimização de custos *versus* maximização do conforto do cliente.

iv. Apesar do desempenho computacional não ser o mais satisfatório, o último modelo descrito foi capaz de gerar soluções diferentes das existentes no contexto da delegação Amadora – Sintra da Cruz Vermelha Portuguesa, obtidos por Loureiro (2010), pelo facto de contemplar factores relativos à qualidade de serviço.

v. Estudo da envolvência do *Dial-a-Ride Problem*, bem como do caso de estudo considerado, a delegação Amadora – Sintra da Cruz Vermelha Portuguesa, e integração de ambas as realidades.

1.4. Estrutura do Documento

Esta dissertação encontra-se dividida em nove capítulos, cada um dos quais referente a aspectos diferentes do trabalho realizado. No corrente capítulo é feita uma breve descrição do enquadramento do trabalho realizado e dos objectivos propostos.

No Capítulo 2 é realizada uma breve descrição da Cruz Vermelha Portuguesa, com foco nas características e problemáticas da delegação Amadora – Sintra relativamente ao serviço de transporte de doentes programados.

No Capítulo 3 é apresentado o *Dial-a-Ride Problem*, um problema de definição de rotas para veículos que se enquadra no contexto da Cruz Vermelha Portuguesa, bem como algumas

das suas particularidades. É também apresentada uma formulação genérica para a versão base do problema que servirá como rampa de lançamento para os modelos a desenvolver, além de permitir uma melhor contextualização.

O Capítulo 4 é dedicado ao estado da arte das diferentes abordagens para a resolução dos *Vehicle Routing Problems* em geral, e do *Dial-a-Ride Problem* em particular. São consideradas duas abordagens diferentes para a resolução do problema em causa que, por sua vez, recorrem a variadíssimas técnicas.

Tendo em conta que um dos objectivos desta dissertação é o desenvolvimento de um modelo para a resolução do DARP segundo uma abordagem exacta, o Capítulo 5 está reservado à apresentação da técnica mais frequentemente usada para a resolução deste problema, a Programação Linear. É feita uma pequena descrição introdutória da mesma com especial destaque para a vertente multiobjectivo em que se enquadra o problema da CVP. Por fim, apresenta-se um modelo linear inteiro misto proposto por um autor de referência na área.

No Capítulo 6, e à semelhança do capítulo anterior, é feita uma apresentação geral da Programação com Restrições e das técnicas existentes para a resolução de problemas combinatórios dada a pretensão da sua utilização nesta dissertação.

O Capítulo 7 está destino à apresentação de diferentes modelos de Programação com Restrições para o *Dial-a-Ride Problem*, com o objectivo de aplicá-los ao caso de estudo considerado.

No Capítulo 8 são apresentadas as instâncias de teste, tanto as relacionadas com a Cruz Vermelha Portuguesa como um outro conjunto de instâncias referência no âmbito do DARP. Após a apresentação das instâncias são também apresentados os resultados obtidos pelo modelo escolhido para os conjuntos de instâncias apresentados.

Por fim, no Capítulo 9 são apresentadas as conclusões do trabalho realizado ao longo desta dissertação e são também apontados possíveis caminhos para desenvolvimento futuro.

De notar que no Anexo A se disponibilizam informações relativas à Cruz Vermelha Portuguesa, nomeadamente no que diz respeito às entidades protocoladas com a delegação Amadora – Sintra, ao contexto geográfico bem como aos pedidos de transporte das instâncias contempladas. Ainda no mesmo anexo são apresentadas as soluções obtidas para cada dia de trabalho da delegação em causa. Por outro lado, no Anexo B apresentam-se os dados relativos à instância de teste criada e no Anexo C apresentam-se as instâncias referência criadas por Cordeau (2006), com foco nas suas características e no seu formato.

2

2. O Problema da Cruz Vermelha Portuguesa (Delegação Amadora – Sintra)

O foco desta dissertação incide na delegação Amadora – Sintra da Cruz Vermelha Portuguesa que, como a sua designação indica, opera nos concelhos de Amadora e de Sintra, tornando-a responsável por um aglomerado populacional de aproximadamente 500.000 habitantes. Como é apanágio no contexto organizacional da Cruz Vermelha Portuguesa, as delegações com uma amplitude geográfica considerável costumam ter extensões que servem, acima de tudo, para dar apoio logístico à sede da delegação. A delegação Amadora – Sintra não é excepção além de estar sediada na Amadora, tem a seu cargo 2 extensões: Queluz e Sintra, com o acréscimo de coordenar também uma Unidade de Socorro que se situa, estrategicamente, na Brandoa (ver o mapa contemplado no Anexo A, na Figura A.1).

Esta Unidade de Socorro da Brandoa é composta por dois pólos de serviço que, embora integrados, operam de forma independente entre si. O primeiro pólo, conhecido como pólo das urgências (Pólo 1), está encarregue não só dos serviços de urgência, como também do transporte de doentes do Hospital Amadora/Sintra, com quem esta delegação tem um acordo. Este pólo funciona 24 horas por dia, durante toda a semana, incluindo feriados. O segundo pólo, conhecido como pólo das coordenações programadas (Pólo 2), incorporado na sede da delegação, é responsável pelo transporte de doentes programados por requisição, extra e particulares. Este pólo funciona de segunda a sexta, das 7 horas até às 19 horas. Em caso de necessidade extra horário, o transporte terá que ser assegurado pelo pólo das urgências. Este transporte é realizado de acordo com as necessidades dos utentes que, tipicamente, passam por deslocações aos centros de fisioterapia, a consultas e a tratamentos. Além das emergências e do transporte de doentes, a Unidade de Socorro da Amadora – Sintra presta

ainda serviços em congressos, espectáculos, provas desportivas, passeios, casamentos, acompanhamento de claques em jogos de futebol, entre outros.

2.1. Pacientes

A delegação Amadora – Sintra presta um serviço de transporte de doentes programados a um conjunto de utentes que necessitam de ser transportados para os locais onde lhes são prestados tratamentos de saúde, onde serão consultados ou onde farão alguns exames de rotina.

Não existem quaisquer restrições no que diz respeito aos doentes que a CVP transporta, isto é, qualquer pessoa pode solicitar este serviço desde que pertença à área geográfica em causa. No entanto, os doentes programados podem ser classificados em diferentes categorias segundo as suas necessidades e periodicidade de transporte. Os doentes cuja necessidade de transporte não varia, seguindo um padrão até o tratamento estar finalizado são classificados como “regulares”. Por outro lado, os doentes “por requisição” são todos aqueles que necessitam de tratamentos esporádicos, não sendo o transporte realizado de forma regular. Existem ainda os doentes “extra” que precisam de transporte para uma entidade diferente da que recorrem habitualmente para realizar o tratamento, ainda que esta necessidade esteja relacionada com o tratamento. Como um possível exemplo desta categoria de doentes, temos todos os utentes de hemodiálise que, com alguma frequência, necessitam de se deslocar ao hospital para realizarem exames devido aos tratamentos a que estão sujeitos. Por fim, mas não menos importante, os doentes “particulares” caracterizam-se por requerer o transporte por iniciativa própria, assumindo por completo os custos do mesmo.

Como se pode perceber, os transportes estão associados a necessidades de saúde, pelo que se espera que os utentes tenham também diferentes necessidades físicas e de transporte, sendo que a frota da Cruz Vermelha Portuguesa contempla estas situações munindo as viaturas de macas e cadeiras de rodas.

2.2. Consultas / Tratamentos

São diversos os factores que actualmente levam as pessoas a ter que visitar os médicos ou outro tipo de profissionais de saúde com alguma regularidade. Doenças crónicas, exames de rotina, tratamentos específicos para lesões musculares e ósseas constituem a maioria das situações observadas. Desta forma a delegação da Amadora – Sintra procurou criar acordos

com algumas entidades que prestam cuidados ao nível da saúde que se encontram geograficamente próximas ou que constituem um acréscimo à diversidade de especificações das mesmas. Destas entidades (ver Tabela A.1) destacam-se alguns centros de fisioterapia, hospitais onde os doentes vão ser consultados ou fazer exames, bem como um centro de hemodiálise, o que se traduz num transporte, por parte da Cruz Vermelha Portuguesa, de uma média de 100 doentes por dia.

O protocolo estabelecido com estas entidades centra-se no transporte dos pacientes em troca de honorários pré-acordados, sendo que o cumprimento de tal acordo é da responsabilidade da Administração Regional de Saúde de Lisboa e Vale do Tejo (ARS). Por cada serviço de transporte, quer seja de ida ou de retorno, a ARS paga à CVP 5,74€ se a distância entre a morada do utente e a entidade que prestará os cuidados de saúde não ultrapassar os 30 quilómetros. Caso a distância seja superior, o pagamento passa a ser feito de acordo com a distância em causa, sendo a remuneração de 0,40€ por quilómetro.

O Hospital Prof. Doutor Fernando Fonseca, mais conhecido como Amadora/Sintra, tem um acordo ligeiramente diferente dos restantes, já que é a entidade com maior volume de pedidos de transporte. O protocolo estabelecido abriga a permuta de um retorno monetário mensal para que o hospital disponha de uma ambulância durante 24 horas para transferências hospitalares e inter-hospitalares, sendo que no período das 8:00 às 20:00, este acordo envolve duas ambulâncias. Este protocolo traz alguns benefícios à CVP pois além do retorno monetário, o ter permanentemente uma ambulância estacionada nas urgências do hospital, permite à CVP usufruir dessa posição para funcionar como um *outdoor* institucional, podendo proporcionar transporte a particulares que necessitem de tal serviço.

2.3. Pedidos

Os pedidos de transporte podem ser realizados via telefone, correio electrónico ou pessoalmente na Unidade de Socorro da delegação Amadora – Sintra da Cruz Vermelha Portuguesa. Ainda que grande parte dos pedidos chegue através das entidades protocoladas que garantem o transporte dos seus utentes através do serviço de transporte de doentes programados da CVP, existem diversos pedidos a nível particular que chegam pelos próprios beneficiários, familiares ou mesmo *caregivers* (prestadores de cuidados de saúde).

A maioria dos pedidos que chegam através das entidades protocoladas dá entrada até às 19:00 do dia anterior ao do serviço, já que a CVP necessita de todos os dados para definir as

rotas. Por outro lado, as entidades também dispõem de informação sobre quais os pacientes que vão ter que atender no dia seguinte. Pedidos para serviços no próprio dia também ocorrem. Estes são, tipicamente, realizados por privados ou, esporadicamente, pelas entidades protocoladas devido a cancelamentos, alterações de última hora ou devido a alguma situação imprevista. Os pedidos realizados com antecedência, ou seja, pelo menos um dia antes do transporte, têm garantia de que serão aceites, algo que pode não acontecer com os pedidos efectuados no próprio dia, uma vez que as rotas estão planeadas e nem sempre é possível refazê-las.

Para proceder ao registo de um pedido, a CVP necessita de saber alguns dados do utente em causa, nomeadamente, nome, número de beneficiário, morada, bem como se este necessita de transporte especial, isto é, se necessita de ser transportado em maca ou em cadeira de rodas. Para finalizar é necessário saber qual a origem e o destino do pedido em causa, sendo que por norma um pedido contempla dois transportes, a ida da residência do utente para a entidade que presta tratamento e o regresso. Existem também pedidos num só sentido, isto é, apenas da residência para a entidade de saúde ou apenas o regresso.

2.4. Origens / Destinos (2 Pólos)

É importante referir que a frota da Cruz Vermelha Portuguesa tem como base os dois pólos da Unidade de Socorro da delegação Amadora – Sintra, estando os veículos distribuídos de acordo com a natureza e, consequentes, necessidades de cada pólo. A título de exemplo, é esperado que o Pólo 1, ligado às urgências, tenha mais veículos com licença para este tipo de actividade que o Pólo 2, questões que serão abordadas com mais detalhe na Secção 2.6.

Esta temática das origens e destinos dos veículos que compõem a frota torna-se pertinente na medida em que os dois pólos devem ser considerados tanto origens como destinos uma vez que as viaturas saem e regressam ao pólo correspondente no fim de cada dia de serviço. Situações em que um veículo inicia o serviço num pólo e termina noutro são excepcionais e, quando ocorrem, estão normalmente relacionadas com realocações da frota.

2.5. Rotas

Como referido anteriormente, as operações de transporte de doentes programados são planeadas a partir das informações recebidas das entidades protocoladas, bem como dos pedidos

realizados pelos utentes ou pelos seus *caregivers*. Com base nestas informações, um funcionário desta delegação da CVP tem a responsabilidade de estruturar as rotas para cada veículo da frota, disponibilizando as informações relevantes aos operadores das viaturas. Nesse conjunto de informação constam, normalmente, o nome dos utentes, as entidades para onde serão transportados ou onde deverão ser recolhidos, bem como as horas a que tais operações deverão ocorrer.

Ainda que estas informações estejam sujeitas a alterações de última hora devido a situações imprevistas, constituem a base para a definição das rotas de transporte de doentes programados. Actualmente estas são elaboradas através da experiência acumulada e do bom senso do colaborador responsável por tal tarefa, não havendo qualquer recurso a métodos teoricamente fundamentados. As requisições não previstas que por norma são realizadas no próprio dia do transporte têm que ser rapidamente encaixadas nas rotas já estabelecidas. No entanto, devido à falta de meios nem sempre é possível dar uma resposta positiva a todas as solicitações.

2.6. Veículos

A frota disponível para a Unidade de Socorro da delegação Amadora – Sintra é bastante heterogénea, variando no tipo de veículo, na lotação, bem como na capacidade de transportar doentes com diferentes necessidades, em particular, cadeiras de rodas, macas ou simplesmente lugares sentados.

Actualmente, esta unidade dispõe de 18 veículos o que perfaz um total de 73 lugares sentados, 5 lugares adaptáveis para cadeira de rodas que, em caso de não haver utentes com tal necessidade, podem ser utilizados como lugares sentados, bem como 6 macas para o transporte de doentes que não possam viajar sentados ou para situações de emergência. Na tabela que se segue (Tabela 2.1) apresenta-se resumidamente a frota e as suas especificações.

Tabela 2.1 - Descrição da frota da delegação Amadora – Sintra.

Veículo	Lugares Sentados	Cadeiras de Rodas	Macas	Unidades Disponíveis	Unidades no Pólo 1	Unidades no Pólo 2
Ambulância B-1	1	0	2	1	1	0
Ambulância B-2	1	0	1	3	3	0
Ambulância A2-1	7	0	0	6	0	6
Ambulância A2-2	5	2	0	2	1	1

Veículo	Lugares Sentados	Cadeiras de Rodas	Macas	Unidades Disponíveis	Unidades no Pólo 1	Unidades no Pólo 2
Ambulância A1	2	1	1	1	0	1
Veículo Ligeiro	3	0	0	5	2	0

De notar que apenas 2 dos veículos ligeiros estão afectos ao Pólo 1, os 3 restantes podem ser utilizados por ambos os pólos mas, por questões logísticas, pernoitam tipicamente no Pólo 2 onde está sediada a delegação. Outra particularidade desta frota é a ambulância do tipo A1 que se encontra no pólo 2 e, apesar de ter capacidade para transportar doentes acamados, não está licenciada para urgências, o que implica que todas as situações deste tipo apenas podem ser realizadas pelas ambulâncias do tipo B, as únicas com licença. No entanto, desde que não seja para fazer urgências, qualquer veículo pode ser utilizado indiscriminadamente por qualquer um dos pólos. Convém ressalvar que apesar dos pólos funcionarem de forma independente, quando um está sobrecarregado tenta coordenar as suas actividades com o outro.

2.7. Custos

Como já foi referido, a Cruz Vermelha Portuguesa é uma Organização Não Governamental pelo que nem sempre é fácil obter financiamento para as suas actividades e, dado o seu espectro de acção, os fundos obtidos devem ser geridos da melhor forma possível para que se possa chegar às diversas “frentes de combate”.

O serviço de transporte de doentes programados não é uma excepção e pretende também gerir os fundos da melhor forma possível. E essa gestão passa fundamentalmente pela redução dos custos variáveis do serviço, directamente relacionados com as distâncias percorridas pelas viaturas. Estes custos envolvem também outros factores como o consumo médio de combustível do veículo em causa, bem como o próprio preço do combustível.

Ainda que tipicamente o foco dos custos seja financeiro, não é o único. Na verdade existe um custo que pode ser considerado neste serviço cuja relevância está ao nível do financeiro, a qualidade de serviço, em particular, o tempo de espera e de viagem de cada utente.

Deste modo pretende-se minimizar os custos do serviço nestas duas vertentes, reduzindo os custos de cada rota, respectivas distâncias e, consequentemente, consumos da frota, sem prejudicar a qualidade de serviço, garantindo que os utentes não estarão demasiado tempo à espera do transporte na origem nem tão pouco dentro da viatura, até chegar ao destino pretendido.

2.8. Conclusões

A realidade da delegação Amadora – Sintra da Cruz Vermelha Portuguesa no que diz respeito ao transporte de doentes pode ser encarada como um problema de definição de rotas, tipicamente conhecidos como *Vehicle Routing Problems*. Este tipo de problema surge pelas mãos de Dantzig e Ramser (1959) no contexto do transporte de mercadorias localizadas num armazém para diversos clientes e tem-se revelado bastante importante em áreas como o transporte, distribuição e logística.

No contexto específico da Cruz Vermelha Portuguesa em que se transporte pessoas e se trata de um serviço orientado ao utilizador, existe, no domínio dos *Vehicle Routing Problems*, uma especificação que dá pelo nome de *Dial-a-Ride Problem* e que caracteriza de forma mais fidedigna o transporte de doentes. Este problema será abordado com mais detalhe no próximo capítulo.

3

3. *Dial-a-Ride Problem*

O *Dial-a-Ride Problem* (DARP) consiste na formulação de rotas e horários para transportar pessoas entre origens e destinos à sua escolha. Este tipo de problema insere-se na área do planeamento de rotas, vulgarmente conhecidos por *Vehicle Routing Problems* (VRP) (Laporte & Osman, 1995), e pode ser tomado como uma generalização dos problemas *Pick-up and Delivery Vehicle Routing Problem* (PDVRP) e *Vehicle Routing Problem with Time Windows* (VRPTW) (Berbeglia et al., 2007). No fundo, o que o distingue dos demais é a sua envolvência humana que obriga a que não se pense apenas em cumprir os objectivos particulares do problema tentando, por exemplo, minimizar os custos das viagens, mas em dar também atenção à perspectiva humana, tentando reduzir a inconveniência dos utentes aumentando assim a qualidade do serviço (Paquette et al., 2009).

Este é um dos aspectos que torna o DARP um pouco mais complexo que os restantes problemas de planeamento de rotas já que é necessário ponderar dois aspectos conflituosos. Por um lado, se a preocupação se centrar apenas em minimizar os custos com a frota, evitando assim que as viaturas façam muitos quilómetros, é provável que muitos dos utentes deste serviço passem demasiado tempo em viagem. No reverso da medalha está a situação em que nos preocupamos bastante com a qualidade do serviço impondo *timings* muito rígidos, o que leva a que cada pessoa usufrua de um serviço praticamente particular, aumentando o custo total de transporte. A abordagem mais comum para a gestão deste conflito passa por minimizar os custos de operação, maximizando também a qualidade do serviço através da minimização do tempo de transporte em excesso despendido pelos utentes. Para tal é necessário medir este excesso que consiste na diferença entre a hora a que foram recolhidos ou deixados num local e a hora ideal em que tais operações deveriam ter ocorrido.

Habitualmente, esta diferença é usada sob a forma de restrição modelando a qualidade de serviço (Cordeau & Laporte, 2007), mas pode também ser encontrada na função objectivo como uma penalização (Berbeglia et al., 2010).

O DARP pode ser abordado de forma estática ou dinâmica (Savelsbergh & Sol, 1995). No primeiro caso, todas as requisições de transporte são conhecidas *a priori*, enquanto no segundo caso, as requisições vão chegando ao longo do dia e as rotas devem ser ajustadas em tempo real de acordo com os pedidos. Na verdade, raramente este tipo de problemas é totalmente dinâmico já que usualmente é conhecido de antemão um subconjunto de requisições (Borndörfer et al., 1997).

A versão base do *Dial-a-Ride Problem* assume uma abordagem estática na qual existem *m* veículos homogéneos, todos concentrados na mesma base, e tem como objectivo planear as diferentes rotas com o menor custo possível. Com o passar do tempo, diferentes realidades e necessidades foram surgindo, principalmente devido ao envelhecimento da população e à evolução dos serviços, pelo que o DARP tem vindo a ganhar outras exigências. Actualmente, além de, na maioria dos casos se ter que lidar com uma abordagem dinâmica, as frotas são em geral heterogéneas. Se tomarmos como exemplo o transporte de doentes, alguns veículos têm capacidade para transportar cadeiras de rodas, outros só ambulatórios, outros ainda, ambos os casos. Estes veículos podem não estar todos concentrados na mesma base, especialmente quando os problemas se inserem em grandes áreas geográficas. A questão pertinente continua a ser a optimização do transporte porta a porta, planeando rotas para os utentes de um serviço, respeitando as restrições temporais, minimizando os custos operacionais e mantendo uma qualidade de serviço aceitável.

3.1. Formulação

A versão base do DARP pode então formular-se com recurso a um conjunto de *m* veículos com capacidade *Q*, $Veículos = \{V_1,...,V_m\}$, um depósito *Depósito*, isto é, o local onde as viaturas pernoitam, e um conjunto de *c* clientes, $Clientes = \{C_1,...,C_c\}$, que é estabelecido com base num outro conjunto, o conjunto de pedidos. Este conjunto de pedidos é representado por *Pedidos,* com cardinalidade *n*, sendo $n \geq \#Clientes$, e é composto por dois outros conjuntos: *PedidosP* e *PedidosD*, de *Pickup* e *Delivery*, respectivamente. No contexto do transporte de doentes, o primeiro conjunto representa os pedidos de transporte de ida para as entidades prestadoras de cuidados de saúde (hospitais, consultórios, clínicas) e o segundo os pedidos de

regresso. Deste modo o conjunto $Pedidos = PedidosP \cup PedidosD$ é composto por tuplos do género

$$(Cliente, Origem, Destino, Hora) \tag{3.1.1}$$

com $Cliente \in Clientes$, *Origem* ≠ *Destino* e *Hora* pertencente a um conjunto de tempos *Hs*. Este último é um parâmetro cuja interpretação varia consoante o tipo de pedido, isto é, no caso de se tratar de um pedido de regresso à residência, *Hora* representa o término do tratamento, por outro lado no pedido "inverso", representa o tempo de início da consulta ou de outro tipo de tratamento no âmbito do transporte de doentes. Com base nos dados fornecidos pelos pedidos de transporte, pode representar-se o conjunto dos vários locais a visitar como $Locais = Origens \cup Destinos$, sendo *Origens* e *Destinos* os vários locais de origem e destino enunciados no conjunto de pedidos, respectivamente.

É também importante considerar o conceito de paragem de um veículo que pode ser definido como um tuplo do tipo

$$(Local, Hora, CsIn, CsOut) \tag{3.1.2}$$

com $Local \in Locais$, $Hora \in Hs$ e os conjuntos *CsIn* e *CsOut* representando os clientes a bordo do veículo à chegada a *Local* e à saída do mesmo, respectivamente. Adicionalmente, está associado, a cada paragem, um tempo que representa a duração do serviço nesse mesmo local, associada ao tempo que os utentes demorar a entrar e/ou sair do veículo, bem como a outros procedimentos que possam ser necessários no contexto do serviço. Por se tratar de uma formulação básica do *Dial-a-Ride Problem*, com o objectivo de contextualizar o problema, assumir-se-á que este tempo será igual para todas as paragens e dado por $D \in Ts$. Ainda em relação aos conjuntos *CsIn* e *CsOut*, há que referir que ambos estão contidos no conjunto *Clientes*, formalmente, $CsIn \subset Clientes$ e $CsOut \subset Clientes$, além de que a sua cardinalidade é sempre menor ou igual à capacidade dos veículos, ou seja, 0 ≤ #*CsIn* ≤ *Q* e 0 ≤ #*CsOut* ≤ *Q*.

É necessário definir-se um conjunto de *m* rotas, $Rotas = \{R_1, ..., R_m\}$, associadas a cada um dos veículos que compõem a frota e que irão constituir a solução do problema. Cada rota é representada por uma sequência de *paragens* do tipo $\langle paragem_1, ..., paragem_f \rangle$, com $1 \le f \le 2n + 2$, devido ao facto de serem realizadas duas paragens por cada pedido e de uma rota vazia ser representada por apenas uma paragem que diz respeito ao depósito. Tendo em conta que cada rota se inicia e termina no depósito ao qual o veículo pertence, pode dizer-se que $\text{Local}(paragem_1) = \text{Local}(paragem_f) = Depósito$, sendo $\text{Local}(paragem_i)$ o primeiro

campo do tuplo correspondente à i-ésima paragem da rota, com $1 \leq i \leq f$. De forma idêntica é possível aceder aos restantes campos dos tuplos *paragem*, bem como aos campos dos tuplos *pedido*.

Por outro lado será útil definir uma função *tempo*: $Locais^2 \rightarrow \mathbb{R}_0^+$ que indica o tempo de uma deslocação entre dois locais e, analogamente, uma função *custo*: $Locais^2 \rightarrow \mathbb{R}_0^+$, que poderá ser linear relativamente à primeira.

Com base nestes dados podem formular-se as condições necessárias para que o problema possa ser resolvido, de onde se destacam as restrições ao nível da precedência dos pedidos, bem como no que diz respeito aos tempos impostos pelas viagens e à capacidade dos veículos. Em particular, torna-se necessário garantir que:

i. Cada pedido é servido por um, e só um, veículo, estando, consequentemente, as paragens referentes à sua origem e ao seu destino na rota do mesmo. Além disso, é esperado que o veículo visite a origem antes do destino. Formalmente:

$$\forall p \in Pedidos \left(\exists! r \in Rotas \left(\begin{array}{c} \exists! paragem_i, paragem_j \in r, i < j : \\ \text{Local}(paragem_i) = \text{Origem}(p) \\ \wedge \\ \text{Local}(paragem_j) = \text{Destino}(p) \end{array} \right) \right) \tag{3.1.3}$$

ii. No contexto da restrição anterior, há que garantir também que o tempo de viagem de um cliente, desde a origem até ao seu destino, não excede um limite *L*. Aproveitando a formalização da restrição anterior, é possível expressar esta condição da seguinte forma:

$$\text{Hora}(paragem_j) - \text{Hora}(paragem_i) \leq L \tag{3.1.4}$$

iii. Os clientes chegam aos seus destinos atempadamente, não havendo atrasos e que são recolhidos de forma breve para o caso dos pedidos de regresso às suas residências. De modo a introduzir alguns parâmetros relativos à qualidade de serviço, estabelece-se que o cliente pode, por exemplo, esperar no máximo 30 minutos pela consulta ou, após esta, esperar o mesmo intervalo de tempo pela viatura que o transportará para a sua residência. Formalmente, e generalizando este intervalo para *M*, com $M \in Ts$, é possível definir-se as seguintes restrições, com base na formalização anterior:

$$\text{Hora}(p) - M \leq \text{Hora}(paragem_j) \leq \text{Hora}(p) \tag{3.1.5}$$

com $p \in PedidosP$ e $paragem_j$ como a paragem correspondente ao local de destino do pedido *p*.

De forma idêntica, pode definir-se a restrição para o caso em que se trata de pedidos das entidades para as residências dos utentes,

$$\text{Hora}(p) \leq \text{Hora}(paragem_i) \leq \text{Hora}(p) + M \tag{3.1.6}$$

com $p \in PedidosD$ e com $paragem_i$ pertencendo à rota correspondente ao veículo que serve o pedido *p* e representando a paragem no local de origem desse mesmo pedido.

iv. Não se "perdem" clientes ao longo das rotas, ou seja, os conjuntos *CsIn* e *CsOut* associados a cada paragem correlacionam-se:

$$\forall r \in Rotas\left(\forall i \in \{1,...,\#r-1\}\left(\text{CsOut}(paragem_i) = \text{CsIn}(paragem_{i+1})\right)\right) \tag{3.1.7}$$

Assumindo a existência de uma função *passageiros*: $Locais \rightarrow \{-Q,...,Q\} \setminus \{0\}$, que indica o número de passageiros que entra ou sai de um determinado local, consoante o pedido associado ao local pertença a *PedidosP* ou *PedidosD*, respectivamente. Deste modo pode dizer-se que:

$$\forall r \in Rotas\left(\forall i \in \{1,...,\#r\}\left(\begin{array}{c}\text{CsOut}(paragem_i)\\=\\\text{CsIn}(paragem_i)\\+\\\text{passageiros}(\text{Local}(paragem_i))\end{array}\right)\right) \tag{3.1.8}$$

v. A cardinalidade das rotas é estimável, devido ao facto de cada pedido se materializar em duas paragens, e sempre igual ou superior a quatro para rotas não vazias. Caso se trata de uma rota vazia, no sentido em que não é dada resposta a nenhum pedido, é apenas contemplada a paragem relativa ao depósito. Assumindo que a expressão *a%b* devolve o resto da divisão inteira de *a* por *b*, é possível dizer que:

$$\forall r \in Rotas\left(\begin{array}{c}1 \leq \#r \leq \#Pedidos + 2\\\wedge\\\#r \neq 1 \Leftrightarrow \#r\%2 = 0\end{array}\right) \tag{3.1.9}$$

vi. As rotas servem todos os pedidos, sem que estes se repitam nem, tão pouco, que se façam desvios desnecessários:

$$\#Pedidos \times 2 + 1 + m \leq \sum_{r \in Rotas} \#r \leq \#Pedidos \times 2 + 2m \tag{3.1.10}$$

Para contemplar a vertente de optimização do *Dial-a-Ride Problem* será necessário proceder à minimização dos custos de serviço. Tal operação poderá, a título de exemplo, ser definida por:

$$\min Z = \sum_{R \in Rotas} \sum_{i=1}^{\#R-1} \text{custo}\left(\text{Local}\left(paragem_i\right), \text{Local}\left(paragem_{i+1}\right)\right) \tag{3.1.11}$$

3.1.1. Extensões

A formulação base do DARP considera uma frota homogénea, no entanto em muitas das suas instâncias os veículos que a compõem podem ter características diferentes quer a nível de capacidade (podendo inclusive contemplar lugares adaptados a pessoas incapacitadas, e.g.), quer a nível de consumo de combustível, entre outras. De modo a considerar esta extensão, poderia associar-se a cada viatura do conjunto *Veículos* uma variável *tipo* através da qual se acederia às suas características.

Outra extensão ao DARP está relacionada com a qualidade de serviço, algo impelido pela mudança de paradigma associada ao conceito de serviço que se tem vindo a verificar na última década. De acordo com Paquette et al. (2009) a qualidade de serviço pode ser entendida de diversas formas. A perspectiva mais simples garante apenas que o tempo de viagem entre o local de origem de um pedido e o seu destino não excedem um determinado limite, considerando também que, no contexto do transporte de doentes, o utente pode chegar à entidade prestadora de cuidados de saúde com um dado intervalo de antecedência, aplicando-se o mesmo conceito para o regresso à sua residência, isto é, poderá apenas esperar esse intervalo por uma viatura. Na formulação acima estas condições são garantidas pelas restrições representadas em (3.1.4), (3.1.5) e (3.1.6). No entanto têm sido cada vez mais as abordagens que consideram também o tempo de espera dos clientes a bordo do veículo como um factor representativo da qualidade de serviço, dada a influência psicológica que este tem na sua percepção (Parragh, 2011).

3.2. Conclusões

Neste capítulo foi apresentada uma contextualização do *Dial-a-Ride Problem*, expondo as razões pelas quais este se destaca dos restantes *Vehicle Routing Problems* e, consequentemente, se torna aplicável ao problema enfrentado pela delegação Amadora – Sintra da Cruz Ver-

melha Portuguesa. Adicionalmente foi apresentada uma formulação genérica do DARP com o objectivo de clarificar algumas das suas particularidades.

No próximo capítulo será apresentada uma revisão dos trabalhos publicados que se relacionam com o problema abordado nesta dissertação. Nela serão realçadas algumas das abordagens que têm sido utilizadas para a resolução dos VRP's em geral e do DARP em particular.

4

4. Trabalho Relacionado

Segundo Savelsbergh e Sol (1995) existem dois tipos de abordagens aos problemas de *Pickup and Delivery*: exactas e aproximadas. As primeiras, as exactas, abordam o problema de uma forma genérica modelando todas as restrições que lhes estão associadas. Dada a sua complexidade computacional, provocada pelo elevado número de pedidos e de restrições necessárias, nem sempre as formulações exactas permitem a resolução de problemas reais. Assim, as abordagens aproximadas, também conhecidas como heurísticas acabam por ganhar algum relevo já que são capazes de obter soluções quase óptimas ou mesmo óptimas dentro de um tempo computacional aceitável.

Nas próximas secções são relatadas cronologicamente algumas destas abordagens de modo a dar um panorama geral do trabalho realizado nesta área. Para mais informações recomenda-se a leitura de alguns artigos *survey* (Cordeau & Laporte, 2003, 2007; Berbeglia et al., 2007; Berbeglia et al., 2010).

4.1. Abordagens Exactas

Classicamente as abordagens exactas estão ligadas a alguma exaustividade no que diz respeito à pesquisa por soluções (Woeginger, 2003). No entanto, os problemas que por norma lhe são associados não são de resolução trivial, muito menos quando se necessita de uma solução óptima.

Ainda que estas abordagens tenham vindo a ser objecto de estudo ao longo dos últimos anos, em particular no que diz respeito aos *Vehicle Routing Problems*, nem sempre conseguem dar resposta a problemas com uma dimensão considerável (Ropke, 2005).

4.1.1. Programação Dinâmica

Um dos primeiros casos do *Dial-a-Ride Problem* foi apresentado em 1980 por Psaraftis (Psaraftis, 1980), um problema relativamente simples onde todos os pedidos são conhecidos de antemão e todos os utilizadores são transportados por um único veículo. Psaraftis formulou e resolveu o problema através de programação dinâmica onde a função objectivo consistia na minimização de uma soma ponderada entre o tempo total da rota e a insatisfação do cliente, expressa pela combinação pesada entre o tempo de espera antes da recolha e o tempo de viagem. Mais tarde o mesmo autor propôs uma extensão onde se contemplam janelas temporais definidas pelos utilizadores (Psaraftis, 1983). No entanto, dada a sua elevada complexidade temporal, $O\left(n^2 3^n\right)$ com n o número de localizações, apenas foi possível resolver instâncias pequenas, com um número de pedidos inferior a 10. Ainda assim o autor considerou que, embora a maioria dos DARP's que surgem em contexto real tenham exigências maiores, a abordagem que propôs poderia ser útil como uma sub-rotina de um algoritmo para problemas com múltiplos veículos, dado que o número de utilizadores por rota e veículo são tipicamente pequenos. Não só Psaraftis defendia esta ideia mas também Sexton (1979) e Sexton e Bodin (1985a, 1985b) viam o DARP com um único veículo como uma fase da heurística do DARP com múltiplos veículos, na qual os utilizadores tinham já sido agrupados.

Desrosiers et al.(1986) apresentaram uma formulação para o DARP com apenas um veículo, incluindo já janelas temporais e restrições de precedência e de capacidade do veículo. O algoritmo, concebido através de programação dinâmica e baseado na técnica de *column generation*, e conseguiu soluções óptimas para instâncias com um número de pedidos inferior a 40.

4.1.2. Programação Inteira

Com os resultados obtidos pelas abordagens que recorrem à Programação Dinâmica surge a necessidade de modelar o *Dial-a-Ride Problem* de outra forma, pelo que em 1997 Ruland e Rodin (1997) formulam o DARP como um caso particular do *Pickup and Delivery Problem* (PDP), através de programação inteira. À semelhança de outras abordagens exactas, o algoritmo de *Branch-and-Cut* utilizado tem um desempenho interessante para instâncias pequenas, até seis pedidos, no entanto um aumento do número de pedidos conduz a tempos de computação não aceitáveis.

Mas a saga pela procura de alternativas não cessou e cerca de uma década depois, Cordeau (2006) apresenta uma formulação de programação inteira mista e um algoritmo de

Branch-and-Cut que usa desigualdades válidas tanto específicas do DARP como gerais aos VRP. O algoritmo desenvolvido recorre também a técnicas de pré-processamento para reduzir a dimensão do problema. Ainda que os resultados apresentados tenham sido mais interessantes, confirmavam que uma abordagem deste género não pode ser utilizada para resolver grandes instâncias dado o seu desempenho computacional. No entanto, o autor defende que poderá ser usada para optimizar rotas individuais ou pequenos subconjuntos de rotas.

No ano seguinte, Ropke et al. (2007) apresentam duas novas formulações para o *Pickup and Delivery Problem with Time* Windows (PDPTW) que, conjuntamente com novas desigualdades válidas, são usadas num algoritmo de *Branch-and-Cut*. Os testes foram realizados em instâncias com um número de veículos menor ou igual a 8 e cerca de 96 pedidos, tendo sido resolvidas até à optimalidade.

Ainda no decorrer de 2007 Melachrinoudis et al. (2007) descrevem um modelo para um problema de *Dial-a-Ride* para aplicação no *Center for Addictive Behavior Health and Recovery Services*, localizado na área metropolitana de Boston, que presta cuidados de saúde e realiza transportes de utentes. Os autores propõem um modelo estático onde os pedidos que chegam no dia anterior são afectos a veículos que compõem uma frota heterogénea distribuída por vários locais, minimizando os custos de transporte e a inconveniência do utilizador. Este modelo permite também considerar *soft time windows* através da minimização dos desvios a estas janelas temporais à semelhança do que é feito na Programação por Metas. Além disso, o modelo permite ainda a definição de rotas abertas, isto é, o veículo inicia a rota num depósito e termina noutro de modo a facilitar os serviços do dia seguinte. Foram propostas duas abordagens diferentes para resolução do problema, uma exacta e uma aproximada baseada em *tabu search* (descrita na Secção 4.2.3) motivada pela eficiência da primeira que não obteve resultados em tempo considerado razoável pelos autores.

Em 2010, Loureiro (2010) baseia-se no modelo proposto por Cordeau (2006) e aplica-o ao caso de estudo desta dissertação, o problema da definição de rotas de transporte porta à porta da delegação Amadora – Sintra, da Cruz Vermelha Portuguesa. Os resultados obtidos com esta abordagem permitem à CVP encontrar rotas óptimas e, consequentemente, minimizar os custos de operação, bem como melhorar a qualidade do serviço prestado, na medida em que a distância percorrida é menor. Além disso, o facto de existir uma restrição que limita o tempo de viagem de um utilizador, neste caso a uma hora, será, à partida, um melhoramento no que diz respeito à qualidade de serviço pois a CVP não controla este factor devidamente. No entanto

foram apenas consideradas instâncias de teste com um número de pedidos até 21, alegando dificuldades computacionais em instâncias de maior dimensão. As instâncias de teste utilizadas correspondem a quatro dias de operação de uma ambulância que apenas tem a capacidade de transportar pacientes sentados, ou seja, sem necessidades especiais de transporte. Os resultados indicam que com este modelo a CVP poderia reduzir, em média, 30% dos custos variáveis por rota, ainda que não tenha capacidade para lidar com todos os pedidos que chegam à delegação.

4.1.3. Programação com Restrições

Ainda que a abrangência da Programação com Restrições (PR) (Rossi et al., 2006) a pudesse classificar tanto como uma abordagem exacta, como aproximada, a sua génese e a sua capacidade de ser exaustiva, colocam-na no âmbito das abordagens exactas.

Uma das primeiras abordagens a contemplar o uso da Programação com Restrições na resolução dos problemas de definição de rotas surge por Paul Shaw em 1998 que, embora implemente uma heurística de pesquisa local, recorre a algumas técnicas da PR como a propagação e conceitos como a árvore de pesquisa para direccionar os movimentos de forma mais incisiva, obtendo resultados interessantes (Shaw, 1998). No entanto, ainda no decorrer de 1998, Gilles Pesant e os seus colaboradores apresentaram um modelo de PR para o problema do caixeiro-viajante com janelas temporais (Pesant et al., 1998). Os resultados obtidos foram melhores que os das abordagens de Programação Dinâmica o que levou a que os autores testassem este tipo de abordagem noutros problemas combinatórios idênticos ao *Traveling Salesman Problem with Time Windows* (TSPTW). Destacando a flexibilidade dos modelos de PR e a sua capacidade de endereçar de forma mais fidedigna as restrições dos problemas reais como duas das grandes vantagens deste tipo de abordagem, conduziram mais um estudo com vista a resolver um problema com novas características (Pesant et al., 1999).

Em 2002, Focacci et al. (2002) classificam a Programação com Restrições como uma disciplina emergente que permite a integração de conceitos da Investigação Operacional e da Inteligência Artificial, revelando potencial para complementar as técnicas na programação matemática. Esta ligação levou a que Focacci integrasse uma equipa que propôs um método de *column generation* baseado em PR para problemas de definição de rotas de veículos (Rousseau et al., 2004).

No caso do *Dial-a-Ride Problem* em particular, os trabalhos que recorrem à Programação com Restrições para a sua resolução são escassos, sendo o trabalho de Berbeglia et al. (2010)

uma das mais recentes contribuições na área. Estes autores usam a PR para determinar uma solução admissível de uma instância do DARP ou para provar a sua impossibilidade. Esta última tarefa, dada a sua complexidade, tem vindo a ser alvo de estudo por outros autores (Hunsaker & Savelsbergh, 2002; Tang et al., 2010). Do conhecimento do autor desta dissertação este é o único artigo que, à data da realização deste documento, apresenta uma modelação por Programação com Restrições do DARP com uma frota homogénea, um único depósito e janelas temporais. O problema estudado pode ser classificado como um *Constraint Satisfaction Problem* (CSP) o qual será apresentado com detalhe no Capítulo 6 deste documento. Para atingir o objectivo a que se propõe, o algoritmo recorre a diversas técnicas de redução do espaço de pesquisa, nomeadamente algoritmos de filtragem e heurísticas para a selecção de variáveis e de valores. A implementação foi concebida através da linguagem C++ e do ILOG Solver 6.0, sendo todos os testes realizados num AMD® Opteron™ 2,5 GHz Dual Core. As instâncias usadas foram recolhidas de Ropke et al. (2007) e só foram consideradas as que continham pelo menos 40 pedidos. O autor conclui que as técnicas de redução do espaço de pesquisa e de filtragem conseguem reduzir o tempo necessário para encontrar uma solução admissível em cerca de 80%, comparativamente ao algoritmo em que a Programação com Restrições está apenas por sua conta. Comparativamente à abordagem de Cordeau e Laporte (2003), os resultados, para as instâncias originais, indicam que este algoritmo é geralmente mais lento a encontrar uma solução admissível. No entanto, em problemas mais restringidos, tem um melhor desempenho que a pesquisa tabu. Acresce ainda o facto de conseguir demonstrar que um problema é impossível, algo que a abordagem de Cordeau e Laporte nem sempre consegue.

4.2. Abordagens Aproximadas

Motivados pela qualidade dos resultados computacionais das abordagens exactas começam a surgir diversos artigos onde são propostas heurísticas e metaheurísticas para resolução de problemas complexos como o caso dos VRP's em geral e do DARP em particular.

São diversas as técnicas existentes e aplicadas ao DARP como se pode verificar através de artigos comparativos de várias destas técnicas (D'Souza et al., 2011; Hosny & Mumford, 2008) cujo sucesso se prende muito pela sua rapidez na pesquisa de soluções quase óptimas.

4.2.1. Heurísticas de Inserção

As heurísticas de inserção foram das primeiras a surgir no contexto do *Dial-a-Ride Problem* e, além de pretenderem melhorar o desempenho da pesquisa por soluções, o seu principal objectivo é "encaixar" os pedidos nas diversas rotas de modo a potenciar soluções.

No que diz respeito ao DARP com múltiplos veículos, o trabalho de Jaw et al. (1986) é considerado um dos pioneiros neste contexto. Nele, descreve-se um algoritmo heurístico de inserção sequencial no qual são seleccionados utilizadores de acordo com a hora especificada para recolha, sendo a inserção nas rotas feita de forma gradual. O modelo considerado pelos autores contempla janelas temporais, tempo máximo de viagem para cada utilizador e não permite que os veículos estejam parados ou façam compassos de espera com passageiros a bordo. Assim, além da minimização dos custos de operação, a função objectivo pretende também minimizar a inconveniência do utilizador, combinando factores como o tempo de viagem e os desvios aos tempos estabelecidos para recolha e entrega. Este algoritmo foi testado em instâncias fictícias com 250 clientes e 4 ou 5 viaturas, bem como em instâncias reais com 2617 clientes e 28 veículos.

Até meados dos anos 90, a maioria dos trabalhos debruçavam-se sobre a resolução de problemas considerando instâncias fictícias ou baseadas em dados reais. Nesta altura começam a surgir trabalhos cujas instâncias do DARP são de facto reais, sendo uma das primeiras proposta por Madsen et al. (1995) onde, assumindo uma frota heterogénea e múltiplos objectivos, tratam o problema do *Copenhagen Fire-Fighting Service* (CFFS). Os autores desenvolveram um algoritmo que se baseia numa heurística de inserção, tendo sido aplicado a instâncias reais com 24 veículos e 300 utilizadores.

Um ano mais tarde, Toth e Vigo (1996) apresentaram o problema do transporte de pessoas portadoras de deficiências ou com mobilidade reduzida em Bolonha, Itália. Neste trabalho são consideradas janelas temporais, uma frota heterogénea. A abordagem usada pelos autores baseou-se numa heurística para a minimização do custo total do serviço, onde a atribuição de pedidos a rotas era feita por meio de inserção paralela (por oposição à inserção sequencial) sendo posteriormente melhoradas através de trocas intra e inter rota. A abordagem foi testada em instâncias com 267 e 312 pedidos. Posteriormente os mesmos autores propuseram um procedimento melhor para a obtenção de uma solução inicial recorrendo a um algoritmo de *tabu thresholding* (Toth & Vigo, 1997).

Uns anos mais tarde, Diana e Dessouky (2004) aplicaram também um procedimento de inserção paralela que tem em conta os pesos da função objectivo que integra a distância

percorrida, o excesso de tempo em viagem e o tempo a bordo do veículo encontrando-se este parado. Os resultados apresentados são relativos a instâncias reais de 500 a 1000 pedidos geradas aleatoriamente com base em dados reais.

Wong e Bell (2006) recorrem também à inserção paralela e consideram como principal objectivo a minimização de uma função linear que contempla o tempo total de serviço, o tempo de viagem do cliente e os custos com um serviço externo de transporte (táxis, tipicamente) para os pedidos que não são atribuídos a nenhuma rota. Os autores propõem um procedimento de inserção paralela de pedidos, tendo sido os utilizadores previamente classificados de acordo com a dificuldade e inconveniência que causam a outros pedidos aquando da sua inserção numa determinada rota. Assim os pedidos mais complicados são considerados primeiro. Após a fase de inserção, é feita um melhoramento da solução através de reinserções e trocas entre rotas. O algoritmo foi testado em instâncias artificiais envolvendo cerca de 150 pedidos.

4.2.2. Heurísticas de Duas Fases

No mesmo ano em que surge uma das primeiras heurísticas de inserção para o DARP, Bodin e Sexton desenvolvem uma heurística de duas fases, que ficaria conhecida por *Cluster First, Route Second* (Bodin & Sexton, 1986) que, como o nome indica, consiste em agrupar os utilizadores antes de definir a sua distribuição pelos veículos, explorando assim a ideia já apresentada pelos autores em 1985 (Sexton & Bodin, 1985a, 1985b).

Esta heurística foi alvo de melhorias por Dumas et al. (1989) que incorpora parte da fase de *clustering* na fase de *routing* através da definição de *mini-clusters* de utilizadores situados na mesma área geográfica que deverão ser servidos aproximadamente ao mesmo tempo. Com esta abordagem os autores conseguiram com sucesso e rapidamente instâncias com um número de utilizadores inferior a 200, baseadas em dados reais de três cidades canadianas. Dois anos mais tarde, a fase de *mini-clustering* foi melhorada por Desrosiers et al. (1991) através de um método denominado pelos autores como inserções paralelas. Posteriormente, Ioachim et al. (1995) demonstraram que podia ser vantajoso, em termos da qualidade das soluções, recorrer a uma técnica de optimização para construção dos *clusters*.

Recentemente, Rekiek et al. (2006) apresentaram uma versão do DARP na qual o principal objectivo era minimizar o número de veículos usados. Os autores propuseram um algoritmo genético para a fase de *clustering* e um mecanismo de inserção para a de *routing*, obtendo bons resultados em instâncias baseadas em dados da cidade de Bruxelas, na Bélgica.

Beaudry et al. (2008) apresentam um estudo que analisa e resolve um problema de transporte de pacientes que começa a surgir em grandes hospitais onde têm que ser feitos transportes dentro do campus. Os pedidos chegam de forma dinâmica e pretende-se a sua inserção rápida em rotas de modo a providenciar um serviço eficiente. Apesar de ser um DARP este problema apresenta algumas particularidades dado ao contexto hospitalar. Os autores propõem uma heurística de duas fases para a resolução do problema. Na primeira fase é gerada uma solução admissível que, na segunda, é melhorada através de um algoritmo de pesquisa tabu. Dada a falta de dados e de resultados de problemas similares, é difícil avaliar a qualidade da abordagem produzida, no entanto os autores crêem que esta gera melhores soluções que as praticadas pelo hospital, dado o elevado número de queixas existentes.

4.2.3. Pesquisa Tabu

A pesquisa tabu surge em 1986 pelas mãos de Fred W. Glover (Glover & McMillan, 1986) e cedo começou a ter alguma relevância na resolução de problemas combinatórios de optimização, em particular em problemas de definição de rotas como o *Traveling Salesman Problem* (Malek et al., 1989).

No contexto do DARP esta metaheurística começou a ser utilizada como complemento a outras heurísticas como as referidas acima. Um destes exemplos é dado por Landrieu et al. (2001) que apresentam uma heurística baseada em pesquisa tabu para a resolução do PDPTW com apenas um veículo. Numa primeira fase o algoritmo cria uma rota não necessariamente admissível, mas que cumpre as restrições de precedência e capacidade usando uma heurística de inserção. Numa segunda fase a rota gerada é melhorada através da pesquisa tabu e minimizando a distância percorrida. Os resultados demonstram que para problemas com o número de clientes compreendido entre 10 e 40, o algoritmo alcança soluções admissíveis num tempo razoável. No entanto, se se aumentar o tamanho do problema, os tempos de computação poderão ser superiores.

Também usando a pesquisa tabu Cordeau e Laporte (2003) apresentam uma heurística para resolver o DARP estático, com vários veículos. O modelo considera janelas temporais, tempo máximo de viagem e de rota assim como restrições de capacidade dos veículos. Para analisar o comportamento desta heurística foi gerado um conjunto de 20 instâncias com base real com o número de pedidos a variar entre 24 e 144. Adicionalmente, o algoritmo foi testado com dados reais com um número de pedidos entre 200 e 295. Os resultados obtidos através de uma execução do algoritmo, durante 10^4 iterações, sugerem que este procedimento não só

facilita a identificação de soluções admissíveis, como, em geral, ajuda a melhorar a qualidade das soluções produzidas. Por fim, os autores defendem que comparativamente a outros algoritmos, a pesquisa tabu permite acomodar facilmente um vasto leque de restrições e objectivo, mesmo que não sejam lineares.

Xiang et al. (2006) resolveram uma versão do DARP na qual o objectivo era a minimização de uma combinação entre os custos fixos e variáveis dos veículos, os custos dos condutores, o tempo de espera e de serviço, de acordo com as restrições associadas a este tipo de serviço. A heurística apresentada baseia-se em pesquisa local, com alguma incidência na *tabu search*, e na utilização de uma estratégia de diversificação para melhorar as soluções iniciais que, posteriormente, serão melhoradas através de uma outra estratégia denominada pelos autores de intensificação. Os resultados obtidos revelam que esta heurística é bastante eficiente na obtenção de soluções, independentemente da solução inicial, sendo a rapidez fruto de algumas técnicas de pré-processamento usadas para remover soluções não admissíveis.

Em 2007 Melachrinoudis et al. (2007) descrevem um modelo para um problema de *Dial-a-Ride*, já referido na Secção 4.1.2, para aplicação no *Center for Addictive Behavior Health and Recovery Services*. Das duas abordagens concebidas para a sua resolução, a pesquisa tabu mostrou-se consideravelmente melhor permitindo resolver instâncias com 50 pedidos e 15 depósitos diferentes.

4.2.4. Algoritmos Genéticos

Os algortimos genéticos são algoritmos de pesquisa que se inspiram na evolução natural e em técnicas como a mutação, herança, cruzamento, entre outras. Ainda que estes algoritmos tenham começado a ser estudados por meados dos anos 60 e cedo se tenha percebido que poderiam ser um contributo interessante para a resolução de problemas complexos e de optimização (Fraser, 1970), só mais tarde começaram a ser aplicados a problemas da família dos VRP's. Como exemplos deste facto temos alguns dos artigos (Jih & Hsu, 2004; Pereira et al., 2002) que motivaram Jorgensen et al. (2006) a apresentar um algoritmo genético para a resolução do DARP.

Baseados na clássica abordagem *Cluster First, Route Second*, os autores alternam a construção de *clusters* através de um algoritmo genético com uma versão modificada da *Nearest Neighbor Heuristic* (Baugh et al., 1998) para a construção das rotas. A função objectivo apresentada é multicritério e permite o ajuste dos pesos de sete factores relativos ao conflito entre: custos de operação e a qualidade de serviço. As instâncias de teste usadas foram as

propostas por Cordeau e Laporte (2003). Os resultados obtidos indicam que, se por um lado o algoritmo de pesquisa tabu proposto por Cordeau e Laporte (2003) obtém melhores resultados para a duração das rotas, por outro, esta abordagem consegue melhores resultados no que diz respeito à qualidade de serviço, isto é, ao tempo de viagem dos utilizadores e ao seu tempo de espera, com o veículo parado. De referir que a heurística utilizada na fase de *routing* usa 99% do tempo do CPU, reduzindo assim o número de iterações que o algoritmo genético pode executar e, consequentemente, a possibilidade de obter melhores resultados. Ainda assim os autores consideraram este tipo de abordagem como promissora já que alcançou resultados comparáveis aos métodos *state of the art*.

Em 2009, Lacomme et al. (2009) recorrem a uma estratégia baseada na Pesquisa Local Evolucionária para um DARP com uma frota homogénea, um único depósito, janelas temporais e onde se pretende minimizar os custos de operação melhorando, ao mesmo tempo, a qualidade de serviço. Esta técnica é uma extensão da Pesquisa Local Iterativa e consiste na construção sucessiva de melhores soluções, criadas através de operações de mutação e pesquisa local. Nesta abordagem a população inicial é obtida através de uma heurística ambiciosa (*greedy heuristic)* aleatória que se relaciona com um procedimento que permite obter conjuntos de potenciais soluções não dominadas. As instâncias usadas foram as propostas por Cordeau e Laporte (2003)[1]. Os resultados obtidos provam que, em algumas instâncias, este método consegue competir com a proposta de Cordeau e Laporte (2003). No entanto, os autores reconhecem que a abordagem para a geração da população inicial pode não ser a mais indicada já que consome demasiado tempo de computação.

No mesmo ano, Cubillos et al. (2009) apresentam um estudo onde são usados algoritmos genéticos para a resolução do DARP, numa versão estática, com múltiplos objectivos, janelas temporais e uma frota homogénea que se encontra toda no mesmo depósito. O método apresentado aplica técnicas de pré-processamento de dados que recorre a uma tabela de precedência de eventos e uma lista de incompatibilidade entre clientes. Este pré-processamento reduz o espaço de pesquisa a pontos que sejam admissíveis tendo em conta as restrições impostas pelas janelas temporais. Consequentemente, o algoritmo genético estará menos sujeito a construir soluções não admissíveis, ou seja, soluções onde o veículo chega ao local de destino fora das janelas temporais. À semelhança do trabalho anterior, são usadas instâncias *benchmark* para testar o desempenho desta abordagem comparativamente

[1] Disponíveis *online* em http://neumann.hec.ca/chairedistributique/data/darp/ (Acedido a 23/9/2012)

a abordagens como Jorgensen et al. (2006) e Cordeau e Laporte (2003). Os resultados indicam que esta modelo apresenta uma melhor performance do que o proposto por Jorgensen et al. (2006), em particular no que diz respeito aos tempos de viagem dos utilizadores, das rotas dos veículos, bem como de tempo de CPU. No entanto, comparativamente à abordagem baseada em pesquisa tabu, na maioria dos casos a duração das rotas é mais elevada, mas os tempos de viagem para os utilizadores e o tempo computacional são mais baixos. O facto de os testes terem sido realizados num Intel® Pentium™ 4 2,66 GHz, uma máquina ligeiramente mais potente que as usadas nos casos anteriores, não justifica, por si só, uma diminuição dos tempos de computação, sendo este facto justificado principalmente pelo recurso às técnicas de pré-processamento de dados que diminuíram o espaço de pesquisa.

4.2.5. Outras Metodologias

Como se tem vindo a verificar, ao longo da última década têm surgido várias abordagens para a resolução do *Dial-a-Ride Problem*, algumas já apresentadas numa das categorias acima e outras cuja pertinência para este trabalho não justifica a sua referência ou, simplesmente, não se enquadram na categorização levada a cabo.

Para o segundo caso temos o como o exemplo o trabalho levado a cabo por Calvo e Colorni (2006) que descreve uma heurística bastante rápida, simples e eficaz para resolver o DARP, baseada num método anteriormente proposto por um dos autores (Calvo, 2000). O algoritmo descrito constrói um grafo auxiliar que tem um nó por cliente. Entre cada par de clientes e aos arcos que os ligam, é associado um custo representativo da distância espacial e temporal que os separa. A partir desta estrutura são criadas, através de algumas transformações e relaxações do problema, *n* rotas e alguns *subtours* que constituirão a solução. A função objectivo contempla duas situações de forma hierarquizada, numa primeira fase, o objectivo passa por servir o maior número possível de utentes, seguindo-se, numa segunda fase, a minimização da inconveniência dos mesmos, definida pela soma do tempo de espera com o tempo excessivo de viagem. Os testes foram realizados sobre a rede de Milão, em Itália, constituída por cerca de 7000 nós e 17000 arcos. Os resultados obtidos, para instâncias com o número de utilizadores a variar entre 10 e 180, foram considerados pelos autores como bastante satisfatórios.

Recentemente, Parragh e co-autores desenvolveram um extenso trabalho na resolução de problemas de *Dial-a*-Ride. Num primeiro trabalho, Parragh et al. (2010) consideram uma variante do DARP onde o objectivo consiste em minimizar os custos totais do serviço enquanto

se respeitam os tempos máximos de duração de rota, das viagens dos clientes, bem como as janelas temporais, propondo uma heurística baseada em *Variable Neighborhood Search* (VNS), dado os resultados publicados na área dos VRP's (Polacek et al., 2004; Parragh et al., 2009). Nesta implementação são usadas técnicas de pré-processamento à semelhança do que acontece em Cordeau (2006). A solução inicial também se baseia numa criação dos mesmos autores (Cordeau & Laporte, 2003) mas sendo agora construída de forma completamente aleatória, levando em conta a proximidade temporal e espacial. Os resultados indicam que esta abordagem não conseguiu obter, em média, melhores resultados que a pesquisa tabu de Cordeau e Laporte (2003). No entanto se se considerarem apenas as melhores soluções e todas as configurações de parâmetros, foram melhoradas 16 soluções. Comparativamente aos algoritmos genéticos propostos por Jorgensen et al. (2006), esta abordagem conseguiu melhoramentos notáveis, na ordem dos 70%.

Um ano mais tarde, Parragh (2011) introduz heterogeneidade a vários níveis para a versão base/standard do DARP proposta por Cordeau (2006), motivada pelas observações feitas na *Austrian Red Cross* (ARC) no âmbito do transporte de pacientes. Uma particularidade deste modelo e pouco referido na literatura é considerar explicitamente o tempo de espera a bordo do veículo, o que é psicologicamente extenuante para os utilizadores. Assim, a função objectivo contempla uma penalização para o tempo a bordo das viaturas, sendo preferível que os utilizadores esperem fora das viaturas no período de tempo dado pela janela temporal. Os autores propõem duas formulações para o problema apresentado, sendo que numa delas se assume que os custos e tempos de viagem dos veículos são homogéneos e adaptam a heurística VNS, desenvolvida anteriormente, ao novo problema. À semelhança de Baldacci et al. (2009) são introduzidos depósitos artificiais de modo a contemplar a heterogeneidade da frota. A abordagem heurística apresentou baixos tempos de execução e conseguiu resultados médios, para cinco execuções aleatórias, considerados pelos autores como bastante interessantes. De realçar que os testes realizados com e sem penalização revelaram que evitar o tempo de espera dos clientes aumenta, em média, os custos na ordem dos 2,5%, com um aumento máximo de 6%.

4.3. Conclusões

Neste capítulo foi dada uma visão geral do trabalho que tem sido realizado na área dos *Vehicle Routing Problems* com especial incidência no *Dial-a-Ride* e nas abordagens que têm vindo a ser utilizadas para a sua resolução.

O facto de já existirem diversas abordagens aproximadas e de, aparentemente, o seu sucesso estar comprovado levou a que o trabalho realizado no âmbito desta dissertação se focasse nas abordagens exactas que mesmo não tendo tantas provas dadas como as aproximadas, têm vindo a mostrar bons indicadores, inclusive, no contexto da delegação Amadora – Sintra da Cruz Vermelha Portuguesa e na optimização de custos associados ao serviço, sem desprimor pela qualidade do mesmo.

Deste modo nos próximos capítulos serão abordadas duas das técnicas usadas pelas abordagens exactas: Programação Linear Inteira e Programação com Restrições. A primeira será contemplada pelo facto de ser a técnica com mais sucesso neste âmbito, ainda assim o foco incidirá na Programação com Restrições devido ao pouco trabalho publicado neste contexto e motivado pelos resultados recentes obtidos por Berbeglia et al. (2010).

5

5. Programação Linear

A Programação Linear é um método matemático para, num dado problema, determinar uma forma de alcançar o melhor resultado de acordo com uma determinada métrica, usualmente o lucro máximo ou o custo mínimo. Mais formalmente pode definir-se a Programação Linear como uma técnica de optimização de funções objectivo lineares, sujeitas a um conjunto de equações ou inequações lineares que restringem o problema (Hillier & Lieberman, 2001).

Ainda que o desenvolvimento desta técnica tenha sido considerado um dos maiores avanços científicos da segunda metade do século XX, a sua origem remonta a 1939 e é atribuída ao matemático russo Leonid Vitaliyevich Kantorovich (Kantorovich, 1940). Foi utilizada durante a segunda grande guerra com a finalidade de planear os investimentos de modo a reduzir os custos com o exército e aumentar as perdas do inimigo. Durante o conflito o método permaneceu secreto tendo surgido, após o seu término, diversos desenvolvimentos e aplicações em diferentes indústrias (Waugh, 1951; Charnes et al., 1953).

Actualmente a Programação Linear é considerada uma ferramenta que tem permitido a muitas empresas poupar milhões de euros. O seu uso tem sido difundido para diversos sectores da sociedade e tem crescido consideravelmente (Hillier & Lieberman, 2001).

A Programação Linear é usada para resolver problemas que envolvem a distribuição de recursos limitados da melhor forma possível entre actividades concorrentes. Este problema é normalmente bastante difícil de resolver dada a existência de diversas actividades que, para funcionarem, competem por recursos que são escassos. A Programação Linear vem dar um contributo a essa questão auxiliando a afectação dos referidos recursos pelas diferentes actividades. Esta afectação não é feita de forma aleatória mas sim, de modo a tirar o máximo partido dos recursos. Esta metodologia pode ser aplicada a um número muito elevado de

problemas enfrentados na vida real. Dada a diversidade e a dimensão da grande maioria dos problemas, os modelos de Programação Linear também diferem bastante pelo que existem muitos métodos de resolução assim como linguagens de modelação de modo a tornar a sua resolução mais eficiente. Como exemplo de linguagem podemos referir AMPL, MPL, GAMS, LINGO, entre outras (Hillier & Lieberman, 2001).

5.1. Multiobjectivo

Uma extensão da Programação Linear (tradicionalmente com um único objectivo) é a Programação Linear Multi Objectivo (PLMO), também conhecida como optimização multiobjectivo, que pretende optimizar simultaneamente dois ou mais objectivos, normalmente conflituosos, mediante um conjunto de restrições (Ehrgott & Gandibleux, 2002). Sendo uma extensão da Programação Linear é igualmente aplicável em problemas reais, onde se poderá, por exemplo, minimizar o consumo de combustível de um automóvel e, ao mesmo tempo, maximizar a sua performance, ou maximizar o lucro ao mesmo tempo que se minimizam custos de produção. No entanto, e curiosamente, esta questão da satisfação de diferentes objectivos em simultâneo foi introduzida por Vilfredo Pareto no final do século XIX, ainda antes da existência da Programação Linear (Ehrgott, 2005).

Os problemas multiobjectivo não são de resolução trivial uma vez que a optimização de um dos objectivos conduz a valores não óptimos para os restantes, pelo que não é comum encontrar uma única solução que optimize todos os objectivos (Clímaco et al., 2003). Assim a resolução de problemas de optimização multiobjectivo passa por avaliar várias possíveis soluções, os seus *trade-offs* e dotar o decisor, que em última instância é ou tem influência humana, da informação necessária para optar por uma das alternativas.

Na literatura são propostas diversas abordagens para a resolução deste tipo de problemas, entre as quais se destacam modelos com somas agregadas, com vectores de pesos, modelos de redução da região admissível, entre outras. Dada a natureza conflituosa dos objectivos destes problemas e da complexidade que daí advém, não existem só metodologias para obter as melhores soluções, mas também para a tomada de decisão, existindo, inclusive, áreas da Investigação Operacional dedicadas exclusivamente à tomada de decisões em contexto multiobjectivo (Hillier & Lieberman, 2001).

Os problemas de Programação Linear Multi Objectivo são um caso especial da Programação Multi Objectivo que surge devido à linearidade de todas as funções objectivo,

bem como de todas as restrições. Deste modo este tipo de problemas pode ser apresentado da seguinte forma:

$$\begin{aligned}
&\text{Max } Z_1 = c_{11}X_1 + c_{12}X_2 + ... + c_{1n}X_n \\
&\text{Max } Z_2 = c_{21}X_1 + c_{22}X_2 + ... + c_{2n}X_n \\
&... \\
&\text{Max } Z_k = c_{k1}X_1 + c_{k2}X_2 + ... + c_{kn}X_n
\end{aligned}$$

Sujeito a:

$$\begin{aligned}
&a_{11}X_1 + a_{12}X_2 + ... + a_{1n}X_n = b_1 \\
&... \\
&a_{m1}X_1 + a_{m2}X_2 + ... + a_{mn}X_n = b_m
\end{aligned}$$

$$X_1, X_2, ..., X_n \geq 0$$

Como referido anteriormente, a natureza conflituosa dos objectivos em PLMO não permite a existência de uma solução óptima como entendido na programação com apenas um objectivo. No entanto, existe o conceito de solução eficiente. Num problema com *k* objectivos uma solução X diz-se eficiente se, e só se, não existir nenhuma solução *X'*, tal que: $Z_i(X') \geq Z_i(X), \forall i \in \{1,2,...,k\} \setminus \{p\}$ e $Z_p(X') > Z_p(X)$, sendo $Z_i(X)$ o valor da variável X considerando o objectivo *i*. Ou seja, uma solução diz-se eficiente caso não exista uma outra solução admissível (solução que verifica todas as restrições) que seja igual ou melhor, nas *k* funções objectivo, e estritamente melhor em pelo menos uma delas (Sawaragi et al. 1985). Assim, em vez de ter um único ponto como solução, a solução dos problemas com múltiplos objectivos consiste num conjunto, geralmente infinito, de soluções eficientes.

Depois de conhecidas todas as soluções eficientes, e os respectivos valores nos diferentes objectivos, define-se a chamada Frente de Pareto. Sobre ela, é necessário tomar decisões de modo a optar uma solução. Segundo Clímaco et al. (2003) existem diferentes tipos de abordagens: o método lexicográfico, o método da soma ponderada, o método das restrições, o método STEM, entre outros. A título de exemplo destacamos apenas as duas primeiras. No método lexicográfico, as funções objectivo são ordenadas de acordo com as preferências do decisor, seguindo-se a optimização de cada função objectivo pela ordem estabelecida. Em cada passo optimiza-se apenas um objectivo e, a partir do valor óptimo obtido, adiciona-se uma restrição ao problema que condicionará a resolução dos problemas seguintes. Em muitas situações, a resolução dos primeiros objectivos conduz a regiões admissíveis vazias, pelo que a solução encontrada não considera os restantes objectivos. Um segundo método é a soma ponderada onde um problema com múltiplos objectivos, é transformado num problema com um

só objectivo: a soma pesada dos *k* objectivos do problema original recorrendo a pesos $\lambda_1,...,\lambda_k$, tais que $\sum_{i=1}^{k}\lambda_i = 1$ e $\lambda_i \geq 0$, *i* = 1,…,*k*. Deste modo a função objectivo a optimizar será $\text{Max } Z = \lambda_1 Z_1 + ... + \lambda_k Z_k$.

5.2. Modelo do *Dial-a-Ride Problem*

Têm sido várias as propostas para o *Dial-a-Ride Problem* baseadas na metodologia da Programação Linear, destacando-se propostas como as de Cordeau e Laporte (2007) ou Ropke et al. (2007). Ambas apresentaram uma formulação baseada num grafo completo e direccionado $G = (V, A)$, onde *V* é o conjunto de todos os vértices, também denominados de nós, e *A* o conjunto de todos os arcos do grafo.

Na versão base do problema, existe um conjunto *K*, com *k* veículos homogéneos que devem servir um conjunto de *n* pedidos de transporte, ou seja, um conjunto de *n* tuplos

$$(utente, origem, destino, hora) \tag{5.2.1}$$

O conjunto de vértices *V* é particionado em $\{\{0, 2n+1\}, P, D\}$, onde 0 e 2*n* + 1 representam o local de onde as viaturas partem e ao qual devem chegar no final do serviço. O conjunto $P = \{1,...,n\}$ representa os vértices de origem e $D = \{n+1,...,2n\}$ os vértices de destino. Cada veículo começa a rota no depósito de origem, denominado por 0, e terminará em 2*n* + 1, no depósito de destino, com um limite máximo *T* para a duração da rota. Note-se que muitas vezes o depósito de origem e destino são o mesmo local. Cada arco (i, j) tem associado um custo de viagem c_{ij} e um tempo de viagem t_{ij}, ambos não negativos. Por outro lado, cada pedido é identificado por um par de vértices $\{i, n+i\}$, onde $i \in P$ e $n+i \in D$, e, para cada vértice $v_i \in V$, existe uma carga associada q_i, com $q_0 = q_{2n+1} = 0$, $q_i \geq 0$ e $q_{n+i} = -q_i$, $\forall i \in [1, n]$, e uma duração de serviço $d_i \geq 0$ com $d_0 = d_{2n+1} = 0$. Finalmente, a capacidade de cada veículo é dada por *Q*, no caso dos veículos homogéneos, e o tempo máximo de viagem para um passageiro é dado por *L*, de modo a garantir um determinado nível de qualidade do serviço. Ainda no que diz respeito à qualidade do serviço e ao tempo que os utentes usufruem deste, existe um parâmetro que consiste numa janela temporal que indica, para a recolha ou entrega, o intervalo de tempo em que a operação deve ocorrer, $[e_i, l_i]$.

Formalmente o conjunto de todos os vértices do grafo *G*, é dado por $V = P \cup D \cup \{0, 2n+1\}$ e o conjunto de todos os arcos interessantes para o problema por $A = \{(i, j) : i = 0, j \in P \vee (i, j \in P \cup D, i \neq j \wedge i \neq n+j) \vee i \in D, j = 2n+1\}$.

O modelo considera como variáveis, uma variável binária x_{ij}, que toma o valor 1 caso o arco (i,j) seja atravessado por um veículo, 0 caso contrário. Uma variável B_i que representa o tempo em que se inicia o serviço no vértice *i*, Q_i representa o número de passageiros a bordo do veículo depois de visitar o nó *i* e L_i o tempo de viagem do utente *i*, correspondente ao pedido $(i, n+i)$.

Deste modo pretende-se minimizar o custo total das rotas e pode ser expresso da seguinte forma:

$$\text{Min}\left(\sum_{i\in V}\sum_{j\in V} c_{ij}x_{ij}\right) \tag{5.2.2}$$

Sujeito a:

$$\sum_{i\in P, j\in V} x_{ij} = 1 \tag{5.2.3}$$

$$\sum_{j\in V} x_{0j} = \sum_{i\in V} x_{i,2n+1} = 1 \tag{5.2.4}$$

$$\sum_{i\in P, j\in V} x_{ij} - \sum_{i\in P, j\in V} x_{n+i,j} = 0 \tag{5.2.5}$$

$$\sum_{i\in P\cup D, j\in V} x_{ji} - \sum_{i\in P\cup D, j\in V} x_{ij} = 0 \tag{5.2.6}$$

$$B_j \geq \left(B_i + d_i + t_{ij}\right)x_{ij}, \text{ com } i,j \in V \tag{5.2.7}$$

$$Q_j \geq \left(Q_i + q_j\right)x_{ij}, \text{ com } i,j \in V \tag{5.2.8}$$

$$L_i \geq B_{n+i} - \left(B_i + d_i\right), i \in P \tag{5.2.9}$$

$$B_{2n+1} - B_0 \leq T \tag{5.2.10}$$

$$e_i \leq B_i \leq l_i, i \in V \tag{5.2.11}$$

$$t_{i,n+i} \leq L_i \leq L, i \in P \tag{5.2.12}$$

$$\max\{0, q_i\} \leq Q_i \leq \min\{Q, Q+q_i\}, i \in V \tag{5.2.13}$$

$$x_{ij} = 0 \vee 1, \text{ com } i,j \in V \tag{5.2.14}$$

Nesta formulação as restrições (5.2.3) e (5.2.5) garantem que cada pedido é servido apenas uma vez pelo mesmo veículo, enquanto as restrições (5.2.4) e (5.2.6) garantem que cada veículo começa e termina a rota nos respectivos depósitos inicial e final. Por outro lado as

restrições (5.2.7) e (5.2.9) lidam com os tempos de serviço, com a carga dos veículos e com os tempos de viagem dos utentes, sendo complementadas pelas (5.2.10) e (5.2.13) que garantem que as rotas geradas são admissíveis tendo em conta a capacidade dos veículos e o limite máximo de tempo para uma rota ser efectuada.

5.2.1. Extensões

A formulação anterior considera implicitamente a modelação de uma frota homogénea constituída por um ou mais veículos. No entanto, muitas instâncias do *Dial-a-Ride Problem* são compostas por frotas não homogéneas. Para contemplar esta situação diversos parâmetros têm que ser redefinidos, sendo que, a título de exemplo, o facto de se contemplar uma frota onde os veículos têm diferentes características obriga a que se considere um novo índice $k \in K$, que permite identificar cada um dos veículos que compõem a frota. Por outro lado o custo da viagem associada ao arco (i, j) também será redefinido para c_{ij}^{k}, permitindo reflectir a heterogeneidade da frota. Além dos parâmetros, todas as variáveis da versão base do problema terão que sofrer alterações para contemplar a frota, agora, heterogénea (Parragh, 2011).

Esta heterogeneidade da rota pode não ser apenas ao nível dos consumos de combustível, da potência e, consequentemente, velocidade de ponta dos veículos, mas também relativamente ao tipo de lugares disponíveis para transporte dos clientes do problema em causa. Como já referido neste documento, estes poderão ter necessidade especiais de transporte, como por exemplo, transporte de cadeiras de rodas, em macas, em cadeiras apropriadas para crianças, entre outros. Assim a variável Q_i que modela o número de passageiro a bordo do veículo após o vértice *i* ser servido, deverá ser reformulada de modo a ser possível diferenciar não só o veículo como também os diferentes tipos de lugar: $Q_i^{r\,k}$, com $k \in K$ e *r* sendo o índice relacionado com as diferentes necessidades especiais contempladas pelo veículo em causa, e.g., *r* = 0 pode simbolizar lugares sentados, *r* = *1*, lugares em cadeiras de rodas e *r* = 2 em maca.

Por fim, mas não menos importante, pode ter-se uma preocupação acrescida com a qualidade do serviço muitas vezes modelada por um factor de penalização na função objectivo (Paquette et al., 2009). Neste modelo em particular tal factor poderia ser aplicado através da criação de uma nova variável W_i^k que representa o custo do tempo de espera do veículo *k*, com passageiros a bordo, no vértice *i*. Assim a função a minimizar passaria a ser representada da seguinte forma:

$$\sum_{k\in K}\sum_{i\in V}\sum_{j\in V} c_{ij}^{k} x_{ij}^{k} + \sum_{k\in K}\sum_{i\in P\cup D} W_i^k \tag{5.2.15}$$

Ao modelo apresentado teriam que ser adicionadas algumas restrições e uma variável que indicasse o tempo de chegada de um dado veículo a um local já que agora este tempo pode não corresponder ao tempo de início do serviço devido ao possível tempo de espera. Deste modo considere-se a variável A_i^k a variável que indica tal momento, com $i \in V$ e $k \in K$, que iria dar origem a restrições como:

$$W_i^k \geq 0 \wedge W_i^k \geq B_i^k - A_i^k, \forall i \in V, k \in K \tag{5.2.16}$$

$$B_i^k \geq A_i^k, \forall i \in V, k \in K \tag{5.2.17}$$

$$x_{ij}^k = 1 \Rightarrow A_j^k = B_i^k + d_i + t_{ij}^k, \forall i, j \in V, k \in K \tag{5.2.18}$$

De notar que a primeira restrição traduz o tempo de espera como o intervalo de tempo entre o início do serviço num dado vértice e o tempo a que a viatura chegou ao local, sendo o primeiro sempre superior ou igual ao segundo, como referido na segunda restrição. A última restrição mapeia o tempo de chegada a um dado vértice como a soma do tempo de início do serviço no seu antecessor, mais a duração do serviço nesse mesmo local e o tempo de viagem entre os locais em causa. Apesar da restrição apresentada não ser linear, é possível a sua linearização com recurso a variáveis binárias.

.

6

6. Programação com Restrições

A Programação com Restrições (Lauriere, 1978) é um poderoso paradigma para a resolução de problemas de pesquisa e de combinatória que recorre a técnicas de diversas áreas, das quais se destacam, a Inteligência Artificial, as Bases de Dados, as Linguagens de Programação, a Ciência da Computação, bem como Investigação Operacional. A sua aplicação está cada vez mais difundida, sendo que actualmente domínios como o planeamento, agendamento, definição de rotas para veículos, redes e bioinformática são provas do sucesso deste paradigma (Rossi et al., 2006).

A base desta abordagem passa pela modelação de um dado problema através da definição das restrições que lhe estão associadas, sendo estas definidas através das relações entre as variáveis do problema. O modelo obtido é inserido num resolvedor que, por sua vez, irá explorar o espaço de soluções do problema de modo a encontrar valores para todas as variáveis que satisfaçam as restrições impostas.

Deste modo é comum referir-se que a Programação com Restrições é um paradigma que mistura as abordagens matemática e computacional já que são declaradas variáveis, domínios e restrições que o resolvedor terá que satisfazer. No entanto, para que o processo de resolução não se torne ineficiente, é também necessário delinear estratégias de pesquisa.

Classicamente um Problema de Satisfação de Restrições, em inglês *Constraint Satisfaction Problem* (CSP), é definido por um triplo (X,D,C) onde X é um tuplo de n variáveis, $X = \langle X_1, X_2, ..., X_n \rangle$, D é o tuplo dos domínios correspondentes $D = \langle D_1, D_2, ..., D_n \rangle$ tal que $X_i \in D_i$. Por fim C é um tuplo de restrições $C = \langle C_1, C_2, ..., C_m \rangle$, no qual cada restrição C_j pode ser vista como um par $\langle R_{S_j}, S_j \rangle$ onde R_{S_j} representa a relação entre as variáveis presentes em S_j, sendo este último o escopo da restrição C_j. Por outras palavras, R_i é um subconjunto do

produto cartesiano dos domínios das variáveis presentes em S_i. Uma solução deste CSP é um tuplo $A = \langle A_1, A_2, ..., A_n \rangle$ onde $A_i \in D_i$ e cada restrição C_j é satisfeita de modo a que R_{S_j} mantenha a projecção de *A* para o escopo de S_j. Se o conjunto de soluções for vazio, o problema não é satisfazível (Marriott & Stuckey, 1998; Rossi et al., 2006).

Caso a dimensão do escopo de cada restrição seja limitada a 1 ou 2, então as restrições dizem-se unárias ou binárias, respectivamente, podendo, deste modo, representar o CSP através de um grafo de restrições onde os vértices simbolizam as variáveis e os arcos, as restrições. Por outro lado, se as restrições forem *n*-árias, com *n* > 2, o problema pode ser representado através de um hipergrafo.

Ainda que este *framework* seja simples e tenha uma grande abrangência, pode ser especializado ou generalizado de diversas formas. Em particular, se o problema a resolver não é apenas de satisfação mas também de optimização, acrescentar-se-á a minimização/maximização de uma ou mais funções objectivo.

Rossi et al. (2006) consideram que os algoritmos para resolução de CSP's se inserem numa de duas categorias: inferência e pesquisa, sendo que por diversas vezes se combinam ambas as abordagens. Dados os domínios das diversas variáveis do problema, o espaço de soluções poderia ser "facilmente" enumerado e cada combinação seria testada para determinar se constitui ou não uma solução. No entanto, esta abordagem pode tornar-se demasiado pesada, pelo que a inferência e a pesquisa pretendem melhorar a procura de soluções. Inevitavelmente a combinação de ambas as técnicas tem um *trade-off* associado pois nem sempre o custo computacional gasto na redução do problema é proporcional à redução conseguida no tempo total de resolução do problema, ainda que quanto mais reduzido estiver um problema, mais fácil este será de resolver. Esta correlação entre a pesquisa e a inferência pode ser expressa através do gráfico que se apresenta na Figura 6.1.

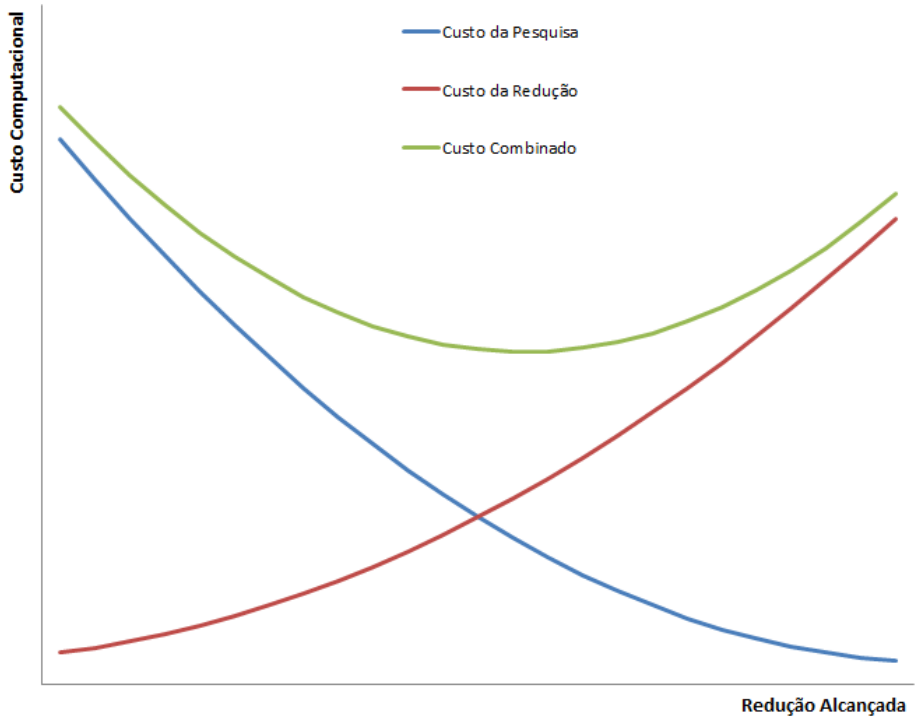

Figura 6.1 - *Trade-off* entre a Pesquisa e a Redução.

6.1. Pesquisa

Existem dois tipos principais de algoritmos de pesquisa utilizados na resolução de Problemas de Satisfação de Restrições: os completos, também conhecidos como sistemáticos, e os incompletos (Rossi et al., 2006). Os primeiros garantem que encontrarão uma solução, caso esta exista, pelo que podem ser também usados para mostrar que um problema não tem soluções, ou mesmo para encontrar uma solução comprovadamente óptima. Os algoritmos ditos incompletos, ou não sistemáticos, não têm capacidade para demonstrar que o problema a resolver tem soluções nem, tão pouco, que uma solução é óptima. No entanto, este tipo de algoritmos pode ser eficiente na procura de uma solução, caso esta exista, bem como na aproximação a uma solução óptima. Como exemplos de algoritmos completos temos o retrocesso, em inglês *backtracking*, e a programação dinâmica, sendo os métodos de pesquisa local e de pesquisa estocástica exemplos de algoritmos incompletos.

6.1.1. Retrocesso

O retrocesso é um algoritmo para encontrar todas, ou apenas algumas, soluções de problemas de satisfação de restrições. O facto de se inserir na categoria dos algoritmos completos garante

que, caso exista uma solução, esta será encontrada. Resumidamente este método constrói uma solução parcial através da escolha de valores para as variáveis até que chega a “um beco sem saída”, ou seja, a solução parcial não pode ser estendida às restantes variáveis. Chegado a este ponto, é desfeita a última escolha e tenta-se outra. Este procedimento é realizado de forma sistemática de modo a garantir que todas as possibilidades são testadas (Dechter, 2003). Esta é considerada uma das virtudes deste algoritmo pois, ainda que originalmente não seja dotado de capacidades de aprendizagem e noções de progresso, a sua exaustividade acaba por melhorar o “simples” cálculo de todas as possibilidades por “força bruta” já que tem a capacidade de analisar a satisfação das restrições cada vez que faz uma nova escolha, não necessitando de esperar pelo cálculo da toda a solução candidata (Rossi et al., 2006). Esta última técnica apelidada por “força bruta” é bastante comum na programação dinâmica, o que a coloca num segundo plano devido à sua necessidade de quantidades exponenciais de tempo e espaço, contrastando com a quantidade polinomial de espaço requerida pelo retrocesso (Marriott & Stuckey, 1998).

Conceptualmente o espaço de soluções de um CSP pode ser representado através de uma árvore de pesquisa. Este tipo de estrutura é composto por diversos nós e por ramos, sendo estes últimos que compõem as ligações entre os nós. É uma estrutura hierárquica pelo que existe um nó que não tem superior e que constitui a raiz da árvore, bem como um conjunto de nós que não têm inferiores e são chamados de folhas. Tipicamente os nomes das relações entre os nós remetem para as relações familiares pelo que um nó acima é apelidado de “pai”, se estiver abaixo é um nó “filho” e, ao mesmo nível, “irmão”.

No nosso caso em particular, a raiz representa um conjunto vazio de atribuições a variáveis já que nenhuma destas se encontra instanciada. Cada nó abaixo da raiz representa a escolha de um valor para uma determinada variável e, cada ramo, representa uma candidata a solução parcial. Deste modo, em cada nó da árvore de pesquisa, uma variável não instanciada é escolhida e os ramos que saem desse nó definem todas as possibilidades de extensão para a variável em causa. O algoritmo de retrocesso, em inglês *backtracking*, verifica, em cada nó, se alguma das restrições cujas variáveis já estão instanciadas não é satisfeita e, caso esse cenário aconteça, o algoritmo irá escolher outro valor do domínio para a variável em causa. Neste contexto, a descoberta de uma solução parcial que não pode ser estendida corresponde ao corte da subárvore em causa.

Para que todos estes conceitos se possam assimilar de uma forma mais intuitiva, considere-se o famoso problema das N-Rainhas de Max Bezzel (Madachy, 1979) no qual se

pretendem colocar *N* rainhas, num tabuleiro de xadrez $N \times N$, sem que estas se ataquem mutuamente, isto é, não estejam nem na mesma linha, nem na mesma coluna, nem, tão pouco, em diagonais iguais. A árvore de pesquisa gerada, bem como o funcionamento do algoritmo de retrocesso, para a instância de *N* = 4 segue representada pela Figura 6.2.

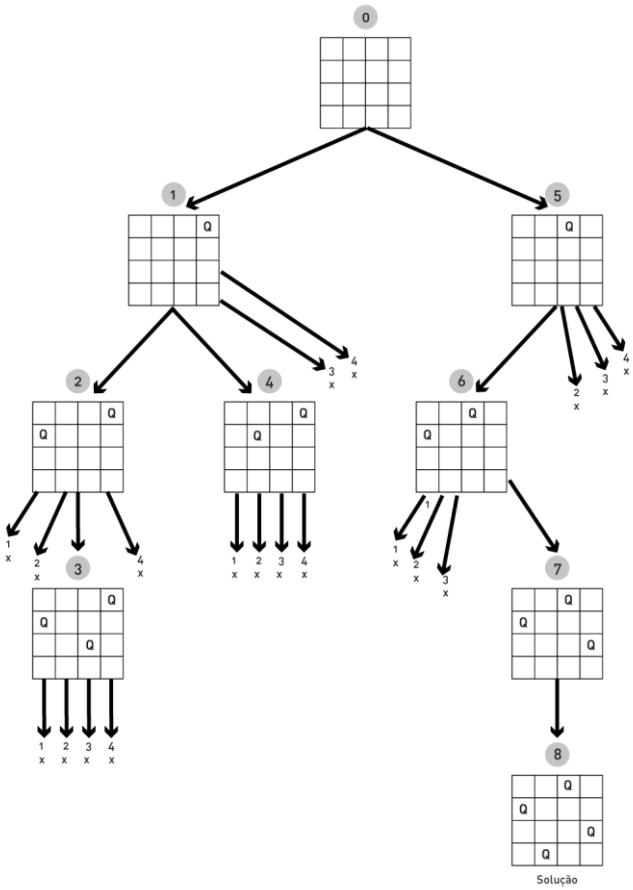

Figura 6.2 - Árvore de Pesquisa para o problema das 4-Rainhas.

Numa primeira fase o algoritmo de retrocesso irá explorar a opção em que a primeira rainha se encontra na quarta coluna, sendo que para a rainha que se segue irá fazer atribuições sequenciais, começando na primeira coluna e mudando à medida que se detecta uma falha, pelo que apenas consegue colocar a segunda rainha na primeira ou segunda colunas. As restantes tentativas têm uma dinâmica idêntica, sendo que na quarta solução potencial, quando se coloca a segunda rainha na segunda coluna, se chega à conclusão que com a primeira rainha na quarta coluna o problema não tem soluções. É realizado um retrocesso e a primeira rainha é colocada na terceira coluna, conseguindo assim alcançar uma solução, como demonstrado pela oitava solução potencial.

6.1.2. Heurísticas

Quando recorremos ao retrocesso para resolver Problemas de Satisfação de Restrições, existem diversas decisões que devem ser tomadas e que, de uma forma ou de outra, poderão influenciar a solução obtida. Por norma estas decisões passam pela escolha de quais os ramos a explorar e, consequentemente, pela escolha dos valores a atribuir às variáveis que lhes estão associadas, escolhas que podem ser cruciais para a resolução eficiente do problema (Rossi et al., 2006). Para que a ordem pela qual as variáveis ou os valores são escolhidos se diga óptima, é necessário que esta garanta que é visitado o menor número de nós da árvore de pesquisa, quando o problema tem solução, ou, caso contrário, demonstrar que tal não ocorre. No entanto, tais ordenações óptimas são extremamente difíceis de descobrir, sendo que é tão ou mais complicado decidir qual a primeira variável a ser escolhida do que descobrir se o CSP em causa tem, ou não, solução (Liberatore, 2000) .

Aqui entram em jogo a heurísticas, métodos baseados em experiência, intuição, bom senso ou outro tipo sustento sem garantia formal, matemática ou científica, cujo objectivo passa por descobrir qual a melhor decisão a tomar (Pearl, 1984).

No contexto da pesquisa por retrocesso o foco incidirá nas heurísticas para a escolha das variáveis já que estas, ao contrário das de escolha de valores, poderão afectar a dimensão da árvore de pesquisa. Para esta tomada de decisão as heurísticas podem classificar-se como estáticas ou dinâmicas (Rossi et al., 2006), sendo que nas primeiras a ordenação das variáveis é feita ainda antes da pesquisa, descartando possíveis efeitos da inferência. Por outro lado, na abordagem dinâmica, a escolha da variável é determinada com base nas atribuições já realizadas e em inferência. Por norma a abordagem estática é baseada na estrutura do problema, representada através de um grafo, e a dinâmica no tamanho do domínio das variáveis.

Muitas das heurísticas dinâmicas baseiam-se num princípio denominado *First-Fail* (Haralick & Elliott, 1980) que defende que para ter sucesso é importante tentar enquadrar as variáveis mais complicadas primeiro, onde a probabilidade de falhar é superior. Uma das heurísticas mais conhecidas, *DOM Heuristic* (Rossi et al., 2006), escolhe as variáveis cujo domínio é menor, ou seja, escolhe a variável à qual é mais difícil fazer atribuições já que o leque de valores é menor. A utilidade desta heurística pode também ser vista da perspectiva de que havendo menos valores disponíveis, a probabilidade de o valor escolhido pertencer à solução é maior. Como complemento a esta heurística, existem outras que se baseiam, por

exemplo, no número de restrições que envolvem uma variável e que tipicamente se tornam importantes em situação de empate.

No que diz respeito à ordenação guiada pela estrutura, o seu funcionamento assenta na representação do problema na forma de grafo e nas suas particularidades, em particular em conceitos como o seu *width* e *bandwidth*. Algumas destas heurísticas recorrem à técnica *Divide and Conquer* (Knuth, 1998) que decompõe o grafo usando separadores, isto é, retirando um conjunto de arcos que separam o grafo em dois grafos disjuntos, reduzindo o problema. Outras técnicas sugerem que a instanciação de variáveis que cortar ciclos no grafo deve ser imediata.

Nas heurísticas para a escolha de valores pretende-se escolher o valor que tenha mais probabilidade de ter sucesso ou de fazer parte da solução (Dechter, 2003). Obviamente que se estas informações fossem conhecidas de antemão, seria facílimo gerar uma ordenação óptima. No entanto, não possuindo tal informação, recorre-se, por norma, a aproximações acerca do número de soluções concebidas através de relaxações da árvore de pesquisa ou mesmo usando conceitos de Redes Bayesianas. Algumas das heurísticas mais conhecidas passam pela maximização do produto e/ou da soma dos tamanhos dos domínios após a instanciação, denominadas por *promise* e *min-conflicts*, respectivamente (Rossi et al., 2006).

6.1.3. Algoritmos Incompletos

Por outro lado existem os algoritmos incompletos que, ainda que não sejam exaustivos como os apresentados acima, acabam por constituir um paradigma fundamental para resolver problemas de combinatória como os CSP's, sendo muitas vezes a única abordagem exequível para a resolução de instâncias complexas (Rossi et al., 2006). Em particular, a pesquisa local assenta sob o conceito base de começar a operar numa solução candidata gerada aleatoriamente ou através de alguma heurística, independentemente da sua optimalidade. Segue-se um processo iterativo cujo objectivo é melhorar a solução candidata através de pequenas modificações, existindo métodos para assegurar que o processo não fica estagnado em soluções candidatas insatisfatórias. De entre os métodos de pesquisa local, conhecidos também como metaheurísticas, destacam-se o *Hill-Climbing*, *Simulated Annealing*, *Tabu Search*, Algoritmos Evolucionários/Genéticos e *Ant Colony* (Dechter, 2003; Marriott & Stuckey, 1998; Rossi et al., 2006). Esta classe de algoritmos de pesquisa local é, tipicamente, fácil de implementar, além de demonstrar uma performance bastante interessante em diversos problemas.

6.1.4. Optimização

Em geral os algoritmos de pesquisa local são naturalmente apropriados para lidar com critérios de optimização que surgem em diversos problemas da vida real já que muitas vezes conseguem descobrir soluções com bastante qualidade de uma forma muito mais eficiente que as outras abordagens (Dechter, 2003).

Aos Problemas de Satisfação de Restrições acrescenta-se, tipicamente, uma função objectivo que deve ser optimizada. Uma das abordagens mais divulgadas passa por uma versão baseada em restrições do algoritmo de *Branch-and-Bound* (Land & Doig, 1960) na qual tipicamente se obtém uma solução que satisfaça as restrições impostas recorrendo a *backtracking* ou a pesquisa local. De seguida adiciona-se uma nova restrição ao problema de modo a excluir as soluções que não são melhores que a obtida, sendo este processo repetido até que o problema se torne insatisfazível, com a última solução obtida como óptima (Podelski, 1995).

Independentemente dos algoritmos e heurísticas usadas, para que estas abordagens se tornem realmente eficientes, é importante que seja aplicada inferência à resolução do problema, em particular técnicas de propagação de restrições que acabam por ser um complemento fundamental a muitos dos algoritmos apresentados (Rossi et al., 2006).

6.2. Inferência

Como se pode perceber através da análise do algoritmo de retrocesso, o seu *modus operandi* demonstra, na maioria das vezes, comportamentos indesejáveis já que se explora repetidamente subárvores da árvore de pesquisa que irão falhar por serem diferentes apenas em variáveis irrelevantes para o insucesso das mesmas. Deste modo o principal objectivo da inferência passa pela identificação e eliminação de muitos destes comportamentos indesejados através de técnicas como a propagação de restrições, sendo a responsabilidade da sua aplicação dos resolvedores (Rossi et al., 2006).

O conceito da inferência neste contexto, e da propagação de restrições em particular, assenta na dedução de que certos valores não poderão constar no domínio de certas variáveis porque se forem atribuídos vão violar alguma restrição do problema. Assim, consegue reduzir-se o espaço de procura descartando soluções impossíveis, através da eliminação de valores obsoletos dos domínios das variáveis envolvidas (Azevedo, 2003; Rossi et al., 2006). No caso em que o domínio de uma variável fica vazio devido à propagação de restrições, pode dizer-se

que a instanciação feita nunca irá constituir uma solução, ou pode mesmo indicar que o problema não tem soluções.

De modo a ilustrar o conceito, recordemos o problema das N-Rainhas, para a instância $N = 4$.

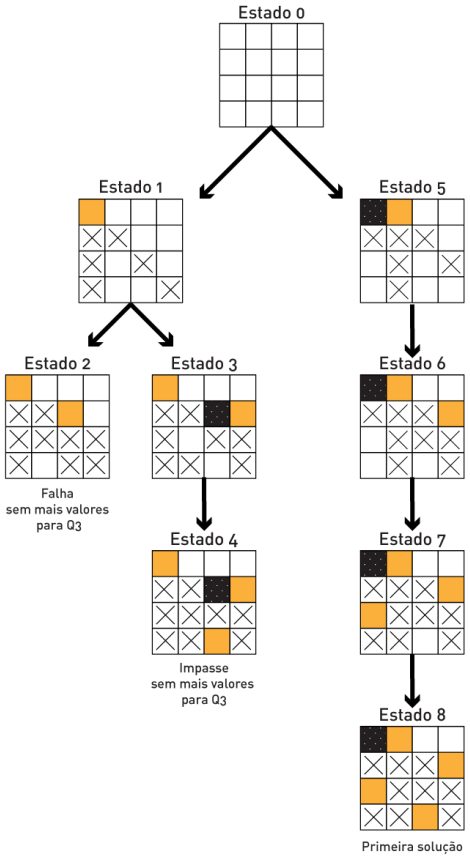

Figura 6.3 - Propagação de Restrições para o problema 4-Rainhas.

Como se pode perceber através da Figura 6.3, a propagação de restrições acaba por simular o raciocínio humano pois, a partir do momento em que se coloca a primeira rainha na primeira coluna, conhecendo as regras do problema, é automático pensar que a segunda rainha não poderá estar nem na primeira nem na segunda colunas. Com esta técnica, após a primeira atribuição, o domínio da segunda rainha é reduzido para a terceira e quarta colunas, conseguindo, portanto, uma redução tanto no espaço de pesquisa como no número de tentativas falhadas e, consequentemente, de *backtracks*.

6.2.1. Consistência

Neste contexto surge também o conceito de consistência local e todos os métodos e critérios que lhe estão associados de modo a que se possa manter a consistência de um grupo de variáveis e/ou restrições. O objectivo deste conceito passa pela detecção de valores redundantes nos domínios das variáveis, sendo que os processos responsáveis por garantir a consistência irão eliminar tais valores durante a fase de propagação, de modo a reduzir ainda mais o espaço de pesquisa. Ainda que existam diversos critérios de consistência, os mais conhecidos, baseados na estrutura do problema, são: Consistência de Nó, Consistência de Arco, Consistência de Caminho e Consistência-i (Dechter, 2003; Marriott & Stuckey, 1998; Rossi et al., 2006).

O algoritmo de consistência mais simples é a consistência de nó e pode dizer-se que um CSP é *node-consistent* se, e só se, todos os nós que compõem o grafo também o forem. Para tal é necessário que os valores no domínio de cada variável, representada através de um nó, satisfaçam as restrições unárias que lhe são aplicadas. Formalmente pode dizer-se que o nó *i*, representando a variável X_i, de domínio D_i, é consistente ao nível do nó se, e só se, $D_i \subseteq R_i$.

Ainda que possa parecer demasiado trivial e, eventualmente, desnecessário, este critério pode ser particularmente útil no contexto de execução no qual as soluções parciais vão sendo, incrementalmente, construídas. Em particular, no problema das N-Rainhas, com $N = 4$, temos o seguinte grafo:

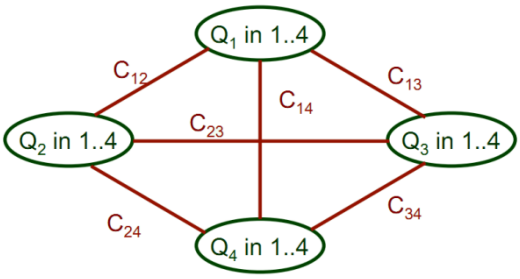

Figura 6.4 - Grafo de Restrições para o problema 4-Rainhas.

Caso a instanciação de variáveis comece pela variável Q_1 e lhe seja atribuído o valor 1, as restrições C_{12}, C_{13} e C_{14} passam a ser unárias, passando o grafo a ter o seguinte aspecto:

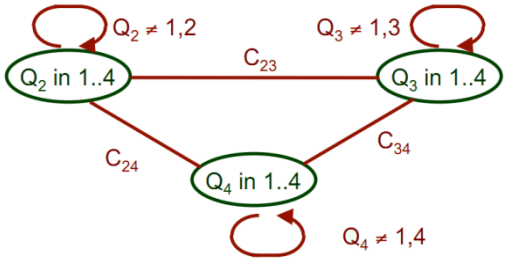

Figura 6.5 - Grafo de Restrições após instanciação da variável Q_1.

Neste momento o algoritmo responsável pela consistência deverá remover dos domínios das variáveis os valores obsoletos, de modo a que estes fiquem de acordo com as imposições da propagação.

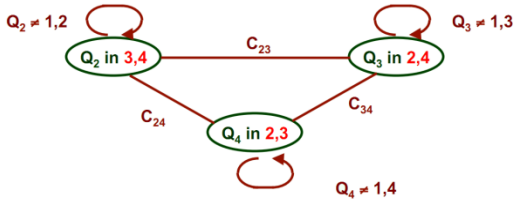

Figura 6.6 - Grafo de Restrições após acção do algoritmo de consistência.

Garantindo a Consistência de Nó é possível obter a seguinte redução de domínio, baseada apenas na instanciação da primeira rainha.

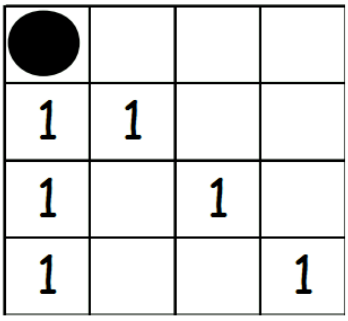

Figura 6.7 - Ilustração do conceito de Propagação de Restrições para o problema das 4-Rainhas com $Q_1 = 1$.

A Consistência de Arco é um critério mais complexo e que pretende restringir ainda mais os domínios usando restrições binárias. Para que um CSP seja consistente ao nível de arco é necessário, antes de mais, que o seja ao nível do nó e que, para todas as variáveis *X*, para todas as atribuições $\langle X,a \rangle$ que satisfaçam as restrições em *X*, exista um valor *b* para todas as variáveis *Y* tal que o par $(\langle X,a \rangle,\langle Y,b \rangle)$ satisfaça todas as restrições em *X* e *Y*. Por outras

palavras pode dizer-se que, dada a relação R_{ij} entre as variáveis X_i e X_j, o arco (i,j) é *arc-consistent* se, e só se, $D_i \subset \pi_i(R_{ij} \bowtie D_j)$. No fundo, o que ocorre é a eliminação de todos os elementos de D_i que não têm correspondência em D_j de modo a satisfazer R_{ij}.

À semelhança do que ocorreu aquando da afectação $Q_1 = 1$ ao problema das 4-Rainhas, o grafo com consistência ao nível dos nós e após a propagação representa-se segundo a Figura 6.7. No entanto, com alguma atenção, é possível reparar que se $Q_2 = 3$, não existe nenhum valor no domínio de Q_3, tal que a restrição C_{23} seja satisfeita.

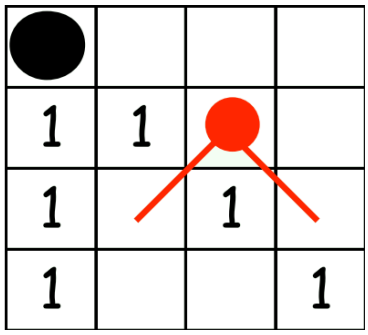

Figura 6.8 - Ilustração do conceito de *Arc Consistency* no problema 4-Rainhas, para a variável Q_2.

Deste modo o valor 3 pode, e deve, ser retirado do domínio da variável Q_2. Por outro lado, é de notar que nenhum dos valores presentes no domínio de Q_3 dá suporte às variáveis Q_2 e Q_4, isto porque se $Q_3 = 2$, não é possível satisfazer a restrição C_{34} ou, caso $Q_3 = 4$, não é possível satisfazer a restrição C_{23}.

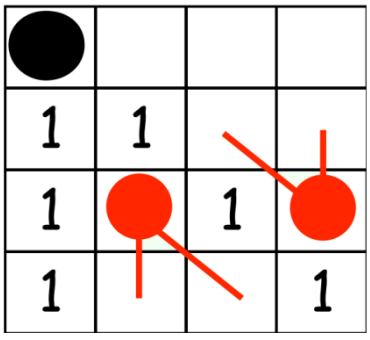

Figura 6.9 - Ilustração do conceito de Consistência de Arco no problema 4-Rainhas, para a variável Q_3.

Além de esvaziar o domínio de Q_3, visto não proporcionar suporte às variáveis Q_2 e Q_4, o critério Consistência de Arco acaba por ser também responsável pela detecção de insatisfação na variável Q_3, permitindo que seja realizado um *backtracking* de $Q_1 = 1$, mesmo antes da instanciação da variável Q_2.

Pode ainda definir-se o conceito de Consistência de Caminho, que assenta sobre a noção de caminho num grafo mas que, no entanto, acaba por não ser muito usado devido à sua complexidade computacional.

Para que um CSP seja *path-consistent* é necessário, primeiramente, que seja *arc-consistent* e se, para todas as variáveis *X* e *Y*, sempre que $(\langle X,a\rangle,\langle Y,b\rangle)$ satisfaça as restrições em ambas as variáveis, exista um par $\langle Z,c\rangle$ para todas as variáveis *Z* tal que $(\langle X,a\rangle,\langle Y,b\rangle,\langle Z,c\rangle)$ satisfaça todas as restrições em *X*, *Y* e *Z*. Segundo a nomenclatura usada neste documento para os Problemas de Satisfação de Restrições, diz-se que um caminho de comprimento dois do nó *i* para o nó *j*, passando pelo nó *m*, é consistente ao nível de caminho se, e só se, $R_{ij} \subset \pi_{ij}(R_{im} \bowtie D_m \bowtie R_{mj})$. Ou seja, para todos os pares de valores (a,b) permitidos pela relação R_{ij}, existe um valor *c*, para X_m, tal que (*a, c*) é permitido por R_{im} e (c,b) é permitido por R_{mj}.

Com este critério é possível evitar retrocesso no problema 4-Rainhas visto que mesmo antes de atribuir o valor 1 a Q_1, o algoritmo que garante a consistência terá que testar os diversos caminhos e chegará à conclusão que essa atribuição só tem correspondência de dois valores em Q_3, no entanto, um dos valores não permite extensão para Q_4 e outro também não o permite para Q_2.

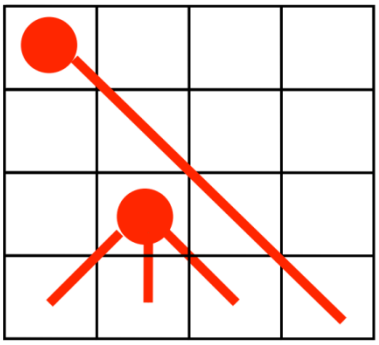

Figura 6.10 - Ilustração do conceito de Consistência de Caminho no problema 4-Rainhas, para as variáveis $Q_1 = 1$ e $Q_3 = 2$.

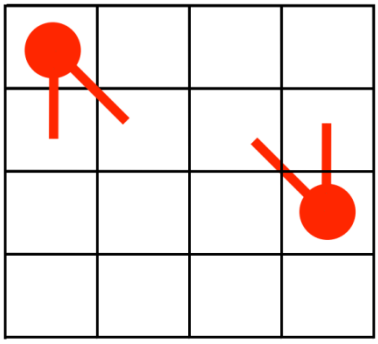

Figura 6.11 - Ilustração do conceito de Consistência de Caminho no problema 4-Rainhas, para as variáveis $Q_1 = 1$ e $Q_3 = 4$.

Assim, o valor 1 poderá ser retirado do domínio de Q_1 e, com um raciocínio idêntico é possível demonstrar que nenhum dos cantos, nem das posições centrais do tabuleiro pode conter rainhas.

Por fim, o critério Consistência-i acaba por ser uma generalização dos já apresentados Consistência de Nó, de Arco e de Caminho, para um subgrafo de cardinalidade 1, 2 e 3, respectivamente. Deste modo, um grafo diz-se *i-consistent* se, e só se, todas as atribuições $\left(\langle X_1,v_1\rangle,\langle X_2,v_2\rangle,...,\langle X_k,v_k\rangle\right)$ de cardinalidade $k = i - 1$, possam ser estendidas a uma i-ésima variável X_i. Adicionalmente pode dizer-se que um grafo com *n* vértices é *strongly i-consistent* se for *k-consistent* para todo o $k \leq i$, sendo que, se um grafo for *strongly n-consistent* é também globalmente consistente o que significa que se pode estender qualquer instanciação parcial sem chegar a um beco sem saída.

6.2.2. Restrições Globais

Muitas vezes estes critérios de consistência não são suficientes para garantir uma redução da árvore de pesquisa, já que as restrições formam um grafo consistente e, portanto, nenhum ramo será cortado. É neste contexto que as restrições globais se tornam relevantes já que tipicamente conseguem aplicar cortes à árvore de pesquisa que os algoritmos de consistência já referidos, por si só, não aplicariam. Uma restrição global pode definir-se como uma restrição que envolve um número não fixo de variáveis, cuja relação pode ser expressa através de um conjunto de restrições (Rossi et al., 2006). No entanto, e como já foi referido, a sua existência não se baseia apenas em facilitar o processo de modelação e programação, mas essencialmente facilitar o trabalho do resolvedor.

Um dos exemplos mais comuns é a restrição $\text{alldifferent}(X_1, X_2,..., X_n)$ que indica que os valores atribuídos às variáveis $X_1, X_2,..., X_n$ devem ser todos diferentes entre si, algo que podia

ser expresso através de um conjunto de C_2^n restrições de diferença, $X_i \neq X_j, \forall_{i,j} (1 \leq i < j \leq n)$. Relativamente a esta restrição, tome-se o seguinte exemplo para a instância de *N* = 9, com os domínios que se seguem, para cada uma das variáveis: $X_1 \in \{1,2,3\}$, $X_2 \in \{1,2,3,4,5,6\}$, $X_3 \in \{1,2,3,4,5,6,7,8,9\}$, $X_4 \in \{1,2,3,4,5,6\}$, $X_5 \in \{1,2,3\}$, $X_6 \in \{1,2,3,4,5,6,7,8,9\}$, $X_7 \in \{1,2,3,4,5,6,7,8,9\}$, $X_8 \in \{1,2,3\}$ e $X_9 \in \{1,2,3,4,5,6\}$. O grafo formado pelas $C_2^9 = 36$ restrições de desigualdade é *arc-consistent* pelo que o critério de consistência não irá realizar nenhuma redução dos domínios das variáveis e, consequentemente, do espaço de pesquisa. No entanto, com alguma atenção, é possível perceber que as variáveis X_1, X_5 e X_8 terão que tomar valores, entre si, do domínio $\{1,2,3\}$ e, dada a natureza da restrição que pretendemos modelar, é intuitivo que tais valores sejam retirados dos domínios das outras variáveis já que constituem valores obsoletos que nunca poderão ser usados numa solução. O mesmo ocorrerá com as variáveis X_2, X_4 e X_9 para o domínio $\{4,5,6\}$ e, posteriormente, com X_3, X_6 e X_7 para $\{7,8,9\}$. Neste caso, através do uso de restrições globais, é possível conseguir, à partida, uma redução considerável do domínio tornando a resolução do problema bastante mais eficiente.

A lista de restrições globais é relativamente extensa já que muitas das restrições que se verificam em diversos problemas acabam por ser incorporadas como tal na maioria dos resolvedores. Entre elas destacam-se a *alldifferent* apresentada acima, a *cumulative*, bastante usada em problemas de atribuição de tarefas, recursos e escalonamento, já que garante que, em cada ponto do tempo, o consumo de recursos não excede uma capacidade especificada. Outra restrição bastante usada e particularmente útil para os *Vehicle Routing Problems* é a *circuit* que associada a um grafo $G = (X, A)$, garante que o conjunto de arcos *A* forma um circuito. O problema do caixeiro-viajante, por exemplo, pode ser modelado com recurso a esta restrição global (Rossi et al., 2006). Admita-se que c_{ij} representa a distância entre a cidade *i* e a cidade *j*, com 1 ≤ *i*, *j* ≤ *n*, e, para cada cidade *i*, existe uma variável X_i com domínio $D(X_i) = \{1,...,n\} \setminus \{i\}$ que representa a cidade a visitar imediatamente a seguir. Por outro lado existe uma outra variável, $d_i, \forall i \in \{1,...,n\}$ que indica a distância entre a cidade *i* e a cidade X_i. Assim, o problema do caixeiro-viajante pode ser modelado da seguinte forma:

$$\begin{gathered} \min Z, \\ \text{circuit}(X_1,...,X_n), \\ Z = \sum_{i=1}^{n} d_i \\ d_i = c_{ix_i}, 1 \leq i \leq n \end{gathered}$$

6.3. Domínios

Apesar de ao longo deste capítulo terem sido apenas referidos domínios finitos, a Programação com Restrições permite considerar outros tipos de domínio. O facto de todos os conceitos e técnicas terem sido apresentados num contexto de domínios finitos, nos quais os inteiros são um exemplo típico, não significa que estes não sejam gerais e aplicáveis a diferentes domínios. O recurso a exemplos no domínio dos inteiros pretendeu apenas apresentar todos os conceitos de forma mais intuitiva.

No âmbito do *Dial-a-Ride Problem* pode ser interessante considerar outros domínios para além dos inteiros, visto que a formulação genérica da maioria dos problemas de definição de rotas se baseia em conceitos de teoria de grafos (Berbeglia et al., 2007). Neste contexto destaca-se um *framework* de nome GRASPER que pretende representar, através de restrições, os conceitos associados aos grafos de modo a permitir a modelação e resolução de problemas através deles (Viegas & Azevedo, 2007). Este *framework* baseia-se num outro, de nome *Cardinal*, que tem um objectivo idêntico mas, desta feita, para conjuntos (Azevedo, 2007).

De modo a ilustrar melhor os conceitos associados a estes *frameworks*, bem como a forma como se relacionam, nas próximas secções serão feitas breves descrições de ambos.

6.3.1. Conjuntos

As restrições ao nível dos conjuntos tiveram bastante atenção por parte da comunidade ligada à Programação com Restrições na última década do século XX dado seu potencial, sendo considerada uma área bastante promissora (Azevedo, 2003). Muitos tipos de relações complexas entre conjuntos podem ser expressos através de restrições, nomeadamente relações de inclusão, disjunção ou mesmo igualdade, recorrendo ainda a operadores como intersecção, união e diferença de conjuntos.

Com o objectivo de contemplar estas questões surge a biblioteca *Conjunto* (Gervet, 1994) que especifica os domínios dos conjuntos através de intervalos, com limite superior e inferior, tipicamente referidos como *glb* e *lub*, do inglês, *greatest lower bound* e *least upper bound*. No entanto, nem só este tipo de relações interessam, também alguns atributos e/ou funções dos conjuntos podem ter um papel importante na resolução de problemas, como por exemplo, a sua cardinalidade. Em muitos problemas o objectivo pode passar por maximizar/minimizar a

cardinalidade de um conjunto, restringi-la a um número par, ou mesmo fixá-la numa determinada ordem.

Assim, surge a biblioteca *Cardinal* com o objectivo de melhorar o desempenho do *Conjunto* contemplando a cardinalidade na fase de inferência, permitindo, entre outros, deduzir rapidamente se um conjunto de restrições não é satisfazível (Azevedo, 2007). Para exemplificar considere-se: *Z* é um conjunto que é igual à diferença entre *Y* e *X*, ambos contidos no conjunto $\{a,b,c,d\}$. Sabe-se também que *X* tem exactamente 2 elementos. Assim, deveria inferir-se que a cardinalidade de *Z* nunca poderá ser superior a 2 elementos. Formalmente, se $X,Y \subseteq \{a,b,c,d\}$, #*X* = 2, *Z* = *Y* \ *X*, deve inferir-se que #*Z* ≤ 2.

Ainda que não se pretenda apresentar de forma exaustiva o *Cardinal*, será importante explicar mais alguns conceitos que lhe estão associados, nomeadamente os já referidos intervalos que permitem a representação dos conjuntos. Estes intervalos definem um reticulado de conjuntos, do inglês *lattice* (Gratzer, 1971). Abaixo, na Figura 6.12, representa-se o reticulado para o exemplo em que o domínio do conjunto é $U = \{a,b,c,d\}$ e a linha que liga o conjunto C_1 a um conjunto abaixo C_2 traduz a relação $C_2 \subseteq C_1$.

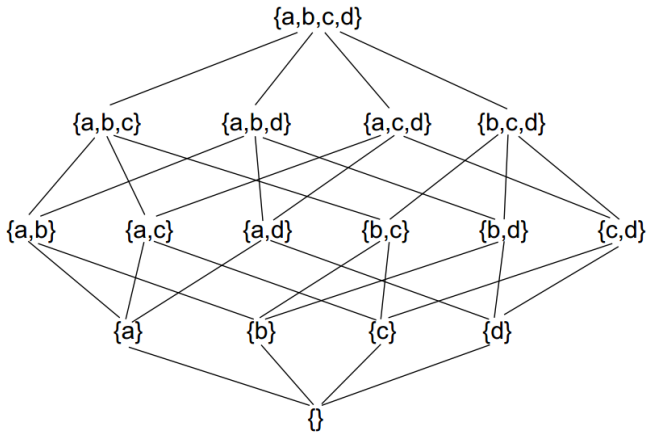

Figura 6.12 - Reticulado para $U = \{a,b,c,d\}$.

Note-se que o conjunto do topo inclui todos os potenciais subconjuntos de *U*, enquanto o conjunto do fundo está incluído em todos os outros. Deste modo são considerados o limite superior e inferior, *lub* e *glb*, respectivamente.

Por outro lado, caso se considere o sub-reticulado da Figura 6.13, ainda que os conjuntos $\{\ \}$ e $\{a,b,c,d\}$ continuem a ser limites inferior e superior, neste caso a *glb* é $\{b\}$ e a *lub* $\{a,b,d\}$.

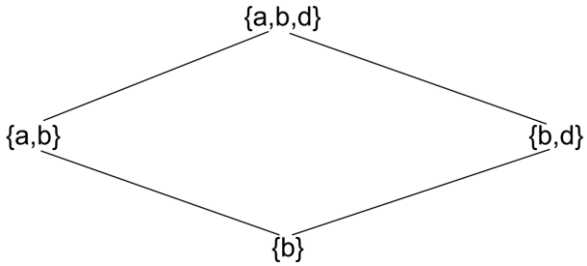

Figura 6.13 - Reticulado do intervalo $\left[\{b\},\{a,b,d\}\right]$.

Os dois limites definem o intervalo do conjunto, para este exemplo $\left[\{b\},\{a,b,d\}\right]$, e formam o domínio de uma variável conjunto *S* que tomará um dos valores de conjunto presentes no reticulado, sendo que todos os outros conjuntos fora deste domínio serão excluídos da solução. Assim, *b* estará garantidamente presente no conjunto *S*, enquanto *a* e *d* são os outros termos possíveis, como representado no diagrama de Venn abaixo.

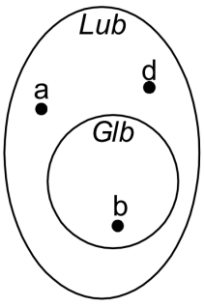

Figura 6.14 - Diagrama de Venn para o conjunto com domínio $\left[\{b\},\{a,b,d\}\right]$.

Deste modo, e a título de exemplo, qualquer variável de conjunto que possa tomar valores do seguinte conjunto $\{\{a,b,\},\{a,c\},\{d\}\}$ tem como representação $\left[\{\ \},\{a,b,c,d\}\right]$. Genericamente o intervalo $[glb,lub]$ pode ser expresso, para *n* possíveis conjuntos $C_1,...,C_n$, da seguinte forma:

$$glb = \bigcap_{i=1}^{n} C_i \wedge lub = \bigcup_{i=1}^{n} C_i$$

Estes conceitos serão importantes para se perceber a forma como esta biblioteca se integra com a que se apresenta de seguida e que servirá para uma das modelações do DARP apresentadas no Capítulo 7.

6.3.2. Grafos

Historicamente o artigo de Leonhard Euler sobre o problema das sete pontes de Königsberg é considerado o primeiro na história da teoria dos grafos, uma disciplina da matemática discreta que estuda estruturas que mapeiam as relações entre objectos de um determinado conjunto (Biggs et al., 1986). No contexto da ciência da computação, um grafo é tipicamente representado como um conjunto de vértices e um conjunto de arcos, fazendo estes últimos a ligação entre pares de elementos do conjunto de vértices. A flexibilidade deste tipo de estrutura é tal que a sua utilização tem vindo a ser utilizada para a modelação de diversos tipos de relações em diversas áreas (Tuttle & Nash-Williams, 2001).

O GRASPER, do inglês *GRAph constraint Satisfaction Problem solver*, surge como um *framework* que disponibiliza funcionalidades e restrições ligadas à teoria dos grafos, nomeadamente, grafos direccionados, pesados, optimização de problemas de caminhos, bem como as propriedades mais comuns que lhes estão associadas (Viegas & Azevedo, 2007). Foi criado com base na biblioteca *Cardinal*, apresentada acima, e permite a modelação de problemas e respectivas restrições como grafos.

De modo a apresentar uma pequena explicação de como ambas as bibliotecas se integram seguem-se alguns exemplos e formalizações de predicados ligados ao domínio dos grafos, no entanto aconselha-se a leitura do artigo de Viegas e Azevedo (2007) para mais pormenores, tal como o de Azevedo (2003) para o caso do *Cardinal*.

Formalmente, sendo o grafo $G = (V, A)$, com V como o conjunto de vértices e A o conjunto de arcos, defina-se o caso de um grafo direccionado através do seguinte predicado: $\text{dirgraph}(G, V, A)$.

A partir desta definição é possível apresentar outras restrições como por exemplo $\text{weight}(G, M, W)$, onde M é um mapa onde estão guardados os pesos associados a cada vértice e a cada arco do grafo G e W o peso do grafo. Este peso é calculado através da soma dos pesos associados a cada um dos vértices e dos arcos que compõem o grafo, pelo que a restrição pode ser definida da seguinte forma:

$$\text{weight}\left(\text{dirgraph}(G, V, A), M, W\right) \equiv \begin{cases} l = \sum_{v \in \text{glb}(V)} \text{M}(v) + \sum_{a \in \text{glb}(A)} \text{M}(a) \\ u = \sum_{v \in \text{lub}(V)} \text{M}(v) + \sum_{a \in \text{lub}(A)} \text{M}(a) \\ W \in [l, u] \end{cases} \quad (6.3.1)$$

O valor de *W* varia de acordo com os limites inferiores (*glb*) e superiores (*lub*) tanto do conjunto de vértices como do conjunto de arcos, sendo que este tomará um valor exacto quando estes limites forem iguais para ambos os conjuntos.

Outro tipo de restrição que se pode formalizar facilmente é a de subgrafo: um subgrafo de um grafo *G* é, por si, também um grafo cujos vértices e arcos estão contidos em *G*. Com as noções de conjunto, esta restrição pode expressar-se como se segue:

$$\text{subgraph}\big(\text{dirgraph}(G_1,V_1,A_1),\text{dirgraph}(G_2,V_2,A_2)\big) \equiv V_1 \subseteq V_2 \wedge A_1 \subseteq A_2 \qquad (6.3.2)$$

sendo G_1 um subgrafo do grafo G_2.

Ainda que o objectivo desta secção não seja fazer uma apresentação exaustiva do *framework* GRASPER, apresentar-se-ão ainda alguns dos predicados disponibilizados dada a sua utilização no capítulo seguinte no qual se descrevem algumas propostas para a modelação do *Dial-a-Ride Problem*.

Destes predicados destaca-se o $\text{predecessors}(G,v,P)$ que, dado um grafo *G*, devolve o conjunto dos antecessores do vértice *v*. Formalmente, este predicado pode ser expresso da seguinte forma:

$$\text{predecessors}\big(\text{dirgraph}(V,A),v,P\big) \equiv P \subseteq V \wedge \forall v' \in V : \big(v' \in P \equiv (v',v) \in A\big) \qquad (6.3.3)$$

Analogamente, para os sucessores de um vértice existe o predicado *successors*:

$$\text{successors}\big(\text{dirgraph}(V,A),v,S\big) \equiv S \subseteq V \wedge \forall v' \in V : \big(v' \in S \equiv (v,v') \in A\big) \qquad (6.3.4)$$

Conjuntamente com o predicado *successors* defina-se um outro predicado que permite obter não apenas os sucessores directos de um vértice, mas sim todos os vértices alcançáveis a partir de um dado vértice. Este predicado dá pelo nome de $\text{reachables}(G,v,R)$ e, recorrendo à função *paths* que devolve todos os caminhos possíveis entre dois vértices, pode formalizar-se da seguinte forma:

$$\begin{gathered}\text{reachables}\big(\text{dirgraph}(V,A),v,R\big) \\ \equiv \\ R \subseteq V \wedge \forall r \in V : \big(r \in R \equiv \exists p : p \in \text{paths}\big(\text{dirgraph}(V,A),v,r\big)\big)\end{gathered} \qquad (6.3.5)$$

A partir deste último predicado é possível definir a restrição $\text{connected}(G)$ que garante que cada vértice do grafo *G* é acessível por qualquer outro vértice grado, assumindo um grafo não direccionado. Esta restrição pode formalizar-se da seguinte forma:

$$\text{connected}\big(\text{graph}(V,A)\big) \equiv \forall v \in V : \text{reachables}\big(\text{graph}(V,A),v,R\big) \wedge R = V \qquad (6.3.6)$$

Recorrendo aos predicados *predecessors* e *successors* já referidos e que, por questões de apresentação serão mencionados como *preds* e *succs*, bem como à formulação de grafo direccionado, $\text{dirgraph}(G,V,A)$ (que será referido como $\text{graph}(V,A)$) é possível formalizar o predicado *quasipath* da seguinte forma:

$$quasipath\left(graph\left(V,A\right),v_0,v_f\right) \equiv \forall v \in V \begin{cases} succs\left(graph\left(V,A\right),S\right) \wedge \#S = 1, \text{ se } v = v_0 \\ preds\left(graph\left(V,A\right),P\right) \wedge \#P = 1, \text{ se } v = v_f \\ preds\left(graph\left(V,A\right),P\right) \wedge \#P = 1 \\ \wedge \\ succs\left(graph\left(V,A\right),S\right) \wedge \#S = 1, \text{ c.c.} \end{cases} \quad (6.3.7)$$

Ainda que aparente ser uma restrição complexa, trata-se de algo bastante intuitivo: garante que cada vértice que é adicionado ao grafo tem exactamente um antecessor e um sucessor, com excepção para o vértice inicial v_0 e o final, v_f que têm apenas um sucessor ou um antecessor, respectivamente, definindo um caminho entre os vértices inicial e final.

Por fim, os dois últimos predicados apresentados formam um outro que restringe um grafo *G* a representar um caminho entre dois vértices, $\text{path}(G,v_0,v_f)$. Para tal é necessário que exista um caminho entre o vértice inicial v_0 e o final v_f e todos os outros vértices que compõem o grafo pertençam ao caminho e sejam visitados apenas uma vez, algo garantido pelo predicado *quasipath*. Por outro lado, o predicado *connected* fica com a responsabilidade de garantir que os vértices que compõem o grafo pertencem ao caminho estabelecido e de que o grafo em causa é conexo. Deste modo, a formalização pode ser dada por:

$$\text{path}(G,v_0,v_f) \equiv \text{quasipath}(G,v_0,v_f) \wedge \text{connected}(G) \quad (6.3.8)$$

7

7. Modelação

No presente capítulo são apresentadas algumas modelações possíveis para a resolução do *Dial-a-Ride Problem*, recorrendo à Programação com Restrições. Ainda que se tenha como caso de estudo a delegação Amadora – Sintra da Cruz Vermelha Portuguesa, pretende-se desenvolver um modelo genérico que possa não só dar resposta à realidade desta delegação, mas também a outras instâncias do problema, quer sejam estas da Cruz Vermelha Portuguesa ou de outra instituição que realize transporte de doentes.

Deste modo o principal objectivo passa pela concepção de um modelo para uma versão estática do DARP, com uma frota heterogénea, sendo que a numa primeira fase a heterogeneidade passará apenas pela existência de veículos com diferentes lotações e, *a posteriori*, pretender-se-á considerar necessidades específicas de transporte, como por exemplo, cadeira de rodas ou macas.

Como será dado a entender ao longo deste capítulo e consumado na Secção 8.2, o facto de serem apresentadas diferentes modelações diz respeito a um processo iterativo de modelação pois nem todos os modelos tiveram o sucesso desejado.

Na secção seguinte serão apresentados alguns dados transversais a qualquer modelação que dizem respeito ao *input* das instâncias e que se baseiam na formulação já apresentada neste documento (ver Secção 3.1).

7.1. Dados Comuns

No decorrer do Capítulo 3 foi apresentada uma formulação para uma versão básica do DARP que contemplava uma frota homogénea, um único depósito e onde a qualidade de serviço era

contemplada de forma minimalista., com o objectivo de contextualizar melhor e problema e de conferir-lhe uma apresentação formal.

Para uma versão do DARP como a que se pretende modelar neste capítulo, com frota heterogénea e possibilidade de contemplar mais que um depósito, existem algumas adaptações que têm que ser feitas à formulação já apresentada e que serão expostas ao longo das próximas secções.

7.1.1. Veículos

De forma idêntica ao que foi apresentado anteriormente, o DARP é genericamente caracterizado por uma frota composta por *m* veículos, $Veículos = \{V_1,...,V_m\}$, que, dada a heterogeneidade pretendida, têm capacidades de transporte diferenciadas, pelo que a este conjunto se associa um outro que representa a capacidade de transporte do veículo *k*, com $1 \leq k \leq m$, $Capacidades = \{C_1,C_2,...,C_m\}$.

7.1.2. Depósitos

Ainda relacionado com o conjunto *Veículos* estão os depósitos que desta feita estão associados a cada veículo, podendo ser fisicamente diferentes, por oposição ao formulado anteriormente em que existia apenas um depósito comum a toda a frota.

Este conjunto pode ser expresso como $Depósitos = \{D_1,...,D_{2m}\}$, com uma cardinalidade de 2*m*, sendo *m* o número de veículos que compõem a frota, existindo um depósito inicial e um final associados a cada viatura. Ainda que em situações em que o depósito é o mesmo para todos os veículos esta abordagem possa parecer excessiva, a verdade é que permite contemplar diferentes depósitos, o que pode ser interessante para algumas instâncias, e também contemplar os tempos de serviço de uma forma mais coerente. De notar que este conjunto representa virtualmente os depósitos, o conjunto real será composto apenas pelos diferentes depósitos físicos que, a título de exemplo, no caso de estudo contemplado são dois (ver Secção 2.4).

Neste contexto, o depósito D_k, com $k \in \{1,...,m\}$, corresponde ao local de onde parte o veículo *k*, sendo o depósito onde este deverá regressar e ficar estacionado para o dia seguinte dado por D_{k+m}. Fisicamente estes depósitos podem, ou não, corresponder à mesma localização, consoante as necessidades do serviço.

7.1.3. Pedidos de Transporte

Um outro conjunto de dados que caracteriza o *Dial-a-Ride Problem* é o conjunto que representa os diversos pedidos de transporte a que se pretende dar resposta. Com cardinalidade *n*, o conjunto *Pedidos* também será ligeiramente modificado relativamente ao que foi apresentado no Capítulo 3.

Por uma questão de generalidade este conjunto não será separado pelo tipo de pedido de transporte, ou seja, transporte para as entidades que prestam serviços na área da saúde ou de regresso à residência, e os seus elementos são do tipo

$$\begin{aligned}(&\mathit{Origem},\mathit{Destino},\mathit{Janela\,Temporal\,Origem},\mathit{Janela\,Temporal\,Destino},\\&\mathit{Tempo\,de\,Serviço\,Origem},\mathit{Tempo\,de\,Serviço\,Destino},\mathit{Carga})\end{aligned} \tag{7.1.1}$$

com *Origem* ≠ *Destino* e as janelas temporais sendo do tipo $[L,U]$, com $L \leq U$. Estas janelas temporais representam o intervalo de tempo no qual o veículo que dá resposta ao pedido em causa pode chegar ao local, independentemente de se tratar de uma entidade de saúde ou do local de residência. No primeiro caso estas janelas tendem a ser mais apertadas.

Por outro lado, os tempos de serviço indicam o tempo esperado que demorará a operação de entrada e/ou saída do utente no veículo. Estes tempos podem ser significativos dado que alguns dos pacientes têm mobilidade reduzida. Os seus valores podem ser calculados de forma empírica, tendo em conta experiências passadas, ou podem ser baseados num tempo médio, sem indicações particulares.

Por fim, a *Carga* representa o número de passageiros a transportar dado que por vezes o requerente tem necessidade de se fazer acompanhar de outrem, por norma um familiar ou *caregiver.*

Com base nestes dados é possível representar o conjunto dos vários locais a visitar: $\mathit{Locais} = \mathit{Origens} \cup \mathit{Destinos}$. Este conjunto contempla tanto as moradas das residências dos utentes, como as das entidades de saúde, podendo conter moradas iguais caso exista a necessidade do local ser visitado mais do que uma vez. Assim, dados *n* pedidos de transporte, a cardinalidade de *Locais* será 2*n*, podendo alguns representar o mesmo local físico. A partir destes dados de entrada é possível formar o conjunto de vértices do problema,

$$\mathit{Vértices} = \mathit{Depósitos} \cup \mathit{Locais},\ \text{com}\ |\mathit{Vértices}| = 2m + 2n\ ,$$

já que como se referiu anteriormente, |*Depósitos*| = 2*m* devido à opção de se modelar, para cada veículo, o depósito de onde este parte e onde chega.

À semelhança da codificação utilizada na secção acima referente aos depósitos, os diversos locais associados aos pedidos de transporte também serão representados por inteiros, por uma questão de simplicidade e coerência. Deste modo, dados *n* pedidos de transporte e uma frota composta por *m* veículos, os locais de origem dos pedidos são representados por $\{2m+1,...,2m+n+1\}$ e os seus destinos por $\{2m+n+2,...,2m+2n\}$, o que possibilita estabelecer que o destino do local *i*, com $i \in \{2m+1,...,2m+n+1\}$ é o local dado por $i + n$.

7.1.4. Tempos do Serviço

Ainda no contexto dos dados de entrada do problema, destacam-se diversos parâmetros relacionados com os *timings* do serviço, nomeadamente: *Início do Serviço, Término do Serviço, Tempo Máximo de Viagem, Tempo Máximo de Serviço*.

Os dois primeiros dizem respeito ao período de operação da entidade em causa que, no caso da Cruz Vermelha Portuguesa, corresponderão às 7:00 e 19:00, hora a partir da qual se inicia o serviço de transporte e hora até à qual este é realizado, respectivamente. O *Tempo Máximo de Viagem*, como o próprio nome indica, é o um limite superior para o tempo de viagem de um utente, isto é, o tempo que decorre desde que encontra no veículo até ao momento em que chega ao destino pretendido. Por fim, o *Tempo Máximo de Serviço* estabelece um limite para a operação dos veículos efectuarem a rota que lhes foi atribuída, sendo que na ausência de indicação explícita considera-se que este corresponde a todo o horizonte temporal, isto é, *Término do Serviço – Início do Serviço*. Por outro lado na ausência de indicações relativamente ao *Tempo Máximo de Viagem*, considera-se como unidade o tempo directo de viagem entre os dois locais (*Origem* e *Destino*) e limita-se a duração da viagem a um máximo dado por *F* e que será igual a $d + \max(d \times X, T)$, com *d* sendo a duração da viagem entre *Origem* e *Destino*, $X = 0{,}4$ e $T = 25$ minutos. O facto de não se considerar um valor constante, mas sim geométrico baseia-se na ideia de que não seria sensato um trajecto que, a título de exemplo, demora 5 minutos ter que ser realizado num máximo de 15 minutos e, por outro lado, um trajecto de 2 horas, poder ser realizado num período até 6 horas. Com este tipo de abordagem, o trajecto de 5 minutos poderá demorar até 30 minutos e o de 120, até 148.

Em complemento a estes dados existe uma matriz que mapeia o tempo de viagem entre todos os pares de vértices, de acordo com a sua distância. A título de exemplo, para o caso do conjunto *Vértices* referido acima, com cardinalidade $2m + 2n$, esta matriz terá o seguinte aspecto (Tabela 7.1):

Tabela 7.1 - Exemplo de Matriz Temporal.

	Vértice 1	**Vértice 2**	**…**	**Vértice *2m* + 2*n***
Vértice 1	0	4	…	7
Vértice 2	3	0	…	9
…	…	…	…	…
Vértice 2*m* + 2*n*	5	11	…	0

Assumindo que a unidade temporal em causa é o minuto, da tabela acima é possível perceber, por exemplo, que uma viagem entre o *Local 1* e o *Local 2* demora 4 minutos e que o caminho inverso, de *Local 2* para *Local 1*, demora menos um minuto. O facto de a distância não ser simétrica não é de todo estranho pois no contexto do *Dial-a-Ride Problem* estas distâncias são muitas vezes calculadas com recurso a sistemas de georreferenciação, de acordo com as vias disponíveis. O caso da delegação Amadora – Sintra da Cruz Vermelha Portuguesa não é excepção e as suas instâncias serão apresentadas no próximo capítulo e poderão ser consultadas no Anexo A.

7.2. Modelo de Grafos

Por norma os *Vehicle Routing Problems* são formulados recorrendo a grafos devido à propensão que estes têm para representar os conceitos envolvidos nestes problemas: rotas, caminhos, localizações, etc. (Laporte & Osman, 1995). Ainda assim estes são tipicamente implementados com recurso a inteiros devido à hegemonia das soluções baseadas em Programação Inteira.

Esta primeira modelação pretende ser uma novidade no contexto do *Dial-a-Ride Problem* dada a utilização um domínio ligeiramente diferente do usual, recorrendo directamente a grafos, através do *framework* GRASPER apresentado na Secção 6.3.2. Com esta característica espera-se também que a representação do problema possa ser feita de forma mais directa e intuitiva, indo de encontro à formulação genérica do DARP, onde este é mapeado por um grafo completo e direccionado onde os vértices representam os diversos locais a visitar e os arcos os caminhos entre eles.

7.2.1. Variáveis

Existem m variáveis neste modelo que constituem grafos direccionados, $G_k = (V_k, A_k)$, com $1 \leq k \leq m$, não necessariamente completos, ao contrário do que é tipicamente apresentado na

literatura. O conjunto de vértices do grafo G_k é dado por V_k que tem domínio em $Vértices = \{1,...,2m+2n\}$, com *m* o número de veículos e *n* o número de pedidos de transporte.

Na literatura o domínio do conjunto de arcos é tipicamente representado por $\{(i,j), i,j \in Locais \wedge i \neq j\}$, no entanto neste modelo será reduzido tendo em conta que existem alguns arcos neste domínio que não têm interesse para a resolução do problema. Entre estes destacam-se arcos cuja origem é um depósito final, cujo destino é um depósito inicial, ou mesmo arcos entre depósitos que não pertencerão à mesma rota e, consequentemente, ao mesmo veículo. Deste modo o domínio do conjunto de arcos pode ser redefinido da seguinte forma, de modo a não contemplar estes arcos improdutivos para o problema e referidos acima: $\{(i,j), i \in \{1,...,m\} \cup Locais, j \in \{m+1,...,2m\} \cup Locais \wedge i \neq j\}$, representando $\{1,...,m\}$ os depósitos iniciais e $\{m+1,...,2m\}$ os depósitos finais. Ao domínio anteriormente apresentado podem ainda ser removidos, para cada pedido, os arcos do local de destino do pedido para a sua origem, dado que representam uma viagem que nunca será realizada no contexto do DARP. Este conjunto de arcos desnecessários é dado por $N = \{(i,j), i,j \in Locais \wedge i = j+n\}$, com *n* o número de pedidos de transporte. Na Secção 7.5, referente à redução do espaço de pesquisa, esta questão dos arcos improdutivos para resolução do problema será retomada.

7.2.2. Restrições

Por uma questão de comodidade os *m* grafos apresentados acima serão a partir de agora expressos através da lista $Rotas = [R_1,...,R_m]$, sendo R_k, com 1 ≤ *k* ≤ *m*, o grafo correspondente à rota do veículo *k*. Da forma idêntica, e mesmo que o GRASPER disponibilize diversos predicados para expressar condições necessárias com uma sintaxe bastante própria, tomar-se-á a liberdade de alterá-la ligeiramente para a modelação das restrições abaixo. Entre as adaptações destacam-se, a título de exemplo, os predicados

predecessors$(Graph, Vertex, Predecessors)$ e successors$(Graph, Vertex, Successors)$,

formalizados em (6.3.3) e (6.3.4) ,que dado um grafo *Graph*, devolvem, respectivamente, os predecessores e sucessores de do vértice *Vertex*. Por uma questão de simplicidade passarão a ser referidos, para um grafo *G*, como antecessores(i,G) e sucessores(i,G), com $i \in Vértices$.

Com base nesta nomenclatura podem definir-se as seguintes restrições:

i. Os veículos que compõem a frota devem servir todos os pedidos de transporte, ou seja, devem passar por todos os vértices que formam o grafo, sem repetições. Assim, assumindo V_k como o conjunto de vértices do grafo R_k e A_k como o seu conjunto de arcos, pode dizer-se que:

$$\begin{gathered}\bigcup_{k=1}^{m} V_k = \textit{Vértices} \wedge \bigcap_{k=1}^{m} V_k = \varnothing \\ \wedge \\ \bigcup_{k=1}^{m} A_k \subset A \wedge \bigcap_{k=1}^{m} A_k = \varnothing\end{gathered} \tag{7.2.1}$$

e

$$\forall R_k = (V_k, A_k), R_l = (V_l, A_l) \in \textit{Rotas} \wedge k < l \begin{pmatrix} V_k \cap V_l = \varnothing \\ \wedge \\ A_k \cap A_l = \varnothing \end{pmatrix} \tag{7.2.2}$$

ii. A cardinalidade dos conjuntos de vértices e arcos de cada rota são estimáveis pois a natureza do modelo garante que o número de vértices de um grafo será sempre par, e superior ou igual a 2. Como já tinha sido apresentado aquando da formulação do DARP no Capítulo 3, o facto de cada pedido ser composto por dois vértices obriga a que o número de vértices servidos, independentemente do número de pedidos a que se dá resposta, seja par pois não é possível servir um pedido sem visitar os dois vértices que o compõem (origem e destino). Caso a cardinalidade do conjunto seja igual a 2 significa que a rota é vazia e que apenas os depósitos inicial e final são visitados. Por outro lado, e tendo em conta que uma rota representa um caminho no grafo, o número de arcos será inferior, em uma unidade, ao número de vértices. Formalmente,

$$\forall 1 \leq k \leq m, R_k = (V_k, A_k) \begin{cases} \#V_k \,\%\, 2 = 0 \wedge \#V_k \geq 2 \\ \wedge \\ \#A_k = \#V_k - 1 \end{cases} \tag{7.2.3}$$

iii. Tendo em conta que cada grafo $R_k = (V_k, A_k)$, com 1 ≤ *k* ≤ *m*, representa a rota do veículo *k* e que cada veículo tem associados um depósito inicial e um final, pode dizer-se que

$$\forall k \in \textit{Depósitos} \wedge k \leq m, \{k, k+m\} \subset V_k \tag{7.2.4}$$

iv. Ainda acerca dos depósitos, é esperado que os depósitos iniciais não tenham antecessores, nem que os depósitos finais tenham sucessores. Dada a redução de arcos referida anteriormente, esta restrição acaba por estar declarada implicitamente. No entanto, assumindo *m* como o número de veículos, pode explicitar-se da seguinte forma:

$$\begin{gathered}\forall R_k \in Rotas, \text{antecessores}\left(k, R_k\right) = \varnothing \\ \wedge \\ \forall R_k \in Rotas, \text{sucessores}\left(k+m, R_k\right) = \varnothing\end{gathered} \tag{7.2.5}$$

v. De forma idêntica é possível definir o número de antecessores e sucessores para todos os restantes vértices, isto é, vértices pertencentes ao conjunto *Locais*. Dada a natureza do problema e das soluções que se pretendem gerar, cada vértice do conjunto acima terá apenas duas ligações, um antecessor e um sucessor.

$$\forall R_k = \left(V_k, A_k\right) \in Rotas \left(\forall j \in V_k \cap Locais \left(\begin{array}{c} \#\text{antecessores}\left(j, R_k\right) = 1 \\ \wedge \\ \#\text{sucessores}\left(j, R_k\right) = 1 \end{array} \right) \right) \tag{7.2.6}$$

vi. Por outro lado, cada rota define um caminho entre o seu depósito inicial e final. Com recurso ao predicado $\text{caminho}\left(G, v_i, v_f\right)$ (ver definição em (6.3.8)) que define que o grafo *G* representa um caminho entre o vértice v_i e o vértice v_j, é possível formalizar esta restrição da seguinte forma:

$$\forall R_k \in Rotas, \text{caminho}\left(R_k, k, k+m\right) \tag{7.2.7}$$

vii. Algo que também poderá ser garantido através da redução do domínio dos arcos dos grafos são os sucessores e antecessores dos depósitos. Os depósitos iniciais não têm como sucessores vértices de *delivery*, isto é, vértices onde se deixam clientes, bem como os antecessores dos depósitos finais não podem ser vértices onde se vão buscar clientes.

$$\begin{gathered}\forall R_k \in Rotas, \text{sucessores}\left(k, R_k\right) \cap Destinos = \varnothing \\ \wedge \\ \forall R_k \in Rotas, \text{antecessores}\left(k+m, R_k\right) \cap Origens = \varnothing\end{gathered} \tag{7.2.8}$$

viii. A união de todas as rotas gera um grafo que representa todo o problema, contemplando todos os seus vértices e arcos. Considerando que cada rota é dada por $R_k = \left(V_k, A_k\right)$, com $1 \leq k \leq m$, o grafo geral pode ser expresso como:

$$G = \left(V, A\right) \begin{cases} V = \bigcup_{k=1}^{m} V_k \\ A = \bigcup_{k=1}^{m} A_k \end{cases} \tag{7.2.9}$$

ix. Há que garantir que de alguma forma as diversas rotas que o compõem se unem a um dado ponto, tornando o grafo *G* num grafo conexo. Para tal podem unir-se todos os depósitos finais ao primeiro do seu conjunto, ou seja

$$\forall k \in \textit{Depósitos} \wedge k > m\left(k \in \text{sucessores}\left(m+1,G\right)\right) \tag{7.2.10}$$

x. A este grafo *G* que contém todos os vértices e arcos das *m* rotas que compõem a solução do problema, também serão aplicadas restrições no que diz respeito aos pedidos de transporte. Para se obter soluções é necessário que todos os pedidos sejam servidos pelo que deve existir um caminho entre o local de origem e o local de destino. De modo a garantir tal condição pode optar-se por uma abordagem de subgrafos recorrendo ao predicado $\text{subgrafo}\left(G,S\right)$, formalizado em (6.3.2) e que restringe *S* a ser um subgrafo do grafo *G*, é possível definir tal restrição como:

$$\forall p \in \textit{Pedidos}\left(\exists \text{subgrafo}\left(G,S\right) : \text{caminho}\left(S,\text{Origem}\left(p\right),\text{Destino}\left(p\right)\right)\right) \tag{7.2.11}$$

vii. Para cada rota há que garantir que a capacidade do veículo não é excedida em nenhum momento. Para tal, considere-se que cada rota corresponde a um grafo pesado pelo que, com recurso ao predicado $\text{weight}\left(G,M\right)$, formalizado em (6.3.1), que calcula o peso do grafo *G* um grafo mediante um mapa *M* que contém os pesos associados aos seus vértices e/ou arcos, é possível calcular o peso deste. Assim e assumindo um mapa *M* em que os pesos estão apenas atribuídos aos vértices e reflectem o número de passageiros que entram ou saem nesse local, é possível definir esta restrição da seguinte forma:

$$\forall R_k \in \textit{Rotas}\left(\forall \text{subgrafo}\left(R_k,S\right)\left(0 \le \text{weight}(S,M) \le C_k\right)\right) \tag{7.2.12}$$

O facto de não se considerar apenas o grafo completo referente a uma rota, mas também os seus subgrafos, deve-se ao facto ter que se garantir que tal condição se verifica ao longo do caminho e não apenas no final.

viii. A precedência de vértices deve também ser implementada segundo os tempos de serviço dos diversos vértices. Para tal recorrer-se-á ao predicado reachables(*G*,*V*) que devolve o conjunto de vértices alcançáveis pelo vértice *V* no grafo *G* (formalizado em (6.3.5)). Assumindo o grafo *G* como o grafo que representa a união de todas as rotas (ver (7.2.9)), $\left[L_i,U_i\right]$ como a janela temporal associada ao vértice *i*, d_i como a duração do serviço nesse mesmo vértice e $T\left[i,j\right]$ como o

tempo de viagem entre os vértices *i* e *j*, é possível restringir as precedências da seguinte forma:

$$\forall i, j \in Locais\left(L_i + d_i + T\left[i, j\right] > U_j \Rightarrow j \notin \text{reachables}\left(i, G\right)\right) \quad (7.2.13)$$

7.3. Simplificação do Modelo de Grafos

Após os resultados obtidos pelo modelo apresentado anteriormente (ver Secção 8.2.1), optou-se por conceber um outro que, ainda que recorra à mesma biblioteca e aos conceitos associados à teoria dos grafos, consiste numa relaxação/simplificação do problema. Entre as simplificações, destaca-se a redução do número de nós, assumindo apenas um depósito inicial e um final para toda a frota, bem como a relaxação de algumas restrições e redução do número de variáveis.

O objectivo desta modelação passa por averiguar o quão adequada poderá ser a biblioteca de restrições GRASPER utilizada no âmbito do *Dial-a-Ride Problem*.

7.3.1. Variáveis

Ao nível das variáveis, apenas se contemplará uma variável principal que mapeia todo o problema e que consiste num grafo direccionado, assimétrico e também não necessariamente completo. Posteriormente serão criadas mais *m* variáveis correspondentes às rotas dos *m* veículos que compõem a frota e que estarão na lista *Rotas*.

O conjunto de vértices do grafo principal tem domínio em *Vértices* e, de forma semelhante ao que já foi apresentado, o conjunto de arcos deste grafo, dado por $G = \left(V, A\right)$, tem domínio em $\left\{\left(i, j\right), i \in \left\{1,...,m\right\} \cup Locais, j \in \left\{m,+1...,2m\right\} \cup Locais \wedge i \neq j \wedge i \neq j + n\right\}$. Este domínio já se encontra reduzido relativamente à questão dos arcos improdutivos para o problema, levantada na Secção 7.2 e explorada com mais detalhe na 7.5. Ainda assim, e como será abordado na secção referente à redução do espaço de pesquisa, este domínio ainda poderá ser reduzido tendo em conta alguns caminhos não exequíveis que possam existir.

7.3.2. Restrições

O grafo *G* representa as rotas de todos os veículos contemplados, no entanto, nesta modelação, estas só serão explicitadas mais tarde, como parte integrante de *G*, e modeladas como

restrições. Assim, com base neste grafo e nos dados gerais do *Dial-a-Ride Problem* podem definir-se as seguintes restrições:

i. Tendo em conta que se pretende dar resposta a todos os pedidos, o grafo *G* será composto por todos os vértices que definem o problema, ou seja, o seu conjunto de vértices deverá ser igual ao conjunto *Vértices* acima definido. Por outro lado, dada a natureza do problema, é possível relacionar a cardinalidade do seu conjunto de arcos com o de vértices. Formalmente o grafo pode expressar-se da seguinte forma:

$$G = (V, A) \begin{cases} V = Vértices \\ \#A = \#Locais + m \end{cases} \tag{7.3.1}$$

ii. Esta relação acerca da cardinalidade do conjunto de arcos acaba por também se ligar ao número de antecessores e sucessores de cada vértice do grafo, restringido de acordo com o contexto do problema e das suas soluções. Ainda que o conjunto de depósitos iniciais seja constituído apenas por um elemento, este será sempre referido no contexto do conjunto a que pertence. Assim, tem-se que:

$$\forall k \in Depósitos \wedge k \leq m \left(\begin{array}{c} \#\text{antecessores}(k, G) = 0 \\ \wedge \\ \#\text{sucessores}(k, G) = m \end{array} \right) \tag{7.3.2}$$

Analogamente, para os depósitos finais, tem-se:

$$\forall k \in Depósitos \wedge k > m \left(\begin{array}{c} \#\text{antecessores}(k, G) = m \\ \wedge \\ \#\text{sucessores}(k, G) = 0 \end{array} \right) \tag{7.3.3}$$

Para todos os restantes vértices, isto é, vértices que pertencem ao conjunto *Locais*, o conceito de rota ou caminho restringe a cardinalidade de ambos os conjuntos ao valor 1, já que apenas terão um antecessor e um sucessor. Formalmente,

$$\forall i \in Locais \left(\begin{array}{c} \#\text{antecessores}(i, G) = 1 \\ \wedge \\ \#\text{sucessores}(i, G) = 1 \end{array} \right) \tag{7.3.4}$$

iii. Sendo o principal objectivo do DARP dar resposta a pedidos de transporte, é necessário garantir que o grafo que representa o problema contempla esta

vertente. Assim, recorrendo ao predicado *reachables* já referido no decorrer deste documento pode formalizar-se esta restrição do seguinte modo:

$$\forall p \in \textit{Pedidos}\left(\text{Destino}(p) \in \text{reachables}\left(\text{Origem}(p), G\right)\right) \tag{7.3.5}$$

Devido à precedência imposta pelos pedidos, poderá também dizer-se que:

$$\forall p \in \textit{Pedidos}\left(\text{Origem}(p) \notin \text{reachables}\left(\text{Destino}(p), G\right)\right) \tag{7.3.6}$$

iv. As rotas relativas aos diversos veículos podem ser expressas como parte integrante do grafo *G*, em particular, na forma de um subgrafo que representa um caminho entre o depósito inicial e o depósito final. Os *m* subgrafos obtidos são armazenados na lista *Rotas* que tomará a forma $\left[R_1,...,R_m\right]$. Formalmente a restrição descrita pode ser expressa da seguinte forma:

$$\forall 1 \le k \le m\left(\exists \text{subgrafo}\left(G, R_k\right) : \text{caminho}\left(R_k, k, m+k\right)\right) \tag{7.3.7}$$

Implicitamente a restrição *caminho* impõe que ambos os locais estejam contidos no conjunto de vértices do grafo, algo que pode ser formalizado por:

$$\forall R_k = \left(V_k, A_k\right) \in \textit{Rotas}, \left(\textit{Depósitos} \subset V_k\right) \tag{7.3.8}$$

v. De forma semelhante ao que acontece no modelo anterior, estes subgrafos que representam as rotas dos vários veículos têm conjunto de vértices e arcos com cardinalidade estimável. Considerando a lista $\textit{Rotas} = \left[R_1,...,R_k\right]$, a restrição (7.2.3) pode ser reescrita da seguinte forma:

$$\forall R_k = \left(V_k, A_k\right) \in \textit{Rotas} \begin{cases} \#V_k \,\%\, 2 = 0 \wedge \#V_k \ge 2 \\ \wedge \\ \#A_k = \#V_k - 1 \end{cases} \tag{7.3.9}$$

vi. Em complemento a esta restrição acima há que garantir que as rotas não repetem vértices, à excepção dos depósitos de modo a garantir a consistência das soluções. Tal condição pode ser modelada da seguinte forma:

$$\begin{gathered} \bigcup_{k=1}^{m} V_k = \textit{Vértices} \wedge \bigcap_{k=1}^{m} V_k = \textit{Depósitos} \\ \wedge \\ \forall R_k = \left(V_k, A_k\right), R_l = \left(V_l, A_l\right) \in \textit{Rotas} \wedge k < l \left(V_k \cap V_l = \textit{Depósitos}\right) \end{gathered} \tag{7.3.10}$$

Ainda que não seja necessário explicitar-se, os conjuntos de arcos dos diversos subgrafos são todos disjuntos. Formalmente:

$$\bigcap_{k=1}^{m} A_k = \varnothing \quad \wedge \quad \forall R_k = (V_k, A_k), R_l = (V_l, A_l) \in Rotas \wedge k < l \left(A_k \cap A_l = \varnothing\right) \tag{7.3.11}$$

7.4. Modelo de Domínios Finitos

Com base nos resultados das modelações anteriores, tornou-se necessário mudar ligeiramente a abordagem de modelação sem, no entanto, abandonar a génese dos *Vehicle Routing Problems* que está associada à teoria de grafos.

Assim os domínios finitos são aplicáveis neste contexto pois, mesmo existindo dados relativos ao problema cujo domínio é contínuo, caso de todos os relativos a tempo, a grande maioria tem domínios inteiros finitos e estes permitem simular a formulação genérica: um grafo em que os vértices representam os locais e os arcos os caminhos.

Considerando os objectivos iniciais, pretende-se com este novo modelo obter soluções de forma eficiente para um problema de *Dial-a-Ride* com uma frota heterogénea, vários depósitos e conhecendo-se todos os pedidos antes da geração de soluções.

Deste modo a modelação que se descreve de seguida baseia-se na proposta de Berbeglia et al. (2010) e também nas características dos modelos com mais sucesso no âmbito das abordagens exactas, concebidos com recurso a Programação Inteira.

7.4.1. Variáveis

Neste modelo de Programação com Restrições em domínios finitos pretende-se simular uma formulação de grafos em que existem ligações entre locais que, por sua vez, constituem caminhos que formam uma solução para instâncias do DARP. Estas características, aliadas aos tempos do serviço e à qualidade do mesmo, conduziram à definição de cinco listas com cardinalidade igual à do conjunto *Vértices*, ou seja, 2*m* + 2*n*, que contêm todas as variáveis inerentes a esta modelação, nomeadamente:

- A lista *Sucessores* com domínio em *Vértices* e que, para cada vértice $i \in Vértices$ identifica o seu sucessor S_i que se encontra na i-ésima posição da lista.

- A lista *Veículos* com domínio em $\{1,...,m\}$ que, à semelhança da lista anterior, para cada vértice $i \in Vértices$, identifica o veículo que o serve.

- A lista *Passageiros* que identifica, para cada vértice $i \in Vértices$, o número de passageiros a bordo do veículo $Veículos[i]$ à saída desse vértice. O domínio desta lista pode ser definido pelo intervalo $\{0,...,C_{\max}\}$, sendo $C_{\max}$ o valor máximo do conjunto *Capacidades* apresentado acima. Pode-se ainda limitar mais o domínio de cada $P_i \in Passageiros$, definindo como $\{0,...,C_{Veículos[i]}\}$.

- A lista *Tempos* que para cada vértice $i \in Vértices$ identifica o tempo, em minutos, em que o veículo que serve *i* chega a esse mesmo local. Assim o domínio desta lista consiste no intervalo de tempo de operação, em minutos, do serviço em causa. A título de exemplo, no caso de estudo considerado, a operação da delegação Amadora – Sintra da Cruz Vermelha Portuguesa decorre entre as 7:00 e as 19:00, pelo que o domínio será $\{420,...,1140\}$, consistindo numa transformação directa das horas em minutos dado que $7 \times 60 = 420$ e $19 \times 60 = 1140$. Este domínio pode ser reduzido para cada vértice $i \in Vértices$ pois, dado um pedido $(O,D,[L_o,U_o],[L_d,U_d],T_o,T_d,C)$, pode reduzir-se o domínio de $Tempos[O]$ ao intervalo $\{L_o,...,U_o\}$ e $Tempos[D]$ ao intervalo $\{L_d,U_d\}$, uma vez que o veículo tem que visitar os vértices durante a correspondente janela temporal.

- Por fim, a lista *Esperas* que para cada vértice $i \in Vértices$, identifica o tempo de espera, em minutos, efectuado pelo veículo que serve *i* após o serviço nesse local. Ainda que indirectamente um dos objectivos seja também minimizar os tempos de espera, principalmente com utentes a bordo do veículo, o domínio da lista *Esperas* será o intervalo entre zero e a diferença entre o término e início do serviço, ou seja, $\{0,...,\text{Término do Serviço} - \text{Início do Serviço}\}$.

A partir destas listas de variáveis, em particular, da lista *Sucessores*, é possível deduzir os dados que constituem a solução do problema, isto é, o conjunto de rotas que satisfazem os pedidos. A cada veículo que compõe a frota é atribuída uma rota que poderá conter apenas o depósito inicial e depósito final, significando que o veículo não é necessário para dar resposta aos pedidos de transporte.

Assim sendo, existe um conjunto de *m* rotas, cada uma expressa através de uma lista ordenada de vértices do género $R_k = [D_k,Seq,D_{k+m}]$, com $1 \leq k \leq m$, em que D_k e D_{k+m} representam os depósitos de origem e destino do veículo. *Seq* representa uma sequência

potencialmente vazia de sucessores do tipo $\langle S[D_k], S[S[D_k]], ... \rangle$, sendo $S[i]$ o sucessor do vértice *i*. Deste modo cada lista R_k, com 1 ≤ *k* ≤ *m* e domínio em *Vértices* será composta pelo menos por dois vértices, os depósitos inicial e final, aos quais acresce um número par de elementos já que por cada pedido que um veículo satisfaça, são sempre dois os vértices. Assim, a sua dimensão máxima é dada por 2*n* + 2, caso este visite todos os vértices pertencentes a *Locais* e tendo em conta que sai do depósito D_k e chega ao D_{k+m}.

As rotas fornecem informações que servirão para a modelação das restrições do problema, dos quais se destacam as cargas dos veículos ao longo do percurso. Quando os veículos deixam o depósito a sua carga é nula, isto é, não transportam nenhum passageiro. Caso a rota do veículo não seja vazia, o vértice *i* que sucede o depósito inicial corresponderá a um local onde se irão buscar passageiros e, sendo o pedido que o contempla $(O, D, [L_o, U_o], [L_d, U_d], T_o, T_d, C)$, o valor de $Passageiros[i]$, com *i* = *O*, será igual a *C*. Este processo repete-se para os restantes vértices da rota. Quando se trata de um vértice destino, isto é, um vértice onde se deixam passageiros, a carga será decrementada do mesmo valor *C*. Por fim, à chegada do depósito, o veículo não pode levar passageiros a bordo.

De forma idêntica é possível realizar uma aproximação temporal do serviço, tendo em conta o tempo de viagem entre vértices. Assim, com base na rota $R_k = [v_1, ..., v_j]$, correspondente ao veículo *k* e com v_j sendo um vértice do conjunto *Vértices* e 2 ≤ *j* ≤ 2*n* + 2, bem como na matriz *T* que mapeia os tempos entre os diversos vértices que compõem o problema, é possível deduzir que $Tempos[S[i]] = (Tempos[i] + d_i + Esperas[i]) + T[i, S[i]]$, com $i \in Vértices$. Nesta expressão, $S[i]$ representa o sucessor do vértice *i*, d_i a duração do serviço no vértice *i*, $Esperas[i]$ um possível tempo de espera após servir *i* e $T[i, S[i]]$ a distância temporal entre o vértice *i* e o seu sucessor, todos em minutos.

Directa ou indirectamente, todas as variáveis se relacionam com *Sucessores*, em particular as rotas que são deduzidas com base nesta lista. Deste modo, conhecendo o primeiro elemento de uma dada rota, isto é, o local de onde o veículo parte, é possível deduzir-se todo o percurso. Esta dedução é efectuada recorrendo ao predicado *element* que tem a capacidade de garantir que o elemento contido numa determinada posição de uma lista tem o valor pretendido. De modo a exemplificar o conceito, considere-se o seguinte exemplo:

Seja *L* uma lista de inteiros, $L = [A, B, C, ...]$. Para garantir que *A* toma o valor 5 pode recorrer-se ao predicado *element* da seguinte forma: $\text{element}(1, L, 5)$. Ou seja, diz-se que a primeira posição da lista *L* deve conter o valor 5.

Assim, pode-se deduzir o percurso das rotas da seguinte forma:

element(*Nó Inicial*,*Sucessores*,*Nó* 2)
element(*Nó* 2,*Sucessores*,*Nó* 3)
...
element(*Penúltimo Nó*,*Sucessores*,*Nó Final*)

Neste caso, sendo *Nó Inicial* o vértice correspondente ao depósito inicial, o predicado acima garante que o seu sucessor será *Nó 2* e assim sucessivamente.

A partir das rotas e de forma idêntica, calculam-se os tempos de viagem e de espera, bem como a carga dos veículos em cada ponto da rota. O mapeamento destas listas, indexadas por vértice, é feito também com recurso ao predicado *element*. Para a lista *Passageiros*, por exemplo, para cada rota R_k, com $1 \leq k \leq m$, recorre-se a uma instrução $\text{element}(r,R_k,v_r)$, $\forall r \in [1,\text{length}(R_k)]$, sendo v_r o r-ésimo vértice da rota k e $\text{length}(R_k)$ o número de vértices que compõem a rota em causa. Calculada a carga como explicitado anteriormente e sendo esta c_r, recorre-se novamente ao predicado *element* para mapeá-la na lista *Passageiros*: $\text{element}(v_r,\textit{Passageiros},c_r)$. Um processo idêntico é usado para as listas *Tempos* e *Esperas*. Para a lista *Veículos* sabe-se que os vértices presentes na rota *k* são servidos pelo veículo *k*, dada a ligação entre rotas, veículos e depósitos.

7.4.2. Restrições

Com base em todas as variáveis apresentadas acima, é possível expressar as restrições do modelo:

i. O número de ocorrências dos depósitos iniciais na lista *Sucessores* é igual a zero já que estes não são sucessores de nenhum outro vértice. Para tal pode recorrer-se ao predicado $\text{occurrences}(V,L,N)$ que restringe o valor *V* a ocorrer *N* vezes na lista *L*. Formalmente, sendo *m* o número de veículos,

$$\forall k \in \textit{Depósitos} \wedge k \leq m, \text{occurrences}(k,\textit{Sucessores},0) \qquad (7.4.1)$$

ii. Por outro lado, optou-se por definir que o sucessor de um depósito final é o próprio, ou seja, $\textit{Sucessores}[k+m] = k+m$, com $k \in \textit{Depósitos} \wedge k \leq m$. Esta opção leva a que os depósitos finais apareçam duas vezes na lista *Sucessores* pois este ocorre também como sucessor do vértice que o antecede na rota correspondente. Este facto pode ser expresso da seguinte forma, recorrendo novamente ao predicado *occurrences*:

$$\forall k \in \textit{Depósitos} \wedge k \leq m, \text{occurrences}(k+m,\textit{Sucessores},2) \qquad (7.4.2)$$

iii. Para todos os restantes vértices, isto é os vértices do conjunto *Locais*, existe uma restrição idêntica que garante que estes apenas ocorrem uma única vez na lista *Sucessores*, garantindo que são visitados exactamente uma vez:

$$\forall i \in Locais, \text{occurrences}(i, Sucessores, 1) \tag{7.4.3}$$

iv. A restrição seguinte garante que o sucessor do último vértice de uma dada rota, o depósito final, é o próprio,

$$\forall k \in Depósitos \wedge k \leq m, Sucessores\left[k+m\right] = k+m \tag{7.4.4}$$

v. Exceptuando os depósitos finais, nenhum outro vértice pode ser sucessor de si próprio, pelo que

$$\forall i \in Locais \cup Depósitos : i \leq m \vee i > 2m, Sucessores\left[i\right] \neq i \tag{7.4.5}$$

vi. Algo que também restringe a lista *Sucessores* é o facto de um "vértice origem" não poder ser sucessor do "vértice destino" do mesmo pedido, ou seja:

$$\forall p \in Pedidos, Sucessores\left[\text{Destino}(p)\right] \neq \text{Origem}(p) \tag{7.4.6}$$

vii. A relação entre veículos, depósitos e rotas pode começar a ser construída garantindo que o veículo que serve o depósito *k* é o que serve *k* + *m*, sabendo de antemão que se trata do veículo *k*, 1 ≤ *k* ≤ *m*. Formalmente:

$$\forall k \in Depósitos \wedge k \leq m, Veículos[k] = Veículos[k+m] = k \tag{7.4.7}$$

viii. Ainda no que diz respeito aos veículos, o mesmo veículo deve recolher o utente na sua origem e levá-lo até ao local de destino, ou seja,

$$\forall p \in Pedidos, Veículos\left[\text{Origem}(p)\right] = Veículos\left[\text{Destino}(p)\right] \tag{7.4.8}$$

ix. Estas restrições relativas à lista *Veículos* podem ser generalizadas para todos os vértices que compõem o problema da seguinte forma:

$$\forall i \in Vértices, Veículos\left[Sucessores\left[i\right]\right] = Veículos\left[i\right] \tag{7.4.9}$$

x. Em complemento às restrições acima, há que referir que todos os nós da mesma rota são servidos pelo mesmo veículo, sendo que a rota R_k é servida pelo veículo *k*, com 1 ≤ *k* ≤ *m*. Formalmente,

$$\forall k \in Veículos\left(\forall j \in R_k, Veículos\left[j\right] = k\right) \tag{7.4.10}$$

xi. Todas as rotas têm um tamanho fixo que é igual ao maior número possível de vértices que uma rota pode conter, isto é, $2n + 2$. Como na generalidade dos casos o número de vértices que compõem a rota é inferior a $2n + 2$, a lista que representa a rota é preenchida com o depósito final para que se possa perceber que a rota terminou. Exemplificando, seja $2n + 2 = 6$, os vértices 5 e 7 servidos pela rota R_1 e os vértices 6 e 8 pela rota R_2. Os depósitos inicial e final para a rota R_1 correspondem ao 1 e 3, e para a rota R_2 a 2 e 4, respectivamente. Deste modo a primeira rota é representada da seguinte forma: $R = [1,5,7,3,3,3]$. Esta particularidade é conseguida através da seguinte restrição:

$$\forall R_k = [r_1,...,r_{2n+2}], 1 \le k \le m \left(\forall j \in \{2,...,2n+1\} \begin{pmatrix} (r_j = D_{k+m}) \\ \Rightarrow \\ (r_{j+1} = D_{k+m}) \end{pmatrix} \right) \quad (7.4.11)$$

xii. Sendo $\text{myLength}(R_k)$ o número de vértices diferentes presentes na rota *k*, esse número será sempre par e superior ou igual a dois tal como nas restrições (7.2.3) e (7.3.9) apresentadas anteriormente no contexto do modelo de grafos. De uma formal, pode dizer-se que:

$$\forall 1 \le k \le m, \text{myLength}(R_k) \ge 2 \wedge \text{myLength}(R_k) \%2 = 0 \quad (7.4.12)$$

xiii. Ainda que acabe por reiterar alguma informação contida já na restrição (7.4.8), os vértices de um determinado pedido devem estar na mesma rota, o que implica que sejam servidos pelo mesmo veículo como referido em (7.4.10). Formalmente

$$\forall p \in Pedidos, \text{Origem}(p) \in R_k, 1 \le k \le m \Leftrightarrow \text{Destino}(p) \in R_k \quad (7.4.13)$$

xiv. Dadas as características das rotas, é possível definir tanto o seu primeiro vértice, como o último, algo que pode ser expresso da seguinte forma:

$$\forall k \in Depósitos \wedge k \le m, R_k[1] = k \wedge R_k[2n+2] = k + m \quad (7.4.14)$$

xv. As diferentes rotas devem servir todos os pedidos passando, consequentemente, por todos os vértices que compõem o problema. Assim, pode dizer-se que

$$\sum_{k=1}^{m} \text{myLength}(R_k) = \text{length}(Vértices) \quad (7.4.15)$$

$$\bigcup_{k=1}^{m} R_k = Vértices \wedge \bigcap_{k=1}^{m} R_k = \varnothing \quad (7.4.16)$$

e

$$\forall R_k, R_l \in Rotas \wedge k < l \left(R_k \cap R_l \neq \varnothing \right) \tag{7.4.17}$$

xvi. Há também que garantir que a capacidade de um dado veículo nunca é excedida ao longo da rota, havendo também uma relação entre as cargas e descargas. Assumindo *C* como a carga máxima do veículo que serve o vértice *i* e $c_{Sucessores[i]}$ o número de utentes que entra ou sai do veículo, número este que depende do tipo de pedido que caracteriza o vértice *i*, tomando um valor positivo no primeiro caso e negativo no segundo. Assim,

$$\forall i \in Vértices \left(\begin{array}{c} Passageiros\left[Sucessores\left[i \right] \right] = Passageiros\left[i \right] + c_{Sucessores[i]} \\ \wedge \\ Passageiros\left[i \right] \leq C \end{array} \right) \tag{7.4.18}$$

xvii. Por uma questão de qualidade de serviço para o utente, o tempo que este passa em transporte não deve exceder o limite máximo *L* que tipicamente é igual ao parâmetro *Tempo Máximo de Viagem* ou, como referido na Secção 7.1.4, tomará um valor geometricamente proporcional à distância temporal entre os locais. Por exemplo, para o pedido *p*, tal valor pode ser dado por

$$L = T\left[\text{Origem}(p), \text{Destino}(p) \right] + \max\left(X \times T\left[\text{Origem}(p), \text{Destino}(p) \right], T \right),$$

com $X = 0{,}4$ e $T = 25$ minutos. Formalmente,

$$\forall p \in Pedidos \left(Tempo\left[\text{Destino}(p) \right] - Tempo\left[\text{Origem}(p) \right] \leq L \right) \tag{7.4.19}$$

xviii. Há também que garantir que o tempo em que o veículo chega à origem de um pedido é menor ou igual ao tempo a que esse mesmo veículo alcança o vértice de destino, tendo em conta o tempo de viagem entre ambos bem como eventuais tempos de espera e de serviço. Assim garante-se que o vértice de origem de um pedido é sempre servido primeiro que o seu vértice de destino:

$$\forall p \in Pedidos \left(\begin{array}{c} Tempo\left[\text{Origem}(p) \right] \\ \leq \\ Tempo\left[\text{Destino}(p) \right] \\ - \\ \left(T\left[\text{Origem}(p), \text{Destino}(p) \right] + d_{Origem(p)} + Esperas\left[\text{Origem}(p) \right] \right) \end{array} \right) \tag{7.4.20}$$

xix. Analogamente para os sucessores tem-se que o tempo de chegada do veículo ao vértice *i* é igual ao tempo de chegada ao seu sucessor, subtraindo-se os tempos de viagem, de espera e de serviço.

$$\forall i \in Vértices, Tempo[i] = \begin{pmatrix} Tempo[Sucessores[i]] \\ - \\ T[i, Sucessores[i]] + d_i + Esperas[i] \end{pmatrix} \quad (7.4.21)$$

xx. Por uma questão de modelação e coerência, considera-se que a hora a que os depósitos iniciais são servidos é a hora inicial do serviço dada por *Início do Serviço*.

$$\forall k \in Depósitos \wedge k \leq m, Tempo[k] = \text{Início do Serviço} \quad (7.4.22)$$

7.4.3. Função Objectivo

Como referido anteriormente, pretende-se minimizar os custos de operação, sem esquecer a qualidade do serviço, sendo que o primeiro critério está directamente relacionado com a distância percorrida. Já a qualidade de serviço pode ser expressa de diversas formas (Paquette et al., 2009), das quais se destacam as seguintes:

i. Diferença entre o tempo de viagem realizado pelo veículo entre o "vértice origem" e o "vértice destino" de um pedido e o tempo de viagem directo entre os locais em causa. Assim, há que contabilizar este diferencial para todos os pedidos, da seguinte forma:

$$\sum_{p \in Pedidos} \begin{pmatrix} \left(Tempo[\text{Destino}(p)] - Tempo[\text{Origem}(p)]\right) \\ - \\ T[\text{Origem}(p), \text{Destino}(p)] \end{pmatrix} \quad (7.4.23)$$

ii. Considerando que a qualidade de serviço pode ser estimada através do tempo de espera a bordo do veículo, isto é, considera-se penalizador para a qualidade de serviço o facto de estarem utentes a bordo do veículo e este estar parado, num compasso de espera. A quantidade deste tempo improdutivo pode ser expressa por:

$$\sum_{i \in Vértices} \left(Passageiros[i] \times Esperas[i]\right) \quad (7.4.24)$$

Devido ao facto de representarem conceitos diferentes e contribuírem complementarmente para a qualidade de serviço optou-se por considerar ambas as expressões acima para a função objectivo. Considerando r_i como o i-ésimo vértice da rota *R*, é possível formalizar a função $\text{tempo}(R)$, que retorna o tempo da rota *R*, da seguinte forma:

$$\text{tempo}(R) = \sum_{i=1}^{\#R-1} T[r_i, r_{i+1}] \tag{7.4.25}$$

Recorrendo a esta função e assumindo que a expressão (7.4.23) se traduz por *Diferencial* e a (7.4.24) por *EsperasPenalizadas*, é possível expressar a função objectivo usando o método das som agregadas referido na Secção 5.1, da seguinte forma:

$$\min Z = \sum_{k=1}^{m} \text{tempo}(R_k) + (\textit{Diferencial} + \textit{EsperasPenalizadas}) \tag{7.4.26}$$

O facto de existirem dois termos na função objectivo que se referem à qualidade de serviço, pode enviesar a solução obtida uma vez que se contempla apenas um termo para a quantificação do custo da solução, expresso em função do tempo necessário para viajar entre os diversos locais que é linear relativamente à despesa contraída pelo serviço. No entanto, este equilíbrio pode ser estabelecido consoante as necessidades específicas das entidades prestadoras deste tipo de serviço, considerando factores multiplicativos que reflictam a importância dada a cada um dos termos da função:

$$\min Z = \alpha \sum_{k=1}^{m} \text{tempo}(R_k) + (\beta \textit{Diferencial} + \delta \textit{EsperasPenalizadas}) \tag{7.4.27}$$

Estes coeficientes deverão ser valores positivos e deverão ter em conta, para além das preferências do decisor, a grandeza de cada termo.

7.4.4. Heurísticas

Como abordado no Capítulo 6, as heurísticas constituem um mecanismo fulcral para conduzir a pesquisa e, ainda que não tenham nenhum fundamento formal, tipicamente conseguem conduzir a resultados de forma mais rápida.

Para este modelo foram pensadas e concebidas algumas heurísticas, tanto para a escolha de variável como de valor, sendo que para o primeiro caso se destacam:

i. Uma variante da *First-Fail* (Haralick & Elliott, 1980) que além de seleccionar a variável cujo domínio é mais pequeno, toma também em consideração o número de restrições que lhe estão associadas.

ii. Escolha da variável de forma sequencial, isto é, de acordo com o percurso que vai sendo estabelecido pelas rotas. Inicialmente a heurística iria considerar a primeira rota, R_1, e sabendo que o seu primeiro vértice é 1, iria prosseguir para a variável que diz respeito ao seu sucessor, isto é, $Sucessores[1]$. De seguida iria considerar $Sucessores[Sucessores[1]]$ e assim sucessivamente até chegar ao depósito final, isto é, o último vértice da rota. Depois passaria para a rota seguinte aplicando o mesmo processo. Esta heurística, como é perceptível, tentar mapear o raciocínio humano na definição das rotas que é feito de forma sequencial.

Por outro lado destacam-se também as heurísticas para a escolha de valor cujo intuito é de facto acelerar a obtenção de soluções admissíveis enquanto que as acima, ainda que partilhem do mesmo objectivo, permitem, acima de tudo, cortar ramos da árvore de pesquisa. Deste modo segue a descrição de três heurísticas utilizadas no modelo:

iii. A primeira heurística foi baseada no conceito *Divide and Conquer* dado que divide o domínio da variável em causa em diferentes grupos que são testados individualmente. Em rigor, trata-se de uma heurística sem grande linha condutora que atribui, incrementalmente, os valores presentes no domínio. O facto de dividir o domínio em diferentes grupos permite apenas detectar atribuições inválidas de forma mais rápida que uma simples enumeração sequencial contemplando todo o domínio.

iv. Por outro lado, foi considerada uma outra heurística que dá primazia aos vértices de *delivery*, isto é, vértices onde se vão deixar os doentes, cujos respectivos vértices de *pickup* já estão instanciados. Seguidamente a prioridade passa pelos vértices de *pickup*, onde se vão buscar os utentes. Em qualquer das situações, as variáveis são instanciadas com valores aleatórios do domínio já priorizado.

   De forma mais formal, seja *D* o conjunto de "vértices destino" cuja origem já foi visitada, caso $D \cap Domínio(v) \neq \varnothing$, com *v* sendo a variável a instanciar, é seleccionado um valor da intersecção referida acima. Caso esta seja vazia, será escolhido um valor do conjunto *P*, o de "vértices origem" ainda não instanciados. Caso este último conjunto seja vazio, o valor escolhido corresponderá ao depósito final da rota em causa.

v. Por fim, surge outra heurística que se baseia na anterior, mas cuja prioridade de atribuição de valores não está relacionada com o tipo de vértice em causa, mas sim com as suas janelas temporais. Assim a primazia é dada aos vértices que devem ser servidos primeiro de acordo com os *timings* do serviço.

Formalmente, para a variável *v* e com *S* sendo o conjunto de vértices ordenados pelos respectivos tempos de serviço, serão considerados, incrementalmente, os valores do conjunto $S \cap Domínio(v)$.

Classificando as heurísticas para escolha de variável, i e ii, como *FF* e *S* e as de escolha de valor, iii, iv e v, como *DC*, *TP* e *TM*, respectivamente, as quatro combinações consideradas são: *FF-DC*, *FF-TP*, *S-TP* e *S-TM*.

7.5. Redução do Espaço de Pesquisa

A natureza combinatória do *Dial-a-Ride Problem* leva a que a ramificação da árvore de pesquisa seja muito elevada, pelo que todas as técnicas que possam reduzi-la poderão ser um contributo importante para alcançar soluções neste contexto (Rossi et al., 2006). No nosso contexto e dada a utilização da Programação com Restrições, foram aplicadas algumas técnicas baseadas nas formulações acima e nos conceitos presentes na literatura (Berbeglia et al., 2010; Cordeau & Laporte, 2003; Cordeau, 2006; Melachrinoudis et al., 2007; Parragh, 2011).

Uma das técnicas mais usadas para reduzir o espaço de pesquisa consiste na identificação de simetrias (Rossi et al., 2006) que possam ser quebradas de modo a não serem geradas soluções equivalentes, isto é, cuja utilidade e resultados nada acrescentam por serem iguais aos de outras soluções. Este tipo de situações ocorre particularmente com os veículos pois, ao existirem veículos idênticos e sendo a utilização de qualquer um deles válida, é esperado que se obtenham soluções simétricas. Assim, e de modo a obter tal redução, impõe-se que os tempos de saída dos depósitos iniciais são todos crescentes (para veículos do mesmo tipo), isto é, o primeiro veículo sairá primeiro que o segundo, o segundo que o terceiro, etc… Formalmente, e assumindo uma frota homogénea por uma questão de simplicidade, esta restrição pode ser expressa, para o modelo de domínios finitos, da seguinte forma:

$$\forall k \in Depósitos \wedge k \leq m\left(\forall j \in \{k+1,...,m\}, Esperas[j] < Esperas[k]\right)$$

Numa frota heterogénea esta restrição poderá também ser aplicada de forma idêntica para veículos do mesmo tipo. Assumindo $\text{tipo}(V)$ como uma função que devolve o tipo do veículo $V \in Veículos$, a restrição pode formalizar-se como:

$$\forall V_k, V_l \in Veículos \wedge k < l\left(\text{tipo}(V_k) = \text{tipo}(V_l) \Rightarrow Esperas[k] < Esperas[l]\right)$$

Como alternativa, para modelações que não contemplem os tempos de espera, poderá optar-se por estabelecer, mediante uma ordenação dos veículos que compõem a frota, que a

cardinalidade das rotas dos k-ésimos veículos são superiores ao dos seguintes. Contemplando a situação em que a cardinalidade de duas rotas distintas é igual, poderá cortar-se simetrias obrigando a que o primeiro vértice servido (não considerando o ponto de partida) terá que ter uma codificação inferior, por exemplo, que o correspondente vértice da rota seguinte.

Por outro lado, e como já foi referido no decorrer deste capítulo, existem ligações que não fazem sentido do ponto de vista do problema, por questões de *timing* do serviço ou mesmo devido à carga do veículo. Tendo presentes alguns conceitos e variáveis apresentados na Secção 7.1 como a duração do serviço no vértice *i,* d_i, a janela temporal $[L_i,U_i]$ associada a esse mesmo vértice e também o tempo de viagem $T[i,j]$ entre os vértices *i* e *j*, é intuitivo perceber que os seguintes arcos devem ser eliminados do espaço de pesquisa:

- (k,d), com $k \in Depósitos \wedge k \leq m$ e $d \in Destinos$, pois nenhum veículo terá como primeira paragem um local onde se deixam passageiros.
- De forma idêntica todos os arcos do tipo (o,k), com $k \in Depósitos \wedge k > m$ e $o \in Origens$, e do tipo $(n+i,i)$, com $i \in Origens$ e *n* sendo o número total de pedidos de transporte.
- As janelas temporais associadas a cada vértice podem também levar a arcos do tipo (i,j), com $i,j \in Vértices$, sejam removidos se $U_i + d_i + T[i,j] > L_j$.
- Ainda baseado nos *timings* do serviço e nas janelas temporais, os arcos (i,j) e $(j,n+i)$, com $i \in Origens$, $j \in Vértices$ e *n* como o número de pedidos de transporte, são ambos não admissíveis caso

$$T[i,j] + d_j + T[j,n+i] > Tempo\ Máximo\ de\ Viagem$$

Ainda neste âmbito, será interessante apresentar o conceito de caminho não admissível. Este define-se como uma sequência de vértices que violam alguma das restrições, tipicamente ligadas aos *timings* ou à capacidade do veículo. Com base neste conceito é também possível excluir os arcos:

- $(i,n+j)$ se o caminho $\langle j,i,n+j,n+i \rangle$ não for admissível.
- $(n+i,j)$ se o caminho $\langle i,n+i,j,n+j \rangle$ não for admissível.
- (i,j) se os caminhos $\langle i,j,n+i,n+j \rangle$ e $\langle i,j,n+j,n+i \rangle$ não forem admissíveis.
- $(n+i,n+j)$ se os caminhos $\langle i,j,n+i,n+j \rangle$ e $\langle j,i,n+i,n+j \rangle$ não forem admissíveis.

Além de permitir eliminar os arcos referidos acima, o conceito de caminho não admissível permite também descobrir pares de pedidos incompatíveis, ou seja, pares de pedidos que não

podem ser servidos pelo mesmo veículo devido aos *timings* do serviço. Os pedidos *i* e *j* são considerados incompatíveis caso nenhum dos caminhos entre eles seja admissível. Caso os caminhos: $\langle i,j,n+i,n+j\rangle$, $\langle i,j,n+j,n+i\rangle$, $\langle j,i,n+i,n+j\rangle$, $\langle j,i,n+j,n+i\rangle$, $\langle i,n+i,j,n+j\rangle$ e $\langle j,n+j,i,n+i\rangle$ sejam não admissíveis, os oito arcos entre *i*, *j*, *n* + *i* e *n* + *j* podem também ser removidos do espaço de pesquisa.

Verificar se um dado caminho é admissível não é uma tarefa propriamente trivial dado o número de restrições, janelas temporais e tempos de espera pelo que deve basear-se no conceito de *Forward Time Slack* proposto por Savelsbergh (1992) e generalizado por Cordeau (2006). Este conceito diz que, para o caminho $\langle i,j,n+i,n+j\rangle$, por exemplo, será necessário verificar se cada troço cumpre as restrições temporais e de carga do veículo. Para as primeiras, terão que se verificar as seguintes condições:

$$Tempos[i]+d_i+Esperas[i]+T[i,j]\in[L_j,U_j] \tag{7.4.28}$$

com d_i sendo a duração do serviço no vértice *i* e $[L_j,U_j]$ a janela temporal associada ao vértice *j*. Validado este primeiro troço, torna-se necessário validar o seguinte tendo em conta o anterior, ou seja:

$$\begin{pmatrix} Tempos[i]+D_i+Esperas[i]+T[i,j] \\ + \\ D_j+Esperas[j]+T[j,n+i] \end{pmatrix}\in[L_{n+i},U_{n+i}] \tag{7.4.29}$$

Este tipo de condições acaba por traduzir os tempos de chegada aos diversos vértices que devem pertencer ao intervalo $[L,U]$ que representa o intervalo de tempo em que é permitido chegar a um dado local. Por fim, a condição para o último troço do caminho é dada por:

$$\begin{pmatrix} Tempos[i]+D_i+Esperas[i]+T[i,j] \\ + \\ D_j+Esperas[j]+T[j,n+i] \\ + \\ D_{n+i}+Esperas[n+i]+T[n+i,n+j] \end{pmatrix}\in[L_{n+j},U_{n+j}] \tag{7.4.30}$$

Adicionalmente, as restrições relativas à capacidade dos veículos poderão apenas invalidar caminhos em que os "vértices origem" sejam consecutivos, como é o caso do caminho exemplificado acima, $\langle i,j,n+i,n+j\rangle$.

7.6. Conclusões

O presente capítulo destinou-se à apresentação de várias modelações, sendo que entre os modelos apresentados, o último, de domínios finitos, destaca-se pelo seu desempenho na obtenção de soluções relativamente às abordagens que recorrem à biblioteca GRASPER, como poderá ser verificado no próximo capítulo.

Nesse capítulo serão apresentados os diversos resultados dos testes computacionais que, entre outros, justificarão a concepção de três modelos diferentes e permitirão averiguar o seu desempenho e também a sua aplicabilidade tanto ao caso de estudo considerado, como a outras instâncias do *Dial-a-Ride Problem*.

8

8. Testes Computacionais

No presente capítulo serão apresentadas as instâncias que servirão de *input* aos modelos de Programação com Restrições apresentados no capítulo anterior, de modo a averiguar seu desempenho e a qualidade das soluções geradas.

Será realizado um primeiro teste aos três modelos propostos segundo uma instância criada no contexto desta dissertação e apresentada na Secção 8.1.1. De acordo com este teste, o modelo de domínios finitos irá destacar-se dos restantes o que levará a que seja sujeito às instâncias referência apresentadas na Secção 8.1.2 e utilizadas em diversos artigos na área do *Dial-a-Ride Problem* (Berbeglia et al., 2010; Cordeau, 2006; Ropke et al., 2007). Os testes realizados com estas instâncias pretendem determinar quais as heurísticas que relevam um comportamento mais interessante, bem como obter uma percepção do desempenho do modelo relativamente a outras abordagens presentes na literatura.

Por fim, serão resolvidas as instâncias da delegação Amadora – Sintra da Cruz Vermelha Portuguesa e os resultados serão comparados com os obtidos em Loureiro (2010).

8.1. Instâncias

Para testar os modelos desenvolvidos serão consideradas diversas instâncias com diferentes objectivos.

Numa primeira fase será considerada uma instância de teste concebida propositadamente para avaliar a qualidade conceptual do modelo no que diz respeito à capacidade de lidar com uma frota heterogénea e também com pedidos incompatíveis, problemática apresentada na Secção 7.5. As características desta instância (apresentadas na Secção 8.1.1) e os resultados

obtidos (Secção 8.2.1), revelaram o fraco desempenho dos dois modelos de grafos apresentados nas Secções 7.2 e 7.3.

Seguidamente serão utilizadas instâncias criadas por Cordeau e Laporte (2003) pelo facto de se terem tornado uma referência para o *Dial-a-Ride Problem*, sendo utilizadas por diversos autores que estudam este problema (Berbeglia et al., 2010; Cordeau & Laporte, 2003; Cordeau, 2006; Ropke et al., 2007). Com estas instâncias pretende-se avaliar o modelo de domínios finitos, de forma mais aprofundada, antes da sua aplicação do caso de estudo.

Em última instância serão considerados quatro dias de trabalho representativos da operação da delegação Amadora – Sintra da Cruz Vermelha Portuguesa e presentes em Loureiro (2010) e os resultados serão apresentados na Secção 8.2.3.

8.1.1. Instância de Teste

Como referido anteriormente esta instância surge com o intuito de avaliar a qualidade dos diferentes modelos concebidos tanto nível conceptual, como a nível de desempenho, sendo o primeiro responsável pela qualidade das soluções geradas e da verificação das restrições. Por outro lado, a nível de desempenho pretende-se apenas aferir se o modelo consegue gerar soluções em tempo “aceitável”.

Trata-se de um exemplo de dimensão reduzida na área do transporte de doentes que apresenta algumas particularidades interessantes no âmbito da heterogeneidade da frota, considerando veículos com diferentes capacidades, bem como ao nível dos pedidos incompatíveis, isto é, pedidos que não podem ser servidos pela mesma viatura devido a limitações de tempo ou de lugares (ver Secção 7.5).

Composto apenas por 8 pedidos de transporte, dos quais metade são pedidos para uma entidade de tratamento e os restantes, pedidos de regresso aos locais de residência, este exemplo considera uma frota composta por duas viaturas com capacidades distintas, uma que dispõe de 1 lugar sentado e outra de 2, podendo realizar transporte de doentes entre as 7:00 e as 19:00 horas. Os tempos de serviço são variáveis entre os diversos vértices de acordo com o utente e o local em causa. No que diz respeito aos passageiros, são considerados casos em que o utente se faz acompanhar por um familiar ou *caregiver* para o tratamento. Para informações mais específicas acerca deste exemplo de teste pode ser consultado o Anexo B onde se descrevem os 8 pedidos de transporte.

8.1.2. Instâncias Referência

Estas instâncias foram consideradas neste trabalho pelo facto de serem classificadas como *benchmark* no que diz respeito ao *Dial-a-Ride Problem*, contemplando uma vasta gama de características que serão interessantes para avaliar os modelos antes da sua aplicação ao caso de estudo.

Concebidas por Cordeau (2006), e disponíveis *online*[2], estas instâncias são compostas por dois conjuntos diferentes, sendo um mais vocacionado para problemas onde são usados veículos com menor capacidade de transporte, tipicamente ligeiros (denominar-se-á por *conjunto a*), e o outro contemplando veículos com uma maior capacidade (*conjunto b*). No contexto desta dissertação ambos os conjuntos são interessantes dada a heterogeneidade da frota à disposição da delegação Amadora – Sintra da Cruz Vermelha Portuguesa (ver Tabela 2.1).

Todas as instâncias presentes nestes dois conjuntos foram concebidas aleatoriamente, estando as diversas localizações que as compõem situadas num quadrado $[-10,10]\times[-10,10]$, sendo que o único depósito real existente se localiza no centro do quadrado, com coordenadas $(0,0)$ (Cordeau, 2006). As janelas temporais contêm um intervalo de 15 minutos, o que significa que os utentes poderão chegar no máximo com 15 minutos de antecedência às entidades de saúde e, após o seu tratamento, apenas é tolerável que esperem no máximo 15 minutos por transporte de regresso às suas residências. O período de operação varia de instância para instância num intervalo entre as 4 e as 12 horas.

A principal diferença entre estes dois conjuntos de teste prende-se com a capacidade das viaturas que compõem a frota (homogénea) que dará resposta aos pedidos de transporte, 3 lugares sentados para o primeiro conjunto e 6 para o segundo. No primeiro conjunto o número de passageiros associado a cada pedido é sempre igual a 1, isto é, apenas o utente se desloca dentro da viatura de serviço e é estimado um tempo médio de serviço de 3 minutos. Por outro lado, no segundo conjunto de instâncias, o número de passageiros varia entre 1 e 6, sendo este último a capacidade da viatura, e o tempo de serviço associado a cada pedido varia linearmente consoante o número de passageiros. Outra diferença entre os dois conjuntos de instâncias é o tempo máximo de viagem para cada utente, fixo em 30 minutos para o primeiro conjunto e em 45 para o segundo.

[2] Disponíveis *online* em http://neumann.hec.ca/chairedistributique/data/darp/ (Acedido a 23/9/2012)

Pelo facto destas instâncias não estarem relacionadas com dias de serviço ou nenhuma outra característica, o seu nome será o único indicador superficial do conteúdo, sendo da forma "*am-n*" ou "*bm-n*", com *m* o número de veículos que compõem a frota, *n* o número de pedidos de transporte e a primeira letra como indicativo do conjunto em que se inserem. Mais detalhes acerca destas instâncias poderão ser consultados no Anexo C e em Cordeau (2006)

8.1.3. Instâncias da Cruz Vermelha Portuguesa

As instâncias disponibilizadas pela delegação Amadora – Sintra da CVP baseiam-se em quatro dias representativos da sua actividade para uma ambulância do tipo A2-1 (ver Tabela 2.1), consistindo assim numa redução do problema real enfrentado pela delegação.

Estas instâncias diárias abrangem, no fundo, os pedidos de transporte realizados à Cruz Vermelha Portuguesa para os quatro dias em questão. O número de pedidos associado a cada um dos dias não varia muito, sendo o terceiro dia o menos concorrido, com apenas 11 pedidos de transporte, enquanto que no primeiro dia, o mais concorrido, existem 21 pedidos. Nos restantes dias são contemplados 19 pedidos. Para estes quatro dias apenas é disponibilizada uma ambulância com uma capacidade de 7 lugares sentados para dar resposta aos pedidos de transporte. Outro dado comum a todas as instâncias da CVP é o tempo de serviço para o qual se considera um valor médio de 5 minutos, bem como o tempo máximo de viagem para cada utente que é fixado em 60 minutos. Este intervalo é também tido em conta como margem de chegada aos locais, isto é, um utente poderá chegar, no máximo, uma hora mais cedo que o início do seu tratamento, devendo a ambulância da CVP chegar para recolher o utente até uma hora depois de terminado o tratamento. Nos próximos parágrafos seguem-se pequenas descrições dos quatro dias de serviço da CVP apresentados com mais detalhe no Anexo A.

Das instâncias representativas da actividade da delegação da CVP em estudo, a correspondente ao primeiro dia é a que tem mais pedidos de transporte, com um total de 21. Destes pedidos, 10 representam a necessidade de utentes se deslocarem dos seus locais de residência para as entidades prestadoras de cuidados de saúde e 11 para o percurso inverso. Todos os pedidos se centram em apenas duas entidades, a Movi Física e Reabe e podem ser consultados no Anexo A.2.

O segundo dia é composto por 19 pedidos de transporte, sendo 9 referentes a deslocações da residência para entidades de saúde e 10 para o percurso inverso, sendo no entanto o número de entidades visitadas superior ao do primeiro dia, como pode ser analisado através do Anexo A.3.

A instância referente ao terceiro dia de serviço é a que contém apenas 11 pedidos de transporte, sendo a maioria referente a deslocações para entidades. Mais precisamente, apenas 4 dos 11 pedidos são de regresso às residências dos utentes (Anexo A.4).

Por fim, o quarto dia tem o mesmo número de pedidos que o segundo dia, diferindo apenas pelo facto de existirem mais pedidos das residências para as entidades, 10 nesta situação e apenas 9 na situação inversa (Anexo A.5).

8.2. Resultados Computacionais

Os modelos desenvolvidos para o *Dial-a-Ride Problem* foram implementados em Prolog com recurso à ferramenta gratuita ECLiPSe 6.1#112. O computador utilizado para a realização de todos testes foi um Intel® Core™2 Q6600, com 2.7 GHz e 4 GB de memória RAM.

Numa primeira fase serão avaliados os três modelos desenvolvidos e apresentados no Capítulo 7 através da obtenção de uma solução para a instância de teste apresentada anteriormente (Secção 8.1.1).

Após esta primeira fase, na qual os modelos de grafos não demonstraram um desempenho satisfatório, as instâncias referência serão submetidas ao modelo de domínios finitos com o objectivo de avaliar as combinações heurísticas propostas na Secção 7.4.4 e também a qualidade do modelo em causa.

Numa última fase, serão resolvidas as instâncias da Cruz Vermelha Portuguesa e os resultados serão comparados com os obtidos em Loureiro (2010). De notar que todos os resultados que se seguem foram obtidos com uma rotina de pesquisa completa que explora todo o espaço de pesquisa.

8.2.1. Primeiro Teste

Com esta primeira fase de testes pretende-se averiguar se os modelos conseguem gerar boas soluções em tempo "útil". Para tal recorreu-se à instância de dimensão reduzida apresentada na Secção 8.1.1.

Modelo de Grafos

O primeiro modelo, apresentado na Secção 7.2, defraudou as expectativas de gerar soluções num intervalo de tempo aceitável pois o ECLiPSe não foi capaz de o fazer em 120 segundos. Dada a reduzida dimensão do problema, esta incapacidade do modelo em gerar soluções de

forma eficiente aparenta estar relacionada com a complexidade das restrições e da sua fraca propagação, o que conduz a uma redução lacunar do espaço de pesquisa. Tal suspeita levou a que fosse concebido um outro modelo para o DARP no qual se evitou demasiada complexidade na definição de restrições, considerando, inclusive, a relaxação de algumas delas.

Simplificação do Modelo de Grafos

A simplificação do modelo acima referido apresenta-se na Secção 7.3 e foi testada em condições semelhantes ao anterior, não sendo também capaz de proporcionar uma solução ao fim de 120 segundos. Este facto impeliu a necessidade de averiguar quais as razões para tal desempenho que se considera peculiar dada a dimensão da instância de teste.

Revisitando o modelo e a sua simplicidade comparativamente ao modelo que o antecede, considerou-se retirar a restrição expressa pelas equações (7.3.5) e (7.3.6) que indica que um local de destino de um pedido tem que ser servido antes do local de origem. A ausência desta restrição torna esta simplificação numa relaxação do DARP visto que admite soluções que não são válidas no seu contexto. Sem esta restrição o modelo foi capaz de gerar a primeira solução em 0,78 segundos e à medida que se iam obtendo outras soluções foi-se verificando que algumas delas seriam válidas de acordo com a restrição retirada, o que sugere, novamente, uma propagação não tão eficiente como o desejado, mantendo um espaço de pesquisa extenso.

Foi ainda avaliado o desempenho do modelo à medida que ia sendo adicionada, pedido a pedido, a restrição referida acima e expressa pela equação (7.3.5). Este teste revelou que à medida que iam sendo inseridas novas restrições, o tempo necessário para obter uma solução para o problema ia sendo cada vez maior, como se pode verificar através da Tabela 8.1.

Tabela 8.1 - Escalabilidade do predicado *reachables.*

Nº de Restrições	Tempo de Execução (s)
0	0,78
1	0,92
2	1,59
3	2822,06

Sem lançar a restrição que garante que o local de destino de um pedido é visitado antes do respectivo local de origem para nenhum pedido, foi obtida uma solução em 0,78 segundos. Lançando essa mesma restrição para o primeiro pedido de transporte (ver Anexo B) o modelo também foi capaz de gerar uma solução em 0,92 segundos. Caso esta restrição contemplasse

o primeiro, segundo e terceiro pedidos, o tempo necessário para obter uma solução escalava para 2822,06 segundos. Incluindo também o quarto pedido de transporte neste leque, não foi possível obter uma solução em 3600 segundos.

Estes resultados corroboram com as conclusões do artigo de apresentação do *framework* GRASPER (Viegas & Azevedo, 2007) onde é dito que alguns dos predicados disponibilizados ainda precisariam de algum refinamento, em particular os predicados *reachables* e *connected*.

Modelo de Domínios Finitos

O insucesso das modelações que recorrem à biblioteca de restrições GRASPER levou a que fosse concebido um outro modelo, desta feita no contexto dos domínios finitos. Este modelo apresentado na Secção 7.4 teve um comportamento completamente distinto dos restantes, conseguindo obter uma solução para a instância de teste em 0,19 segundos. Os percursos obtidos nesta solução constituem as duas rotas presentes na figura abaixo.

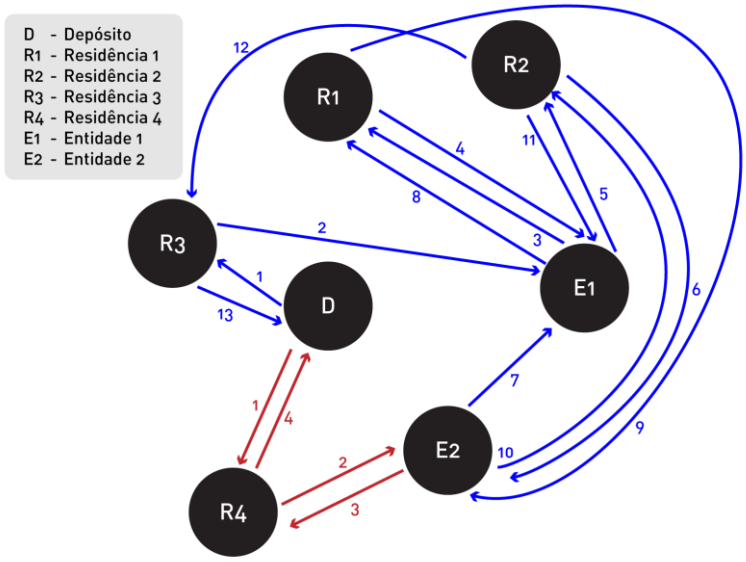

Figura 8.1 - Rotas geradas para instância de teste.

De notar que as rotas acima representadas (Figura 8.1) foram obtidas sem recorrer a qualquer uma das heurísticas apresentadas na Secção 7.4.4, o que indica que existe alguma margem para melhorar o desempenho do modelo na obtenção de soluções. Nesta solução em particular, a escolha de variável foi realizada incrementalmente e os valores atribuídos, também de forma incremental, de acordo com o seu domínio.

Adicionalmente foi utilizada uma estratégia para atribuição dos tempos de espera dos veículos ao longo da rota. Tendo em conta que se pretendem minimizar os tempos de espera

com utentes a bordo do veículo, esta estratégia tenta apenas traduzir este objectivo aumentando os tempos de espera quando a viatura não tem utentes a bordo para que possa minimizá-los quando existem utentes. Assim, foram obtidas as seguintes aproximações temporais para a primeira rota:

Tabela 8.2 - Rota obtida para a primeira viatura.

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Depósito	07h00	01h15	0
Residência 4	08h35	00h00	1
Entidade 2	08h48	01h08	0
Entidade 2	10h00	00h00	1
Residência 4	10h15	00h00	0
Depósito	10h38	00h00	0

Ainda que se trate de uma solução bastante interessante do ponto de vista da qualidade de serviço, visto que os clientes servidos usufruem praticamente de um serviço individualizado, há que realçar que, como pretendido, os tempos de espera serem realizados sem utentes a bordo. De notar ainda que o tempo de espera no início da rota indica que a viatura não necessita de sair imediatamente às 7:00 do depósito, podendo sair apenas às 8:15 sem comprometer os *timings* do serviço.

Analogamente, os valores obtidos para a rota da segunda viatura que compõe a frota desta instância de teste são apresentados na Tabela 8.3.

Tabela 8.3 – Rota obtida para a segunda viatura.

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Depósito	07h00	01h04	0
Residência 3	08h10	00h00	2
Entidade 1	08h26	00h00	0
Residência 1	08h43	00h00	1
Entidade 1	09h00	00h50	0
Residência 2	10h00	00h00	2
Entidade 2	10h10	00h00	0
Entidade 1	10h25	00h00	1
Residência 1	10h40	00h44	0
Entidade 2	11h40	00h00	2
Residência 2	11h51	00h00	0
Entidade 1	12h08	00h00	2
Residência 3	12h19	00h00	0
Depósito	12h30	00h00	0

De forma idêntica à rota anterior, esta pode não ser a melhor do ponto de vista dos custos do serviço visto que se fornece um serviço praticamente individualizado. De qualquer das formas, a introdução de heurísticas para a obtenção de soluções e também o acréscimo de uma função objectivo onde se explicitarão os critérios a considerar para a avaliação das soluções poderão gerar soluções mais interessantes do ponto do visto do custo total do serviço.

Não obstante, a aplicação deste modelo permitiu gerar uma solução na qual todas as necessidades de transporte dos utentes foram cumpridas dentro da ordem prevista. Além disso permitiu também lidar com a existência de pedidos incompatíveis, nomeadamente o caso do quarto pedido de transporte (ver Tabela B.1) que, devido às restrições temporais e de carga, tem que ser servido num veículo diferente do terceiro e quinto pedidos.

8.2.2. Heurísticas

Como referido aquando da introdução desta secção, pretende-se aliar a apreciação das várias heurísticas apresentadas na Secção 7.4.4 para o modelo de domínios finitos com a avaliação do desempenho do mesmo.

Deste modo serão consideradas algumas das instâncias de referência para as quais se pretenderá obter uma solução num tempo limite de 1200 segundos utilizando as diversas combinações heurísticas, tanto para a escolha de variável como para a escolha de valor. Ainda que nesta fase ainda não seja considerada a optimização, a função objectivo presente na equação (7.4.27) será tida em conta para averiguar quais as heurísticas que geram melhores primeiras soluções. Os factores utilizados foram $\alpha = 6$, $\beta = 2$ e $\delta = 2$, valorizando ligeiramente mais a componente referente aos custos do serviço que a relativa à qualidade de serviço.

Na Secção 7.4.4 foram apresentadas as combinações heurísticas *FF-DC*, *FF-TP*, *S-TP* e *S-TM* que serão avaliadas no que diz respeito à obtenção da primeira solução, considerando o tempo necessário, em segundos, o número de *backtracks* e o valor da solução calculado de acordo com a função objectivo e os factores apresentados no parágrafo anterior.

As heurísticas baseadas na escolha de valor *TP* apresentaram resultados consideravelmente inferiores às restantes (*FF-DC* e *S-TM*). A combinação *S-TP*, em particular, excedeu o limite de 1200 segundos para gerar uma solução para uma das instâncias mais simples, necessitando de cerca de 1465 segundos para tal. Devido ao desempenho descrito

estas heurísticas foram desconsideradas no âmbito desta dissertação, sendo que na Tabela 8.4 apenas são apresentados os resultados para as combinações heurísticas *FF-DC* e *S-TM*.

Tabela 8.4 - Resultados obtidos com as heurísticas *FF-DC* e *S-TM*.

Instância	Tempo para Solução (s)		Nº de Retrocessos		Função Objectivo	
	FF-DC	*S-TM*	*FF-DC*	*S-TM*	*FF-DC*	*S-TM*
a2-16	0,39	1,53	0	8	2258	2422
a2-20	-	0,69	-	1	-	2512
a2-24	1,61	7,16	1	4	1107	3096
a3-18	1,5	3,04	1	4	1124	2580
a3-24	-	11,62	-	11	-	3066
a3-30	-	-	-	-	-	-
a3-36	41,75	1100,96	20	1123	4902	4676
a4-16	15,34	5,8	101	14	2388	1914
a4-24	10,37	11,12	20	15	3284	2960
a4-32	45,88	102,77	21	36	4000	4060
a4-40	-	313,67	-	30	-	5160
b2-16	0,78	0,33	3	0	2188	2178
b2-20	1,05	0,64	1	0	2284	2510
b2-24	1,96	1,67	2	2	3128	3180
b3-18	1,26	1,67	1	5	2384	3024
b3-24	4,51	-	4	-	3070	-
b3-30	26,38	14,31	24	5	4190	4650
b3-36	-	-	-	-	-	-
b4-16	5,07	1,08	43	0	2616	2344
b4-24	13,9	13,79	10	8	3004	3428
b4-32	-	37,36	-	3	-	4854
b4-40	-	49,31	-	3	-	6098

Nota: As execuções que excederam os 1200 segundos estão marcadas como "-".

A comparação de resultados entre estas duas heurísticas não é trivial. Ainda que a combinação *S-TM* tenha resolvido mais instâncias que a *FF-DC*, 19 num total de 22, contra apenas 15 da parte da *FF-DC*, nem sempre teve um melhor desempenho no que diz respeito ao tempo necessário para gerar uma solução. Também o número de retrocessos necessários para tal e o valor da solução obtida devem ser tidos em conta, pelo que todos estes dados se sintetizam no gráfico abaixo (Figura 8.2) para as 14 instâncias resolvidas por ambas as combinações heurísticas.

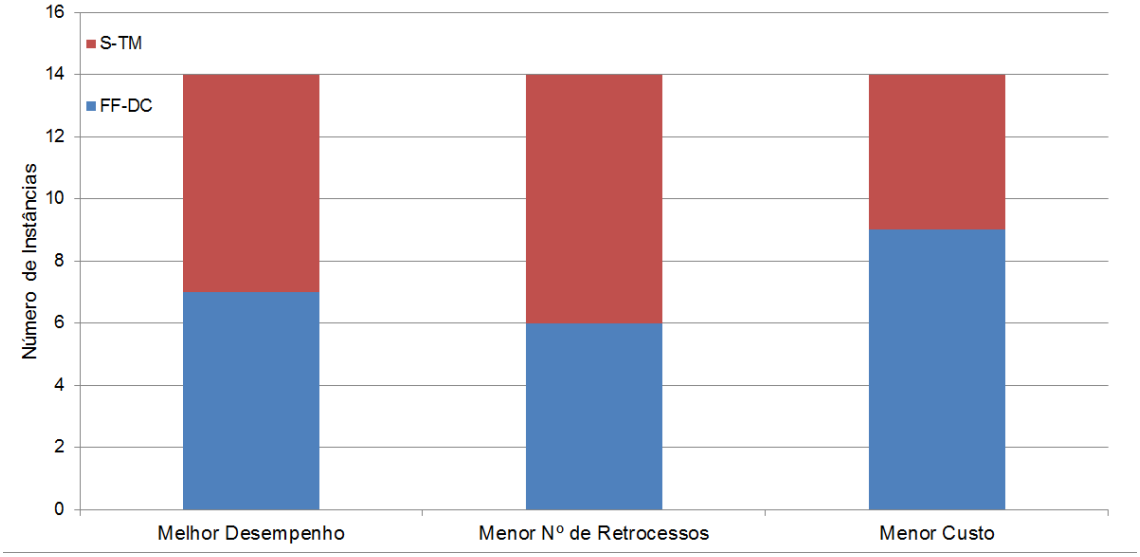

Figura 8.2 - Comparativo dos resultados obtidos pelas heurísticas *FF-DC* e *S-TM*.

Das 14 instâncias em que ambas as heurísticas conseguiram obter uma solução em menos de 1200 segundos, revelou-se um empate no que diz respeito ao desempenho, sendo que em sete ocasiões a combinação *FF-DC* conseguiu gerar uma solução mais rapidamente que a combinação *S-TM*, sendo que o inverso também se verificou. No entanto, no que diz respeito a retrocessos, por 8 ocasiões a heurística *S-TM* realizou menos retrocessos que a sua concorrente *FF-DC*. Por outro lado esta última, em 9 das 14 instâncias, gerou uma primeira solução com custo inferior às geradas pela combinação *S-TM*.

Tendo em conta que os resultados não são muito conclusivos relativamente à superioridade de uma das combinações, na resolução do caso de estudo que se inicia na próxima secção ainda serão consideradas ambas as combinações heurísticas.

Há que mencionar que comparativamente com os resultados reportados em artigos que recorreram a estas mesmas instâncias, nomeadamente Cordeau (2006) e Ropke et al (2007), os apresentados acima demonstram um desempenho inferior, pois o tempo necessário para obter a primeira solução em algumas instâncias, é suficiente para obter a solução óptima nas abordagens de Programação Inteira acima referidas.

8.2.3. Instâncias CVP

A análise conduzida no capítulo anterior levou a que apenas as duas heurísticas que mais se destacaram fossem utilizadas nas instâncias da delegação Amadora – Sintra da Cruz Vermelha Portuguesa.

Os resultados para a obtenção de uma solução são apresentados na Tabela 8.5 para as combinações heurísticas *FF-DC* e *S-TM*, à semelhança da secção anterior.

Tabela 8.5 - Sumário dos resultados obtidos pelas heurísticas *FF-DC* e *S-TM*.

Instância	Tempo de Execução (s)		Nº de Retrocessos		Função Objectivo	
	FF-DC	*S-TM*	*FF-DC*	*S-TM*	*FF-DC*	*S-TM*
1º Dia	0,51	-	10	-	2596	-
2º Dia	0,37	3,04	0	35	2142	2856
3º Dia	0,06	0,08	0	0	1006	994
4º Dia	0,37	3,6	1	44	2352	2442

Nota: As execuções que excederam os 1200 segundos estão marcadas como "-".

Na Figura 8.3, Figura 8.4 e Figura 8.5 sintetizam-se, respectivamente, os tempos de execução, o número de retrocessos realizados e o valor da função objectivo para a solução gerada paras ambas as heurísticas. De notar que a combinação heurística *S-TM* não foi capaz de obter uma solução para o primeiro dia de operação da CVP num limite de 1200 segundos.

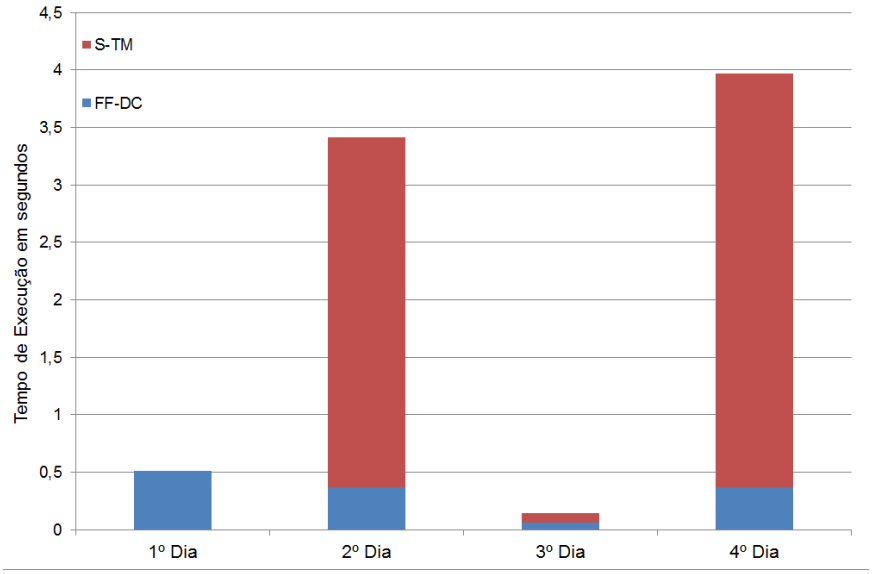

Figura 8.3 - Tempo necessário para obtenção de uma solução.

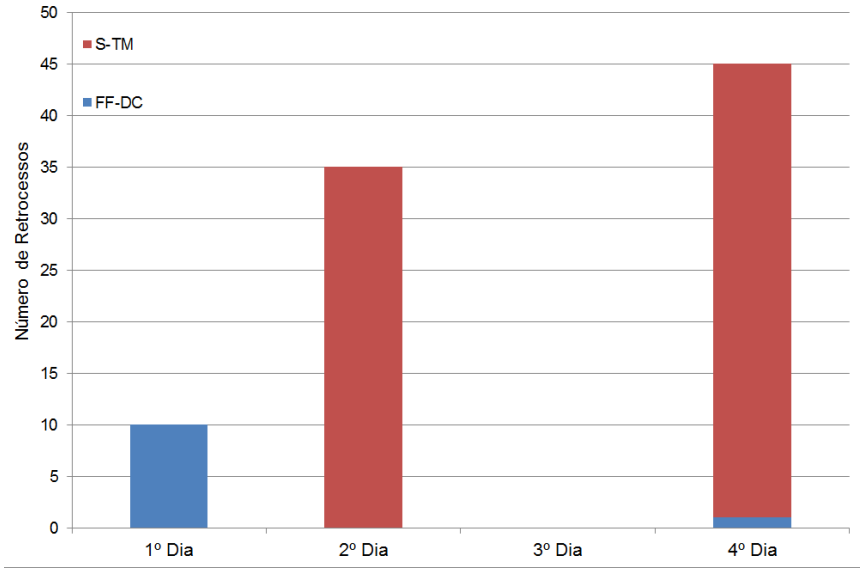

Figura 8.4 - Número de retrocessos realizados.

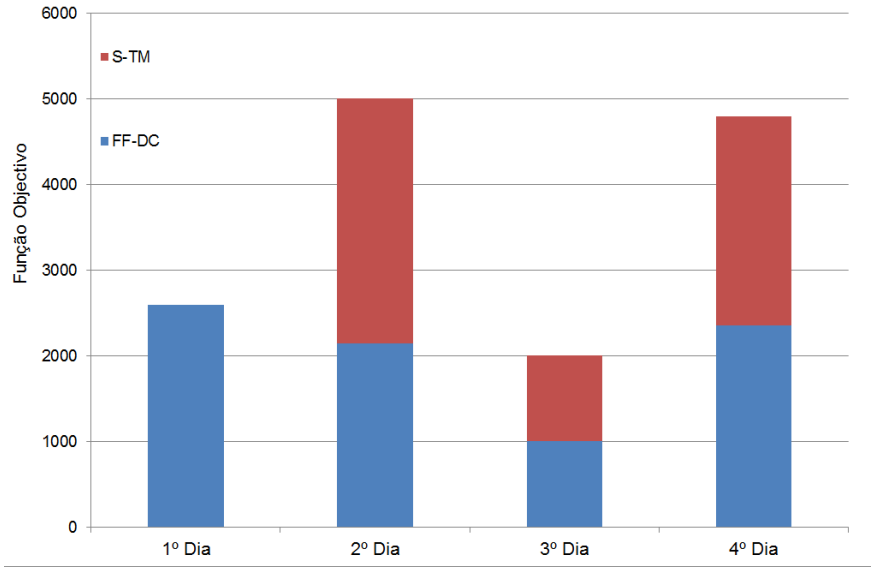

Figura 8.5 - Valor da função objectivo.

Além de não ter conseguido gerar uma solução em menos de 1200 segundos para a instância relativa ao primeiro dia, a heurística *S-TM* demonstrou necessitar de mais tempo que a combinação *FF-DC* para gerar uma solução para qualquer uma das restantes instâncias (Figura 8.3). O número de retrocessos realizados pela heurística *S-TM* foi consideravelmente superior que para a *FF-DC* nas instâncias relativas ao segundo e quarto dias, como se pode verificar através da Figura 8.4. À excepção do terceiro dia, o valor das soluções obtidas pela heurística *FF-DC* foi sempre mais interessante que o seu correspondente na combinação

heurística *S-TM*, algo expresso pela Figura 8.5 tendo em conta o objectivo de minimizar o valor da função objectivo.

Dado este cenário a heurística *FF-DC* foi considerada para a obtenção das soluções óptimas para os quatro dias de serviço da delegação Amadora – Sintra da Cruz Vermelha Portuguesa. Na Tabela 8.6 apresenta-se o valor da solução obtida, o tempo necessário para alcança-la e, por fim, o tempo necessário para provar a optimalidade da solução.

Tabela 8.6 - Resultados obtidos pelo modelo de domínios finitos.

Instância	Função Objectivo	Obtenção da Solução	Provar Optimalidade
1º Dia	2028	6h23	-
2º Dia	1734	2h44	-
3º Dia	758	7h42	38h06
4º Dia	1954	6h52	-

Nota: As provas que excederam as 72 horas estão marcadas como "-".

De notar que no período de 72 horas, apenas foi possível obter e provar a solução óptima para a instância relativa ao terceiro dia de trabalho da CVP. As soluções obtidas para os restantes dias foram as melhores no período de tempo referido, no entanto a sua optimalidade não foi provada. As descrições das rotas obtidas podem ser consultadas no Anexo A.

Comparativamente aos resultados obtidos por Loureiro (2010), os presentes neste documento são consideravelmente inferiores no que diz respeito ao desempenho pois a abordagem de Programação Inteira alcança a solução óptima antes que a abordagem presente nesta dissertação alcance uma única solução, como se pode comprovar através da Tabela 8.7 que reproduz os resultados presentes em Loureiro (2010).

Tabela 8.7 - Resultados obtidos por abordagem de Programação Inteira (Loureiro, 2010).

Instância	Tempo de Execução (s)	Função Objectivo (€)
1º Dia	0,33	9,66
2º Dia	0,31	10,05
3º Dia	0,16	5,12
4º Dia	0,27	9,9

No entanto há que salvaguardar o facto de as soluções obtidas serem diferentes das presentes em Loureiro (2010), em particular no caso do terceiro dia cuja solução óptima foi provada no âmbito desta dissertação. Esta diferença deve-se essencialmente ao facto de terem sido contemplados factores relativos à qualidade de serviço no modelo de domínios finitos proposto neste documento.

A título de exemplo, e considerando as correspondências presentes na Tabela A.14, a solução óptima obtida no contexto desta dissertação pode ser representada da seguinte forma:

$$Rota = [1,3,2,13,13,5,13,4,13,13,13,6,8,7,14,10,14,13,12,9,13,11,13,1]$$

Este formato indica que o veículo sai do depósito 1 e irá recolher um utente no vértice 3 e outro no vértice 2 que serão levados ao vértice 13, referente à Movi Física, uma entidade prestadora de cuidados de saúde situada na Amadora. Posteriormente irá ao vértice 5 buscar mais um utente e voltará à Movi Física para deixá-lo. Uma situação idêntica acontece com o utente associado ao vértice 4, que tem um tratamento marcado para a mesma entidade, simbolizada pelo vértice 13. Na sequência, recolherá dois utentes que já terminaram o tratamento na Movi Física e irá transportá-los para os seus locais de residência, dados pelos vértices 6 e 8. Pretende-se apenas com esta breve descrição explicar a codificação utilizada, sendo todos os dados da rota podem ser consultados na Tabela A.16.

Por outro lado, no trabalho de Loureiro (2010) a rota óptima obtida para este terceiro dia de trabalho é dada por

$$Rota = [1,3,2,4,5,13,13,13,13,7,10,9,14,14,13,13,13,6,8,13,13,11,12,1]$$

onde se torna clara a minimização dos custos associados ao serviço logo pelo facto de se recolherem os quatro primeiros utentes antes de qualquer entrega na Movi Física. Este facto leva a que os utentes referentes aos vértices 4 e 5 fiquem mais tempo à espera do início do tratamento. Uma descrição mais pormenorizada pode ser encontrada em Loureiro (2010).

Deste modo, e como seria esperado, calculando a distância temporal de ambas as rotas, verifica-se que a solução apresentada por Loureiro (2010) apresenta um valor de 90 minutos, inferior em 7 minutos comparativamente com a solução obtida no contexto desta dissertação. Este cálculo é feito de acordo com a função *tempo* formalizada em (7.4.25) e com a matriz temporal expressa pela Tabela A.15.

Por uma questão de rigor, há que referir que as soluções obtidas por Loureiro (2010) não são validadas pelo modelo proposto na Secção 7.4, mesmo relaxando as restrições relativas à qualidade de serviço, o que sugere que o modelo concebido por Loureiro (2010) pode não estar totalmente correcto.

8.3. Conclusões

De acordo com os resultados computacionais obtidos e apresentados ao longo deste capítulo é possível concluir que o modelo de domínios finitos tem um desempenho francamente pior que as abordagens que recorrem à Programação Inteira pois estas aparentam conseguir uma redução mais eficiente do espaço de pesquisa, o que inevitavelmente acaba por pesar no que diz respeito ao tempo necessário para obter soluções, em particular, para obter uma solução óptima e provar essa optimalidade.

No que diz respeito às soluções para as instâncias da Cruz Vermelha Portuguesa, os resultados obtidos, como já foi referido, demonstram um desempenho pior relativamente ao trabalho de Loureiro (2010). Ainda assim, e reiterando um pouco o que foi concluído na Secção 8.2.3, há que referir que as soluções óptimas de Loureiro (2010) não válidas no modelo proposto neste documento, não só por questões relativas à qualidade de serviço, mas também por eventuais más formações da modelação de Programação Inteira.

A comparação dos resultados obtidos com os presentes em Berbeglia et al. (2010) não é feita directamente pois, ainda que algumas das instâncias de teste sejam comuns às usadas neste capítulo, no trabalho de Berbeglia et al. (2010) são usadas relaxações do *Dial-a-Ride Problem* onde algumas restrições são descartadas, nomeadamente no que diz respeito à carga do veículo e aos *timings* do serviço. Esta opção torna-se mais clara se for tida em conta a actualização do seu trabalho (Berbeglia et al., 2011) que integra a solução concebida através de Programação com Restrições para obter uma solução para uma relaxação do problema que é posteriormente melhorada com recurso a pesquisa tabu. Ainda assim os resultados relatados por Berbeglia et al. (2010) superam, em termos de desempenho, os obtidos nesta dissertação. Adicionalmente foram relatados resultados para instâncias de maior dimensão, algo limitado neste trabalho pelas dificuldades computacionais.

Não obstante os resultados obtidos com o modelo descrito na Secção 7.4, seria interessante averiguar a influência da ferramenta no tempo necessário para a obtenção de soluções. Na impossibilidade de averiguar o desempenho do modelo de domínios finitos em diferentes ferramentas, foram pesquisados estudos comparativos do desempenho da ferramenta ECLiPSe com outras disponíveis no mercado, com o objectivo de perceber se por se tratar de uma ferramenta gratuita, com menos investimento que soluções profissionais, poderá ser penalizador a nível de desempenho e qual a ordem de grandeza desse diferencial. De acordo com Dovier et. al (2005) o ECLiPSe tem um desempenho inferior ao Ilog Solver da

IBM, ainda que não seja apresentada explicitamente uma ordem de grandeza para o diferencial de desempenho. Aminu e Eglese (2006) também o afirmam no contexto de um *Vehicle Routing Problem* mas, novamente, sem que seja estabelecida uma ordem de grandeza no que diz respeito à performance das ferramentas. Em rigor esta temática já teria sido explorada por Fernandez e Hill (1998, 2000) que compararam o desempenho de oito sistemas de restrições. Dos resultados apresentados por estes autores conclui-se que o desempenho do ECLiPSe é inferior a praticamente todas as ferramentas estudadas, em particular, 2 a 3 vezes inferior ao SICStus. Segundo qualquer um dos estudos referidos, o Ilog Solver é também apontado como o sistema mais rápido e com uma robustez acima da média.

9

9. Conclusões

Neste capítulo são apresentadas as ilações retiradas ao longo da concepção desta dissertação, além de se deixarem algumas directivas no que diz respeito a trabalho futuro, indicando algumas direcções que poderão ser interessantes seguir.

9.1. Conclusões

A Programação com Restrições tem sido apresentada como um poderoso paradigma para a resolução de problemas combinatórios (Little & Tsang, 1995; Rossi et al., 2006). Apesar de se enquadrar como uma abordagem exacta, devido à sua exaustividade, permite também explorar outras características que tipicamente se encontram associadas às abordagens aproximadas. Ainda que a sua utilização já tenha sido difundida por diversas áreas, inclusive, no contexto dos *Vehicle Routing Problems*, são escassas as modelações e quaisquer outras referências para o *Dial-a-Ride Problem* segundo este paradigma.

Assim, ao principal objectivo desta dissertação, que consistia em tentar oferecer uma melhoria no processo de definição de rotas da delegação Amadora – Sintra da Cruz Vermelha Portuguesa, foi adicionado um outro com o intuito de aplicar a Programação com Restrições ao problema enfrentado por esta instituição. Para tal, e tendo em conta que este tipo de problemas são tipicamente apresentados através de grafos, decidiu-se conceber um modelo recorrendo ao *framework* GRASPER que disponibiliza restrições no contexto da teoria de grafos, como referido na Secção 6.3.2. Do ponto de vista conceptual, esta modelação apresenta características que a tornam interessante uma vez que permite representar diversos conceitos ligados aos VRP's (e.g. vértices, arcos, caminhos, predecessores, sucessores). No entanto, a

sua aplicação numa pequena instância de teste (Secção 8.2.1) revelou resultados bastante fracos em termos de tempo computacional para a geração de uma solução.

Devido aos resultados obtidos, mas com a motivação da facilidade de modelação, decidiu-se conceber um outro modelo baseado também nos conceitos de teoria de grafos, mas relaxando algumas das especificações modeladas interiormente. Nomeadamente, optou-se por efectuar uma redução no número de variáveis envolvidas, em particular nos diversos subgrafos, e transformar algumas das restrições de modo a reduzir a complexidade do modelo. Ainda assim, os resultados não foram satisfatórios pois, à semelhança da primeira modelação, não foi possível gerar soluções para uma instância de teste com dimensão reduzida. Estes resultados conduziram à realização de mais alguns testes descritos na Secção 8.2.1, de onde se concluiu que um dos predicados fundamentais para ambas as modelações, o predicado *reachables*, não tem um desempenho adequado no que diz respeito à propagação e consequente redução do espaço de pesquisa. Apesar dos resultados obtidos, a utilização dos domínios de grafos constitui uma novidade no contexto do *Dial-a-Ride Problem* e não deixa de ser promissora pelas questões já referidas de facilidade de representação de diversos conceitos inerentes à problemática da definição de rotas.

Dado este cenário e os objectivos propostos, partiu-se para outra modelação, desta feita recorrendo aos domínios finitos, em particular aos inteiros. Com base em diversas directrizes da Programação Linear Inteira, dado o seu sucesso no contexto do DARP, e no trabalho desenvolvido por Berbeglia et al. (2010), foi concebido um modelo que se baseia no conceito de sucessor para representar as rotas, sendo estas representadas pela sequência de sucessores indicam quais os locais a serem visitados de acordo com as características do serviço. Ao contrário das modelações anteriores, esta foi capaz de gerar soluções em tempo aceitável em particular para instâncias de dimensão não muito elevada (Secção 8.2). No entanto, o seu desempenho e inferior quando comparado com as abordagens de Programação Linear Inteira (Cordeau, 2006; Loureiro, 2010).

Não obstante o seu desempenho, há que realçar o facto da abordagem apresentada neste trabalho ser uma das pouquíssimas modelações do DARP recorrendo à Programação com Restrições e a única, do conhecimento do autor deste documento, que contempla a questão da qualidade de serviço com mais detalhe, considerando o tempo de espera dos utentes, bem como o tempo que estes passam a bordo dos veículos, minimizando ambos além dos custos monetários associados a um serviço. Este facto levou a que as soluções óptimas obtidas para as instâncias da CVP não pudessem ser directamente comparadas com as publicadas por

Loureiro (2010), onde o objectivo passava apenas pela minimização dos custos monetários suportados pela Cruz Vermelha Portuguesa.

Destaque-se também o trabalho efectuado na concepção e comparação de diversas heurísticas para guiar o resolvedor no espaço de pesquisa com o objectivo de obter melhores primeiras soluções, assim como na implementação de diferentes abordagens para o problema de transporte porta à porta.

Apesar de todo o trabalho apresentado, não se esgotaram as possibilidades de abordar e melhorar o trabalho realizado nesta dissertação, algumas das quais serão apresentadas na secção seguinte como trabalho futuro.

9.2. Trabalho Futuro

Um trabalho desta natureza nunca se dá por completo, existindo sempre novas ideias a explorar que se crê trazerem melhorias ao que já está concebido. Nesta dissertação em particular, o desempenho das modelações é algo que motiva a avaliação de novas possibilidades que permitam confirmar, ou não, o potencial da Programação com Restrições no âmbito do *Dial-a-Ride Problem*.

Redução do Espaço de Pesquisa

Certamente que o fraco desempenho das modelações apresentadas de deve a uma ténue redução do espaço de pesquisa, relativamente à alcançada pelas abordagens de Programação Linear Inteira. Deste modo, será interessante estudar a adaptação de algumas das técnicas da Programação Linear Inteira ao contexto da Programação com Restrições, nomeadamente as desigualdades válidas que são um tema de bastante interesse na área pois através da adição de restrições aparentemente redundantes, conseguem alcançar uma maior redução. Ainda que não tenham sido formalmente apresentadas como tal, algumas desigualdades foram já contempladas nesta dissertação estando presentes na Secção 7.5. No entanto uma das adaptações que poderia ter mais interesse diz respeito às sub-rotas, isto é, rotas que não visitam todos os locais envolvidos no problema. À semelhança do conceito de pedidos incompatíveis abordado na Secção 7.5, poderiam adaptar-se as desigualdades das abordagens de Programação Linear Inteira de modo a contemplar sub-rotas incompatíveis, ou seja, sub-rotas que não podem ser servidas pelo mesmo veículo.

Outro tipo de redução que não foi endereçada nas modelações propostas e que pode revelar-se interessante tem que ver com o corte de simetrias ao nível dos locais virtuais que representam o mesmo local físico. Como foi referido no Capítulo 7, optou-se por caracterizar os locais envolvidos nos pedidos de transporte por vértices distintos, devido a questões relativas aos tempos a que estes são servidos. A título de exemplo, se existirem dois pedidos para utentes distintos, ambos a residir em *Lar* e a necessitar de transporte para *Entidade* à mesma hora, estes irão gerar locais virtuais (vértices) diferentes. Caso os vértices associados ao primeiro pedido sejam *A* e *B* e ao segundo *C* e *D*, respectivamente, as combinações *A-B-C-D*, *A-B-D-C*, *B-A-C-D* e *B-A-D-C* são equivalentes. Este tipo de simetrias poderá ser removido, por exemplo, restringindo os vértices com uma codificação inferior a serem servidos primeiro, o que na situação descrita validaria apenas a primeira combinação.

Este tipo de técnicas aliadas a algum pré-processamento no contexto das soluções parciais, vulgo as sub-rotas referidas acima, poderia contribuir de forma interessante para a redução do espaço de pesquisa e, consequentemente, para uma melhoria de desempenho computacional. Todas estas opções deverão ser avaliadas de modo a perceber se a redução alcançada justifica o tempo e o poder computacional gasto, em particular na questão da geração de várias soluções parciais e da sua concatenação com outras de modo a descobrir sub-rotas incompatíveis.

No âmbito da redução do espaço de pesquisa poderá ainda ser interessante estudar a possibilidade de evitar repetir vértices virtuais para caracterizar locais físicos iguais. Certamente que uma redução no número de vértices reduziria consideravelmente o espaço de pesquisa, no entanto trata-se de uma questão delicada devido ao facto de os vértices que mapeiam o DARP terem informação associada aos utentes que são servidos, bem como aos *timings* do próprio serviço.

Heterogeneidade das Frotas

Devido aos resultados obtidos nos testes efectuados para uma frota homogénea, a questão da heterogeneidade das frotas não foi aprofundada como se pretendia inicialmente. Ainda assim a modelação de domínios finitos proposta nesta dissertação foi desenvolvida com o objectivo de ser o mais genérica possível, o que permite incorporar, como inicialmente foi pensado, a capacidade de lidar com utentes com necessidades especiais, e.g., transporte em cadeira de rodas ou em maca. Dada a pouca investigação nesta área, poderia ser interessante a cria-

ção/adaptação de instâncias à realidade de transporte enfrentada pela Cruz Vermelha Portuguesa, à semelhança do trabalho realizado por Parragh (2011).

Heurísticas

Como foi referido ao longo do Capítulo 6, as heurísticas constituem um mecanismo fundamental para alcançar soluções de forma mais eficiente pelo que será uma temática interessante de ser explorada com mais profundidade.

Ainda que não tenha demonstrado um muito mau desempenho, a heurística apresentada na Secção 7.4.4 que pretende simular o raciocínio humano definindo as rotas gradualmente através da escolha dos pedidos que têm que ser servidos mais cedo poderia beneficiar caso contemplasse também a distância aos restantes locais. O tipo de raciocínio desta heurística exige algum poder computacional devido à repetida intersecção de domínios das variáveis envolvidas. Por outro lado poderia optar-se por uma atribuição menos inteligente de valores que pouparia os cálculos inerentes às intersecções em troca de *backtracks,* o que também não parece sensato. Assim sendo, a existência de uma restrição e respectivos propagadores concebidos de acordo com estas necessidades poderiam trazer uma melhoria em termos de desempenho.

Cruz Vermelha Portuguesa

Ainda que o facto de contribuir para a realidade da Cruz Vermelha Portuguesa tenha sido a principal motivação desta dissertação, devido à nobreza da sua missão, à sua vertente social e ao serviço que prestam a toda uma comunidade, a verdade é que a urgência da delegação Amadora – Sintra por um sistema de gestão de rotas foi atenuador devido a algumas alterações que se esperam afectar o funcionamento do serviço. No entanto, caso as sugestões presentes nesta secção não conseguissem dar resposta a instâncias do *Dial-a-Ride Problem* como as enfrentadas pela CVP, a Pesquisa Local (com base nos resultados que têm vindo a obter e que se referem no Capítulo 4) seria uma alternativa a considerar quando a delegação necessitar realmente de implementar um sistema.

Bibliografia

Aminu, U. F., & Eglese, R. W. (2006). A constraint programming approach to the Chinese postman problem with time windows. *Computers & Operations Research*, *33*(12), 3423–3431. doi:10.1016/j.cor.2005.02.012

Azevedo, F. (2003). *Constraint Solving over Multi-Valued Logics*. *Knowledge and Information Systems* (p. 224). IOS Press.

Azevedo, F. (2007). Cardinal: A Finite Sets Constraint Solver. *Constraints*, *12*(1), 93–129. doi:10.1007/s10601-006-9012-6

Baldacci, R., Battarra, M., & Vigo, D. (2009). Valid inequalities for the fleet size and mix vehicle routing problem with fixed costs. *Networks*, *54*(4), 178–189. doi:10.1002/net.20331

Baugh, J. W., Kakivaya, G. K. R., & Stone, J. R. (1998). INTRACTABILITY OF THE DIAL-A-RIDE PROBLEM AND A MULTIOBJECTIVE SOLUTION USING SIMULATED ANNEALING. *Engineering Optimization*, *30*(2), 91–123. doi:10.1080/03052159808941240

Beaudry, A., Laporte, G., Melo, T., & Nickel, S. (2008). Dynamic transportation of patients in hospitals. *OR Spectrum*, *32*(1), 77–107. doi:10.1007/s00291-008-0135-6

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: a classification scheme and survey. *Top*, *15*(1), 1–31. doi:10.1007/s11750-007-0009-0

Berbeglia, G., Cordeau, J.-F., & Laporte, G. (2010). Dynamic pickup and delivery problems. *European Journal of Operational Research*, *202*(1), 8–15. doi:10.1016/j.ejor.2009.04.024

Berbeglia, G., Cordeau, J.-F., & Laporte, G. (2011). A Hybrid Tabu Search and Constraint Programming Algorithm for the Dynamic Dial-a-Ride Problem. *INFORMS Journal on Computing*, 0–27. doi:10.1287/ijoc.1110.0454

Berbeglia, G., Pesant, G., & Rousseau, L.-M. (2010). Checking the Feasibility of Dial-a-Ride Instances Using Constraint Programming. *Transportation Science*, *45*(3), 399–412. doi:10.1287/trsc.1100.0336

Biggs, N. L., Lloyd, E. K., & Wilson, R. J. (1986). *Graph Theory 1736-1936* (New Ed edi., p. 252). Clarendon Press.

Bodin, L. D., & Sexton, T. R. (1986). The multi-vehicle subscriber dial-a-ride problem. *TIMS Studies in Management Science*, *26*, 73–86.

Borndörfer, R., Grötschel, M., Klostermeier, F., & Küttner, C. (1997). Telebus Berlin: Vehicle Scheduling in a Dial-a-Ride System. *Informationstechnik*, (May), 391–422.

Calvo, R. W. (2000). A New Heuristic for the Traveling Salesman Problem with Time Windows. *Transportation Science*, *34*(1), 113–124. doi:10.1287/trsc.34.1.113.12284

Calvo, R. W., & Colorni, A. (2006). An effective and fast heuristic for the Dial-a-Ride problem. *4or*, *5*(1), 61–73. doi:10.1007/s10288-006-0018-0

Charnes, A., Cooper, W. W., & Farr, D. (1953). Linear Programming and Profit Preference Scheduling for a Manufacturing Firm. *Operations Research*, *1*(3), 114–129. doi:10.1287/opre.1.3.114

Clímaco, J. N., Antunes, C. H., & Gomes Alves, M. J. (2003). *Programação Linear Multiobjectivo* (p. 386). Imprensa da Universidade de Coimbra.

Cordeau, J., & Laporte, G. (2003). A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, *37*(6), 579–594. doi:10.1016/S0191-2615(02)00045-0

Cordeau, Jean-Francois. (2006). A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research*, *54*(3), 573–586. doi:10.1287/opre.1060.0283

Cordeau, Jean-François, & Laporte, G. (2003). The Dial-a-Ride Problem (DARP): Variants, modeling issues and algorithms. *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, *1*(2), 89–101. doi:10.1007/s10288-002-0009-8

Cordeau, Jean-François, & Laporte, G. (2007). The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, *153*(1), 29–46. doi:10.1007/s10479-007-0170-8

Cubillos, C., Urra, E., & Rodríguez, N. (2009). Application of Genetic Algorithms for the DARPTW Problem. *International Journal of Computers, Communication and Control*, *IV*(2), 127–136.

Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, *6*(1), 80–91. doi:10.1287/mnsc.6.1.80

Dechter, R. (2003). *Constraint Processing*. *Artificial Intelligence* (Vol. 169). Morgan Kaufmann. Retrieved from http://www.sciencedirect.com/science/book/9781558608900

Desrosiers, J., Dumas, Y., & Soumis, F. (1986). A dynamic programming solution of the large-scale single vehicle dial-a-ride problem with time windows. *American Journal of Mathematical and Management Sciences*, *6*, 301–325.

Desrosiers, J., Dumas, Y., Soumis, F., Taillefer, S., & Villeneuve, D. (1991). *An algorithm for mini-clustering in handicapped transport*. Montreal.

Diana, M., & Dessouky, M. M. (2004). A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Transportation Research Part B: Methodological*, *38*(6), 539–557. doi:10.1016/j.trb.2003.07.001

Dovier, A., Formisano, A., & Pontelli, E. (2005). A Comparison of CLP (FD) and ASP Solutions to NP-Complete Problems, 67–82.

Dumas, Y., Desrosiers, J., & Soumis, F. (1989). *Large scale multi-vehicle dial-a-ride problems*. Montreal.

D'Souza, C., Omkar, S. N., & Senthilnath, J. (2011). Pickup and delivery problem using metaheuristics techniques. *Expert Systems with Applications*, *39*(1), 328–334. doi:10.1016/j.eswa.2011.07.022

Ehrgott, M., & Gandibleux, X. (2002). *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*. (M. Ehrgott & X. Gandibleux, Eds.)*Supply Chain Management An International Journal* (Vol. 52). Kluwer Academic Publishers. doi:10.1007/b101915

Fernandez, A. J., & Hill, P. M. (1998). An Impartial Efficiency Comparison of FD Constraint Systems. *Principles and Practice of Constraint Programming - CP98*, *1520*, 468.

Fernandez, A. J., & Hill, P. M. (2000). A Comparative Study of Eight Constraint Programming Languages Over the Boolean and Finite Domains, *301*, 275–301.

Focacci, F., Lodi, A., & Milano, M. (2002). Mathematical Programming Techniques in Constraint Programming: A Short Overview. *Journal of Heuristics*, *8*(1998), 7–17. doi:10.1023/A:1013653332557

Fraser, A. (1970). *Computer Models in Genetics*. (D. Burnell, Ed.) (p. 192). New York: McGraw-Hill.

Gervet, C. (1994). Conjunto: Constraint Logic Programming with Finite Set Domains. *Logic Programming - Proceedings of the 1994 International Symposium*, 339–358.

Glover, F., & McMillan, C. (1986). The general employee scheduling problem. An integration of MS and AI. *Computers & Operations Research*, *13*(5), 563–573. doi:10.1016/0305-0548(86)90050-X

Gratzer, G. (1971). *Lattice Theory; First Concepts and Distributive Lattices* (1st ed., p. 227). W H Freeman & Co (Sd).

Haralick, R. M., & Elliott, G. L. (1980). Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, *14*(3), 263–313. doi:10.1016/0004-3702(80)90051-X

Hillier, F. S., & Lieberman, G. J. (2001). *Introduction to Operations Research*. *Foundations* (Vol. 45, p. 1237). McGraw-Hill. doi:10.1016/j.jacc.2004.11.012

Hosny, M. I., & Mumford, C. L. (2008). The single vehicle pickup and delivery problem with time windows: intelligent operators for heuristic and metaheuristic algorithms. *Journal of Heuristics*, *16*(3), 417–439. doi:10.1007/s10732-008-9083-1

Hunsaker, B., & Savelsbergh, M. (2002). Efficient feasibility testing for dial-a-ride problems. *Operations Research Letters*, *30*(3), 169–173. doi:10.1016/S0167-6377(02)00120-7

Ioachim, I., Desrosiers, J., Dumas, Y., Solomon, M. M., & Villeneuve, D. (1995). A Request Clustering Algorithm for Door-to-Door Handicapped Transportation. *Transportation Science*, *29*(1), 63–78. doi:10.1287/trsc.29.1.63

Jaw, J.-J., Odoni, A. R., Psaraftis, H. N., & Wilson, N. H. M. (1986). A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, *20*(3), 243–257. doi:10.1016/0191-2615(86)90020-2

Jih, W., & Hsu, J. Y. (2004). A Family Competition Genetic Algorithm for the Pickup and Delivery Problems with Time Window. *Bulletin of the College of Engineering*, *90*, 121–130.

Jorgensen, R. M., Larsen, J., & Bergvinsdottir, K. B. (2006). Solving the Dial-a-Ride problem using genetic algorithms. *Journal of the Operational Research Society*, *58*(10), 1321–1331. doi:10.1057/palgrave.jors.2602287

Kantorovich, L. V. (1940). A new method of solving some classes of extremal problems. *Doklady Akad Sci USSR*, *28*, 211–214.

Knuth, D. E. (1998). *Art of Computer Programming, Volume 3: Sorting and Searching* (p. 800). Addison-Wesley Professional.

Lacomme, P., Quilliot, A., & Zhao, X. (2009). Dial-a-ride: A tentative resolution using a multi-objective resolution scheme based on ELS. *2009 International Conference on Computers & Industrial Engineering* (pp. 1100–1105). IEEE. doi:10.1109/ICCIE.2009.5223774

Land, A. H., & Doig, A. G. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, *28*(3), 497. doi:10.2307/1910129

Landrieu, A., Mati, Y., & Binder, Z. (2001). A tabu search heuristic for the single vehicle pickup and delivery problem wi ... *Journal of Intelligent Manufacturing*, *12*, 5–6, 497–508. doi:10.1023/A:1012204504849

Laporte, G., & Osman, I. H. (1995). Routing problems: A bibliography. *Annals of Operations Research*, *61*(1), 227–262. doi:10.1007/BF02098290

Lauriere, J.-L. (1978). A language and a program for stating and solving combinatorial problems. *Artificial Intelligence*, *10*(1), 29–127. doi:10.1016/0004-3702(78)90029-2

Liberatore, P. (2000). On the complexity of choosing the branching literal in DPLL. *Artificial Intelligence*, *116*(1-2), 315–326. doi:10.1016/S0004-3702(99)00097-1

Little, J., & Tsang, E. (1995). Foundations of Constraint Satisfaction. *The Journal of the Operational Research Society*, *46*(5), 666. doi:10.2307/2584541

Loureiro, M. d'Arbués M. R. (2010). *Optimização de Rotas de Transporte de Doentes Programados : O Caso da Cruz Vermelha Portuguesa Amadora – Sintra*. *Dados*. Instituto Superior Técnico, Universidade Técnica de Lisboa.

Madachy, J. F. (1979). *Madachy's Mathematical Recreations* (p. 251). New York: Dover Publications.

Madsen, O. B. G., Ravn, H. F., & Rygaard, J. M. (1995). A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of Operations Research*, *60*(1), 193–208. doi:10.1007/BF02031946

Malek, M., Guruswamy, M., Pandya, M., & Owens, H. (1989). Serial and parallel simulated annealing and tabu search algorithms for the traveling salesman problem. *Annals of Operations Research*, *21*(1), 59–84. doi:10.1007/BF02022093

Marriott, K., & Stuckey, P. J. (1998). *Programming with Constraints: An Introduction*. *Communication*. MIT Press. Retrieved from http://www.mendeley.com/research/genetic-programming-an-introduction/

Matthias Ehrgott. (2005). *Multicriteria Optimization*. *Fuzzy Sets and Systems* (Vol. 134). Berlin/Heidelberg: Springer-Verlag. doi:10.1007/3-540-27659-9

Melachrinoudis, E., Ilhan, A. B., & Min, H. (2007). A dial-a-ride problem for client transportation in a health-care organization. *Computers & Operations Research*, *34*(3), 742–759. doi:10.1016/j.cor.2005.03.024

Paquette, J., Cordeau, J.-F., & Laporte, G. (2009). Quality of service in dial-a-ride operations. *Computers & Industrial Engineering*, *56*(4), 1721–1734. doi:10.1016/j.cie.2008.07.005

Parragh, S. N. (2011). Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. *Transportation Research Part C: Emerging Technologies*, *19*(5), 912–930. doi:10.1016/j.trc.2010.06.002

Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2010). Variable neighborhood search for the dial-a-ride problem. *Computers & Operations Research*, *37*(6), 1129–1138. doi:10.1016/j.cor.2009.10.003

Parragh, S. N., Doerner, K. F., Hartl, R. F., & Gandibleux, X. (2009). A heuristic two-phase solution approach for the multi-objective dial-a-ride problem. *Networks*, *54*(4), 227–242. doi:10.1002/net.20335

Pearl, J. (1984). *Heuristics: Intelligent Search Strategies for Computer Problem Solving* (p. 399). Addison-Wesley Pub (Sd).

Pereira, F. B., Tavares, J., Machado, P., & Costa, E. (2002). GVR : A New Genetic Representation for the Vehicle Routing Problem, 95–102.

Pesant, G., Gendreau, M., Potvin, J.-Y., & Rousseau, J.-M. (1998). An Exact Constraint Logic Programming Algorithm for the Traveling Salesman Problem with Time Windows. *Transportation Science*, *32*(1), 12–29. doi:10.1287/trsc.32.1.12

Pesant, Gilles, Gendreau, M., Potvin, J.-Y., & Rousseau, J.-M. (1999). On the flexibility of constraint programming models: From single to multiple time windows for the traveling salesman problem. *European Journal of Operational Research*, *117*(2), 253–263. doi:10.1016/S0377-2217(98)00248-3

Podelski, A. (1995). *Constraint Programming: Basics and Trends*. (A. Podelski, Ed.) (Vol. 910, p. 910). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/3-540-59155-9

Polacek, M., Hartl, R. F., Doerner, K., & Reimann, M. (2004). A Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows. *Journal of Heuristics*, *10*(6), 613–627. doi:10.1007/s10732-005-5432-5

Psaraftis, H. N. (1980). A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science*, *14*(2), 130–154. doi:10.1287/trsc.14.2.130

Psaraftis, H. N. (1983). An Exact Algorithm for the Single Vehicle Many-to-Many Dial-A-Ride Problem with Time Windows. *Transportation Science*, *17*(3), 351–357. doi:10.1287/trsc.17.3.351

Rekiek, B., Delchambre, A., & Saleh, H. (2006). Handicapped Person Transportation: An application of the Grouping Genetic Algorithm. *Engineering Applications of Artificial Intelligence*, *19*(5), 511–520. doi:10.1016/j.engappai.2005.12.013

Ropke, S. (2005). *Heuristic and exact algorithms for vehicle routing problems*.

Ropke, S., Cordeau, J.-F., & Laporte, G. (2007). Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, *49*(4), 258–272. doi:10.1002/net.20177

Rossi, F., Van Beek, P., & Walsh, T. (2006). *Handbook of Constraint Programming*. (Francisco Rossi, P. Van Beek, & T. Walsh, Eds.)*Change* (p. 978). Elsevier. Retrieved from http://books.google.com/books?hl=en&lr=&id=Kjap9ZWcKOoC&oi=fnd&pg=PR5&dq=Handbook+of+Constraint+Programming&ots=QBviFU4UOj&sig=apDF7U-5Fr2Hz_EiM1jyJN4pI_M

Rousseau, L., Gendreau, M., Pesant, G., & Focacci, F. (2004). Solving VRPTWs with Constraint Programming Based Column Generation. *Annals of Operations Research*, *130*(1-4), 199–216. doi:10.1023/B:ANOR.0000032576.73681.29

Ruland, K., & Rodin, E. (1997). The Pickup and Delivery Problem: Faces and Branch-and-Cut Algorithm. *Computers & Mathematics with Applications*, *33*(12), 1–13. doi:10.1016/S0898-1221(97)00090-4

Savelsbergh, M. W. P. (1992). The Vehicle Routing Problem with Time Windows: Minimizing Route Duration. *INFORMS Journal on Computing*, *4*(2), 146–154. doi:10.1287/ijoc.4.2.146

Savelsbergh, M. W. P., & Sol, M. (1995). The General Pickup and Delivery Problem. *Transportation Science*, *29*(1), 17–29. doi:10.1287/trsc.29.1.17

Sawaragi, Y., Nakayama, H., & Tanino, T. (1985). *Theory of Multiobjective Optimization*. *Mathematics in Science and Engineering* (Vol. 176). Academic.

Sexton, T. R., & Bodin, L. D. (1985a). Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: I. Scheduling. *Transportation Science*, *19*(4), 378–410. doi:10.1287/trsc.19.4.378

Sexton, T. R., & Bodin, L. D. (1985b). Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: II. Routing. *Transportation Science*, *19*(4), 411–435. doi:10.1287/trsc.19.4.411

Sexton, Thomas Raymond. (1979). *The single vehicle many to many routing and scheduling problem*. State University of New York.

Shaw, P. (1998). Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. *Computer*, *1520*, 417–431. doi:10.1007/3-540-49481-2_30

Tang, J., Kong, Y., Lau, H., & Ip, A. W. H. (2010). A note on “Efficient feasibility testing for dial-a-ride problems.” *Operations Research Letters*, *38*(5), 405–407. doi:10.1016/j.orl.2010.05.002

Toth, P., & Vigo, D. (1996). Fast Local Search Algorithms for the Handicapped Persons Transportation Problem. *Meta-Heuristics: Theory and Applications* (pp. 677–690).

Toth, P., & Vigo, D. (1997). Heuristic Algorithms for the Handicapped Persons Transportation Problem. *Transportation Science*, *31*(1), 60–71. doi:10.1287/trsc.31.1.60

Tuttle, W. T., & Nash-Williams, C. S. J. A. (2001). *Graph Theory* (1st thus e., p. 360). Cambridge University Press.

Viegas, R., & Azevedo, F. (2007). GRASPER: A Framework for Graph Constraint Satisfaction Problems. *EPIA'07 Proceedings of the aritficial intelligence 13th Portuguese conference on Progress in artificial intelligence*, 633–644.

Waugh, F. V. (1951). The Minimum-Cost Dairy Feed (An Application of "Linear Programming"). *Journal of Farm Economics*, *33*, 299–310. Retrieved from http://www.jstor.org/stable/1233608

Woeginger, G. J. (2003). Exact algorithms for NP-hard problems: A survey. *Combinatorial Optimization - Eureka, You Shrink!, LNCS*, 185–207.

Wong, K. I., & Bell, M. G. H. (2006). Solution of the Dial-a-Ride Problem with multi-dimensional capacity constraints. *International Transactions in Operational Research*, *13*(3), 195–208. doi:10.1111/j.1475-3995.2006.00544.x

Xiang, Z., Chu, C., & Chen, H. (2006). A fast heuristic for solving a large-scale static dial-a-ride problem under complex constraints. *European Journal of Operational Research*, *174*(2), 1117–1139. doi:10.1016/j.ejor.2004.09.060

Anexo A. Delegação Amadora – Sintra da Cruz Vermelha Portuguesa

Neste anexo apresentam-se alguns dados relativos às instâncias da delegação Amadora - Sintra complementado os apresentados no corpo deste documento. São também apresentados mapas com o intuito de contextualizar geograficamente a delegação, as suas unidades e extensões, bem como proporcionar uma perspectiva da dimensão geográfica da acção desta delegação.

Por outro lado nas secções que se seguem são também apresentados os pedidos de transporte relativos aos quatro dias de trabalho da delegação em estudo. O formato utilizado será ligeiramente diferente do disponibilizado pela CVP e presente em Loureiro (2010) com o objectivo de tornar mais explícitas e intuitivas as informações relativas aos pedidos de transporte, evitando os códigos usadas pela Cruz Vermelha Portuguesa e omitindo dados relativos aos nomes dos utentes e respectivas moradas, por questões de privacidade.

A.1. Envolvência Geográfica

Na Tabela A.1 constam as entidades que têm protocolo com a Cruz Vermelha Portuguesa para que esta possa transportar os utentes que necessitam dos seus serviços. Por ordem alfabética, as 12 entidades protocoladas surgem acompanhadas da sua localização e de qual a sua especialidade. Mais abaixo, na Figura A.1 as entidades protocoladas são novamente apresentadas, mas desta feita num mapa para que se possa ter noção da dimensão geográfica envolvida. Adicionalmente, um segundo mapa (Figura A.2) ilustra a dimensão geográfica da delegação Amadora – Sintra da Cruz Vermelha Portuguesa, contemplando os dois pólos e as duas extensões que tem a cargo.

Tabela A.1 - Entidades com protocolo com a Cruz Vermelha Portuguesa.

Entidade	Localização	Especialidade
Centro de Medicina de Reabilitação de Alcoitão	Alcoitão	Hospital
Centro de Reabilitação S. Jorge	Queluz	Centro de Fisioterapia
Fisiame	Lisboa	Centro de Fisioterapia
Fisiocentro	Amadora	Centro de Fisioterapia
Hemodial	Vila Franca de Xira	Centro de Hemodiálise
Hospital Curry Cabral	Lisboa	Hospital
Hospital de Sant'Ana	Lisboa	Hospital
Hospital Prof. Doutor Fernando Fonseca	Amadora	Hospital
Hospital Santa Cruz	Carnaxide	Hospital
Medifax	Lisboa	Centro de Fisioterapia
Movi Física	Amadora	Centro de Fisioterapia
Reabe	Lisboa	Centro de Fisioterapia

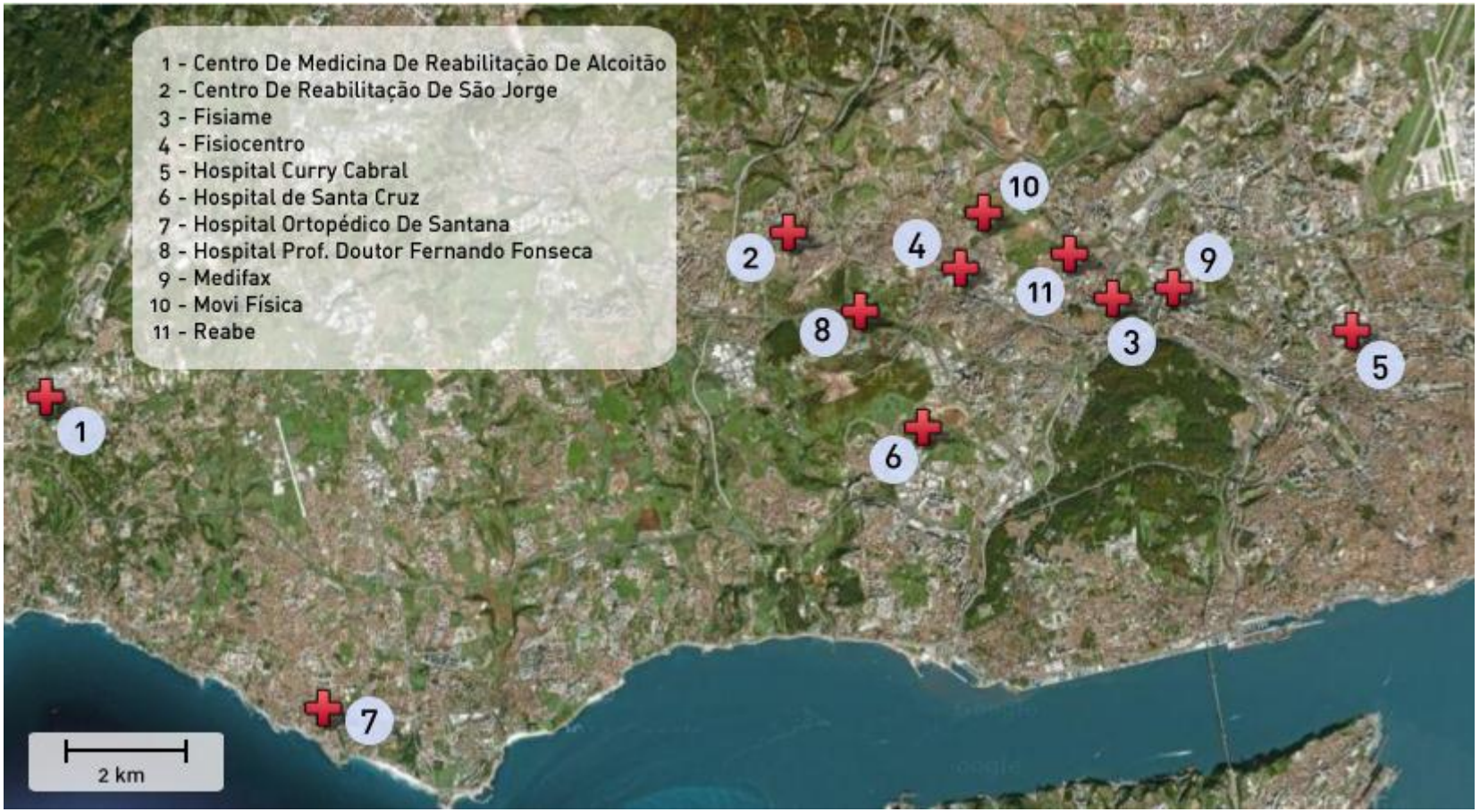

Figura A.1 - Representação geográfica da actividade da CVP.

Figura A.2 - Dimensão geográfica da delegação Amadora - Sintra da Cruz Vermelha Portuguesa.

A.2. Primeiro Dia

Tabela A.2 - Pedidos de Transporte pré tratamento para o primeiro dia.

Utente	Origem	Destino	Hora
C1	Rua N	Movi Física	08:30
C2	Largo E	Movi Física	08:45
C3	Rua B	Movi Física	09:00
C4	Rua C	Reabe	09:00
C7	Rua CC	Reabe	09:30
C9	Rua DF	Reabe	10:00
C11	Estrada M	Reabe	10:30
C12	Praceta R	Reabe	11:30
C14	Rua M	Reabe	12:30
C15	Rua S	Reabe	12:30

Tabela A.3 - Pedidos de Transporte pós tratamento para o primeiro dia.

Utente	Origem	Destino	Hora
C5	Movi Física	Rua DM	09:00
C6	Reabe	Rua V	09:15
C8	Reabe	Rua DF	10:00
C10	Reabe	Rua CA	10:15
C13	Reabe	Praceta DA	12:00
C16	Reabe	Praceta G	13:30
C15	Reabe	Rua S	14:00
C17	Reabe	Rua A	14:00
C18	Reabe	Estrada F	14:00
C19	Reabe	Avenida C	15:00
C20	Movi Física	Rua CD	16:00

Tabela A.4 - Mapeamento para as localizações do primeiro dia.

Número	Morada
1	Sede
2	Rua N
3	Largo E
4	Rua B
5	Rua C
6	Rua DM
7	Rua V
8	Rua CC
9	Rua DF
10	Rua CA

Número	Morada
11	Estrada M
12	Praceta R
13	Praceta DA
14	Rua M
15	Rua S
16	Praceta G
17	Rua A
18	Estrada F
19	Avenida C
20	Rua CD
21	Movi Física
22	Reabe

Tabela A.5 - Tempos de viagem (em minutos) entre os vários locais do primeiro dia.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	0	5	3	5	11	4	5	10	3	10	6	6	6	6	7	3	4	3	3	5	5	9
2	6	0	7	4	10	7	7	4	7	7	6	6	3	7	8	3	4	1	7	5	5	9
3	4	6	0	7	11	6	2	10	1	10	9	9	8	3	4	5	5	5	5	6	6	9
4	6	3	7	0	11	6	7	8	7	9	5	5	2	8	8	4	4	2	6	5	5	9
5	11	9	9	11	0	13	10	7	12	7	15	15	12	10	10	9	7	9	13	7	7	3
6	2	7	6	4	13	0	8	12	5	12	6	5	7	9	9	5	7	5	1	8	8	11
7	6	6	2	7	11	8	0	10	4	10	11	11	8	3	3	5	5	5	8	6	6	9
8	9	4	9	8	6	12	9	0	10	3	11	10	7	10	10	7	6	5	11	5	5	5
9	4	7	2	7	12	7	4	12	0	12	10	10	9	5	6	6	6	6	6	7	7	11
10	10	6	11	10	7	13	10	3	11	0	11	11	9	11	11	8	7	8	12	6	6	6
11	6	7	10	6	14	6	12	11	9	9	0	2	6	12	13	7	9	7	6	10	10	13
12	7	5	11	7	14	6	10	10	10	9	2	0	4	11	11	7	8	5	6	9	8	12
13	7	3	8	4	12	7	8	8	8	8	6	6	0	9	9	4	5	2	7	6	6	10
14	7	7	3	7	11	9	3	10	5	10	12	12	8	0	1	5	5	5	9	6	6	8
15	7	7	4	7	11	10	4	11	5	11	12	12	9	2	0	6	6	6	9	6	7	7
16	4	3	7	2	11	5	6	9	6	10	7	7	5	7	7	0	4	2	6	5	5	9
17	5	5	5	7	7	8	4	6	6	6	10	10	6	5	5	3	0	1	7	2	2	5
18	4	1	5	5	9	5	5	6	5	8	6	6	3	6	6	2	3	0	6	3	3	7
19	3	7	6	5	13	1	8	12	5	12	6	6	7	9	9	5	7	6	0	8	8	11
20	5	4	5	4	6	7	5	5	6	5	9	9	6	6	6	3	2	3	7	0	0	5
21	4	4	5	4	7	7	5	6	6	6	9	9	6	6	6	3	2	3	7	0	0	7
22	8	8	8	8	3	11	8	6	9	6	13	13	9	8	8	7	5	6	10	4	9	0

A.2.1. Melhor Solução Obtida

Tabela A.6 - Rota obtida para o primeiro dia de trabalho da CVP.

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Sede	07h00	00h27	0
Largo E	07h30	00h00	1
Movi Física	07h41	00h00	0
Rua N	07h50	00h00	1
Movi Física	08h00	00h04	0
Rua B	08h13	00h00	1
Movi Física	08h23	00h00	0
Movi Física	08h34	00h00	1
Rua DM	08h44	00h00	0
Rua C	08h52	00h00	1
Reabe	09h00	00h00	0
Praceta R	09h05	00h00	1
Reabe	09h18	00h00	0
Reabe	09h27	00h00	1
Rua V	09h39	00h00	0
Estrada M	09h44	00h00	1
Reabe	09h56	00h00	0
Rua CC	10h07	00h00	1
Reabe	10h14	00h00	2
Reabe	10h27	00h00	1
Rua DF	10h32	00h00	0
Rua M	10h37	00h00	1
Reabe	10h42	00h00	2
Praceta DA	10h53	00h00	1
Reabe	11h10	00h00	0
Reabe	11h26	00h00	1
Rua CA	11h38	00h00	0
Rua DF	11h49	00h00	1
Reabe	12h00	00h55	0
Rua S	13h00	00h25	1
Reabe	13h30	00h00	2
Reabe	13h41	00h00	1
Avenida C	13h50	00h56	0
Movi Física	15h00	00h00	1
Rua CD	15h10	00h13	0
Reabe	15h30	00h01	1
Reabe	15h36	00h19	2
Reabe	16h00	00h00	3
Reabe	16h12	00h00	4

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Rua S	16h25	00h00	3
Rua A	16h36	00h00	2
Estrada F	16h47	00h00	1
Praceta G	17h00	00h00	0
Sede	17h10	00h00	0

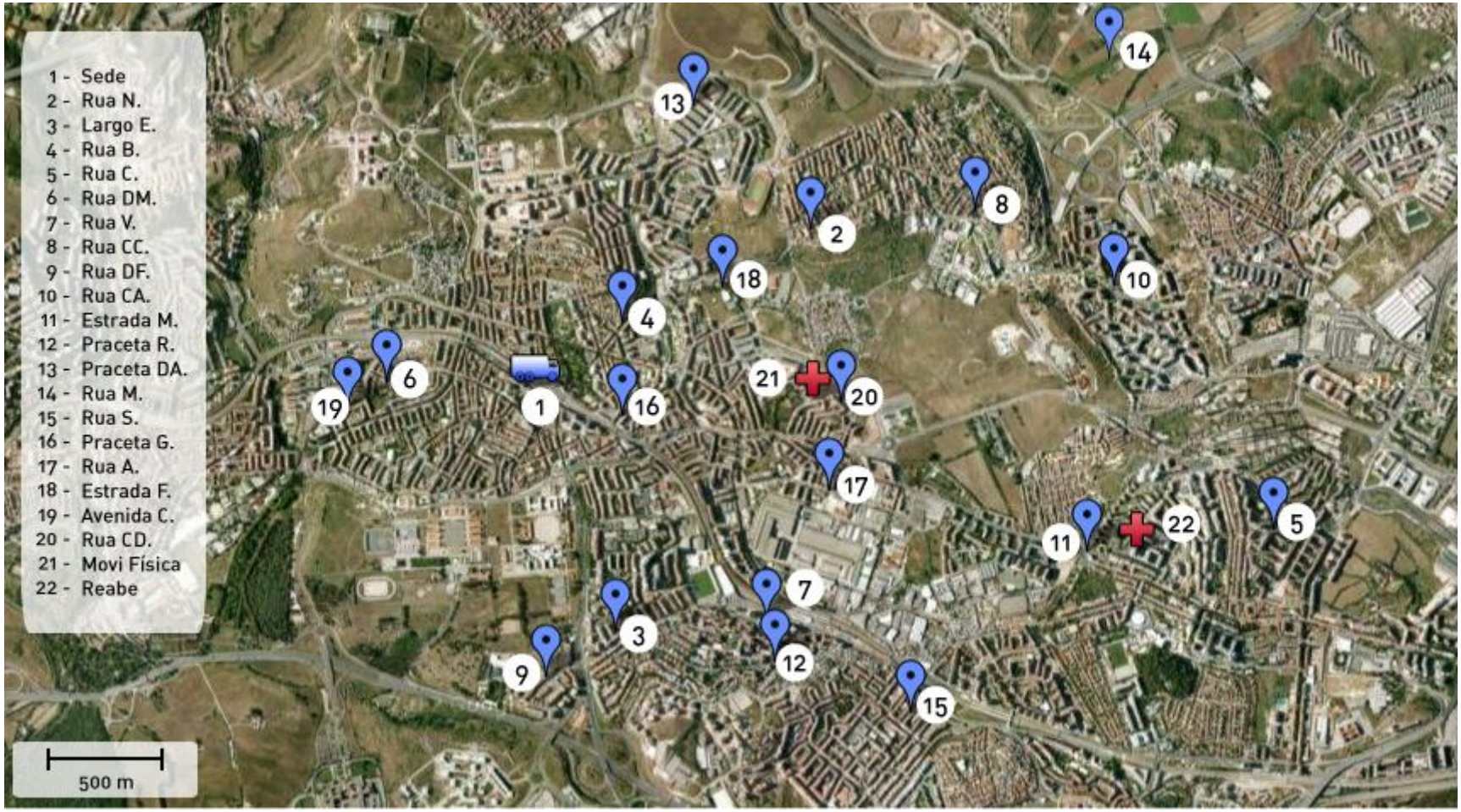

Figura A.3 - Contexto geográfico do primeiro dia de trabalho da CVP.

A.3. Segundo Dia

Tabela A.7 - Pedidos de Transporte pré tratamento para o segundo dia.

Utente	Origem	Destino	Hora
C1	Rua Q	Fisiame	08.30
C2	Rua DA	Fisiame	08.45
C3	Rua D	Reabe	09.00
C7	Rua CR	Movi Física	12.00
C8	Bairro B	Movi Física	12.30
C9	Rua G	Movi Física	13.15
C12	Rua R	Movi Física	15.00
C13	Rua T	Movi Física	15.30
C15	Rua F	Reabe	17.00

Tabela A.8 - Pedidos de Transporte pós tratamento para o segundo dia.

Utente	Origem	Destino	Hora
C4	Fisiame	Praceta M	09.30
C5	Reabe	Rua C	10.00
C6	Fisiame	Avenida G	10.15
C10	Movi Física	Avenida G	13.30
C11	Movi Física	Rua H	14.00
C14	Movi Física	Avenida G	16.00
C16	Reabe	Estrada M	17.00
C12	Movi Física	Rua R	17.30
C17	Movi Física	Rua A	17.30
C18	Reabe	Rua R	18.15

Tabela A.9 - Mapeamento para as localizações do segundo dia.

Número	Morada
1	Sede
2	Rua Q
3	Rua DA
4	Rua D
5	Praceta M
6	Rua C
7	Avenida G
8	Rua CR
9	Bairro B
10	Rua G
11	Rua H
12	Rua R

Número	Morada
13	Rua T
14	Rua F
15	Estrada M
16	Rua A
17	Movi Física
18	Reabe
19	Fisiame

Tabela A.10 - Tempos de viagem (em minutos) entre os vários locais do segundo dia.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
1	0	4	5	1	10	7	3	8	11	8	5	7	7	15	8	4	5	9	13
2	5	0	4	5	9	6	4	11	14	8	3	4	4	19	7	3	2	8	14
3	6	5	0	7	5	6	7	10	12	7	5	7	8	18	4	3	3	4	10
4	3	6	7	0	12	8	3	10	13	10	5	9	8	17	10	6	7	11	14
5	9	8	6	10	0	10	10	15	16	11	8	5	6	22	8	6	6	5	11
6	7	7	6	8	11	0	8	7	10	3	7	10	9	15	5	6	7	9	10
7	2	4	6	4	11	8	0	11	14	10	2	7	6	18	10	6	6	10	16
8	9	11	10	9	15	6	11	0	4	5	12	14	14	13	8	10	11	12	9
9	10	13	12	11	15	8	12	3	0	8	13	16	13	10	9	12	13	12	8
10	8	9	8	8	13	3	10	6	8	0	9	12	11	16	6	8	9	10	10
11	4	3	5	5	10	7	3	11	15	9	0	5	5	18	8	4	5	9	15
12	7	4	7	8	6	9	6	14	17	11	6	0	2	21	11	6	7	9	15
13	7	4	7	8	6	9	6	13	17	11	5	1	0	21	10	6	6	9	15
14	16	18	19	17	21	15	18	12	10	15	18	21	21	0	15	18	19	18	14
15	8	7	4	9	9	4	9	8	8	4	9	10	10	15	0	5	6	4	7
16	5	4	1	6	6	5	5	10	13	7	4	7	6	18	5	0	2	5	11
17	4	2	3	5	6	6	5	11	14	8	4	7	6	18	6	2	0	9	12
18	8	7	4	9	6	8	9	11	11	8	7	8	9	18	3	5	8	0	9
19	10	9	6	10	8	8	11	8	9	7	10	11	12	16	7	8	7	4	0

A.3.1. Melhor Solução Obtida

Tabela A.11 - Rota obtida para o segundo dia de trabalho da CVP.

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Sede	07h00	00h46	0
Rua D	07h47	00h00	1
Rua Q	07h58	00h00	2
Rua DA	08h07	00h00	3
Reabe	08h16	00h00	2
Fisiame	08h30	00h00	1
Fisiame	08h35	01h29	0
Movi Física	10h09	00h00	1

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Movi Física	10h14	00h00	2
Avenida G	10h27	00h00	1
Rua H	10h42	00h00	0
Rua R	10h57	00h00	1
Movi Física	11h10	00h00	0
Fisiame	11h18	00h00	1
Reabe	11h31	00h00	2
Fisiame	11h39	00h00	3
Rua C	11h55	00h00	2
Praceta M	12h00	00h00	1
Avenida G	12h05	01h49	0
Rua T	13h59	00h00	1
Movi Física	14h09	00h00	0
Bairro B	14h20	00h00	1
Movi Física	14h31	00h00	0
Rua CR	14h43	00h00	1
Movi Física	14h55	00h00	2
Movi Física	15h00	00h00	1
Avenida G	15h09	00h41	0
Rua G	16h13	00h00	1
Movi Física	16h36	00h00	2
Movi Física	16h41	00h00	1
Rua A	16h49	00h00	0
Rua F	17h00	00h00	1
Reabe	17h10	00h38	0
Movi Física	18h03	00h00	1
Reabe	18h16	00h00	2
Estrada M	18h23	00h00	1
Reabe	18h30	00h00	2
Rua R	18h42	00h00	1
Rua R	18h47	00h00	0
Sede	19h00	00h00	0

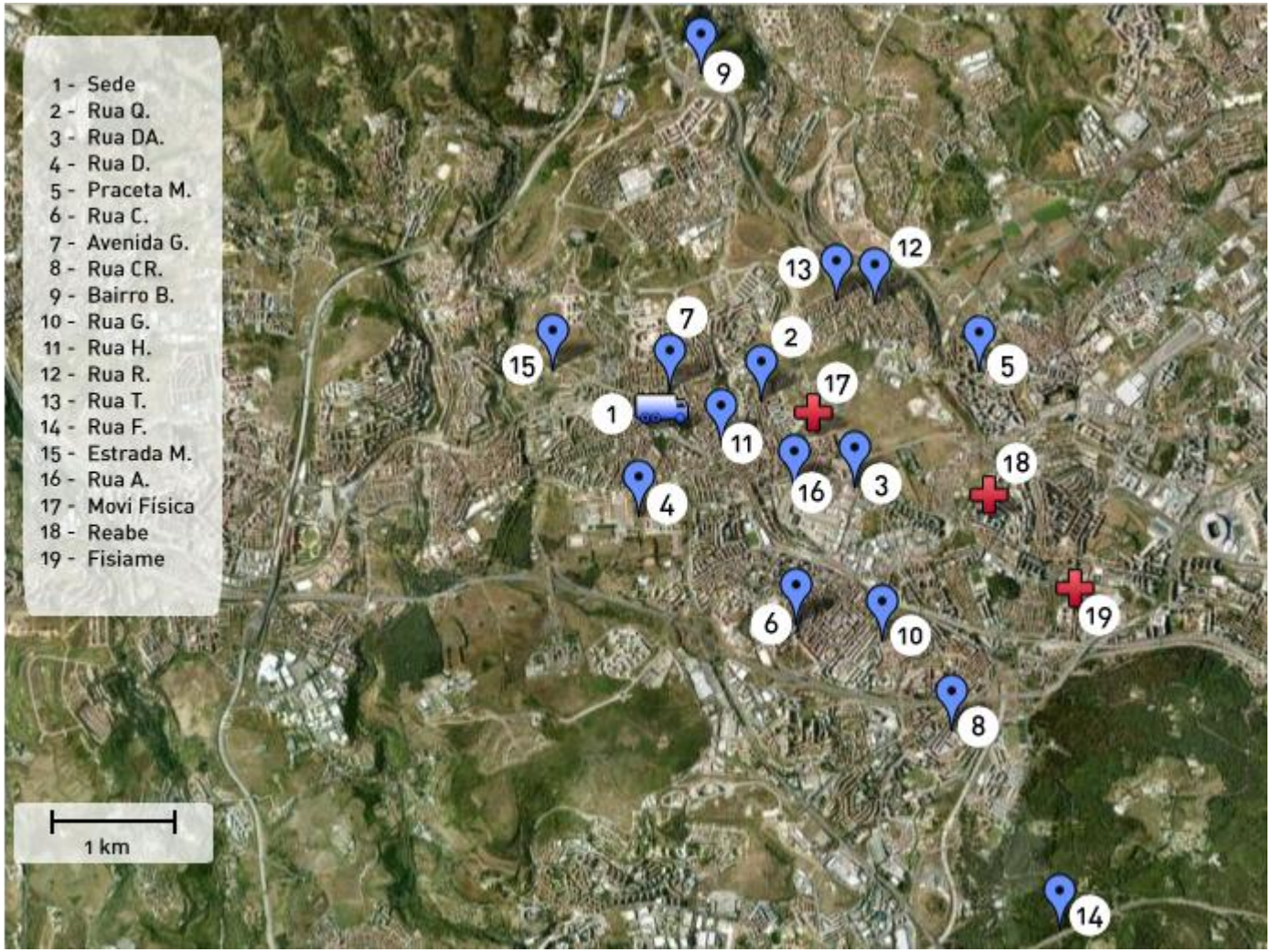

Figura A.4 - Contexto geográfico do segundo dia de trabalho da CVP.

A.4. Terceiro Dia

Tabela A.12 - Pedidos de Transporte pré tratamento para o terceiro dia.

Utente	Origem	Destino	Hora
C1	Praceta DF	Movi Física	10:15
C2	Estrada M	Movi Física	10:30
C3	Rua A	Movi Física	10:50
C4	Avenida R	Movi Física	11:30
C6	Avenida G	Reabe	12:30
C8	Avenida RU	Movi Física	13:15
C9	Avenida C	Reabe	13:30

Tabela A.13 - Pedidos de Transporte pós tratamento para o terceiro dia.

Utente	Origem	Destino	Hora
C5	Movi Física	Rua L	12:00
C7	Movi Física	Rua R	13:00
C10	Movi Física	Avenida M	15:00
C11	Movi Física	Rua E	15:30

Tabela A.14 - Mapeamento para as localizações do terceiro dia.

Número	Morada
1	Sede
2	Praceta DF
3	Estrada M
4	Rua A
5	Avenida R
6	Rua L
7	Avenida G
8	Rua R
9	Avenida RU
10	Avenida C
11	Avenida M
12	Rua E
13	Movi Física
14	Reabe

Tabela A.15 - Tempos de viagem (em minutos) entre os vários locais do terceiro dia.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	0	6	6	4	7	0	4	7	8	3	4	3	5	9
2	6	0	6	3	10	6	3	4	8	7	3	6	5	10
3	6	6	0	8	14	6	5	9	10	6	7	5	10	13

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
4	4	3	8	0	7	4	2	5	7	7	2	5	2	7
5	5	10	11	8	0	6	7	11	12	7	7	7	8	10
6	0	6	6	4	7	0	4	7	8	3	4	2	5	9
7	4	3	5	2	8	4	0	5	8	4	1	3	4	8
8	7	3	9	4	11	8	5	0	5	9	5	8	7	9
9	9	7	11	7	11	9	7	4	0	11	7	10	5	5
10	3	7	6	6	10	3	4	9	11	0	5	1	8	11
11	3	4	7	4	8	3	1	7	8	5	0	4	4	8
12	1	6	5	5	9	2	3	8	10	1	4	0	7	11
13	4	5	9	2	7	5	3	7	5	7	3	6	0	7
14	8	9	13	6	8	8	7	8	5	10	7	10	9	0

A.4.1. Melhor Solução Obtida

Tabela A.16 - Rota obtida para o terceiro dia de trabalho da CVP.

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Sede	07h00	02h19	0
Estrada M	09h25	00h00	1
Praceta DF	09h36	00h00	2
Movi Física	09h46	00h00	1
Movi Física	09h51	00h00	0
Avenida R	10h03	00h00	1
Movi Física	10h16	00h00	0
Rua A	10h23	00h00	1
Movi Física	10h30	01h26	0
Movi Física	12h08	00h00	1
Movi Física	12h17	00h00	2
Rua L	12h30	00h00	1
Rua R	12h35	00h05	0
Avenida G	12h50	00h00	1
Reabe	13h00	00h25	0
Avenida C	13h30	00h00	1
Reabe	13h40	02h11	0
Movi Física	16h01	00h00	1
Rua E	16h13	00h00	0
Avenida RU	16h25	00h00	1
Movi Física	16h30	00h00	2
Avenida M	16h38	00h00	1
Movi Física	16h47	00h00	0
Sede	16h53	00h00	0

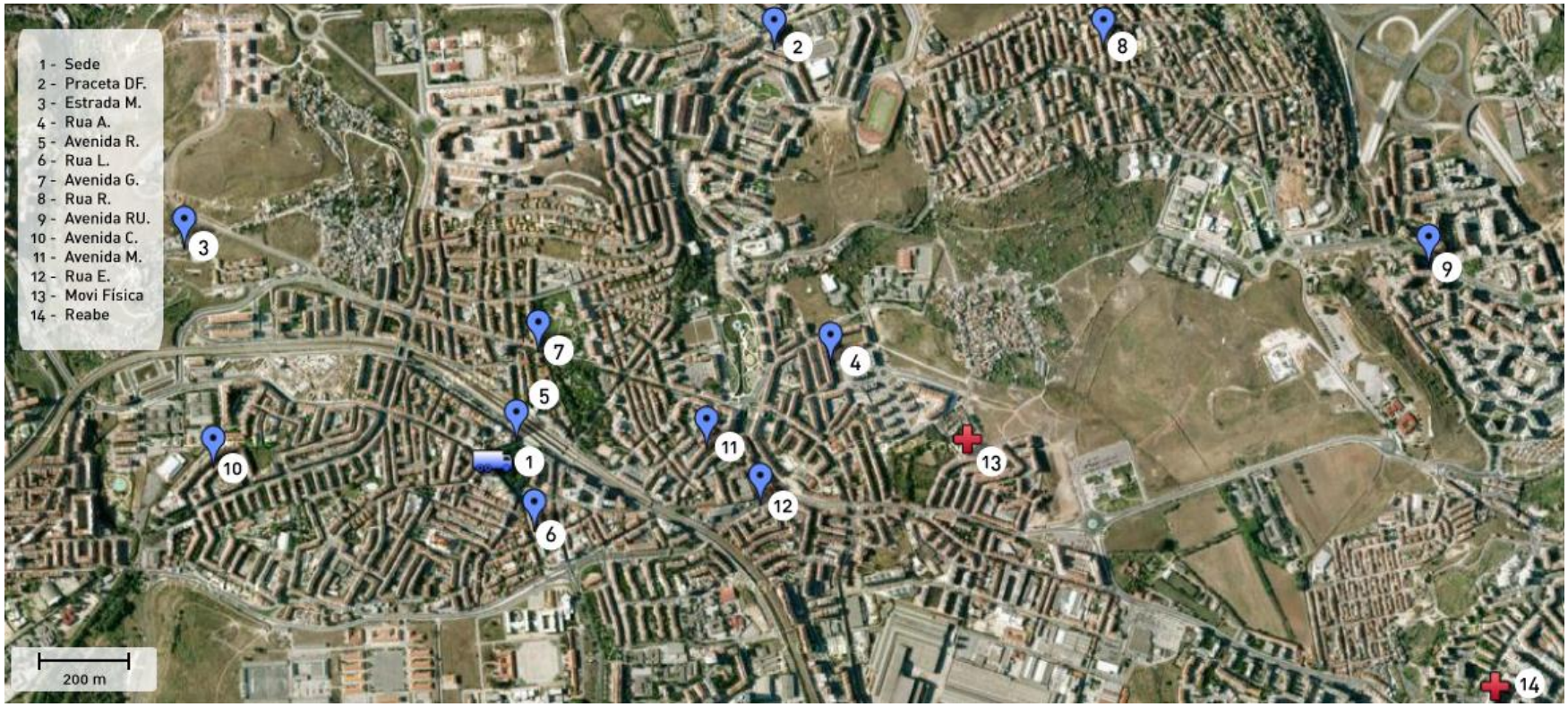

Figura A.5 - Contexto geográfico do terceiro dia de trabalho da CVP.

A.5. Quarto Dia

Tabela A.17 - Pedidos de Transporte pré tratamento para o quarto dia.

Utente	Origem	Destino	Hora
C1	Rua V	Movi Física	09:00
C2	Rua D	Reabe	09:00
C5	Rua DN	Movi Física	10:30
C6	Rua G	Movi Física	10:50
C7	Avenida DT	Movi Física	11:30
C9	Avenida D	Movi Física	12:00
C10	Rua R	Movi Física	12:30
C14	Rua I	Reabe	14:00
C15	Avenida P	Reabe	16:00
C16	Bairro S	Reabe	16:00

Tabela A.18 - Pedidos de Transporte pós tratamento para o quarto dia.

Utente	Origem	Destino	Hora
C3	Reabe	Rua A	10:00
C4	Movi Física	Praceta M	10:30
C8	Movi Física	Rua L	11:30
C11	Movi Física	Rua DO	12:30
C12	Movi Física	Rua M	13:00
C13	Movi Física	Rua F	13:30
C17	Reabe	Rua MO	16:00
C18	Fisiame	Rua E	17:00
C19	Reabe	Avenida R	17:00

Tabela A.19 - Mapeamento para as localizações do quarto dia.

Número	Morada
1	Sede
2	Rua V
3	Rua D
4	Rua A
5	Praceta M
6	Rua DN
7	Rua G
8	Avenida DT
9	Rua L
10	Avenida D
11	Rua R
12	Rua DO

Número	Morada
13	Rua M
14	Rua F
15	Rua I
16	Avenida P
17	Bairro S
18	Rua MO
19	Rua E
20	Avenida R
21	Movi Física
22	Reabe
23	Fisiame

Tabela A.20 - Tempos de viagem (em minutos) entre os vários locais do quarto dia.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	0	4	2	3	8	3	6	6	10	7	8	5	10	7	6	8	4	8	1	8	5	9	13
2	4	0	5	6	9	6	5	7	12	6	7	6	12	6	5	7	6	7	5	4	1	5	12
3	1	6	0	4	10	2	8	7	11	8	9	6	11	9	7	9	3	9	1	10	6	10	14
4	5	7	5	0	7	6	5	4	8	6	11	3	8	10	5	7	7	7	5	11	8	11	11
5	7	9	8	8	0	9	3	5	6	2	13	6	6	12	4	2	10	1	8	12	9	10	11
6	2	7	2	5	11	0	9	7	11	8	10	5	11	9	9	9	3	9	2	10	7	11	15
7	7	6	8	6	3	9	0	3	8	1	10	5	8	10	2	1	9	3	8	10	6	9	10
8	6	8	6	4	7	8	5	0	9	4	12	2	9	11	2	4	10	5	7	11	9	12	15
9	9	12	10	10	7	12	8	11	0	7	17	9	0	16	8	7	13	7	10	15	12	13	9
10	7	6	7	6	2	9	1	6	7	0	11	5	8	10	2	0	10	1	8	10	7	8	9
11	9	6	10	11	14	10	11	12	15	11	0	10	16	2	10	11	9	12	9	3	7	7	13
12	3	6	4	3	7	7	4	2	7	5	12	0	9	9	15	5	6	6	4	10	6	10	13
13	9	12	10	10	7	12	8	11	0	7	17	8	0	16	8	7	14	7	11	16	12	14	10
14	7	7	8	9	13	9	9	11	15	10	3	10	15	0	10	10	8	11	8	4	7	9	15
15	7	6	8	6	5	8	1	4	8	2	10	4	8	9	0	2	9	3	6	9	6	8	10
16	7	7	8	6	2	10	1	6	8	0	12	6	8	10	2	0	11	1	8	10	7	8	9
17	3	7	4	7	13	3	9	8	12	10	9	8	12	8	9	10	0	10	2	10	8	11	18
18	8	8	8	7	2	10	2	7	7	1	12	6	8	11	3	1	11	0	9	10	8	9	9
19	1	5	1	4	9	2	7	6	10	7	9	5	10	8	8	8	3	8	0	9	6	10	14
20	9	4	9	11	12	11	9	12	16	10	3	10	16	5	9	10	11	11	9	0	5	5	11
21	4	1	5	6	9	6	6	7	12	6	7	6	12	7	5	7	6	7	5	5	0	9	12
22	8	4	9	10	10	10	8	10	13	7	7	9	13	8	7	7	10	8	9	5	8	0	9
23	10	7	11	11	7	11	8	10	8	7	10	8	8	11	8	8	13	7	10	8	7	4	0

A.5.1. Melhor Solução Obtida

Tabela A.21 - Rota obtida para o quarto dia de trabalho da CVP.

Local	Tempo de Chegada	Tempo de Espera	Utentes a Bordo
Sede	07h00	01h28	0
Rua D	08h30	00h00	1
Reabe	08h45	00h00	0
Rua V	08h54	00h00	1
Movi Física	09h00	00h31	0
Reabe	09h42	00h00	1
Rua A	09h54	00h00	0
Movi Física	10h08	00h00	1
Rua DO	10h23	00h00	0
Rua DN	10h32	00h00	1
Movi Física	10h42	00h00	2
Praceta M	10h53	00h00	1
Movi Física	10h58	00h00	0
Rua G	11h09	00h00	1
Movi Física	11h21	00h00	0
Movi Física	11h26	00h00	1
Movi Física	11h31	00h00	2
Rua M	11h45	00h00	1
Rua F	11h56	00h00	0
Avenida DT	12h18	00h00	1
Movi Física	12h30	00h24	0
Movi Física	13h04	00h00	1
Rua L	13h17	00h00	0
Rua I	13h30	00h00	1
Avenida P	13h35	00h00	2
Reabe	13h52	00h00	1
Avenida D	14h04	00h00	2
Bairro S	14h16	00h00	3
Reabe	14h27	00h00	2
Reabe	14h41	00h05	1
Movi Física	15h00	00h29	0
Rua R	15h44	00h00	1
Movi Física	16h00	00h55	0
Reabe	17h00	00h00	1
Rua MO	17h13	00h24	0
Fisiame	17h51	00h00	1
Reabe	18h00	00h00	2
Avenida R	18h10	00h00	1
Rua E	18h24	00h00	0
Sede	18h30	00h00	0

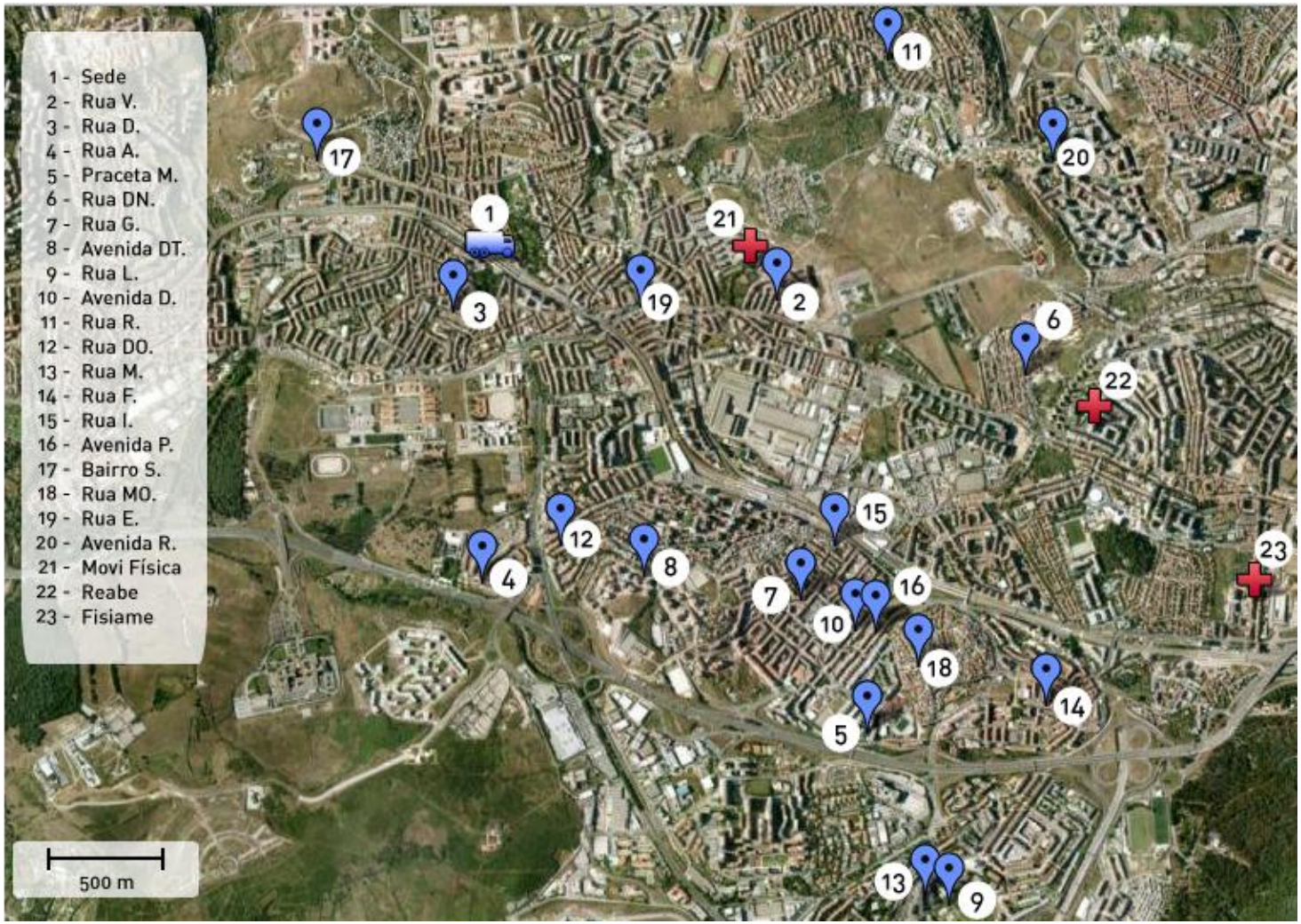

Figura A.6 - Contexto geográfico do quarto dia de trabalho da CVP.

Anexo B. Instância de teste para os Modelos

Neste anexo apresentam-se os dados da instância de teste cujo objectivo é avaliar os modelos propostos no que diz respeito à sua qualidade conceptual, isto é, declaração de variáveis e restrições com o objectivo de solucionar o *Dial-a-Ride Problem*.

Abaixo apresentam-se os dados relativos aos pedidos de transporte desta instância separando em diferentes tabelas os pedidos que antecedem e os que precedem o tratamento com o objectivo de tornar mais intuitiva a percepção e análise dos mesmos. Assim na Tabela B.1 apresentam-se os quatro pedidos de ida para as entidades de saúde, sendo que para cada pedido constam os locais de origem e destino, a hora a que os clientes têm o tratamento agendado, bem como a duração do serviço, em minutos, para cada local. Neste último caso assume-se que a duração do serviço, isto é, o tempo que o utente demora a entrar e sair do veículo é igual para os locais de um pedido, ou seja, para a sua origem e destino. Analogamente, a Tabela B.2 apresenta os dados relativos aos pedidos de regresso ao local de residência. Por fim, a Tabela B.3 mapeia as distâncias temporais, em minutos, entre os diversos locais envolvidos nos pedidos onde, o número presente na linha *i* e na coluna *j* representa o tempo de viagem entre o local *i* e o local *j*, como explicado no decorrer da Secção 7.1.4.

Tabela B.1 - Pedidos de Transporte da Instância de Validação.

Cliente	Origem	Destino	Hora	Duração
C1	Residência 1	Entidade 1	9:00	05m00
C2	Residência 2	Entidade 2	10:30	03m00
C3	Residência 3	Entidade 1	08:45	10m00
C4	Residência 4	Entidade 2	08:35	04m00

Tabela B.2 - Pedidos de Transporte após tratamento da Instância de Validação.

Cliente	Origem	Destino	Hora	Duração (min)
C1	Entidade 1	Residência 1	10:15	03m00
C5	Entidade 2	Residência 2	11:15	04m00
C6	Entidade 1	Residência 3	11:40	05m00
C4	Entidade 2	Residência 4	10:00	06m00

Tabela B.3 - Tempos de viagem (em minutos) entre os vários locais da Instância de Teste.

	Depósito	**Residência 1**	**Residência 2**	**Residência 3**	**Residência 4**	**Entidade 1**	**Entidade 2**
Depósito	0	11	17	6	20	13	24
Residência 1	11	0	5	12	25	12	10
Residência 2	17	5	0	16	10	7	7
Residência 3	6	12	16	0	12	6	23
Residência 4	20	25	10	12	0	7	9
Entidade 1	13	12	7	6	7	0	13
Entidade 2	24	10	7	23	9	13	0

Anexo C. Instâncias Referência

Ainda que as instâncias de referência já tinham sido abordadas na Secção 8.1.2, pretende-se apresentar o seu formato que não é de todo intuitivo e, através de um exemplo, transformá-lo para o formato que tem vindo a ser usado nesta dissertação para os pedidos de transporte.

Assim sendo, nestas instâncias os pedidos são genericamente apresentados de acordo com a Tabela C.1 e exemplificados, para a instância *a2-16*, pela Tabela C.2.

Tabela C.1 - Formato genérico das instâncias referência.

Nº de Veículos	Nº de Pedidos	Tempo Máximo por Rota	Capacidade dos Veículos	Tempo Máximo por Pedido
N º do Pedido	Posição Cartesiana	Duração do Serviço	Nº Passageiros	Janela Temporal
…	…	…	…	…
N º do Pedido	Posição Cartesiana	Duração do Serviço	Nº Passageiros	Janela Temporal

Tabela C.2 - Instância a2-16.

2	16	480	3	30
0	[0, 0]	0	0	[0, 1140]
1	[-1.198,-5.164]	3	1	[0, 1140]
2	[5.573, 7.114]	3	1	[0, 1140]
3	[-6.614, 0.072]	3	1	[0, 1140]
4	[-7.374, -1.107]	3	1	[0, 1140]
5	[-9.251, 8.321]	3	1	[0, 1140]
6	[6.498, -6.036]	3	1	[0, 1140]
7	[0.861, 6.903]	3	1	[0, 1140]
8	[3.904, -5.261]	3	1	[0, 1140]
9	[7.976, -9.000]	3	1	[276, 291]
10	[-2.610, 0.039]	3	1	[32, 47]
11	[4.487, 7.142]	3	1	[115, 130]

12	[8.938, -4.388]	3	1	[14, 29]
13	[-4.172, -9.096]	3	1	[198, 213]
14	[7.835, -9.269]	3	1	[160, 175]
15	[2.792, -7.944]	3	1	[180, 195]
16	[5.212, 9.271]	3	1	[366, 381]
17	[6.687, 6.731]	3	-1	[402, 417]
18	[-2.192, -9.210]	3	-1	[322, 337]
19	[-1.061, 8.752]	3	-1	[179, 194]
20	[6.883, 0.882]	3	-1	[138, 153]
21	[5.586, -1.554]	3	-1	[82, 97]
22	[-9.865, 1.398]	3	-1	[49, 64]
23	[-9.800, 5.697]	3	-1	[400, 415]
24	[1.271, 1.018]	3	-1	[298, 313]
25	[4.404, -1.952]	3	-1	[0, 1140]
26	[0.673, 6.283]	3	-1	[0, 1140]
27	[7.032, 2.808]	3	-1	[0, 1140]
28	[-0.694, -7.098]	3	-1	[0, 1140]
29	[3.763, -7.269]	3	-1	[0, 1140]
30	[6.634, -7.426]	3	-1	[0, 1140]
31	[-9.450, 3.792]	3	-1	[0, 1140]
32	[-8.819, -4.749]	3	-1	[0, 1140]

Na Tabela C.2 a primeira linha indica que a instância em causa considera 16 pedidos que serão servidos por uma frota homogénea composta por dois veículos com capacidade para 3 passageiros sentados. O tempo máximo para servir um dado pedido de transporte é de meia hora, isto é 30 minutos, e cada viatura que compõe a frota pode operar num máximo de 480 minutos, isto é, 8 horas, o período diário de operação da entidade que presta o serviço de transporte. De seguida seguem-se os dados relativos ao depósito de onde partem as viaturas e onde devem também regressar no final do serviço. Neste caso o tempo de serviço considerado é nulo já que não existe nenhuma operação de entrada ou saída de utentes da viatura, daí o campo respectivo ao número de passageiros ser também nulo.

Após a linha do depósito, as linhas que se seguem dizem respeito aos pedidos de transporte propriamente ditos, sendo um pedido de transporte dividido por duas linhas, a i-ésima e a i-ésima + *n*, com *n* = 16. A primeira diz respeito ao local de origem e a segunda ao de destino. Em particular, a primeira linha relaciona-se com a décima sétima, tratando-se de um pedido de transporte de uma pessoa, com um tempo de serviço de 3 minutos, do local dado pelas coordenadas [-1.198,-5.164] para o local dado por [6.687, 6.731] ao qual o utente deve chegar no intervalo de tempo dado pela janela temporal [402, 417].

Por uma questão de simplicidade e coerência apresentam-se abaixo a tabela que contém os pedidos de ida para as entidades prestadoras de serviços ligados à área da saúde e outra que contém os pedidos de regresso à residência. Na ausência de referências relativas a clientes, as tabelas serão compostas apenas por três colunas, respeitantes ao local de origem do pedido, ao local de destino e à hora associada ao pedido. Apenas se permite transportar um passageiro e o tempo de serviço é sempre de 3 minutos.

Tabela C.3 - Pedidos de transporte para entidades da instância a2-16.

Origem	Destino	Hora
[-1.198,-5.164]	[6.687, 6.731]	06:57
[5.573, 7.114]	[-2.192, -9.210]	05:37
[-6.614, 0.072]	[-1.061, 8.752]	03:14
[-7.374, -1.107]	[6.883, 0.882]	02:33
[-9.251, 8.321]	[5.586, -1.554]	01:37
[6.498, -6.036]	[-9.865, 1.398]	01:04
[0.861, 6.903]	[-9.800, 5.697]	06:55
[3.904, -5.261]	[1.271, 1.018]	05:13

Tabela C.4 - Pedidos de transporte após tratamento da instância a2-16.

Origem	Destino	Hora
[7.976, -9.000]	[4.404, -1.952]	04:36
[-2.610, 0.039]	[0.673, 6.283]	00:32
[4.487, 7.142]	[7.032, 2.808]	01:55
[8.938, -4.388]	[-0.694, -7.098]	00:14
[-4.172, -9.096]	[3.763, -7.269]	03:18
[7.835, -9.269]	[6.634, -7.426]	02:40
[2.792, -7.944]	[-9.450, 3.792]	03:00
[5.212, 9.271]	[-8.819, -4.749]	06:06

A tabela das distâncias temporais entre os diversos locais não será apresentada, no entanto fica a nota de que neste tipo de instâncias as distâncias são calculadas e acordo com a fórmula da distância euclidiana.