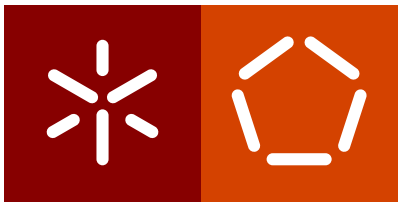
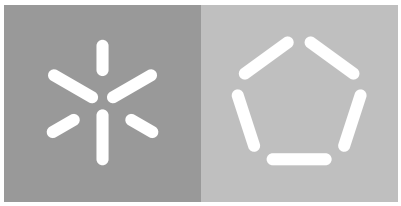


Universidade do Minho
Escola de Engenharia
Departamento de Informática

Gonçalo Dias Camaz Moreira

Otimização de Recursos de Rede para Encadeamento de Serviços em Segment Routing

Janeiro 2021



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Gonçalo Dias Camaz Moreira

Otimização de Recursos de Rede para Encadeamento de Serviços em Segment Routing

Dissertação de Mestrado
Mestrado Integrado em Engenharia Informática

Dissertação orientada por
Prof. Doutor Pedro Nuno Sousa
Prof. Doutor Vitor Sá Pereira

Janeiro 2021

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.
Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Gonçalo Dias Camaz Moreira

AGRADECIMENTOS

O desenvolvimento desta dissertação culminou num processo longo de aprendizagem e desenvolvimento pessoal. Nesse sentido, gostaria de começar por agradecer ao Prof. Dr. Pedro Nuno Sousa e ao Prof. Dr. Vítor Sá Pereira pelo acompanhamento, empenho, mas acima de tudo por me terem dado a oportunidade de aprender e aplicar novos conceitos neste trabalho. Gostaria ainda de agradecer por todo o apoio que me deram e pela forma como me cativaram para o tema.

Não poderia deixar de mencionar todos aqueles que estiveram mais próximos durante esta fase e que me deram apoio incondicional, mesmo em alturas de maior ausência. Agradeço então aos meus pais, António e Rosa, à minha irmã, Ana Beatriz, à minha namorada, Margarida e a todos os meus amigos mais próximos.

Por fim, agradeço a todos os meus colegas de curso e professores que me acompanharam ao longo destes 5 anos.

A todos, um muito obrigado.

ABSTRACT

Segment Routing (SR) is an implementation of Software-Defined Networking (SDN) for traffic routing. It derives from a source routing paradigm where each node, at the network's periphery, adds a list of labels to the packet's header defining explicitly the path each traffic flow should follow to its destination. Each label is a segment that identifies a topological instruction, a service, or even a processing instruction. Although SR can be very similar to Multi-Protocol Label Switching (MPLS), concerning the simple routing of packets, its utilization simplifies the network management tasks. On the other hand, SR provides solutions to existing scalability problems on SDN implementations, for example, OpenFlow-based. This technology is rapidly becoming a standard and is already supported by big players such as Cisco and Huawei, performing a fundamental role in the architecture of future 5G networks. Also, due to these new technology capabilities, there is an increasing number of novel services. This growth, allied with the rise of 5G capable devices, motivated the deployment of services on crucial points of the network. Consequently, mobile operators created the Network Function Virtualization (NFV) concept.

However, some important questions should be considered, such as how to optimize network resources regarding service locations to reduce the congestion level on the network. The main contribution of this work consists of exploring solutions based on SR and NFV. Thus, relying on optimization techniques to pursue optimal resource allocation for traffic routing and service processing requirements.

Keywords: Evolutionary Algorithms, Optimization, Machine Learning, Network Function Virtualization, Segment Routing

RESUMO

O *Segment Routing* (SR) é uma implementação de *Software Defined Networking* (SDN) para encaminhamento de tráfego. Baseado num paradigma de *source routing*, cada nodo na fronteira da rede adiciona um conjunto de etiquetas ao cabeçalho dos pacotes e define explicitamente o caminho que cada fluxo de tráfego deverá percorrer até ao destino. Cada etiqueta é um segmento que identifica uma instrução topológica, um serviço ou mesmo uma instrução de processamento. Embora o SR seja muito parecido com o *Multi Protocol Label Switching* (MPLS) no que respeita ao encaminhamento simples de pacotes, a sua utilização simplifica os processos de gestão da rede. Por outro lado, o SR também proporciona soluções para problemas de escalabilidade existente em implementações SDN como o *OpenFlow*. Esta tecnologia está rapidamente a tornar-se um *standard* e é já suportada por empresas importantes, como a Cisco e a Huawei, desempenhando um papel importante na arquitetura das futuras redes 5G. Face ao desenvolvimento desta nova tecnologia, surgiram novos tipos de serviço que tiram partido das capacidades da mesma. Devido ao crescente número de dispositivos 5G, surgiu a necessidade de disponibilizar os serviços em diversos pontos da rede, fornecendo assim resposta às necessidades requeridas. Como tal, as operadoras de telecomunicações desenvolveram o conceito de *Network Function Virtualization* (NFV).

Todavia existem questões importantes que necessitam ser tratadas, nomeadamente, como distribuir os serviços pela topologia de forma a reduzir o nível de utilização das ligações e nodos da mesma. Este trabalho tem como objetivo a exploração de soluções baseadas em SR e em NFV, recorrendo a técnicas de otimização como algoritmos evolucionários e *machine learning*.

Palavras Chave: Algoritmos Evolucionários, Otimização, *Machine Learning*, *Network Function Virtualization*, *Segment Routing*

CONTEÚDO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE LISTAGENS

LISTA DE SIGLAS

BGP Border Gateway Protocol. 6

COTS Commercial-Off-The-Shelf. 20

DEFT Distributed Exponentially-weighted Flow SpliTting. 19
DNN Deep Neural Network. 26
DV Distance-Vector. 8

eBGP External Border Gateway Protocol. 9
ECMP Equal Cost Multi Path. 14
EGP Exterior Gateway Protocol. 6
EM Element Management. 20
ETSI European Telecommunications Standards Institute. 20

FEC Forwarding Equivalence Class. 10
FIB Forwarding Information Base. 7

iBGP Internal Border Gateway Protocol. 9
IETF Internet Engineering Task Force. 1
IGP Interior Gateway Protocols. 6
IS-IS Intermediate System to Intermediate System. 8

LDP Label Distribution Protocol. 12
LER Label Edge Router. 11
LS Link-State. 8
LSP Label Switching Path. 11
LSR Label Switch Router. 11
LSTM Long Short Term Memory. 27

MEI Mestrado em Engenharia Informática. 1
MILP Mixed-integer linear programming. 32
ML Machine Learning. 25
MLU Maximum Link Utilization. 16
MNU Maximum Node Utilization. 33
MPLS Multi Protocol Label Switching. 10

NFV Network Function Virtualization. 20

NFVI Network Function Virtualization Infrastructure. 20
N-PoP Network Point of Presence. 21

OSPF Open Shortest Path First. 8

PCE Path Computer Element. 19
PEFT Penalizing Exponential Flow-spliTting. 19
PNF Physical Network Function. 20

RBF Gaussian Radial Basis Function. 27
RF Random Forest. 27
RIB Routing Information Base. 7
RIP Routing Information Protocol. 8
RNN Recursive Neural Network. 27

SDN Software Defined Networking. 13
SFC Service Function Chaining. 38
SID Segment Identifier. 13
SR Segment Routing. 13
SVM Support Vector Machine. 27
SVR Support Vector Regression. 27

TTL Time to live. 11

UM Universidade do Minho. 1

VNF Virtual Network Function. 20

1

INTRODUÇÃO

1.1 INTRODUÇÃO

As redes IP têm-se tornado, ao longo dos últimos anos, cada vez mais difíceis de gerir. O aumento das necessidades de encaminhamento e o aparecimento de novos serviços são alguns dos fatores que contribuem para uma maior complexidade das redes, ao qual se associa a necessidade dos operadores cumprirem com acordos de nível e qualidade de serviço. Para responder a estas novas exigências, o Internet Engineering Task Force (IETF) introduziu o conceito de *Software Defined Networking* (SDN) no intuito de tornar as redes mais flexíveis e programáveis. O *Segment Routing* (SR) [15], uma implementação de SDN, explora as vantagens da comutação baseada em etiquetas, como o *Multi Protocol Label Switching* (MPLS), respondendo às limitações deste último, com especial ênfase em problemas de escalabilidade, simplicidade e gestão de operações [14].

O SR implementa um paradigma de encaminhamento no qual cada nodo de origem direciona os fluxos de pacotes especificando uma lista de pontos de entrega intermédios que deverão ser visitados no caminho para o destino final. A capacidade de direcionar o fluxo de tráfego por determinados pontos da rede torna o SR adequado para solucionar problemas relacionados com engenharia de tráfego. Existem diversas propostas que procuram dar resposta a diferentes problemas de engenharia de tráfego em SR, considerando implementações totais ou parciais da tecnologia. No entanto, essas soluções têm um custo. Algumas são muito complexas, enquanto outras podem ter um impacto negativo no desempenho da rede, como o aumento do atraso de entrega dos pacotes resultante de caminhos mais longos ou congestionados. Muitos estudos na área mencionam, como principal preocupação do SR, o *overhead* excessivo, em especial no que toca a largura de banda desperdiçada, devido à inserção de etiquetas no cabeçalho dos pacotes [43].

Com o aparecimento de novos tipos de serviços e face às crescentes necessidades de encaminhamento, especialmente após o surgimento de novas tecnologias como o 5G, os operadores de telecomunicações viram-se obrigados a seguir novas soluções. Estas surgem através de *software* com recurso a *Network Function Virtualization*. Esta arquitetura tem como objetivo reduzir os custos de implementação de novos serviços, que passam a estar dis-

tribuídos na rede. No entanto, esta distribuição levanta novos problemas relacionados com a localização dos mesmos. Nesse sentido, uma distribuição correta dos serviços, é essencial para garantir uma utilização ótima dos recursos de encaminhamento e processamento de tráfego disponíveis.

Os problemas de gestão e otimização de recursos são todavia, em grande parte, de complexidade elevada e não resolúveis em tempo polinomial (NP-Difíceis). Como tal, envolvem frequentemente a optimização simultânea de mais do que um objetivo sujeito a um conjunto de restrições. Consequentemente, são utilizadas meta-heurísticas como, por exemplo, algoritmos evolucionários, *machine learning*, *simulated annealing*, *local search*, entre outras.

1.2 OBJETIVOS

Esta dissertação pretende explorar e analisar diferentes soluções de encaminhamento baseado em SR, bem como estudar a aplicabilidade das mesmas a novos paradigmas das redes de computadores como o *Network Function Virtualization*. Nesse sentido será realizado um estudo acerca dos conceitos relacionados com redes SDN nas quais se enquadram a tecnologia SR e a arquitetura NFV. Para além disso será efetuado um levantamento de trabalhos relacionados com problemas de engenharia de tráfego que abordem os conceitos mencionados.

O objetivo global deste trabalho consiste na proposta e análise de novas soluções para a otimização da utilização de recursos NFV em SR com vista à sua aplicação num contexto *5G*. Estes recursos incluem não somente a utilização dos nodos (custo de instalação, cpu, memória, etc...) mas também os recursos de encaminhamento de tráfego, como a utilização de ligações.

Neste contexto, a dissertação tem os seguintes objetivos parciais:

- Investigar os conceitos associados à tecnologia SR e à arquitetura NFV.
- Analisar trabalhos relacionados com problemas de engenharia de tráfego de forma a estudar as possíveis abordagens a considerar.
- Propor mecanismos de otimização capazes de otimizar a utilização de recursos.
- Definir os casos de estudo para avaliar as soluções propostas com as abordagens definidas.
- Implementar os mecanismos de otimização estendendo uma *framework* já existente.
- Analisar os resultados obtidos e concluir acerca da aplicabilidade da solução à otimização de SR na arquitetura NFV.

1.3 ORGANIZAÇÃO DO DOCUMENTO

Este documento está organizado em seis capítulos. A descrição de cada um dos capítulos é a seguinte:

- Introdução: Este capítulo efetuou o enquadramento e a contextualização do trabalho a ser desenvolvido. Apresentou o tema geral e definiu os objetivos a serem atingidos na dissertação.

- Estado da Arte: Aborda a base teórica que irá permitir a especificação e implementação de uma solução com a finalidade de dar resposta à problemática de alocação de recursos numa rede com base em SR e NFV. Para além do conteúdo inerente a estes conceitos, são ainda abordados mecanismos de optimização capazes de resolver problemas NP-Difíceis. Por fim é apresentada a *framework* na qual o trabalho vai ser implementado.

- Especificação e Modelação do Problema: Aborda a especificação do problema descrevendo todas as etapas necessárias para atingir a sua resolução. Destas fazem parte a modelação de dois modelos lineares capazes de analisar configurações topológicas com capacidade NFV. É ainda explicado neste capítulo a importância dos modelos definidos e de que forma estes são utilizados no decorrer das diversas abordagens.

- Implementação e Integração na *Framework* NetOpt: Aborda a implementação do problema na NetOpt. Nesse sentido é apresentado um diagrama geral do trabalho desenvolvido. Nas secções seguintes é apresentado em detalhe cada um dos componentes desenvolvidos.

- Análise de Resultados: Demonstra os vários casos de estudo, a metodologia de testes utilizada e avaliação dos resultados obtidos.

- Conclusões: Neste capítulo são apresentadas as principais conclusões obtidas do trabalho realizado. É ainda realizada uma análise aos capítulos prévios, referenciando as principais contribuições obtidas. Por fim serão expostas possíveis melhorias à solução proposta.

2

ESTADO DA ARTE

Este capítulo abordada conceitos essenciais do encaminhamento de tráfego (secção 2.1). Nas secções seguintes, serão introduzidas noções importantes do *Label Switching* e da sua implementação no protocolo MPLS (secção 2.2) bem como no *Segment Routing* (secção 2.3). Serão apresentados trabalhos realizados na área do SR, com particular ênfase nos relacionados com problemas de engenharia de tráfego. Na Secção 2.4 será abordado o conceito de *Network Function Virtualization* e a relação do mesmo com tecnologias 5G. Serão ainda abordados mecanismos de otimização como computação evolucionária e *machine learning* (secção 2.5). Para finalizar será apresentada a arquitetura da *framework* NetOpt (secção 2.6), os casos de uso da mesma e identificados os principais mecanismos de otimização disponíveis.

2.1 PROTOCOLOS DE ENCAMINHAMENTO

Define-se como *routing* o processo de seleção e definição de rotas que permita o encaminhamento de tráfego entre dois ou mais pontos numa rede. Este é normalmente realizado com recurso a algoritmos e protocolos de encaminhamento, tendo como objetivo construir tabelas de encaminhamento nos diversos nodos da rede. Essas tabelas são geralmente mantidas dinamicamente com o intuito de refletir a topologia atualizada [2].

Os protocolos de encaminhamento são desenvolvidos procurando atingir determinados objetivos. Incluídos nestes objetivos encontram-se: a *Precisão* - capacidade do protocolo encontrar a melhor rota para um determinado destino; a *Simplicidade* - associado ao *overhead* que deve ter em conta o uso de memória e tempo de CPU; a *Robustez* - capacidade de suportar falhas na rede; a *Convergência* - capacidade do protocolo convergir rapidamente quando a rede sofre alterações, implicando o recalculo de rotas; a *Flexibilidade* - existência de diferentes métricas e rotas; e a *Escalabilidade* - capacidade do protocolo manter um bom desempenho em redes com um número crescente de nodos e ligações entre eles.

Com a expansão da Internet, deu-se um aumento da complexidade da rede, tornando impossível manter uma representação detalhada da mesma. Surgiu assim a necessidade

de agrupar diferentes sub redes numa só, formando um grupo de sub-redes com diversos *end-points*, reduzindo assim a complexidade.

Um sistema autónomo [23] consiste num segmento da rede topológica, que tem associado um conjunto de prefixos ou sub-redes. Estas estão conectadas por um grupo de *routers*, que possuem um conjunto comum de regras, como por exemplo estratégias de encaminhamento. Os sistemas autónomos estão normalmente sob controlo de uma única organização, possuindo como identificação única um número (e.g. 65200).

No caminho para o destino, os pacotes poderão ter de passar por um ou mais *routers* de sistemas autónomos diferentes. Como tal, foi necessário definir protocolos de comunicação, dentro (*Intradomain Routing*) e fora (*Interdomain Routing*) dos sistemas autónomos. Dentro do sistema autónomo são usados protocolos Interior Gateway Protocols (IGP). Entre sistemas autónomos são usados protocolos Exterior Gateway Protocol (EGP). O EGP sofreu algumas evoluções sendo que, atualmente, o protocolo mais utilizado é o Border Gateway Protocol (BGP).

O encaminhamento pode ser definido como sendo estático ou dinâmico. No encaminhamento estático, as tabelas não sofrem atualizações, sendo esta solução mais adequada a redes de menor dimensão. No entanto, esta modalidade de encaminhamento exige uma constante monitorização por parte de um administrador da rede. Por sua vez, no encaminhamento dinâmico, as tabelas sofrem atualizações automáticas através dos protocolos de encaminhamento, permitindo assim aumentar a escalabilidade e robustez.

Na Figura 2.1 é possível ver alguns exemplos de protocolos que realizam encaminhamento dinâmico bem como a família de algoritmos na qual se incluem.

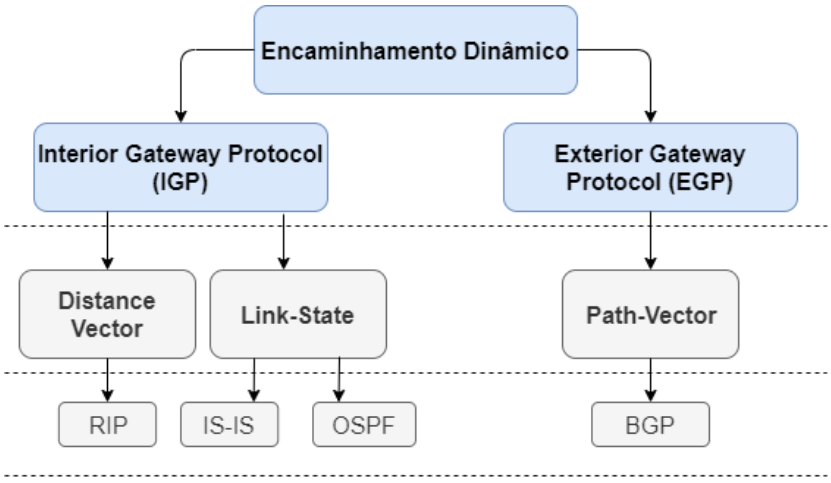


Figura 2.1: Protocolos de Encaminhamento (Adaptado de [40]).

No encaminhamento dinâmico, como referido anteriormente, as tabelas sofrem atualizações automáticas sob ação dos protocolos de encaminhamento. Essas atualizações são refletidas

nas tabelas dos equipamentos da rede que são compostos por dois níveis lógicos: plano de controlo (*Control Plane*) e plano de encaminhamento (*Data Plane* ou *Forwarding plane*), que desempenham papeis distintos no encaminhamento de pacotes.

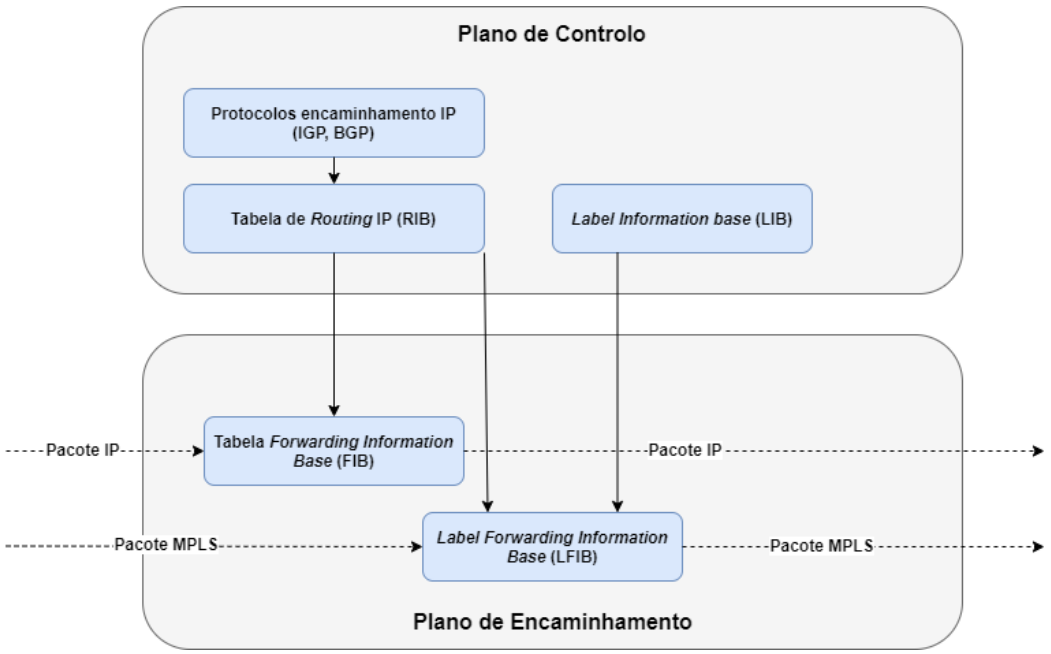


Figura 2.2: Arquitectura de um equipamento de rede (Adaptado de [33]).

O plano de controlo é responsável pela decisão de encaminhamento de cada pacote, tomando a mesma com base na tabela de *routing* IP Routing Information Base (RIB). Essa tabela possui os resultados da aplicação dos protocolos de encaminhamento, IGP e BGP. O plano de encaminhamento, através das decisões tomadas por parte do plano de controlo, concretiza o encaminhamento dos pacotes. A Figura 2.2 ilustra como os planos de controlo e de encaminhamento se relacionam.

A secção seguinte discute os vários protocolos de encaminhamento responsáveis por atualizar a tabela RIB. A partir desta última as tabelas Forwarding Information Base (FIB) são populadas com instruções associando prefixos de destinos a interfaces de rede do equipamento [33].

2.1.1 *Protocolos IGP*

Os protocolos IGP são utilizados para tomar decisões de encaminhamento de tráfego dentro de sistemas autónomos. Estes protocolos podem ser classificados de acordo com o conhecimento que possuem da rede no processo de decisão:

- Informação global - *Link-State*

Todos os *routers* possuem conhecimento relativo às ligações e respectivos custos de todos os outros nodos da rede do sistema autónomo.

- Informação descentralizada - *Distance-Vector*

  Os *routers* apenas possuem conhecimento relativo às ligações e respectivos custos dos vizinhos a que estão fisicamente ligados.

Protocolos Link-State

O protocolo Open Shortest Path First (OSPF) [28] é um protocolo de encaminhamento Link-State (LS). Cada *router* contém uma base de dados idêntica que descreve na totalidade a topologia do sistema autónomo.

Cada entrada dessa base de dados identifica o estado de um *router*, bem como informações sobre as suas interfaces e vizinhos alcançáveis. Esta informação é distribuída por todos os nodos dentro do sistema autónomo através de *flooding*.

A partir da base de dados, cada *router* constrói, com recurso ao algoritmo Dijkstra, a árvore dos caminhos mais curtos entre ele próprio (origem) e os restantes *routers* (destino). Quando existe mais do que uma rota para um mesmo destino com um mesmo custo, o tráfego é distribuído de maneira uniforme entre elas.

Um outro protocolo de encaminhamento, também classificado como LS, é o protocolo Intermediate System to Intermediate System (IS-IS) [8]. Tal como o OSPF, utiliza o algoritmo Dijkstra para calcular os caminhos mais curtos entre os diferentes nodos da rede, que podem estar ou não dentro da mesma área.

Protocolos Distance-Vector

Como já referido, os protocolos Distance-Vector (DV), como por exemplo o Routing Information Protocol (RIP) [24] possuem a informação descentralizada. Estes recorrem ao algoritmo de Bellman-Ford para tomar decisões de encaminhamento. Apesar destes protocolos serem considerados obsoletos, quando comparados com protocolos IGP mais recentes, possuem cabeçalhos de menor dimensão, o que contribui para um menor consumo de largura de banda em redes mais pequenas.

Comparação entre protocolos com Algoritmos Link-State e Distance-Vector

- Complexidade das mensagens trocadas - Os protocolos que se baseiam num algoritmo LS são mais complexos. Se o custo de uma das ligações variar, implica enviar o novo custo a todos os nós e recalcular. Os protocolos que se baseiam num algoritmo DV,

apenas divulgam alterações se e só se o novo custo puder contribuir para um novo caminho de menor custo para os nodos envolvidos.

- Velocidade de convergência - O algoritmo LS é mais rápido a convergir que o algoritmo DV (este pode convergir lentamente se ocorrerem certos ciclos de encaminhamento) sendo necessário a implementação de técnicas para o impedir.

- Robustez - O algoritmo LS é mais robusto do que o DV na medida em que permite um maior isolamento face a falhas na rede.

Apesar dos protocolos LS serem mais estáveis e robustos, estes são menos escaláveis. Isto deve-se à necessidade de cada nodo ter o conhecimento total da rede, implicando um grande poder de computação aliado a grandes capacidades de armazenamento. Por esse motivo, a utilização deste tipo de protocolos implica que a rede seja dividida em áreas.

2.1.2 *Protocolo BGP*

O BGP [36] é um protocolo que implementa políticas de encaminhamento entre diferentes sistemas autónomos, que usa TCP como protocolo de transporte. A principal função de um sistema BGP é trocar informação relativa a acessibilidade com outros sistemas BGP. Com isto é possível construir um grafo que representa a conectividade entre sistemas autónomos. Este protocolo possui um algoritmo *Path-Vector* que deriva do algoritmo DV. A métrica para determinar a qualidade das rotas pode ser baseada no número de saltos, sendo que cada nodo é representado pelo número do sistema autónomo. No entanto, no BGP, a escolha do melhor caminho pode depender de inúmeros outros parâmetros, bem como de políticas de encaminhamento definidas pelo administrador.

Uma sessão BGP refere-se à adjacência entre dois *routers* BGP. Estas sessões são sempre ponto a ponto, sendo categorizadas em dois tipos:

- External Border Gateway Protocol (eBGP) - Nodos vizinhos estão em diferentes sistemas autónomos. Os sistemas autónomos possuem múltiplos *routers* eBGP que efetuam a interligação entre os sistemas vizinhos. Estes aprendem assim múltiplas rotas para os mesmos destinos a partir dos seus vizinhos.

- Internal Border Gateway Protocol (iBGP) - Nodos vizinhos estão no mesmo sistema autónomo. Surge para colmatar a existência de múltiplas políticas de *routing* dentro do mesmo sistema autónomo, servindo também para redistribuir as rotas aprendidas pelo eBGP.

Na Figura 2.3 é possível ver um caso de aplicabilidade do protocolo iBGP, na medida em que, dentro do sistema autónomo AS65200 existem dois protocolos de encaminhamento.

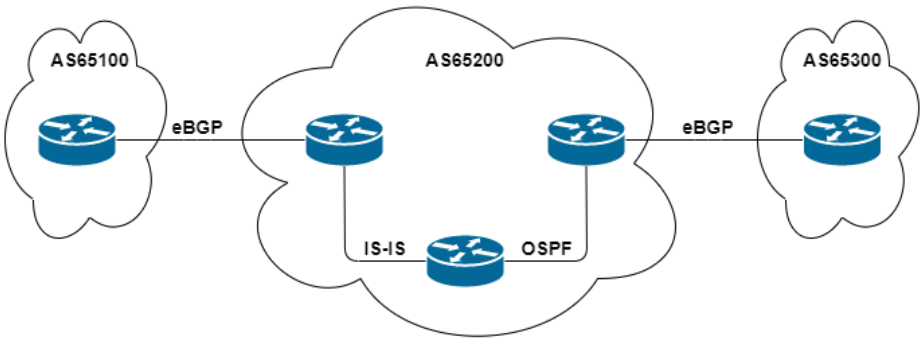


Figura 2.3: Exemplo de aplicabilidade de iBGP (Adaptado de [20]).

A comunicação dentro do AS65200 é efetuada via protocolos iBGP. Entre os sistemas AS65100, AS65200 e AS65300 a comunicação é efetuada via eBGP.

2.2 INTRODUÇÃO AO LABEL SWITCHING

Com o aparecimento de novos tipos de serviços e posterior crescimento das redes, surgiu a necessidade de optimizar o processo de encaminhamento dos pacotes com o objetivo de ultrapassar as limitações associadas ao encaminhamento IP [33]. No encaminhamento IP, cada *router* efetua decisões independentemente dos restantes no que toca à operação de encaminhamento, sendo essas mesmas decisões baseadas no endereço destino do pacote. Isto implica que cada nodo tenha de verificar na tabela de encaminhamento qual o destino de cada pacote. Este processo, tendo em conta a expansão da rede e a sua crescente utilização, verifica-se prejudicial em termos de desempenho. Para além disso, regras de encaminhamento baseadas apenas em caminhos mais curtos são pouco flexíveis. Dessa forma podem não ser suficientes para dar resposta às necessidades de encaminhamento. Nesse sentido e com o objetivo de superar as limitações do encaminhamento IP, foi desenvolvido o Multi Protocol Label Switching (MPLS). Este opera entre as camadas 2 e 3 da *stack* protocolar e tem como finalidade agilizar a escolha do próximo salto dentro do domínio MPLS.

2.2.1 *Multi Protocol Label Switching*

A otimização obtida ao nível da velocidade de comutação dos pacotes é alcançada usando a composição de duas funções. Uma que particiona o conjunto dos pacotes, que dão entrada no *router* de acesso ao domínio MPLS, em classes de equivalência de encaminhamento denominadas Forwarding Equivalence Class (FEC), sendo este processo apenas

executado uma vez. A segunda efetua o mapeamento de cada FEC para o próximo salto. Uma FEC é um grupo de pacotes IP que segue um mesmo caminho, denominado Label Switching Path (LSP) [45]. A FEC é identificada por uma etiqueta. Esta é usada como índice para uma tabela que especifica o próximo salto e uma nova etiqueta, identificada pelo campo *Label*. A etiqueta anterior é substituída pela mais recente e o pacote é encaminhado. Com isto é possível evitar a análise do cabeçalho da camada de rede de cada pacote a cada salto, visto que todo o encaminhamento é efetuado com recurso às etiquetas. A Figura 2.4 representa a estrutura do cabeçalho do protocolo MPLS. Como foi referido, a etiqueta identifica a FEC ao qual o pacote pertence. Para além da etiqueta existe ainda o Time to live (TTL), que é usado tal como o equivalente do cabeçalho IPv4, para garantir proteção contra ciclos no encaminhamento que possam surgir devido a falha ou até devido a uma convergência lenta por parte do algoritmo de encaminhamento. O campo experimental (Exp) é atualmente denominado de *traffic class* (TC), sendo através do mesmo possível atribuir prioridades aos pacotes. Por fim, o campo S indica se o pacote correspondente é o último pertencente a uma dada FEC.

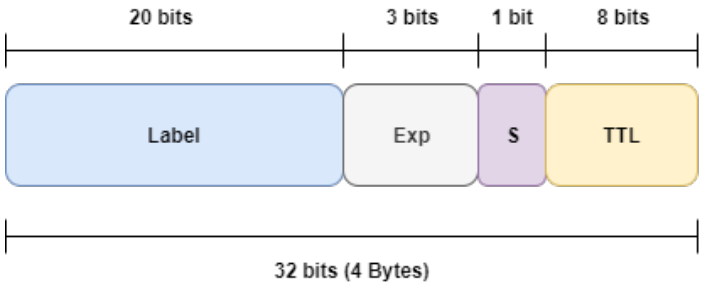


Figura 2.4: Cabeçalho do MPLS.

A topologia inerente à aplicação do protocolo MPLS inclui dois tipos de routers.

- Label Edge Router (LER)- *Router* que se encontra na fronteira da topologia, dando acesso ao *core* da mesma. Tem como principal função efetuar encaminhamento aos pacotes referentes a tráfego MPLS após estabelecer os LSPs.
- Label Switch Router (LSR) - *Router* que tem como objetivo efetuar a comutação de etiquetas, encaminhando o pacote para o próximo salto, ainda dentro do domínio MPLS.

Como é demonstrado na Figura 2.5, os pacotes dão entrada no domínio MPLS através de um *router* LER. Após efetuada a atribuição do pacote a uma FEC e introduzida uma etiqueta, este é enviado para o nodo seguinte definido pelo LSP. Quando chega ao *router* destino, dentro do domínio MPLS, a etiqueta é removida e o pacote segue até ao seu destino, utilizando encaminhamento IP.

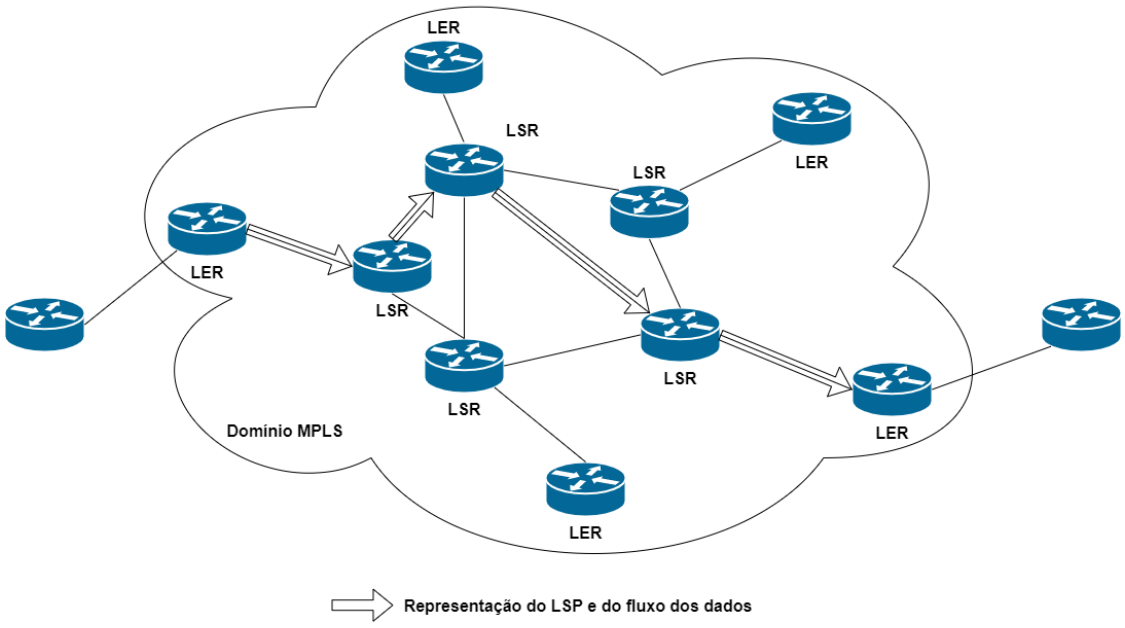


Figura 2.5: Exemplo do encaminhamento de tráfego numa topologia MPLS.

A informação proveniente dos protocolos de encaminhamento é usada para distribuir as etiquetas dentro do domínio MPLS. Estas podem ser distribuídas pelos diversos nodos da rede MPLS através do protocolo Label Distribution Protocol (LDP). Por sua vez, este baseia-se nos protocolos IGP para definir os LSP.

2.2.2 *Limitações do Multi Protocol Label Switching*

O uso do protocolo MPLS permite definir, em cenários extremos, caminhos para cada pacote. Esta capacidade garante ao MPLS uma grande flexibilidade o que o torna indicado para problemas relacionados com gestão e otimização de recursos na rede. No entanto, e ao contrário de outros protocolos IGP, a tecnologia MPLS não se encontra testada nem implementada a grande escala. Para além disso, a sua implementação apresenta dificuldades acrescidas ao contrário do OSPF que é um dos protocolos mais utilizados [16].

Outro dos grandes problemas associados ao MPLS, consiste na necessidade de manter o estado do LSP nos *routers* LSR. Esse mesmo estado é de elevada importância para o plano de controlo e plano de encaminhamento visto que, sem ele não é possível reconhecer as etiquetas associadas ao LSP. Com o aumento do número de *routers* LSR, manter o estado do LSP torna-se num problema de escalabilidade pois a quantidade de memória necessária para armazenar o mesmo é superior. Ainda no contexto de problemas de escalabilidade,

devido ao crescimento da rede, o aumento do número de LSPs a passar nos LSR pode prejudicar a velocidade de processamento, que por sua vez é limitada nos *routers*. Como tal, o uso deste protocolo poderá estar limitado a um número máximo de LSP [14].

2.3 SEGMENT ROUTING

O Segment Routing (SR) é uma implementação de Software Defined Networking (SDN) proposta pelo grupo SPRING do IETF, tendo como objetivo melhorar a seleção de rotas em redes IP. Nesta tecnologia, a rota é dividida em segmentos procurando atingir uma melhor utilização da rede [5].

O SR é baseado em *Source Routing* [34]. Este conceito define-se como a habilidade de um nodo origem especificar o caminho que o pacote irá percorrer até chegar ao destino, sem que este seja necessariamente baseado no caminho mais curto definido pelos protocolos de encaminhamento IGP.

Assim sendo, um nodo do domínio SR insere no cabeçalho dos pacotes uma lista ordenada de instruções. Estas permitem que o pacote siga um caminho pré-definido até chegar ao nodo destino. Essa lista de instruções corresponde a uma lista de segmentos, cada um identificado por um Segment Identifier (SID). Um segmento, para além de poder especificar um nodo que o pacote tem de visitar, pode ainda ser utilizado para efetuar operações no mesmo [43].

A implementação da arquitectura SR pode ser efetuada tanto no MPLS (SR-MPLS) como no protocolo IPv6 (SRv6). A implementação da mesma é possível sem que seja necessário efetuar alterações significativas ao plano de encaminhamento para ambos os casos. No que toca ao plano de controlo, a arquitectura SR suporta um cenário distribuído. Neste, os segmentos são alocados e sinalizados pelos protocolos de encaminhamento. Cada nodo efetua a decisão relativa à atribuição de cada pacote a uma determinada lista de segmentos [15].

2.3.1 *Descrição do Funcionamento do Segment Routing*

O cabeçalho SR contém a lista de segmentos e um apontador para o segmento ativo [15]. No plano de encaminhamento do SR, a operação a efetuar no pacote corresponde a uma das seguintes:

- *Next* - Coloca o segmento seguinte da lista como segmento ativo. Esta operação é utilizada quando o pacote dá entrada num dos nodos pertencentes ao caminho pré-definido. É inclusive utilizada no último nodo do domínio SR, a partir do qual o encaminhamento é efetuado por IP.

- *Continue* - Ação de encaminhamento efetuada no segmento ativo. A operação é utilizada nos nodos que se encontram entre dois nodos estabelecidos no caminho pré-definido.

- *Push* - Operação que corresponde à inserção de um segmento, à cabeça do cabeçalho SR do pacote. Esse mesmo segmento é colocado como sendo segmento ativo.

Para demonstrar a aplicação das operações referidas, considere-se a seguinte lista de segmentos C = [R2, R7, R8] que pretende dirigir os pacotes pelos nodos com o SID R2, R7 e R8 pertencentes a um domínio SR. Na Figura 2.6 é demonstrado um domínio SR e as operações que são efetuadas em cada nodo do caminho pelo qual os pacotes vão seguir, incluindo o estado da lista de segmentos.

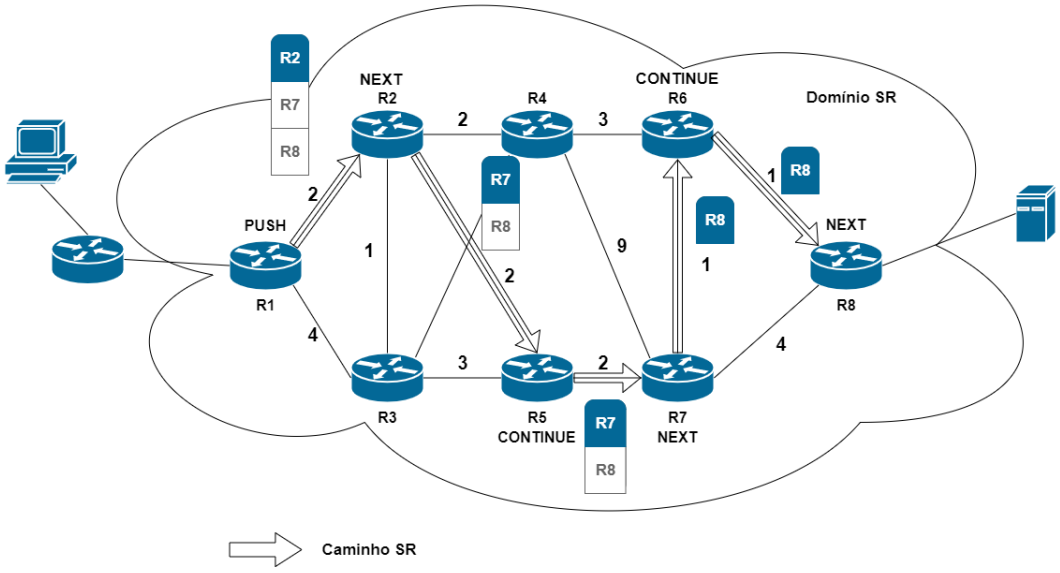


Figura 2.6: Exemplo de encaminhamento em SR (Adaptado de [43]).

Os segmentos são inseridos em R1, quando os pacotes dão entrada no domínio SR, através da operação *Push*. No nodo R2 e no nodo R7, através da operação *Next* é alterado o segmento ativo. Entre os nodos R2 e R7, R7 e R8, os pacotes são encaminhados com recurso a protocolos IGP, sendo nos nodos intermédios R5 e R6 utilizada a operação *Continue*. Existe ainda a possibilidade de a topologia conter múltiplos caminhos entre o mesmo par origem destino com o mesmo custo. Para este caso, o SR é capaz de explorar as vantagens do uso do algoritmo Equal Cost Multi Path (ECMP), dentro do seu próprio domínio. Em R8, os pacotes saem do domínio SR e são encaminhados com base nos protocolos IGP.

Os segmentos podem ser classificados como sendo locais ou globais. Os segmentos globais são instruções que podem ser executadas em todos os nodos do domínio SR [33]. Os

segmentos globais podem ser usados com o objetivo de encaminhar um pacote para um destino IP, quando um protocolo IGP é usado dentro do domínio SR. Estas instruções são denominadas *IGP-prefix segment* ou *Prefix-SID*.

Os segmentos locais apenas podem ser interpretados pelo nodo a que dizem respeito. Pode-se considerar o exemplo do encaminhamento de um pacote para um nodo adjacente. Neste caso, a instrução assume o nome de *IGP-Adjacency Segment* ou *Adjacency SID*[1] [43].

Como referido, a arquitetura SR pode ser aplicada como uma extensão aos protocolos MPLS e IPv6, visto que, foi desenvolvida com esse intuito. Na Tabela 2.1 é possível verificar as operações semelhantes, quando o *Segment Routing* é aplicado no plano de encaminhamento do MPLS e do IPv6.

SR	SR-MPLS	SRv6
SR Policy	*Label Stack*	Lista de Segmentos de endereços IPv6 no cabeçalho do SR
Segmento Ativo	Elemento no topo da *stack*	Endereço IPv6 indicado no campo *Segment Left*
Operação *Push*	*Label Push*	Adiciona um endereço IPv6 na lista de segmentos no cabeçalho SR
Operação *Next*	*Label Pop*	Decrementa o campo *Segment Left* Copia o Segmento Ativo para o endereço de destino IPv6
Operação *Continue*	*Label Swap*	Efetua encaminhamento de acordo com o endereço de destino IPv6

Tabela 2.1: Operações do SR aplicadas em MPLS e IPv6.

O uso de SR em IPv6 implica a criação de um novo tipo de cabeçalho denominado *SR Header* (SRH). Este contém uma lista ordenada de endereços IPv6 que representa a lista de segmentos em SR, sendo que cada endereço é um SID. O uso e desenvolvimento do SRv6 é de elevada importância para a implementação de serviços mais complexos como *Service Function Chaining* [43].

2.3.2 *Ojetivos de Otimização na Tecnologia de Segment Routing*

A capacidade do SR dar flexibilidade na escolha de rotas e por sua vez encaminhar e distribuir o tráfego pelas mesmas permite atingir uma melhor utilização dos recursos. Isto deve-se ao facto dos protocolos de encaminhamento tradicionais tomarem decisões de encaminhamento com base no caminho mais curto ou de menor custo. O custo de uma ligação entre dois nodos da rede é tradicionalmente baseada no inverso da capacidade da mesma. Assim, as ligações com maior capacidade possuem um custo menor pelo que, ao

1 Um *Adjacency Segment* pode ser global

executar os algoritmos, essa ligação será a mais utilizada por parte dos protocolos de encaminhamento convencionais. Isto pode levar a casos de congestão, visto que todo o tráfego é encaminhado pelas mesmas ligações, deixando outras com uma menor capacidade, porém sem congestão, por utilizar [33].

Alguns trabalhos de investigação realizados na área como [26], referenciam como métrica de optimização o Maximum Link Utilization (MLU). No entanto, a minimização do MLU apresenta algumas limitações, como por exemplo, a escolha de rotas mais longas para reservar largura de banda para rotas mais curtas caso picos de tráfego ocorram, o que pode levar a uma degradação do desempenho de algumas aplicações. Para além disso, a minimização não tem em conta a quantidade de ligações que estão perto do valor MLU, não garantindo uma boa distribuição do tráfego pela infraestrutura da rede. Enquanto uma ou mais ligações estão perto do MLU, outras podem não estar a ser utilizadas. Isto pode levar à existência de *bottlenecks*, que comprometem ainda mais o valor do MLU, visto que este poderá permanecer inalterado face a alterações nas necessidades de tráfego.

Fortz e Thorup em [16] propõe uma função que tem em conta a utilização de todas as ligações da rede. As necessidades de tráfego são representadas por uma matriz D que para cada par origem (s) e destino (t), indica a quantidade de tráfego a enviar de s para t. A capacidade de um arco a é dada por $c(a)$. O fluxo de um arco $l(a)$ é obtido a partir da soma de todos fluxos que passam por a. A utilização de um arco é dada pela equação $l(a)/c(a)$. Assim sendo, a função Φ definida em 2.1, soma o custo de utilização de todos os arcos.

$$\Phi = \sum_{a \epsilon A} \Phi_a(l(a)) \tag{2.1}$$

Quando o arco não é utilizado, não são aplicadas penalizações. Assim, para todo arco $a \epsilon A, \Phi_a(0) = 0$. Nesse sentido, a função $\Phi'_a(l(a))$ definida em 2.2, é a derivada de $\Phi_a(l(a))$.

$$\Phi'_a(l(a)) = \begin{cases} 1, & 0 < x/c(a) < \frac{1}{3} \\ 3, & \frac{1}{3} \leq x/c(a) < \frac{2}{3} \\ 10, & \frac{2}{3} \leq x/c(a) < \frac{9}{10} \\ 70, & \frac{9}{10} \leq x/c(a) < 1 \\ 500, & 1 \leq x/c(a) < \frac{11}{10} \\ 5000, & x/c(a) \geq \frac{11}{10} \end{cases} \tag{2.2}$$

O gráfico resultante da função $\Phi_a(l(a))$ é demonstrado na Figura 2.7, sendo que o custo de cada arco está relacionado com a utilização do mesmo.

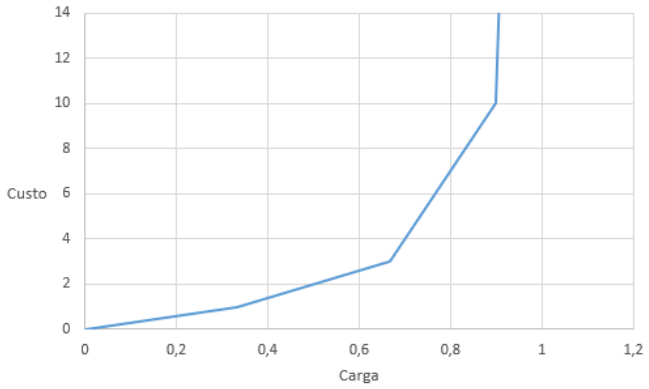


Figura 2.7: Custo do arco $\Phi a(l(a))$ como função de carga $l(a)$ para um arco de capacidade $c(a) = 1$ (Adaptado de [16]).

A normalização é introduzida pelos autores, Fortz e Thorup, com o objetivo de permitir efetuar comparações de custos entre topologias de diferentes dimensões e capacidades das ligações. Para isso são introduzidas um conjunto de equações com o objetivo de normalizar os valores obtidos. Assim, seja $D(s,t)$ as necessidades de encaminhamento entre um par origem destino (s,t) e $dist_1(s,t)$ a distância entre os mesmos com base no número de saltos. Esta última formulação permite obter um fator de normalização apresentado na Equação 2.3.

$$\Phi_{Uncap} = \sum_{(s,t)\epsilon N*N} (D(s,t) * dist_1(s,t)) \tag{2.3}$$

O custo normalizado Φ^* é obtido a partir da Equação 2.4.

$$\Phi^* = \frac{\Phi}{\Phi_{Uncap}} \tag{2.4}$$

Quando $\Phi^* = 1$, significa que todas as ligações da topologia estão abaixo de $1/3$ da capacidade. Em contrapartida, quando a carga de todos as ligações é igual ao limite da capacidade das mesmas, $\Phi^* = 10 - 16/3 = 1.(6)$. Depois deste valor, Φ^* aumenta rapidamente até 5000. Nesse sentido, é possível afirmar que uma topologia encontra-se congestionada se $\Phi^* \geq 1.(6)$.

A utilização da função custo Φ permite ultrapassar algumas das limitações da métrica MLU mencionadas. Apesar disso, a optimização dos pesos IGP de modo a minimizar esta métrica com base em necessidades de encaminhamento, é considerado um problema NP-Difícil. Como consequência, não pode ser resolvido apenas com recurso a técnicas como programação linear.

2.3.3 *Engenharia de Tráfego com Segment Routing*

Engenharia de tráfego define-se como a área que estuda a utilização de forma eficiente dos recursos da rede. Dado que o SR é capaz de proporcionar uma elevada flexibilidade na seleção de rotas para efetuar encaminhamento de pacotes, é muito utilizado para solucionar problemas de engenharia de tráfego.

Um dos principais problemas que podem ser resolvidos com recurso a engenharia de tráfego é a otimização do nível de congestão da rede. No entanto, com o uso de SR, existe um aumento do consumo da largura de banda devido à inserção de segmentos nos cabeçalhos dos pacotes bem como uma limitação no número de *labels* imposto pelos equipamentos.

O trabalho desenvolvido em [10], investiga a implementação de SR numa rede IP. Os autores indicam que o processo de implementação deve ser feito de forma incremental através de extensões aos *routers* IP. O domínio SR (SRD) é definido com recurso a um subconjunto de nodos capazes de suportar o mesmo. Com isto, são propostos dois modelos: *Single-SRD* (S-SRD) e *Multiple-SRD* (M-SRD). No caso do S-SRD, dado tratar-se de apenas um domínio SR, o comprimento médio da lista de segmentos é reduzida, o que se traduz numa diminuição do consumo da largura de banda. O M-SRD possui uma maior flexibilidade na definição de rotas podendo as mesmas ser mais complexas. A gestão do número de nodos e domínios capazes de suportar SR é um problema associado a engenharia de tráfego que os autores tentam solucionar através de técnicas de programação linear, tendo como principal objetivo reduzir o valor do MLU.

O problema de engenharia de tráfego é também explorado pelos autores em [41] com a finalidade de reduzir o valor do MLU. A complexidade da definição dos caminhos tendo em conta necessidades de tráfego é considerado pelos autores um problema NP-Difícil, podendo apenas ser resolvido em tempo polinomial caso o número de segmentos a ser considerado seja fixo.

Em [5], os autores propõe uma solução baseada em SR para dar resposta à alocação de necessidades de encaminhamento através de ECMP. A solução encontrada visa a utilização de SR com 2 segmentos, ou seja, assumem a existência de apenas um nodo intermédio entre os nodos de saída e entrada. Os autores consideraram que a utilização de 2 segmentos exige um maior poder de computação, no entanto permite minimizar a congestão na rede. São considerados três cenários para o problema descrito: cenário *offline* com e sem conhecimento do fluxo de tráfego, cujo objetivo é minimizar o pior caso da utilização das ligações, e um cenário *online* cujos valores de tráfego podem sofrer alterações, tendo como objetivo a redução do número de pedidos rejeitados. Os autores chegam à conclusão que a utilização de 2 segmentos consegue aproximar-se da solução que utiliza um número indeterminado de segmentos, otimizando o valor do MLU.

Em [31], os autores propõe uma solução que, na sua configuração, pode utilizar até três segmentos, afirmando que é possível atingir uma solução quase ótima dos recursos disponíveis. Para tal, a proposta recorre à otimização do IGP sendo que, simultaneamente, uma pequena percentagem dos fluxos é desviada no máximo um salto do caminho mais curto. Esta pequena alteração de rotas, conseguida com recurso a *Adjacency Segments*, introduz um ganho significativo na utilização dos recursos. A otimização é feita numa *framework* capaz de simular uma rede *SR/Link-State*, utilizando algoritmos evolucionários. Os dados de tráfego são recolhidos através de um controlador Path Computer Element (PCE)/SDN que recolhe informações da topologia e tráfego. O tráfego é posteriormente distribuído com recurso a extensões das funções de balanceamento de tráfego dos algoritmos Distributed Exponentially-weighted Flow SpliTting (DEFT) [49] e Penalizing Exponential Flow-spliTting (PEFT) [50]. Mais recentemente, os mesmos autores propuseram a utilização de programação linear, combinada com algoritmos evolucionários, para otimizar o balanceamento de tráfego entre caminhos SR paralelos [32]. Os resultados obtidos mostram, novamente, que o uso de SR permite atingir configurações que distribuem de forma quase ótima o tráfego através dos recursos disponíveis, conseguindo desempenhos melhores do que aqueles obtidos por [5].

Por fim, os autores em [22] efetuam um estudo relacionado com engenharia de tráfego numa SDN. O objetivo passa por otimizar mais uma vez o desempenho da rede e a alocação de recursos. Os autores utilizam um controlador SDN com recurso a SR, que apenas necessita de incluir no cabeçalho de cada pacote, o caminho desde a origem até ao destino pretendido. Como tal, propõe um algoritmo que permita estabelecer um caminho entre dois nodos com garantias a nível de largura de banda no que toca a uma determinada necessidade de tráfego. É ainda considerado no algoritmo o *overhead* resultante das etiquetas inseridas nos cabeçalhos dos pacotes.

Em suma, os trabalhos realizados na área abordam a complexidade de criação de rotas associadas ao número de segmentos a inserir no cabeçalho dos pacotes. Com o aumento da complexidade da rede, a possibilidade de resolver os problemas de engenharia de tráfego através de métodos como programação linear torna-se inviável visto que os mesmos são NP-Difíceis. Apesar disso, a introdução de SR nas redes IP pode levar a uma melhoria significativa no que toca ao nível de utilização dos recursos.

2.4 NETWORK FUNCTION VIRTUALIZATION

As redes de computadores tradicionais estão altamente dependentes de diversos equipamentos. Com isso, o lançamento de um novo serviço pode implicar a introdução de mais equipamentos na rede o que pode ser altamente dispendioso e complexo. Para além disso, a introdução de novos serviços pode levar ao desenvolvimento de novos protocolos,

cuja implementação é normalmente demorada. Um dos setores que tem sofrido avanços tecnológicos constantes, é o das telecomunicações. Com o objetivo de diminuir os custos de implementação de novos serviços, os operadores de telecomunicações têm procurado soluções que permitam utilizar *software* para implementar os serviços em equipamentos de rede Commercial-Off-The-Shelf (COTS). Nesse sentido, alguns dos maiores operadores de telecomunicações como a *American Telephone and Telegraph* (AT&T), *British Telecom* (BT) e *Deutsche Telekom* (DT), formaram um *Industry Specification Group* (ISG), pertencente ao European Telecommunications Standards Institute (ETSI) para desenvolver e definir o conceito de Network Function Virtualization (NFV) [51]. O uso de NFV procura reduzir o tempo gasto a desenvolver *hardware* específico e respetivos custos para novos serviços, permitindo inclusive que os mesmos sejam devidamente testados. Para além disso, permite aos operadores reservar recursos de acordo com as necessidades de encaminhamento de tráfego.

Com o surgimento de *software* na rede, a execução de funções sobre a mesma tornou-se uma realidade. No contexto do NFV, essas funções designam-se como Virtual Network Function (VNF). Um conjunto de VNFs pode representar um serviço disponibilizado por um operador, quando as mesmas são agrupadas ordenadamente (*VNF chaining*). O NFV permite ainda aos operadores configurar a alocação de recursos pelas diversas VNFs permitindo assim dar mais flexibilidade e gestão na utilização de recursos. Para além das VNFs, podem ainda existir funções implementadas em nodos da rede, sendo que não se encontram virtualizadas. Estas denominam-se Physical Network Function (PNF) e correspondem a um bloco de funções especializado para um determinado serviço [51]. Cada VNF possui um elemento de gestão, denominado Element Management (EM). Este é responsável pela gestão de cada VNF em termos de instanciação, instalação e execução [51].

2.4.1 *Arquitectura Network Function Virtualization*

A arquitectura NFV é composta por um plano de encaminhamento e um plano de controlo. O plano de encaminhamento é representado pela infraestrutura do NFV denominada Network Function Virtualization Infrastructure (NFVI) e o plano de controlo representado pela componente gestão e orquestração, também designada por MANO [4].

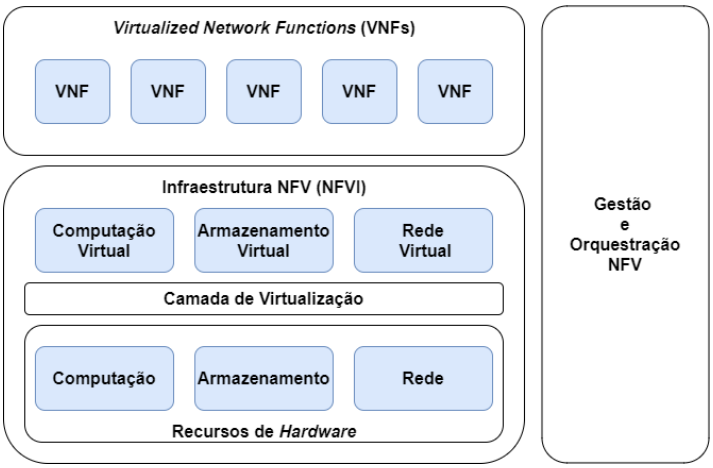


Figura 2.8: Arquitectura ETSI NFV *framework* (Baseado em [12]).

Este é responsável pela gestão do ciclo de vida dos recursos de *hardware* e *software* de todos os componentes da arquitetura NFV, que está representada na Figura 2.8. As funções são instanciadas através de um Network Point of Presence (N-PoP). Estes pontos indicam a localização da implementação das diversas VNFs ou PNFs.

Os N-PoP, representados na Figura 2.9, estão por sua vez situados na NFVI. Esta pode ser dividida em três camadas distintas: infraestrutura física, camada de virtualização e infraestrutura virtual. A infraestrutura física é constituída por servidores com capacidades de computação e armazenamento, podendo ainda comunicar com outros elementos da rede através de interfaces internas [51]. A camada de virtualização é uma plataforma que permite efetuar uma abstração entre os componentes de *hardware* e de *software* através, por exemplo, do uso de máquinas virtuais. Estas podem ser migradas para outros pontos da rede e sofrer alterações no que toca aos recursos que estão a ser utilizados em tempo real. Para isso, o hipervisor consegue ajustar dinamicamente os recursos que estão atribuídos às máquinas virtuais [51].

A infraestrutura virtual situa-se acima da camada de virtualização e é constituída pelos componentes: computação virtual, armazenamento virtual e rede virtual. Tal como o próprio nome indica, estes componentes são provenientes da virtualização dos componentes físicos.

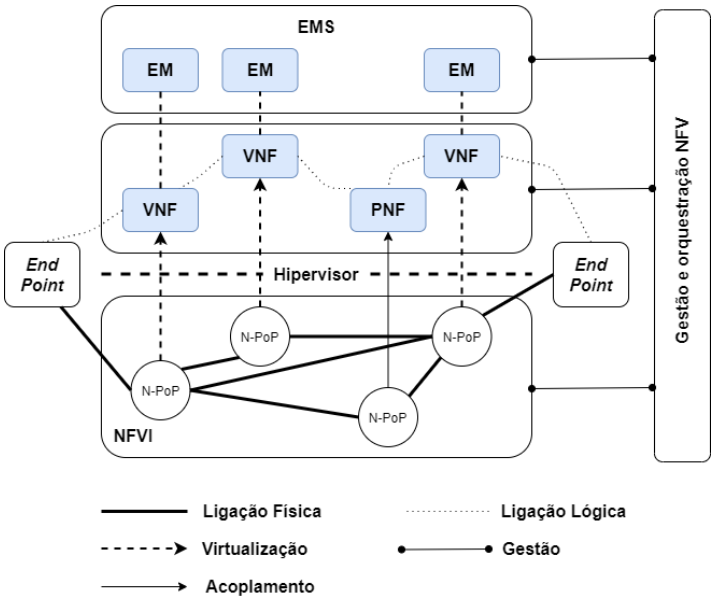


Figura 2.9: Relação entre terminologias do NFV (Baseado em [51]).

A infraestrutura NFVI suporta assim a execução de diversos VNFs através da virtualização de múltiplos servidores. A possibilidade de fornecer serviços através de *software* trouxe alguns desafios relacionados com o processo de localização de VNFs face a determinadas necessidades de encaminhamento. Este problema de localização é chamado de *VNF-Placement* (VNF-P) e é considerado NP-Difícil [3], o que traz novamente a possibilidade de aplicação de soluções baseadas em engenharia de tráfego.

2.4.2 *Network Function Virtualization em 5G*

O surgimento da tecnologia 5G traz benefícios em relação a anteriores tecnologias no que toca a volume de dados e velocidade no processamento de tráfego. Com o desenvolvimento de novos equipamentos capazes de suportar esta tecnologia, ocorreu um aumento na diversidade do tipo de serviços, aplicações e número de utilizadores. Nesse sentido, com o aparecimento de cada vez mais serviços e respectivas necessidades de encaminhamento, o uso de uma arquitectura NFV aliada a esta tecnologia emergente é de elevado interesse para os operadores de telecomunicações, que pretendem fazer uso de diversos serviços que estão espalhados na rede. O contexto das redes 5G encontra-se fortemente ligado a *Network Slicing*. Este define-se como a partição de uma rede física em múltiplas redes virtuais, sendo que, cada rede pode ser configurada e optimizada para um tipo de serviço, aplicação ou subscritor. No caso de uma rede 5G, uma *network slice* pode ser composta por um conjunto de *network functions* e as respetivas definições que são relativas a um *use case* específico [53].

2.5 MECANISMOS DE OTIMIZAÇÃO

Quando as topologias possuem uma complexidade elevada, a resolução dos problemas relacionados com engenharia de tráfego na alocação de recursos, de acordo com os requisitos de encaminhamento, é considerado NP-Difícil. Nesse sentido, estes problemas não podem ser resolvidos em tempo polinomial, inviabilizando a utilização de programação linear. Alguns trabalhos na área de otimização abordaram o problema de alocação de recursos através de algoritmos evolucionários e *machine learning*.

2.5.1 *Computação Evolucionária*

A computação evolucionária surge no contexto de optimização como meta-heurística capaz de dar soluções a problemas NP-Difíceis. É inspirada no trabalho de evolução natural proposto por *Darwin*. Este trabalho considera que uma ou mais populações de indivíduos sofre alterações ao longo do tempo, originando novas formas possíveis, *fitness landscape*. Essa nova forma é apresentada pelos indivíduos que melhor se adaptam às alterações [44]. Estes reproduzem-se passando as suas características, fenótipos, para as gerações seguintes. Estas são resultantes do genótipo que representa o conjunto de genes herdados. A partir deste processo é possível eliminar os elementos da população que não possuem os fenótipos que lhes permitam adaptar-se às alterações do ambiente envolvente. Este processo denomina-se seleção natural. Os algoritmos evolucionários assemelham-se a este processo evolutivo procurando manter um equilíbrio entre seleção e variação. A primeira representa a evolução no que toca a características específicas que se adequam a determinados cenários, reduzindo assim a diversidade genética da população. A segunda representa o surgimento de novas características através de operadores de recombinação e de mutação, aumentando a diversidade genética da população. A Figura 2.10 representa a estrutura de um algoritmo evolucionário. As diversas componentes do mesmo são descritas por [11] como sendo as seguintes:

- Representação - Especifica a estrutura de dados (genótipo) escolhida para codificar as soluções. Esta deverá ser adequada ao problema bem como à aplicação de operadores genéticos.
- Aptidão ou Função de avaliação - Representa a seleção das melhores características de uma população. Possui uma função objetivo associada que a cada genótipo (solução) associa um valor quantitativo de qualidade.
- *Representation-dependent*
  - Inicialização - Representa a criação de uma população com características aleatórias que possa representar um possível conjunto de soluções.

  - *Crossover* - Representa o cruzamento da informação de dois ou mais progenitores, gerando posteriormente novas soluções.
  - Mutação - Os operadores de mutação representam as transformações que são efetuadas a um indivíduo, permitindo assim aumentar a diversidade genética ao introduzir na população novas características.

- *Representation-independent*
  - Darwinismo Artificial - Representa diferentes processos de seleção e substituição de progenitores de forma artificial.
  - Caso de paragem - Representa a finalização do algoritmo. Visto que este por vezes pode ser muito complexo, o caso de paragem pode ser definido pelo tempo de execução, número de iterações ou pelo período de tempo que decorreu sem serem observadas novas alterações.

Os algoritmos evolucionários podem ser utilizados com uma ou mais funções objetivo. Nesse sentido é possível classificar os mesmos como sendo:

- *Single-Objective* - Visam a optimização de apenas uma função objetivo.
- *Multi-Objective* - Visam resolver problemas que envolvem mais do que uma função objetivo para optimização.

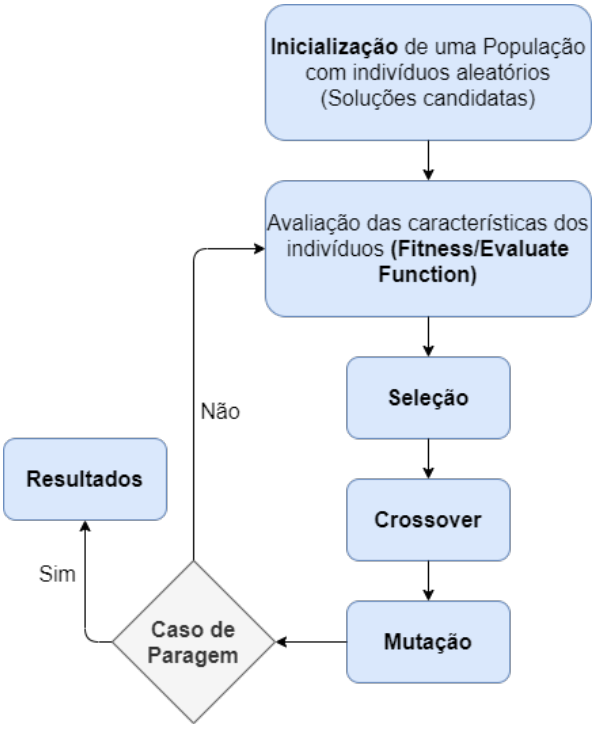


Figura 2.10: Estrutura de um algoritmo evolucionário (Baseado no algoritmo descrito por [11]).

Os algoritmos *multi-objective* oferecem a possibilidade de analisar o *trade-off* entre os diferentes objetivos a atingir, visto que, permitem a otimização de diferentes funções objetivo em simultâneo ao invés de as agregar numa só.

Em [30], os autores propõe o uso dos algoritmos *multi-objective Non-dominated Sorting Genetic Algorithm* (NSGA-II) [39], *Strength Pareto Evolutionary Algorithm* (SPEA2) [54] e o uso de um algoritmo *single objective* para solucionar problemas de engenharia de tráfego. O trabalho utiliza a função objetivo definida na subsecção 2.3.2 para optimizar a distribuição do tráfego em redes OSPF. Este trabalho procura, para uma ou mais matrizes Di, encontrar um conjunto de pesos w para minimizar as funções objetivo Φ_1^* e Φ_2^*. A primeira representa o nível de congestão da rede num modo normal de operação. A segunda representa o nível de congestão após efetuada uma alteração às condições operacionais da rede. Os resultados obtidos mostram que os algoritmos descritos permitem obter resultados com boa qualidade para topologias de pequena dimensão. Em redes de maior dimensão, o algoritmo (NSGA-II) destaca-se dos restantes, apresentando melhores resultados.

O uso de algoritmos evolucionários para solucionar problemas de engenharia de tráfego pode ser bastante útil, no entanto, estes apresentam algumas limitações relacionadas com o tempo de execução. Por norma, o critério de paragem considerado é precisamente o tempo de execução. Isso implica que por vezes, as soluções obtidas nesse espaço de tempo possam não ser as melhores.

2.5.2 *Machine Learning*

Machine Learning (ML) é uma tecnologia que utiliza algoritmos que possuem a capacidade de aprender e de fazer suposições com base em dados. O uso de ML tem vindo a aumentar devido à sua capacidade de se adaptar a diversas aplicações como deteção de fraude, reconhecimento de padrões, deteção de *spam* em *email*, entre outros. Em relação à técnica de otimização com base em algoritmos evolucionários, o uso de ML permite a obtenção de soluções num período de tempo menor.

As técnicas usadas em ML podem ser divididas em três tipos: *Supervised*, *Semi-Supervised* e *Unsupervised Learning*.

- ***Supervised Learning*** - Os algoritmos *Supervised Learning* requerem a existência de um *input*, sendo este composto por um conjunto de dados classificados organizados por colunas, e do *output* que é esperado para cada *input*. Após o treino do modelo estar concluído, é possível aplicar o conhecimento adquirido num novo conjunto de dados.

- ***Unsupervised Learning*** Este tipo de algoritmos é utilizado quando o *output* não é conhecido. Assim sendo, o efeito que cada variável de *input* poderá ter no resultado

final não é conhecido. O objetivo deste tipo de algoritmos passa por localizar pequenos grupos nos dados permitindo obter conhecimento relativos aos mesmos.

- ***Semi-Supervised Learning*** Os algoritmos *Semi-Supervised Learning* permitem treinar um modelo através de uma mistura de dados classificados e não classificados, sendo particularmente útil pois a maior parte dos dados disponíveis são não classificados [27].

Para resolver os problemas de optimização relacionados com a alocação de recursos da rede serão utilizadas redes neuronais. Estas são flexíveis na medida em que se adaptam a diversos cenários, podendo ser utilizadas como *supervised* e *unsupervised learning*. Foram criadas com base no cérebro humano, aproveitando a estrutura do mesmo, constituída por neurónios e as respectivas conexões.

Nas máquinas, uma rede neuronal consiste num conjunto de elementos de processamento. Estes estão normalmente organizados por uma sequência de camadas com conexões estabelecidas entre todos os elementos ou de forma aleatória. Normalmente, as redes neuronais são constituídas por três ou mais camadas. Uma camada de entrada de dados que são apresentados à rede através de um *input buffer*, uma camada de *output* com um *buffer* que agrega o *output* gerado pela camada de entrada de dados e uma ou mais camadas intermédias [42].

Na Figura 2.11 está presente uma estrutura de uma Deep Neural Network (DNN). Este tipo de rede neuronal é constituída por múltiplas camadas de nodos de processamento intermédias, treinada por algoritmos capazes de aprender através de um conjuntos de dados [38]. Este tipo de arquitetura permite a análise de funções não lineares mais complexas, que não podem ser resolvidas com redes neuronais com um número reduzido de camadas.

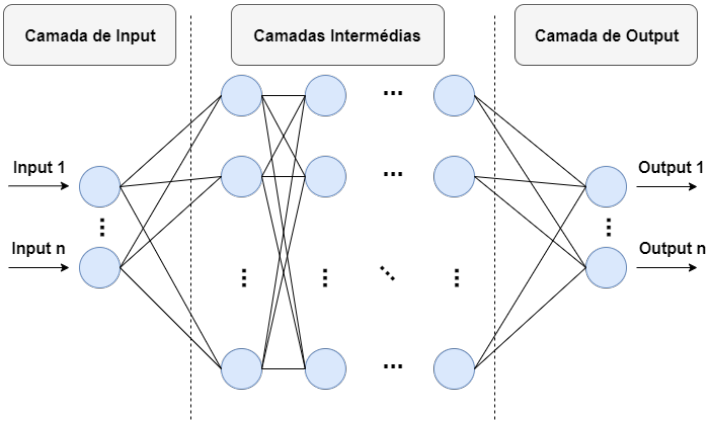


Figura 2.11: Estrutura de uma rede neuronal.

Os autores de [35] propõe o uso de uma solução baseada em redes neuronais para solucionar problemas de encaminhamento de fluxo de tráfego na rede. O trabalho utilizou a

técnica *Supervised Learning* para, com recurso a uma DNN, determinar o melhor caminho para cada fluxo de tráfego, assumindo como *input* um conjunto de *features* como a origem, destino e uma estimativa da largura de banda a ser utilizada. O *output* obtido consiste no custo a atribuir a cada ligação. Posteriormente, com recurso a programação linear, são obtidos os caminhos óptimos para cada fluxo de tráfego através das funções objetivo definidas na subsecção 2.3.2 que procuram minimizar o nível de congestão na rede.

O modelo Long Short Term Memory (LSTM) [17] é uma evolução de uma Recursive Neural Network (RNN). Este tipo de rede neuronal é adequado a conjuntos de dados estruturados ou com uma sequência temporal associada. Em [52], os autores propõe o uso de redes LSTM para auxiliar o processo de disponibilização de VNF's na rede com base no histórico de tráfego na rede. Neste trabalho são ainda comparados os resultados obtidos entre vários tipos de redes LSTM, DNNs e um modelo Support Vector Regression (SVR). Os autores obtiveram resultados ligeiramente melhores com recurso a DNNs, em detrimento das redes LSTM.

O modelo Support Vector Machine (SVM) é usado em diversos trabalhos relacionados com problemas de classificação com uma ou mais classes. O principal objetivo do modelo SVM consiste em separar os dados em várias classes através de um plano ou superfície que permita maximizar a capacidade de generalização do modelo [9]. Esta técnica de *machine learning* permite conjugar diferentes funções de *kernel* com o objetivo de encontrar a solução óptima [6]. Dentro das mais regularmente utilizadas, de acordo com [9], encontram-se: *Linear*, *Polynomial* e Gaussian Radial Basis Function (RBF). Este modelo é regularmente utilizado em problemas de classificação de tráfego.

Por fim, o modelo Random Forest (RF) [7] consiste num conjunto de árvores de decisão nas quais, uma classe é definida conforme o número de votos obtidos entre elas. Nesse sentido, a classe que tiver mais ocorrências em comparação com as restantes é a escolhida. As vantagens deste modelo, face a árvores de decisão, são descritas em [1] como as seguintes:

- Redução de problemas associados a *overfitting*.
- Menor sensibilidade com *outliers*.
- Necessidade de *pruning* descartada.
- Métricas de desempenho geradas automaticamente.

Tal como o modelo SVM, o modelo RF é regularmente utilizado em problemas de classificação de tráfego.

2.6 FRAMEWORK NETOPT

A *framework* NetOpt [32], utilizada no contexto desta dissertação, é uma ferramenta *open-source* implementada em Java, que foi desenvolvida com o objetivo de auxiliar os administradores de redes de computadores na configuração das mesmas. Esta tem em consideração o trabalho de Fortz e Thorup relativo à minimização do nível de congestão em redes com os protocolos de encaminhamento OSPF, IS-IS, incluindo redes SDN e SR. A *framework* permite a utilização de topologias no formato GraphML ou GML provenientes do Topology Zoo [21]. É também capaz de utilizar topologias aleatórias obtidas a partir do gerador Brite [25], sendo possível alterar os atributos das ligações das topologias como os pesos, os valores de *delay* e de capacidade através da componente gráfica da mesma.

A *framework* utiliza algoritmos de computação evolucionária, bem como programação linear, para resolver problemas de configuração de redes, falha de ligações, e variação de tráfego, com um ou mais objetivos [33]. O NetOpt disponibiliza várias métricas para avaliar configurações, como o MLU ou a função de congestão de Fortz, mas também a latência. Na Figura 2.12 está representada a arquitetura da *framework* com todos os seus componentes. Estes podem ser sub-divididos nos seguintes pontos:

- NetOptAIBench - permite interagir com a *framework* via interface gráfica ou via terminal. Através desta é possível ver em tempo real o estado das topologias e alterar a configuração das mesmas.

- NetOpt - responsável pelas simulações e configuração das mesmas. Como argumentos são considerados diversos ficheiros que permitem representar topologias, requisitos de encaminhamento e atrasos na rede. Através destes é possível simular situações de encaminhamento tirando partido de balanceadores de tráfego e estratégias de recuperação de falhas.

- NetOptJEcoli - consiste numa versão alterada da biblioteca Jecoli [13], tendo por finalidade resolver problemas de minimização e maximização. É responsável por providenciar implementações de algoritmos evolucionários *single* e *multi-objective* que permitem dar resposta a problemas de congestão da rede e de balanceamento de tráfego.

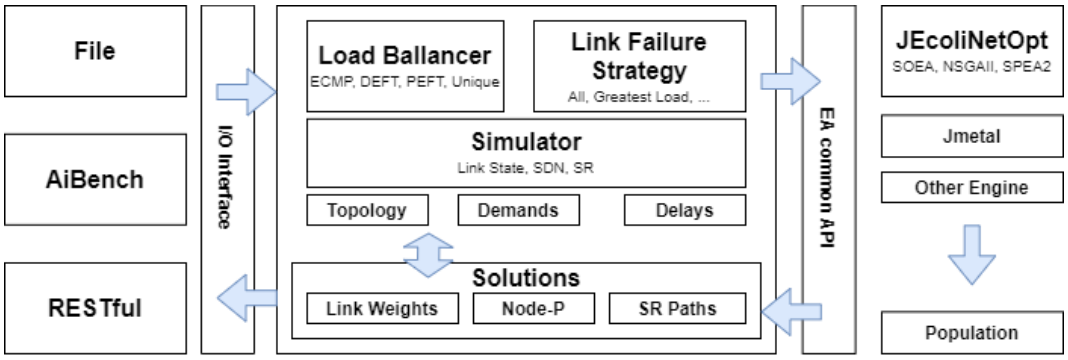


Figura 2.12: Arquitetura da *framework* NetOpt (Retirado de [33]).

O simulador é baseado no protocolo de encaminhamento OSPF e permite a integração de conceitos como SR e redes SDN. Assim, dado uma configuração de pesos, matrizes de necessidades de encaminhamento e possíveis atrasos nas ligações da topologia, o simulador distribui o tráfego entre os nodos da topologia com recurso ao algoritmo Dijkstra [33].

2.7 Sumário

Neste capítulo foram abordados os diversos conceitos essenciais ao trabalho desenvolvido, nos quais se inserem os protocolos de encaminhamento dinâmico e a introdução a novos tipos de encaminhamento baseado em etiquetas como o MPLS e o SR. Foi ainda abordado o conceito de NFV e de que forma é que este se associa às redes 5G possibilitando a introdução de novos serviços na rede. Foram ainda abordados trabalhos relacionados com engenharia de tráfego e métodos de otimização. Por último foi apresentada a arquitetura da *framework* NetOpt que servirá de base para o desenvolvimento desta dissertação visando dar resposta aos problemas identificados nas áreas de gestão de recursos.

3

ESPECIFICAÇÃO E MODELAÇÃO DO PROBLEMA

Neste capítulo são descritas as abordagens utilizadas que permitem dar resposta à problemática identificada no Capítulo 2. Como tal é efetuada uma especificação do problema e explicado de que forma este é endereçado nesta dissertação (secção 3.1). De seguida são apresentados dois modelos lineares capazes de avaliar os níveis de utilização de uma topologia, nomeadamente das ligações e nodos da mesma (secção 3.2). As secções seguintes apresentam vários métodos que combinam os modelos lineares com algoritmos evolucionários: Na Secção 3.3 é definido um método para seleção de nodos de execução de serviços enquanto na Secção 3.4 é apresentada uma estratégia para otimizar os pesos IGP. Para finalizar, na Secção 3.5 é explicado o processo de construção de um conjunto de dados para o treino de modelos de ML supervisionados.

3.1 ESPECIFICAÇÃO DO PROBLEMA

A utilização de redes SDN em conjunto com a arquitetura NFV foi a solução encontrada pelos operadores de telecomunicações para dar resposta às crescentes necessidades de encaminhamento e às constantes introduções de novos serviços na rede. Assim, o problema ao qual esta dissertação procura dar resposta, diz respeito à otimização dos recursos disponíveis por parte dos operadores. Destes fazem parte os serviços que pretendem disponibilizar, cuja tendência será sofrerem um aumento da sua utilização devido à introdução do 5G. De forma a garantir uma utilização eficiente dos recursos, mantendo os níveis de qualidade de serviço, é necessário averiguar a distribuição ótima dos serviços na rede. No entanto, tal como já foi identificado no capítulo anterior, pode existir a necessidade de executar vários serviços em sequência. O SR é uma tecnologia capaz de dar resposta a esta necessidade através da introdução de etiquetas nos cabeçalhos dos pacotes, indicando assim os nodos que estes devem visitar. No entanto, com a introdução destes novos requisitos, o problema de gerir os recursos torna-se cada vez mais difícil de resolver.

Duas estratégias de otimização foram consideradas. A primeira, *online*, preocupa-se em identificar os nodos de processamento e o caminho SR para cada pedido consoante estes forem sendo solicitados. A segunda, *offline*, considera um conjunto de pedidos, um estado

da rede, e procura identificar não somente os nodos de processamento e caminhos SR para cada pedido, mas também as configurações do IGP. Este último é particularmente relevante pois não deverá ser mutável ao longo do tempo. De facto, alterações às configurações do IGP têm um impacto significativo na forma como o tráfego flui na rede e consequentemente, na utilização dos recursos de encaminhamento.

Inicialmente foi considerada uma abordagem *offline* para o problema da optimização da distribuição dos serviços e simultânea otimização dos recursos. Todavia, definir um método de otimização capaz de fornecer uma solução completa, na qual se enquadra a otimização dos pesos IGP, localização dos serviços, distribuição dos pedidos, e caminhos SR resultantes, numa primeira abordagem revelou-se uma tarefa excessivamente complexa. Assim, optou-se por, partindo de problemas mais simples e de possíveis estratégias de resolução, ir acrescentando complexidade ao problema.

Nesse sentido, o problema foi sub-dividido em várias etapas, ou subproblemas. O objetivo consiste em obter, face a um conjunto de pedidos de encaminhamento, a configuração óptima da rede que converge nos seguintes pontos:

1. Distribuição dos serviços pelos nodos da topologia.
2. Obtenção dos caminhos SR para cada pedido de encaminhamento.
3. Obtenção dos pesos IGP para as ligações da topologia.

3.2 modelos de otimização multi-commodity flow com serviços fixos

Para atingir os pontos mencionados é necessário começar por definir uma configuração topológica capaz de integrar os conceitos inerentes ao NFV. A distribuição dos serviços pela topologia afeta diretamente os níveis de utilização nas ligações e nos nodos. Como tal, é importante utilizar um método de otimização capaz de avaliar os níveis de utilização resultantes dessa distribuição. Para isso podem ser utilizados modelos lineares. Assim, a primeira abordagem ignora os mecanismos de encaminhamento de tráfego, considerando que este pode fluir pela rede sem qualquer tipo de restrição. Para além de garantir o processamento dos pedidos, o principal objetivo consiste em encaminhar o tráfego da origem ao destino otimizando a utilização dos recursos. Este constitui o que é geralmente denominado por um problema de *Multi-Commodity Flow*. Pretende-se então nesta primeira etapa obter a distribuição dos serviços e os caminhos, sem considerar nenhum protocolo de encaminhamento, para cada pedido.

Foram definidos dois modelos de programação linear, ou seja, dois modelos de Mixed-integer linear programming (MILP). Estes consideram as restrições de serviços e a capacidade de processamento dos nodos para associar cada pedido a um nodo (ou mais caso no caso de VNF *chaining*). Os serviços estão fixos nos nodos, podendo o mesmo serviço

estar disponível em mais do que um nodo em simultâneo. O objetivo consiste em distribuir o tráfego pelas ligações da topologia, minimizando as funções objetivo Φ^* ou MLU identificadas na subsecção 2.3.2, de modo a cumprir com as restrições impostas.

Um objetivo adicional mede a utilização dos nodos ao serem executados serviços nos mesmos, introduzindo uma penalização naqueles que possuem uma maior utilização. Nesse caso é utilizada a função $\Gamma(u(n))$ onde $u(n)$ é a fração de utilização do nodo n e Γ é a função Φ. É possível combinar os dois objetivos agregando-os linearmente, isto é, minimizar $\alpha\Phi^* + (1-\alpha)\Gamma$, para $\alpha\epsilon[0,1]$.

De forma análoga às funções anteriormente definidas, para o modelo linear que utiliza como função objetivo o MLU, a medição da utilização dos nodos é feita através do Maximum Node Utilization (MNU).

3.2.1 *Definição e Representação das Variáveis dos Modelos*

A rede é modelada como um grafo $G(N, A, c(a), q(n))$, onde N representa os nodos com uma capacidade de processamento $q(n)$, e A os arcos com capacidade $c(a)$. A capacidade $q(n)$ é uma função utilidade que caracteriza o nodo (número de CPUs, memória disponível, etc.), identificando num único valor real um limite de utilização. Cada serviço tem as suas próprias necessidades de processamento, consumindo assim recursos do nodo onde é processado. Cada nodo $n \in N$ disponibiliza um conjunto de serviços $S_n = \{s_k\}$.

Os pedidos são definidos como tuplos $(n_i^s, n_i^t, x_i, p_i = [1..K], X = [x_{i,0}, x_{i,1}, ..., x_{i,K}])$, onde $i = 1..I$ identifica o pedido, n_i^s e n_i^t são respectivamente nodos de entrada e saída na rede, x_i os requisitos iniciais de largura de banda, e p_i a lista ordenada de K serviços requisitados pelo pedido i. A lista de nodos que implementa um serviço k é definida por $N_k \subset N$. O processamento de um serviço pode resultar numa alteração do volume de tráfego. Assim, o vetor X define os volumes de tráfego resultante do processamento do pedido i no serviço k.

Cada pedido é segmentado num conjunto $P_i = [(o_{i,k}, d_{i,k}, x_{i,k})]$, respectivamente o "*inicio*", "*fim*" e "*volume*" do segmento. O "*inicio*" e "*fim*" identificam um nodo ou um serviço. No caso de $o_{i,k}$ identificar um serviço, este será o ID do serviço que acabou de ser processado. Alternativamente, caso identifique um nodo, este será o nodo de origem n_i^s. De forma análoga, $d_{i,k}$ pode identificar o nodo de destino do pedido, n_i^t, ou bem o ID do serviço que será processado a seguir. O "*volume*" define o volume de tráfego que sai do nó ou serviço inicial por unidade de tempo.

No exemplo da Figura 3.1, os segmentos são $P_i = [(s, 1, x_{i,0}), (1, 2, x_{i,1}), (2, t, x_{i,2})]$. Os volumes de tráfego $x_{i,k}$ são previamente calculados com base nos serviços solicitados e o volume de tráfego inicial x_i. Note-se que $x_{i,0} = x_i$.

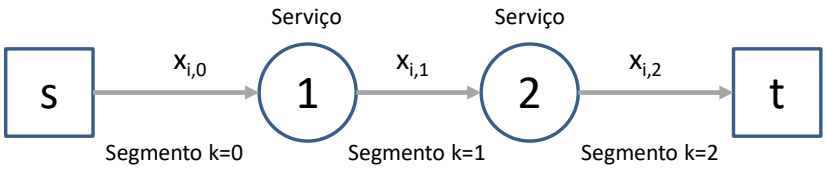


Figura 3.1: Pedido i.

Para cada segmento $(o_{i,k}, d_{i,k}, x_{i,k})$ são calculadas cargas parciais $l_a^{i,k}$, ou seja, o volume de tráfego referente ao segmento $(o_{i,k}, d_{i,k}, x_{i,k})$ encaminhado através do arco $a \in A$. Note-se que a indexação dos segmentos varia de 0 a K enquanto a indexação dos serviços de 1 a K, ou seja, um segmento k inicia no nodo que implementa o serviço k e termina no nodo que implementa o serviço $k+1$. Para $k = 0$, o "*serviço*" no nodo inicial corresponde ao nodo de origem do pedido (n_i^s); para $k = K$, o nodo final do segmento implementa o "*serviço*" $K+1$ corresponde ao nodo de destino (n_i^t). A Tabela 3.1 resume a simbologia utilizada na representação dos modelos.

Variável	Descrição
N	Conjunto de Nodos
X	Volumes de tráfego resultantes do processamento dos serviços
A	Conjunto de arcos (u, v)
$q(n)$	Custo máximo de processamento do nodo $n \epsilon N$
$c(a)$	Capacidade do arco $a \epsilon A$
$S = \{S_k\}$	Lista de Serviços disponíveis
$S_n = \{S_k\}$	Lista de serviços disponíveis no nodo $n \epsilon N$
$C_k(x)$	Custo do serviço s_k em função do tráfego x a processar
n_i^s, n_i^t, s_k^i	Nodo de origem, nodo de destino e serviços de um pedido p_i
x_i	Requisitos de tráfego inicial do pedido i
$x_{i,k}$	Requisitos de encaminhamento do pedido i à saída do serviço k
$(o_{i,k}, d_{i,k}, x_{i,k})$	Segmento k do pedido i com origem $o_{i,k}$ e destino $d_{i,k}$
l_a	Volume de tráfego no arco $a \epsilon A$
$l_a^{i,k}$	Volume de tráfego no arco $a \epsilon A$ referente ao segmento $(o_{i,k}, d_{i,k}, x_{i,k})$
r_n	Soma dos custos da utilização do nodo n
$\alpha_{i,k}^n$	1 se o serviço k for processado no nodo n para o pedido i, caso contrátrio 0
$\beta_a^{i,k}$	1 se o link a transportar tráfego do segmento $(o_{i,k}, d_{i,k}, x_{i,k})$, caso contrário 0
MLU	Utilização máxima da ligação
MNU	Utilização máxima do nodo
Φ	Função de medição de congestão Φ
Γ	Função de medição da utilização dos nodos, baseada na função Φ

Tabela 3.1: Simbologia utilizada nos MILPs.

Desta forma é possível garantir que os modelos lineares têm em consideração a ordem de execução dos serviços para cada pedido, visto que, o destino do segmento é alterado conforme o próximo serviço a ser executado.

3.2.2 *Modelo de Otimização para a Função Proposta por Fortz e Thorup*

O modelo linear, desenvolvido nesta subsecção, baseia-se na função de avaliação de congestão Φ proposta por Fortz e Thorup. Tal como foi referido, para além de ser necessário avaliar os níveis de utilização nas ligações da topologia, é também necessário avaliar os níveis de utilização dos nodos da mesma.

O modelo linear (MILP) pode ser definido como:

Objetivo:

$$\text{minimize } \frac{\alpha}{|A|} \times \Phi + \frac{(1-\alpha)}{|N|} \times \Gamma$$

subject to:

$$l_a = \sum_i \sum_k l_a^{i,k} \qquad ,\forall a = (u,v) \in A, \forall i,k \tag{3.1}$$

$$0 \leq \alpha_{i,k}^n \leq 1 \tag{3.2}$$

$$\alpha_{i,k}^n = 0 \text{ if } k \notin S_n \tag{3.3}$$

$$\sum_{n \in N_k} \alpha_{i,k}^n = 1 \qquad ,\forall k \in p_i, i = 1..I \tag{3.4}$$

$$\Phi_a - l_a \geq 0 \tag{3.5}$$

$$\Phi_a - 3l_a \geq -\frac{2}{3}c_a \tag{3.6}$$

$$\Phi_a - 10l_a \geq -\frac{16}{3}c_a \tag{3.7}$$

$$\Phi_a - 70l_a \geq -\frac{178}{3}c_a \tag{3.8}$$

$$\Phi_a - 500l_a \geq -\frac{1468}{3}c_a \tag{3.9}$$

$$\Phi_a - 5000l_a \geq -\frac{16318}{3}c_a \qquad \forall a \in A \tag{3.10}$$

$$\Phi = \sum \Phi_a \tag{3.11}$$

$$r_n = \sum_i \sum_{k \in S_n} C_k \left(\alpha_{i,k}^n \times x_{i,k}\right) \qquad ,\forall n \in N \tag{3.12}$$

$$\Gamma_n - r_n \geq 0 \tag{3.13}$$
$$\Gamma_n - 3r_n \geq -\frac{2}{3} q_n \tag{3.14}$$
$$\Gamma_n - 10r_n \geq -\frac{16}{3} q_n \tag{3.15}$$
$$\Gamma_n - 70r_n \geq -\frac{178}{3} q_n \tag{3.16}$$
$$\Gamma_n - 500r_n \geq -\frac{1468}{3} q_n \tag{3.17}$$
$$\Gamma_n - 5000r_n \geq -\frac{16318}{3} q_n, \qquad \forall n \in N \tag{3.18}$$
$$\Gamma = \sum \Gamma_n \tag{3.19}$$
$$l_a^{i,k} = \beta_a^{i,k} \times x_{i,k} \tag{3.20}$$

$$\sum_v l_{(n,v)}^{i,k} - \sum_u l_{(u,n)}^{i,k} = \begin{cases} 0, & \text{if } n = o_{i,k} = n_i^s = d_{i,k} = n_i^t \\ (1 - \alpha_{i,k}^n) \times x_{i,k}, & \text{if } n = o_{i,k} = n_i^s \\ (\alpha_{i,k+1}^n - 1) \times x_{i,k}, & \text{if } n = d_{i,k} = n_i^t \\ (\alpha_{i,k+1}^n - \alpha_{i,k}^n) \times x_{i,k}, & \text{otherwise} \end{cases} \qquad ,\forall n,u,v \in N, \forall i,k \tag{3.21}$$

As condições :

- (3.1) define o volume de tráfego no link $a \in A$. Este é dado através da soma de todo o tráfego de segmento que o utilizam.
- (3.2-3.4) define que o tráfego associado a um pedido deverá ser processado uma vez num nodo que implementa esse mesmo serviço.
- (3.5)-(3.11) definem a função de custo Φ.
- (3.12) define o custo de utilização de um nodo.
- (3.13)-(3.19) definem a função de custo Γ.
- (3.20) define que todo o tráfego de um mesmo pedido (fluxo) tem que seguir um mesmo caminho. Poderá ser removido com *Multi Path* TCP.

- (3.21) define a conservação de fluxo, ou seja, o nodo de origem e os nodos que implementaram o serviço do segmento anterior produzem tráfego, o nodo de destino os nodos que implementam o serviço do segmento corrente consomem tráfego, os restantes nodos não produzem nem consomem tráfego.

O valor de *trade-off* por defeito entre os objetivos será 0.5, sendo o objectivo neste caso minimizar : $\frac{\Phi+\Gamma}{2}$.

3.2.3 *Modelo de Otimização para o MLU e MNU*

O modelo linear, desenvolvido nesta subsecção, baseia-se na função de avaliação de congestão MLU. De forma análoga ao modelo anterior, esta função foi adaptada de forma a poder avaliar o nível de utilização nos nodos. A função MNU é obtida através da divisão entre a utilização de um nodo pela capacidade do mesmo.

O MILP pode ser definido como:

Objetivo:

$$\text{minimizar } \alpha \times \text{MLU} + (1-\alpha) \times \text{MNU}$$

sujeito a:

$$l_a^{i,k} = \beta_a^{i,k} x_{i,k} \tag{3.22}$$

$$l_a = \sum_i \sum_k l_a^{i,k} \qquad ,\forall a = (u,v) \in A, \forall i,k \tag{3.23}$$

$$0 \leq \alpha_{i,k}^n \leq 1 \tag{3.24}$$

$$\alpha_{i,k}^n = 0 \text{ if } k \notin S_n \tag{3.25}$$

$$\sum_{n \in N_k} \alpha_{i,k}^n = 1 \qquad ,\forall k \in p_i, i = 1..I \tag{3.26}$$

$$r_n = \sum_i \sum_{k \in S_n} C_k \left(\alpha_{i,k}^n \times x_{i,k}\right) \qquad ,\forall n \in N \tag{3.27}$$

$$\frac{l_a}{c_a} \leq \text{MLU} \qquad \forall a \in A \tag{3.28}$$

$$\frac{r_n}{q_n} \leq \text{MNU} \qquad \forall n \in N \tag{3.29}$$

$$\sum_{v} l^{i,k}_{(n,v)} - \sum_{u} l^{i,k}_{(u,n)} = \begin{cases} 0, & \text{if } n = o_{i,k} = n^s_i = d_{i,k} = n^t_i \\ (1 - \alpha^n_{i,k}) \times x_{i,k}, & \text{if } n = o_{i,k} = n^s_i \\ (\alpha^n_{i,k+1} - 1) \times x_{i,k}, & \text{if } n = d_{i,k} = n^t_i \\ (\alpha^n_{i,k+1} - \alpha^n_{i,k}) \times x_{i,k}, & \text{otherwise} \end{cases} , \forall n, u, v \in N, \forall i, k \tag{3.30}$$

As condições definem:

- (3.28) definição da função custo MLU.
- (3.29) definição da função custo MNU.

As restantes funções deste modelo linear são análogas às do modelo linear anterior. Tal como no modelo anterior, o valor de *trade-off* por defeito entre os objetivos será 0.5, sendo o objectivo neste caso minimizar : $\frac{MLU+MNU}{2}$.

3.2.4 *Comparação com outros Modelos Lineares*

No trabalho realizado em [48], é efetuada uma análise às possíveis abordagens para solucionar problemas de alocação de recursos para Service Function Chaining (SFC). Os autores identificam três etapas para solucionar o problema:

1. Calcular o número ótimo de VNFs a disponibilizar.
2. Distribuir as VNFs em nodos físicos sem que a capacidade dos nodos não seja excedida.
3. Associar pedidos de encaminhamento a VNFs sem que a capacidade das mesmas não seja excedida.

No entanto, o problema é considerado NP-Difícil. O custo associado à sua resolução, especialmente em topologias de grandes dimensões, é demasiado elevado. Nesse sentido, visto que a solução exata (MILPs) é sensível à complexidade da topologia, são consideradas soluções híbridas baseadas em meta-heurísticas.

No trabalho desenvolvido em [47] é abordado o processo de otimização SR para VNF *chaining*. Para isso, os autores formularam um modelo linear com o objetivo de minimizar o tamanho do cabeçalho SR para todos os pedidos que requerem um conjunto de VNFs. Dado que o modelo definido é NP-Difícil, os autores propõem a utilização de um algoritmo baseado em *backtracking* e programação dinâmica. Este permite gerar a lista SR para cada pedido de encaminhamento.

O modelo formula a topologia como sendo um grafo $G(V, E)$ no qual V define um conjunto de nodos físicos e E as ligações físicas entre nodos. Um nodo possui atribuído um conjunto de instâncias VNF e cada pedido de encaminhamento requer uma lista ordenada de funções.

Comparativamente com os modelos apresentados nesta dissertação, os autores dependem de variáveis binárias para definir o conjunto de restrições relacionadas com a conservação do fluxo de tráfego e ordem de processamento das funções. Nos modelos definidos nesta dissertação, todas as restrições identificadas anteriormente são aplicadas através das Equações 3.21 e 3.30 dos modelos Φ e MLU respetivamente. Estas asseguram assim a ordem de execução dos serviços pretendidos por cada pedido bem como a conservação de fluxo.

Em suma, o modelo analisa o impacto que a lista SR, proposta pelo algoritmo para cada pedido, tem no tamanho do cabeçalho. Apesar da abordagem e da função objetivo serem diferentes, é possível identificar semelhanças com os modelos definidos nesta dissertação ao nível das restrições associadas ao processamento de tráfego nos nodos que implementam os serviços.

Os autores em [46] abordam o tema de SFC e alocação de recursos em NFV. Primeiramente é formulado um MILP que tem em consideração, para além dos custos de utilização dos recursos, os custos de instalação associados às instâncias VNF. Dado que se trata de um problema NP-Difícil, os autores propõe uma abordagem, baseada em heurísticas, dividida em duas etapas. A primeira é formulada através de programação linear e aborda a calendarização de tráfego entre instâncias VNF adjacentes, servindo de métrica de avaliação para a etapa seguinte. Para além disso tem também a vantagem de ser facilmente resolvida com algoritmos simples permitindo obter soluções determinísticas num curto intervalo de tempo. A segunda etapa considera a utilização de um *Multi-path greedy algorithm*, resolvendo em simultâneo as questões relativas ao número de instâncias a disponibilizar bem como a localização das mesmas. Comparativamente com os modelos definidos nesta dissertação, os autores têm em consideração, para além dos custos associados à utilização dos recursos, os custos associados à instalação de instâncias. Assim, é possível penalizar a utilização de soluções com um número elevado de instâncias VNF.

3.3 Otimização da Localização dos Serviços

Nesta secção é abordado o processo de otimização da distribuição dos serviços pelos nodos da topologia. Este processo conta com duas abordagens. Uma utiliza somente um modelo linear e a outra algoritmos evolucionários, que por sua vez recorrem aos modelos lineares para avaliar as *fitness* dos indivíduos.

3.3.1 *Abordagem com os Modelos Lineares*

A abordagem anterior supõe que cada nodo implementa um conjunto específico de serviços, podendo estes ser desde todos ou bem nenhum. Todavia, um problema alternativo, e que estende o anterior, consiste em identificar os serviços a disponibilizar em cada nodo de modo a minimizar a utilização de recursos. Uma simples alteração ao modelo definido na secção anterior permite contemplar este problema. Para isso, é suficiente remover a restrição correspondente à Equação 3.3 no modelo linear de otimização da função Φ, ou à Equação 3.25 no modelo da função MLU. Assim, qualquer nodo da topologia pode implementar qualquer um dos serviços, o que permite ao MILP definir a localização dos mesmos. Após a resolução do problema MILP, a variável binária $\alpha_{i,k}^{n}$ permite determinar, para cada pedido de encaminhamento, o nodo ou nodos nos quais os serviços requeridos foram executados. Uma solução para esse problema inclui, para cada pedido de encaminhamento, as seguintes informações:

- Origem do pedido.
- Destino do pedido.
- Largura de banda necessária.
- Lista, com a identificação do nodo no qual cada serviço foi executado, ordenada por ordem de execução dos serviços conforme o pedido de encaminhamento.

Este processo de obtenção de soluções relativas à localização dos serviços é um problema NP-Difícil. O tempo necessário para obter soluções a partir do modelo linear aumenta com a complexidade da topologia e com o número de pedidos de encaminhamento. Adicionalmente, diferentes condições iniciais para as variáveis $\alpha_{i,k}^{n}$ permitem impor restrições sobre os serviços disponíveis em cada nodo da topologia. Esta abordagem permite assim analisar diversos cenários topológicos onde é imposto que um dado serviço se encontra ativo/inativo em nodos específicos.

3.3.2 *Abordagem com Algoritmos Evolucionários*

Como referido anteriormente, otimizações baseadas exclusivamente em modelos MILP têm problemas de escalabilidade. Algoritmos híbridos que combinam programação linear com meta-heurísticas, como algoritmos evolucionários, têm sido cada vez mais utilizados na resolução deste tipo de problemas. Nestes, as soluções (indivíduos) geradas são avaliadas pelos modelos lineares. Uma solução do algoritmo evolucionário define assim a lista de serviços a implementar em cada nodo. Este processo é efetuado com recurso a

uma codificação da solução que, através de uma lista de inteiros com a dimensão igual ao número de nodos, faz corresponder a cada nodo uma combinação de serviços. Alternativamente, poder-se-ia codificar os serviços ativos e inativos como uma lista booleana convertendo essa mesma lista numa representação inteira.

Assim, considerando como exemplo a seguinte lista de serviços $l = [0, 1, 2]$, é possível associar as *fitness* a uma lista de serviços correspondente para disponibilizar em cada nodo. Este processo de codificação está representado na Tabela 3.2.

Valor	Lista de serviços correspondente
0	[]
1	[0]
2	[1]
3	[2]
4	[0, 1]
5	[0, 2]
6	[1, 2]
7	[0, 1, 2]

Tabela 3.2: Codificação da solução dos algoritmos evolucionários.

A título de exemplo, a Tabela 3.3 corresponde a uma configuração de uma solução gerada pelo algoritmo evolucionário, representada pela lista $L = [0, 1, 4, 2, 7]$. Esta solução é referente a uma topologia com cinco nodos. Cada inteiro na lista corresponde a um valor da codificação para o nodo com o ID correspondente à posição do valor na lista L.

O problema da distribuição dos serviços pode ser abordado de duas maneiras distintas. Uma considera que existe um número máximo de serviços a distribuir pelos nodos. Dessa forma, o algoritmo evolucionário vai procurar minimizar os valores de congestão obtidos pelos modelos lineares e, caso a configuração apresente um número total de serviços acima do limite definido, será aplicada uma penalização ao valor da solução obtida. Esta penalização vai permitir ao algoritmo evolucionário procurar configurações que possuam um número máximo de serviços menor ou igual ao limite definido.

Nodo	Lista de serviços disponíveis
0	[]
1	[0]
2	[0, 1]
3	[1]
4	[0, 1, 2]

Tabela 3.3: Exemplo de distribuição de um conjunto de serviços pelos nodos de uma topologia.

Considerando as funções objetivo dos modelos lineares definidos na Secção 3.2, é possível definir os objetivos do algoritmo como sendo:

- 1º Objetivo: Φ ou MLU
- 2º Objetivo: Γ ou MNU

Esta abordagem permite aferir o melhor *trade-off* entre a utilização dos nodos e das ligações. A outra abordagem considera o número de serviços a disponibilizar como uma das *fitness* dos indivíduos. O algoritmo, para além de minimizar os valores de congestão da topologia, vai também minimizar o número total de serviços disponíveis. Os objetivos do mesmo podem ser definidos como:

- 1º Objetivo: $\frac{\alpha}{|A|} \times \Phi + \frac{(1-\alpha)}{|N|} \times \Gamma$ ou $\alpha \times \text{MLU} + (1-\alpha) \times \text{MNU}$
- 2º Objetivo: Número de serviços distribuídos na configuração

Ambas as abordagens são consideradas *multi-objective*. Poder-se-á ainda considerar uma otimização da distribuição dos serviços com um único objetivo definido como funções agregadoras $\frac{\alpha}{|A|} \times \Phi + \frac{(1-\alpha)}{|N|} \times \Gamma$ ou $\alpha \times \text{MLU} + (1-\alpha) \times \text{MNU}$ com $\alpha \epsilon [0,1]$

3.4 Network Function Virtualization em Segment Routing

Ambas as abordagens anteriores são agnósticas quanto ao mecanismo de encaminhamento utilizado. Nesta secção são estendidas de modo a considerar uma configuração IGP, tendo em conta o peso das ligações entre nodos. O objetivo consiste na otimização dos pesos da topologia a aplicar a cada ligação face às mesmas necessidades de encaminhamento apresentadas ao algoritmo de distribuição dos serviços.

3.4.1 *Optimização dos Pesos IGP*

A decomposição de um pedido numa lista de $(o_{i,k}, d_{i,k}, x_{i,k})$, Figura 3.1, define implicitamente um caminho SR para cada pedido, onde cada segmento é univocamente identificado pelo nodo $d_{i,k}$ (*Nodal Segment*), nodo de destino do pedido ou nodo onde o próximo serviço é executado. Neste contexto, a distribuição do tráfego pelas ligações da topologia depende não somente da lista de segmentos SR, mas também da configuração dos pesos do IGP. Por conseguinte, estes também necessitam ser otimizados de modo a minimizar a utilização dos recursos da rede.

O processo de otimização desta componente conta com os resultados obtidos no processo de distribuição dos serviços para compor a lista de segmentos para cada pedido de encaminhamento. Por conseguinte, a lista representa um caminho, em função do qual, o tráfego será distribuído na rede. Assim, o tráfego associado a um pedido é distribuído pela topologia com recurso ao simulador de *Segment Routing* da *Framework* NetOpt. A distribuição

do tráfego tem em consideração a estratégia de encaminhamento ECMP com o objetivo de explorar múltiplos caminhos na topologia com o mesmo custo.

Para efetuar a otimização dos pesos IGP serão novamente utilizados algoritmos evolucionários. Numa primeira abordagem, a solução do algoritmo evolucionário inclui somente os pesos IGP, sendo cada solução codificada como uma lista de pesos $W = [w_1, ..., w_m]$, onde m é o número de ligações da topologia. Dada uma solução de distribuição de pedidos, obtida com recurso a um MILP, Secção 3.2, e respetiva interpretação como caminhos SR, o objetivo consiste em identificar o conjunto de pesos do IGP que minimizam a diferença entre a distribuição do tráfego em MCF e SR. A avaliação da utilização das ligações em SR será conseguida com recurso à *framework* NetOpt.

3.4.2 *Optimização dos Pesos IGP e Localização de Serviços*

Numa segunda abordagem, pretende-se otimizar simultaneamente os pesos do IGP e a distribuição dos serviços pelos nodos da topologia. Assim, uma solução inclui os pesos IGP mas também, e à semelhança da subsecção 3.3.2, a distribuição dos serviços pelos nodos configurada como uma lista de inteiros. A título de exemplo, e considerando uma topologia com cinco nodos e cinco ligações, uma possível solução é: $[0, 2, 4, 0, 6, 20, 10, 18, 6, 4]$. Nesta lista, os primeiros 5 inteiros representam a codificação apresentada na Tabela 3.2 que para cada nodo da topologia identifica o conjunto de serviços que deverão ter instalados. Os restantes inteiros representam o peso a aplicar a cada ligação da topologia. O objetivo da otimização é o mesmo que o definido para a abordagem anterior, ou seja, minimizar a diferença entre a distribuição do tráfego em MCF e SR.

3.5 DISTRIBUIÇÃO DE PEDIDOS COM MACHINE LEARNING

As abordagens anteriores consideram processos de otimização de configurações num contexto *offline*, ou seja, considerando um estado da rede onde é necessário responder a um conjunto de pedidos simultaneamente. Uma abordagem alternativa consiste em, dada uma configuração topológica e um estado de utilização dos recursos, identificar para cada novo pedido a lista de nodos onde este será processado.

Esta abordagem *online* pode ser alijada com recurso a uma variante do MILP definido na secção 3.2 que inclua o estado de utilização dos recursos. Todavia, para topologias de maiores dimensões o tempo necessário para obter uma solução para o processamento de um pedido pode ser excessivamente longo. Assim, nesta secção considera-se uma abordagem alternativa que recorre a modelos de *machine learning* para identificar os nodos onde um pedido deverá ser processado.

Como tal é necessário construir um conjunto de dados para poder treinar um modelo. Na Tabela 3.4 está representada a simbologia utilizada para identificar as colunas do conjunto de dados.

Variável	Descrição
S	Serviço
N	Nodo
E	Ligações
NS	Número total de serviços
NE	Número total de ligações
NN	Número total de nodos
RN_S_	Nodo de processamento do serviço requisitado

Tabela 3.4: Simbologia do conjunto de dados.

Na Tabela 3.5 é efetuada a descrição de cada *feature*, ou coluna, do conjunto de dados.

Coluna	Tipo de variável	Descrição
origin	Inteiro	ID do nodo origem do pedido
destination	Decimal	ID do nodo destino do pedido
bandwidth	Decimal	Largura de banda necessária
duration	Inteiro	Número de iterações nas quais o pedido estará ativo
S[0...NS]	Inteiro	Se o serviço é requisitado, 1 0 caso contrário
E[0...NE]	Decimal	Taxa de utilização das ligações da topologia
N[0...NN]	Decimal	Taxa de utilização dos nodos da topologia
RN[0...NN]S[0...NS]	Inteiro	Se o serviço S foi executado no nodo N, 1 0 caso contrário

Tabela 3.5: Descrição das colunas do conjunto de dados.

Cada entrada no conjunto de dados inclui a origem e o destino do pedido, a largura de banda, a duração e o estado geral da rede. O resultado a obter seria o caminho SR para um pedido de encaminhamento o qual teria em conta o estado de utilização dos nodos e das ligações da topologia. Os dados estão devidamente etiquetados, ou seja, cada coluna está devidamente identificada.

RN0S0	RN1S0	RN0S1	RN1S1
0	1	1	0

Tabela 3.6: Exemplo de representação do *output* no conjunto de dados.

A Tabela 3.6 representa um possível resultado obtido pelo MILP para uma entrada no conjunto de dados. Neste resultado é considerada a utilização de uma topologia com dois

nodos e dois serviços. Neste, o serviço 0 foi executado no nodo com o ID 1 e o serviço 1 executado no nodo com ID 0.

É possível considerar que passa a existir uma evolução temporal do estado da rede relativamente à utilização dos recursos. Cada pedido de encaminhamento, ao ser adicionado, vai afetar as taxas de utilização das ligações e dos nodos das entradas seguintes no conjunto de dados enquanto o pedido estiver ativo. A cada inserção, a duração dos pedidos é decrementada. Assim que chegar a 0, o pedido é removido da lista de serviços ativos.

3.6 SUMÁRIO

Neste capítulo foram apresentadas várias abordagens incrementais para solucionar os problemas levantados no Capítulo 2. Nesse sentido, foram definidos dois modelos lineares que tem como objetivo minimizar o nível de utilização das ligações e o nível de utilização dos nodos da topologia face a um conjunto de pedidos de encaminhamento. Posteriormente foram introduzidos algoritmos evolucionários e explicada de que forma estes foram integrados com os modelos. Visto que a utilização dos MILP não tem em consideração protocolos de encaminhamento, foram definidas estratégias que incluem a otimização dos pesos da topologia. Estas tem por base os caminhos SR para cada pedido de encaminhamento proveniente do processo de otimização anterior. Todos os processos de otimização descritos permitem assim, para um conjunto de pedidos de encaminhamento, obter uma configuração topológica e os pesos das ligações da topologia. Apesar disso, como o tempo de execução dos processos de otimização descritos é elevado e não aplicável em cenários de tempo real, foi explicado de que forma a utilização de *machine learning* pode potencialmente resolver o problema.

4

IMPLEMENTAÇÃO E INTEGRAÇÃO NA FRAMEWORK NETOPT

Neste capítulo é abordada a implementação do problema identificado no capítulo anterior. Nesse sentido são apresentados os principais módulos desenvolvidos e os já existentes na *framework* que foram utilizados (secção 4.1). De seguida é apresentado o gerador de pedidos de encaminhamento utilizado para gerar um conjunto de pedidos e configurações topológicas associadas aos mesmos (secção 4.2). Na Secção 4.3 é abordado o processo de otimização com recurso aos modelos lineares no qual se enquadra a definição de uma configuração NFV. Na Secção 4.4 é abordado o processo de otimização com recurso aos algoritmos evolucionários. Por fim é abordada a otimização da utilização dos recursos via *machine learning* (secção 4.5).

4.1 REPRESENTAÇÃO DOS MÓDULOS DESENVOLVIDOS NA FRAMEWORK NETOPT

Nesta secção são apresentados os principais módulos, desenvolvidos e os já existentes na *framework*, que foram utilizados. Nestes últimos estão incluídos os módulos que contém os simuladores OSPF e SR. No módulo *NetOpt-JEcoli* estão implementados os algoritmos evolucionários que foram utilizados nesta dissertação. Dada a grande complexidade da *framework*, está representado na Figura 4.1 o diagrama de *packages* que permite visualizar os componentes utilizados na implementação e de que forma estão estruturados. Nas secções seguintes são identificados os módulos que foram necessários desenvolver para acrescentar os conceitos NFV e de que forma estes se enquadram com os já existentes.

O módulo Cplex inclui a implementação dos dois modelos lineares especificados na Secção 3.2. Para além desses contém ainda uma versão alternativa do modelo de otimização do Φ para a componente de *machine learning*. O módulo OSPF contém todos os simuladores, balanceadores de tráfego e representações topológicas provenientes da *framework* NetOpt. O módulo NFV engloba todo o trabalho desenvolvido desde a implementação de uma configuração topológica com características associadas ao NFV, à implementação de todo o processo de otimização com os algoritmos evolucionários. Estes são configurados no módulo *evaluation*. A configuração inclui a escolha do algoritmo evolucionário a utilizar e dos parâmetros associados aos mesmos. No módulo *Utils* estão agregados todos

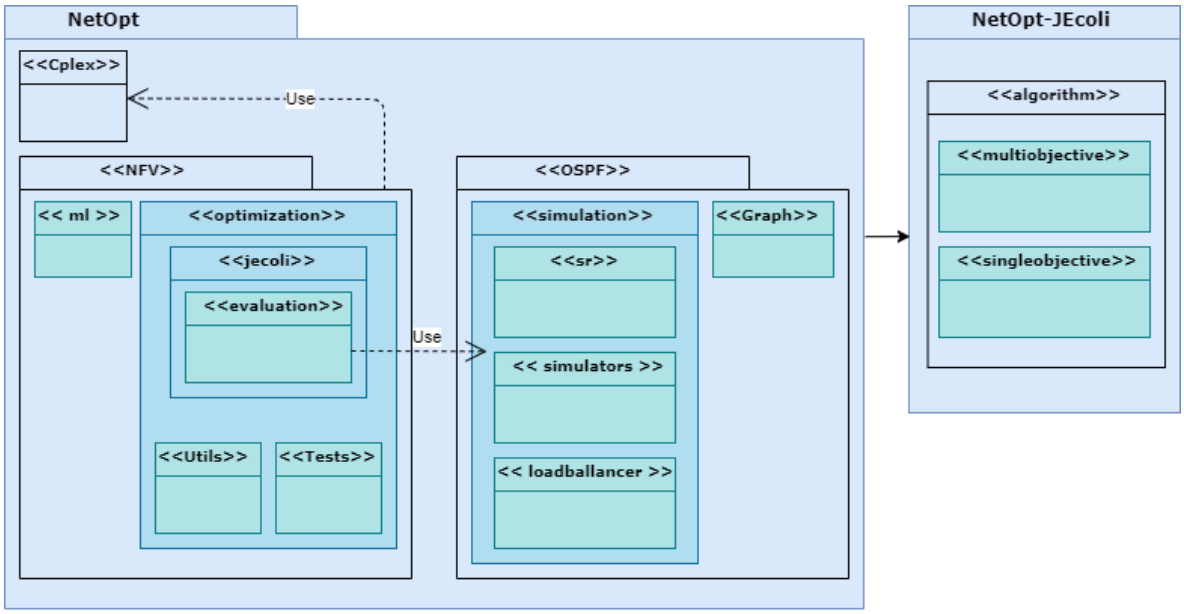


Figura 4.1: Diagrama de *packages* de integração na *framework* NetOpt.

os métodos necessários para carregar configurações e guardar os resultados obtidos. No módulo *Tests* estão desenvolvidas as classes *main* de cada processo de otimização. Estas tem como finalidade permitir executar os algoritmos num terminal a partir de ficheiros executáveis (jar).

4.2 GERADOR DE PEDIDOS DE ENCAMINHAMENTO

O gerador de pedidos de encaminhamento foi desenvolvido com o intuito de, a partir de um ficheiro com informações acerca da topologia, gerar necessidades de encaminhamento aleatórias entre os vários pontos da rede.

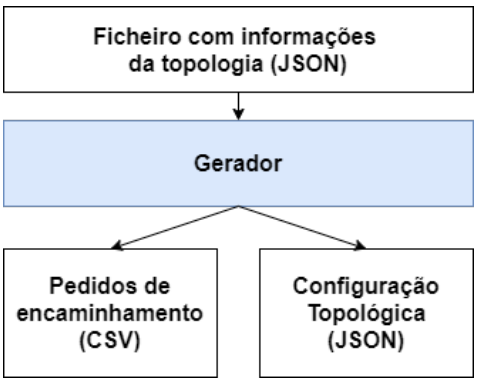


Figura 4.2: Gerador de pedidos de encaminhamento.

Na Figura 4.2 está representado o fluxo de ficheiros de entrada e saída do gerador. Primeiramente é carregado um ficheiro que contém informações relativas à entidades da topologia, nomeadamente: número total de nodos, número de nodos com serviços NFV, número de serviços NFV disponíveis e distribuição dos serviços pelos nodos. A Listagem 4.1 apresenta um exemplo do ficheiro passado como *input* que representa uma topologia com 6 nodos, 3 com serviços instalados sendo que estes não possuem todos os serviços disponíveis.

```
{
    "numberOfNodes": 6,
    "numberOfNodesWithServices": 3,
    "nodesWithAllServices": false,
    "allNodesWithServices": false,
    "numberOfRequests": 300,
    "resultFilepath": ./pedidos.csv,
    "ServicesID": [0,1,2],
    "nodesWithServices": [1,2,3],
    "headers": [id,originID,destinationID,bandwidth,0,1,2]
}
```

Listagem 4.1: Ficheiro de configuração do gerador de pedidos de encaminhamento.

A Listagem 4.2 representa o ficheiro relativo a uma configuração topológica que corresponde às informações apresentadas na Listagem anterior.

```
{
  "nodes":[
    {"id": 0,"capacity": 0, "availableServices":[]},
    {"id": 1, "capacity": 2500, "availableServices":[2]},
    {"id": 2, "capacity": 2500, "availableServices":[1,2]},
    {"id": 3, "capacity": 2500, "availableServices":[0,2]},
    {"id": 4, "capacity": 0, "availableServices":[]},
    {"id": 5, "capacity": 0, "availableServices":[]},
  ],
  "services":[
    {"id": 0, "cost": 4.0, "name": firewall},
    {"id": 1, "cost": 5.0, "name": encryption},
    {"id": 2, "cost": 3.5, "name": VPN concentrator}
  ],
  "maxNodes": 6,
  "nodesWithServices": 3
}
```

Listagem 4.2: Ficheiro representativo da configuração topológica.

O ficheiro relativo à configuração topológica irá ser posteriormente utilizado pela *framework*. Através deste será possível carregar a configuração da topologia pela qual os pedidos de tráfego serão distribuídos. O ficheiro com os pedidos de tráfego apresenta os pedidos aleatoriamente gerados. Cada pedido é composto pelos nodos origem e destino, o volume de tráfego por unidade de tempo necessário e a identificação de cada serviço requisitado por ordem de execução, como é possível observar na Tabela 4.1. Nesta assume-se a existência de três serviços numerados de 0 a 2.

ID	Origem	Destino	Volume de tráfego/s	Execução 1	Execução 2	Execução 3
0	3	6	7	0	2	-1
1	2	9	5	2	0	1
2	6	1	6	1	-1	-1

Tabela 4.1: Exemplos de pedidos de tráfego.

As três últimas colunas representam o número de execuções possíveis, que corresponde ao número de serviços disponíveis. Caso o valor seja negativo (-1), como é possível verificar nos pedidos com o ID 0 e 2, significa que mais nenhum serviço é pretendido.

4.3 Processo de otimização com recurso aos modelos lineares

Nesta Secção é abordada a implementação dos modelos lineares, inseridos no módulo *Cplex* identificado na Secção 4.1. A implementação relativa aos modelos inclui a definição dos conceitos associados ao NFV, na Secção 4.3.1. De seguida é apresentado o processo de execução dos modelos e de que forma são obtidos os resultados a partir dos mesmos.

4.3.1 *Representação de uma Configuração Network Function Virtualization*

A extensão à *framework* consistiu no desenvolvimento dos conceitos associados ao NFV, nomeadamente nodos com capacidade de processamento, serviços correspondentes a *Network Functions* e pedidos de encaminhamento que requerem a execução de um ou mais serviços. Um nodo *NFNode* é composto pela capacidade de processamento que indica o limite máximo associado ao processamento de serviços no nodo e uma lista com o número de identificação dos serviços disponíveis no mesmo. Os pedidos de encaminhamento *NFRequest* são compostos pelo nodo origem, nodo destino, o volume de tráfego por unidade de tempo e a lista com o número de identificação dos serviços que serão necessários por ordem de execução. A classe *NFRequestSegment* permite a introdução dos segmentos, ilustrados na Figura 3.1, garantindo assim a ordem de execução dos serviços. Um serviço *NFService* é constituído por um nome (e.g. *firewall*) e o custo associado à execução do mesmo. O estado *NFVState*

agrega todas as entidades NFV associadas à configuração da topologia. Esta configuração é carregada a partir de dois ficheiros que são gerados pelo gerador de pedidos de encaminhamento, descrito na subsecção anterior.

Para incluir as classes mencionadas, foi acrescentado um novo módulo na *framework*. O diagrama de classes resultante está ilustrado na Figura 4.3.

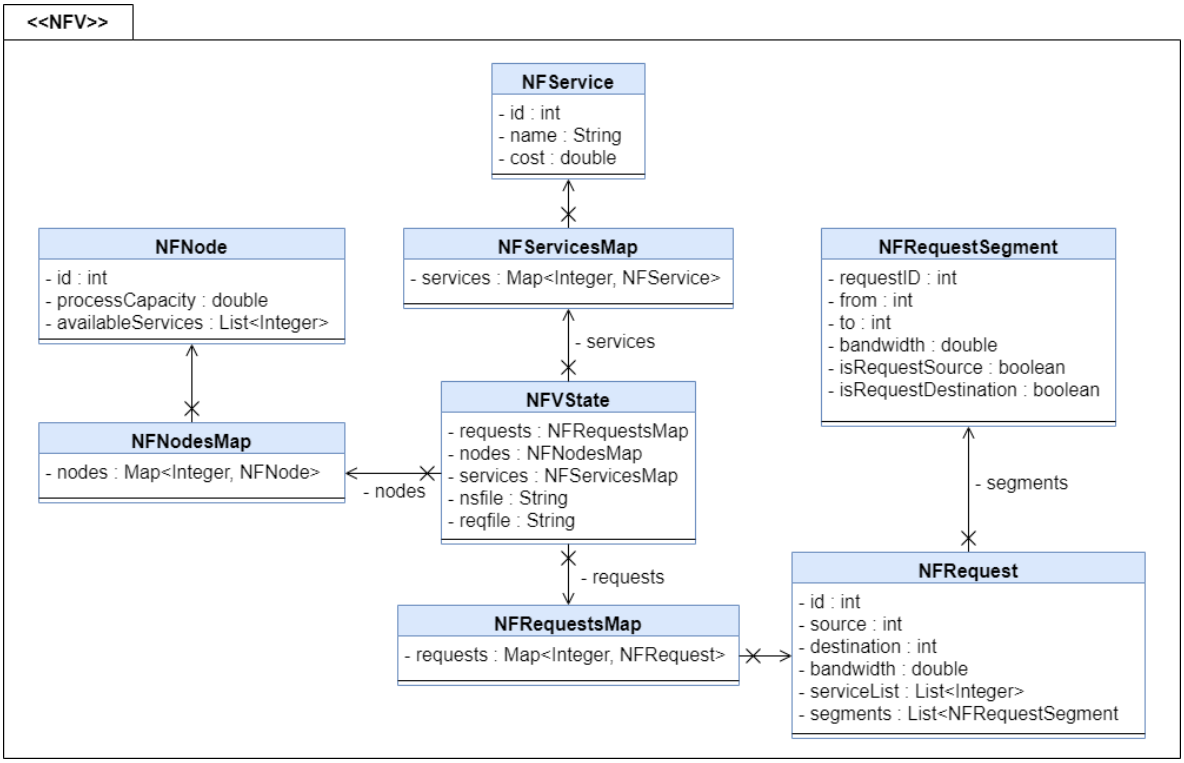


Figura 4.3: Diagrama de classes do módulo NFV.

Estas definições apresentadas não são suficientes por si só para representar uma topologia, visto que, a *framework* efetua essa representação através da classe *NetworkTopology*. Esta é criada a partir do *parsing* de dois ficheiros que correspondem aos nodos e às ligações da topologia ou através de um ficheiro com o formato *GraphML* ou GML.

A Figura 4.4 representa o diagrama de classes associado à classe *NetworTopology*. A esta estão associadas as classes que a *framework* utiliza para representar o grafo *NetGraph*. Cada ligação, representada pela classe *NetEdge*, possui o ID dos nodos origem e destino e a capacidade. A classe *NetNode* representa os nodos da topologia. Assim, os algoritmos definidos requerem a existência desta classe pois esta contém as capacidades das ligações da topologia, representadas a partir de um *array* bidimensional permitindo assim identificar todos os arcos que interligam os nodos da topologia.
Nesse sentido, os conceitos definidos são uma peça fundamental para o processo de otimização

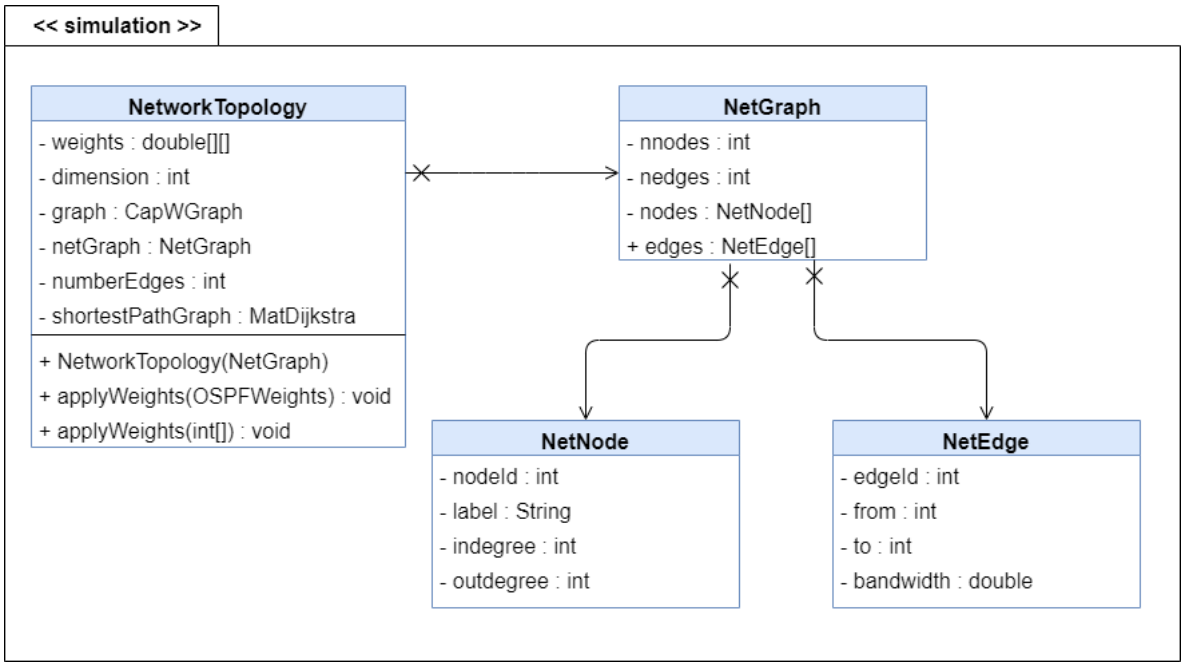


Figura 4.4: Representação de uma topologia na *framework* NetOpt.

da distribuição dos serviços pelos nodos da topologia, complementando as classes já existentes na *framework*.

4.3.2 *Implementação dos Modelos Lineares*

Os modelos lineares definidos foram implementados em Java e recorrem ao *solver* CPLEX para resolver modelos lineares [19]. Para utilizar a ferramenta é necessário definir as variáveis relativas à representação da topologia, da configuração NFV e dos pedidos de encaminhamento. A classe relativa ao modelo recebe como parâmetro uma topologia *NetworkTopology*, uma configuração NFV *NFVState*, o limite de tempo de execução do modelo e o valor da variável α a considerar na função objetivo.

Após a execução do modelo é necessário obter as informações que permitem determinar o local onde cada serviço de cada pedido foi executado. Os resultados são agregados numa classe denominada *OptimizationResultObject*, identificada na Figura 4.5.

Nesta classe, para além das cargas nas ligações e nos nodos resultantes da execução dos MILP, é inserido um objeto que, para cada pedido, indica o caminho SR que por sua vez representa os nodos nos quais os serviços pedidos foram executados. Esta informação é obtida através da variável binária dos modelos $\alpha_{i,k}^{n}$. Os resultados e as configurações *NFVRequestConfiguration* são armazenados num ficheiro JSON. Este é compilado através de métodos existentes no módulo *Utils*.

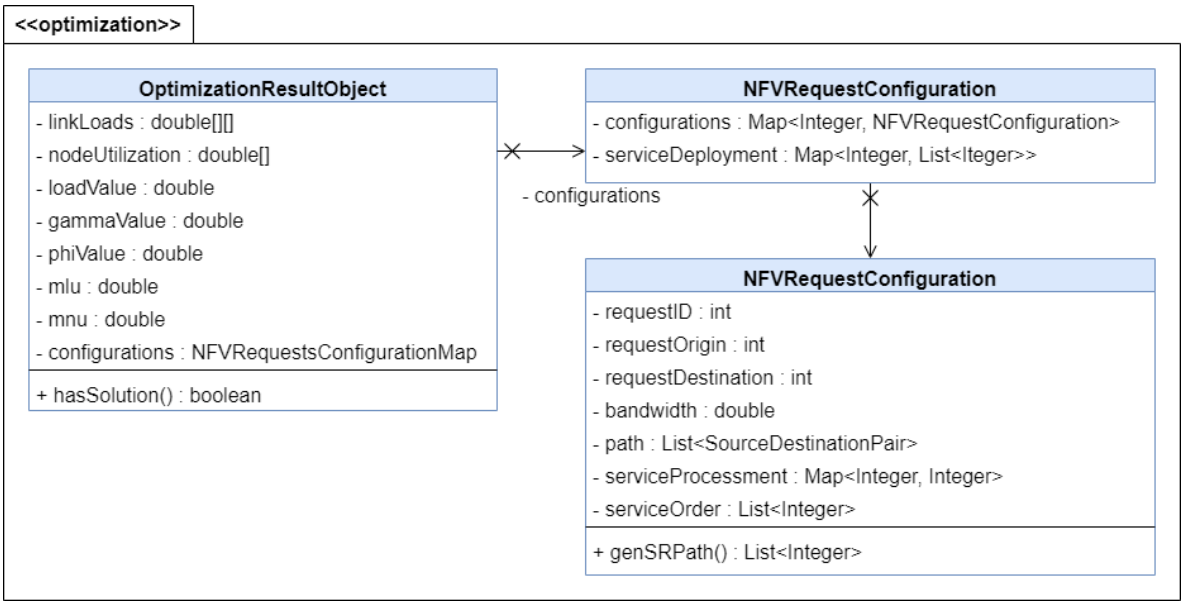


Figura 4.5: Diagrama de classes dos objetos resultantes dos modelos lineares.

A título de exemplo, a Listagem 4.3, ilustra a estruturação do ficheiro com o resultado relativamente a um dos pedidos de encaminhamento. Este é composto pelo ID do pedido, nodos origem e destino, largura de banda estimada, caminho percorrido, local de processamento de cada serviço requerido e, por fim, o caminho SR.

```
{
    "RequestID": 1,
    "Request Origin": 23,
    "Request Destination": 18,
    "Request Bandwidth": 7.51,
    "SegmentPath": [
        { "Origin": 23, "Destination": 6 },
        { "Origin": 6, "Destination": 11 },
        { "Origin": 11, "Destination": 1 },
        { "Origin": 1, "Destination": 18 }
    ],
    "ServiceProcessmentLocation": [
        { "Service ID": 1, "Node ID": 11 },
        { "Service ID": 2, "Node ID": 6 }
     ],
    "NodeIDPath": [6, 11]
}
```

Listagem 4.3: Representação do resultado esperado para cada pedido de encaminhamento.

O caminho SR é obtido através da localização de execução dos serviços e da ordem de execução dos mesmos, definida pelo pedido. O *SegmentPath* foi inserido com o objetivo de corrigir erros no modelo pois, através deste, é possível verificar o funcionamento das equações associadas à circulação de tráfego na rede.

4.4 Processo de otimização com recurso aos algoritmos evolucionários

Nesta Secção é abordado o processo de otimização através dos algoritmos evolucionários. Esta componente de otimização inclui a distribuição dos serviços pela topologia e o conjunto de pesos OSPF a adicionar à mesma de forma a reduzir os níveis de utilização dos recursos.

4.4.1 *Otimização da Localização dos Serviços*

Os algoritmos evolucionários permitem superar a limitação dos modelos lineares no tratamento de problemas NP-Difíceis. Nestes apenas é efetuada uma avaliação por execução, não sendo possível comparar diferentes configurações topológicas sem executar novamente o gerador de pedidos de encaminhamento e posteriormente executar os modelos com a nova configuração. Nesse sentido é necessário que o algoritmo evolucionário possa gerar múltiplas configurações topológicas com o objetivo de averiguar os valores de congestão resultantes. Uma configuração topológica é representada por um indivíduo que, tal como foi identificado na Tabela 3.2 da Subsecção 3.3.2, é codificada através de uma lista de inteiros. Esta indica, para cada nodo da topologia, o conjunto de serviços que devem estar disponíveis e é introduzida ao algoritmo através de um ficheiro JSON. Isto permite agilizar a inserção ou remoção de serviços sem ser necessário efetuar alterações no código fonte.

Na Figura 4.6 está representado o processo de avaliação a um indivíduo de uma população resultante do algoritmo evolucionário. As soluções são convertidas em *NFNodesMap* através do *SolutionParser*. O estado *NFVState* é atualizado com a nova distribuição dos serviços, que se encontra inserida no *NFNodesMap*. O estado é posteriormente analisado pelo modelo linear retornando um objeto, que agrega um conjunto de resultados inerentes à utilização dos recursos da rede.

O algoritmo evolucionário pretende gerar soluções que permitam reduzir a utilização dos nodos e a carga nas ligações da topologia. Uma solução trivial consistiria em instalar todos os serviços em todos os nodos. No entanto, tal solução tem custos de implementação e manutenção excessivos. Apesar destes custos não terem sido considerados nesta dissertação, será preferível encontrar soluções onde somente são instalados os serviços efetivamente necessários e somente em alguns nodos. Na Subsecção 3.3.2 foram identificadas duas

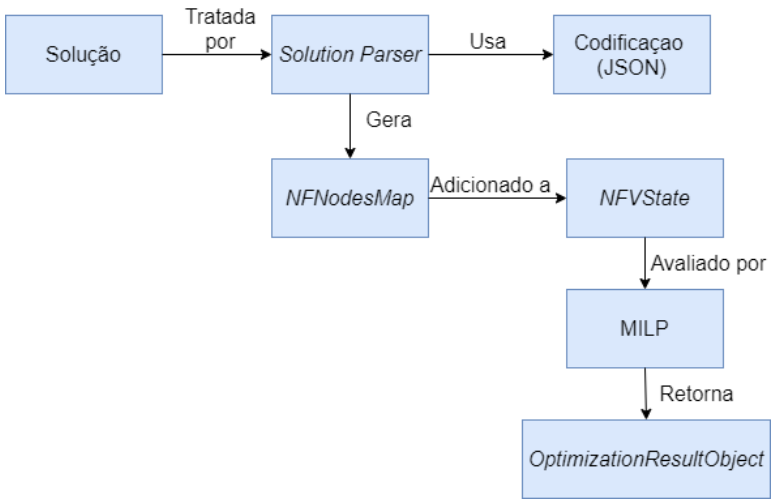


Figura 4.6: Fluxograma do processo de avaliação de uma solução associada à distribuição dos serviços.

possíveis abordagens para o problema. Ambas utilizam o algoritmo *multi-objective* NSGA-II definido no módulo *NetOpt-JEcoli*.

A Figura 4.7 representa o diagrama de classes referente ao processo de otimização da localização dos serviços. A classe *JecoliNFV* é responsável pela configuração do algoritmo evolucionário NSGA-II. Esta recebe parâmetros como os limites superiores e inferiores que as soluções podem assumir, o modelo linear a considerar (baseado na função objetivo Φ ou MLU), valor da variável α, entre outros. A configuração do algoritmo inclui também a escolha da componente de avaliação. Esta componente está situada no módulo *evaluation* e apresenta dois métodos de avaliação que representam as duas abordagens possíveis ao problema. A classe *NFVEvaluationSO*, demonstrada na Figura 4.8, representa a abordagem que tem como objetivos de otimização do valor de congestão na rede e o valor de utilização dos nodos. Esta tem como principais métodos o de avaliação de uma solução (*evaluateSO*) que agrega num *array* os valores de cada objetivo. O método *decode* é responsável por transformar a lista de inteiros proveniente do algoritmo num *NFNodesMap*. Por fim, o método *getPenalization* aplica ao resultado obtido no método de avaliação da solução uma penalização, por exemplo, quando o limite máximo de serviços é ultrapassado.

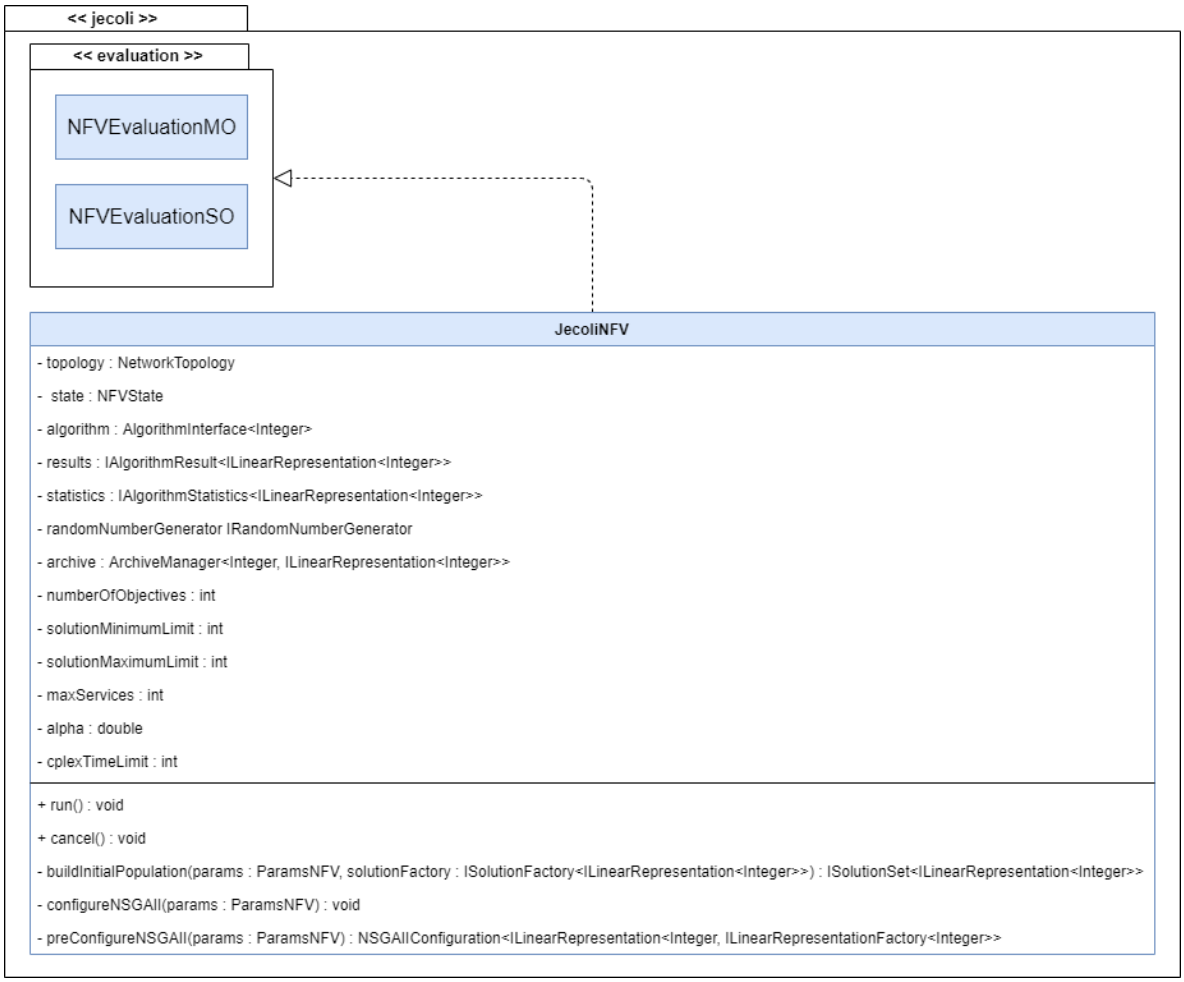


Figura 4.7: Diagrama de classes da configuração do algoritmo evolucionário.

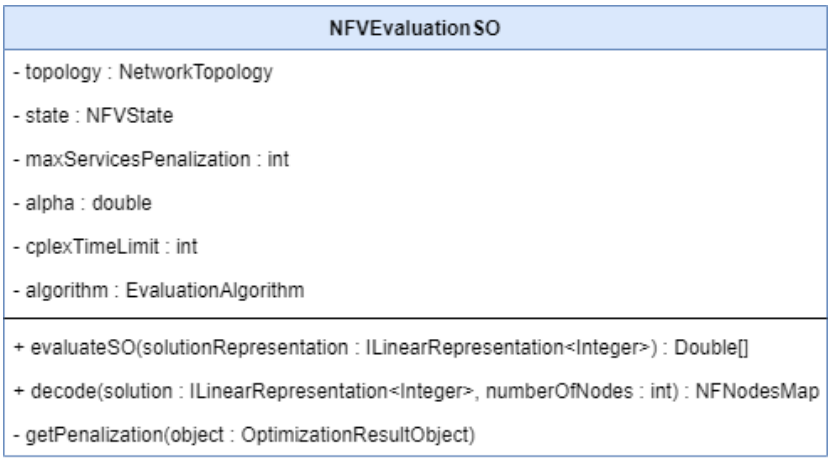


Figura 4.8: Classe de avaliação de uma solução para dois objetivos.

A classe *NFVEvaluationMO*, apresentada na Figura 4.9, representa a abordagem que tem como objetivo extra a minimização do número de serviços disponíveis na configuração. Nesta, o método que aplica a penalização apenas é utilizado quando o algoritmo chega a uma configuração que não tem solução. Esta situação pode ocorrer quando a configuração topológica não possui um dos serviços disponíveis.

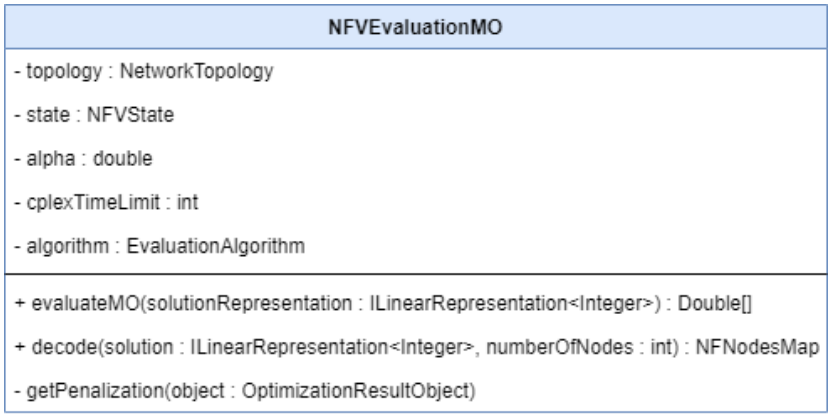


Figura 4.9: Classe de avaliação de uma solução com objetivo de minimização do número de serviços.

Quando o algoritmo evolucionário termina a execução, a melhor solução é novamente avaliada pelo modelo linear. Esta avaliação devolve novamente um *OptimizationResultObject*, no entanto, acrescenta as configurações para cada pedido de encaminhamento. Visto que o algoritmo evolucionário baseia a aprendizagem nos valores obtidos em cada objetivo, as configurações apenas são efetuadas na última execução, reduzindo assim a carga computacional durante o processo de avaliação de indivíduos.

4.4.2 *Otimização dos Pesos da Topologia*

O processo de otimização dos pesos da topologia começa com a leitura do ficheiro com as configurações, proveniente do algoritmo de distribuição dos serviços. Deste ficheiro são extraídos os valores de congestão associados às métricas MLU, MNU, Φ e Γ bem como o caminho SR de cada pedido de encaminhamento. De forma a utilizar os simuladores da *framework*, foi necessário converter o caminho SR de cada pedido em fluxos (*Flow*). Considerando o pedido de encaminhamento representado na Listagem 4.4.

```
{
    "RequestID": 13,
    "Request Origin": 4,
    "Request Destination": 6,
    "Request Bandwidth": 7.5,
    "NodeIDPath": [7,10,6]
```

```
}
```

Listagem 4.4: Exemplo de um pedido de encaminhamento.

A lista de fluxos resultante seria a seguinte:

- Fluxo 1: Nodo 4 - Nodo 7
- Fluxo 2: Nodo 7 - Nodo 10
- Fluxo 4: Nodo 10 - Nodo 6

Nesse sentido, foi acrescentada à *framework* a classe *Request* que inclui para cada pedido o seu ID e a lista de fluxos associada. Na Figura 4.10 está representado o diagrama de classes dos objetos referidos. Cada pedido de encaminhamento é convertido num objeto *Request* que por sua vez contém os respetivos fluxos. Um fluxo é composto pelo nodo origem e destino do pedido, o tipo de fluxo (NFV), que foi adicionado aos já existentes na *framework*, e a necessidade de tráfego associada ao pedido. A variável *aggregated* define se o fluxo agrega mais do que um pedido, o que não se verifica no cenário abordado nesta dissertação.

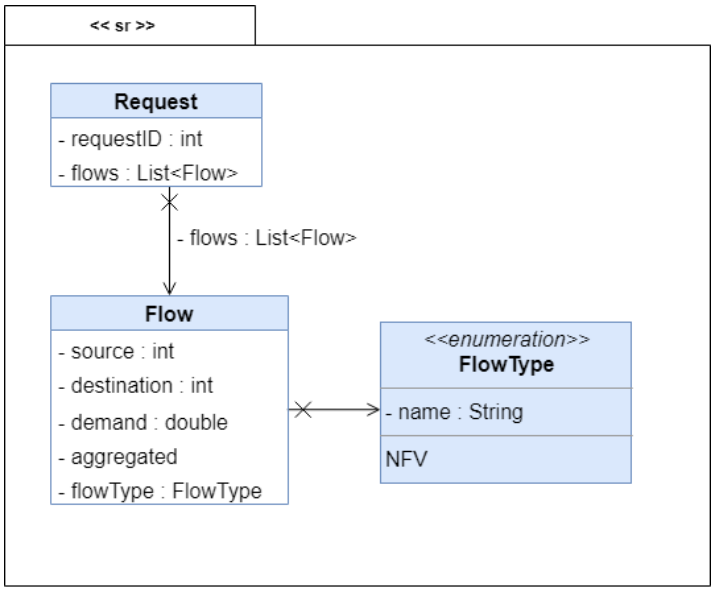


Figura 4.10: Diagrama de classes do algoritmo de otimização dos pesos.

O fluxograma do processo de avaliação das *fitness* está descrito na Figura 4.11. Uma possível solução gerada pelo algoritmo evolucionário é convertida num conjunto de pesos OSPF aplicáveis a uma topologia. Os *Requests* são adicionados sequencialmente ao simulador que, para cada fluxo de cada pedido, descobre o caminho mais curto entre a origem e o

destino do mesmo com base no algoritmo Dijkstra. Após inseridos todos os fluxos é obtido o valor de congestão.

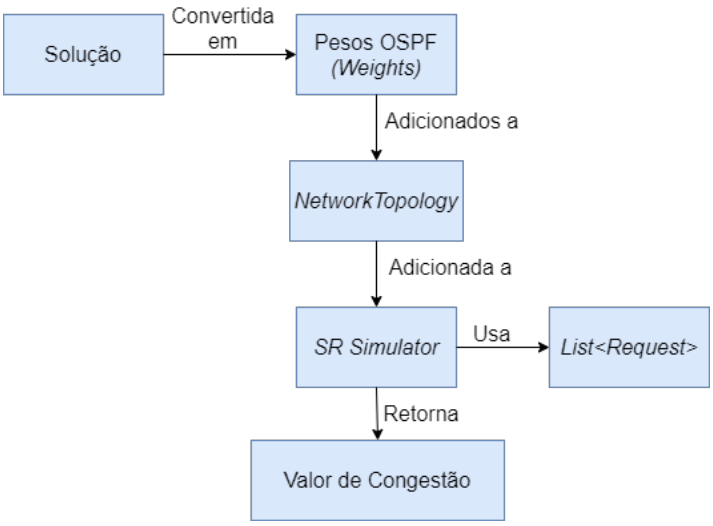


Figura 4.11: Fluxograma do processo de avaliação de uma solução relativa aos pesos IGP.

Figura 4.12: Diagrama de classes do algoritmo de otimização dos pesos IGP.

Na Figura 4.12 está representado o diagrama de classes relativo ao processo de otimização dos pesos IGP. Na classe *JecoliWeights* estão incluídos os métodos responsáveis pela configuração do algoritmo evolucionário *single-objective* (SOEA). A função de avaliação é representada pela classe *NFVWeightsEvaluation*. Esta utiliza os simuladores da topologia para distribuir os fluxos dos pedidos de encaminhamento. Na Figura 4.13 está representada a classe responsável pelo processo de avaliação de indivíduos.

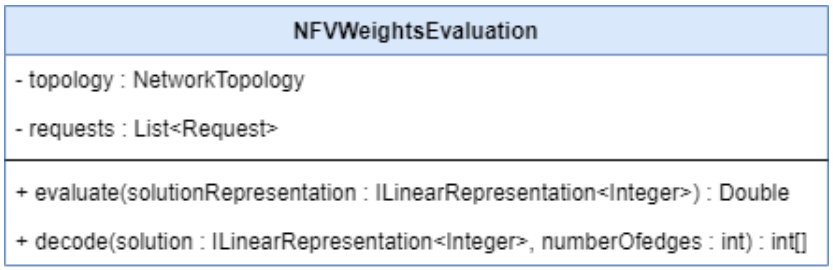


Figura 4.13: Classe de avaliação de indivíduos do processo de otimização dos pesos IGP.

Os resultados obtidos são convertidos num ficheiro que, para cada ligação possui o peso OSPF respectivo. Os pesos possuem um valor definido entre 1 e 20. Embora o OSPF admita valores de pesos no intervalo de 1 a 65535, é aqui considerado o subconjunto de 1 a 20, quer para reduzir o espaço de procura, quer para simultaneamente fomentar um maior número de caminhos com o mesmo peso.

Esta abordagem otimiza somente os pesos IGP dada uma configuração de caminhos SR, resultante da otimização da distribuição dos serviços. É no entanto possível agregar ambas as otimizações num único algoritmo onde cada solução codifica a distribuição dos serviços e o peso para cada ligação da topologia. A configuração deste algoritmo exige a definição das posições no vetor representação que correspondem à configuração dos nodos e as posições dos pesos IGP, tal como está exemplificado na Figura 4.14.

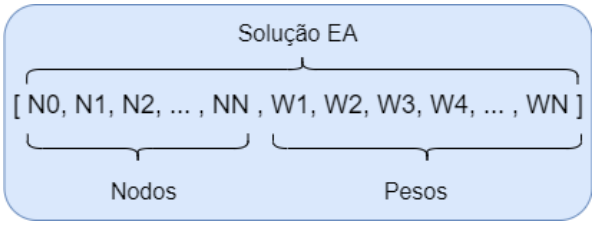


Figura 4.14: Exemplo de uma solução híbrida.

Na Figura 4.15 está representado o diagrama de classes no qual é efetuada a configuração do algoritmo evolucionário de otimização dos pesos IGP e das configurações topológicas em simultâneo. A solução é configurada através do método *configureSolutionsFactoryWP*.

Este define os limites que a solução pode tomar até a um determinado índice do *array* que poderá ser, por exemplo, o número de nodos da topologia. Após essa posição do *array*, os limites da solução representam os pesos IGP.

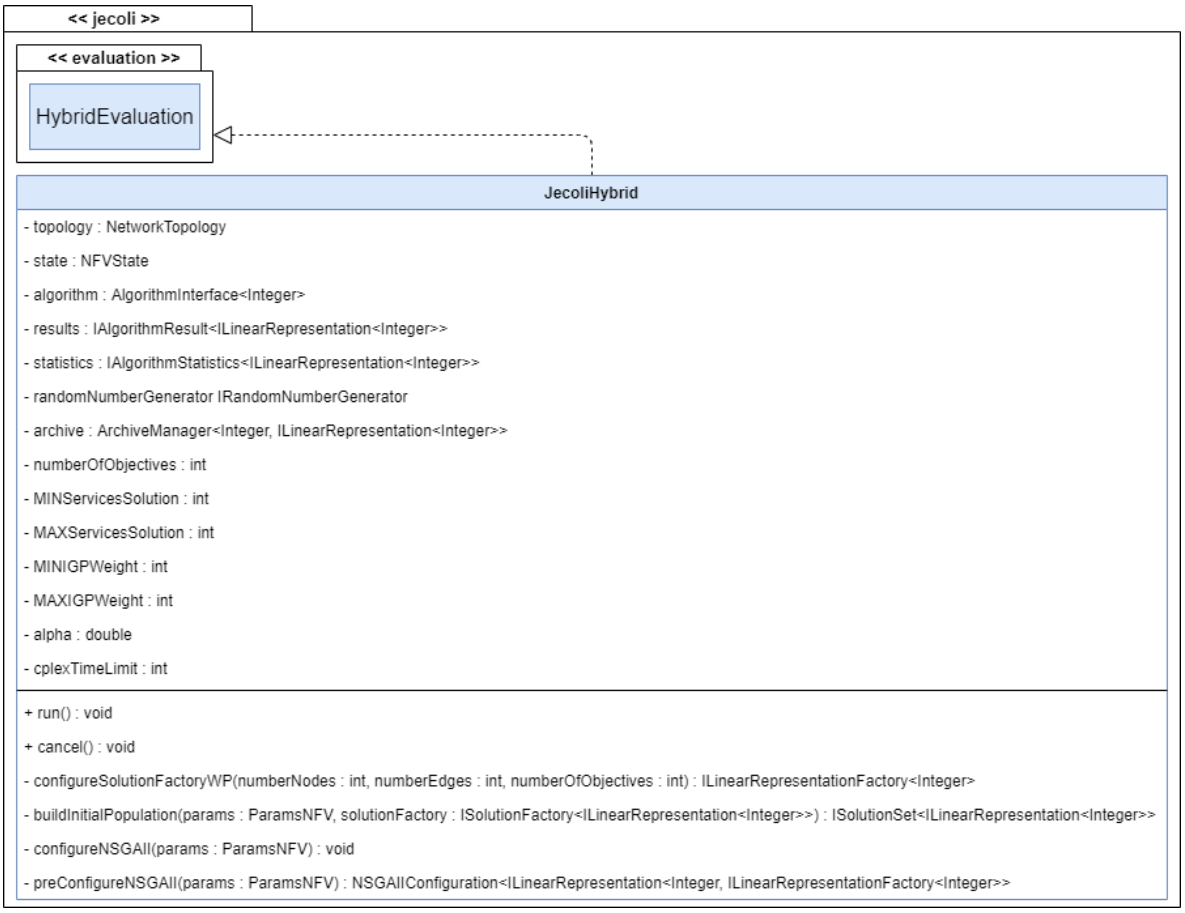


Figura 4.15: Configuração do algoritmo de otimização dos pesos IGP e da distribuição de serviços.

A Figura 4.16 representa a classe responsável pela avaliação dos indivíduos de uma solução híbrida. A solução gerada é avaliada pelo método *evaluateMO* que retorna um *array* com o resultado de cada objetivo do algoritmo. Primeiramente é avaliada a configuração topológica, retornando o valor de congestão obtido. Através do método *getPenalization* é averiguado se o modelo linear encontrou soluções para a configuração proposta. Caso não tenham sido encontradas soluções, a avaliação dos pesos IGP não é efetuada e é aplicada uma penalização a ambos os objetivos.

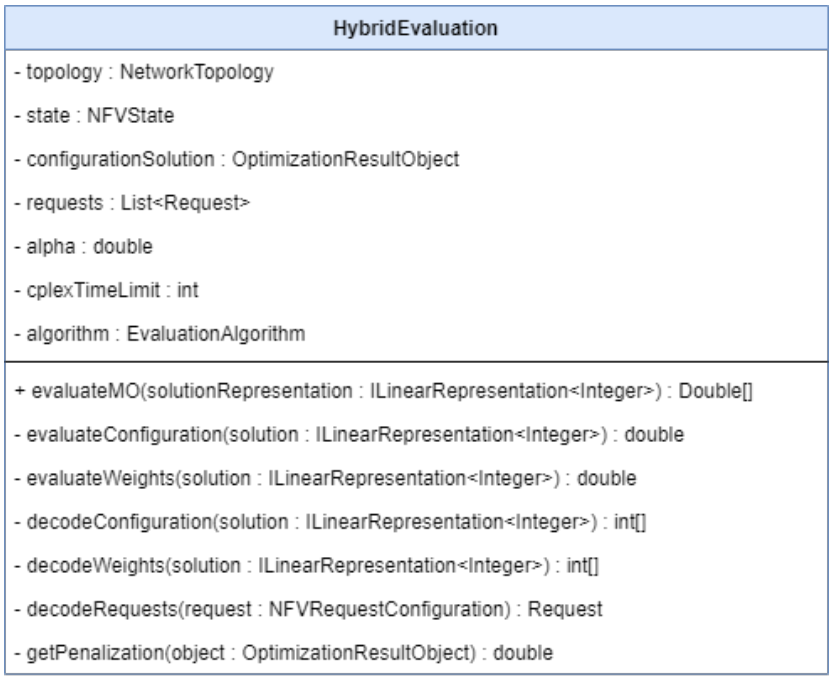


Figura 4.16: Classe de avaliação de indivíduos do processo de otimização dos pesos IGP e da distribuição de serviços.

Caso a solução gerada pelo algoritmo evolucionário represente uma configuração topológica válida, o atributo *configurationSolution* é inicializado e atualizado com a solução obtida pelo modelo linear. A partir deste atributo é obtida a lista de pedidos necessária para utilizar o simulador da *framework* através do método *decodeRequests*. Após compilada a lista de pedidos, a segunda parte da solução é descodificada e convertida em pesos IGP. Estes são adicionados à topologia e é iniciada a avaliação dos pesos através do método *evaluateWeights*.

4.5 processo de otimização com recurso a machine learning

A utilização de *machine learning* tem como principal objetivo providenciar respostas em tempo real. Ou seja, cada pedido que dá entrada na rede é analisado e face ao estado da topologia é sugerido um caminho SR para o mesmo.

4.5.1 *Construção do Conjunto de Dados*

Para treinar o modelo foi desenvolvido em Java um algoritmo que face às necessidades de um pedido e tendo em conta o estado de utilização das ligações da topologia e dos nodos, utiliza um modelo MILP para gerar os caminhos SR. A Figura 4.17 representa o diagrama

de classes relativo à geração do conjunto de dados. Cada entrada, ou linha, do conjunto de dados é representada pela classe *DataSetEntry*.

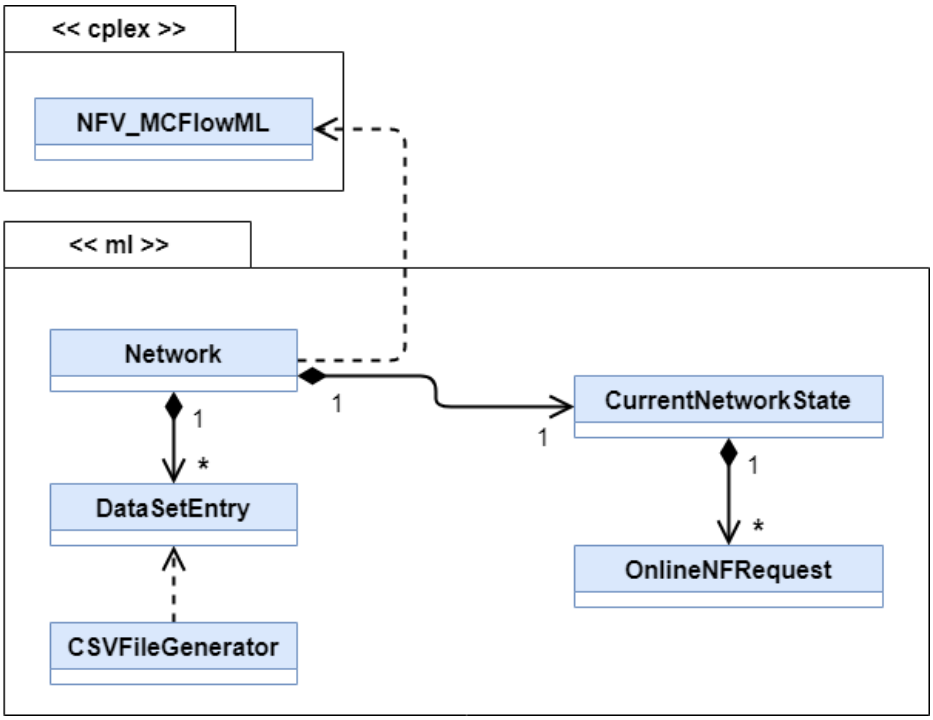


Figura 4.17: Diagrama de classes do módulo de geração do conjunto de dados.

A rede é representada pela classe *Network* (figura 4.18). Esta engloba o estado atual dos recursos da topologia que possui um conjunto de pedidos ativos.

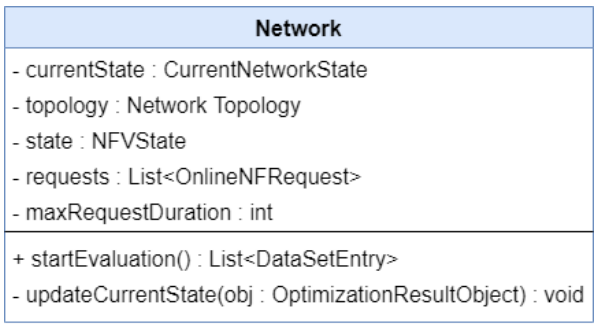


Figura 4.18: Classe representativa de uma rede.

A classe *DataSetEntry* (figura 4.19) representa uma entrada no conjunto de dados. Fazem parte desta os nodos origem e destino do pedido, a largura de banda estimada, a duração do pedido, os serviços requisitados, o estado das ligações e dos nodos no momento do pedido e, por fim, o resultado que indica em que nodo cada serviço foi executado.

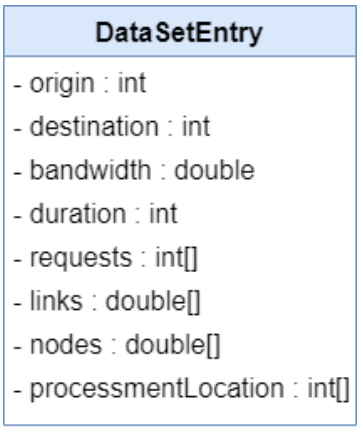


Figura 4.19: Classe representativa de uma entrada no conjunto de dados.

A duração de um pedido é especificada por um número inteiro. Este é decrementado a cada iteração que consiste numa avaliação do estado da rede por parte do MILP.

O MILP avalia um pedido de cada vez, incluindo, todos os restantes pedidos que ainda estão com o valor da variável de instância *duration* maior de que 0. Os pedidos que ainda se encontram em execução estão inseridos na classe *CurrentNetworkState* (figura 4.20).

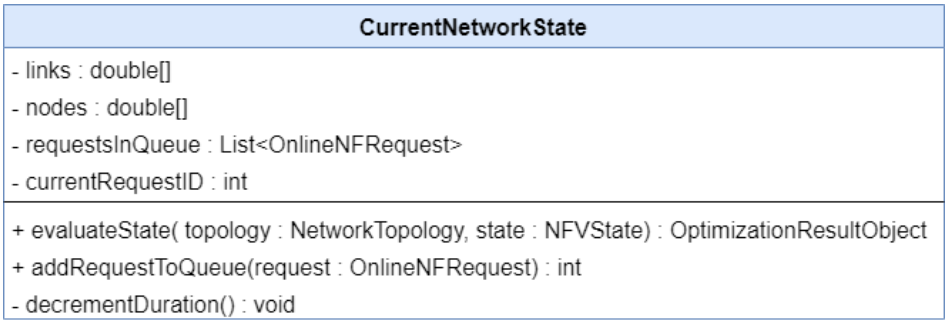


Figura 4.20: Classe representativa do estado atual da rede.

Esta inclui, para além da *queue* com os pedidos em execução, o estado da topologia antes da execução do pedido cujo ID é representado pela variável de instância *currentRequestID*. Após a avaliação do MILP, através do objeto *OptimizationResultObject* são atualizadas as taxas de utilização das ligações e dos nodos da topologia no *CurrentNetworkState* de forma a preparar a avaliação do próximo pedido a ser inserido.

A classe relativa à função de avaliação dos pedidos está representado na Figura 4.22. De forma a garantir que os pedidos que ainda estão na lista mantenham o nodo de execução desde a primeira até à ultima iteração, os caminhos SR gerados para cada pedido são guardados na classe *OnlineNFRequest* (figura 4.21).

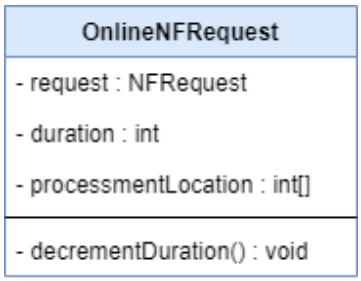


Figura 4.21: Classe representativa de um pedido de encaminhamento.

Os locais de execução são mantidos nas várias iterações com recurso à variável binária $\alpha_{i,k}^{n}$. Através desta, todos os restantes nodos para o pedido e serviço em questão são igualados a 0, impossibilitando assim a execução do serviço num outro nodo.

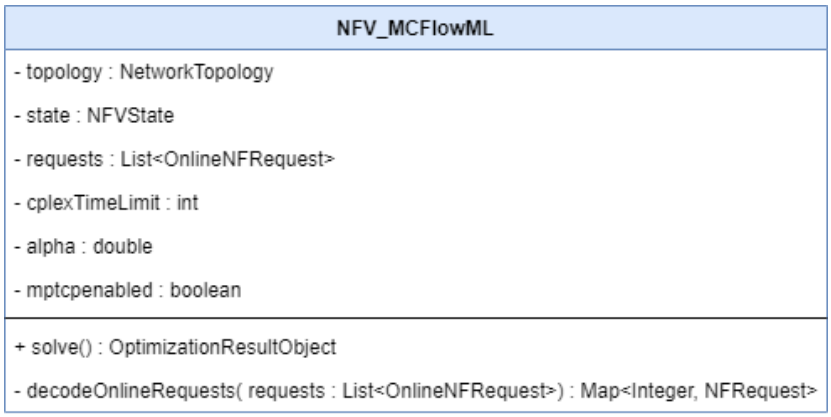


Figura 4.22: Classe de avaliação de indivíduos para construção do conjunto de dados.

Após ter sido concluída a avaliação da entrada no conjunto de dados e obtidos os nodos nos quais os serviços são executados, a entrada é de seguida acrescentada a um ficheiro em formato csv. A partir da variável de instância *binaryOutput* presente na classe *CSVFileGenerator* é possível definir de que forma o *output* é guardado. Esta opção tem como principal objetivo auxiliar no processo de depuração de erros, mostrando o *output* com o ID do nodo no qual o serviço requisitado foi executado.

4.5.2 *Modelos de Classificação*

A implementação dos modelos de classificação utilizados foi efetuada com recurso à linguagem de programação Python. Esta é adequada para o desenvolvimento de modelos de *machine learning* devido às inúmeras bibliotecas como *pandas*, *keras*, *sklearn*, entre outras, que auxiliam nas diversas etapas para a aplicação deste método de otimização.

As etapas implementadas foram as seguintes:

- Visualização e tratamento dos dados
  - Visualização dos dados
  - Identificação das *features* (*input*) e do *target* (*output*)
  - Tratamento dos dados através da seleção de colunas
  - Divisão do conjunto de dados
- Treino e teste dos modelos
  - *Random Forest Classifier*
  - SVM
  - *Deep Neural Network*
  - LSTM
- Avaliação dos modelos

Para determinar a configuração da rede neuronal mais adequada ao problema foi utilizado o método de *grid search*. Através deste é possível combinar um conjunto de parâmetros e retirar a configuração que obteve melhor classificação relativamente à *accuracy* de validação. Como alternativa a este método poderiam ser utilizados algoritmos evolucionários. Cada inteiro presente na solução iria corresponder a um parâmetro na configuração/estruturação da rede neuronal.

Para além da *accuracy* são utilizadas outras métricas para avaliar o desempenho dos modelos utilizados. Nestas incluem-se:

- *Precision* - Indica a percentagem de valores positivos corretamente previstos no total de observações positivas. Ou seja, quão frequente é que o valor previsto está de facto correto. É obtida a partir da seguinte equação:

$$Precision = \frac{TruePositives}{TruePositives+FalsePositives}$$

- *Recall* (*sensitivity*) - Indica a percentagem de valores previstos no total de observações. Ou seja, indica a proporção de respostas corretas. É dada pela seguinte equação:

$$Recall = \frac{TruePositives}{TruePositives+FalseNegatives}$$

- *f1-score* - Indica a média aritmética ponderada dos valores das métricas *precision* e *recall*. É dada pela seguinte equação:

$$f1score = \frac{2*(Recall*Precision)}{Recall+Precision}$$

4.6 SUMÁRIO

Neste capítulo foram abordados os módulos desenvolvidos e de que forma estes foram integrados na *framework* NetOpt. Este processo de integração implicou uma análise ao código fonte já desenvolvido na mesma. Assim foi possível identificar de que forma é que poderia ser feita a integração, que contou com a adição de novos conceitos associados ao NFV. Estes conceitos possibilitaram a implementação dos modelos lineares definidos na Secção 3.2. Foram descritos todos os processos de otimização que utilizaram os modelos lineares, uma solução híbrida que combina os algoritmos evolucionários com os modelos e, por fim, um modelo de *machine learning* capaz de dar resultados em cenários de tempo real.

5

ANÁLISE DE RESULTADOS

Neste capítulo são apresentados os cenários experimentais definidos para a otimização dos recursos da rede, bem como a análise aos resultados obtidos. Nesse sentido, a Secção 5.1 apresenta as topologias utilizadas e as configurações usadas nas experiências. A Secção 5.2 apresenta os resultados obtidos para otimização dos problemas MCF. A Secção 5.3 apresenta os resultados das otimizações com recurso aos algoritmos evolucionários. Por fim, a Secção 5.4, engloba o processo de construção do conjunto de dados e analisa os resultados obtidos com recurso a *machine learning*.

5.1 DESCRIÇÃO DOS CENÁRIOS EXPERIMENTAIS

A otimização de recursos de rede, de transporte e processamento de tráfego, requer um elevado poder computacional. Consequentemente, grande parte das experiências foram realizadas num *High Performance Computing Cluster* do departamento de informática da Universidade do Minho [37]. A utilização deste ambiente agiliza o processo de testes dado permitir a execução simultânea de várias tarefas em paralelo, cada utilizando um número considerável de *cores*. Esta particularidade é principalmente relevante na execução dos algoritmos evolucionários.

Na Tabela 5.1 estão identificadas as topologias utilizadas nos diversos cenários experimentais.

Topologia	Número de Nodos	Capacidade dos Nodos (Gbps)	Número de Ligações	Capacidade das Ligações (Gbps)
AbileneLC	11	2	28	1,4
Abilene		2,5		10
BTEuropeLC	24	1	72	0,75
BTEurope		2,5		2,5
30_2LC	30	0,8	110	$0,3 - 1,5$
30_2		2,5		$2,5 - 12$

Tabela 5.1: Características das topologias utilizadas.

A topologia 30_2 e *Abilene* estão disponíveis na *framework* NetOpt, sendo a última bastante utilizada em trabalhos de engenharia de tráfego. A topologia *BT Europe* foi retirada do *Topology Zoo* e representa uma topologia real. Todas as topologias são modeladas como grafos direcionados.

Nas topologias *Abilene* e *BT Europe*, as capacidades das ligações são constantes. A capacidade das ligações da topologia *Abilene* está definida nos ficheiros que representam a mesma. Por sua vez, a topologia *BT Europe*, não continha informações acerca das capacidades, pelo que foi definido um valor fixo de 2.5 Gbps para cada ligação. Relativamente à topologia 30_2, apresenta ligações com capacidade entre 2.5 Gbps e 12 Gbps estando estes valores também definidos nos ficheiros de representação da mesma. As topologias cujo nome inclui a designação "LC"são semelhantes às restantes, no entanto, possuem uma capacidade das ligações e dos nodos reduzida. Com estas topologias de menor capacidade pretende-se analisar cenários de maior utilização, tanto das ligações como dos nodos.

Os nodos de todas as topologias podem ou não possuir capacidade de processamento. Esta capacidade é definida através da existência de serviços disponíveis nos mesmos. As experiências executadas nesta topologia tiveram em consideração nodos com 2.5 Gbps de capacidade de processamento.

Com o uso destas topologias pretende-se analisar a distribuição dos pedidos pelas ligações e pelos nodos das mesmas, em cenários de menor ou maior complexidade, com a finalidade de averiguar o comportamento dos modelos lineares e dos simuladores da *framework*. Cada topologia será avaliada face a dois ficheiros com pedidos de encaminhamento de tráfego, um com 300 e o outro com 1200 pedidos, com exceção das topologias com a designação "LC"que apenas serão avaliadas face a 1200 pedidos de encaminhamento. Cada pedido de encaminhamento pode assumir um valor de requisito de largura de banda entre 1 Mbps e 12 Mbps e requisitar qualquer conjunto de serviços, devidamente ordenados pela ordem pretendida. Assumiu-se a existência de três serviços distintos podendo estes estar disponíveis em qualquer nodo da topologia. Cada nodo pode disponibilizar qualquer combinação de serviços e cada serviço possui um custo de execução associado.

Com o objetivo de obter um bom intervalo de confiança nos métodos em que foram utilizados algoritmos evolucionários, foram realizadas quinze execuções por algoritmo. Os resultados foram posteriormente tratados e agregados sendo de seguida analisados através de um programa desenvolvido em Python, com recurso à biblioteca *MatPlotLib* [18], para gerar gráficos, como por exemplo *box-plots*. Por motivos de simplificação, os modelos lineares definidos vão ser identificados neste capítulo como: Modelo Φ no caso do MILP cujas funções objetivo são baseadas na função Φ; Modelo MLU no caso do MILP cujas funções objetivo são baseadas na função MLU.

5.2 OTIMIZAÇÃO COM RECURSO AOS MODELOS LINEARES

Nesta secção vai ser abordado o processo de otimização da localização dos serviços com recurso aos modelos lineares. Primeiramente é analisado o tempo de execução de cada modelo nas topologias a utilizar. É também efetuada uma análise aos resultados obtidos a partir do processo de otimização, que utiliza somente os modelos lineares, para determinar a localização dos serviços.

5.2.1 *Análise ao Tempo de Execução dos Modelos Lineares*

O tempo necessário para avaliar uma solução é importante para definir um limite expectável para o tempo de execução dos algoritmos evolucionários. Nesse sentido, foi efetuada uma análise ao comportamento dos modelos realizando otimizações em cada topologia. O tempo computacional necessário para resolver cada MILP, variando em número de pedidos e topologia, é apresentado na Tabela 5.2.

Topologia	Número de Pedidos	Modelo MILP	Tempo de Execução (s)	*Gap* (%)
AbileneLC	1200	Φ	23	0,13
		MLU	30	0,13
Abilene	300	Φ	7	0
		MLU	10	0,43
	1200	Φ	13	0
		MLU	30	0,3
BT EuropeLC	1200	Φ	62	0,32
		MLU	90	0,49
BT Europe	300	Φ	11	0
		MLU	22	2,88
	1200	Φ	44	0
		MLU	125	1,53
30_2LC	1200	Φ	130	0,59
		MLU	160	1,12
30_2	300	Φ	13	0
		MLU	35	3,49
	1200	Φ	36	0
		MLU	185	1,27

Tabela 5.2: Comparação dos tempos de execução dos modelos lineares.

Para além do tempo de execução, é também importante assegurar a capacidade dos modelos atingirem a solução ótima face ao conjunto de pedidos fornecido. A ferramenta Cplex permite aplicar um tempo limite para a relaxação e obtenção de soluções dos modelos. Esta

característica é particularmente relevante para impedir que o tempo de avaliação de uma solução do algoritmo evolucionário seja demasiado elevado. Isto porque, face ao número de indivíduos e gerações a considerar bem como o número de testes a realizar, iria implicar que a utilização deste método de otimização fosse inviável.

Nesse sentido, para esta análise, o tempo máximo de execução foi limitado a 500 segundos. Os resultados apresentados representam o instante em que foi obtido o menor intervalo para a solução ótima (*gap*) dentro do período definido. Um *gap* acima de 0 indica que poderá haver uma solução melhor face à obtida até ao momento. Como é possível observar, quanto maior a complexidade da topologia, maior é o tempo de execução necessário para obter a solução ótima. No entanto, face a um número baixo de pedidos, o *gap* acaba por ser superior. Isto deve-se ao facto de que, como os níveis de utilização obtidos são praticamente nulos, a tarefa de melhorar a solução obtida com base na função objetivo torna-se difícil. O processo de otimização através do modelo MLU obteve *gap* sempre superior ao do modelo Φ. Apesar de os modelos serem matematicamente parecidos, existem diferenças nas equações de avaliação da utilização dos nodos e das ligações. No modelo Φ, ao contrário do MLU, é aplicada uma penalização sempre que a taxa de utilização do nodo ou da ligação ultrapassa um determinado valor. A introdução de penalizações permite ao modelo Φ identificar situações de elevada utilização dos nodos e ou das ligações, o que agiliza o processo de relaxação da solução, visto que, o espaço de soluções diminui. Assim, o modelo Φ tem como objetivo minimizar a média das penalizações. Por sua vez, o modelo MLU, tem como objetivo minimizar a taxa máxima de utilização.

Em suma, a análise ao tempo de execução permite determinar o período médio de avaliação de uma configuração topológica. Esta análise é relevante para limitar o tempo de avaliação de uma solução gerada pelos algoritmos evolucionários, sem comprometer a qualidade da solução encontrada. É expectável que o tempo de execução dos modelos seja inferior ao indicado na Tabela 5.2 visto que, a análise considera uma distribuição total dos serviços pelos nodos da topologia. Esta distribuição aumenta consideravelmente o número de variáveis para representar a configuração topológica o que, por sua vez, aumenta a complexidade dos modelos, prejudicando o desempenho dos mesmos.

5.2.2 *Análise à Otimização da Distribuição dos Serviços com os Modelos Lineares*

O processo de otimização da localização dos serviços com recurso a modelos lineares implica a utilização do gerador de pedidos para obter configurações de um estado *NFVState*. Isto porque, pretende-se que, para esta análise, todos os nodos disponibilizem todos os serviços com o objetivo de que os modelos definam indiretamente a localização dos mesmos. Nesse sentido, os modelos vão poder determinar a solução ótima para cada pedido de

encaminhamento visto que não são aplicadas restrições à execução dos serviços nos nodos (através das equações 3.3 e 3.25).

Os resultados obtidos permitem determinar a localização de execução de cada serviço. Para além disso, é possível determinar quais são os nodos com maior taxa de utilização. Como tal, em configurações futuras, poderiam ser disponibilizados mais recursos para esses nodos com o objetivo de dar resposta às necessidades de encaminhamento.

De seguida, para cada topologia, são apresentados os resultados obtidos para ambos os modelos lineares definidos nos quais foi considerado a utilização de MPTCP, ou seja, o tráfego de um mesmo pedido pode seguir por caminhos diferentes.

Topologia AbileneLC

As Figuras 5.1 e 5.2 representam as taxas de utilização obtidas na topologia AbileneLC através dos modelos MLU e Φ respetivamente.

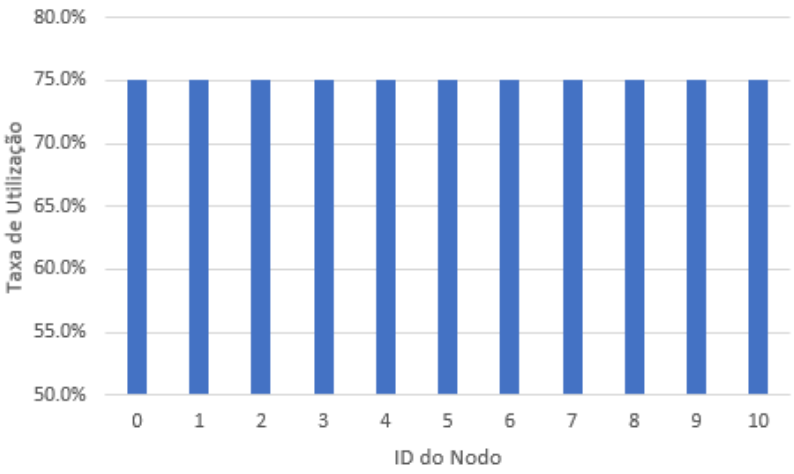


Figura 5.1: Taxas de utilização por nodo na AbileneLC com o modelo MLU com 1200 pedidos.

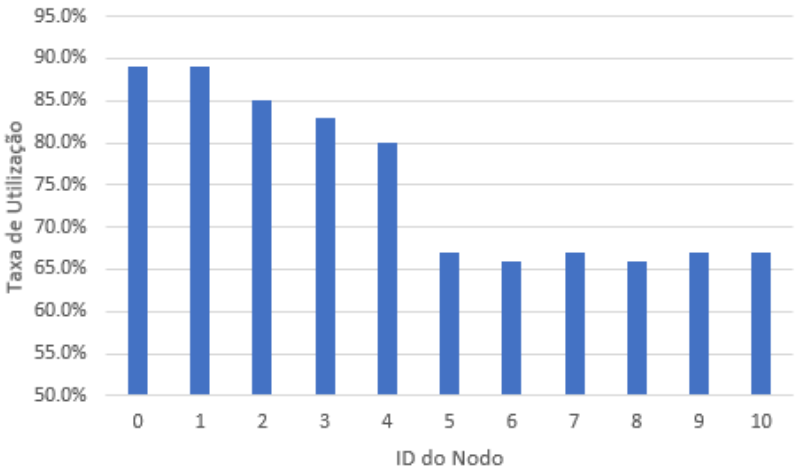


Figura 5.2: Taxas de utilização por nodo na AbileneLC com o modelo Φ com 1200 pedidos.

A Tabela 5.3 representa as taxas de utilização média dos nodos e das ligações na topologia AbileneLC. A variabilidade da utilização dos nodos com o modelo MLU é menor do que a observada para o modelo Φ, todavia as médias são equivalentes. No que respeita à utilização das ligações, o modelo Φ tem um melhor desempenho. Relativamente aos nodos, as taxas de utilização são semelhantes, o que comprova a boa capacidade dos modelos de distribuir o tráfego pelos nodos da topologia.

Topologia	Utilização média das ligações (%)		Utilização média dos nodos (%)	
	Modelo MLU	Modelo Φ	Modelo MLU	Modelo Φ
Abilene LC	67	45,8	75	75,1

Tabela 5.3: Taxas médias de utilização na AbileneLC com todos os serviços disponíveis.

Topologia Abilene

Nas Figuras 5.3 e 5.4 estão representadas as taxas de utilização dos nodos da topologia *Abilene* para 300 pedidos de encaminhamento para os modelos MLU e Φ respetivamente.

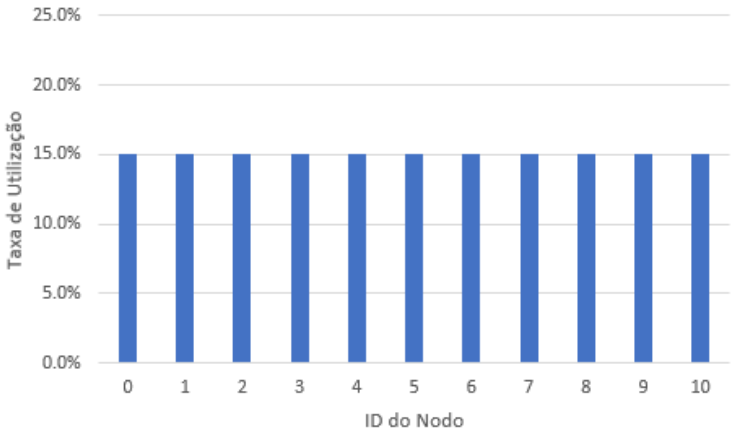


Figura 5.3: Taxas de utilização por nodo na Abilene com o modelo MLU com 300 pedidos.

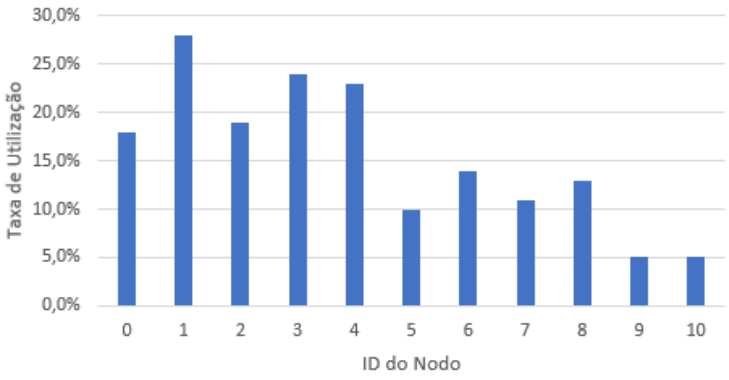


Figura 5.4: Taxas de utilização por nodo na Abilene com o modelo Φ com 300 pedidos.

As taxas de utilização obtidas com o modelo MLU foram semelhantes em todos os nodos. Com o modelo Φ existem nodos com uma taxa de utilização acima dos 20%. Apesar disso, a penalização aplicada neste caso é insignificante. Isto levou a que existissem outros nodos com taxas de utilização abaixo dos 5%. Nesse sentido, seria de prever que em futuras análises, os nodos com o ID 9 e 10, pudessem ser descartados como detentores de serviços. Em alternativa, poderiam ser alocados menos recursos a estes nodos.

As Figuras 5.5 e 5.6 representam as taxas de utilização obtidas nos nodos da topologia *Abilene* para 1200 pedidos de encaminhamento.

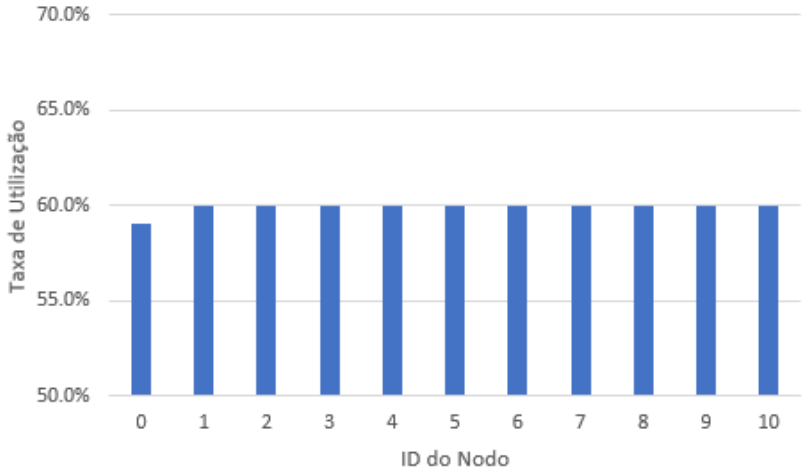


Figura 5.5: Taxas de utilização por nodo na Abilene com o modelo MLU com 1200 pedidos.

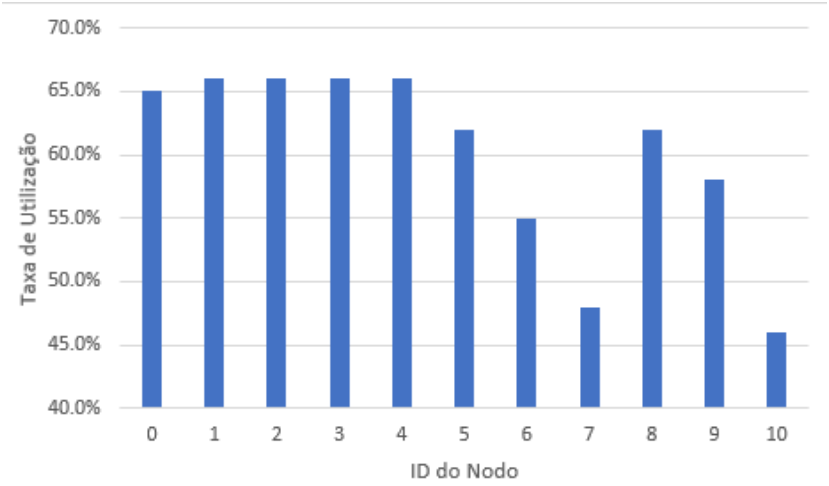


Figura 5.6: Taxas de utilização por nodo na Abilene com o modelo Φ com 1200 pedidos.

Novamente, as taxas de utilização no modelo MLU foram semelhantes em todos os nodos. O modelo Φ apresentou uma taxa de utilização a rondar os 65% em metade dos nodos da topologia. A Tabela 5.4 representa as taxas de utilização médias dos nodos e das ligações obtidas. Tal como na topologia anterior, as taxas de utilização médias das ligações no modelo Φ foram inferiores.

Topologia	Número de Pedidos	Utilização média das ligações (%)		Utilização média dos nodos (%)	
		Modelo MLU	Modelo Φ	Modelo MLU	Modelo Φ
Abilene	300	2	1,3	15	15,4
	1200	9	6	60	60

Tabela 5.4: Taxas médias de utilização na Abilene com todos os serviços disponíveis.

Topologia BT EuropeLC

Nas Figuras 5.7 e 5.8 estão representadas as taxas de utilização dos nodos, para a topologia BT EuropeLC, com os modelos MLU e Φ respetivamente.

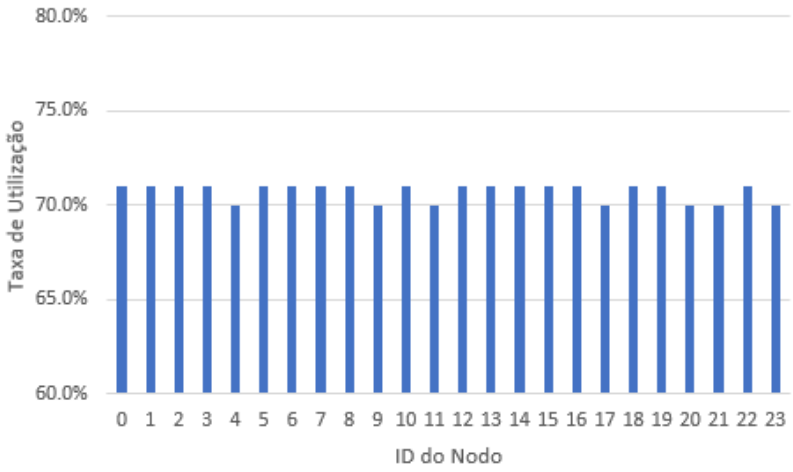


Figura 5.7: Taxas de utilização por nodo na BT EuropeLC com o modelo MLU com 1200 pedidos.

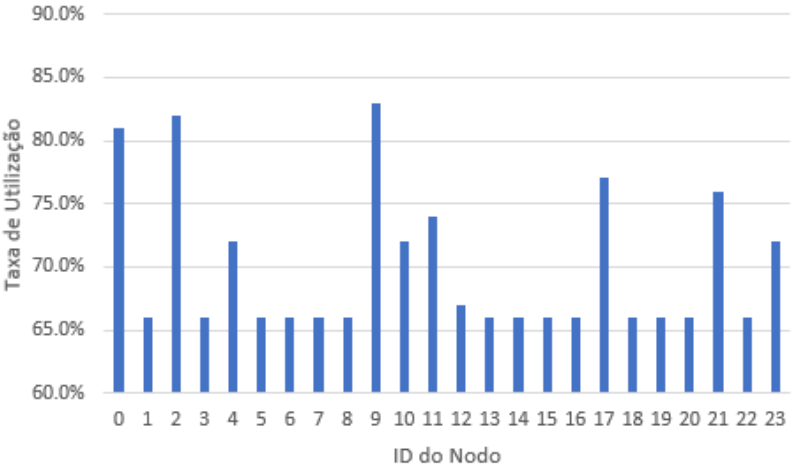


Figura 5.8: Taxas de utilização por nodo na BT EuropeLC com o modelo Φ com 1200 pedidos.

Tal como no caso da topologia AbileneLC, o modelo MLU manteve uma distribuição equitativa da carga pelos nodos da topologia. O modelo Φ atribuiu a alguns nodos uma carga superior, no entanto, sempre abaixo dos 83%.

A Tabela 5.5 representa a taxa de utilização média das ligações e dos nodos na topologia *BT EuropeLC*. Novamente, a taxa de utilização das ligações é inferior no modelo Φ.

Topologia	Utilização média das ligações (%)		Utilização média dos nodos (%)	
	Modelo MLU	Modelo Φ	Modelo MLU	Modelo Φ
BT EuropeLC	56	35	71	71

Tabela 5.5: Taxas médias de utilização na BT EuropeLC com todos os serviços disponíveis.

Topologia BT Europe

Nas Figuras 5.9 e 5.10 estão representadas as taxas de utilização dos nodos na topologia *BT Europe* para 300 pedidos de encaminhamento. O modelo MLU apresenta novamente uma taxa de utilização muito semelhante em todos os nodos. Contrariamente, o modelo Φ permite distinguir os nodos nos quais foram ou não executados serviços. A taxa de utilização máxima atingida foi cerca de 34% em dois dos nodos da topologia. Existiram também nodos que nos quais não foram executados serviços pelo que nestes poderia ser descartada a instalação dos mesmos.

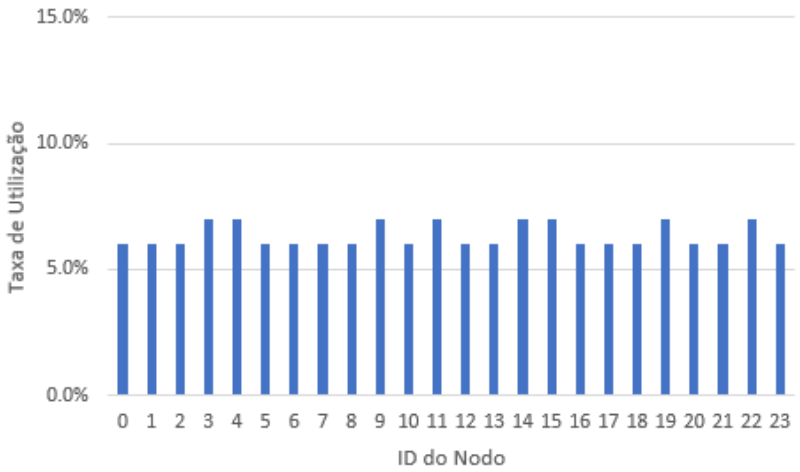


Figura 5.9: Taxas de utilização por nodo na BT Europe com o modelo MLU com 300 pedidos.

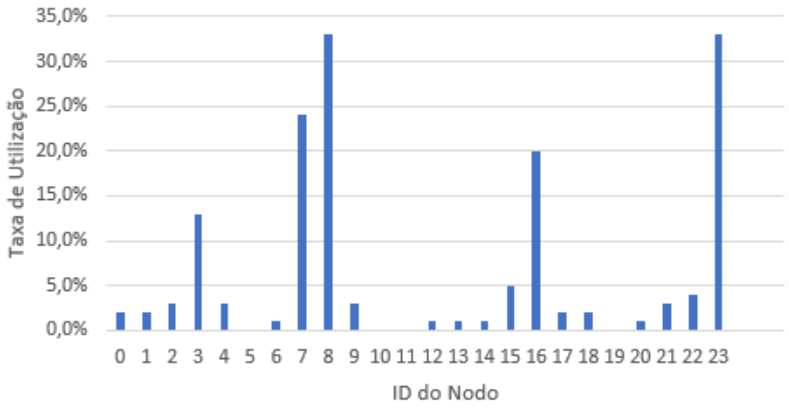


Figura 5.10: Taxas de utilização por nodo na BT Europe com o modelo Φ com 300 pedidos.

Nas Figuras 5.11 e 5.12 estão representadas as taxas de utilização por nodo para o ficheiro com 1200 pedidos de encaminhamento. Novamente, o modelo MLU apresentou taxas de utilização semelhantes em todos os nodos da topologia. O modelo Φ apresenta mais nodos com taxas de utilização próximas dos 34% comparativamente com a execução representada pela Figura 5.10. Naturalmente, mais nodos foram utilizados permitindo assim distribuir a carga pelos restantes nodos da topologia que não estavam a ser utilizados.

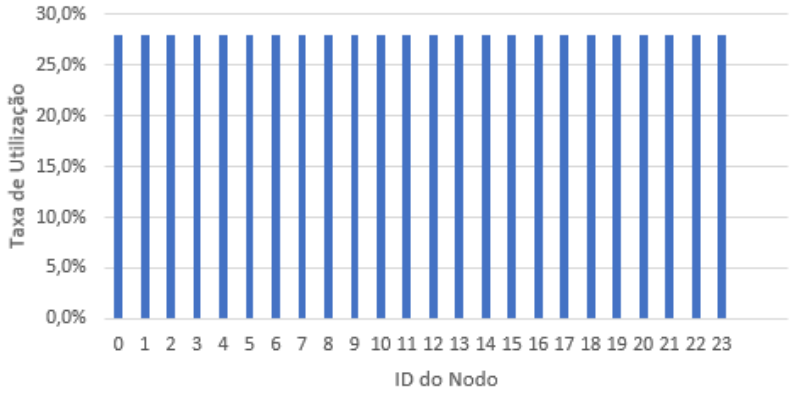


Figura 5.11: Taxas de utilização por nodo na BT Europe com o modelo MLU com 1200 pedidos.

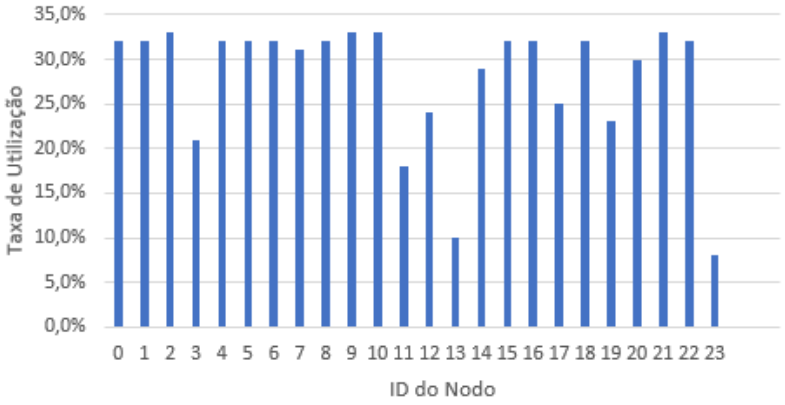


Figura 5.12: Taxas de utilização por nodo na BT Europe com o modelo Φ com 1200 pedidos.

A Tabela 5.6 representa a taxa de utilização média dos nodos e das ligações no topologia *BT Europe*. Tal como nas topologias anteriores, a taxa de utilização das ligações é inferior no modelo Φ.

Topologia	Número de Pedidos	Utilização média das ligações (%)		Utilização média dos nodos (%)	
		Modelo MLU	Modelo Φ	Modelo MLU	Modelo Φ
BT Europe	300	4	2	6	7
	1200	15,4	10	28	28

Tabela 5.6: Taxas médias de utilização na BTEurope com todos os serviços disponíveis.

Topologia 30_2LC

Nas Figuras 5.13 e 5.14 estão representadas as taxas de utilização dos nodos, para a topologia 30_2LC, com os modelos MLU e Φ respetivamente. Novamente, o modelo MLU possui uma carga de utilização superior em quase todos os nodos. Em contra partida, o modelo Φ apresenta alguns nodos com uma taxa de utilização perto dos 80% o que, por sua vez, não ultrapassa a capacidade de processamento dos mesmos.

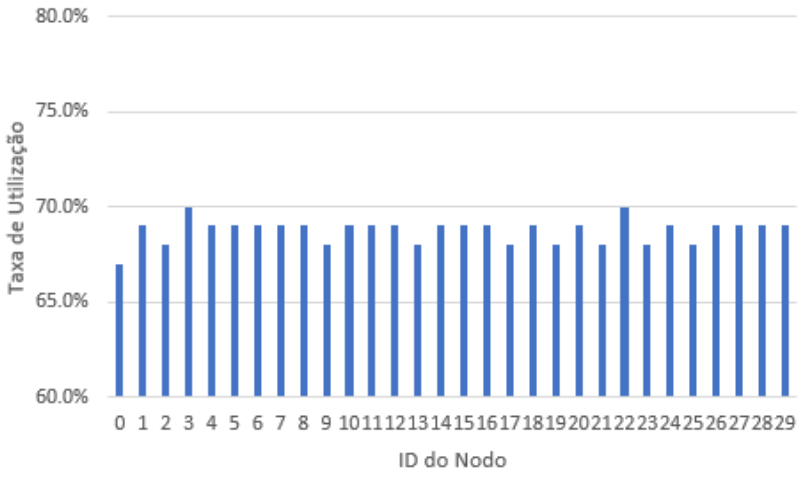


Figura 5.13: Taxas de utilização por nodo na 30_2LC com o modelo MLU com 1200 pedidos.

A Tabela 5.7 representa a taxa de utilização média dos nodos e das ligações para a topologia 30_2LC. Novamente, as taxas de utilização das ligações são inferiores no modelo Φ.

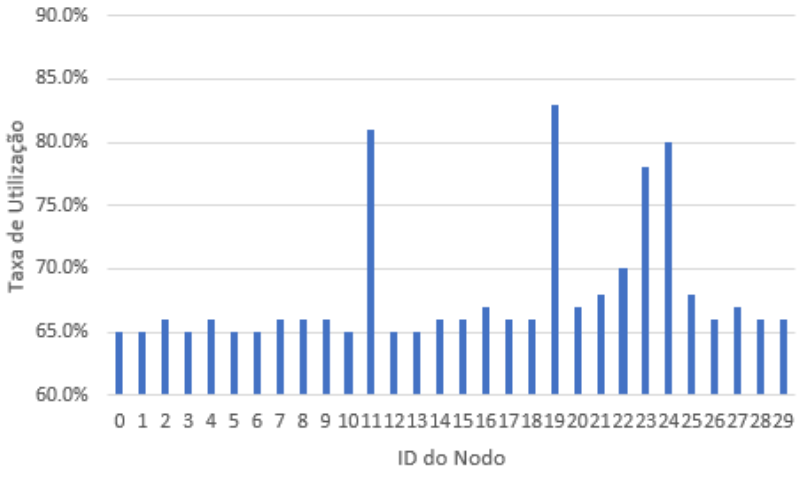


Figura 5.14: Taxas de utilização por nodo na 30_2LC com o modelo Φ com 1200 pedidos.

Topologia	Utilização média das ligações (%)		Utilização média dos nodos (%)	
	Modelo MLU	Modelo Φ	Modelo MLU	Modelo Φ
30_2LC	33	24	69	69

Tabela 5.7: Taxas médias de utilização na 30_2LC com todos os serviços disponíveis.

Topologia 30_2

Nas Figuras 5.15 e 5.16 estão representadas as taxas de utilização dos nodos na topologia 30_2 para 300 pedidos de encaminhamento. As taxas de utilização no modelo MLU foram semelhantes em todos os nodos da topologia. No modelo Φ é possível identificar os nodos com uma maior taxa de utilização. No entanto, não foi aplicada uma penalização nos mesmos dado que a taxa de utilização é inferior a 30%.

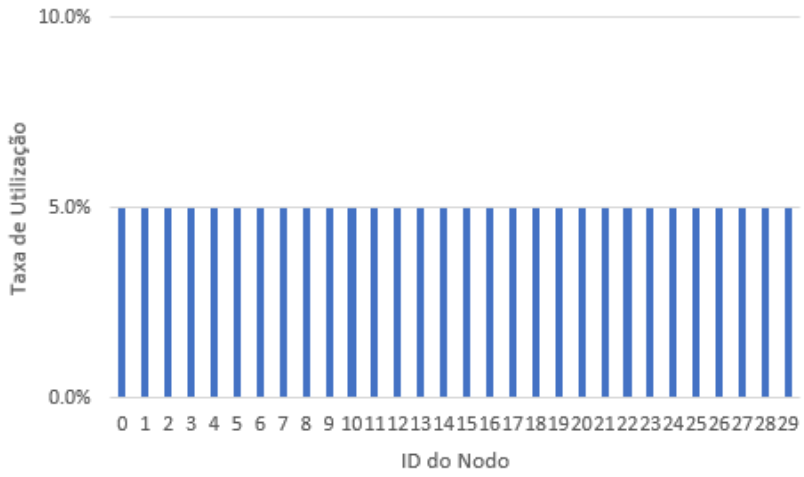


Figura 5.15: Taxas de utilização por nodo na 30_2 com o modelo MLU com 300 pedidos.

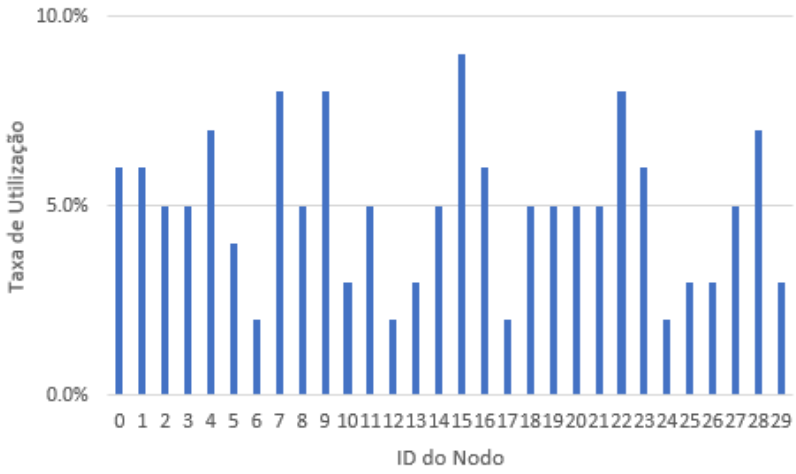


Figura 5.16: Taxas de utilização por nodo na 30_2 com o modelo Φ com 300 pedidos.

Nas Figuras 5.17 e 5.18 estão representadas as taxas de utilização para 1200 pedidos de encaminhamento. Tal como nos testes anteriores, o modelo MLU apresenta taxas de utilizações semelhantes para todos os nodos da topologia. O modelo Φ voltou a apresentar

uma taxa máxima de utilização de 34% evitando assim penalizações elevadas ao distribuir o tráfego por outros nodos da topologia.

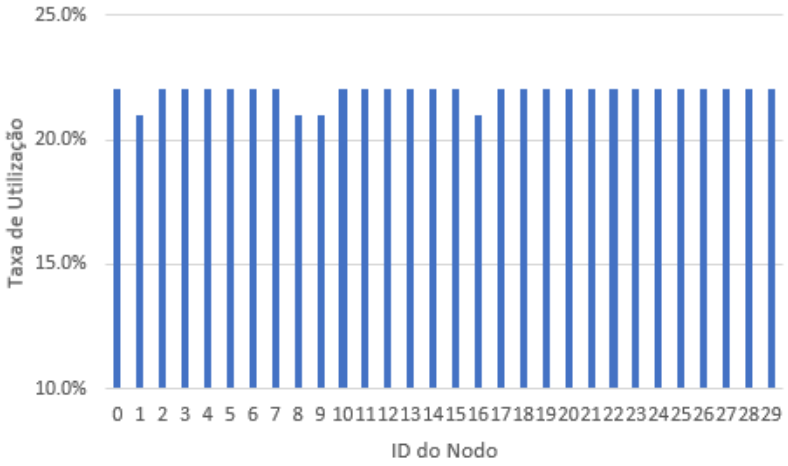


Figura 5.17: Taxas de utilização por nodo na 30_2 com o modelo MLU com 1200 pedidos.

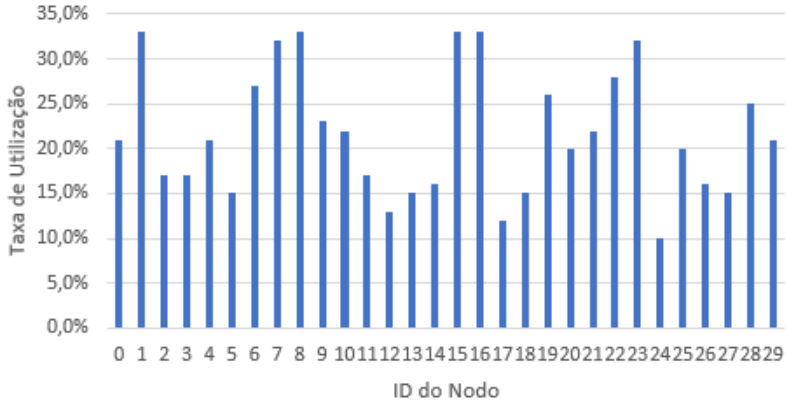


Figura 5.18: Taxas de utilização por nodo na 30_2 com o modelo Φ com 1200 pedidos.

A Tabela 5.8 representa a taxa de utilização média dos nodos e das ligações da topologia 30_2.

Topologia	Número de Pedidos	Utilização média das ligações (%)		Utilização média dos nodos (%)	
		Modelo MLU	Modelo Φ	Modelo MLU	Modelo Φ
30_2	300	0,7	0,4	5	5
	1200	4	5	22	22

Tabela 5.8: Taxas médias de utilização na topologia 30_2 com uma distribuição total dos serviços.

Apesar de ser possível identificar quais os nodos que teriam serviços instalados, num contexto real, este método não permite garantir uma boa gestão dos recursos. Isto deve-se ao facto de ser necessário disponibilizar os mesmos serviços num grande número de nodos visto que todos os serviços acabam por ser utilizados. No entanto, tal como foi referido, através da variável binária $\alpha_{i,k}^{n}$, seria possível impedir a execução de serviços em um ou mais nodos da topologia. Apesar disso, este cenário implicaria introduzir manualmente as restrições nos modelos. Em ambas as abordagens não é possível determinar o impacto que a remoção dos nodos como detentores de um ou mais serviços teria na taxa de utilização sem voltar a executar novamente o modelo. De forma geral, ambos os modelos conseguem garantir uma utilização dos nodos abaixo das capacidades dos mesmos. Relativamente às ligações, o modelo Φ consegue garantir uma taxa de utilização inferior face ao modelo MLU. Um fator que poderá contribuir para a discrepância de valores é o *gap* para a solução ótima que o modelo MLU não consegue reduzir a 0 em alguns cenários da análise. No entanto, é importante referir que para o caso da Topologia *AbileneLC* na qual o *gap* para a solução ótima em ambos os modelos foi 0,13%, a taxa de utilização das ligações foi substancialmente inferior no modelo Φ.

5.2.3 *Comparação dos Níveis de Utilização entre os Modelos Lineares*

Nesta subsecção vão ser analisados os resultados obtidos pelos MILP, considerando ainda uma distribuição total dos serviços. Para isso, vão ser utilizados dois modelos lineares já definidos na *framework*, que tem como objetivo a minimização dos níveis de utilização das ligações da topologia para as métricas MLU e Φ. Estes modelos não têm em consideração a execução de serviços nos nodos pelo que, os níveis de utilização das ligações dizem respeito ao encaminhamento desde a sua origem até ao destino, sem restrições de balanceamento ou protocolo (MCF). Assim, os níveis de utilização das ligações obtidos a partir dos modelos mencionados representam a melhor solução possível para os níveis de utilização na rede e servem de base para avaliar os modelos e algoritmos definidos nesta dissertação. Os valores referentes aos níveis de utilização face aos pedidos de encaminhamento sem execução de serviços estão representados na Tabela 5.9.

Topologia	Número de Pedidos	MLU (s/Serviços)	Φ (s/Serviços)
AbileneLC	1200	0,742	1,801
Abilene	300	0,027	1,0
	1200	0,103	1,0
BT EuropeLC	1200	0,604	1,333
BT Europe	300	0,043	1,0
	1200	0,181	1,0
30_2LC	1200	0,405	1,058
30_2	300	0,012	1,0
	1200	0,054	1,0

Tabela 5.9: Níveis de utilização das ligações para pedidos de encaminhamento sem execução de serviços.

Na Tabela 5.10 são apresentados os níveis de utilização das ligações com base nas métricas MLU e Φ para três cenários distintos. O primeiro cenário, representado na coluna "Modelo Otimizado", considera o resultado da otimização por parte dos modelos MLU e Φ definidos nesta dissertação nos quais os caminhos SR são gerados pelos mesmos. O segundo cenário, representado na coluna "Caminhos SR Aleatórios", considera caminhos aleatórios para cada pedido de encaminhamento, ou seja, os serviços são executados num nodo aleatório. Por fim, o último cenário representa o pior caso. Neste, cada serviço está disponível em apenas um nodo da topologia. Naturalmente é possível considerar este cenário como o pior caso, visto que, todos os pedidos que requerem serviços necessitam de passar pelos nodos que possuem os serviços requisitados pela ordem correta. Isto leva a que os pedidos necessitem de efetuar muitos saltos antes de chegar ao destino, aumentando assim os níveis de utilização das ligações.

Topologia	Número de Pedidos	Modelo Otimizado		Caminhos SR Aleatórios		Pior Caso	
		MLU	Φ	MLU	Φ	MLU	Φ
AbileneLC	1200	0,742	1,801	2,159	1439,5	2,691	2436,2
Abilene	300	0,027	1,0	0,08	1,0	0,098	1,0
	1200	0,103	1,0	0,313	1,0	0,376	1,06
BT EuropeLC	1200	0,604	1,333	1,679	3,82	7,864	952,97
BT Europe	300	0,043	1,0	0,134	1,0	0,571	1,13
	1200	0,181	1,0	0,515	1,159	2,359	482,58
30_2LC	1200	0,405	1,058	1,05	2,908	3,029	1050,85
30_2	300	0,012	1,0	0,03	1,0	0,094	1,0
	1200	0,054	1,0	0,121	1,0	0,383	1,04

Tabela 5.10: Níveis de utilização das ligações obtidos a partir dos pedidos de encaminhamento com execução de serviços.

5.3 otimização com recurso a algoritmos evolucionários

Nesta Secção é abordado o processo de otimização com recurso aos algoritmos evolucionários, em conjunto com os modelos lineares. Como tal, é efetuada uma análise ao comportamento das funções de avaliação dos indivíduos para ambas as abordagens identificadas no Capítulo 3. São ainda apresentados os resultados obtidos para a distribuição dos serviços pelos nodos da topologia como também os resultados da aplicação dos pesos IGP.

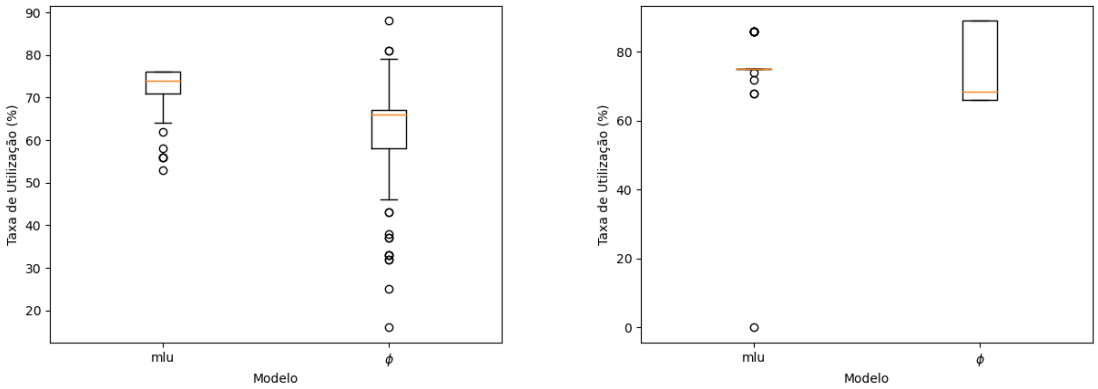
5.3.1 *Comparação das Funções de Avaliação de Indivíduos*

A comparação entre os modelos lineares definidos permite averiguar o comportamento dos mesmos face a diversos cenários de utilização das ligações e dos nodos. Pretende-se determinar qual dos modelos permite obter uma melhor utilização dos recursos. Foi utilizado o algoritmo *multi-objective* NSGA-II com uma população de 100 indivíduos e com um critério de paragem de 50 gerações. Para a abordagem com o número de serviços fixos foi definido um limite máximo de 30 serviços disponíveis para as topologias 30_2, 30_2LC, *BT Europe* e *BT Europe*. Para a topologia *Abilene* e *AbileneLC*, foi definido um limite máximo de 15 serviços, o que se adequa ao número reduzido de nodos das mesmas. Note-se que, o limite máximo de serviços, considera o número total de serviços implementados. Nesse sentido, estão incluídas as replicações de cada serviço. O número de replicações para cada serviço é definido pelos MILP. Nesse sentido, caso no conjunto de pedidos de encaminhamento um serviço seja mais requisitado, é expectável que os modelos apresentem soluções com um maior número de instâncias desse mesmo serviço.

Os resultados obtidos são apresentados de seguida em *boxplots*, que permitem efetuar uma comparação direta entre os dois modelos, face a diferentes configurações topológicas. Para a análise foi considerada a utilização de MPTCP em ambos os modelos, ou seja, não são aplicadas as variáveis $\beta_a^{i,k}$.

Topologia AbileneLC

A Figura 5.19 representa as taxas de utilização das ligações e dos nodos. Nas ligações, o modelo Φ apresenta uma mediana inferior ao MLU, no entanto, admite a existência de alguns *outliers*. Apesar disso, estes não ultrapassam os 90%. Relativamente aos nodos, o modelo Φ obteve uma taxa de utilização ligeiramente inferior. No entanto, tal como nas ligações, existe uma maior variância de taxas de utilização. Como tal, o modelo MLU distribui a carga de maneira mais uniforme entre os nodos com serviços disponíveis.



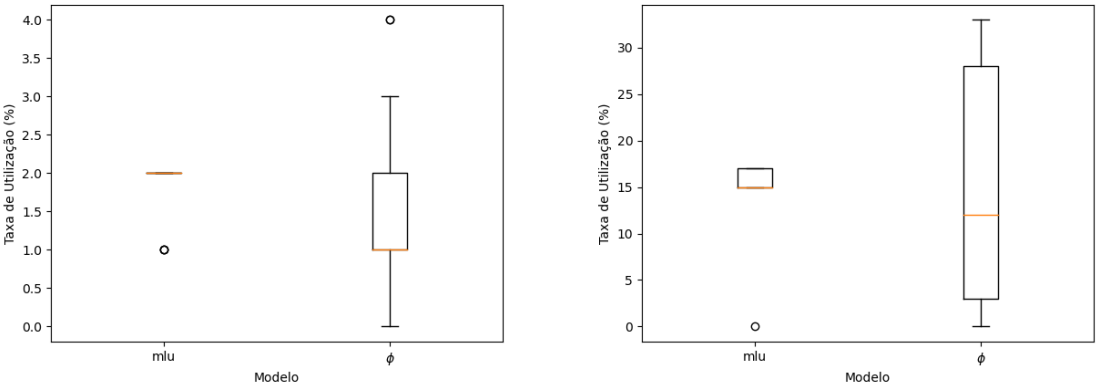
(a) Ligações (b) Nodos

Figura 5.19: Taxas de utilização para 1200 pedidos na AbileneLC.

Topologia Abilene

A Figura 5.20 representa as taxas de utilização das ligações e dos nodos para 300 pedidos de encaminhamento. Nos resultados obtidos é possível observar que nas ligações, o modelo MLU apresenta um desvio padrão inferior face ao do modelo Φ. Para além disso, a taxa máxima de utilização das ligações é superior neste último. No entanto, a mediana obtida a partir do modelo Φ é inferior, ou seja, este modelo apresenta melhores resultados na distribuição dos pedidos pelas ligações da topologia.

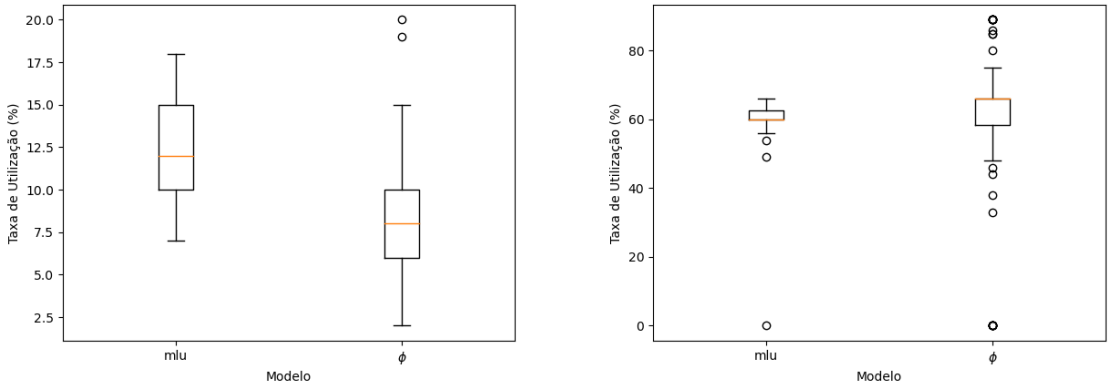
Relativamente aos nodos, o modelo MLU volta a obter um desvio padrão inferior face ao modelo Φ sendo que a taxa máxima de utilização é também menor. Em contrapartida, a mediana obtida no modelo MLU, é ligeiramente superior.



(a) Ligações (b) Nodos

Figura 5.20: Taxas de utilização para 300 pedidos na Abilene.

Na Figura 5.21 estão representadas as taxas de utilização das ligações e dos nodos para 1200 pedidos de encaminhamento. O modelo MLU apresenta um desvio padrão inferior ao modelo Φ. Apesar disso, a mediana obtida é superior no modelo MLU. Relativamente à utilização dos nodos, o modelo Φ apresenta uma mediana superior à do modelo MLU.



(a) Ligações (b) Nodos

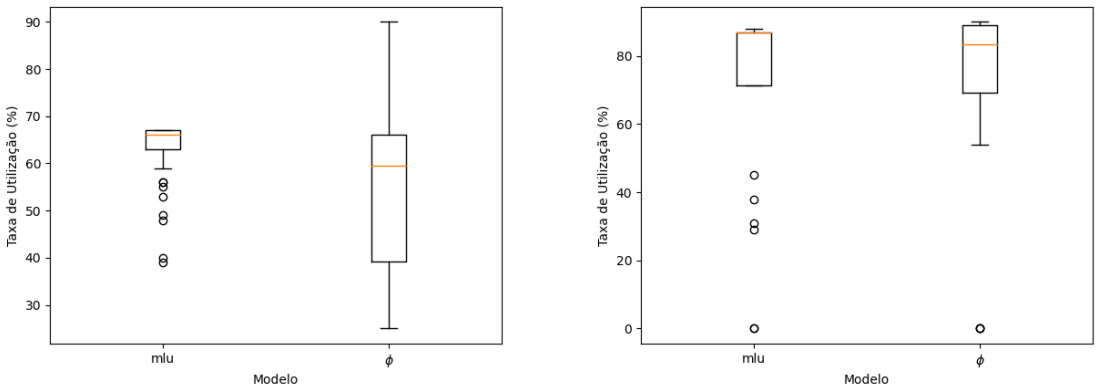
Figura 5.21: Taxas de utilização para 1200 pedidos na Abilene.

Em suma, o modelo Φ destaca-se em ambos os testes relativamente à taxa de utilização das ligações da topologia com valores da mediana inferiores ao modelo MLU. Em contrapartida, o modelo MLU apresenta melhores resultados na taxa de utilização dos nodos, mantendo a taxa máxima de utilização inferior à obtida através do modelo Φ. No entanto, é importante referir que foram obtidas soluções através deste último nas quais existiam nodos sem serviços atribuídos. Isto faz com que as taxas de utilização dos nodos nestas configurações sejam superiores, atingindo no limite taxas de utilização de 89%. O mesmo não se verifica no modelo MLU que tende a aplicar serviços em todos os nodos da topologia, respeitando sempre o limite máximo de serviços imposto. Uma outra consequência da distribuição dos serviços apresentada pelo modelo Φ é a possibilidade de no mesmo nodo serem executados todos os serviços requisitados por um pedido de encaminhamento. Nesta situação, o tráfego associado a esse pedido iria percorrer menos nodos e ligações da topologia, resultando assim em valores de utilização das ligações inferiores tal como é possível observar nos gráficos apresentados.

Topologia BT EuropeLC

Na Figura 5.22 estão representadas as taxas de utilização das ligações e dos nodos, respetivamente, para 1200 pedidos de encaminhamento. A mediana obtida no modelo Φ foi inferior à do modelo MLU apesar de uma maior variância das taxas de utilização dos

nodos e das ligações. Apesar dos níveis de utilização máximos mais elevados no modelo Φ, não foram ultrapassadas as capacidades tanto das ligações como dos nodos.



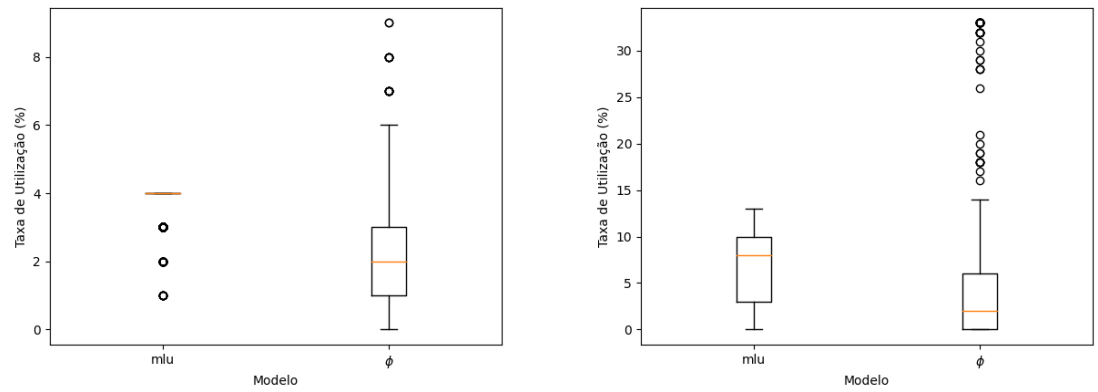
(a) Ligações (b) Nodos

Figura 5.22: Taxas de utilização para 1200 pedidos na BT EuropeLC.

Topologia BT Europe

Na Figura 5.23 estão representadas as taxas de utilização das ligações e dos nodos, respetivamente, da topologia *BT Europe* para 300 pedidos de encaminhamento.

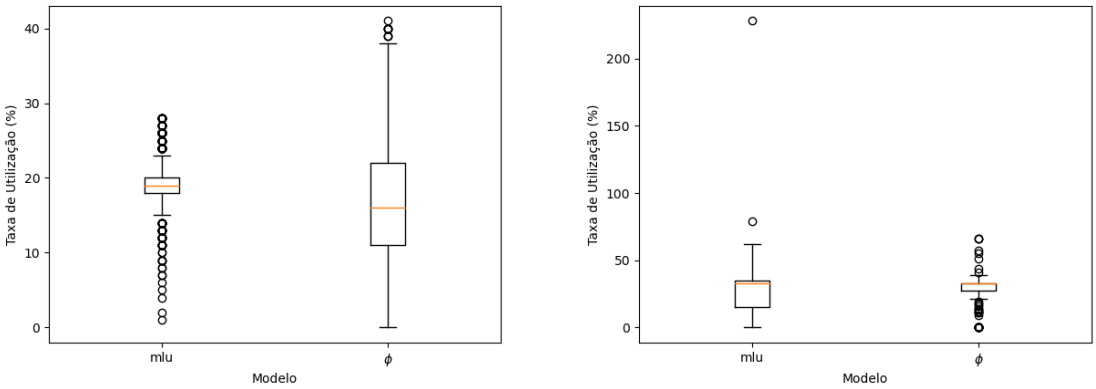
Os valores da mediana obtidos nas taxas de utilização das ligações e dos nodos foram ambos inferiores no modelo Φ. Apesar disso, este modelo apresentou um desvio padrão superior face ao modelo MLU.



(a) Ligações (b) Nodos

Figura 5.23: Taxas de utilização para 300 pedidos na BT Europe.

Na Figura 5.24 estão representadas as taxas de utilização das ligações e dos nodos para 1200 pedidos de encaminhamento. Apesar do modelo Φ apresentar um maior desvio padrão, a mediana obtida é inferior face ao modelo MLU.



(a) Ligações (b) Nodos

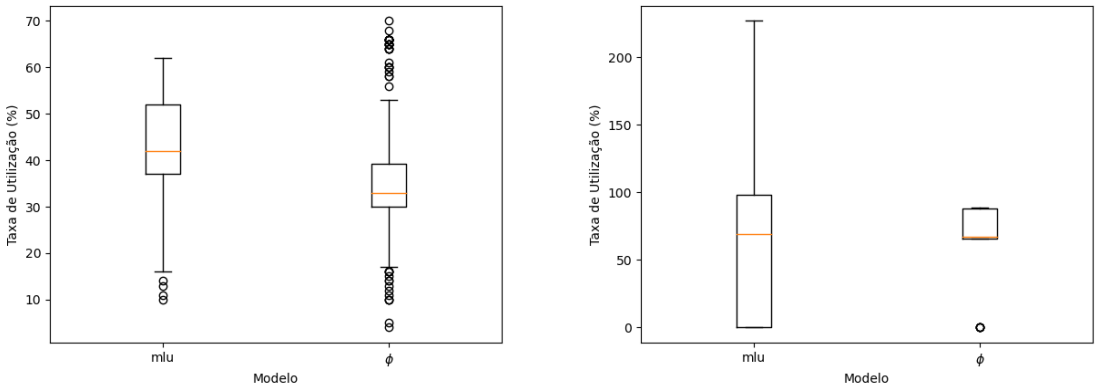
Figura 5.24: Taxas de utilização para 1200 pedidos na BT Europe.

Relativamente aos nodos, apesar do modelo Φ apresentar uma mediana ligeiramente superior, é possível concluir que este modelo conseguiu distribuir melhor a carga pelos nodos da topologia, reduzindo situações de utilização acima da capacidade do nodo. O mesmo não aconteceu a partir do modelo MLU. Neste, uma das soluções obtidas disponibilizou um dos serviços requisitados em apenas um dos nodos tendo atribuído aos restantes nodos os restantes serviços em maior número.

O modelo Φ apresentou melhores resultados nas taxas de utilização das ligações e dos nodos nos testes executados com a topologia *BT Europe*. Nos resultados obtidos a partir do modelo MLU foi possível identificar *outliers* acima da capacidade máxima de utilização dos nodos.

Topologia 30_2LC

A Figura 5.25 representa as taxas de utilização das ligações e dos nodos para 1200 pedidos de encaminhamento na topologia 30_2LC. Tal como nas restantes topologias de baixa capacidade, a mediana dos níveis de congestão é inferior no modelo Φ. Nos nodos surgiram no modelo MLU cenários em que a taxa de utilização ultrapassou consideravelmente a capacidade máxima de processamento dos mesmos.

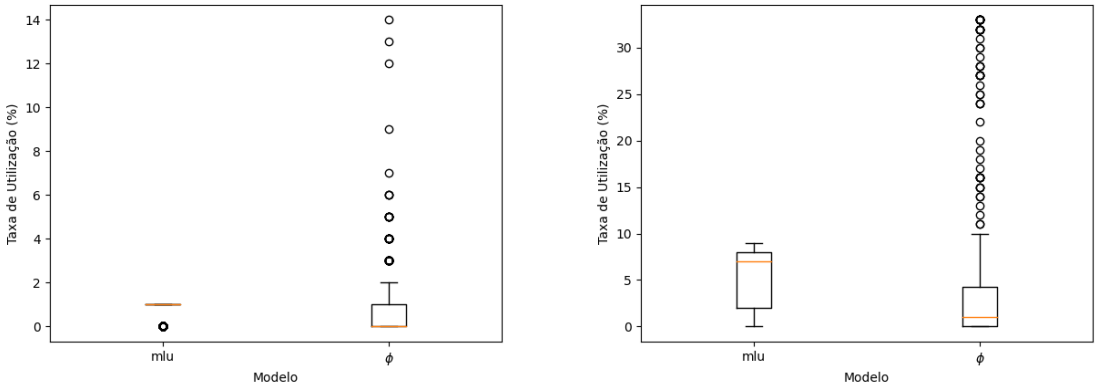


(a) Ligações (b) Nodos

Figura 5.25: Taxas de utilização para 1200 pedidos na 30_2LC.

Topologia 30_2

A Figura 5.26 representa as taxas de utilização das ligações e dos nodos para 300 pedidos de encaminhamento. Relativamente à utilização das ligações, é possível observar que, salvo alguns *outliers*, a mediana do nível de utilização obtida a partir do modelo que visa a otimização da função Φ é inferior face à do modelo de otimização da função objetivo MLU.



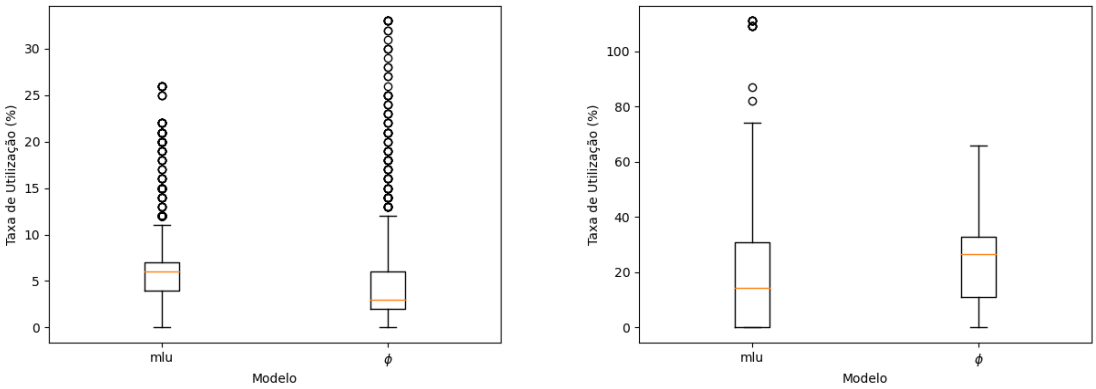
(a) Ligações (b) Nodos

Figura 5.26: Taxas de utilização para 300 pedidos na 30_2.

Relativamente à taxa de utilização dos nodos da topologia, tal como nas ligações, o valor da mediana obtido no modelo linear do Φ foi inferior ao do MLU.

Na Figura 5.27 estão representadas as taxas de utilização das ligações e dos nodos respetivamente. Enquanto que no caso das ligações da topologia o valor da mediana do modelo Φ foi inferior ao MLU, o mesmo não se verifica na carga de utilização dos nodos. No

entanto, apesar da mediana ser ligeiramente superior, é possível verificar que, no caso do MLU, existiram nodos que atingiram uma utilização superior à capacidade dos mesmos.



(a) Ligações (b) Nodos

Figura 5.27: Taxas de utilização para 1200 pedidos na 30_2.

Face aos testes efetuados nos diversos cenários identificados, é possível concluir que o modelo Φ apresenta melhores resultados em cenários de maior volume de pedidos e, como tal, quando as taxas de utilização dos nodos e das ligações são superiores. Tendo em conta a Tabela 5.2, é importante referir que o tempo de execução dos testes com o modelo MLU é sempre superior aos testes com o modelo Φ. Dado que os algoritmos evolucionários tem como critério de paragem mais utilizado o número de iterações, seria inviável utilizar o modelo MLU em cenários cujo tempo de avaliação de um indivíduo é demasiado elevado. Este período de tempo elevado de avaliação leva a que seja necessário aplicar limites ao tempo de execução dos modelos nos quais, por vezes, existe um *gap* significativo para a solução ótima. Esta ocorrência pode prejudicar a solução final apresentada pelo algoritmo. Apesar disso, as soluções obtidas são avaliadas novamente através de uma última execução do MILP, na qual, o tempo limite de execução não é aplicado. Com isto, os caminhos SR são corretamente determinados pelos modelos face à distribuição de serviços obtida.

A Tabela 5.11 apresenta os valores de utilização de ligações obtidos a partir de uma configuração topológica obtida aleatoriamente pelo gerador de pedidos de encaminhamento. Esta Tabela servirá de base para analisar os resultados obtidos a partir dos algoritmos evolucionários no processo de otimização da distribuição dos serviços. O número total de serviços disponíveis nas configurações topológicas é semelhante ao limite aplicado aos algoritmos evolucionários, ou seja, 15 para a topologia *Abilene*/*AbileneLC* e 30 para as restantes.

Topologia	Número de Pedidos	Modelo MLU (c/Serviços) MLU	Modelo Φ (c/Serviços) Φ
AbileneLC	1200	0,963	3,31
Abilene	300	0,045	1,0
	1200	0,184	1,0
BT EuropeLC	1200	1,376	108,39
BT Europe	300	0,143	1,0
	1200	0,585	1,15
30_2LC	1200	0,896	2386,9
30_2	300	0,022	1,0
	1200	0,085	1,0

Tabela 5.11: Níveis de utilização das ligações obtidos com uma configuração topológica aleatória.

Na Tabela 5.12 estão representados os níveis de utilização obtidos nos resultados provenientes dos algoritmos evolucionários com um número de serviços fixos. Nomeadamente, as colunas relativas ao modelo MLU em conjunto com o algoritmo evolucionário NSGA-II, constituem uma configuração obtida a partir do conjunto identificado. De igual modo, as colunas relativas ao modelo Φ em conjunto com o algoritmo evolucionário representam uma configuração obtida pelo conjunto. Face à configuração topológica definida aleatoriamente, os algoritmos evolucionários permitiram sempre obter melhores valores de utilização das ligações, sendo que, o modelo de otimização da métrica Φ obtém melhores resultados em cenários de maior utilização dos recursos disponíveis.

Topologia	Número de Pedidos	NSGA-II + Modelo MLU		NSGA-II + Modelo Φ	
		MLU	Φ	MLU	Φ
AbileneLC	1200	0,767	2,284	0,742	2,09
Abilene	300	0,027	1,0	0,028	1,0
	1200	0,103	1,0	0,108	1,0
BT EuropeLC	1200	0,726	2,08	0,677	2
BT Europe	300	0,05	1,0	0,053	1,0
	1200	0,198	1,98	0,2	1,98
30_2LC	1200	0,656	2,157	0,513	1,325
30_2	300	0,016	1,0	0,019	1,0
	1200	0,066	1,0	0,065	1,0

Tabela 5.12: Níveis de utilização das ligações obtidos com configurações topológicas obtidas pelo algoritmo NSGA-II.

5.3.2 *Análise à Abordagem Multi-Objective*

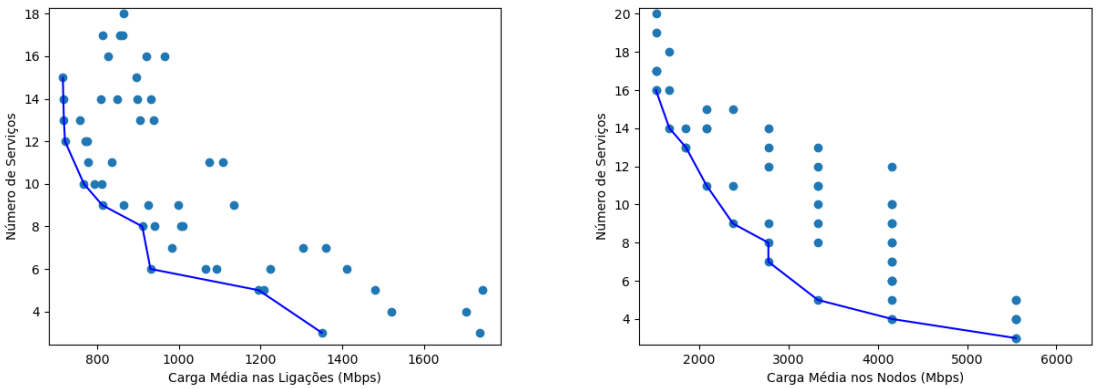
Como foi referido na Subsecção 3.3.2, uma das possíveis abordagens para o problema da distribuição dos serviços, consistia na aplicação do número total de serviços como uma das *fitness* na função de avaliação.

Esta abordagem ao problema é mais difícil de resolver. Isto deve-se ao facto de que não existe uma solução ótima mas sim um conjunto de soluções que permitem analisar o *trade-off* entre melhorar um ou outro objetivo. Este conjunto de soluções é denominado de Frente de Pareto [29]. As soluções são apresentadas em gráficos que permitem determinar qual seria o número ideal de serviços a disponibilizar caso houvesse alguma restrição relativa à carga nos nodos ou nas ligações. Para isso foi utilizado novamente o algoritmo NSGA-II com uma população de 100 indivíduos e um critério de paragem de 100 iterações.

Nesta análise foi considerada a utilização do modelo Φ para cenários de teste com 1200 pedidos de encaminhamento e a utilização do modelo MLU para cenários de teste com 300 pedidos de encaminhamento face aos resultados obtidos na subsecção anterior. Tendo em conta as funções objetivo dos modelos lineares definidos, foram atribuídos à variável α os seguintes valores: $[0.25, 0.5, 0.75]$. Com isto é possível aplicar pesos à função objetivo dos MILP levando a que seja dada mais ou menos importância aos níveis de utilização dos nodos ou das ligações da topologia.

Topologia AbileneLC

A Figura 5.28 representa a frente de Pareto para 1200 pedidos de encaminhamento. Dadas as capacidades da topologia *AbileneLC*, é possível concluir que para o conjunto de pedidos utilizado, seria necessária a implementação de pelo menos 13 serviços na topologia.



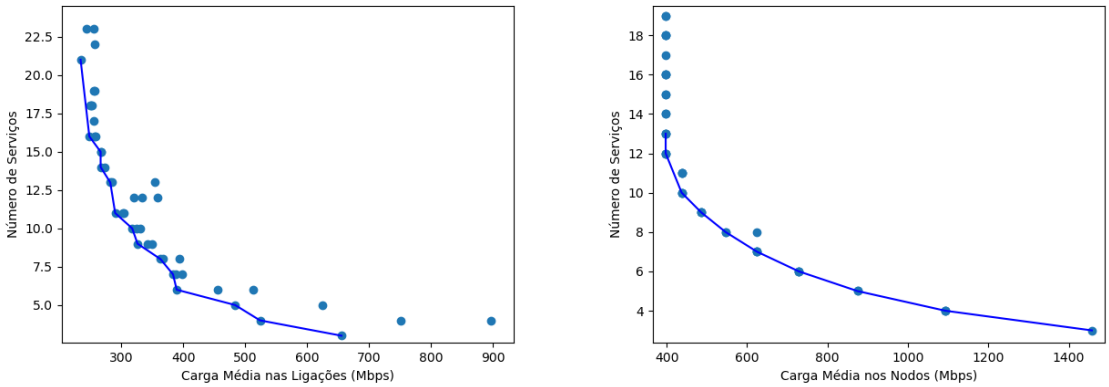
(a) Ligações (b) Nodos

Figura 5.28: Frente de Pareto na AbileneLC para 1200 pedidos.

Este número é obtido a partir da carga média dos nodos que atinge os 2000 Mbps, sendo este valor a capacidade de processamento dos nodos, com aproximadamente 13 serviços distribuídos na rede. Um número de serviços inferior não iria ter um impacto significativo nas ligações da topologia.

Topologia Abilene

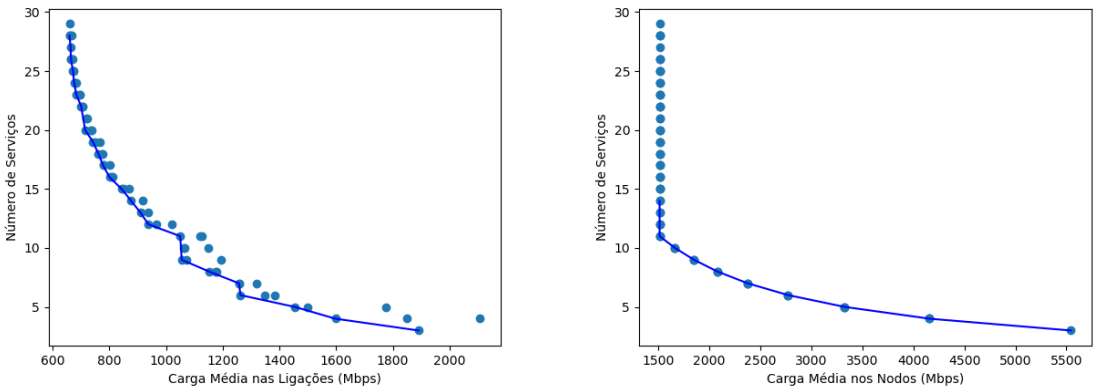
A Figura 5.29 representam a frente de Pareto para 300 pedidos de encaminhamento. É possível identificar que tendo em conta os pedidos de encaminhamento, seriam apenas necessários 12 serviços para manter a carga média nos nodos com serviços abaixo dos 400 Mbps. As ligações, fruto da sua grande capacidade de utilização, não são afetadas pelo número de serviços.



(a) Ligações (b) Nodos

Figura 5.29: Frente de Pareto na Abilene para 300 pedidos.

A Figura 5.30 representa a frente de Pareto para 1200 pedidos de encaminhamento. É possível identificar um aumento significativo na carga média dos nodos e das ligações face aos 300 pedidos analisados nas figuras anteriores.



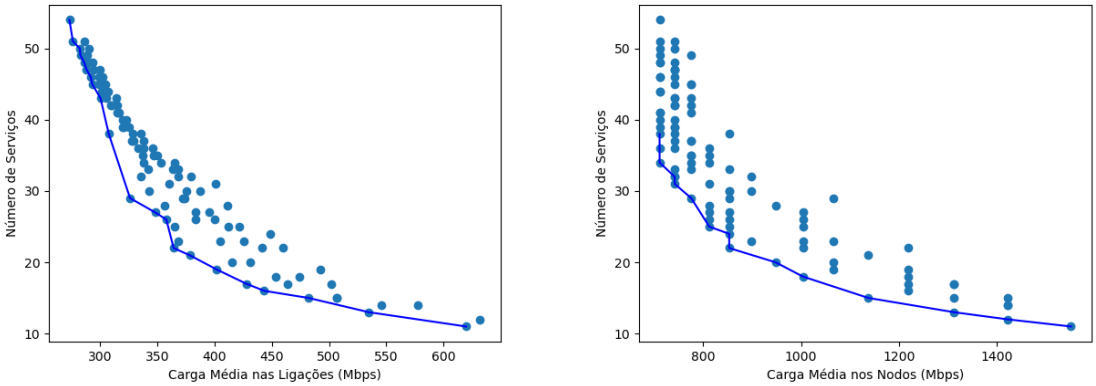
(a) Ligações (b) Nodos

Figura 5.30: Frente de Pareto na Abilene para 1200 pedidos.

Relativamente aos nodos é ainda importante referir que um cenário com menos de 8 serviços vai ultrapassar a capacidade máxima de processamento que está definida em 2500 Mbps.

Topologia BT EuropeLC

A Figura 5.31 representa a frente de Pareto para 1200 pedidos de encaminhamento. É possível concluir que no caso da topologia *BT EuropeLC* seriam necessários pelo menos 20 serviços para que a capacidade máxima de processamento dos nodos não fosse atingida. Relativamente às ligações, um número de serviços inferior não ia implicar cenários de utilização excessiva.

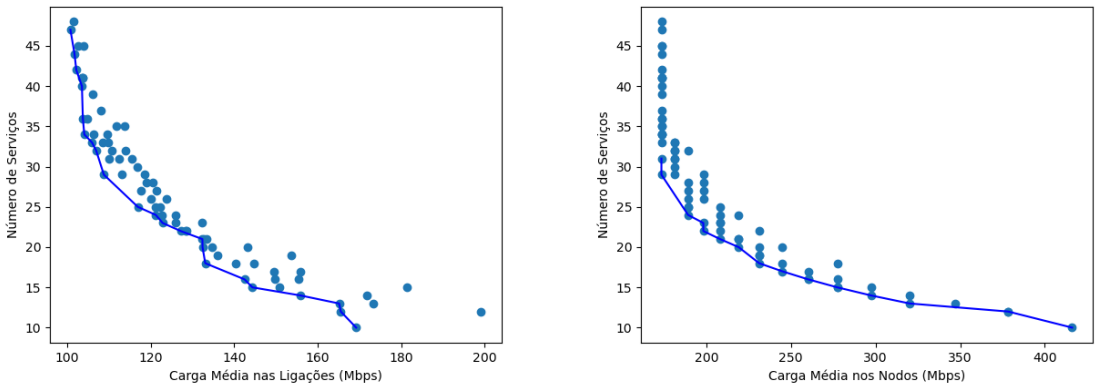


(a) Ligações (b) Nodos

Figura 5.31: Frente de Pareto na BT EuropeLC para 1200 pedidos.

Topologia BT Europe

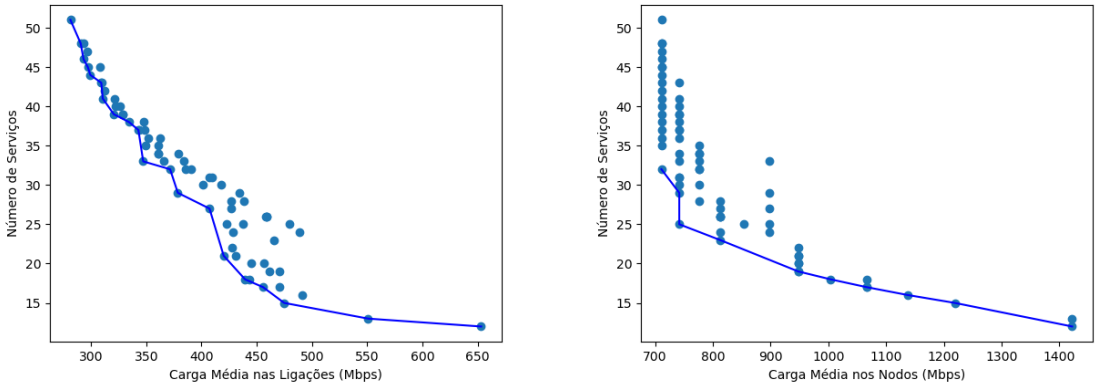
A Figura 5.32 representa a frente de Pareto para 300 pedidos de encaminhamento. A partir das mesmas é possível concluir que uma solução com 10 serviços disponíveis não compromete as taxas de utilização dos nodos e das ligações visto que a capacidade dos mesmos está fixada nos 2500 Mbps.



(a) Ligações (b) Nodos

Figura 5.32: Frente de Pareto na BT Europe para 300 pedidos.

A Figura 5.33 representa a frente de Pareto para 1200 pedidos de encaminhamento. Tal como na análise anterior, uma solução com 10 serviços é capaz de lidar com a carga imposta pelos pedidos de encaminhamento, tanto nos nodos como nas ligações.

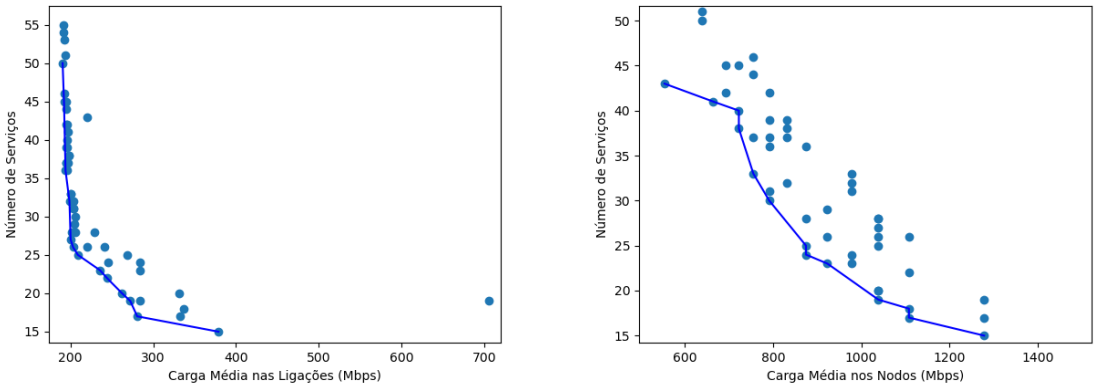


(a) Ligações (b) Nodos

Figura 5.33: Frente de Pareto na BT Europe para 1200 pedidos.

Topologia 30_2LC

A Figura 5.34 representa a frente de Pareto para 1200 pedidos de encaminhamento. É possível concluir que no caso da topologia 30_2LC seriam necessários pelo menos 30 serviços para que a capacidade máxima de processamento dos nodos não fosse atingida. Relativamente às ligações, visto existem casos em que a capacidade máxima das mesmas é apenas 300 Mbps, uma solução com 30 serviços permitiria manter os níveis de utilização sem criar cenários de congestão.

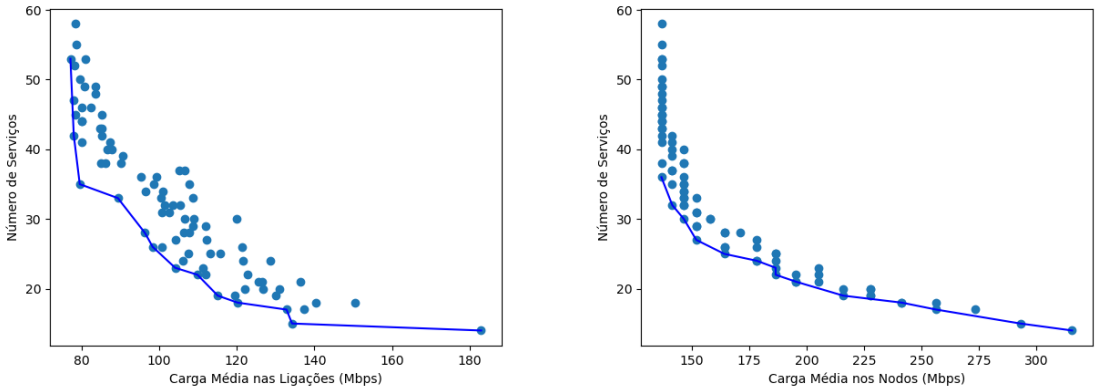


(a) Ligações (b) Nodos

Figura 5.34: Frente de Pareto na 30_2LC para 1200 pedidos.

Topologia 30_2

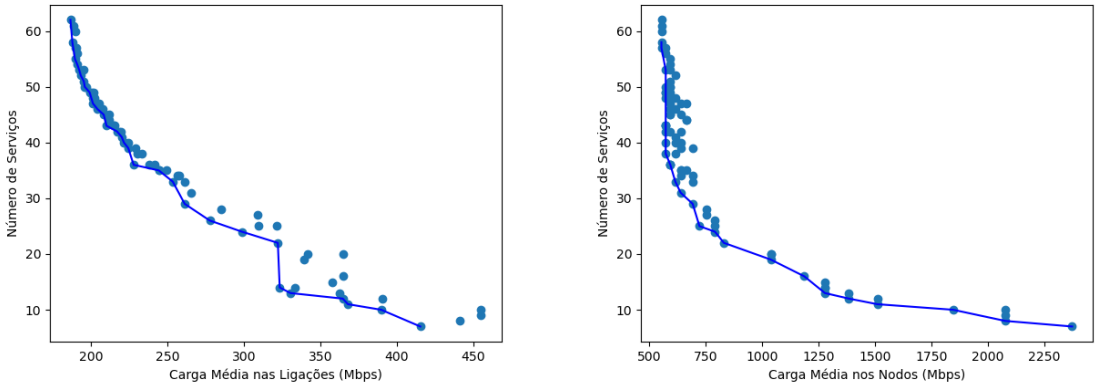
A Figura 5.35 representa a frente de Pareto para 300 pedidos de encaminhamento. Uma solução com aproximadamente 10 serviços consegue dar resposta aos pedidos de encaminhamento sem comprometer as capacidades das ligações e dos nodos da topologia.



(a) Ligações (b) Nodos

Figura 5.35: Frente de Pareto na 30_2 para 300 pedidos.

A Figura 5.36 representa a frente de Pareto para 1200 pedidos de encaminhamento. Nestas, uma solução com apenas 10 serviços poderá comprometer os níveis de utilização dos nodos visto que a capacidade de processamento dos mesmos está próxima de 100%. O mesmo não se verifica nas ligações da topologia que mantém a carga de utilização abaixo de 500 Mbps.



(a) Ligações (b) Nodos

Figura 5.36: Frente de Pareto na 30_2 para 1200 pedidos.

Como é possível observar, quanto maior for o número de serviços, menor será o valor do nível de utilização obtido tanto nos nodos como nas ligações. Este resultado pode ser justificado pelos seguintes fatores:

- Quanto maior o número de nodos com serviços, maior é a capacidade total para executar os mesmos, visto que, cada nodo com serviços disponíveis possui uma capacidade de processamento, ao contrário dos nodos sem serviços atribuídos. Nesse

sentido, quanto maior for o número de nodos utilizados, menor será a carga média por nodo.

- Face à maior disponibilização de serviços, a probabilidade de um pedido encontrar nodos desde a origem até ao destino, com os serviços que pretende disponíveis é maior. Como tal, os pedidos percorrem menos ligações na rede, reduzindo assim o valor do nível de congestão nas ligações.

Todavia, existe normalmente um custo associado à instalação de cada serviço e assim um operador pretenderá minimizar esse mesmo número. Apesar de não ser feita qualquer distinção entre os serviços, o número total consegue ser um bom majorante. Este permite identificar a fração de cada tipo de serviço a ser instalado em função dos diferentes tipos de pedidos.

5.3.3 *Resultados Obtidos na Otimizaçao dos Pesos IGP*

O processo de otimização dos pesos IGP utilizou, numa primeira abordagem, um algoritmo *single-objective* SOEA. Este foi executado com uma população de 100 indivíduos e um critério de paragem de 100 gerações. Neste processo de otimização não é considerado o uso de MPTCP nos modelos lineares que servem de funções de avaliação dos algoritmos evolucionários da distribuição dos serviços (NSGA-II), ou seja, todo o tráfego de um mesmo pedido/fluxo segue um mesmo caminho. Tal como foi mencionado, a componente de otimização dos pesos IGP permite ter em consideração a utilização de protocolos de encaminhamento. Nesse sentido é utilizado o simulador SR da *framework* para comparar os níveis de utilização obtidos a partir dos caminhos SR gerados na distribuição dos serviços pela topologia. O objetivo de otimização deste algoritmo consiste em minimizar a diferença entre o nível de utilização obtido pelo MILP na melhor configuração topológica obtida e o nível de utilização obtido pelo simulador da *framework*. Como termo de comparação são utilizados caminhos SR aleatórios, que para cada pedido, constroem o caminho com base numa configuração topológica obtida a partir do algoritmo evolucionário NSGA-II para cada uma das topologias. Os pesos IGP, obtidos pelo algoritmo SOEA, são posteriormente comparados com uma atribuição de pesos aleatória. Os níveis de utilização obtidos com o uso de caminhos SR aleatórios estão representados na Tabela 5.13. As colunas Modelo MLU e Modelo Φ representam os níveis de utilização obtidos com pesos IGP aleatórios e com caminhos SR aleatórios, tendo por base uma configuração topológica obtida nas soluções do algoritmo de distribuição dos serviços (NSGA-II). As restantes colunas contam com otimização dos pesos IGP através do algoritmo SOEA, no entanto, com caminhos SR aleatórios.

Topologia	Número de Pedidos	Modelo MLU		Modelo Φ		Modelo MLU + SOEA		Modelo Φ + SOEA	
		MLU	Φ	MLU	Φ	MLU	Φ	MLU	Φ
AbileneLC	1200	4,021	2565,9	3,724	2181,5	2,981	1726,2	3,146	1636
Abilene	300	0,13	1.072	0,178	1,115	0,11	1,022	0,149	1,02
	1200	0,525	1,316	0,539	1,28	0,411	1,13	0,432	1,12
BT EuropeLC	1200	6,33	475,4	7,3	723,19	4,523	94,478	5,178	148,2
BT Europe	300	0,48	1,326	0,408	1,21	0,32	1,09	0,302	1,07
	1200	1,342	304,04	2,07	548,9	1,168	34,05	1,39	100,2
30_2LC	1200	13,6	2051,3	12,71	1970,6	10,75	1213	9,78	1287
30_2	300	1,54	122,5	0,623	1,566	0,82	1,38	0,43	1,15
	1200	1,34	75,4	1,51	59,6	0.785	1,37	0,82	1,41

Tabela 5.13: Níveis de utilização das ligações com caminhos SR aleatórios.

A Tabela 5.14 representa os níveis de utilização obtidos com otimização dos pesos IGP, distribuição dos serviços, e dos caminhos SR.

Topologia	Número de Pedidos	NSGA-II + MLU		NSGA-II + Φ		NSGA-II (MLU) + SOEA		NSGA-II (Φ) + SOEA	
		MLU	Φ	MLU	Φ	MLU	Φ	MLU	Φ
AbileneLC	1200	1,87	641,45	1,42	211,54	1,05	6,81	1,0	4,13
Abilene	300	0,06	1,06	0,07	1,08	0,04	1,0	0,06	1,01
	1200	0,192	1,09	0,225	1,09	0,15	1,01	0,175	1,01
BT EuropeLC	1200	2,84	37,03	3,2	60,2	1,49	183,2	1,29	45,1
BT Europe	300	0,205	1,191	0,211	1,213	0,114	1,101	0,175	1,09
	1200	0,857	19,34	0,911	32,47	0,622	1,2	0,491	1,159
30_2LC	1200	5,84	727,3	5,216	540,34	2,439	247,7	2,0	183
30_2	300	0,133	1,239	0,219	1,205	0,095	1,128	0,267	1,129
	1200	0,634	4,324	0,537	1,317	0,463	1,126	0,415	1,13

Tabela 5.14: Níveis de utilização das ligações com otimização dos caminhos SR e dos pesos IGP.

As duas primeiras colunas "NSGA-II + MLU"e "NSGA-II + Φ"consideram apenas a otimização da distribuição dos serviços através do algoritmo evolucionário NSGA-II cujas funções de avaliação de indivíduos são os modelos MLU e Φ. São posteriormente aplicados pesos aleatórios às ligações da topologia e após inserir todos os fluxos no simulador são retirados os valores das métricas MLU e Φ. As duas últimas colunas representam a otimização conjunta da distribuição dos serviços (NSGA-II) e dos pesos IGP (SOEA).

Dos resultados obtidos é possível observar que o algoritmo evolucionário de otimização da distribuição dos serviços em conjunto com a função de avaliação Φ obteve melhores resultados face ao mesmo algoritmo evolucionário mas com a função de avaliação MLU,

com excepção da topologia 30_2 para 1200 pedidos. Apesar disso, a diferença entre níveis de congestão neste último caso é mínima.

Por fim, na abordagem *multiobjective*, pretendia-se otimizar em simultâneo a distribuição dos serviços e os pesos IGP. Este processo utiliza o algoritmo NSGA-II. Como os níveis de utilização, obtidos a partir dos modelos MILP e a partir do simulador SR, estão dependentes do número total de serviços disponíveis na configuração topológica, verificou-se que as soluções obtidas procuravam sempre disponibilizar 2 ou 3 serviços em cada nodo. Naturalmente, quanto maior for o número de serviços, maior será a probabilidade de o caminho mais curto desde a origem até ao destino de um pedido conter os serviços requisitados pelo mesmo. Como resultado existe uma minimização dos níveis de utilização nas ligações das topologias. De igual forma, com um número elevado de serviços disponíveis, a taxa de utilização dos nodos é baixa. Para contornar este problema, poderiam ser aplicadas penalizações a soluções com um número elevado de serviços. Apesar desta abordagem ser uma alternativa válida face à otimização em fases distintas da distribuição dos serviços e obtenção dos pesos IGP, é também limitada pelo tempo de execução dos modelos lineares para obter os locais de execução dos serviços requisitados por cada pedido. Como a solução a gerar pelo algoritmo evolucionário tem um maior número de características, o número de combinações possíveis das mesmas é também superior. Como tal, face às abordagens de otimização em fases distintas, seria necessário a utilização de um maior número de indivíduos e gerações o que, por sua vez, confere a esta abordagem um custo associado ao tempo de execução mais elevado.

Em suma, a otimização dos pesos IGP permite minimizar os níveis de utilização nas ligações da topologia. Apesar disso, é fundamental considerar uma boa distribuição dos serviços pelos nodos da mesma visto que, para cenários em que a distribuição dos mesmos não é otimizada, os níveis de utilização são elevados. Nesse sentido, a conjugação dos métodos de otimização da distribuição dos serviços e dos pesos IGP, é importante para assegurar uma correta utilização dos recursos. Outro fator relevante é o número total de serviços disponibilizados. Este influência os níveis de utilização dos recursos visto que, quanto maior for o número de serviços, menor será o número de saltos necessários para um pedido executar todos os serviços requisitados e posteriormente chegar ao destino. No entanto, o número de serviços a disponibilizar, será sempre limitado pela capacidade de investimento dos operadores de telecomunicações.

5.4 otimização com recurso a machine learning

Esta secção aborda as etapas necessárias para aplicar *machine learning* ao problema abordado nesta dissertação. Para isso vai ser mencionado o processo tratamento do conjunto de dados. Para além disso, vão ser analisados diversos algoritmos, de forma a perceber qual

é o que melhor se adequa ao problema. Por fim os resultados obtidos vão ser comparados com os resultados provenientes dos modelos lineares.

5.4.1 *Análise e Tratamento do Conjunto de Dados*

O processo de otimização com recurso a *machine learning* considerou vários tipos de redes neuronais. Apesar de serem analisados vários modelos, um dos passos mais importantes para a utilização de *machine learning* consiste no tratamento do conjunto de dados. A utilização deste método de otimização foi efetuada tendo por base a topologia *Abilene* cuja distribuição dos serviços está identificada na Tabela 5.15. Os restantes nodos da topologia não possuem serviços atribuídos.

Dado que a configuração topológica apresenta nodos sem serviços associados, todas as *features* relativas à taxa de utilização dos nodos que não estão identificados na tabela anterior, não são consideradas. De igual forma, no *output* são removidas as colunas relativas à execução do serviço num nodo em que o mesmo não se encontra disponível bem como os nodos que não possuem serviços associados.

Configuração Topológica	
ID do Nodo	Lista de Serviços
1	[1, 2]
4	[0, 1]
5	[0, 2]
9	[1, 2]
10	[0, 1]

Tabela 5.15: Configuração topológica utilizada no processo de otimização com *machine learning*.

O processo de treino de um modelo de *machine learning* requer por norma um conjunto de dados com um elevado número de entradas. No entanto, foi encontrada uma limitação no gerador que não permitia obter um conjunto de dados com mais de 25000 entradas. Esta limitação deve-se à utilização da ferramenta Cplex, que por sua vez consome muitos recursos. Para além desta limitação foram detetadas entradas cujas taxas de utilização das ligações e dos nodos estavam muito acima do expectável. Estas entradas encontram-se nas últimas linhas do conjunto de dados. Naturalmente, com a falta de recursos, não é possível melhorar as soluções levando a que o *gap* para a solução óptima seja elevado. Consequentemente, estas entradas foram removidas.

Com o objetivo de aumentar o número de entradas para treinar o modelo, fruto das limitações encontradas no gerador, foram agregados dois conjuntos de dados. Estes são gerados a partir de ficheiros com pedidos de encaminhamento distintos. Com esta junção

foi necessário adicionar uma nova *feature* que indica a que conjunto de dados cada entrada no mesmo pertence.

De forma a melhorar o conjunto de dados, foram alteradas algumas colunas. As colunas origem e destino, que indicam o ID do nodo para o ponto de partida e chegada do pedido, foram alteradas para um *array* binário. Na Tabela 5.16 está representado um exemplo da alteração efetuada.

origin	*destination*	O1	O2	O3	O4	D1	D2	D3	D4
2	3	0	1	0	0	0	0	1	0

Tabela 5.16: Exemplo de transformação das colunas origem e destino do conjunto de dados.

Para além das colunas *origin* e *destination*, cujos valores eram díspares face aos restantes, também as colunas "*duration*"e "*bandwidth*"foram alteradas a partir de um processo de normalização. A partir deste é possível reduzir o intervalo de valores destas colunas. A normalização é efetuada através da seguinte fórmula:

$$ValorFinal = \frac{ValorAtual - Media}{DesvioPadrao}$$

Na Tabela 5.17 estão indicadas o número de entradas por serviço no conjunto de dados que foi utilizado para treinar o modelo. O conjunto de dados foi separado em 80% para treino e 20% para validação.

Serviço	Total Treino	Total Validação	Total
0	22970	5743	28713
1	23171	5793	28964
2	23210	5803	29013

Tabela 5.17: Número de entradas por serviço do conjunto de dados.

5.4.2 *Análise aos Modelos de Classificação*

O *output* esperado representa o local de execução dos serviços requisitados. Assim, para dar resposta a todos os serviços simultaneamente, seria necessário um modelo capaz de classificar o problema com recurso a algoritmos *multilabel*. Nesta, se um dado serviço S[0..2] é requisitado, uma coluna do conjunto de colunas relativas aos locais de execução desse mesmo serviço deverá ter a *label* 1. Caso o serviço não seja requisitado, todas as colunas deverão ter a *label* 0. Para este tipo de problemas, a função de *loss* mais adequada seria a *binary_crossentropy* em conjunto com uma função de ativação *sigmoid*. Para determinar a melhor estrutura da rede neuronal, foi utilizado o método de *grid search* nos modelos de redes neuronais e LSTM. No entanto, resultados preliminares, mostraram que os modelos tinham alguma dificuldade em obter bons resultados de *accuracy*.

Como os resultados obtidos com algoritmos *multilabel* não foram de encontro ao esperado, devido ao baixo valor da *accuracy* de validação e em treino, foi necessário considerar outra alternativa. Esta consiste em treinar três modelos distintos, um para cada serviço. Desta forma o problema seria uma classificação *multiclass*. Nestes, apenas uma posição do *array output* estará com o valor 1. No entanto e ao contrário do tipo de classificação *multilabel*, a utilização de algoritmos *multiclass* implica a remoção das entradas que não requisitam o serviço em causa de forma a impedir a introdução de ruído no conjunto de dados visto que, seria expectável observar uma tendência do modelo para dar demasiado importância à *feature* que indica se o serviço é ou não requisitado. Esta separação do problema permite diminuir a complexidade do mesmo facilitando assim o processo de treino das redes neuronais.

Modelo Random Forest

Os modelos de classificação RF não são influenciados por *outliers* nem assumem qualquer assunção sobre a distribuição dos dados. Constituem assim um bom ponto de partida para identificar um modelo classificativo que possa prever o melhor nodo para o processamento de cada pedido. A Tabela 5.18 representa a *confusion matrix* para o serviço 0. Esta matriz permite averiguar quantos nodos foram atribuídos corretamente. Naturalmente, face às métricas de desempenho ilustradas na Tabela 5.19, é esperado que a maior parte dos nodos previstos pelo modelo sejam iguais aos nodos reais.

Serviço 0		Previsto		
		4	5	10
Real	4	1423	283	325
	5	434	1049	261
	10	299	205	1464

Tabela 5.18: *Confusion Matrix* do modelo *Random Forest* para o serviço 0.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
4	0,66	0,70	0,68	69%
5	0,68	0,60	0,64	
10	0,71	0,74	0,73	

Tabela 5.19: Métricas obtidas para o serviço 0 com o modelo *Random Forest*.

Relativamente ao serviço 1, visto que este possui mais um nodo como possível solução face aos restantes serviços, surge uma redução na quantidade de nodos previstos iguais aos reais (tabela 5.20).

Serviço 1		Previsto			
		1	4	9	10
Real	1	1091	304	85	129
	4	255	1149	152	146
	9	89	230	637	204
	10	151	233	165	773

Tabela 5.20: *Confusion Matrix* do modelo *Random Forest* para o serviço 1.

Estes resultados são suportados pelos resultados ilustrados na Tabela 5.21 que mostram uma redução na percentagem de *accuracy* bem como nas restantes métricas.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
1	0,69	0,68	0,68	63%
4	0,60	0,68	0,64	
9	0,61	0,55	0,58	
10	0,62	0,58	0,60	

Tabela 5.21: Métricas obtidas para o serviço 1 com o modelo *Random Forest*.

O modelo de treino do serviço 2 foi o que obteve melhores resultados tal como é possível observar nas Tabelas 5.22 e 5.23.

Serviço 2		Previsto		
		1	5	9
Real	1	1286	288	200
	5	288	1263	359
	9	190	310	1619

Tabela 5.22: *Confusion Matrix* do modelo *Random Forest* para o serviço 2.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
1	0,73	0,72	0,73	72%
5	0,68	0,66	0,67	
9	0,74	0,76	0,75	

Tabela 5.23: Métricas obtidas para o serviço 2 com o modelo *Random Forest*.

Em suma, os modelos de treino dos serviços 0 e 2 obtiveram os melhores resultados fruto do menor número de classes existentes.

Modelo SVM

Para treinar o modelo SVM foram testadas três funções de *Kernel*, nomeadamente: *Linear*, RBF e *Polynomial*.

Nas Tabelas 5.24 e 5.25 e 5.26 estão representadas as *confusion matrix* para o serviço 0 com as três funções de ativação mencionadas.

Linear		Previsto		
Serviço 0		4	5	10
Real	4	1406	343	316
	5	407	1073	250
	10	288	229	1431

Tabela 5.24: *Confusion Matrix* do modelo SVM com a função *Linear* para o serviço 0.

RBF		Previsto		
Serviço 0		4	5	10
Real	4	1452	301	312
	5	383	1093	254
	10	294	219	1435

Tabela 5.25: *Confusion Matrix* do modelo SVM com a função RBF para o serviço 0.

Polynomial		Previsto		
Serviço 0		4	5	10
Real	4	1344	411	310
	5	431	1045	254
	10	365	268	1315

Tabela 5.26: *Confusion Matrix* do modelo SVM com a função *Polynomial* para o serviço 0.

A partir dos resultados obtidos na Tabela 5.27 é possível concluir que a melhor função *kernel* a aplicar para o modelo do serviço 0 é a RBF.

Kernel Function	Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
Linear	4	0,67	0,68	0,67	68%
	5	0,65	0,62	0,64	
	10	0,72	0,73	0,73	
RBF	4	0,68	0,70	0,69	69%
	5	0,68	0,63	0,65	
	10	0,72	0,74	0,73	
Polynomial	4	0,63	0,65	0,64	64%
	5	0,61	0,60	0,61	
	10	0,70	0,68	0,65	

Tabela 5.27: Métricas obtidas para o serviço 0 com o modelo SVM.

As Tabelas 5.28, 5.29 e 5.30 representam as *confusion matrix* para o serviço 1.

Linear Serviço 1		Previsto 1	4	9	10
Real	1	1083	281	99	112
	4	309	1102	130	179
	9	111	182	690	219
	10	125	208	185	778

Tabela 5.28: *Confusion Matrix* do modelo SVM com a função *Linear* para o serviço 1.

RBF Serviço 1		Previsto 1	4	9	10
Real	1	1160	242	81	92
	4	270	1150	146	154
	9	89	169	743	201
	10	115	211	189	781

Tabela 5.29: *Confusion Matrix* do modelo SVM com a função RBF para o serviço 1.

Polynomial Serviço 1		Previsto 1	4	9	10
Real	1	1059	311	93	112
	4	341	1046	151	182
	9	115	220	630	237
	10	163	244	215	674

Tabela 5.30: *Confusion Matrix* do modelo SVM com a função *Polynomial* para o serviço 1.

Tal como no modelo *Random Forest*, as métricas obtidas para este serviço são inferiores devido ao maior número de classes. As métricas estão representadas na Tabela 5.31. Novamente, a melhor função *kernel* foi a RBF.

Kernel Function	Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
Linear	1	0,67	0,69	0,68	63%
	4	0,62	0,64	0,63	
	9	0,62	0,57	0,60	
	10	0,60	0,60	0,60	
RBF	1	0,71	0,74	0,72	66%
	4	0,65	0,67	0,66	
	9	0,64	0,62	0,63	
	10	0,64	0,60	0,62	
Polynomial	1	0,63	0,67	0,65	59%
	4	0,57	0,61	0,59	
	9	0,58	0,52	0,55	
	10	0,56	0,52	0,54	

Tabela 5.31: Métricas obtidas para o serviço 1 com o modelo SVM.

Por fim, para o serviço 2, estão representadas as *confusion matrix* nas Tabelas 5.32, 5.33 e 5.34.

Linear		Previsto		
Serviço 2		1	5	9
Real	1	1340	308	185
	5	296	1261	345
	9	178	288	1602

Tabela 5.32: *Confusion Matrix* do modelo SVM com a função *Linear* para o serviço 2.

RBF		Previsto		
Serviço 2		1	5	9
Real	1	1397	284	152
	5	310	1286	306
	9	160	280	1628

Tabela 5.33: *Confusion Matrix* do modelo SVM com a função RBF para o serviço 2.

Polynomial		Previsto		
Serviço 2		1	5	9
Real	1	1299	354	180
	5	361	1197	344
	9	188	364	1516

Tabela 5.34: *Confusion Matrix* do modelo SVM com a função Kernel *Polynomial* para o serviço 2.

Na Tabela 5.35 estão representadas as métricas obtidas para treino do modelo do serviço 2. Novamente é possível verificar um aumento da *accuracy* para este serviço. Em suma, face aos resultados obtidos, a melhor função *kernel* a utilizar seria a RBF.

Kernel Function	Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
Linear	1	0,74	0,73	0,73	68%
	5	0,68	0,66	0,67	
	9	0,75	0,77	0,76	
RBF	1	0,75	0,76	0,76	69%
	5	0,70	0,68	0,69	
	9	0,78	0,79	0,78	
Polynomial	1	0,70	0,71	0,71	64%
	5	0,63	0,63	0,63	
	9	0,74	0,73	0,74	

Tabela 5.35: Métricas obtidas para o serviço 2 com o modelo SVM.

Modelo Rede Neuronal

Para descobrir a melhor configuração da rede neuronal, foi utilizado o método de *grid-search*. Através deste é possível mudar os parâmetros da rede e perceber de que forma estas alteram os resultados obtidos. Os principais parâmetros alterados são as funções de ativação, o número de camadas da rede neuronal, o número de nodos em cada camada e a percentagem de *dropout*. Esta última, quando aplicada, permite evitar cenários de *overfitting*. Os parâmetros testados estão representados na Listagem 5.1.

```
grid_param = {
    'n_hidden': [2,4],
    'size_nodo': [128,256],
    'ativ': ['sigmoid','softmax','relu'],
    'dropout': [0.2,0.5],
}
```

Listagem 5.1: Parâmetros utilizados com o método *grid search* no modelo de redes neuronais.

Para um total de 200 *epochs*, a melhor configuração obtida foi a seguinte:

- n_hidden: 2
- size_nodo: 256
- ativ: sigmoid
- dropout: 0.5

Estes parâmetros foram aplicados aos modelos de cada serviço. A Tabela 5.36 representa a *confusion matrix* para o serviço 0. As métricas de desempenho obtidas estão representadas na Tabela 5.37.

Serviço 0		Previsto		
		4	5	10
Real	4	1398	389	264
	5	325	1234	192
	10	257	264	1420

Tabela 5.36: *Confusion Matrix* do modelo com redes neuronais para o serviço 0.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
4	0,71	0,68	0,69	71%
5	0,65	0,70	0,68	
10	0,76	0,73	0,74	

Tabela 5.37: Métricas obtidas para o serviço 0 com o modelo de redes neuronais.

As Tabelas 5.38 e 5.39 representam os resultados obtidos para o serviço 1. Tal como nos modelos RF e SVM, as métricas de performance sofreram uma diminuição face ao serviço 0 devido ao aumento do número de classes.

Serviço 1		Previsto			
		1	4	9	10
Real	1	1190	249	69	114
	4	282	1119	119	191
	9	98	148	690	215
	10	113	183	189	824

Tabela 5.38: *Confusion Matrix* do modelo de redes neuronais para o serviço 1.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
1	0,71	0,73	0,72	66%
4	0,66	0,65	0,66	
9	0,65	0,60	0,62	
10	0,62	0,63	0,62	

Tabela 5.39: Métricas obtidas para o serviço 1 com o modelo de redes neuronais.

As Tabelas 5.40 e 5.41 representam os resultados obtidos no treino do modelo para o serviço 2.

Serviço 2		Previsto		
		1	5	9
Real	1	1434	269	160
	5	294	1309	319
	9	162	265	1591

Tabela 5.40: *Confusion Matrix* do modelo com redes neuronais para o serviço 2.

Nodo	Precision	Recall	f1-score	Accuracy
1	0,76	0,77	0,76	75%
5	0,71	0,68	0,70	
9	0,77	0,79	0,78	

Tabela 5.41: Métricas obtidas para o serviço 2 com o modelo de redes neuronais.

Modelo LSTM

As redes recursivas permitem treinar um modelo cujo conjunto de dados tem por base uma evolução temporal. O conjunto de dados obtido pelo gerador, consiste numa evolução temporal da utilização dos nodos e das ligações da topologia. Naturalmente, este tipo de redes neuronais seria adequada ao problema. Nesse sentido, é importante garantir que o

conjunto de dados mantém a ordem original. Para isso foi removida a opção de *shuffle* dos dados, que foi uma técnica utilizada nos modelos anteriores.

Tal como no modelo de rede neuronal foi aplicado o método de *grid-search* com o objetivo de extrair a melhor configuração. Os parâmetros testados estão representados na Listagem 5.2.

```
grid_param = {
    'n_hidden': [2,4],
    'size_nodo': [128,256],
    'dropout': [0.2,0.5,0.8],
}
```

Listagem 5.2: Parâmetros utilizados com o método *grid search* no modelo LSTM.

Não foi inserida nenhuma função de ativação nas camadas da rede LSTM. Estas, por defeito, utilizam a função *tanh*.

Para um total de 200 *epochs*, a melhor configuração obtida foi a seguinte:

- n_hidden: 2
- size_nodo: 256
- dropout: 0.8

Comparativamente com o modelo de redes neuronais, a melhor configuração obteve um *dropout* superior. Isto mostra que para o problema em questão, as LSTM são mais susceptíveis a sofrer de *overfitting*.

Nas Tabelas 5.42 e 5.43 estão representadas as métricas de desempenho para o serviço 0.

Serviço 0		Previsto		
		4	5	10
Real	4	1270	358	424
	5	718	744	271
	10	447	222	1289

Tabela 5.42: *Confusion Matrix* do modelo com LSTM para o serviço 0.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
4	0,52	0,62	0,57	58%
5	0,56	0,43	0,49	
10	0,65	0,66	0,65	

Tabela 5.43: Métricas obtidas para o serviço 0 com o modelo de LSTM.

Nas Tabelas 5.44 e 5.45 estão representadas as métricas de desempenho para o serviço 1.

Serviço 1		Previsto			
		1	4	9	10
Real	1	1037	271	124	126
	4	600	720	244	173
	9	148	268	510	258
	10	216	267	353	478

Tabela 5.44: *Confusion Matrix* do modelo LSTM para o serviço 1.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
1	0,52	0,67	0,58	47%
4	0,47	0,41	0,44	
9	0,41	0,43	0,42	
10	0,46	0,36	0,41	

Tabela 5.45: Métricas obtidas para o serviço 1 com o modelo LSTM.

Nas Tabelas 5.46 e 5.47 estão representadas as métricas de desempenho para o serviço 2.

Serviço 2		Previsto		
		1	5	9
Real	1	1363	266	198
	5	560	881	447
	9	266	360	1462

Tabela 5.46: *Confusion Matrix* do modelo LSTM para o serviço 2.

Nodo	*Precision*	*Recall*	*f1-score*	*Accuracy*
1	0,62	0,75	0,69	64%
5	0,58	0,47	0,52	
9	0,69	0,70	0,70	

Tabela 5.47: Métricas obtidas para o serviço 2 com o modelo de LSTM.

Comparativamente com os modelos identificados anteriormente, as redes LSTM obtiveram piores resultados nos modelos de cada serviço. Estes resultados poderão ser justificados pelo tamanho reduzido do conjunto de dados.

5.4.3 *Resultados Obtidos*

De forma a analisar a qualidade das soluções obtidas a partir de *machine learning*, os resultados vão ser comparados com os do MILP que gerou o conjunto de dados. Através de um modelo linear vão ser extraídos os níveis de utilização dos recursos resultantes do ficheiro de *output* das redes neuronais. Este modelo linear considera o estado da topologia

e, através da variável binária $\alpha_{i,k}^{n}$, executa os serviços nos nodos indicados pelo *output*. Os resultados do MILP, obtidos a partir do conjunto de dados, são também avaliados a partir deste modelo. Para a geração do ficheiro *output*, resultante da aplicação do modelo de redes neuronais ao conjunto de dados fornecido, não foi considerada a aplicação de um valor *threshold*. Nesse sentido, de forma a descobrir em que nodo o serviço foi executado, é considerada como solução a posição do *array* com a percentagem mais elevada. O conjunto de dados utilizado para a análise foi compilado a partir de um novo ficheiro de pedidos de encaminhamento com 2000 entradas.

Na Tabela 5.48 estão representados os níveis de congestão obtidos pelos MILP em comparação com valores obtidos por redes neuronais. Os valores Φ e Γ não se encontram normalizados, visto que, são extraídos a partir das funções objetivo do modelo que foi utilizado na avaliação dos resultados. Os valores do MLU e do MNU são obtidos a partir das variáveis de utilização e capacidade dos nodos e ligações.

Método de Otimização	Φ	Γ	MLU	MNU	$(\Phi+\Gamma)/2$
MILP	16,6	5,94	1,0	0,35	11,27
Redes Neuronais	16,6	14,25	1,0	0,35	15,42

Tabela 5.48: Comparação dos níveis de utilização obtidos.

Comparativamente com os modelos MILP, as soluções obtidas a partir dos modelos de redes neuronais obtiveram piores resultados na utilização dos nodos. Apesar dos resultados obtidos nas métricas Φ e Γ não se encontrarem normalizados, os valores elevados podem ser justificados a partir de algumas taxas de utilização no conjunto de dados próximas de 100%. Estas no entanto ocorrem em apenas um nodo e ou ligação o que faz com que possíveis cenários de congestão nem sempre sejam refletidos pelas métricas MLU e MNU.

Em suma, apesar de os resultados das redes neuronais terem sido ligeiramente piores que os MILP, a utilização deste método de otimização tem a vantagem de permitir obter resultados em tempo real. Nesse sentido, é uma alternativa viável a considerar face à limitação dos métodos de otimização como os MILPs e ou métodos híbridos que utilizam MILPs e algoritmos evolucionários em simultâneo.

5.5 sumário

Neste capítulo foram analisados os resultados obtidos nas diferentes etapas de resolução do problema. A partir dos mesmos foi possível averiguar quais as configurações topológicas óptimas para cada topologia e respetivo conjunto de pedidos de encaminhamento. Estas configurações englobam a distribuição dos serviços pelos nodos da topologia e os pesos IGP a aplicar às ligações, tirando partido da utilização de balanceadores de tráfego como o ECMP. As várias abordagens referidas representam um processo iterativo que culmina

com a otimização simultânea dos objetivos propostos. No entanto, uma otimização em simultâneo requer a aplicação de um limite ao número máximo de serviços a disponibilizar que irá sempre depender daquilo que são as possibilidades a nível de recursos por parte das operadoras de telecomunicações. Por fim foram apresentados os resultados obtidos a partir do processo de otimização com recurso a ML.

6

CONCLUSÕES

Neste capítulo serão abordadas as conclusões obtidas face aos capítulos anteriores, compilando um resumo do trabalho desenvolvido, limitações do mesmo e, por fim, terminando com uma reflexão acerca do trabalho futuro a realizar.

6.1 RESUMO DO TRABALHO DESENVOLVIDO

Esta dissertação teve como objetivo explorar novas soluções para a otimização da utilização de recursos com SR. Nesse sentido, teve como etapas a elaboração do estado da arte, a modelação do problema e posterior desenvolvimento e integração na *framework* NetOpt, culminando com a análise aos resultados obtidos.

No estado da arte foi efetuado um levantamento dos conceitos referentes ao encaminhamento dinâmico no qual se destacam as arquiteturas *Segment Routing* e *Network Function Virtualization*. Diversos trabalhos na área abordam o NFV, associando-o a novos desenvolvimentos na área das telecomunicações como o 5G. Esta nova tecnologia está de momento a ser disponibilizada, prometendo mais capacidade e velocidade face à geração anterior. A sua integração com tecnologias como o SR pretende agilizar o processo de implementação de novos tipos de serviços por parte das operadoras de telecomunicações. Ainda no estado da arte foram identificadas duas funções de avaliação dos níveis de utilização da rede, nomeadamente o Φ e o MLU. Estas funções assumiram um papel crucial na avaliação das diversas configurações topológicas testadas. A escolha das mesmas foi baseada na analise efetuada em diversos trabalhos, a partir dos quais foram identificadas diversas abordagens capazes de solucionar problemas de engenharia de tráfego. O Capítulo 2 termina com um estudo a diversos métodos de otimização, capazes de contornar a complexidade de problemas NP-Difíceis, e com a apresentação da *framework* na qual a implementação foi efetuada.

Seguiu-se a etapa de especificação e modelação do problema a resolver. Esta teve como principal finalidade identificar os pontos necessários para o solucionar. Primeiramente foram definidos dois modelos lineares para resolver uma versão relaxada do problema. Pretendia-se que os modelos permitissem avaliar os níveis de utilização dos recursos da

rede, nos quais se incluem os nodos da topologia que implementam os serviços e as ligações. Como tal, os modelos contemplam a utilização de dois pares de métricas MLU e MNU que minimizam a máxima utilização de ligações e nodos de processamento de pedidos, e Φ e Γ que procuram distribuir o tráfego pelas ligações e nodos de modo a minimizar uma função convexa de penalização. A principal diferença entre ambos os modelos situa-se na função objetivo que, no caso do modelo Φ permite evitar cenários de *bottlenecks* tanto nas ligações como nos nodos. Ainda nesta etapa foram introduzidos os algoritmos evolucionários, que utilizam os modelos MILP para selecionar os nodos de processamento e otimizar os pesos IGP. Para finalizar foram explicadas as vantagens de utilizar *machine learning* e de que forma este método de otimização se enquadrou no trabalho desenvolvido.

A etapa de desenvolvimento e integração na *framework* NetOpt consistiu num processo de análise à mesma por forma a integrar os novos conceitos relacionados com o NFV. Nesse sentido, foram reaproveitadas algumas classes já desenvolvidas na mesma para efetuar a representação da topologia e para configurar os algoritmos evolucionários. Para além do desenvolvimento dos modelos lineares, foram utilizados os simuladores da *framework* NetOpt. Estes tem por base o protocolo OSPF e a tecnologia SR, bem como, a utilização de balanceadores de tráfego como o ECMP. O desenvolvimento é representado neste capítulo através da linguagem de modelação UML. Devido à grande complexidade da *framework*, foi dada prioridade a diagramas de classes e de *packages*.

A etapa de análise de resultados avaliou a qualidade das abordagens propostas para solucionar o problema. Nesta componente é primeiramente realizado um estudo ao comportamento dos modelos lineares face a vários conjuntos de pedidos de encaminhamento e topologias. Esta análise permitiu estimar o tempo necessário para avaliar os níveis de utilização de uma configuração topológica. Face aos resultados obtidos é possível concluir que o tempo de execução do modelo é afetado pela complexidade da topologia, ou seja, pelo número de nodos e ligações da mesma. Outro fator que contribui para o aumento do tempo de execução dos MILPs é o número de pedidos. Apesar disso, a partir da comparação efetuada entre os modelos definidos no Capítulo 3 e os modelos disponibilizados pela *framework* NetOpt, é possível concluir que num cenário em que existe uma distribuição total dos serviços, os níveis de utilização das ligações são semelhantes. Isto porque, todos os serviços requisitados por um pedido são executados no caminho mais curto entre a origem e o destino do mesmo. No entanto, distribuir todos os serviços em todos os nodos da topologia não é uma solução aceitável, visto que, a instalação dos serviços tem custos associados. Nesse sentido, uma das soluções sugeridas seria utilizar os modelos lineares para determinar o local de execução dos serviços. No entanto, mesmo nos cenários de teste com 300 pedidos de encaminhamento, todos os serviços em cada nodo acabavam por ser utilizados. A introdução dos algoritmos evolucionários permitiu superar esta limitação gerando configurações topológicas nas quais a distribuição dos serviços

é alterada. A partir deste método de otimização foram consideradas várias abordagens. Uma consiste na minimização dos níveis de utilização dos nodos e das ligações. Naturalmente, a tendência das soluções encontradas foi a de procurar configurações com uma distribuição praticamente total dos serviços em todos os nodos da topologia. Nesse sentido foi necessário definir um limite ao número total de serviços, aplicando uma penalização a soluções cujo número de serviços fosse superior ao limite definido. A outra abordagem consistia em considerar o número de serviços como uma das características dos indivíduos nos algoritmos evolucionários. A partir da análise efetuada a esta abordagem foi possível analisar o *trade-off*, através de gráficos de Frente de Pareto, entre o número de serviços e a carga nas ligações e nos nodos. Através destes gráficos é possível obter o número de serviços mínimo, incluindo replicações dos mesmos, a partir do qual os níveis de utilização ultrapassariam as capacidades dos nodos e das ligações.

Em suma, a utilização dos algoritmos evolucionários, face a configurações geradas aleatoriamente, permitiu a obtenção de níveis de utilização dos recursos inferiores. No entanto, por forma a garantir uma utilização ponderada do *cluster* Search-ON2, a introdução de limites ao tempo de análise dos indivíduos, levou a que fossem detetadas avaliações com um *gap* para a solução ótima superior a 10%. Esta limitação faz com que os resultados sofram um decréscimo de qualidade nas topologias mais complexas. Isto porque, como a solução não está 100% otimizada, os níveis de utilização resultantes dessa configuração topológica, são superiores. Este fator pode levar a que os algoritmos evolucionários optem por configurações piores mas que, apesar disso, obtiveram níveis de utilização dos recursos inferiores devido ao *gap* menos elevado.

Dado que a utilização dos modelos em conjunto com os algoritmos evolucionários é um método de otimização com um elevado custo associado ao tempo de execução, a utilização destes métodos em cenários de otimização *online* é inviável. Para isso foram utilizadas redes neuronais. A partir destas é possível obter resultados, ainda que ligeiramente piores que os MILP, em tempo útil.

6.2 Principais Contribuições

Esta dissertação abordou a possibilidade de combinar duas tecnologias emergentes, nomeadamente o *Segment Routing* e o *Network Function Virtuzalization*. Estas possuem uma forte aplicabilidade em problemas relacionados com engenharia de tráfego que visam otimizar a utilização dos recursos disponíveis. Nesse sentido, este documento apresenta um conjunto de abordagens capazes de solucionar problemas relacionados com engenharia de tráfego. Face aos resultados obtidos, é possível concluir que a conjunção destas duas arquiteturas permite garantir uma boa gestão dos recursos disponíveis por parte das operadoras

de telecomunicações. Garantir uma correta utilização dos recursos é cada vez mais importante pois estes, devido ao surgimento de redes 5G, são cada vez mais escassos.

6.3 TRABALHO FUTURO

Os objetivos definidos para esta dissertação foram cumpridos. Apesar disso, o trabalho desenvolvido poderia ser melhorado em alguns aspetos. Uma das melhorias a considerar seria a possibilidade de executar os algoritmos evolucionários com um intervalo de tempo superior. Mais concretamente, aumentar o número de gerações e o tempo de execução das funções de avaliação de indivíduos. Este aumento iria melhorar a qualidade das soluções obtidas, especialmente nas topologias de maior complexidade, dado que o tempo de execução é prejudicado pelo aumento do número de variáveis dos MILP. Outra melhoria a considerar seria a compilação de um conjunto de dados com um maior número de entradas. Face a esse aumento seria expectável conseguir um melhor desempenho com redes neuronais e LSTMs face aos modelos mais simples, SVM e RF. Com esta melhoria seria possível minimizar a diferença entre os níveis de utilização obtidos pelas redes neuronais e os níveis obtidos pelos MILPs. Para além disso, seria importante investigar mais trabalhos na área com redes LSTM com o objetivo de identificar implementações que possam ser mais adequadas ao problema que foi endereçado nesta dissertação.

BIBLIOGRAFIA

[1] J. Ali, R. Khan, N. Ahmad, and I. Maqsood. Random forests and decision trees. *International Journal of Computer Science Issues(IJCSI)*, Vol. 9, September 2012.

[2] F. Baker. Requirements for IP Version 4 Routers. RFC 1812, June 1995. URL https://rfc-editor.org/rfc/rfc1812.txt.

[3] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte. Orchestrating virtualized network functions. *IEEE Transactions on Network and Service Management*, Vol. 13(4):725–739, Dec 2016. ISSN 2373-7379. doi: 10.1109/TNSM.2016.2569020.

[4] C. J. Bernardos, A. Rahman, J.-C. Zúñiga, L. M. Contreras, P. A. Aranda, and P. Lynch. Network Virtualization Research Challenges. RFC 8568, Apr. 2019. URL https://rfc-editor.org/rfc/rfc8568.txt.

[5] R. Bhatia, F. Hao, M. Kodialam, and T. V. Lakshman. Optimized network traffic engineering using segment routing. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 657–665, April 2015. doi: 10.1109/INFOCOM.2015.7218434.

[6] R. Boutaba, M. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. Caicedo Rendon. A comprehensive survey on machine learning for networking: Evolution, applications and research opportunities. *Journal of Internet Services and Applications*, Vol. 9, May 2018. doi: 10.1186/s13174-018-0087-2.

[7] L. Breiman. Random forests. *Machine Learning*, Vol. 45(1):5–32, Oct. 2001. ISSN 0885-6125. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

[8] R. Callon. Use of OSI IS-IS for routing in TCP/IP and dual environments. RFC 1195, Dec. 1990. URL https://rfc-editor.org/rfc/rfc1195.txt.

[9] J. Cervantes, F. García-Lamont, L. Rodríguez, and A. Lopez-Chau. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, Vol. 408, May 2020. doi: 10.1016/j.neucom.2019.10.118.

[10] A. Cianfrani, M. Listanti, and M. Polverini. Incremental deployment of segment routing into an isp network: a traffic engineering perspective. *IEEE/ACM Transactions on Networking*, Vol. 25(5):3146–3160, Oct 2017. ISSN 1558-2566. doi: 10.1109/TNET.2017.2731419.

[11] A. E. Eiben and M. Schoenauer. Evolutionary computing. *CoRR*, abs/cs/0511004, 2005. URL http://arxiv.org/abs/cs/0511004.

[12] ETSI GS NFV 002. Network functions virtualization (NFV); architectural framework v1.1.1. Technical report, ETSI, October 2013. URL http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf.

[13] P. Evangelista, P. Maia, and M. Rocha. Implementing metaheuristic optimization algorithms with jecoli. In *Proceedings of the 2009 Ninth International Conference on Intelligent Systems Design and Applications*, ISDA '09, page 505–510, USA, 2009. IEEE Computer Society. ISBN 9780769538723. doi: 10.1109/ISDA.2009.161. URL https://doi.org/10.1109/ISDA.2009.161.

[14] A. Farrel, O. Komolafe, and S. Yasukawa. An Analysis of Scaling Issues in MPLS-TE Core Networks. RFC 5439, Feb. 2009. URL https://rfc-editor.org/rfc/rfc5439.txt.

[15] C. Filsfils, N. K. Nainar, C. Pignataro, J. C. Cardona, and P. Francois. The segment routing architecture. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec 2015. doi: 10.1109/GLOCOM.2015.7417124.

[16] B. Fortz and M. Thorup. Internet traffic engineering by optimizing ospf weights. In *in Proc. IEEE INFOCOM*, volume Vol. 2, pages 519 – 528, 02 2000. ISBN 0-7803-5880-5. doi: 10.1109/INFCOM.2000.832225.

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, Vol. 9(8):1735–1780, Nov. 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[18] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, Vol. 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

[19] I. ILOG. Ibm cplex ilog optimization studio, 2020. URL https://www.ibm.com/products/ilog-cplex-optimization-studio.

[20] V. Jain and B. Edgeworth. *Troubleshooting BGP: A Practical Guide to Understanding and Troubleshooting BGP*. Cisco Press, 1st edition, 2016. ISBN 1587144646.

[21] S. Knight, H. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The internet topology zoo. *Selected Areas in Communications, IEEE Journal on*, Vol. 29(9):1765 –1775, october 2011. ISSN 0733-8716. doi: 10.1109/JSAC.2011.111002.

[22] M.-C. Lee and J.-P. Sheu. An efficient routing algorithm based on segment routing in software-defined networking. *Computer Networks*, Vol. 103, Apr 2016. doi: 10.1016/j.comnet.2016.03.017.

[23] D. Magoni and J.-J. Pansiot. Analysis of the autonomous system network topology. *Computer Communication Review*, Vol. 31:pages 26–37, July 2001. doi: 10.1145/505659.505663.

[24] G. S. Malkin. RIP Version 2. RFC 2453, Nov. 1998. URL https://rfc-editor.org/rfc/rfc2453.txt.

[25] A. Medina, A. Lakhina, I. Matta, and J. Byers. Brite: an approach to universal topology generation. In *MASCOTS 2001, Proceedings Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 346–353, 2001.

[26] A. Mishra, A. Sahoo, B. Dalvi, and T. Zhu. Wospf: A traffic engineering solution for ospf networks. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, Dec 2016. doi: 10.1109/GLOCOM.2016.7842393.

[27] M. Mohammed, M. Khan, and E. Bashier. *Machine Learning: Algorithms and Applications*. CRC Press., 07 2016. ISBN 9781498705387. doi: 10.1201/9781315371658.

[28] J. Moy. OSPF Version 2. RFC 2328, Apr. 1998. URL https://rfc-editor.org/rfc/rfc2328.txt.

[29] P. Ngatchou, A. Zarei, and A. El-Sharkawi. Pareto multi objective optimization. In *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*, pages 84–91, 2005.

[30] V. Pereira, P. Sousa, P. Cortez, M. Rio, and M. Rocha. Comparison of single and multi-objective evolutionary algorithms for robust link-state routing. In A. Gaspar-Cunha, C. Henggeler Antunes, and C. C. Coello, editors, *Evolutionary Multi-Criterion Optimization*, pages 573–587, Cham, 2015. Springer International Publishing. ISBN 978-3-319-15892-1.

[31] V. Pereira, M. Rocha, and P. Sousa. Optimizing segment routing using evolutionary computation. *Procedia Computer Science*, Vol. 110:312 – 319, 2017. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2017.06.100. URL http://www.sciencedirect.com/science/article/pii/S1877050917312784. 14th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017) / 12th International Conference on Future Networks and Communications (FNC 2017).

[32] V. Pereira, M. Rocha, and P. Sousa. Traffic engineering with three-segments routing. *IEEE Transactions on Network and Service Management*, Vol. 17(3):1896–1909, 2020. doi: 10.1109/TNSM.2020.2993207.

[33] V. M. S. Pereira. *Intradomain routing optimization based on evolutionary computation*. PhD thesis, University of Minho, 2019.

[34] S. Previdi, C. Filsfils, B. Decraene, S. Litkowski, M. Horneffer, and R. Shakir. Source Packet Routing in Networking (SPRING) Problem Statement and Requirements. RFC 7855, May 2016. URL https://rfc-editor.org/rfc/rfc7855.txt.

[35] J. Reis, M. Rocha, T. K. Phan, D. Griffin, F. Le, and M. Rio. Deep neural networks for network routing. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2019. doi: 10.1109/IJCNN.2019.8851733.

[36] Y. Rekhter, S. Hares, and T. Li. A Border Gateway Protocol 4 (BGP-4). RFC 4271, Jan. 2006. URL https://rfc-editor.org/rfc/rfc4271.txt.

[37] Search-ON2. "search-on2: Revitalization of hpc infrastructure of uminho", 2014. URL http://search6.di.uminho.pt/wordpress/?p=246.

[38] A. Shrestha and A. Mahmood. Review of deep learning algorithms and architectures. *IEEE Access*, Vol. 7:53040–53065, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2912200.

[39] N. Srinivas and K. Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, Vol. 2(3):221–248, 1994.

[40] M. Tabassum, K. Mathew, and A. Ali. *A Comparative Study of IGP and EGP Routing Protocols, Performance Evaluation along Load Balancing and Redundancy across Different AS*, pages 493 – 498. Newswood Limited, 03 2016. ISBN 2078-0958.

[41] G. Trimponias, Y. Xiao, H. Xu, X. Wu, and Y. Geng. On traffic engineering with segment routing in SDN based wans. *IEEE/ACM Transactions on Networking*, abs/1703.05907, 2017. URL http://arxiv.org/abs/1703.05907.

[42] R. E. Uhrig. Introduction to artificial neural networks. In *Proceedings of IECON '95 - 21st Annual Conference on IEEE Industrial Electronics*, volume 1, pages 33–37, Nov 1995. doi: 10.1109/IECON.1995.483329.

[43] P. L. Ventre, S. Salsano, M. Polverini, A. Cianfrani, A. Abdelsalam, C. Filsfils, P. Camarillo, and F. Clad. Segment routing: a comprehensive survey of research activities, standardization efforts and implementation results. *IEEE Communications Surveys Tutorials*, pages 1–1, 2020. doi: 10.1109/COMST.2020.3036826.

[44] P. A. Vikhar. Evolutionary algorithms: A critical review and its future prospects. In *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, pages 261–265, Dec 2016. doi: 10.1109/ICGTSPICC.2016.7955308.

[45] A. Viswanathan, E. C. Rosen, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, Jan. 2001. URL https://rfc-editor.org/rfc/rfc3031.txt.

[46] L. Wang, Z. Lu, X. Wen, R. Knopp, and R. Gupta. Joint optimization of service function chaining and resource allocation in network function virtualization. *IEEE Access*, Vol. 4:8084–8094, 2016. doi: 10.1109/ACCESS.2016.2629278.

[47] Y. Wang, X. Zhang, L. Fan, S. Yu, and R. Lin. Segment routing optimization for vnf chaining. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–7, 2019.

[48] Y. Xie, Z. Liu, S. Wang, and Y. Wang. Service function chaining resource allocation: A survey, 2016.

[49] D. Xu, M. Chiang, and J. Rexford. Deft: Distributed exponentially-weighted flow splitting. In *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, pages 71–79, May 2007. doi: 10.1109/INFCOM.2007.17.

[50] D. Xu, M. Chiang, and J. Rexford. Link-state routing with hop-by-hop forwarding can achieve optimal traffic engineering. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 466–474, April 2008. doi: 10.1109/INFOCOM.2008.94.

[51] B. Yi, X. Wang, K. Li, S. k. Das, and M. Huang. A comprehensive survey of network function virtualization. *Computer Networks*, Vol. 133:212 – 262, 2018. ISSN 1389-1286. doi: https://doi.org/10.1016/j.comnet.2018.01.021. URL http://www.sciencedirect.com/science/article/pii/S1389128618300306.

[52] Z. Zaman, S. Rahman, and M. Naznin. Novel approaches for vnf requirement prediction using dnn and lstm. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2019. doi: 10.1109/GLOBECOM38437.2019.9014320.

[53] S. Zhang. An overview of network slicing for 5g. *IEEE Wireless Communications*, Vol. 26(3):111–117, June 2019. ISSN 1558-0687. doi: 10.1109/MWC.2019.1800234.

[54] E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems. Proceedings of the EUROGEN'2001*, volume 3242, 01 2001.