POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione
DOTTORATO DI RICERCA IN INGEGNERIA DELL'INFORMAZIONE

Overlay Networks Formation: Models and Algorithms

Doctoral Dissertation of:
Jocelyne Elias

Advisor:
Prof. Antonio Capone

Tutor:
Prof. Michele D'Amico

Supervisor of the Doctoral Program:
Prof. Patrizio Colaneri

2009-XXII

Abstract

Overlay networks have recently emerged as an effective means to provide a flexible, robust, and scalable platform for distributed applications, while leaving the underlying Internet infrastructure unchanged. This work tackles the overlay network design problem considering both centralized and fully distributed approaches.

On one hand, we address the centralized overlay network design problem using the Service Overlay Network (SON) paradigm, and we propose several mathematical models and heuristics for the optimal design of SONs. More specifically, we introduce two network optimization models that determine the optimal assignment of users to access overlay nodes, as well as the capacity reserved for each overlay link, while taking accurate account of traffic routing. We also propose two overlay network design models that further select the optimal number and location of the overlay nodes to be deployed, as well as the optimal coverage of network users to maximize the SON operator's profit. Furthermore, we develop a set of efficient SON design heuristics to get near-optimal solutions for large-scale network instances in a reasonable computation time. Finally, we perform an extensive performance evaluation of the proposed centralized optimization framework in several realistic network scenarios.

On the other hand, this thesis also proposes two novel socially-aware overlay network design games to deal with the fully distributed overlay network formation problem. The first game combines both individual and social concerns in a unified and flexible manner, and the second game uses a Stackelberg approach, where the overlay network administrator leads the users to a system-wide efficient equilibrium by buying an appropriate subset of the overlay network links. We evaluate the performance of the proposed games, through the determination of bounds on the Price of Anarchy and other efficiency measures, as well as by simulating several realistic network scenarios, including real ISP topologies.

Numerical results demonstrate that: (1) our proposed SON design models and heuristics plan very effective overlay networks, even in the case of very large network instances, and (2) our proposed distributed network formation algorithms are able to lead client users to form stable and efficient overlay networks, obtaining in

several cases the optimal solution that could be planned by a central authority. Hence, we conclude that the proposed solutions can be very effective when applied to real Internet scenarios.

Riassunto

Le reti overlay rappresentano una tecnologia efficace per fornire una piattaforma flessibile, robusta e scalabile per le applicazioni distribuite, mentenendo al contempo intatta la struttura della rete Internet. La tesi di dottorato affronta il problema del design di reti overlay sia da un punto di vista centralizzato che totalmente distribuito.

Per prima cosa è stato affrontato il problema del design centralizzato di reti overlay utilizzando il paradigma delle cosiddette Service Overlay Networks (SONs), proponendo diversi modelli matematici ed euristiche per il progetto ottimale delle SON. Più in dettaglio, sono stati introdotti due modelli di ottimizzazione che determinano l'assegnazione ottimale degli utenti ai nodi overlay di accesso, assieme al routing ed alla capacità ottimale riservata su ogni link overlay.

Inoltre, sono stati proposti due modelli che selezionano anche il numero e la posizione ottimale dei nodi overlay da installare, così come la copertura più efficiente degli utenti della rete al fine di massimizzare il profitto delloperatore di rete. Sono state poi sviluppate delle euristiche efficienti per il design di reti SON, che ottengono risultati vicini all'ottimo in breve tempo, anche per reti di grandi dimensioni. Infine, si è effettuata un'accurata valutazione delle prestazioni degli algoritmi e modelli proposti in diversi scenari di rete realistici.

Come ulteriore contributo, si sono proposti due nuovi giochi "socially-aware" per il design distribuito delle reti overlay, in particolare, e più in generale per il problema della formazione distribuita delle reti di telecomunicazione. Il primo gioco permette di combinare sia interessi di tipo egoistico che sociale in modo unificato, mentre il secondo gioco utilizza un approccio alla Stackelberg nel quale l'amministratore della rete stimola gli utenti a raggiungere un equilibrio di Nash globalmente efficiente, contribuendo alla formazione di un sottoinsieme appropriato dei link della rete overlay. Si sono valutate le prestazione dei giochi proposti, determinando dei limiti al "price of anarchy" e ad altre misure di efficienza, simulando inoltre diversi scenari di rete che includono anche topologie reali di Internet Service Provider. In sintesi, i risultati numerici hanno dimostrato che: (1) i modelli proposti per il design delle reti SON sono in grado di pianificare reti overlay molto efficienti, anche in presenza di istanze di rete molto grandi e (2) gli algoritmi distribuiti proposti per la formazione

delle reti conducono gli utenti a formare reti overlay stabili ed efficienti, ottenendo nella maggior parte dei casi la soluzione ottimale che potrebbe essere progettata da un’autorit centrale.

List of Related Publications

1. J. ELIAS, F. Martignon, K. Avrachenkov, G. Neglia, Socially-Aware Network Design Games, in Proceedings of the 29th IEEE Conference on Computer Communications (INFOCOM 2010), March 2010, San Diego, CA, USA.

2. E. Altman, J. ELIAS, F. Martignon, A Game Theoretic Framework for joint Routing and Pricing in Networks with Elastic Demands, in Proceedings of the Fourth International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS 2009), October 2009, Pisa, Italy.

3. J. ELIAS, F. Martignon, G. Carello, Very Large-Scale Neighborhood Search Algorithms for the Design of Service Overlay Networks, submitted to Telecommunication Systems, March 2009.

4. A. Capone, J. ELIAS, F. Martignon, Routing and Resource Optimization in Service Overlay Networks, Elsevier Computer Networks, vol. 53, no. 2, 13 February 2009, pp. 180-190.

5. A. Capone, J. ELIAS, F. Martignon, Models and Algorithms for the Design of Service Overlay Networks, IEEE Transactions on Network and Service Management, vol. 5, no. 3, September 2008, pp. 143-156.

6. A. Capone, J. ELIAS, F. Martignon, Optimal Design of Service Overlay Networks, in Proceedings of the Fourth International Telecommunication Networking Workshop on QoS in Multiservice IP Networks, IT-NEWS 2008, Venice, Italy, February 2008.

Acknowledgements

I would like to thank my advisor, Professor Antonio Capone, for his support and guidance during my P.hD. studies at Dipartimento di Elettronica e Informazione, Politecnico di Milano. I will always appreciate his cheerful encouragement over these three years.

I would also like to thank Professor Luigi Fratta and my tutor Michele D'Amico for supporting me during my P.hD. studies.

I would like to express my sincere thanks to Professors Vicente Casares Giner and Konstantin Avrachenkov for accepting to be my Thesis Reviewers.

I would like to thank Mr. Fabio Martignon for collaborating with me on several valuable ideas.

I am particularly grateful to Professors Philippe Nain, Eitan Altman, Konstantin Avrachenkov, Giovanni Neglia and all the MAESTRO team for my research visits at INRIA, Sophia Antipolis.

I wish to thank all my colleagues and friends at DEI, Politecnico di Milano.

This thesis is dedicated to my family, to my parents on both sides and to all my village.

There is a long list of people to thank. All contributed in one way or another to this thesis. To them all - I am forever thankful.

Table of contents

Chapter 1

Introduction

The Overlay Network paradigm has recently emerged as a viable and very effective means to avoid network-level inefficiencies in the Internet, enabling, at the same time, a variety of popular applications including peer-to-peer file sharing, content distribution and server deployment. Overlay Networks today represent an alternative and very promising architecture able to provide end-to-end Quality of Service guarantees in the Internet, while leaving the underlying Internet infrastructure unchanged [1, 2, 3, 4].

Overlay networks are virtual topologies that use a combination of shared and dedicated resources, to provide a simple network view that conceals unnecessary details about the underlying topology. A typical overlay network architecture is depicted in Figure 1.1, where overlay nodes reside in the underlying ISP networks, and are interconnected by virtual links which correspond to one or more IP-layer links.

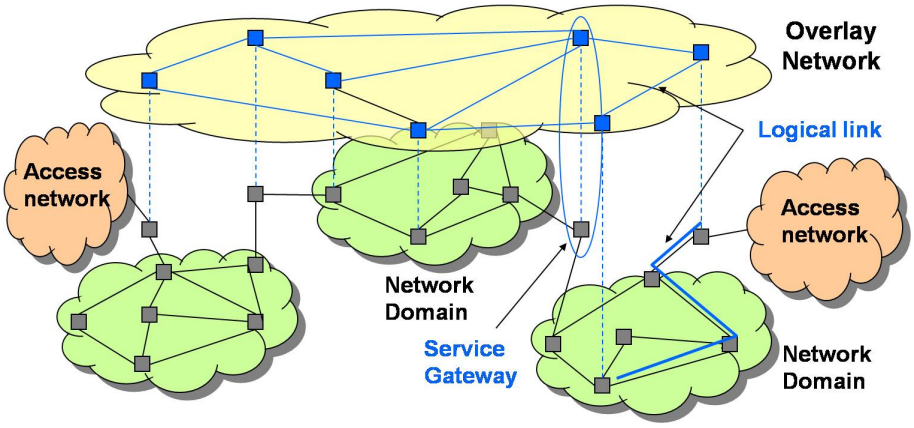


Figure 1.1: Overlay Network Architecture.

The choice of where to install overlay nodes, which links establish between such nodes and, in general, how to allocate resources to the overlay network has a deep impact on the effectiveness of the resulting network and on its overall cost.

This thesis investigates two alternative approaches for the design of efficient overlay networks, namely:

1. a centralized network optimization approach, which is based on the Service Overlay Network (SON) paradigm [1];

2. a fully distributed approach, where the overlay network design and operation is carried out by a large number of independent actors (e.g., the overlay user clients), all of whom seek to selfishly optimize their own utility.

In the remainder of this Chapter, we first illustrate the proposed centralized optimization approach for the overlay network design problem, introducing at the same time the SON paradigm. Then, we focus on distributed overlay network management, proposing two novel overlay network formation games that lead user clients to form efficient overlay networks in a fully distributed, non-cooperative manner. Finally, we summarize the main contributions and give the detailed structure of this thesis.

1.1 Service Overlay Network Design

Service Overlay Networks (SONs) have recently emerged as alternative and very promising architectures able to provide end-to-end Quality of Service guarantees in the Internet, while leaving the underlying Internet infrastructure unchanged [1, 2, 3, 4, 5].

A SON is an application-layer network built on top of traditional IP-layer networks. In general, the SON is operated by a third-party ISP that owns a set of overlay nodes residing in the underlying ISP domains. These overlay nodes perform service-specific data forwarding and control functions, and are interconnected by virtual overlay links which correspond to one or more IP-layer links [1].

The service overlay architecture is based on business relationships between the SON, the underlying ISPs, and the users. The SON establishes bilateral service level agreements with the individual underlying ISPs to install overlay nodes and purchase the bandwidth needed for serving its users. On the other hand, the users subscribe to SON services, which will be guaranteed regardless of how many IP domains are crossed by the users' connection. The SON gains from users' subscriptions. Although the quality requirements that a SON must satisfy may be different (e.g. bandwidth, delay, delay jitter, packet loss), we assume they are mapped to an equivalent

bandwidth [1, 5]. To assure the bandwidth for the SON, the underlying ISPs have several technical options: they can lease a transmission line to the SON, use bandwidth reservation mechanisms or create a separate Label Switched Path if MPLS [6] is available in their networks.

Obviously, the deployment of Service Overlay Networks can be a capital-intensive investment. It is therefore imperative to develop efficient network design tools that consider the cost recovery issue for a SON. The main costs of SON deployment include the overlay nodes installation cost and the cost of the bandwidth that the SON must purchase from the underlying network domains to support its services.

The topology design problem for Service Overlay Networks is considered by very few works [5, 7, 8, 9, 10, 11, 12, 13] which make several limiting assumptions:

- the number and location of overlay nodes are pre-determined, while the overlay node placement is a critical issue in the deployment of the SON architecture.
- A full coverage of all traffic demands must be provided, while the main goal of a SON operator would be to maximize its profit by choosing which users to serve based on the expected revenue.
- The capacities of overlay nodes/links are unlimited, thus assuming that the underlying ISPs will always be able to provide bandwidth to the SON.
- Only small network instances are considered, with a limited number of connections and overlay nodes.

This thesis overcomes these limitations by first addressing the joint user assignment and traffic routing problem, proposing two novel optimization models that determine the optimal assignment of users to access overlay nodes, as well as the capacity reserved for each overlay link, while taking accurate account of traffic routing. The first model minimizes the network installation cost while providing full coverage to all the network's users. The second model maximizes the SON profit by further selecting which users to serve in order to make its operation profitable, and also includes a budget constraint that the SON operator can specify to limit its economic risks in the deployment of the overlay network. We then extend such models to consider the more complex SON design problem, where the number and positions of overlay nodes to be deployed are optimized. To this end we present two SON design models that jointly optimize (1) the number and location of overlay nodes, (2) the user assignment to access overlay nodes, (3) the traffic routing and (4) the capacity dimensioning of overlay links.

The SON design problems are NP-hard, however, the proposed Mixed Integer Linear Programming (MILP) formulations can be solved to the optimum for realistic-size instances in reasonable time. More specifically, the formulation that considers

only the user assignment and routing problem can be solved to the optimum even for large-scale instances in a short computing time.

To tackle large-size instances for the global SON design problem, we propose two simple but effective heuristic approaches able to provide near-optimal solutions in a reasonable computation time. The proposed algorithms are based on the decomposition of the model into sub-problems and on the solution of the continuous relaxation. 0-1 feasible solutions are then obtained using a randomized rounding technique.

Regrettably, in some scenarios the proposed heuristics are unable to provide a good solution to the SON design problem due, in particular, to huge memory consumption and computational effort.

For this reason, we further propose an efficient tabu search based approach that uses polynomial size and Very Large-Scale Neighborhoods (VLSN). VLSN is used once the local minimum is reached, to "escape" from it and widen the set of explored solutions; it can therefore be seen as a diversification step for the tabu search. We demonstrate that the proposed VLSN-based heuristic is able to design efficient overlay networks even in very large-scale topology scenarios.

1.2 Distributed Overlay Network Formation

In many scenarios, the overlay network design is not enforced by a central authority, but arises from the interactions of several self-interested agents: each user client can decide the set of connections to establish.

Network design with selfish users has been the focus of several recent works [14, 15, 16, 17, 18, 19, 20], which have modeled how independent selfish agents can build or maintain a large network by paying for possible edges. Each user's goal is to connect a given set of terminals with the minimum possible cost. Game theory is the natural framework to address the interaction of such self-interested users (or players). A *Nash Equilibrium* (NE) is a set of users choices, such that none of them has an incentive to deviate unilaterally. For this reason the corresponding networks are said to be *stable*.

However, Nash equilibria in network design games can be much more expensive than the optimal, centralized solution. This is mainly due to the lack of cooperation among network users, which leads to design costly networks.

Actually, the majority of existing works assume that users are completely non-cooperative. However, this assumption is not entirely realistic, for example when network design involves long-term decisions (e.g., in the case of Autonomous Systems peering relations). Moreover, incentives could be introduced by some external authority (e.g., the overlay administrator) in order to increase the users' cooperation

level.

In this work we overcome this limitation by first proposing a novel overlay network design game, the Socially-Aware Network Design (SAND) game, where users are characterized by an objective function that combines both *individual* and *social* concerns in a unified and flexible manner. More specifically, the cost function of each user is a combination of its own path cost (the *selfish* component) and the overall network cost, which represents the *social* component. A parameter (α) weights the relative importance of the network cost with respect to the user path cost. Changing the value of α permits to take into account different levels of social awareness or user cooperation.

We investigate systematically the impact of cooperation among network agents on the system performance, through the determination of bounds on the *Price of Anarchy* (PoA), the *Price of Stability* (PoS) and the *Reachable Price of Anarchy* ($RPoA$) of the proposed game. They all quantify the loss of efficiency as the ratio between the cost of a specific stable network and the cost of the optimal network, which could be designed by a central authority. In particular the PoA, first introduced in [21], considers the worst stable network (that with the highest cost), while the PoS [14] considers the best stable network (that with the lowest cost); finally, the $RPoA$ considers only Nash equilibria reachable via best response dynamics from the empty solution [19]. Hence, PoA and $RPoA$ indicate the maximum degradation due to distributed users decisions (anarchy), while the PoS indicates the minimum cost to pay to have a solution robust to unilateral deviations.

Our analytical results show that as α increases, i.e., when users are more sensitive to the social cost, the PoS converges to 1, i.e., the best stable network is more efficient, as expected. Surprisingly, an opposite result holds for the worst case. Indeed, for large α values (highly socially-aware users) the worst stable network can be much more expensive than the networks designed by purely selfish users.

For this reason, we further propose a Stackelberg approach, the Network Administrator - Driven SAND game (NAD-SAND), which enables very efficient Nash equilibria, avoiding worst-case scenarios: a leader (e.g., the network administrator) buys an appropriate subset of the network links (i.e., those belonging to the minimum cost generalized Steiner tree covering all source/destination pairs), inducing the followers (the network users) to reach an efficient Nash equilibrium.

Summary of the main Numerical Results

We evaluate the performance of the proposed centralized and distributed overlay network design approaches: first, we provide numerical results for the centralized

overlay network design approach in a set of realistic-size instances and investigate the impact of different parameters on the SON design problem, such as number and installation cost of overlay nodes, bandwidth costs, traffic demands and SON operator's budget. We further determine bounds to the performance achievable by any optimization algorithm solving continuous relaxations of the proposed models. The numerical results show that in the considered network scenarios the proposed heuristics perform very close to the optimal solution provided by the integer models with a short computing time.

Then, we measure the performance of the proposed distributed overlay network formation games in several network topologies, including realistic scenarios where players build an overlay on top of real Internet Service Provider networks, and we observed that socially-aware users always generate better networks. Furthermore, we observe that the proposed Stackelberg approach achieves dramatic performance improvements in all the considered scenarios, even for small α values, since it leads most of the times to the optimal (least cost) network. Hence, introducing some incentives to make users more socially-aware could be an effective solution to achieve stable and efficient networks in a distributed way.

1.3 Thesis Contributions

In summary, the main contributions of this thesis are:

- two network optimization models that determine the *optimal assignment* of users to access overlay nodes, as well as the *capacity* reserved for each overlay link, while taking accurate account of traffic routing.
- Two overlay network design models that further select the *optimal number* and *location* of the overlay nodes to be deployed, as well as the *optimal coverage* of network users to maximize the SON operator's profit.
- A set of efficient SON design heuristics that get near-optimal solutions for large-scale instances in a reasonable computation time.
- An extensive performance evaluation of the proposed centralized optimization framework in several realistic network scenarios.
- Two novel socially-aware overlay network design games: the Socially-Aware Network Design game, which combines both *individual* and *social* concerns in a unified and flexible manner, and a Stackelberg game where the overlay network administrator leads the users to a system-wide efficient equilibrium by buying an appropriate subset of the overlay network links.

- A thorough numerical evaluation of the proposed games, through the determination of bounds on the Price of Anarchy, the Price of Stability and the Reachable Price of Anarchy of such games, as well as by simulation of several realistic network scenarios, including real ISP topologies.

1.4 Thesis Structure

The thesis is structured as follows:

Chapter 2 discusses the most notable works related to the centralized overlay design problem and to the distributed overlay network formation approach. Furthermore, the novelties of the approaches proposed in this thesis are underlined compared to existing network design schemes.

In Chapter 3, we first propose two novel user assignment and routing optimization models, based on mathematical programming, which take into account the individual requirements of the end-users, the connectivity between overlay nodes and the management of traffic flows. The objective of the first model is the minimization of the overall network installation cost while ensuring full coverage of all end-users. The second model maximizes the SON profit by choosing which users to serve based on the expected gain and budget constraints specified by the SON operator. We then address the topology design problem for Service Overlay Networks, optimizing the number and positions of the overlay nodes to be deployed in addition to all the variables considered in the user assignment and routing problem. Finally, we introduce two efficient heuristics that obtain near-optimal solutions for large-scale instances in a reasonable computation time. Such heuristics are based on continuous relaxation and randomized rounding techniques.

Chapter 4 evaluates the performance of the models and heuristics introduced in the previous Chapter in several realistic network scenarios.

In Chapter 5, we develop a novel and efficient heuristic approach that solves the user assignment and overlay node placement problem using tabu search along with polynomial size and very large-scale (VLSN) neighborhoods. VLSN is used to perform the diversification step on the local optimum obtained by tabu search. We finally provide numerical results of the proposed heuristic on a set of realistic, large-size instances, and discuss the effect of different parameters on the characteristics of the planned networks.

In Chapter 6, we give a short introduction to *Game Theory*, including strategic games, dominant strategies, the Nash equilibrium concept and algorithmic game theory. An overview is provided on the existence and computation of Nash equilibria, as well as the definition of the *price of anarchy* and the *price of stability* performance

figures.

In Chapter 7, we address the distributed overlay network formation problem using the game theory paradigm. To this aim, we first propose the Socially-Aware Network Design (SAND) game, where overlay clients are partially socially-aware because their utility function is a weighted sum of individual and global costs. We also study the efficiency of the equilibria achieved by our game, deriving bounds to the Price of Anarchy, the Price of Stability and the Reachable Price of Anarchy. Then, we introduce the Network Administrator-Driven SAND (NAD-SAND) game using a Stackelberg approach, where a leader (e.g., the overlay network administrator) architects the desired network buying an appropriate subset of network's links, inducing the followers (the network users) to reach an efficient Nash equilibrium. The performance of the SAND and NAD-SAND games is measured in several network scenarios, including real ISP topologies, where players build an overlay on top of real Internet Service Provider networks.

Conclusions and directions for future work are presented in Chapter 8. Finally, Appendix A provides an overview of the Hub Location problem, the Generalized Steiner and Steiner Tree problems. A brief description of *AMPL* is also given.

Chapter 2

Related Work

This Chapter reviews the most notable works related to the centralized optimization of overlay networks (Section 2.1) and to the distributed overlay network formation problem (Section 2.2). In both cases, we point out the limitations of existing works and the main novelties introduced in this thesis.

2.1 Centralized Optimal Design of Service Overlay Networks

Several works have appeared in the literature with the purpose of providing optimal routing and topology design in different contexts, such as wired backbone networks [22, 23, 24, 25], wireless networks [26, 27], and recently Service Overlay Networks [5, 7, 8, 9, 10, 11, 12, 13].

An adaptive topology design framework for SONs is presented in [5] to assure inter-domain QoS, and a set of heuristics is proposed to solve the least-cost topology design problem. A similar problem is investigated in [7], where end-systems and overlay nodes are connected through ISPs that support bandwidth reservations; simulated annealing is used as heuristic to provide solutions for large-sized networks. Another set of heuristics for SON design is proposed in [8]; these heuristics aim to construct an overlay topology maintaining the connectivity between overlay nodes under various IP-layer path failure scenarios. However, all these works formulate the design problem considering full coverage of all traffic demands and assuming that locations of overlay nodes are given and the underlying ISPs are always able to provide resources to the SON.

The work in [28] considers a generalized cost model in the formulation of the design problem, and provides both exact and approximate solutions.

A generalized framework for the SON design problem is proposed in [29] where different characteristics are considered: 1) single-homed/multi-homed end-systems 2) usage-based/fixed cost model and 3) capacitated/uncapacitated networks. Optimal and approximation algorithms are also provided.

Reference [9] deals with dynamic topology construction to adapt to the underlying network topology changes. An architecture for topology-aware overlay networks is proposed to enhance the availability and performance of end-to-end applications by exploring the dependency between overlay paths. Several clustering-based heuristics for overlay node placement and a routing mechanism are also introduced.

The dynamic overlay network reconfiguration issue is addressed in [10], where the main goal is to find the optimal reconfiguration policies that can both accommodate time-varying communication requirements and minimize the total overlay network cost.

The problem of overlay node placement is addressed in [11, 12, 13]. In [11] the authors consider how to place service nodes optimally in a network, balancing the need to minimize the number of nodes and to limit the distance between users and service nodes. This work, however, only proposes optimization algorithms for the uncapacitated version of the coverage problem. The work in [12] focuses on designing an overlay network that maximizes the number of unicast and multicast connections with deterministic delay requirements, without considering link costs. Finally, the overlay node placement problem is investigated in [13] to improve routing reliability and TCP performance. This paper, however, assumes that overlay nodes and links have infinite capacities, and does not take into account the costs involved in the deployment of the overlay network.

The construction of efficient multicast trees in overlay networks is the primary focus of several studies [30, 31, 32, 33], and is not considered in this thesis since we focus our analysis on unicast traffic.

In summary, the above cited techniques are less general than our current work since they deal with the design problem considering at least one of the following special cases: 1) the number and location of overlay nodes are pre-determined, 2) the routing is fixed and known, 3) there are no capacity constraints on overlay links/nodes, 4) full coverage of all network users is provided without considering the SON profit maximization issue, and 5) only small and medium-size topologies are considered, with a limited number of connections and overlay nodes. Our work tackles the SON design problem taking into account all these issues within a general optimization framework that in addition considers the expected profit of the SON operator and a budget constraint that can limit the economic risk. Furthermore, we underline that none of the above works uses VLSN search techniques to solve the SON design problem. A survey on such techniques can be found in [34, 35].

2.2 Distributed Overlay Network Design

Several recent works focused on network design with selfish users [14, 15, 16, 17, 18, 19].

The so-called *Shapley network design game* is proposed in [14]. In this game, each player chooses a path from its source to its destination, and the overall network cost is shared among the players in the following way: each player pays for each edge a proportional share of the edge cost, i.e., the edge cost divided by the number of players that pass through such edge.

Of all the ways to share the social cost among the players, this proportional sharing method enjoys several desirable properties. First, it is budget balanced, in that it partitions the social cost among the players. Second, it can be derived from the Shapley value, and as a consequence is the unique cost-sharing method satisfying certain fairness axioms. Third, it admits pure strategy NEs. Specifically, Anshelevich et al. showed in [14] that a pure-strategy Nash equilibrium always exists, that $PoA = k$ and the Price of Stability is equal to the k-th harmonic number, i.e. $PoS = \mathcal{H}_k = \sum_{i=1}^{k} 1/i = O(\ln(k))$.

The network design model presented in [15] by Anshelevich et al. is general and does not admit pure Nash equilibria, even for very simple network instances. Briefly, in such model each player i has a set of terminal nodes that he must connect. A strategy of a player is a payment function p_i, where $p_i(e)$ is how much player i is offering to contribute to the cost of edge e. Any edge e such that $\sum_i p_i(e) \geq c(e)$ is considered $bought$ ($c(e)$ being the link cost), and G_p denotes the graph of bought edges with the players offering payments $p = (p_1, ..., p_k)$. Since each player must connect its terminals, all of the player's terminals must be connected in G_p. However, each player i tries to minimize its total payments, $\sum_{e \in E} p_i(e)$.

The authors in [16] have extended the network design model in [14], including weighted players: if w_i denotes the weight of player i, then i's cost share of an edge e is $c_e \cdot w_i / W_e$, where W_e is the total weight of the players that use a path containing the edge e. But while easy to define, this weighted network design game is challenging to analyze. In particular, it is shown in [16] that:

- Pure-strategy Nash equilibria exist in all weighted Shapley network design games with two players.
- There are no larger classes of weighted Shapley network design games that always possess pure-strategy Nash equilibria. For example, it is illustrated a 3-player game that does not possess a Nash equilibrium.

The works in [17, 18] study the existence of strong Nash equilibria (i.e., equilibria where no coalition can improve the cost of each of its members) in network design

games under different cost sharing mechanisms. Strong Nash equilibria ensure stability against deviations by every conceivable coalition of agents. More specifically, the authors in [17] show that there are graphs that do not admit strong Nash equilibria, and then give sufficient conditions on the existence of approximate strong Nash equilibria.

Furthermore, the problem of designing a protocol that optimizes the equilibrium behavior of the induced network game is investigated in [19]. The authors study the design of optimal cost-sharing protocols for undirected and directed graphs, single-sink and multicommodity networks, different classes of cost-sharing methods, and different measures of the inefficiency of equilibria. Moreover, they provide upper and lower bounds on the best possible performance of non-uniform cost-sharing protocols.

Few works have considered individual and social concerns of user agents while dealing with different types of networking problems [36, 37].

A proposition that takes into account individual and social benefits has been considered in [36] in the general context of multi-agent systems, where individual and social concerns can conflict, leading to inefficient system performance. To address such problem, the authors have proposed a formal decision making framework, based on social welfare functions, that combines both social and individual perspectives.

An experimental investigation of the impact of cooperation in the context of routing games is conducted in [37]. The game is studied considering particular network topologies (i.e., parallel links and load balancing networks), shared by several users. Each user seeks to optimize either its own performance or some combination between its own performance and that of other users, by controlling the routing of its given flow demand.

However, unlike our work, none of the above papers applies these concepts to the network design game, nor provides a theoretical analysis of the efficiency of the achieved Nash equilibria.

Chapter 3

Service Overlay Networks Design: Models and Algorithms

Service Overlay Networks (SONs) create a virtual topology on top of the Internet, thus providing a valuable platform able to guarantee end-to-end quality of service without requiring support by the underlying network.

The optimization of the resources utilized by a SON is a fundamental issue for an overlay operator owing to the costs involved and the need to satisfy user requirements. Careful decisions are necessary to provide enough capacity to overlay links, to route traffic, to assign users to access nodes and to deploy overlay nodes.

This Chapter takes into account all these issues, proposing two mathematical programming models for the user assignment problem, the traffic routing optimization and the dimensioning of the capacity reserved on overlay links in SONs. The first model minimizes the SON installation cost while providing full access to all users. The second model maximizes the SON profit by selecting which users to serve, based on the expected gain, and taking into consideration budget constraints of the SON operator. Furthermore, we extend these models to include the optimization of the number and position of overlay nodes. Finally, we introduce two efficient heuristics to get near-optimal solutions for large-scale instances in a reasonable computation time.

The Chapter is structured as follows: Section 3.1 describes the proposed user assignment and traffic routing models, while Section 3.2 introduces the SON design formulations. Finally, Section 3.3 illustrates the proposed heuristics to plan large-scale overlay networks.

3.1 User Assignment and Routing Models in Service Overlay Networks

Table 3.1: Basic Notation for the SON design problem.

I	Set of Test Points (TPs)
D	Set of Destination Nodes (DNs)
R	Set of Overlay Nodes installed in the SON
c^B_{jl}	Cost for buying one bandwidth unit between CSs j and l
c^A_{ij}	Access cost per bandwidth unit between TP i and CS j
c^E_{jk}	Egress cost per bandwidth unit between CS j and DN k
d_{ik}	Traffic generated by TP i towards DN k
u_{jl}	Maximum capacity that can be reserved on overlay link (j, l)
v_j	Maximum capacity of the access link of CS j
h_{jk}	Maximum capacity that can be reserved on egress link (j, k)
a_{ij}	0-1 parameter that indicates if TP i can access the SON through CS j
e_{jk}	0-1 parameter that indicates if CS j can be connected to DN k
b_{jl}	0-1 parameter that indicates if CSs j and l can be connected with an overlay link
g_i	Revenue per bandwidth unit obtained for serving TP i
B	SON operator's budget
S	Set of Candidate Sites (CSs)
c^I_j	Cost for installing an overlay node in CS j
x_{ij}	0-1 variable that indicates if TP i is assigned to CS j
w_{jk}	0-1 variable that indicates if CS j is connected to DN k
y_{jl}	0-1 variable that indicates if there is an overlay link between nodes j and k
f^k_{jl}	Flow variable which denotes the traffic flow routed on link (j, l) destined to DN k
f_{jk}	Flow variable which denotes the traffic flow routed on egress link (j, k)
z_j	0-1 variable that indicates if an overlay node is installed in CS j

A common approach to the user assignment and routing problem is to consider feasible positions of traffic concentration points in the service area (Test Points, TPs), which generate traffic towards one or more Destination Nodes (DNs) [22]; the placement of TPs and DNs depends on the expected traffic distribution. Although the concept of *test point* is distinguished from *end-user* (formally, the end-user is the traffic generation agent that is placed in a TP), we will use the two terms as synonyms throughout the thesis. *Destination nodes* can represent either terminal nodes or access points to other networks.

Let $I = 1, \ldots, n$ denote the set of TPs, $D = 1, \ldots, p$ the set of destinations and $R = 1, \ldots, r$ the set of overlay nodes installed in the SON. The basic notation used in this Chapter is presented in Table 3.1.

The cost for the SON operator to buy one bandwidth unit between overlay nodes j and l from the underlying ISPs is denoted by c^B_{jl}, while c^A_{ij} is the access cost per bandwidth unit required between TP i and node j. Finally, c^E_{jk} represents the cost per bandwidth unit for the traffic transmitted on the egress link between node j and destination node $k \in D$.

The traffic generated by TP i towards destination node k is given by the parameter d_{ik}, $i \in I, k \in D$. The maximum capacity that can be reserved by the SON operator between nodes j and l on the overlay link (j, l) is denoted by u_{jl}, $j, l \in R$, while the maximum capacity of the access link of node j is denoted by v_j, $j \in R$.

According to TPs, DNs and overlay nodes' geographic location and the underlying physical topology, the following connectivity parameters can be calculated.

Let a_{ij}, $i \in I, j \in R$ be the test point coverage parameters:

$$a_{ij} = \begin{cases} 1 & \text{if TP } i \text{ can access the SON through} \\ & \text{overlay node } j \\ 0 & \text{otherwise} \end{cases}$$

Similarly, let e_{jk}, $j \in R, k \in D$ denote destination nodes coverage parameters:

$$e_{jk} = \begin{cases} 1 & \text{if overlay node } j \text{ can be connected} \\ & \text{with destination node } k \\ 0 & \text{otherwise} \end{cases}$$

Obviously, a_{ij} depends on the proximity of TP i to node j, that is on the access coverage provided by the SON operator with node j through agreements with local network operators. Similarly, e_{jk} is related to the distance between DN k and node j.

Let b_{jl}, $j, l \in R$ denote the connectivity parameters between two different overlay nodes, which may depend on the proximity of the overlay nodes j and l in the underlay network, as well as on the agreements between the SON and the different ISPs.

$$b_{jl} = \begin{cases} 1 & \text{if nodes } j \text{ and } l \text{ can be connected} \\ & \text{with an overlay link} \\ 0 & \text{otherwise} \end{cases}$$

Decision variables of the problem include TP assignment variables x_{ij}, $i \in I$, $j \in R$:

$$x_{ij} = \begin{cases} 1 & \text{if TP } i \text{ is assigned to overlay node } j \\ 0 & \text{otherwise} \end{cases}$$

destination assignment variables w_{jk}, $j \in R, k \in D$:

$$w_{jk} = \begin{cases} 1 & \text{if node } j \text{ is connected to destination node } k \\ 0 & \text{otherwise} \end{cases}$$

connection variables y_{jl}, $j, l \in R$:

$$y_{jl} = \begin{cases} 1 & \text{if there is an overlay link between nodes } j \text{ and } l \\ 0 & \text{otherwise} \end{cases}$$

and finally flow variables f_{jl}^k which denote the traffic flow routed on link (j, l) destined for destination node $k \in D$. The special variables f_{jk} denote the traffic flow on the egress link between node j and destination node k.

Given the above parameters and variables, we propose two different user assignment and routing formulations. The first, called Full Coverage User Assignment and Routing model (FC-UAR), optimizes the user assignment and traffic routing minimizing the total network cost while ensuring full coverage of all end-users. The second formulation, called Profit Maximization User Assignment and Routing model (PM-UAR), maximizes the total network profit, choosing which users to serve based on the revenue generated by their subscription to the SON services and the cost necessary to cover them.

3.1.1 Full Coverage User Assignment and Routing model

The Full Coverage User Assignment and Routing model (FC-UAR) optimizes the users' assignment and traffic routing minimizing the total network cost while ensuring full coverage of all end-users.

$$Minimize \quad \left\{ \sum_{j,l \in R} \sum_{k \in D} c_{jl}^B f_{jl}^k + \right.$$
$$\left. + \sum_{i \in I, j \in R, k \in D} c_{ij}^A d_{ik} x_{ij} + \sum_{j \in R, k \in D} c_{jk}^E f_{jk} \right\} \tag{3.1}$$

$$s.t.$$

$$\sum_{j \in R} x_{ij} = 1, \quad \forall i \in I \tag{3.2}$$

$$x_{ij} \leq a_{ij}, \quad \forall i \in I, j \in R \tag{3.3}$$

$$\sum_{i \in I} d_{ik} x_{ij} + \sum_{l \in R} (f_{lj}^k - f_{jl}^k) - f_{jk} = 0, \quad \forall j \in R, k \in D \tag{3.4}$$

$$\sum_{k \in D} f_{jl}^k \leq u_{jl} y_{jl}, \quad \forall j, l \in R \tag{3.5}$$

$$\sum_{i \in I, k \in D} d_{ik} x_{ij} \leq v_j, \quad \forall j \in R \tag{3.6}$$

$$f_{jk} \leq h_{jk} w_{jk}, \quad \forall j \in R, k \in D \tag{3.7}$$

$$y_{jl} \leq b_{jl}, \forall j, l \in R \tag{3.8}$$

$$w_{jk} \leq e_{jk}, \quad \forall j \in R, k \in D \tag{3.9}$$

$$x_{ij}, w_{jk}, y_{jl} \in \{0, 1\}, \quad \forall i \in I, j, l \in R, k \in D \tag{3.10}$$

The objective function (3.1) accounts for the Service Overlay Network cost, including the costs related to the connection of overlay nodes, users' access and egress costs.

Constraints (3.2) provide full coverage of all TPs, while constraints (3.3) are coherence constraints ensuring that TP i can be assigned to overlay node j only if i can be connected to j.

Constraints (3.4) define the flow balance in node j for all the traffic destined for node k. These constraints are the same as those adopted for classical multicommodity flow problems. The term $\sum_{i \in I} d_{ik} x_{ij}$ is the total traffic generated by the assigned TPs destined for destination node k, $\sum_{l \in R} f_{lj}^k$ is the total traffic received by j from neighboring nodes, $\sum_{l \in R} f_{jl}^k$ is the total traffic transmitted by j to neighboring nodes, and f_{jk} is the traffic transmitted towards the destination node k.

Constraints (3.5) impose that the total flow on the link between overlay nodes j and l does not exceed the capacity of the link itself (u_{jl}). Constraints (3.6) impose for each overlay node that the ingress traffic serviced by such network device does not exceed the capacity of the link used for the access, whilst constraints (3.7) force

the flow between node j and the destination node k to zero if node j is not connected to k, and impose that such flow does not exceed the maximum capacity (h_{jk}) of the egress link between overlay node j and destination node k.

Constraints (3.8) define the existence of an overlay link between nodes j and l, depending on the connectivity parameters b_{jl}. Constraints (3.9) are coherence constraints ensuring that a node j can be connected to a destination node k only if k is located in the proximity of node j. Finally, constraints (3.10) are the integrality constraints for the binary decision variables.

It is easy to see that this problem is equivalent to the integer multi-commodity flow problem and therefore is NP-hard [38]. We show in Chapter 4, however, that this problem can be solved to the optimum even for large-size instances with a short computing time.

Note that we can consider alternative formulations to the FC-UAR model. For example, we might want end-users to be connected to more than one overlay node, for redundancy. This is easily accomplished by modifying constraints (3.2) as:

$$\sum_{j \in R} x_{ij} = \eta, \quad \forall i \in I \tag{3.11}$$

where η is the number of overlay nodes per end-user.

3.1.2 Profit Maximization User Assignment and Routing Model

The Profit Maximization User Assignment and Routing model (PM-UAR) maximizes the SON operator's profit, choosing which users to serve based on the revenue generated by their subscription to the SON services and the required cost to the SON provider for covering them.

The objective function (3.1) is therefore changed as follows:

$$\begin{aligned} Maximize \sum_{i \in I, j \in R, k \in D} g_i d_{ik} x_{ij} - \Big\{ \sum_{j,l \in R} \sum_{k \in D} c^B_{jl} f^k_{jl} \\ + \sum_{i \in I, j \in R, k \in D} c^A_{ij} d_{ik} x_{ij} + \sum_{j \in R, k \in D} c^E_{jk} f_{jk} \Big\} \end{aligned} \tag{3.12}$$

where $g_i, \forall i \in I$, represents the revenue per bandwidth unit that the SON operator obtains covering Test Point i. Here we assume for simplicity that the price paid by the $i-th$ user is proportional to the amount of traffic the user introduces in the SON, $\sum_{k \in D} d_{ik}$, with g_i being the proportionality coefficient, but some general pricing models can be easily accounted for.

Constraints (3.2) are changed as follows, while all the other constraints are the same as in the FC-UAR model:

$$\sum_{j \in R} x_{ij} \leq 1, \forall i \in I \tag{3.13}$$

With such formulation, the SON operator maximizes the network profit, obtained by subtracting the total revenue, achieved by covering a subset of the Test Points, to the cost necessary to deploy an overlay network satisfying the users' requirements. Note that, differently from constraints (3.2) in the FC-UAR model, in this formulation constraints (3.13) do not impose full coverage of all TPs.

The Service Overlay Network designer may be required to stay within a given cost budget (B) to limit the economic risks in the deployment of the network. The PM-UAR formulation can be modified to account for budget limitation simply by the addition of the following constraint:

$$\sum_{j,l \in R} \sum_{k \in D} c^B_{jl} f^k_{jl} + \sum_{i \in I, j \in R, k \in D} c^A_{ij} d_{ik} x_{ij} + \sum_{j \in R, k \in D} c^E_{jk} f_{jk} \leq B \tag{3.14}$$

3.2 Service Overlay Network Design Models

In this Section, we extend the models presented in Section 3.1, proposing two novel Service Overlay Network design formulations, namely Full Coverage SON Design and Profit Maximization SON Design models, which also optimize the number and location of overlay nodes to be deployed.

To this end, in addition to Test Points and Destination Nodes, we consider feasible positions, called Candidate Sites (CSs), where overlay nodes can be installed [22]. The placement of CSs depends on the underlying network topology and the agreements of the SON operator with ISPs. Let $S = 1, \ldots, m$ denote the set of CSs, and c^I_j the cost associated with installing an overlay node at CS j.

Decision variables now include overlay node installation variables z_j, $j \in S$:

$$z_j = \begin{cases} 1 & \text{if an overlay node is installed in CS } j \\ 0 & \text{otherwise} \end{cases}$$

All the other variables and parameters are the same as defined in Section 3.1, where the overlay nodes set R is now replaced by the set of Candidate Sites, S.

3.2.1 Full Coverage SON Design Model

The Full Coverage SON Design model (FCSD), whose formulation is reported below, optimizes the number and location of overlay nodes minimizing at the same time the total network cost and ensuring full coverage of all SON users.

The objective function can be obtained from (3.1) including the overlay nodes installation costs:

$$Minimize \quad \left\{ \sum_{j \in S} c_j^I z_j + \sum_{j,l \in S} \sum_{k \in D} c_{jl}^B f_{jl}^k + \right.$$
$$\left. + \sum_{i \in I, j \in S, k \in D} c_{ij}^A d_{ik} x_{ij} + \sum_{j \in S, k \in D} c_{jk}^E f_{jk} \right\} \tag{3.15}$$

The problem variables are subject to constraints (3.2), (3.4)-(3.8) and to the following constraints.

TP i can be assigned to CS j only if an overlay node is installed in j and if i can be connected to j:

$$x_{ij} \leq z_j a_{ij}, \quad \forall i \in I, j \in S \tag{3.16}$$

The existence of an overlay link between CSs j and l depends on the installation of nodes in j and l, and is defined by:

$$y_{jl} \leq z_j, y_{jl} \leq z_l, \quad \forall j, l \in S \tag{3.17}$$

Coherence constraints ensure that a CS j can be connected to a destination node k only if an overlay node is installed in j and if k can be connected to j:

$$w_{jk} \leq e_{jk} z_j, \quad \forall j \in S, k \in D \tag{3.18}$$

Finally, the integrality constraints for the binary decision variables are:

$$x_{ij}, z_j, w_{jk}, y_{jl} \in \{0, 1\}, \quad \forall i \in I, j, l \in S, k \in D \tag{3.19}$$

The FCSD problem is therefore defined by the objective function (3.15) subject to constraints (3.2), (3.4)-(3.8) and (3.16)-(3.19).

3.2.2 Profit Maximization SON Design Model

The Profit Maximization SON Design model (PMSD) maximizes the SON operator's profit, selecting the optimal number and location of overlay nodes to be deployed, and choosing which users to serve based on the expected gain and the cost necessary to satisfy their traffic demands.

The PMSD formulation is obtained by modifying the objective function (3.15) as follows:

$$Maximize \sum_{i\in I, j\in S, k\in D} g_i d_{ik} x_{ij} - \Big\{ \sum_{j\in S} c_j^I z_j + \\ + \sum_{j,l\in S} \sum_{k\in D} c_{jl}^B f_{jl}^k + \sum_{i\in I, j\in S, k\in D} c_{ij}^A d_{ik} x_{ij} + \sum_{j\in S, k\in D} c_{jk}^E f_{jk} \Big\} \tag{3.20}$$

and by introducing constraints (3.21), as for the PM-UAR model:

$$\sum_{j\in S} x_{ij} \leq 1, \quad \forall i \in I \tag{3.21}$$

All other constraints are the same as in the FCSD model.

Finally, a cost budget can be introduced in PMSD simply by adding the following constraint:

$$\sum_{j\in S} c_j^I z_j + \sum_{j,l\in S} \sum_{k\in D} c_{jl}^B f_{jl}^k + \\ + \sum_{i\in I, j\in S, k\in D} c_{ij}^A d_{ik} x_{ij} + \sum_{j\in S, k\in D} c_{jk}^E f_{jk} \leq B \tag{3.22}$$

3.3 Heuristics to Solve the SON Design Problems - Continuous relaxation and randomized rounding based heuristics

The computing time required to obtain an optimal solution for both the FCSD and PMSD problems might be very long for medium-to-large-size instances. As an example, the average computing time required to solve at optimum random network instances with the FCSD model grows exponentially from 10 s for CS=30 up to 4600 s for CS=50, when run on an Intel Pentium 4 (TM) processor with CPUs operating at

3 GHz and with 1024 Mbyte of RAM, which is the workstation used to obtain the numerical results reported in the next Chapter. For this reason, we propose hereafter two simple but effective heuristics, named H-FCSD and H-PMSD, that provide near-optimal solutions in a reasonable amount of time for the Full Coverage and the Profit Maximization SON Design problems, respectively.

3.3.1 H-FCSD: a Heuristic to solve the Full Coverage SON Design Problem

In the case of the full coverage problem, we apply a problem reduction technique which sets some variables solving first the access coverage and distribution sub-problems, then considering a continuous relaxation of the FCSD formulation, and a randomized rounding procedure to obtain an integer solution.

The H-FCSD heuristic is composed of three steps, which are described in detail in the following and are also illustrated in the flow diagram of Figure 3.1.

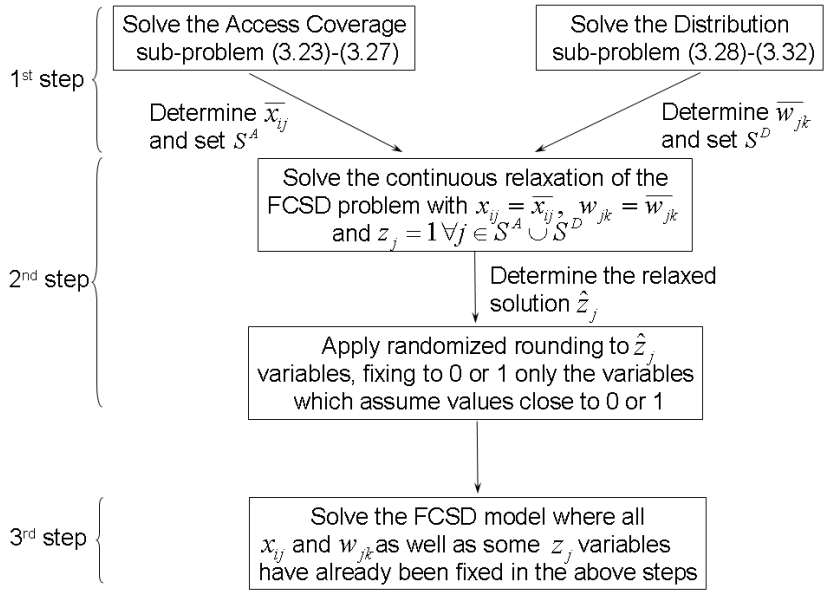


Figure 3.1: Flow diagram of the H-FCSD heuristic.

First Step

In this step we solve separately the access coverage and the distribution sub-problems, whose formulations are reported below. All the variables and parameters have the same definition as described in the previous Section for the FCSD model.

The minimum-cost access coverage sub-problem consists in computing the optimal location of the overlay nodes through which end-users can access the SON with minimum cost, and it is formulated as follows:

$$Minimize \sum_{i\in I, j\in S, k\in D} c^A_{ij} d_{ik} x_{ij} + \sum_{j\in S} c^I_j z_j \tag{3.23}$$

$$s.t.$$

$$\sum_{j\in S} x_{ij} = 1, \quad \forall i \in I \tag{3.24}$$

$$x_{ij} \leq z_j a_{ij}, \quad \forall i \in I, j \in S \tag{3.25}$$

$$\sum_{i\in I, k\in D} d_{ik} x_{ij} \leq v_j, \quad \forall j \in S \tag{3.26}$$

$$x_{ij}, z_j, \in \{0, 1\}, \quad \forall i \in I, j \in S \tag{3.27}$$

The objective function (3.23) accounts for the access cost, composed of the costs related to the users' access and the installation cost of access nodes. Constraints (3.24), (3.25) and (3.26) are the same as constraints (3.2), (3.16) and (3.6) in the FCSD problem formulation, while (3.27) are the integrality constraints for the decision variables.

The minimum-cost distribution sub-problem consists in determining the optimal location of the overlay nodes that can deliver with minimum cost the SON traffic to destination nodes, and it is formulated as follows:

$$Minimize \sum_{j\in S, k\in D} c^E_{jk} f_{jk} + \sum_{j\in S} c^I_j z_j \tag{3.28}$$

$$s.t.$$

$$f_{jk} \leq h_{jk} w_{jk}, \quad \forall j \in S, k \in D \tag{3.29}$$

$$w_{jk} \leq e_{jk} z_j, \quad \forall j \in S, k \in D \tag{3.30}$$

$$\sum_{j \in S} f_{jk} = \sum_{i \in I} d_{ik}, \quad \forall k \in D \tag{3.31}$$

$$z_j, w_{jk} \in \{0, 1\}, \quad \forall j \in S, k \in D \tag{3.32}$$

The objective function (3.28) calculates the egress cost, including the costs related to distribute egress traffic and the installation cost of egress overlay nodes. Constraints (3.29) and (3.30) are the same as constraints (3.7) and (3.18). Constraints (3.31) are coherence constraints which impose that all the traffic destined to destination node k ($\sum_{i \in I} d_{ik}$) is effectively routed in the egress assignment sub-problem. Finally, (3.32) are the integrality constraints for the decision variables.

Solving the access coverage problem (3.23)-(3.27) we determine the optimal assignment of each TP to a corresponding CS, $\overline{x_{ij}}$; at the same time, we select the subset $S^A \subseteq S$ of CSs where overlay nodes must be installed to cover all TPs (i.e. $S^A = \{j \in S | z_j = 1\}$).

In the same way, solving the distribution problem (3.28)-(3.32) we determine both the optimal egress connections between CSs and DNs, $\overline{w_{jk}}$, and the subset $S^D \subseteq S$ of CSs where overlay nodes must be installed to guarantee a connection with the egress nodes (i.e. $S^D = \{j \in S | z_j = 1\}$).

Let $S^F = S^A \cup S^D$ be the subset of CSs where an overlay node must be installed as determined either in the access coverage or distribution sub-problems.

Second Step

In this step we solve a continuous relaxation of the FCSD problem (3.2), (3.4)-(3.8) and (3.15)-(3.19), i.e. the FCSD problem (3.2), (3.4)-(3.8) and (3.15)-(3.18) with the integrality constraints (3.19) replaced as follows:

$$x_{ij}, z_j, w_{jk} \in [0, 1], \quad \forall i \in I, j \in S, k \in D \tag{3.33}$$

and where, in addition, the x_{ij}, w_{jk} and $z_j | j \in S^F$ variables are constrained to assume the values determined in the first step. To this aim, the following constraints are added:

$$x_{ij} = \overline{x_{ij}}, \quad \forall i \in I, j \in S \tag{3.34}$$

$$w_{jk} = \overline{w_{jk}}, \quad \forall j \in S, k \in D \tag{3.35}$$

$$z_j = 1, \quad \forall j \in S^F \tag{3.36}$$

Let $\hat{z}_j$ be the optimal solution of the relaxed problem (3.2), (3.4)-(3.8), (3.15)-(3.18), and (3.33)-(3.36). We then apply randomized rounding on $\hat{z}_j$:

- if $\widehat{z_j} = 0$, then the corresponding z_j variable is set to 0;
- otherwise, the variable z_j is set to 1 with probability $P[z_j = 1] = \widehat{z_j}$; to this end, we extract a random value v uniformly distributed in $[0, 1]$: if $v \leq \widehat{z_j}$ the z_j variable is set to 1 (that is, an overlay node is installed in CS j). If $v > \widehat{z_j}$, no decision is taken in this step on the installation of an overlay node in CS j, and the z_j value will be determined in the third step of the heuristic.

The rational behind such procedure is that we strive for a balance between reducing the problem complexity (i.e. the number of binary variables in the integer problem solved in the third step) and the possibility of obtaining a feasible and close to the optimum solution.

Third Step

Finally, in this step we solve the original integer FCSD problem (3.2), (3.4)-(3.8) and (3.15)-(3.19), with all the x_{ij}, w_{jk} and z_j variables set as described in the first two steps, obtaining both integer solutions for the remaining z_j variables and optimal values for the routing variables f_{ij}^k.

Comments

Randomized rounding is a general technique first proposed in [39] to solve 0-1 optimization problems, which consists in solving the continuous relaxation of the integer problem and then transforming the optimal solution of the relaxed problem into a feasible solution for the integer problem. It has been demonstrated in [39] that such technique provides provably good solutions, in the sense that with high probability this algorithm provides an integer solution in which the objective function assumes a value close to the optimum of the continuous relaxation.

However, it may be difficult to obtain a good integer solution from the fractional one. For this reason, to design an efficient heuristic, we introduced the first two steps described above since we observed that applying directly randomized rounding to the

fractional variables of the relaxed FCSD problem leads very often to unfeasible solutions where several constraints are violated; then, the computation of a feasible and near to the optimum solution is not very efficient even applying scaling techniques as proposed in [39].

Note that in the first step several optimal solutions may exist to the access coverage and distribution sub-problems, and it is therefore interesting to extend the H-FCSD algorithm considering different initial solutions. This can also increase the probability that a feasible solution is obtained in the third step. Obviously, there exists a trade-off between the execution time of the heuristic (which grows with the number of different initial solutions considered) and the improvement in the total network cost. To evaluate this issue we performed several tests, considering up to 15 initial solutions, and we found that in all the considered network scenarios the maximum improvement obtained in the cost of the planned SON was less than 2%. For this reason, in Chapter 4 we report numerical results obtained considering only one optimal solution in the first step of the H-FCSD heuristic.

3.3.2 H-PMSD: a Heuristic to solve the Profit Maximization SON Design Problem

In general, determining the optimal subset of end-users to cover in order to maximize the SON operator's profit is a more difficult problem than designing the minimum cost overlay network that provides full coverage to a given set of users.

The H-PMSD heuristic tackles such difficulty determining preliminarily which users to serve, and then designing the minimum cost SON that covers such users. H-PMSD is therefore composed of two steps, as illustrated in the flow diagram of Figure 3.2 and detailed in the following.

First Step: determining which users to cover

In this step, a continuous relaxation of the PMSD problem is solved. Let $\hat{x}_{ij}$ be the optimal solution for the TPs assignment variables. Then, for each TP i we consider the quantity $Q_i = \sum_{j \in S} \hat{x}_{ij}$, which assumes values in the $[0, 1]$ interval and can be interpreted intuitively as the probability with which the ith Test Point should be covered by the SON. We then perform randomized rounding on Q_i: a random value v uniformly distributed in $[0, 1]$ is extracted; if $v \leq Q_i$, then TP i is selected to be covered by the SON; otherwise, TP i is not selected. Let $I^c \subseteq I$ be the subset of TPs chosen in this step to be covered by the SON.

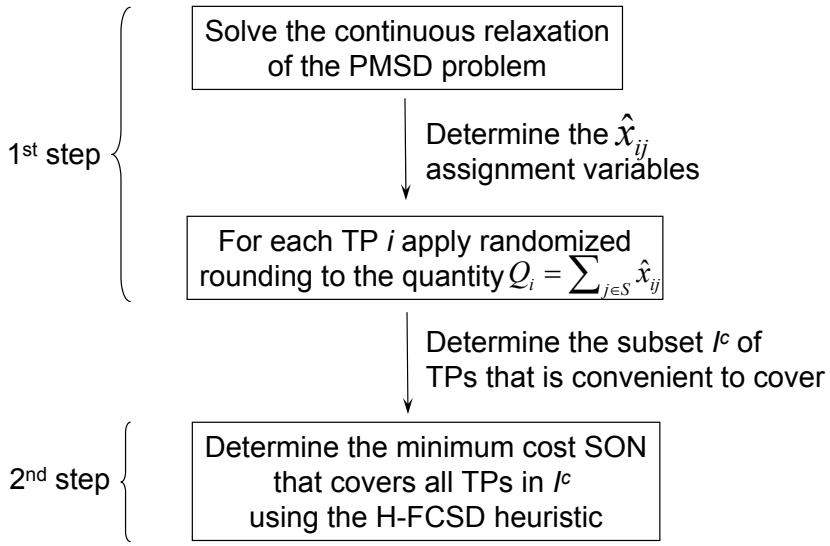


Figure 3.2: Flow diagram of the H-PMSD heuristic.

Second Step: designing the minimum cost SON

In this step we use the H-FCSD heuristic to design the minimum cost SON that covers all the TPs $\in I^c$ chosen in the previous step.

Note that the heuristic may be unable to find any feasible solution of the problem since the cost for covering the selected TPs is higher than the revenue.

H-PMSD with Budget constraints

A cost budget constraint can be taken into account in the first step of the H-PMSD heuristic, introducing constraint (3.22) in the continuous relaxation of the PMSD problem. With B the budget, if the cost of the SON planned in the second step is not greater than B, then the computed solution is acceptable. Otherwise, to obtain a feasible solution we apply a scaling technique [39] in the randomized rounding procedure of the first step. Such technique consists in multiplying the solution of the relaxed problem by a factor $\gamma < 1$, which corresponds to using γQ_i in the first step of H-PMSD. If γ decreases, the probability that user i is covered by the SON, and as a consequence the total network cost, is reduced until the budget constraint is not violated.

In our work we perform a simple iterative procedure that proceeds as follows:

1. Initialize $\gamma = 1$
2. Solve the first step of H-PMSD using γQ_i
3. Solve the second step of H-PMSD
4. If the cost of the SON planned in the second step of H-PMSD is $\leq B$ then STOP (a feasible solution has been obtained). Otherwise reduce the γ value by 0.1 and go to 2)

However, a finer tuning of the γ parameter could be performed, using for example a binary search technique, to find the γ value which guarantees at the same time feasibility and a good quality of the solution.

Chapter 4

Service Overlay Networks Design: Performance Evaluation

In this Chapter we evaluate the performance of the models and heuristics illustrated in the previous Chapter using a set of realistic and large-size instances, and discuss the effect of different parameters on the characteristics of the planned networks.

More specifically, we test the sensitivity of our models and heuristics to different parameters like the number of candidate sites and test points, the traffic demands, the installation costs as well as the revenue obtained by covering end-users and the SON operator's budget. We compare the performance of the exact and heuristic approaches in terms of the obtained results and computing time. We also provide bounds to the performance achievable by any optimization algorithm solving continuous relaxations of the integer models.

To this end we consider both randomly generated network instances and real ISP topologies mapped by the Rocketfuel tool [40, 41]. Random network topologies are obtained using a custom generator as well as hierarchical (Transit-Stub) models generated by the GT-ITM topology generator [42, 43], and finally using a degree-based generator (BRITE [44, 45]) to obtain topologies with node degree power laws.

This Chapter is structured as follows: Section 4.1 illustrates the parameters settings as well as the random topology generators used in our numerical results. In Section 4.2 we discuss the numerical results obtained by the user assignment and traffic routing models. Finally, in Section 4.3 we analyze the results of the SON design models and heuristics, providing a thorough comparison with the user assignment and routing models.

4.1 Random Topology Generators and Parameters Setting

To generate random network instances, we have implemented a topology generator which considers a square area with edge equal to 1000, and randomly extracts the position of m Candidate Sites (CSs), n Test Points (TPs) and p Destination Nodes (DNs). The area is divided into N Internet Service Providers (ISPs); for sake of simplicity in this Chapter we consider $N = 25$ ISPs obtained dividing the whole area into $L \times L$ squares, with $L = 200$. The same procedure is used to generate network instances with r overlay nodes for the model where their position is given.

Unless stated otherwise, we assume that each TP and DN can be connected to a CS only if the CS is at a distance not greater than 100 from the TP or DN.

As for the connectivity parameters between different CSs, we assume that each CS can be directly connected with an overlay link to any other CS (i.e., $b_{jl} = 1, \forall j, l \in S$); this allows our models to investigate all possible link configurations to find the optimal overlay topology.

The cost matrix for bandwidth (c^B_{jl}) is then generated. If CSs j and l belong to the same ISP, we assume that c^B_{jl} is fixed and equal to 1 monetary unit per Mb/s. On the other hand, if CSs j and l belong to different ISPs, c^B_{jl} depends on the peering agreements between such ISPs. For the sake of simplicity, we assume that in this case c^B_{jl} is a random variable uniformly distributed between $C/2$ and $3C/2$, with C being equal to $\frac{L_{jl}}{L}$, that is the distance between j and l (L_{jl}) divided by the width of an ISP domain (L), i.e. 200 with the above settings.

If not specified differently, the installation cost of an overlay node is equal to 10 monetary units. As for the access and egress cost, we assume they are fixed and equal to 1 monetary unit per Mb/s.

The maximum capacity that can be reserved between CSs j and l on the overlay link (j, l) u_{jl}, $j, l \in S$ is set equal to 50 Mb/s, as well as the maximum capacity of the access link of CS j, v_j, $j \in S$. The capacity of the egress links connecting overlay nodes to destination nodes is $h_{jk} = 100$ Mb/s, for all $j \in S$ and $k \in D$.

Obviously, none of the above assumptions affects the proposed models and heuristics which are general and can be applied to any problem instance and network topology.

Identifying Candidate Sites can be a difficult task in real ISPs networks. To solve this problem, a topology-aware node placement heuristic could be used, as proposed in [9], to decide the potential locations for overlay nodes inside an ISP. Such techniques can be used together with our heuristics and models, thus representing a further research topic worth pursuing.

All the results reported hereafter are the optimal and approximate solutions of the considered instances obtained, respectively, by formalizing the proposed models in AMPL [46] (see Appendix A.3) and solving them with CPLEX [47], or using the proposed heuristics, on workstations equipped with an Intel Pentium 4 (TM) processor with CPUs operating at 3 GHz, and with 1024 Mbyte of RAM. For each network scenario, the results are obtained averaging each point on 10 network instances.

4.2 User Assignment and Routing Models

We first tackle the user assignment and routing problem, considering different network scenarios and varying several parameters such as the number of overlay nodes, the traffic demands, the gain the SON operator obtains serving end-users and the cost budget.

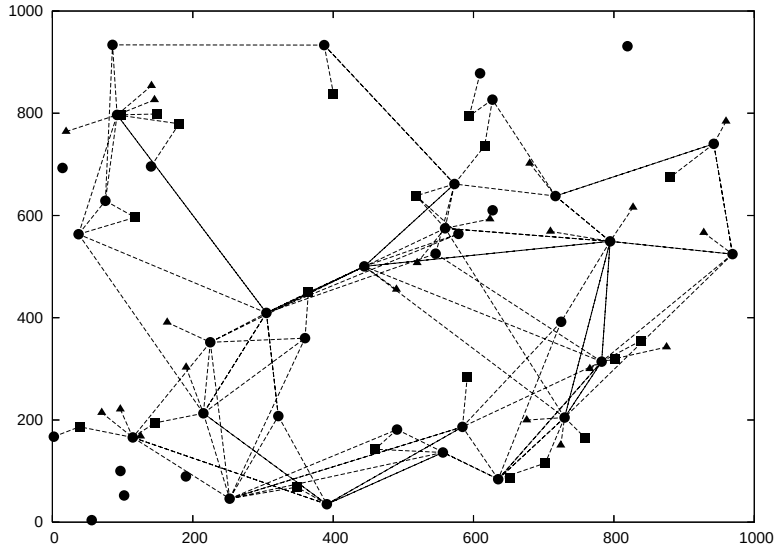
a) Effect of the Traffic Demands: Random network instances

We first consider the Full Coverage User Assignment and Routing model (FC-UAR) in a random network scenario with $n = 20$ TPs and $p = 20$ DNs. Each test point offers the same amount of traffic d_{ik} to all destination nodes.

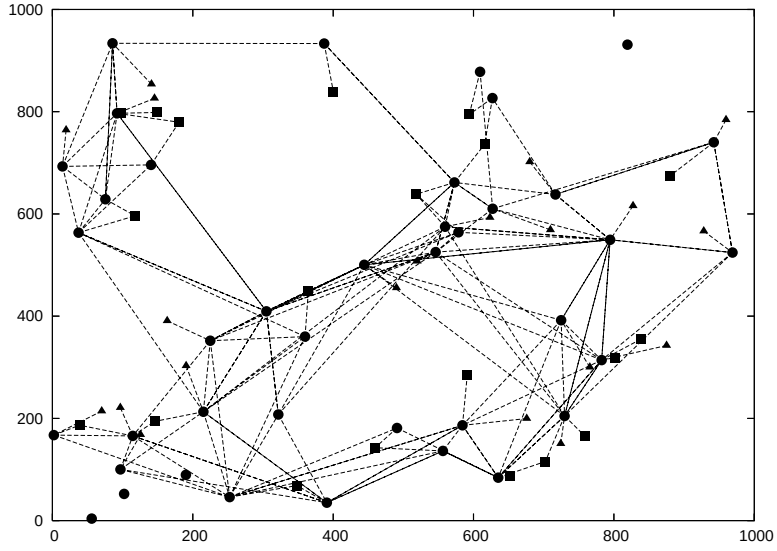
Figure 4.1 reports an example of the planned networks when applying the FC-UAR model to the same instance with $r = 40$ overlay nodes and with two different requirements on the end-user traffic, $d_{ik} = 500$ kb/s and $d_{ik} = 2$ Mb/s for all TPs and DNs. Overlay nodes, TPs and DNs are represented respectively by circles, triangles and squares. We observe that increasing the traffic demands forces the model to use a higher number of overlay nodes and to install more links to convey the traffic towards the destination nodes.

Table 4.1 analyzes the characteristics of the solutions in the same scenario when varying the number of overlay nodes r. For each couple (r, d_{ik}) we report the number of installed overlay links (N_L), the network cost (i.e. the value of the objective function (3.1), page 16) and the processing time to get the optimal solution.

Table 4.1 suggests three main comments. First, the very same effect of traffic increase observed in Figure 4.1 is evident also in averaged results. In fact, owing to capacity constraints, just increasing the link bandwidth is not sufficient and it is necessary to use more overlay nodes and links. Second, for a given traffic value, increasing the number of overlay nodes (r) in the FC-UAR model increases the solution space; as a consequence, the model favors the solutions providing connectivity at a lower cost, which in turn decreases with r.



(a) 500 kb/s



(b) 2 Mb/s

Figure 4.1: Sample SONs planned by the FC-UAR model with increasing traffic demands (500 kb/s and 2 Mb/s). The number of TPs and DNs is 20, while the number of overlay nodes is 40. Overlay nodes, TPs and DNs are represented respectively by circles, triangles and squares.

Table 4.1: Solutions provided by the FC-UAR model with 20 TPs and DNs.

	d_{ik}=500 kb/s			d_{ik}=1000 kb/s		
r	N_L	Cost	Time (s)	N_L	Cost	Time (s)
30	184.6	783.1	0.5	195.8	1568.9	0.6
40	217.3	765.6	1.2	231.4	1532.7	1.2
50	239.8	746.8	2.2	249.4	1494.6	2.3
100	305.7	698.0	23.6	320.4	1395.6	22.8
200	414.6	661.6	220.9	431.9	1323.5	223.4
300	452.3	643.3	718.3	470.8	1285.9	765.7

Finally, it can be noted that the FC-UAR model solves the user assignment and routing problem even for large-scale network instances with a short computing time.

We then simulated a scenario with a higher number of traffic flows, considering 100 TPs and 10 DNs, where DNs can be seen as acting like concentrator nodes or access points to other networks. The results obtained with the FC-UAR model are shown in Table 4.2 with r ranging from 30 to 300 and for different d_{ik} values, and they are in line with the observations reported above.

Table 4.2: Solutions provided by the FC-UAR model with 100 TPs and 10 DNs.

	d_{ik}=20 kb/s			d_{ik}=40 kb/s		
r	N_L	Cost	Time (s)	N_L	Cost	Time (s)
30	260.5	77.3	0.2	260.5	154.7	0.2
40	290.8	74.7	0.3	290.9	149.4	0.3
50	323.7	73.2	0.6	324.0	146.4	0.6
100	425.1	69.0	5.9	425.2	138.1	5.8
200	547.0	65.8	68.0	547.1	131.5	68.2
300	631.7	64.2	239.8	631.9	128.3	242.8

b) Effect of the Traffic Demands: Transit-Stub topologies

To investigate the behavior of the FC-UAR model with a large number of traffic flows, we generated large-scale Transit-Stub topologies using GT-ITM [42].

In such scenarios, the Internet is modeled as a collection of interconnected routing domains, which can be classified as either Transit domains (that contain backbone

nodes) or Stub domains (which have one or more gateway nodes that are connected to transit domains).

We considered 10 random Transit-Stub topologies with $r = 50, 100, 200$ overlay nodes and an average number of links equal to 400, 550 and 1200, respectively, including access and egress links; each link can be selected as an overlay link. For each topology we generated 10 random distributions of $n = 100$ TPs and $p = 100$ DNs, where each TP offers the same amount of traffic $d_{ik} = 10$ kb/s to all destination nodes. All other parameters are the same as in the previous network scenarios.

The numerical results obtained with the FC-UAR model, averaged over all network topologies and random TPs/DNs distributions, are shown in Table 4.3. We observe that owing to the hierarchical structure of Transit-Stub topologies, a large number of overlay links is selected in the planned SON; furthermore, the time necessary to compute the optimal solution is very short.

Table 4.3: Transit-Stub topologies: solutions provided by the FC-UAR model with 100 TPs, 100 DNs and d_{ik}=10 kb/s.

r	N_L	Cost	Time (s)
50	341.9	700.0	0.5
100	418.6	873.8	5.7
200	449.6	1040.4	10.5

c) Effect of the Gain parameter on Profit Maximization

We evaluate the effect of the gain parameter on the Profit Maximization User Assignment and Routing model (PM-UAR) considering a scenario with 50 TPs, 50 DNs and $r = 100$ overlay nodes. We assume that the gain per bandwidth unit that the SON operator obtains for serving an end-user (the parameter g_i in the objective function (3.12), page 18) is a random variable with average equal to G and a uniform distribution between $G/2$ and $3G/2$, with G ranging between 0 and 0.01 monetary units per Mb/s.

Figure 4.2 shows the number of end-users covered by the SON as a function of G. Obviously, for small G values, the SON is not profitable enough to cover any of the end-users; as G increases, the SON covers more end-users, and eventually all of them. Similar results have been observed with different values of r.

Table 4.4 reports, for the same scenario, the number of installed links, the SON operator's profit (i.e. the value of the objective function (3.12)), the network cost

and processing time, as a function of G. Note that when G increases, the planned network covers more end-users, and as a consequence it uses more overlay links.

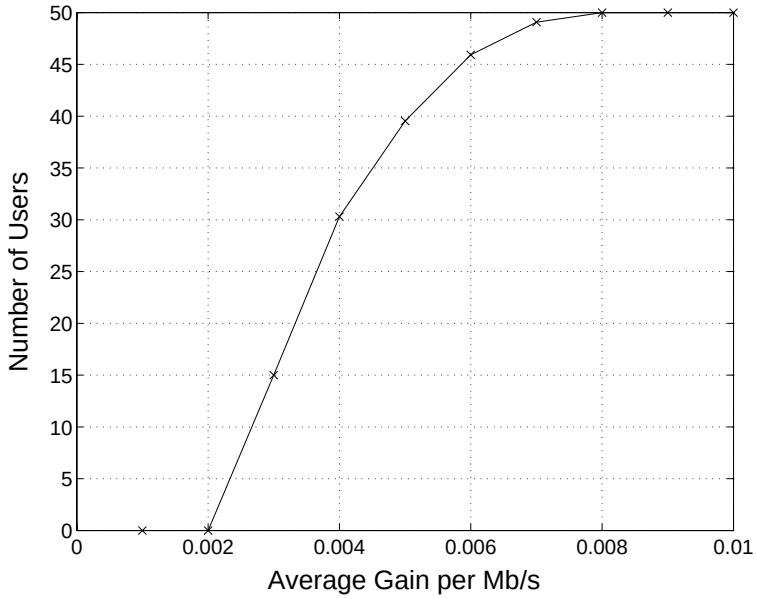


Figure 4.2: Number of end-users covered by the SON as a function of the average gain per bandwidth unit (PM-UAR model), with 50 TPs, 50 DNs, 100 overlay nodes and d_{ik}=100 kb/s.

Table 4.4: Solutions provided by the PM-UAR model with 50 TPs and DNs, 100 overlay nodes and d_{ik}=100 kb/s.

G	N_L	Profit	Cost	Time (s)
0.005	703.3	383.7	682.1	42.2
0.006	755.7	608.7	792.6	42.1
0.007	771.4	847.7	847.6	42.0
0.008	780.7	1091.6	865.2	41.9
0.009	780.8	1336.2	865.2	42.0
0.010	780.9	1580.8	865.2	41.8

d) Effect of the Budget parameter

Finally, to evaluate the effect that a budget constraint has on the planning of an SON, we consider several budget (B) values in the 500 to 1000 range, solving the PM-UAR model in a random network scenario with 20 TPs and DNs, 40 overlay nodes and d_{ik}=500 kb/s.

Figure 4.3 illustrates the number of end-users covered by the SON as a function of the operator's budget, for different G values. For each value of G, as the budget increases, the number of end-users accepted in the network increases until it reaches its maximum.

Table 4.5 reports in detail the characteristics of the solutions provided by the PM-UAR model in such a scenario, for $G = 0.01$ monetary units per Mb/s and for different budget values. The results show that deploying higher-cost networks allows the SON operator to achieve higher network profits. This, however, also increases the economic risk faced by the SON operator in the deployment of the overlay network.

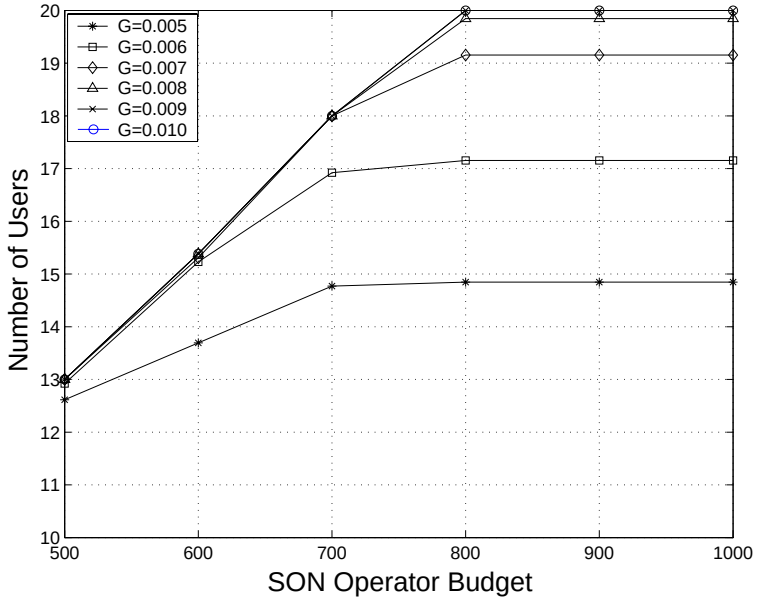


Figure 4.3: Number of end-users covered by the SON as a function of the budget for different values of the average gain per bandwidth unit G (PM-UAR model), with 20 TPs, 20 DNs and 40 overlay nodes.

Table 4.5: Solutions provided by the PM-UAR model with 20 TPs and DNs, 40 overlay nodes, G=0.01 monetary units per Mb/s and d_{ik}=500 kb/s.

B	$Users$	N_L	Profit	Cost	Time (s)
500	13.0	175.1	1015.3	492.8	40.2
600	15.4	192.7	1118.3	583.8	37.4
700	18.0	209.0	1203.2	686.9	33.7
800	20.0	217.4	1241.6	765.6	2.8
900	20.0	217.4	1241.6	765.6	2.7
1000	20.0	217.4	1241.6	765.6	2.7

4.3 Network Design Models and Heuristics

We now present the results of the SON design models, providing a thorough comparison with the continuous relaxation, the proposed heuristics as well as the user assignment and routing models. Also in this case we analyze the impact on the planned overlay networks of traffic demands, installation and bandwidth costs, gain and budget parameters.

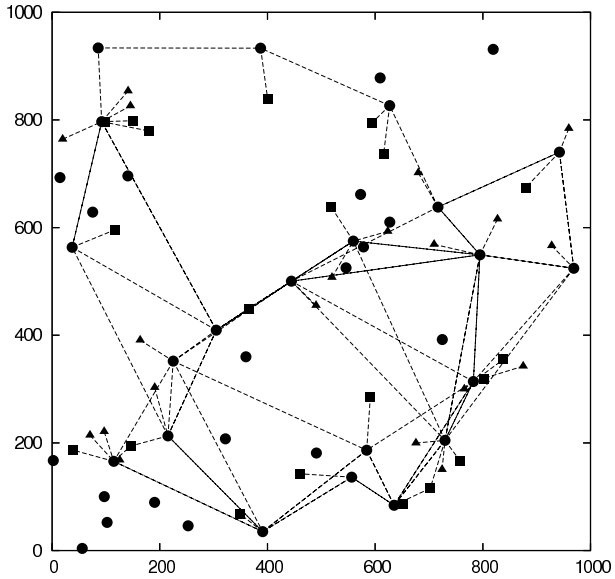
a) Effect of the Traffic Demands: Random network instances

To gauge the effect of the traffic demands on the Full Coverage SON Design (FCSD) model, let us consider a random network scenario with $n = 20$ TPs, $p = 20$ DNs and a variable number of Candidate Sites, m. Each test point offers the same amount of traffic d_{ik} to all destination nodes.

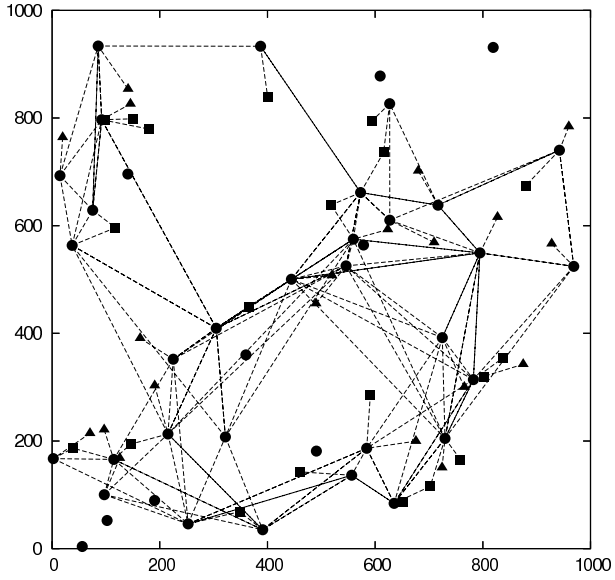
Figure 4.4 reports an example of the planned networks when applying the FCSD model to the same instance with $m = 40$ candidate sites and with two different requirements on the end-user traffic, $d_{ik} = 500$ kb/s and $d_{ik} = 1$ Mb/s for all TPs and DNs. CSs, TPs and DNs are represented with circles, triangles and squares, respectively. As expected, increasing the traffic demands forces the model to install a higher number of overlay nodes and links to convey the traffic towards the destination nodes.

Table 4.6 analyzes the characteristics of the solutions of the FCSD model, its continuous relaxation and the H-FCSD heuristic in the same scenario when varying the number of candidate sites.

For each couple (m, d_{ik}) and optimization algorithm, the Table reports the number of installed overlay nodes (N_R) and links (N_L), the total network cost and the computing time (measured in seconds) to obtain the solution.



(a) 500 kb/s



(b) 1000 kb/s

Figure 4.4: Sample SONs planned by the FCSD model with increasing traffic demands (500 and 1000 kb/s). The number of TPs and DNs is 20, while the number of CSs is 40. CSs, TPs and DNs are represented with circles, triangles and squares, respectively.

For small instances we could obtain the exact solutions with the FCSD model, which enabled a comparison between the H-FCSD heuristic and optimal integer solutions; column gap_I reports the percentage gap between the cost provided by H-FCSD and the optimal cost obtained with the FCSD model. This gap shows how close is H-FCSD to the optimum obtained by FCSD, in terms of the objective function value.

We also reported the results obtained solving the continuous relaxation of the FCSD model, which provides a lower bound on the network cost that can be obtained with any optimization algorithm; column gap_B shows the percentage gap between the cost provided by the H-FCSD heuristic and such lower bound. Note that such relaxation is used exclusively to provide a reference point, especially for large network instances where solutions using FCSD cannot be obtained.

Finally, column gap_L shows the percentage gap between the cost provided by the FCSD model and that obtained solving the continuous relaxation of FCSD. Thus, gap_L measures how far is the optimal integer solution from the relaxed one.

Three main results come from the observation of the Table: first, the very same effect of traffic increase observed in Figure 4.4 is evident also on averaged results; in fact, the number of installed nodes and links increases when increasing the traffic demands.

Second, for a given traffic value, increasing the number of CSs (m) increases the solution space; as a consequence, the model favors the solutions providing connectivity that have a lower impact on the network cost, which in turn decreases with m.

Finally, the proposed heuristic performs very close to the optimal solutions (less than 4.2% for $d_{ik} = 500$ kb/s and less than 3.1% for $d_{ik} = 1$ Mb/s), and in all cases the computation time is below 30 seconds for all the tested instances. Even when compared to the bound provided by the continuous relaxation, the performance gap is less than 20% in all network scenarios.

Since for small network sizes ($m = 30, 40, 50$) it can be observed that gap_I is significantly smaller than gap_B, we can guess that even for larger topologies (where FCSD cannot be solved and consequently gap_I cannot be computed) H-FCSD performs considerably closer to the optimal integer solution than what is indicated by the gap_B value.

We further observe that the gap_L value is quite small (less than 10%) for all network instances, which motivates the choice to use the continuous relaxation in our heuristics as a basis to obtain good integer solutions applying the randomized rounding technique.

To compare the results obtained with FCSD with those of the FC-UAR model, we assume that the set S of candidate sites considered in FCSD is the same as the set R of overlay nodes used in FC-UAR (as a consequence, $m = r$). In this way,

Table 4.6: Solutions provided by the FCSD model, the FCSD continuous relaxation and the H-FCSD heuristic, with 20 TPs and 20 DNs.

d_{ik}=500 kb/s

	FCSD					FCSD Continuous Relaxation				H-FCSD					
m	N_R	N_L	Cost	Time	gap_L %	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I %	gap_B %
30	19.6	150.1	1001.3	12.4	7.72	13.8	205.3	929.5	0.3	19.8	146.3	1032.0	0.5	3.07	11.03
40	19.2	146.6	993.2	244.8	9.08	13.9	269.5	910.5	0.7	21.0	154.2	1025.9	1.3	3.29	12.67
50	19.5	148.3	981.9	4665.8	9.71	13.9	321.2	895.0	1.8	21.4	153.7	1022.6	1.6	4.15	14.26
60						13.8	380.6	879.8	3.2	22.2	160.0	1018.2	3.2		15.73
80						13.7	491.5	858.3	10.9	23.3	163.5	1017.9	8.2		18.59
100						13.7	603.4	845.5	28.0	23.6	163.1	1013.4	15.9		19.86

d_{ik}=1000 kb/s

	FCSD					FCSD Continuous Relaxation				H-FCSD					
m	N_R	N_L	Cost	Time	gap_L %	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I %	gap_B %
30	21.7	167.8	1813.4	5.6	4.54	15.5	208.2	1734.6	0.3	22.2	165.8	1830.8	0.6	0.96	5.55
40	22.7	174.8	1795.1	65.3	5.72	15.9	266.3	1697.9	0.7	23.2	173.2	1829.9	1.3	1.94	7.77
50	22.9	175.1	1776.1	3655.5	6.60	15.8	324.1	1666.2	1.9	24.6	178.3	1829.5	3.5	3.01	9.80
60						15.8	380.6	1639.6	3.2	26.1	187.0	1823.2	6.3		11.20
80						15.4	492.7	1593.7	11.0	27.2	187.8	1821.3	13.2		14.28
100						15.5	595.5	1568.0	27.7	28.0	190.5	1817.5	30.0		15.91

the solution of the FCSD model measures the advantage obtained by optimizing the number and position of the installed overlay nodes, in addition to the user assignment and traffic routing optimization performed by the FC-UAR model. For the sake of comparison we also added two columns in Table 4.7 that show the network cost achieved with the FC-UAR model and the percentage gap (gap_c) between the cost provided by the FCSD model and that obtained solving FC-UAR. To make a fair comparison between the two models, in the Table we reported for FC-UAR the objective function value (3.1) increased by the cost necessary to install r overlay nodes ($r \cdot 10$ monetary units, with the settings used in this scenario). Table 4.7 reports the number of installed overlay nodes (N_R), overlay links (N_L), the total network cost and the processing time to get the optimal solution. We note that the gap between the costs obtained with FC-UAR and FCSD is remarkable and it increases with increasing m values.

Table 4.7: Solutions provided by the FCSD model with 20 TPs and 20 DNs.

d_{ik}=500 kb/s

FCSD					FC-UAR	
m	N_R	N_L	Cost	Time (s)	Cost	$gap_c(\%)$
30	19.6	150.1	1001.3	12.4	1086.8	8.5
40	19.2	146.6	993.2	244.8	1172.9	18.1
50	19.5	148.3	981.9	4665.8	1246.8	27.0

d_{ik}=1000 kb/s

FCSD					FC-UAR	
m	N_R	N_L	Cost	Time (s)	Cost	$gap_c(\%)$
30	21.7	167.8	1813.4	5.6	1879.2	3.6
40	22.7	174.8	1795.1	65.3	2119.7	18.1
50	22.9	175.1	1776.1	3655.5	2185.4	23.0

We also simulated large-scale network scenarios with $n = 100$ TPs and $p = 10$ DNs, which can be seen as acting like concentrator nodes or access points towards other networks.

The results obtained with the H-FCSD heuristic and the continuous relaxation of the FCSD model are shown in Table 4.8, with m ranging from 100 to 300 and for different d_{ik} values (viz. 20 kb/s and 40 kb/s), and they are in line with the observations reported above.

Table 4.8: Large-size instances: solutions provided by the FCSD continuous relaxation and the H-FCSD heuristic, with 100 TPs and 10 DNs.

d_{ik}=20 kb/s

	FCSD Continuous Relaxation				H-FCSD				
m	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_B %
100	23.3	570.1	303.7	7.3	27.3	241.9	351.1	4.8	15.61
200	21.7	1036.0	284.8	128.9	27.2	236.3	350.0	58.8	22.89
300	20.8	1564.9	274.7	792.1	27.1	236.3	348.2	241.8	26.76

d_{ik}=40 kb/s

	FCSD Continuous Relaxation				H-FCSD				
m	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_B %
100	23.4	570.8	374.8	7.6	27.4	242.3	429.8	4.8	14.67
200	21.8	1035.1	353.5	131.3	27.4	237.6	429.7	57.9	21.56
300	20.9	1567.6	342.1	979.9	27.1	236.3	425.8	248.4	24.47

On the other hand, the results summarized in Table 4.9, are obtained with the FCSD and FC-UAR models, with $n = 100$ TPs and $p = 10$ DNs, m ranging from 30 to 50 overlay nodes and for d_{ik}= 20 kb/s and 40 kb/s, and they are also in line with what was reported above. In this case the processing time to obtain the optimal solutions is almost negligible.

A variation of the above scenarios is further considered, where the offered traffic is not constant, but uniformly distributed at random between $\frac{1}{2}d_{ik}$ and $\frac{3}{2}d_{ik}$. We observed that the results obtained are very close to those reported in Tables 4.6 and 4.8, with a maximum gap inferior to 1%.

b) Effect of the Traffic Demands: Transit-Stub topologies

To evaluate the behavior of the FCSD model with Transit-Stub topologies, we considered the same scenarios illustrated previously for FC-UAR.

The numerical results obtained with the FCSD model, its continuous relaxation and the H-FCSD heuristic, averaged over all network topologies and random TPs/DNs distributions, are shown in Table 4.10. It can be seen that due to the hierarchical structure of Transit-Stub topologies, a large number of overlay links is selected in the planned SON. Moreover, even in this scenario H-FCSD performs close to the optimal solution provided by the FCSD model.

If now we compare the results of the FCSD model with those obtained with FC-

Table 4.9: Solutions provided by the FCSD model with 100 TPs and 10 DNs.

d_{ik}=20 kb/s

FCSD					FC-UAR	
m	N_R	N_L	Cost	Time (s)	Cost	$gap_c(\%)$
30	24.1	235.2	320.5	0.5	377.3	17.7
40	24.0	232.1	317.8	4.9	474.7	49.4
50	24.0	231.1	317.4	34.0	573.2	80.6

d_{ik}=40 kb/s

FCSD					FC-UAR	
m	N_R	N_L	Cost	Time (s)	Cost	$gap_c(\%)$
30	24.1	235.2	399.5	0.6	454.7	13.8
40	24.0	232.1	395.7	5.5	549.4	38.8
50	24.0	230.9	394.8	29.1	646.4	63.7

UAR (Tables 4.3 and 4.11), we can observe that the FCSD model installs, in average, considerably fewer overlay nodes and links than FC-UAR, thus reducing consistently the cost of the planned network.

Table 4.10: Transit-Stub topologies: solutions provided by the FCSD model, the FCSD continuous relaxation and the H-FCSD heuristic, with 100 TPs, 100 DNs, 100 CSs and d_{ik}=10 kb/s.

FCSD					FCSD Continuous Relaxation				H-FCSD					
N_R	N_L	Cost	Time	gap_L %	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I %	gap_B %
38.0	338.1	1272.4	484.1	7.61	27.5	508.8	1182.4	5.3	48.5	347.2	1434.1	1.5	12.71	21.29

c) Effect of the Traffic Demands: Power Law topologies

To better capture the power law distributions for node degrees that characterize Internet topologies, we further considered BRITE, a degree-based topology generator [44, 45]. BRITE was used to generate power law topologies with 100 CSs, 20 TPs and 20 DNs on a square area with edge equal to 1000; the Barabási $-$ Albert model [48] was adopted with default parameters provided by BRITE.

Table 4.11: Transit-Stub topologies: solutions provided by the FCSD model with 100 TPs, 100 DNs and d_{ik}=10 kb/s.

FCSD					FC-UAR	
m	N_R	N_L	Cost	Time (s)	Cost	$gap_c(\%)$
50	25.3	310.9	954.6	1.6	1200.0	25.7
100	38.0	338.1	1272.4	484.1	1873.8	47.3
200	72.3	386.9	1768.7	2436.0	3040.4	71.9

The area was divided into $N = 25$ ISPs, each occupying an $L \times L$ square, with $L = 200$. Moreover, the bandwidth cost matrix was generated as for the custom topology generator.

Table 4.12 reports the numerical results obtained with such topologies, with an offered traffic d_{ik} equal to 500 and 1000 kb/s. The results confirm the behavior already observed for the random instances obtained with the custom generator, where the number of installed overlay nodes and links increases with increasing traffic demands. H-FCSD plans SON topologies with a reasonable cost in a very short computing time, which is almost independent of the traffic offered to the network.

Table 4.12: Power Law topologies: solutions provided by the FCSD model, the FCSD continuous relaxation and the H-FCSD heuristic in the topologies with node degree power laws generated using BRITE, with 100 CSs, 20 TPs and 20 DNs.

	FCSD					FCSD Continuous Relaxation				H-FCSD					
d_{ik}	N_R	N_L	Cost	Time	gap_L %	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I %	gap_B %
500	17.1	87.8	987.9	70.1	11.58	11.3	146.0	885.4	0.6	22.5	96.6	1272.7	0.3	28.83	43.74
1000	24.4	110.7	2022.8	48.8	12.18	15.3	155.9	1803.2	0.7	29.6	119.6	2583.2	0.4	27.70	43.26

d) Effect of the Traffic Demands: Rocketfuel topologies

We considered four diverse ISP backbone topologies, listed in Table 4.13, collected by Rocketfuel [40, 41], an Internet mapping tool.

In such topologies, all routers can be selected to install overlay nodes and all links can be selected as overlay links. The cost of each overlay link was set proportional to its length. For each topology we generated 10 random distributions of $n = 20$ TPs and $p = 20$ DNs, where each TP offers the same amount of traffic d_{ik} to all destination nodes. All other parameters are the same as in the previous network scenarios.

Table 4.13: Rocketfuel-inferred ISP topologies: number of backbone routers and links (including access and egress links).

Network	Location	Routers	Links
Ebone	EU	88	362
Tiscali	EU	164	696
Exodus	US	80	334
Abovenet	US	145	788

The numerical results, averaged over all network topologies and random TPs/DNs distributions, are shown in Table 4.14 for $d_{ik} = 500$ and 1000 kb/s.

For these topologies we further measured the average length of the paths chosen by the FCSD model to route traffic flows; the path length is expressed in terms of the number of hops traversed by the flow (Hop Count, H_C). Such performance figure is reported in the last column of Table 4.14.

Note that in all such real scenarios the proposed heuristic performs very well and close to the optimum, thus representing a practical solution for the planning of SONs. Furthermore, the average path length increases with increasing traffic demands, since an increasing fraction of the offered traffic must be routed on longer (and generally costlier) paths due to capacity constraints.

e) Effect of the Installation and Bandwidth Reservation Costs

We evaluate the effect of the bandwidth cost as well as the nodes' installation cost on our models' performance, considering a random network scenario with $n = p = 20$ TPs and DNs and $m = 50$ CSs. The offered traffic d_{ik} is equal to 500 kb/s. The solution, and in particular the number of installed nodes and links, intuitively depends on two factors: the ratio β between the overlay nodes' installation cost and the bandwidth reservation cost, and the ratio δ between the cost of a link connecting two CSs which belong to different ISPs and that of a link connecting two CSs belonging to the same ISP.

Tables 4.15 and 4.16 report the results obtained varying, respectively, the parameters β and δ. Table 4.15 shows that if the cost of installing an overlay node decreases with respect to the bandwidth reservation cost, the proposed models and heuristics tend to install more overlay nodes.

Note that the computing time of the H-FCSD heuristic is very short and the approximate solution is very close to the optimal solution provided by the FCSD model. Moreover, when β diminishes, the performance gap between H-FCSD and both the

Table 4.14: Rocketfuel topologies: solutions provided by the FCSD model, the FCSD continuous relaxation and the H-FCSD heuristic, with 20 TPs and 20 DNs.

d_{ik}=500 kb/s

	FCSD					FCSD Continuous Relaxation				H-FCSD						
Network	N_R	N_L	Cost	Time	gap_L	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I	gap_B	H_C
Ebone	26.8	115.6	2150.3	105.4	7.43	15.8	171.2	2001.6	0.4	34.9	135.0	2418.0	0.3	12.45	20.80	4.96
Tiscali	26.7	111.4	2021.3	79.2	7.72	16.3	184.5	1876.5	0.6	34.5	128.5	2394.4	0.4	18.46	27.60	5.14
Exodus	25.8	116.3	3432.9	36.3	4.32	14.2	159.1	3290.8	0.4	32.4	131.3	3729.6	0.3	8.64	13.33	4.33
Abovenet	24.0	114.0	5071.2	193.9	2.49	13.9	215.2	4948.0	1.0	33.5	135.6	5605.9	0.7	10.54	13.30	4.74

d_{ik}=1000 kb/s

	FCSD					FCSD Continuous Relaxation				H-FCSD						
Network	N_R	N_L	Cost	Time	gap_L	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I	gap_B	H_C
Ebone	32.1	135.5	4069.3	8.3	4.83	20.4	175.0	3907.4	0.6	41.5	155.9	4438.6	0.3	8.36	13.59	5.17
Tiscali	32.1	135.4	3813.7	20.7	4.80	20.7	183.8	3639.0	0.6	41.5	156.1	4408.7	0.5	15.60	21.15	5.30
Exodus	30.3	134.0	6659.0	5.4	2.55	19.1	160.1	6493.4	0.5	37.5	152.1	7170.5	0.2	7.68	10.43	4.55
Abovenet	29.8	138.5	9996.8	27.4	1.79	17.1	223.3	9821.0	1.0	40.4	166.4	11057.5	0.6	10.61	12.59	4.92

FCSD model and the lower bound greatly decreases.

Similarly, when δ increases, we observed that our model and heuristic tend to deploy more overlay nodes and links; the majority of these links connect overlay nodes belonging to the same ISP, thus diminishing the number of longer (and costlier) links that traverse different ISPs in the planned SON. This behavior is reflected in the numerical results reported in Table 4.16.

Table 4.15: Variable cost ratio β: solutions provided by the FCSD model, the FCSD continuous relaxation and the H-FCSD heuristic with 20 TPs, 20 DNs, 50 CSs and d_{ik}=500 kb/s.

	FCSD					FCSD Continuous Relaxation				H-FCSD					
β	N_R	N_L	Cost	Time	gap_L %	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I %	gap_B %
10	19.5	148.3	981.9	4665.8	9.71	13.9	321.2	895.0	1.5	21.4	153.8	1018.2	1.6	3.70	13.77
1	30.8	209.9	783.5	108.9	2.45	16.0	266.5	764.8	1.3	31.4	192.7	800.6	1.3	2.18	4.68
1/10	50.0	239.6	748.5	1.7	0.00	50.0	239.4	748.5	1.1	50.0	237.8	762.2	1.6	1.83	1.83

Moreover, Table 4.17 illustrates the effect of the costs in a network scenario with 20 TPs, 20 DNs, 40 CSs and $d_{ik} = 500$ kb/s. These results are in line with those reported in Table 4.14. For comparison, the results obtained with FC-UAR are also reported in the last row of the Table since they represent a bound for FCSD corresponding to a negligible cost of the overlay nodes.

f) Effect of the Gain parameter

We then evaluate the Profit Maximization SON Design problem, considering a random network scenario with 20 TPs, 20 DNs, $m = 40$ CSs and offered traffic d_{ik}=500 kb/s. We assume, as for PM-UAR, that the gain per bandwidth unit g_i is a random variable with average equal to G and a uniform distribution between $G/2$ and $3G/2$, where G ranges between 0 and 0.01 monetary units per Mb/s.

Figure 4.5 reports an example of the planned networks when applying the PMSD model to the same instance considered in Figure 4.4 with $m = 40$ CSs and with two different requirements on the end-user traffic, $d_{ik} = 500$ kb/s and $d_{ik} = 1$ Mb/s for all TPs and DNs; G is equal to 0.005 monetary units per Mb/s. CSs, TPs and DNs are represented with circles, triangles and squares, respectively. In this case the planned SONs cover only a subset of the TPs to maximize the operator's profit.

Figure 4.6 shows the number of end-users covered by the SON as a function of G for both the PMSD model and the H-PMSD heuristic. Note that the two curves almost overlap, as the H-PMSD heuristic is quite accurate in determining the optimal number of users to serve to maximize the SON operator's profit. Obviously, for small G values, the SON is not profitable enough to cover any of the end-users; as

Table 4.16: Variable cost ratio δ: solutions provided by the FCSD model, the FCSD continuous relaxation and the H-FCSD heuristic with 20 TPs, 20 DNs, 50 CSs and d_{ik}=500 kb/s.

	FCSD					FCSD Continuous Relaxation				H-FCSD					
δ	N_R	N_L	Cost	Time	gap_L %	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap_I %	gap_B %
1	19.5	148.3	981.9	4665.8	9.71	13.9	321.2	895.0	1.5	21.4	153.8	1018.2	1.6	3.70	13.77
2	21.8	154.3	1355.9	13853.6	9.50	14.5	329.7	1238.3	1.4	24.2	162.4	1404.7	2.8	3.60	13.44
5	26.6	168.8	2356.0	7060.3	7.48	15.1	281.3	2192.0	1.5	30.3	174.3	2441.7	4.7	3.64	11.39
10	31.1	182.9	3925.7	2780.5	5.46	15.6	256.5	3722.4	1.3	35.2	184.5	4058.6	3.4	3.39	9.03

Table 4.17: Variable cost ratio β: solutions provided by the FCSD model with 20 TPs, 20 DNs, 40 CSs and d_{ik}=500 kb/s.

β	N_R	N_L	Cost	Time (s)
10	19.3	148.5	987.0	203.4
1	28.7	197.2	795.3	12.0
1/10	40.0	215.9	767.9	0.7
$FC-UAR$	40.0	217.3	765.6	1.2

G increases, the SON covers more end-users, eventually all of them. Similar results have been observed in several network topologies with a varying number of CSs.

Table 4.18 reports, for the PMSD model, its continuous relaxation and the H-PMSD heuristic, the total number of installed nodes and links, the SON operator's profit (i.e., the value of the objective function ((3.20), page 21) or the approximate value obtained with H-PMSD) and computing time, as a function of G. Note that when G increases, the planned network covers more end-users, and as a consequence it comprises more overlay nodes and links.

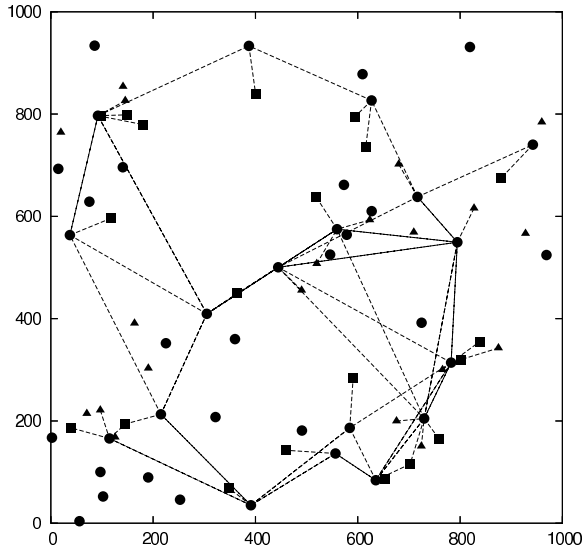
Column gap_I shows the percentage gap between the profit provided by the H-PMSD heuristic and the optimal profit obtained with the PMSD model, while column gap_B reports the gap between H-PMSD and the upper bound provided by the continuous relaxation of the PMSD model. Column gap_L provides the gap between the PMSD model and its continuous relaxation. The performance gap is slightly larger with respect to the full coverage design problem; this is mainly due to the difficulty in determining the optimal subset of users to cover. However, note that the performance gap is still largely acceptable since in all cases it is less than 30%, and it greatly decreases with increasing values of G.

Finally, Figure 4.7 compares the number of end-users covered by the SON for the PMSD and PM-UAR models ($r = m$) as a function of G. Since the cost of installing overlay nodes is not considered in PM-UAR, such model tends to cover more end-users than PMSD for all G values.

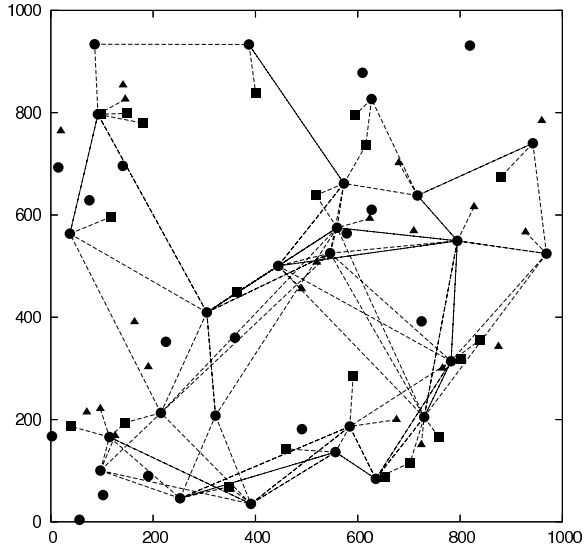
We further tested the H-PMSD heuristic with large-scale network instances. Table 4.19 reports the numerical results obtained with 50 TPs, 50 DNs, 100 CSs and d_{ik}=20 kb/s for both the H-PMSD and the PMSD continuous relaxation. Even in this network scenario the proposed heuristic provides accurate solutions with a considerably reduced computing time.

Table 4.18: Solutions provided by the PMSD model, the H-PMSD heuristic and the continuous relaxation with 20 TPs and DNs, 40 CSs and d_{ik}=500 kb/s.

	PMSD					PMSD Continuous Relaxation				H-PMSD					
G	N_R	N_L	Profit	Time	gap_L %	N_R	N_L	Profit	Time	N_R	N_L	Profit	Time	gap_I %	gap_B %
0.005	15.0	103.0	71.9	560.8	56.50	9.7	234.1	165.3	0.6	14.3	115.5	51.0	1.4	29.07	69.15
0.006	17.2	125.7	241.8	359.5	25.81	11.2	248.0	325.9	0.6	17.9	128.2	213.1	1.5	11.87	34.61
0.007	18.2	138.0	422.7	326.1	16.31	12.7	261.8	505.1	0.6	18.6	144.2	390.0	1.6	7.74	22.79
0.008	18.9	142.8	616.3	263.3	11.76	13.4	267.9	698.4	0.6	20.1	149.7	585.0	1.5	5.08	16.24
0.009	19.2	146.5	814.8	190.5	9.17	13.8	270.7	897.1	0.6	20.7	153.4	786.5	1.6	3.47	12.33
0.010	19.2	146.5	1015.0	195.0	7.51	13.8	270.7	1097.4	0.6	20.7	153.5	987.4	1.6	2.72	10.02



(a) 500 kb/s



(b) 1000 kb/s

Figure 4.5: Sample SONs planned by the PMSD model with increasing traffic demands (500 and 1000 kb/s). The number of TPs and DNs is 20, while the number of CSs is 40. The average gain per bandwidth unit, G, is equal to 0.005 monetary units per Mb/s. CSs, TPs and DNs are represented with circles, triangles and squares, respectively.

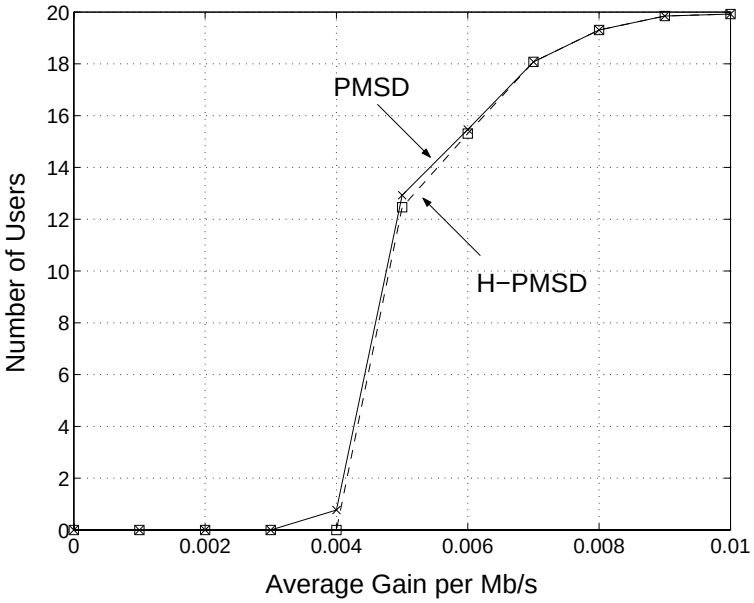


Figure 4.6: Number of end-users covered by the SON with the PMSD model and the H-PMSD heuristic as a function of the average gain per bandwidth unit, with 20 TPs, 20 DNs, 40 CSs and d_{ik}=500 kb/s.

Table 4.19: Large-size instances: solutions provided by the H-PMSD heuristic and the PMSD continuous relaxation, with 50 TPs, 50 DNs, 100 CSs and d_{ik}=20 kb/s.

	PMSD Continuous Relaxation				H-PMSD				
G	N_R	N_L	Profit	Time	N_R	N_L	Profit	Time	gap_B %
0.005	13.6	127.5	213.0	78.8	27.2	269.6	13.2	123.0	92.38
0.006	16.2	131.3	416.5	71.6	29.0	312.5	215.5	114.6	48.26
0.007	17.9	134.4	642.0	66.3	29.5	331.5	444.3	111.7	30.79
0.008	18.8	134.9	879.5	68.9	30.1	342.4	683.6	114.8	22.27
0.009	19.5	134.3	1120.7	66.3	30.5	351.4	929.3	111.8	17.08
0.010	19.9	134.4	1363.9	67.2	31.0	361.4	1171.8	111.9	14.08

g) Effect of the Budget parameter

Finally, we evaluate the impact of the budget parameter on the PMSD model considering a random network scenario with 20 TPs and DNs, 40 CSs, d_{ik} = 500 kb/s and several budget (B) values in the 500 to 1000 range.

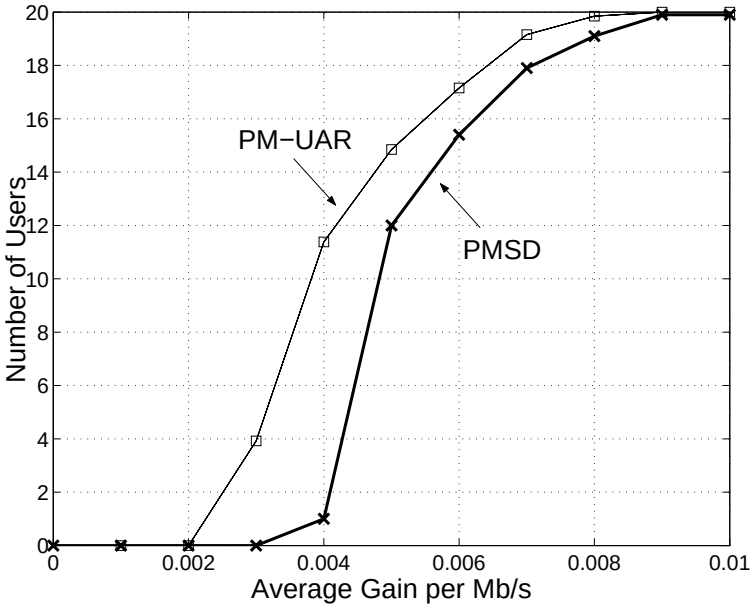


Figure 4.7: Number of end-users covered by the SON as a function of the average gain per bandwidth unit, with 20 TPs, 20 DNs and 40 CSs.

Figure 4.8 shows the number of end-users accepted in the SON as a function of the budget, for different G values. Similarly to the PM-UAR model, given a G value, the number of end-users covered by the SON increases as the budget increases, until it reaches its maximum, which can be obtained from Figure 4.6.

For small budget values ($B \leq 700$), all curves practically overlap except for that corresponding to $G = 0.005$ monetary units per Mb/s. This is due to the fact that for $G > 0.005$ the number of users covered by the PMSD model is limited by the budget constraint, so that the model chooses the most profitable users that can be accepted in the SON given the budget value. On the other hand, for $G = 0.005$ it is not profitable to cover the same number of users than for $G > 0.005$, even if the budget would allow the SON operator to serve them.

Table 4.20 illustrates in detail the characteristics of the solutions provided in such scenario by the PMSD model, its continuous relaxation and the H-PMSD heuristic, for $G = 0.010$ monetary units per Mb/s and for different budget values. Note that in this scenario the scaling technique described in Section 3.3.2 was used for H-PMSD.

The results show that higher profits are achieved by deploying higher-cost SONs. However, this also increases the economic risk the SON operator faces in the deployment of the overlay network.

Finally, note that the performance gap between H-PMSD and PMSD is quite

small and decreases with increasing budget values.

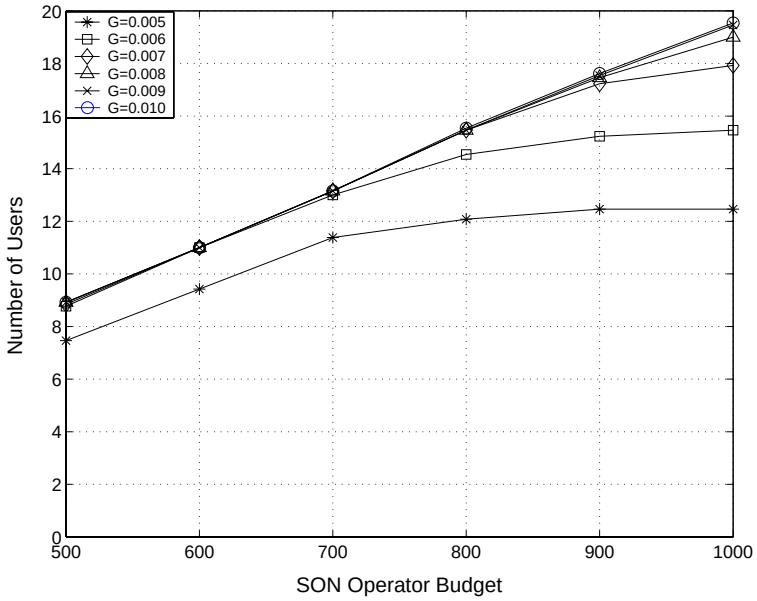


Figure 4.8: Number of end-users covered by the SON with the PMSD model as a function of the budget for different values of the average gain per bandwidth unit G, with 20 TPs, 20 DNs, 40 CSs and d_{ik}=500 kb/s.

Table 4.20: Solutions provided by the PMSD model, its continuous relaxation and the H-PMSD heuristic with 20 TPs and DNs, 40 CSs, $G = 0.010$ monetary units per Mb/s and d_{ik}=500 kb/s.

	PMSD					PMSD Continuous Relaxation				H-PMSD					
B	N_R	N_L	Profit	Time	gap_L %	N_R	N_L	Profit	Time	N_R	N_L	Profit	Time	gap_I %	gap_B %
500	14.3	87.2	588.1	3956.6	27.76	8.4	230.4	814.1	1.9	16.4	87.0	433.8	2.4	26.24	46.71
600	15.6	101.8	723.1	1394.5	21.07	9.8	246.4	916.1	1.5	17.0	99.0	507.2	2.2	29.85	44.63
700	16.4	111.9	827.4	1420.8	17.05	11.3	251.1	997.5	1.2	18.1	112.0	627.4	2.1	24.17	37.10
800	17.4	126.8	913.5	920.8	13.68	12.5	264.0	1058.3	1.1	18.8	123.7	686.0	1.9	24.91	35.18
900	18.2	136.0	973.2	674.9	11.07	13.5	270.7	1094.4	0.9	20.4	141.2	808.9	1.8	16.88	26.09
1000	19.2	145.6	1008.2	339.9	8.13	13.8	270.7	1097.4	0.7	20.6	147.0	877.1	1.8	13.00	20.08

Chapter 5

Very Large-Scale Neighborhood Search Algorithms for the Design of Service Overlay Networks

The SON design problem is NP-hard and computationally intractable for large-size network instances; a practical approach is to employ a heuristic that can find nearly optimal solutions within a reasonable amount of time. However, the heuristics we introduced in Chapter 3 for solving the SON design problem are, in some network scenarios, unable to provide a good solution due to huge memory consumption and computational effort.

For this reason, to deal with large-scale network scenarios, in this Chapter we reformulate the Minimum Cost SON design problem in a way that allows us to solve it in a very effective way, even for large instances, using an efficient *tabu search* based heuristic.

As for the models proposed in Chapter 3, the model illustrated in the following selects the optimal number and position of overlay nodes, as well as the capacity reserved on each overlay link, and assigns users to access overlay nodes. Our tabu search based heuristic combines in an innovative way polynomial and very large-scale neighborhoods (VLSN), where the VLSN of the local minimum given by the tabu search is explored efficiently to obtain in a short time a new solution that is both far from the current solution and cost-decreasing.

We provide numerical results of the proposed heuristic on a set of realistic, large-size instances, and discuss the effect of different parameters on the characteristics of the planned networks. Furthermore, we compare such results with the bound obtained solving our SON design model in medium-scale network scenarios. We show that in the considered network topologies the proposed heuristic performs very

close to the optimum with a short computing time, thus permitting to plan large-scale SONs very efficiently.

The Chapter is structured as follows: Section 5.1 presents the novel formulation of the Minimum Cost SON design model. Section 5.2 defines the polynomial size and very large-scale neighborhoods considered in our heuristic, as well as the methods used to search them efficiently. Section 5.3 introduces the proposed tabu search based heuristic, which solves small-to-very-large-size instances of the SON design problem. Finally, Section 5.4 presents numerical results that show the effect of different parameters on the characteristics of the planned network.

5.1 User Assignment and Overlay Node Placement Model

In this Section we slightly re-formulate the Minimum Cost SON Design model in a way that allows us to solve it efficiently with a tabu search based approach. To this end, we adopt the same variable definitions and notations used in Chapter 3, which are summarized in Table 5.1, along with novel variables and parameters which are necessary to our formulation.

Table 5.1: Basic Notation for the MCSD problem.

S	Set of Candidate Sites
I	Set of Test Points
c_j^I	Cost for installing an overlay node in CS j
c_{jl}^B	Cost for buying one bandwidth unit between CSs j and l
c_{ij}^A	Access cost per bandwidth unit between TP i and CS j
c_{jk}^E	Egress cost per bandwidth unit between CS j and TP k
w_{ik}	Traffic generated by TP i towards TP k
Γ_j	Maximum capacity of the overlay node installed at CS j
o_i	Total traffic originating from TP i
d_i	Total traffic destined to TP i
a_{ij}	0-1 parameter that indicates if TP i can access the SON through CS j
b_{jl}	0-1 parameter that indicates if CSs j and l can be connected with an overlay link
z_j	0-1 variable that indicates if an overlay node is installed in CS j
x_{ij}	0-1 variable that indicates if TP i is assigned to CS j
f_{jl}^i	Traffic flow originating from TP i and routed on link (j, l)

Let c^E_{jk} denote the cost per bandwidth unit for the traffic transmitted on the egress link between CS j and TP $k \in I$.

The traffic generated by TP i towards TP k is given by the parameter w_{ik}, $i, k \in I$. Each overlay node j is now characterized by a maximum capacity, denoted by Γ_j, which restricts the volume of the total traffic flow incoming from the test points that are assigned to such overlay node.

To simplify the notation, we define $o_i = \sum_{k \in I} w_{ik}$ and $d_i = \sum_{k \in I} w_{ki}$, which represent, respectively, the total traffic originating from and destined to TP i.

Finally, flow variables f^i_{jl} represent the traffic flow originating from TP $i \in I$ and routed on link (j, l).

Given the above parameters and variables, the Minimum Cost SON Design (MCSD) problem can be formulated as follows:

$$min \quad \left\{ \sum_{j \in S} c^I_j z_j + \sum_{i \in I} \sum_{j \in S} (o_i c^A_{ij} + d_i c^E_{ji}) x_{ij} + \sum_{i \in I} \sum_{j \in S} \sum_{l \in S} c^B_{jl} f^i_{jl} \right\} \tag{5.1}$$

$$s.t.$$

$$\sum_{j \in S} x_{ij} = 1, \quad \forall i \in I \tag{5.2}$$

$$x_{ij} \leq z_j a_{ij}, \quad \forall i \in I, j \in S \tag{5.3}$$

$$\sum_{i \in I} o_i x_{ij} \leq \Gamma_j z_j, \quad \forall i \in I, j \in S \tag{5.4}$$

$$\sum_{l \in S} f^i_{jl} - \sum_{l \in S} f^i_{lj} = o_i x_{ij} - \sum_{k \in I} w_{ik} x_{kj}, \quad \forall i \in I, j \in S \tag{5.5}$$

$$f^i_{jl} \leq b_{jl} M, \quad \forall i \in I, j, l \in S \tag{5.6}$$

$$z_j, x_{ij} \in \{0, 1\}, \quad \forall i \in I, j \in S \tag{5.7}$$

$$f^i_{jl} \geq 0, \quad \forall i \in I, j, l \in S \tag{5.8}$$

The objective function (5.1) imposes the minimization of the sum of the nodes' installation costs and the flow transportation costs (access, egress and transport costs). Constraints (5.2) and (5.3) dictate that each test point is assigned to one overlay node

subject to the test point coverage parameters. Constraints (5.4) require that the capacity of each overlay node is not violated. Constraints (5.5) are the flow conservation equations at the overlay nodes. Constraints (5.6) impose that traffic flow can be routed on the overlay link (j, l) only if such link exists (M being a very large constant value). Finally, constraints (5.7) and (5.8) are integrality and non-negative constraints on the problem variables.

Note that capacity over-provisioning is one of the most widely accepted dimensioning criterion for providing QoS in the Internet backbone. For this reason, we do not consider capacity bounds for the SON backbone links since we assume that enough capacity is always available between overlay nodes. However, it is easy to extend the MCSD model taking into account also overlay links capacities, simply changing constraints (5.6) as follows:

$$f_{jl}^{i} \leq b_{jl} U_{jl}, \quad \forall i \in I, j, l \in S \tag{5.9}$$

where U_{jl} represents the capacity of overlay link (j, l).

Finally, we observe that the above model is NP-hard since it includes the Hub Location problem [38, 49, 19] as special case. The reader is referred to Appendix A.1 for more details on the Hub Location problem.

5.2 Neighborhoods Definition

The MCSD model (5.1)-(5.8) can be solved in a reasonable computational time only for small and medium-size network instances, as we will show in Section 5.4. Therefore, in the following we describe the proposed heuristic for solving the Minimum Cost SON Design problem (named TS-MCSD), which is based on a tabu search approach.

In TS-MCSD, the SON design problem is split into two subproblems: a *location problem*, in which it is decided where the overlay nodes should be installed, and an *allocation problem*, in which it is determined to which overlay node each test point should be allocated. The rationale of this approach is that a neighborhood search both on the location and the allocation problem would be too expensive in terms of computational effort.

Since TS-MCSD is based on the concept of search in the neighborhood of a given solution, in this Section we define the neighborhoods considered in our heuristic and the techniques used to search them efficiently. Then we proceed by illustrating in detail the TS-MCSD heuristic (Section 5.3).

We consider two types of neighborhoods: polynomial size and very large-scale neighborhoods (VLSN). The first neighborhood is applied both to the *location* and to

the *allocation problem*. VLSN is applied only to the *allocation problem*, as we will explain later in this Section.

5.2.1 Polynomial Size Neighborhoods

We define two polynomial size neighborhoods: the first is considered for the *location problem*, while the second for the *allocation problem*.

Polynomial Size Location Neighborhood (PSLN)

For the location problem, given a set of installed overlay nodes N, the neighborhood is generated by applying three moves:

- $m1$: remove one overlay node from N, i.e. close one overlay node;
- $m2$: add a candidate overlay node to N, i.e. install a new overlay node;
- $m3$: swap an overlay node in the set N with an overlay node which is in the candidate site set S but not in N.

The size of the neighborhood corresponding to applying $m1$, $m2$ and $m3$ is $O(|N|^2)$.

Polynomial Size Allocation Neighborhood (PSAN)

For the allocation problem, the set of installed overlay nodes is fixed and the solution is represented by the allocation of each test point to an overlay node. The neighborhood is generated by applying two moves:

- $a1$: allocate a TP i to an overlay node j different from the one to which it is allocated in the current solution;
- $a2$: swap the overlay nodes to which any two TPs are allocated; for example, supposing that in the current solution TP i is allocated to overlay node j and TP k is allocated to overlay node l, in the neighboring solution, i is allocated to l and k to j.

The size of this neighborhood is $O(|I|^2)$. Note that we accept only swap moves that lead to an improvement in the sum of the allocation costs (i.e., the access and egress costs).

5.2.2 Very Large-Scale Neighborhood (VLSN) for the Allocation Problem

We now define the *Cyclic Exchange Neighborhood* (CEN) [34, 35], which is the particular type of Very Large-Scale Neighborhood used in our work. CEN is used to perform cyclic exchanges of test points among a fixed set of overlay nodes.

We first introduce some notation, then we describe how to build the *test point improvement graph* and how to explore the *single test point cyclic exchange neighborhood*.

Basic Notation for VLSN

Let F be a feasible solution to the MCSD problem (5.1)-(5.8). The set F can be represented as a partition of the test point set I into m subsets, i.e., $F = \{I_1, I_2, ..., I_m\}$, where each subset $I_j, j = 1, ..., m$, corresponds to a potential overlay node and contains the set of test points assigned to that overlay node in the solution F. Eventually, I_j can be empty if no overlay node is installed at candidate site j.

The cost of each subset I_j, in terms of the installation costs and allocation costs (i.e., the access and egress costs), is given by

$$C(I_j) = \begin{cases} c_j^I + \sum_{i \in I_j} C_{ij} & \text{if } I_j \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

where $C_{ij} = o_i c_{ij}^A + d_i c_{ji}^E$.

The cost of the whole partition F is therefore

$$C(F) = \sum_{j=1}^{m} C(I_j).$$

A solution F is feasible if the following inequality holds for $j = 1, ..., m$:

$$O(I_j) = \sum_{i \in I_j} o_i \leq \Gamma_j.$$

The residual capacity r_j of each subset I_j, i.e., the capacity still available at an overlay node installed at j, is defined as:

$$r_j = \Gamma_j - O(I_j) \geq 0.$$

In the following, we denote by $H(i)$, $i \in I$, the overlay node serving test point i.

Test Point Exchanges

Given a solution F to the SON design problem, the *single test point cyclic exchange neighborhood* of F is defined as the set of new solutions obtainable from F by exchanging test points among overlay nodes in a cyclic manner.

Such a neighborhood is constructed by moving among the overlay nodes single test points conveniently chosen. The single test point neighborhood is defined in terms of single test point cyclic exchanges as follows.

A single test point cyclic exchange with respect to a solution F is defined as a test point sequence $R = (i_1, i_2, ..., i_q)$, such that each test point i_r in the sequence is assigned to a different overlay node in F, i.e., $H(i_r) \neq H(i_s)$ for $r \neq s$, $r, s = 1, ..., q$. The cyclic exchange represents therefore the following exchange of test points: test point i_1 is relinquished from overlay node $H(i_1)$ and assigned to overlay node $H(i_2)$, test point i_2 is relinquished from overlay node $H(i_2)$ and assigned to overlay node $H(i_3)$, and so on until test point i_q is relinquished from overlay node $H(i_q)$ and assigned to overlay node $H(i_1)$.

A cyclic exchange R is feasible if the capacity constraint of each overlay node involved in the cycle is still respected after the exchange, i.e., if $o_{i_{r-1}} \leq r_{H(i_r)} + o_{i_r}$ for $r = 2, ..., q$, and $o_{i_q} \leq r_{H(i_1)} + o_{i_1}$.

A cyclic exchange R is profitable, with respect to the total allocation cost, if the new solution F' obtained from F through R is such that $C(F') < C(F)$.

Note that the test point cyclic exchange R can be profitable for the allocation problem but not for the MCSD problem as a whole since this latter includes further the inter-overlay node bandwidth costs. Hence, once a profitable cyclic exchange R is found, the total cost of the new solution must be compared with the cost of the current solution to decide whether the cyclic exchange R is really profitable or not for the SON design problem.

Each feasible single test point cyclic exchange generates a new feasible solution F' from the current solution F. The set of all the new solutions obtainable from F through a feasible exchange R defines the single test point cyclic exchange neighborhood of F.

In order to efficiently detect improving single test point moves, such neighborhood is searched only partially using a heuristic approach which detects special negative cost cycles in a suitably defined graph, called the *test point improvement graph* [50, 51]. The improvement graph along with negative subset-disjoint cycles are described in the following Subsections.

The Test Point Improvement Graph

Given a solution F of the MCSD problem, the *test point improvement graph* relative to F is a directed graph $G^i(F) = (N^i(F), A^i(F))$ defined as follows. The set of nodes $N^i(F)$ contains a regular node, i, for each test point $i \in I$.

The set of arcs $A^i(F)$ contains an arc (i, k) for each pair of regular nodes i and k in $N^i(F)$, provided that test points i and k are assigned to two different overlay nodes in F, i.e., $H(i) \neq H(k)$, and that after the insertion of i in $I_{H(k)}$ and the removal of k from it, the capacity of overlay node $H(k)$ is not exceeded, i.e., $o_i - o_k \leq r_{H(k)}$. An arc connecting two nodes i and k, in fact, signifies that test point i leaves overlay node $H(i)$ and is assigned to overlay node $H(k)$, which in turn relinquishes test point k.

The cost α_{ik} of the arc (i, k) reflects the allocation cost variation of overlay node $H(k)$ and it is therefore defined as:

$$\alpha_{ik} = C_{iH(k)} - C_{kH(k)},$$

where $C_{iH(k)} = o_i c^A_{iH(k)} + d_i c^E_{H(k)i}$.

Negative subset-disjoint cycles

Having provided a description of the test point improvement graph, we can now define negative subset-disjoint cycles in it. A directed cycle $(n_1, ..., n_q)$, with $n_r \in N^i(F), r = 1, ..., q$, in the improvement graph $G^i(F)$ associated with solution F, is a negative subset-disjoint cycle if each one of its nodes is associated with a different overlay node and if the sum of the costs of its arcs is negative. It can be shown that the problem of detecting a negative subset-disjoint cycle in $G^i(F)$ is equivalent to the problem of finding a feasible profitable multi-exchange for F [52].

There is a one-to-one correspondence between the set of feasible cyclic exchanges relative to the current solution F and the set of subset-disjoint cycles in $G^i(F)$. Each cyclic exchange is profitable if and only if the corresponding subset-disjoint cycle is negative. In particular, the fact that the test point improvement graph includes only feasible arcs with respect to the capacity constraints ensures that each exchange corresponding to a cycle in $G^i(F)$ is a feasible exchange.

Note that the way we define the costs of the arcs in $A^i(F)$, considering only test points' allocation costs (access and egress costs), implies that the cost of each cycle in $G^i(F)$ reflects only the allocation cost variation of the solution due to the corresponding test point exchanges. Hence, if the cycle cost is negative, the associated exchange is said to be profitable for the SON design problem if and only if the total cost of the

corresponding solution, including installation costs, allocation (access/egress) costs and transfer costs, is smaller than the cost of the current solution.

Search for negative subset-disjoint cycles

As previously stated, the problem of finding an improving move in the single test point neighborhood can be reformulated as the problem of identifying negative-cost subset-disjoint cycles in $G^i(F)$. This problem is known to be NP-complete [52].

A heuristic algorithm has been proposed in [51] to detect negative-cost subset-disjoint cycles for the Capacitated Minimum Spanning Tree problem and such algorithm is then used in the Facility Location problem for the same goal [50]. Briefly, the algorithm is based on a modification of the label-correcting algorithm for the shortest path problem and changes are introduced to check for path subset-disjointness. The reader is referred to [50, 51] for further details concerning this algorithm.

5.3 TS-MCSD: a Very Large-Scale Neighborhood Search Heuristic to Solve the MCSD Problem

To solve the SON design problem, we develop a novel and efficient heuristic (named TS-MCSD) that uses the tabu search approach along with the polynomial size (PSAN and PSLN) and the Cyclic Exchange (CEN) Neighborhood concepts described in Section 5.2.

As stated before, the SON design problem is split into two subproblems: a *location problem* (i.e., the overlay node installation problem) and an *allocation problem* (i.e., the problem of allocating a TP to an overlay node).

In summary, the TS-MCSD heuristic proceeds as follows:

- Computing a feasible initial solution.
- Performing an iterative procedure (called Iterative Very Large-Scale Tabu Search) which consists of:
    1. an *optimization step* where the location problem is solved using a tabu search on the polynomial size location neighborhood (PSLN, Section 5.2.1);
    2. given the solution obtained in the optimization step, a *diversification step* is performed using the very large-scale neighborhood search (Section 5.2.2).

- Performing a post optimization step which consists in applying a local search to the polynomial size allocation neighborhood (PSAN) of the best solution found by the iterative procedure.

In the following we describe in detail the steps which compose the proposed heuristic.

5.3.1 Initial Solution

The initial solution is computed by means of the greedy algorithm described in the following, which is an extension of the one proposed in [53] for the Hub Location problem.

Given as input the problem parameters (namely: the traffic demand matrix, the costs and the capacity bounds), the algorithm provides an initial feasible solution in which all the test points are allocated. The pseudo-code in Algorithm 1 details how this procedure works.

Starting with an empty set of overlay nodes, a new overlay node is added until we reach a set that provides enough capacity to accommodate the whole flow originated at test points.

Overlay nodes are added according to a function that considers the cost of allocating all test points to the set of chosen overlay nodes. For each overlay node j, test point i is ordered according to non-decreasing access cost (Access Cost Order, $AC_{ord}[i]$) and according to non-decreasing amount of flow offered to the network (Traffic Order, $T_{ord}[i]$). Then, test point i is considered according to a weighted combination of the positions in the described orders, $W_{pos}[i]$ (see Algorithm 1 for details).

All the test points that can be assigned to the overlay node j without violating its capacity constraint are labeled as covered by j, unless they are already labeled as covered by another overlay node. The overlay node providing the maximum number of covered test points is chosen and installed.

Once a set of overlay nodes providing enough capacity to accommodate the whole flow is obtained, the allocations are computed by means of a greedy procedure based on the algorithm proposed in [54] to solve the General Assignment Problem. Such greedy procedure is detailed in Algorithm 3 (Section 5.3.2), since it is also used in the optimization step.

If the capacity constraint is satisfied, an initial feasible solution is found, otherwise the solution evaluated is taken as infeasible. In this latter case, other overlay nodes are added to the set of chosen overlay nodes until a feasible solution is obtained.

Algorithm 1 Pseudo-code specification for computing the initial solution.

for each candidate overlay node j **do**
order test points according to non-decreasing access cost
$AC_{ord}[i]$ = position of test point i in the order
order test points according to non-decreasing amount of the total flow (provided by the traffic demand matrix)
$T_{ord}[i]$ = position of test point i in the order
$W_{pos}[i] = |I| - T_{ord}[i] + 2 \cdot AC_{ord}[i]$
order test points according to non-decreasing value of $W_{pos}[i]$
while solution is not feasible **do**
determine the number ($NNCov$) of non covered test points that each overlay node can cover, according to $W_{pos}[i]$
open an overlay node in location j such that $NNCov[j]$ is maximum
if not all test points are covered **then**
continue
else
allocate test points (using the *Greedy Allocation algorithm*, Algorithm 3)
check if the capacity constraints are satisfied (check the feasibility of the solution)
end if
end while
end for

5.3.2 Iterative Very Large-Scale Tabu Search

The Iterative Very Large-Scale Tabu Search (I-VLS-TS) is an iterative procedure which consists of an optimization step, where the location problem is solved using a tabu search on the polynomial size location neighborhood (PSLN), and of a diversification step, which is performed applying the cyclic exchange neighborhood (CEN) search to the allocation problem.

CEN is applied to the local optimum determined by the tabu search, and it is searched only once in an attempt to obtain a new solution which is both far from the local optimum and cost-decreasing. Note that if the new solution is not profitable in terms of the total cost with respect to the current one, such solution is discarded. The pseudo-code of I-VLS-TS is given in Algorithm 2.

In the following, we first describe the tabu search approach and then the optimization step which is based on tabu search.

Algorithm 2 Pseudo-code specification of I-VLS-TS.

Initialize the current solution (x_{curr}) to the initial solution found using Algorithm 1
Initialize tabu list size and stopping criterion
repeat
(1) Compute the local minimum (x_{TS}) using tabu search: x_{TS} = TabuSearch(x_{curr})
(2) Apply once the Cyclic Exchange Neighborhood search to x_{TS}: x_{CEN} = CENSearch(x_{TS})
Set $x_{curr} = x_{CEN}$
until stopping criterion is met

Tabu Search

Tabu search is a local search based optimization method [55, 56] that can accept cost-increasing solutions in order to escape from local minima. This feature allows the neighborhood search to explore other parts of the solution space.

For each neighborhood, the best neighbor solution is used as new current solution and, if it improves upon the best solution found so far, it replaces this latter. This may cause the algorithm to cycle. To avoid cycles, a memory is kept of the last solutions accepted as current solution, and such solutions are forbidden. The moves leading to new current solutions are stored in a list (tabu list) for a certain number of iterations.

A solution produced by a move belonging to the tabu list, which we refer to as *tabu move*, is discarded. To avoid discarding good solutions, an aspiration criterion is defined: a solution generated by a tabu move is accepted if it satisfies the aspiration criterion. Tabu search stops when it reaches its stopping condition, for instance after a given number of visited neighborhoods without improvement in the best solution found. The tabu search parameters (tabu list dimension, stopping and aspiration criteria) play a fundamental role in implementing such metaheuristic, and in this work they have been set according to computational experience [53]:

- Tabu moves: a tabu move is represented by the node or the nodes involved in the move to be forbidden.
- Tabu list size: a static tabu list is considered containing up to 6 moves.
- Stopping criterion: the algorithm stops after 10 consecutive neighborhood searches without improvements in the best solution found.
- Aspiration criterion: a solution generated by a tabu move is accepted if it improves upon the best solution found so far.

Optimization Step

The optimization step is performed using the tabu search approach and the PSLN neighborhood. The solution is represented by the set of sites in which an overlay node is opened. Then, the allocation of test points to such overlay nodes is performed using the greedy algorithm, presented as Algorithm 3.

Algorithm 3 Pseudo-code specification of the Greedy Allocation algorithm.

while not all test points are allocated **do**
  for each test point i not allocated **do**
    order overlay nodes according to non-decreasing value of $f_{ij} = c_{ij}^A / \sum_{k \in I} w_{ik}$
    $t_1[i]$ = first overlay node in the order
    $t_2[i]$ = second overlay node in the order
    $\Delta[i] = |f_{it_1(i)} - f_{it_2(i)}|$
  end for
  select test point i such that $\Delta[i]$ is maximum
  if no capacity constraint is violated **then**
    allocate i to $t_1[i]$
  end if
end while

We recall that c_{ij}^A is the per-unit flow access cost between test point i and overlay node j, while w_{ik} represents the amount of flow that must be routed from test point i to test point k. The algorithm allocates the test points one by one according to a parameter $f_{ij} = \frac{c_{ij}^A}{\sum_{k \in I} w_{ik}}$, which takes into account both the allocation cost and the total amount of flow originating from the test point.

A test point is allocated to an overlay node if and only if the allocation does not violate the capacity constraint.

For each test point, not yet assigned, the overlay nodes are ordered according to non-decreasing value of f_{ij}. Then, for each test point i the first and the second overlay nodes in the order, $t_1(i)$ and $t_2(i)$, are considered. The test point with the maximum value of $|f_{it_1(i)} - f_{it_2(i)}|$ is selected and assigned to $t_1(i)$ if the capacity constraint is satisfied. The main idea behind this procedure is to assign first those test points which would suffer the greatest disadvantage in case they are not assigned to the closest overlay node.

Given as input the set of overlay nodes, the algorithm attempts to allocate all test points to these nodes. If it fails, the set is discarded as infeasible.

Diversification Step

In this step, the Cyclic Exchange Neighborhood of the local optimum determined in the optimization step is searched using the techniques described in Section 5.2.2, trying to find a new solution that is both far from such local optimum and cost-decreasing.

5.3.3 Post Optimization Step

Finally, in this step we perform a local search of the polynomial size allocation neighborhood (PSAN) of the best solution found by I-VLS-TS in order to improve the chosen assignments. More specifically, starting from the best solution found by the I-VLS-TS procedure, the PSAN is generated and then visited by the local search with the aim of finding a new solution that improves the objective function value.

5.4 Numerical Results

This Section tests the sensitivity of the proposed heuristic to different parameters like the number of candidate sites and test points, the traffic demands, the installation as well as the bandwidth costs. The performance of the exact and heuristic approaches are compared for small and medium-size network instances in terms of the obtained results and computing time.

To this end we consider both randomly generated network instances and real ISP topologies provided by Telecom Italia Lab (TILab). Random network topologies are obtained using the same topology generator already described in Section 4.1 (page 30), as well as BRITE, a degree-based generator, to produce power law topologies.

We assume that each TP can be connected to a CS only if the CS is at a distance not greater than r from the TP, with $r = 200$ and 400 for sparse and dense networks, respectively.

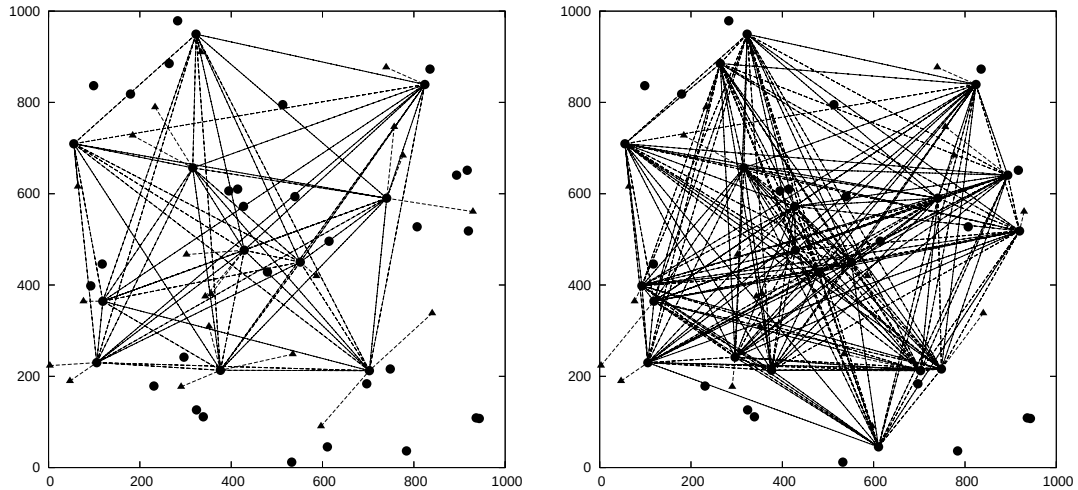
If not specified differently, the installation cost of an overlay node is equal to 10 monetary units. As for the access and egress cost, we assume they are fixed and equal to 1 monetary unit per Mb/s. The maximum capacity of an overlay node j (Γ_j) is set equal to 50 Mb/s for all $j \in S$.

All the results reported hereafter are the optimal and approximate solutions of the considered instances obtained, respectively, by formalizing the proposed model in AMPL [46] and solving it with CPLEX [47], or using the proposed heuristic approach. For each network scenario, the results are obtained averaging each point on 10 network instances.

a) Effect of the Traffic Demands: Random network instances

We first consider the SON design problem in a random network scenario with $n = 20$ TPs. Each test point offers the same amount of traffic w_{ik}.

Figure 5.1 reports an example of the planned networks when applying the MCSD model to the same instance with $m = 40$ candidate sites and with two different requirements on the end-user traffic, $w_{ik} = 500$ kb/s and $w_{ik} = 1$ Mb/s for all TPs. The Test Point coverage radius r is equal to 200. CSs and TPs are represented with circles and triangles, respectively. As expected, increasing the traffic demands forces the model to install a higher number of overlay nodes and links to convey the traffic towards the destinations.



(a) 500 kb/s (b) 1000 kb/s

Figure 5.1: Sample SONs planned by the MCSD model with increasing traffic demands (500 and 1000 kb/s). The number of TPs is 20, while the number of CSs is 40. The Test Point coverage radius r is equal to 200. CSs and TPs are represented with circles and triangles, respectively.

Table 5.2 analyzes the characteristics of the solutions of the MCSD model and the TS-MCSD heuristic in the same scenario when varying the number of candidate sites.

For each couple (m, w_{ik}) and optimization algorithm, the Table reports the number of installed overlay nodes (N_R) and links (N_L), the total network cost and the computing time (measured in seconds) to obtain the solution.

For small instances $(m = 30 - 80)$ we could obtain the exact solutions with the MCSD model, which enabled a comparison between the TS-MCSD heuristic and

optimal integer solutions; column gap reports the percentage gap between the cost provided by TS-MCSD and the optimal cost obtained with the MCSD model. This gap shows how close is TS-MCSD to the optimum obtained by the integer model, in terms of the objective function value. Note that solving the MCSD model for dense networks (i.e. $r = 400$), requires an extremely long computational time even for very small m values, so that only the proposed heuristic can be applied in practice.

Similarly to the results obtained with the models and heuristics introduced in Chapter 3, here we can observe that (1) the number of installed nodes and links increases when increasing the traffic demands, (2) the network cost decreases with m and (3) the proposed heuristic performs very close to the optimal solutions (less than 2% for $w_{ik} = 500$ kb/s and less than 4% for $w_{ik} = 1$ Mb/s) and the computation time is small for all the tested instances. Furthermore, we verified that in several network instances TS-MCSD solved the SON design problem to the optimum.

We then considered a variation of this network scenario with large-size instances containing $n = 100$ TPs.

The results obtained using the TS-MCSD heuristic are shown in Tables 5.3 and 5.4 for $r = 200$ and 400, respectively, with m ranging from 100 to 500 and for different w_{ik} values. These results are in line with the observations reported above. Since for $r = 400$ the solution space is larger than in the $r = 200$ case, TS-MCSD plans Service Overlay Networks with a lower cost.

b) Effect of the Traffic Demands: Power Law topologies

We now consider BRITE to generate power law topologies with 500 CSs and 20 TPs on a square area with edge equal to 1000; the Barabási $-$ Albert model [48] was adopted, as in Section 4.3 (c), with default parameters provided by BRITE and an average node degree equal to 10. Then, the bandwidth cost matrix was generated as for the custom topology generator.

Table 5.5 illustrates the numerical results obtained with such topologies, with an offered traffic w_{ik} equal to 500 and 1000 kb/s. The results confirm the behavior already observed for the random instances obtained with the custom generator, where the number of installed overlay nodes and links increases with increasing traffic demands.

c) Effect of the Bandwidth and Installation Costs

The impact of the bandwidth as well as the nodes' installation costs on the MCSD model are evaluated, considering a random network scenario with $n = 100$ TPs and $m = 100$ CSs. The offered traffic w_{ik} is equal to 50 kb/s and we consider dense

Table 5.2: Solutions provided by the TS-MCSD heuristic and the MCSD model, with 20 TPs and $r = 200$ (sparse networks).

w_{ik}=500 kb/s

	MCSD				TS-MCSD				
m	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap %
30	10.8	147.5	955.1	3.7	10.2	136.1	963.0	1.0	0.8
40	10.9	150.6	926.9	47.7	10.7	145.5	943.5	2.0	1.8
50	11.4	160.3	890.1	146.1	11.0	151.2	904.1	3.4	1.6
60	11.5	163.4	880.3	380.1	11.1	155.4	888.9	5.3	1.0
80	12.1	192.0	851.7	4785.3	11.5	161.1	867.7	11.3	1.9
100					11.8	170.1	846.0	21.3	
200					11.9	172.8	799.3	151.5	
500					12.3	182.0	763.9	2798.7	

w_{ik}=1000 kb/s

	MCSD				TS-MCSD				
m	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time	gap %
30	13.5	208.3	1847.2	10.7	13.7	215.4	1882.0	1.0	1.9
40	13.7	214.5	1772.3	96.0	13.8	218.6	1818.0	2.1	2.6
50	14.1	225.2	1683.6	357.8	13.8	218.8	1729.4	3.7	2.8
60	14.5	238.5	1668.7	5482.8	13.9	218.8	1715.2	5.9	2.8
80	14.0	222.6	1590.2	17003.0	13.8	218.8	1648.0	12.9	3.6
100					13.8	218.6	1603.6	23.7	
200					14.0	222.6	1511.5	177.1	
500					13.8	218.9	1404.7	3245.8	

Table 5.3: Large-size instances: solutions provided by the TS-MCSD heuristic, with 100 TPs and $r = 200$ (sparse networks).

	w_{ik}=50 kb/s				w_{ik}=100 kb/s			
m	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time
100	22.7	696.3	2159.2	217.4	27.6	938.1	4158.7	234.9
200	25.6	835.2	2069.9	959.7	30.2	1092.9	3951.6	1034.5
500	26.4	872.9	1980.9	1555.8	33.0	1264.3	3753.5	2600.4

networks with $r = 400$. The solution depends on β and δ, which are respectively, the ratio between the overlay nodes' installation cost and the bandwidth reservation

Table 5.4: Large-size instances: solutions provided by the TS-MCSD heuristic, with 100 TPs and $r = 400$ (dense networks).

	w_{ik}=50 kb/s				w_{ik}=100 kb/s			
m	N_R	N_L	Cost	Time	N_R	N_L	Cost	Time
100	20.5	600.9	2142.6	210.6	27.1	908.8	4145.2	260.1
200	22.2	678.3	2065.1	933.4	29.8	1066.6	3933.8	1180.0
500	24.8	799.2	1969.0	2144.7	32.1	1207.4	3748.5	2452.4

Table 5.5: Power Law topologies: solutions provided by the TS-MCSD heuristic in the topologies with node degree power laws generated using BRITE, with 500 CSs and 20 TPs.

w_{ik} (kb/s)	N_R	N_L	Cost	Time
500	4.0	52.0	823.6	1297.7
1000	10.0	130.0	2914.9	2493.4

cost, and the ratio between the cost of a link connecting two CSs which belong to different ISPs and that of a link connecting two CSs belonging to the same ISP.

The results obtained varying, respectively, the ratios β and δ are summarized in Tables 5.6 and 5.7, and they are in line with those reported in Section 4.3 (e). More in detail, it can be observed that (1) if β decreases, TS-MCSD tends to deploy more overlay nodes and (2) when δ increases, TS-MCSD tends to install less overlay nodes and links, thus diminishing the number of longer (and costlier) links that traverse different ISPs in the planned SON.

Table 5.6: Variable cost ratio β: solutions provided by the TS-MCSD heuristic with 100 TPs, 100 CSs, w_{ik}=50 kb/s and $r = 400$ (dense networks).

β	N_R	N_L	Cost	Time
10	20.5	600.9	2159.2	210.6
5	26.8	896.9	2041.5	233.8
1	33.9	1325.5	1914.6	255.4
$1/10$	36.3	1494.6	1880.0	256.4

Table 5.7: Variable cost ratio δ: solutions provided by the TS-MCSD heuristic with 100 TPs, 100 CSs, w_{ik}=50 kb/s and $r = 400$ (dense networks).

δ	N_R	N_L	Cost	Time
1	20.5	600.9	2159.2	210.6
2	14.2	390.9	3032.4	177.5
5	12.3	340.0	4870.0	179.0
10	11.7	326.1	7400.8	186.8

d) Real ISP Topologies

Finally, we considered three diverse ISP topologies which have been provided by Telecom Italia Lab (TILab). Table 5.8 lists, for each topology, the number of test points (n) and candidate sites (m), the overlay node installation cost (c_j^I) and capacity (Γ_j). All candidate sites have the same installation cost and the same capacity. The traffic demands, the access/egress costs and the transport costs are those provided by TILab for each network instance.

Table 5.9 shows the results obtained using the TS-MCSD heuristic on the TILab network instances. Note that in all such real scenarios the proposed heuristic solved the MCSD problem with almost negligible computing time, thus representing a practical solution for the planning of SONs.

Table 5.8: TILab Network instances: number of test points and candidate sites, overlay node installation cost and capacity.

Network	n	m	c_j^I	Γ_j
TILab1	15	12	20	2000
TILab2	39	17	100	500
TILab3	49	49	60000	10000

We then derived other instances by reducing the overlay nodes capacities by a factor of 0.6 and 0.8, and by reducing the installation costs by a factor of 0.1, 0.01 and 0.001. Tables 5.10 and 5.11 show the numerical results obtained in the three TILab topologies with, respectively, the overlay node capacity reduction of 0.6 and 0.8, and the installation cost reduction of 0.1, 0.01 and 0.001. For comparison reasons, in these Tables we also reported the numerical results obtained without applying any reduction factor (which are those already shown in Table 5.9).

Table 5.9: TILab topologies: solutions provided by the TS-MCSD heuristic.

Network	N_R	N_L	Cost	Time
TILab1	3.0	36.0	6412.3	0.1
TILab2	4.0	90.0	13612.9	0.8
TILab3	7.0	140.0	46590955.4	11.1

When the overlay nodes capacity decreases, the heuristic is forced to install more overlay nodes, and the total SON cost increases (see Table 5.10). On the other hand, when the overlay nodes installation cost decreases, TS-MCSD tends to install more overlay nodes (see Table 5.11, network topology TILab2), and the overall network cost decreases. Finally, note that also for such network instances the computing time of our proposed heuristic is very short.

Table 5.10: TILab topologies: solutions provided by the TS-MCSD heuristic. An overlay node capacity reduction of 0.6 and 0.8 has been applied. For comparison reasons, the results obtained without capacity reduction (i.e., capacity reduction factor $= 1$) are also reported.

Network	*Cap. Reduction Factor*	N_R	N_L	Cost	Time
TILab1	0.6	4.0	42.0	7261.1	0.2
TILab1	0.8	3.0	36.0	6538.3	0.2
TILab1	1	3.0	36.0	6412.3	0.1
TILab2	0.6	5.0	98.0	13691.9	1.0
TILab2	0.8	5.0	98.0	13691.9	1.1
TILab2	1	4.0	90.0	13612.9	0.8
TILab3	0.6	10.0	188.0	49374604.8	11.5
TILab3	0.8	8.0	154.0	47310633.0	13.2
TILab3	1	7.0	140.0	46590955.4	11.1

Table 5.11: TILab topologies: solutions provided by the TS-MCSD heuristic. The overlay nodes installation costs have been reduced by a factor of 0.1, 0.01 and 0.001. For comparison reasons, the results obtained without cost reduction (i.e., cost reduction factor $= 1$) are also reported.

Network	*Cost Reduction Factor*	N_R	N_L	Cost	Time
TILab1	1	3.0	36.0	6412.3	0.1
TILab1	0.1	3.0	36.0	6358.3	0.2
TILab1	0.01	3.0	36.0	6352.9	0.2
TILab1	0.001	3.0	36.0	6352.3	0.2
TILab2	1	4.0	90.0	13612.9	0.8
TILab2	0.1	10.0	168.0	13057.0	1.0
TILab2	0.01	10.0	168.0	12967.0	1.0
TILab2	0.001	10.0	168.0	12958.0	1.0
TILab3	1	7.0	140.0	46590955.4	11.1
TILab3	0.1	7.0	140.0	46241653.3	12.1
TILab3	0.01	7.0	140.0	46203853.3	12.1
TILab3	0.001	7.0	140.0	46200073.3	12.1

Chapter 6

Introduction to Game Theory

In this chapter we give a short introduction to Algorithmic Game Theory in order to provide the reader with the fundamental framework for understanding the next Chapter. A more extensive discussion can be found in [57].

6.1 Strategic Games

There are a lot of situations occurring in real life in which we seek for the maximization of our own benefit and very often the final outcome of our efforts also depends on the behavior of other people we have no control on. This is what is usually called a strategic game.

Definition 1 *A strategic game $\mathcal{G}$ consists of a set I of $k \geq 2$ players, where each player $i \in I$ has a (finite) set of strategies S_i and a payoff function J^i: $S_1 \times \ldots \times S_k \rightarrow \mathcal{R}^+$, for $1 \leq i \leq k$.*

One of the most famous strategic games is the Prisoner's Dilemma that we recall in the sequel for the sake of completeness.

EXAMPLE (Prisoner's Dilemma)

Two suspects in a major crime are held in separate cells. There is enough evidence to convict each of them of a minor offense, but not enough evidence to convict either of them of the major crime unless one of them acts as an informer against the other (finks). If they both stay quiet, each will be convicted of the minor offense and spend one year in prison. If one and only one of them finks, he will be freed and used as a witness against the other, who will spend four years in prison. If they both fink, each

will spend three years in prison. This situation can be modeled as a strategic game in which we have two players 1 and 2. The set of strategies is the same for both of them and is $S_i = \{Quiet, Fink\}$, for $i \in \{1, 2\}$. Finally, the payoff function is depicted in Table 6.1, in which the rows represent the first player strategies and the columns the second player ones.

Table 6.1: The Prisoner's Dilemma.

	Quiet	Fink
Quiet	(2,2)	(0,3)
Fink	(3,0)	(1,1)

Once defined a particular strategic game, the next step is that of characterizing the players' behavior. In other words, we have to determine which strategy will be chosen by each player in the game. Before settling out this question, it is necessary to fix some important points about the players' nature and how powerful is their decision-making process. The classical assumption is that each player acts rationally and selfishly, that is, he tries to maximize his own payoff and is not interested in cooperating with other players unless this can improve his payoff. It is also usually assumed that a player knows the strategy chosen by all the other players at any time.

Definition 2 *Given a game $\mathcal{G}$, a state s of $\mathcal{G}$ is an assignment of a strategy to each player. More formally, s is an element of the set $S = S_1 \times \ldots \times S_k$.*

The outcome of a strategic game is usually defined as a stable state, that is a state in which all players are satisfied with their payoff so that none of them is willing to change his chosen strategy.

6.2 Dominant Strategies

In any game, a player's strategy strictly dominates another strategy if it is superior, no matter what the other players do.

Definition 3 *In a strategic game with k players, we say that player i's strategy $s_i \in S_i$ strictly dominates his strategy $s'_i \in S_i$, if $J^i(s_1, \ldots, s_i, \ldots, s_k) \geq J^i(s_1, \ldots, s'_i, \ldots, s_k)$, for every strategy $s_j \in S_j$ adopted by any other player $j \in \{1, \ldots, k\}$, $j \neq i$. A dominant strategy for player i is a strategy $s_i \in S_i$ that strictly dominates any other strategy $s'_i \in S_i$.*

In the Prisoner's Dilemma, for example, the strategy *Fink* strictly dominates the strategy *Quiet*; regardless of his opponent's strategy, a player prefers the outcome when he chooses *Fink* to the outcome when he chooses *Quiet*. Since there are only two strategies, it follows that *Fink* is a dominant strategy. Clearly, because of his rational behavior, a player will always adopt a dominant strategy each time he has an available one. Thus, if each player has a dominant strategy, the outcome of the game will be the stable state in which each player adopts his dominant strategy. In particular, for the Prisoner's Dilemma the outcome will be $s = \{Fink, Fink\}$. In the sequel, we will call each such a state s, a Dominant Strategies Equilibrium.

6.3 Nash Equilibrium

The use of dominant strategies in order to characterize the outcome of a strategic game is perfectly reasonable and appropriate. The problem is that the existence of a dominant strategy for each player is a very strong property and the majority of the games we can think of does not satisfy it.

Starting from this point, John Nash in his Ph.D. thesis [58] elaborated his characterization of stable state in a strategic game by introducing his famous concept of equilibrium point.

Definition 4 *A state $s = (s_1, \ldots, s_i, \ldots, s_k)$ is a Nash Equilibrium (NE) if for every player i and strategy $s'_i \in S_i$ it holds that*
$J^i(s_1, \ldots, s_i, \ldots, s_k) \geq J^i(s_1, \ldots, s'_i, \ldots, s_k)$.

It can soon be noted the similarity of this definition with the one of dominant strategies. In effects, the strategy adopted by a player in any NE is a dominant strategy once fixed the strategy adopted by any other player. Hence, Nash introduced a weaker but still reasonable form of dominance thus making his notion of equilibrium more practical (in the sense that we can expect games not having a Dominant Strategies Equilibrium to possess a NE), but clearly not immune from criticisms. The first of them is that a NE is a stable state of a game as soon as we consider non-cooperative games, that is games in which no coalitions of players can be formed. In fact, the dominance condition required in a NE only involves one player at time, but nothing is said when two or more players jointly decide to choose a different strategy. Another important fact is that a game can admit more than just a single NE thus arising the problem of discriminating among them.

Finally, the adoption of NE as model of behavior does not settle out definitively the problem of determining the outcome of a strategic non-cooperative game since even existence of NE cannot be guaranteed in any game.

6.4 Algorithmic Game Theory

A very novel and fascinating research area is developing from the intersection of Game Theory with Theoretical Computer Science. In Christos Papadimitriou's words: "If the second half of the Twentieth Century has been devoted to the understanding of the power of the von Neumann's machine as a computational model, in the next years one of the major goals of our community will be that of providing a good mathematical formulation, understanding and analysis of the new millennium's computational model: the Internet". The main implication coming from this assumption is that now the way in which the classical optimization problems on networks can be addressed and solved has to be heavily reviewed. In non-cooperative networks, in fact, it is usually the case that the variables defining a given combinatorial problem are the representation of independent agents that may not be willing to implement the (optimal, approximate or competitive) solution computed with the classical techniques, for several reasons. It is well known that the lack of cooperation among the players in a game may result in suboptimal situations. To explain how we can define and measure the quality of a stable state, let us recall the Prisoner's Dilemma game. In the only stable state of the game (no matter if we use Dominant Strategies Equilibria or NE for its definition), $s = \{Fink, Fink\}$, both players get a payoff equal to 1, while if they cooperate to agree on the state $s' = \{Quiet, Quiet\}$ the payoff would be equal to 2 for both of them. This is a classical example showing that stable states fail in optimizing the social outcome of a game, where the social outcome of a game is defined here as the sum of the players'payoffs, but can be represented by any function $f : S \rightarrow \mathcal{R}^+$ defined over the set of states of the game and not necessarily related to the players' payoffs.

In practice, this means that the selfish behavior of the agents populating a non-cooperative system can result in a stable state whose social value may be far from the social optimum, thus resulting in an unbearable degradation of the system's performances. There are several approaches that has been developed in order to cope with the selfish behavior of the players in a game so as to keep the social outcome within some acceptable bounds, or simply to force players to act fairly and cooperatively while preserving their independence. All these techniques aim to design simple games so as their outcome will result in the desired one.

In spite of some of its faults, the classical theory of NE has been accepted and exploited worldwide to model the stable outcome of strategic games in non-cooperative networks and, at the same time, is being enriched with the notions of computational complexity and efficiency. Thus, in the main approaches followed in the literature by both Economists and Computer Scientists, stable states are usually modeled as NE. In what follows we give a brief overview of the main challenges posed by the study

of NE in non-cooperative strategic games.

6.4.1 Existence of Nash Equilibria

It is not difficult to convince oneself that the first concern one has to due with when approaching a non-cooperative strategic game is to prove that it admits at least one NE. In fact, if the game we are facing does not possess this basic property, no stable state can be reached by the players and so no further study on such a game can be considered plausible.

6.4.2 Computing Nash Equilibria

One of the fundamental aspects of NE is that they are stable states which can be reached by players independently by simply performing improving steps. But what about the problem of computing a NE? This is a very interesting computational problem since its complexity is still astonishingly open. On the other hand, if we are interesting in computing extreme NE, that is NE minimizing or maximizing the social function, they can be usually proved to be NP-hard to compute. In the case of our interest of computing NE, we observe that, if convergence is proved through a potential function Φ, then Φ can be used to implement an algorithm determining a NE by repeatedly computing for any state $s \in S$, the state $s' \in S$ such that $\Phi(s) - \Phi(s')$ is maximized. Unfortunately, such an algorithm cannot achieve in general a polynomial running time. For an interesting analysis on the complexity of determining NE, see [59]. It is also worth noting that, when the social function f we want to optimize is defined as the sum of all players' payoffs, any NE corresponds to a local optimum for the optimization problem having S as set of feasible solutions and f as objective function, and viceversa. Thus any NE can be seen as the solution returned by a local search algorithm defined by a 1-neighborhood function and the price of anarchy of the game becomes the approximation ratio of such an algorithm.

6.4.3 Bounding the prices of Anarchy and Stability

Once we know that a game possesses a NE an intriguing issue is that of determining the quality of a NE. In order to measure the loss of optimality that an unregulated system has to suffer because of the lack of coordination of his agents with respect to some social function, Koutsoupias and Papadimitriou [21] introduced the notion of price of anarchy of a game as the supremum over all NE of the ratio between the social value of a NE and the social optimal. More formally,

Definition 5 *Given a game $\mathcal{G}$ and a social function f, let $\mathcal{N}$ be the set of all NE of $\mathcal{G}$ and OPT be a state of $\mathcal{G}$ optimizing f. The price of anarchy $PoA_{\mathcal{G}}(f)$ of game $\mathcal{G}$ according to f is defined as*

$$PoA_{\mathcal{G}}(f) = \sup_{s \in \mathcal{N}} \frac{f(s)}{f(OPT)}$$

The price of anarchy is defined as a worst case evaluation of the social value of a NE.

In [14] the term price of stability has been introduced as the ratio between the best NE and the social optimum.

Definition 6 *Given a game $\mathcal{G}$ and a social function f, let $\mathcal{N}$ be the set of all NE of $\mathcal{G}$ and OPT be a state of $\mathcal{G}$ optimizing f. The price of stability $PoS_{\mathcal{G}}(f)$ of game $\mathcal{G}$ according to f is defined as*

$$PoS_{\mathcal{G}}(f) = \inf_{s \in \mathcal{N}} \frac{f(s)}{f(OPT)}$$

This notion measures the minimum loss in optimality that an unregulated system has to suffer. Its importance is strictly related to the problem of computing NE.

In fact, if it is possible to compute the best NE, say $s \in S$, in polynomial time, we can imagine a scenario in which a trusted authority suggests an initial choice to each player when entering the game so that the resulting state is exactly s. No player will have an improving step and so the loss in optimality will be exactly equal to the price of stability, that is the minimum possible one.

Chapter 7

Socially-Aware Overlay Network Design Games

In many scenarios, network design is not enforced by a central authority, but arises from the interactions of several self-interested agents. This is the case of the Internet, where connectivity is due to Autonomous Systems' choices, but also of overlay networks, where each user client can decide the set of connections to establish.

Recent works have used game theory, and in particular the concept of Nash Equilibrium, to characterize stable networks created by a set of selfish agents. The majority of these works assume that users are completely non-cooperative, leading, in most cases, to inefficient equilibria. However, this assumption is not entirely realistic, for example when network design involves long-term decisions. Moreover, incentives could be introduced by some external authority (e.g., the overlay administrator) in order to increase the users' cooperation level.

This Chapter overcomes this limitation by first proposing a novel network formation game, the Socially-Aware Network Design game, where users are partially socially-aware because their utility function is a weighted sum of individual and global costs. We study the efficiency of the equilibria achieved by our game, deriving bounds to the Price of Anarchy, the Price of Stability and the Reachable Price of Anarchy. These results suggest that the more sensitive users are to the global cost, the more efficient the best stable network is. However, we show that highly socially-aware users can also design networks much more expensive than purely selfish users.

For this reason, to eliminate worst-case scenarios while enabling at the same time more efficient Nash equilibria, we further propose a Stackelberg (leader-follower) approach, where a leader (e.g., the overlay network administrator) architects the desired network buying an appropriate subset of network's links, driving in this way the users to overall efficient Nash equilibria.

Finally, we measure the performance of the proposed schemes in several network scenarios, including realistic topologies where players build an overlay on top of real Internet Service Provider networks.

The Chapter is organized as follows: Section 7.1 introduces the proposed Socially-Aware Network Design game. Section 7.2 proposes a greedy algorithm that implements best response dynamics, which permits to reach a Nash equilibrium in the proposed game. Section 7.3 provides precise bounds on the Price of Anarchy, the Price of Stability and the Reachable Price of Anarchy for the SAND game. Section 7.4 describes the proposed Stackelberg game (the Network Administrator-Driven SAND game), which enables very efficient Nash equilibria. Finally, Section 7.5 presents numerical results that demonstrate the effectiveness of the SAND and NAD-SAND games in several realistic network scenarios.

7.1 The Socially-Aware Network Design Game

This Section illustrates the proposed Socially-Aware Network Design (SAND) game, motivating the reason to introduce such model.

The SAND game occurs in a directed graph $G = (V, E)$, where each edge e has a nonnegative cost c_e, and each player $i \in I = \{1, 2, \dots k\}$ is identified with a source-sink pair (s_i, t_i). Every player i picks a path S_i from its source to its destination, thereby creating the network $(V, \cup_i S_i)$ with total cost equal to $\sum_{e \in \cup_i S_i} c_e$, which will be referred to as *social* cost. We will refer to the path S_i also as the strategy chosen by player i.

This social cost is assumed to be shared among the players in the following way: let x_e denote the number of overlay paths that go through overlay link e; if edge e lies in x_e of the chosen paths, then each player choosing such an edge pays a proportional share $\pi_e = \frac{c_e}{x_e}$ of the cost.

The objective function J^i that user i wants to minimize is therefore given by:

$$J^i = \sum_{e \in S_i} \pi_e + \alpha \sum_{e \in \cup_j S_j} c_e \tag{7.1}$$

The first term takes into account the *selfish* nature of each player, since it is the cost for user i to buy the edges belonging to the chosen path, S_i; on the other hand, the second term represents the total network cost (i.e., the *social* cost), α being a parameter that permits to give more weight to one component with respect to the other.

An alternative interpretation of objective function (7.1) is also possible: we can think that users are completely selfish, but the social-aware term in cost function (7.1),

$\alpha \sum_{e \in \cup_j S_j} c_e$, is imposed by the network operator. The advantage of such approach is that the operator does not need to solve a large-scale Integer Linear Programming problem in order to optimize the routing choices for every change in the network (either in the topology, links cost, players location etc . . .), but can let the users solve the problem in a distributed way, converging to a good and stable solution.

Note that for $\alpha = 0$ objective function (7.1) corresponds to that of the Shapley network design game proposed in [14], which represents a particular case of our proposed game. Furthermore, in our game, users need only a limited amount of information, which is exactly equal to that of the Shapley network design game.

The SAND game belongs to the class of *potential games*, which has been identified in [60]. Such games are characterized by a real-valued function, Φ, on the strategy vector (S) which measures exactly the difference in the cost that any player saves if he is the only player to deviate. Mathematically, if we define:

- $S = (S_1, S_2, \ldots, S_k)$ the vector of players' strategies,
- $S'_i \neq S_i$ an alternate strategy for some player i,
- S_{-i} the strategy played by all other players than i, and
- $S' = (S'_i, S_{-i})$,

then a potential game with k players is characterized by a potential function, $\Phi(S)$, such that for any user i we have $\Phi(S) - \Phi(S') = J^i(S) - J^i(S')$.

Potential games have nice properties, such as existence of at least one pure Nash equilibrium, namely the strategy S that minimizes $\Phi(S)$ [57]. Furthermore, in such games, best response dynamics always converge to a Nash equilibrium.

It is easy to verify that the SAND game is characterized by the following potential function:

$$\Phi(S) = \sum_{e \in E} \sum_{x=1}^{x_e} \frac{c_e}{x} + \alpha \sum_{e \in \cup_j S_j} c_e. \tag{7.2}$$

We observe that the SAND game is a potential game without being at the same time a *congestion game* on the given graph G [61]. In fact, the cost incurred by player i, J^i, is not simply the sum of the costs incurred by such player from each link in the chosen path S_i, but it includes a further term, the total network cost (multiplied by the parameter α.)

Since we have introduced a term proportional to the network cost in the objective function of each player, we expect that players should design better networks, even though Section 7.3 shows that this is not necessarily the case. Before addressing this

issue, we present in the following Section a possible asynchronous users operation that implements best response dynamics and hence converges to a Nash equilibrium.

Comments

The SAND game can be easily extended to take into account performance metrics commonly used in routing problems in communication networks. For example, we can imagine that users are sensitive not only to link costs but also to the allocated rate on each link e, which we can assume is inversely proportional to the number of users (x_e) that pass through such link (like in *congestion games* [61]).

The extension can be performed adding a third term (a *congestion* cost) to cost function (7.1), which becomes as follows:

$$\widehat{J}^i = \sum_{e \in S_i} \pi_e + \alpha \sum_{e \in \cup_j S_j} c_e + \gamma \sum_{e \in S_i} \frac{x_e}{r_e} \tag{7.3}$$

where r_e is the maximum transmission rate of link e (i.e., the link's capacity), and γ is a weight that expresses the user's sensitivity to congestion costs. The rationale of adding $\gamma \sum_{e \in S_i} \frac{x_e}{r_e}$ to the cost function is that the congestion cost of using link e increases with the number of players that choose such link, since the allocated rate ($\frac{r_e}{x_e}$) decreases accordingly.

The extended SAND game is still a potential game, with the following potential function:

$$\widehat{\Phi}(S) = \sum_{e \in E} \sum_{x=1}^{x_e} \left(\frac{c_e}{x} + \gamma \frac{x}{r_e} \right) + \alpha \sum_{e \in \cup_j S_j} c_e. \tag{7.4}$$

Therefore, the existence of pure Nash equilibria is guaranteed, as well as the convergence of best response dynamics.

Finally, we note that the best response algorithm illustrated in the next Section can be extended straightforwardly to take into account the newly added congestion cost.

7.2 Best Response Algorithm

We now describe a simple algorithm that implements best response dynamics, allowing each user to improve its cost function in the proposed SAND game. Such algorithm, detailed in the following, is the best response strategy for a user minimizing objective function (7.1), assuming other users are not changing their strategies.

We still consider a directed graph $G = (V, E)$, where each edge e has a nonnegative cost c_e.

Let us consider a vector of players' strategies $S = (S_1, S_2 \dots S_k)$, where $S_i \subset E$ is a set of edges that connect the source-sink pair (s_i, t_i). Let us denote by $S^{-i} = \cup_{j \neq i} S_j$ the set of edges used by all players except player i; as a consequence, the set $E - S^{-i}$ contains all links that are not used by the set of all players except i. Finally, let k_e denote the number of times that edge e lies in the paths chosen by such players (i.e., all players except player i).

Algorithm 4 allows user i to determine the *best response* to all other users' strategies. The rationale behind such algorithm is very simple: the link weights are set equal to the cost incurred by user i to choose them, according to objective function (7.1). Then, the shortest path corresponds to the choice that minimizes the user's cost, and it represents therefore its best response. Note that, if multiple shortest paths exist with the same cost, the player chooses one of them randomly.

Algorithm 4 Pseudo-code specification for the best response dynamics for the SAND game.

1: **if** $e \in S^{-i}$ **then**
2: Set $Cost(e) = \frac{c_e}{k_e+1}$
3: **else**
4: Set $Cost(e) = c_e \cdot (1 + \alpha)$
5: **end if**
6: Compute the least-cost path (using for example the Dijkstra algorithm) with link costs $Cost(e)$
7: Set S_i equal to the set of links contained in the least-cost path

7.3 Bounds on the Price of Anarchy, Price of Stability and Reachable Price of Anarchy for the SAND game

In this Section, we derive bounds on the Price of Anarchy (PoA), the Price of Stability (PoS) and the so-called Reachable Price of Anarchy ($RPoA$) for the Socially-Aware Network Design game, and compare them with the results presented in [14] for the Shapley Network Design game. This allows us to determine the worst and best case performance of our proposed game.

Bound on the Price of Anarchy

We now establish a lower bound on the Price of Anarchy for the SAND game, which is defined as the ratio between the cost of the worst stable network (that with the highest cost), and the cost of the optimal network.

Proposition 1 *In the SAND game, a lower bound on the Price of Anarchy (PoA) is given by the following expression:*

$$PoA \geq k(1+\alpha). \tag{7.5}$$

Proof: Let us consider the simple network scenario of Figure 7.1 with two parallel links, one of cost equal to 1, the other with an arbitrarily high cost equal to C. Each of the k players must connect the common source node s to the common destination node t. In the optimal outcome, each player i chooses the lower link, with cost 1, and the cost of the formed network is obviously 1.

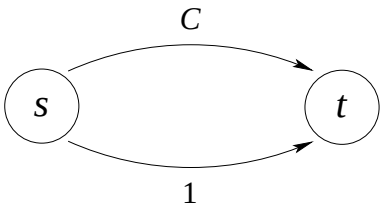


Figure 7.1: Two-link network topology: k players must connect the common source node s to destination node t.

However, this is not the only Nash equilibrium for the SAND game. Let us suppose that the initial network configuration sees all k users routed over the upper link, with cost C. It is easy to show that if $\alpha \geq \frac{C}{k} - 1$, no user has a gain to deviate and choose the link with cost equal to 1. Then, the cost of the network is C. Now if $C = k(1+\alpha)$, the above inequality is satisfied, and we obtain that the Price of Anarchy is at least $\frac{C}{1} = k(1+\alpha)$.

□

Bound on the Price of Stability

In the following we compute an upper bound on the Price of Stability for the SAND game, which is defined as the ratio between the cost of the best stable network (that with the lowest cost), and the cost of the optimal network.

Proposition 2 *In the SAND game, the Price of Stability (PoS) is upper bounded by the following expression:*

$$PoS \leq \frac{\mathcal{H}_k + \alpha}{1 + \alpha}. \tag{7.6}$$

Proof: It is shown in [14, 57] that the function $\Psi(S) = \sum_{e \in E} \sum_{x=1}^{x_e} \frac{c_e}{x}$ is an exact potential function for the Shapley network design game revised in Section 2.2, page 11. Furthermore, it is shown in [57] that such function satisfies the following inequalities:

$$\text{cost}(S) \leq \Psi(S) \leq \mathcal{H}_k \text{cost}(S) \tag{7.7}$$

where $\text{cost}(S) = \sum_{e \in \cup_j S_j} c_e$ and $\mathcal{H}_k = \sum_{i=1}^{k} 1/i$ is the kth harmonic number.

The potential function of the SAND game, $\Phi(S)$, is directly related to that of the Shapley network design game, $\Psi(S)$. In fact, from expression (7.2) it can be easily seen that $\Phi(S) = \Psi(S) + \alpha \cdot \text{cost}(S)$. Hence, if we replace $\Psi(S) = \Phi(S) - \alpha \cdot cost(S)$ in expression (7.7), it follows that:

$$(1 + \alpha)\text{cost}(S) \leq \Phi(S) \leq (\mathcal{H}_k + \alpha)\text{cost}(S). \tag{7.8}$$

Finally, Theorem 19.13 in [57] states that if a potential game with potential function $\Phi(S)$ satisfies the following inequality:

$$\frac{\text{cost}(S)}{A} \leq \Phi(S) \leq B\text{cost}(S) \tag{7.9}$$

with A and B positive constants, then the Price of Stability (PoS) is at most AB.

Therefore, in our problem, we obtain that the Price of Stability is at most $\frac{\mathcal{H}_k + \alpha}{1+\alpha}$. □

Bound on the Reachable Price of Anarchy

We now derive a bound on the so-called Reachable Price of Anarchy ($RPoA$), a quantity defined in [19]. The denominator of this ratio is the cost of the socially optimal network, which we denote by C^{opt}; the numerator is the largest cost of an equilibrium reachable via the following process: the k players enter the game one-by-one in an arbitrary order, and each picks a path of minimum cost (according to objective function (7.1)), given the choices of previous players. After all players have entered, the game proceeds exactly as for the SAND game, with each player re-optimizing its path, given the current strategies of all other players (using for example

the best response algorithm illustrated in the previous Section). When the process reaches a Nash equilibrium (as it must, since it is a potential game), it stops.

Proposition 3 *In the SAND game, the Reachable Price of Anarchy ($RPoA$) is upper bounded by the following expression:*

$$RPoA < k\frac{\alpha+1}{\alpha+\frac{1}{k}}. \tag{7.10}$$

Proof: Let S_i^{opt} define the path from source node s_i to destination node t_i for player i in the optimal outcome (the least cost network), and let $\overline{S_i}$ denote the path chosen by player i in its first move in the game. Let $C(S_i^{opt}) = \sum_{e \in S_i^{opt}} c_e$ and $C(\overline{S_i}) = \sum_{e \in \overline{S_i}} c_e$ denote the costs of such two paths.

For the first player that enters the network, the following inequality holds:

$$\sum_{e \in \overline{S_1}} \pi_e + \alpha \sum_{e \in \overline{S_1}} c_e = (1+\alpha)C(\overline{S_1}) < (1+\alpha)C(S_1^{opt}).$$

For the second player, the following inequality holds:

$$\sum_{e \in \overline{S_2}} \pi_e + \alpha C(\overline{S_1} \cup \overline{S_2}) < \sum_{e \in S_2^{opt}} \pi_e + \alpha C(\overline{S_1} \cup S_2^{opt})$$

which can also be expressed as follows, observing that $C(\overline{S_1} \cup S_2^{opt}) < C(\overline{S_1}) + C(S_2^{opt})$, and that $\sum_{e \in S_2^{opt}} \pi_e \leq C(S_2^{opt})$

$$\sum_{e \in \overline{S_2}} \pi_e + \alpha C(\overline{S_1} \cup \overline{S_2}) < (1+\alpha)C(S_2^{opt}) + \alpha C(S_1^{opt}).$$

It can be further observed that:

$$\sum_{e \in \overline{S_i}} \pi_e \geq \frac{1}{k} C(\overline{S_i}),$$

since in the best outcome, each of the links belonging to the path chosen by player i is shared by all k players.

If we proceed iteratively, we finally obtain the following inequality:

$$(\alpha + \frac{1}{k})C(\bigcup_{i=1}^{k} \overline{S_i}) < (1+\alpha)\sum_{i=1}^{k} C(S_i^{opt})$$

which can be rewritten as follows, observing that $\sum_{i=1}^{k} C(S_i^{opt}) \leq kC^{opt}$:

$$(\alpha + \frac{1}{k})C(\bigcup_{i=1}^{k} \overline{S_i}) < (1+\alpha)kC^{opt}.$$

This allows us to obtain the following result:

$$\frac{C(\bigcup_{i=1}^{k} \overline{S_i})}{C^{opt}} < k\frac{\alpha+1}{\alpha+\frac{1}{k}}. \tag{7.11}$$

Hence, the cost of the network reached after the first move of each of the k players, $C(\bigcup_{i=1}^{k} \overline{S_i})$, is no more than $k\frac{\alpha+1}{\alpha+\frac{1}{k}}$ times the optimal cost, C^{opt}. Since subsequent best response moves performed by players decrease the potential function value $\Phi(S)$ given by expression (7.2) (simulating a local search on it), the players will converge to a Nash equilibrium whose cost is still within $k\frac{\alpha+1}{\alpha+\frac{1}{k}}$ times the optimum.

Therefore, we can conclude that the Reachable Price of Anarchy is at most $k\frac{\alpha+1}{\alpha+\frac{1}{k}}$.

□

Comments

For $\alpha = 0$ the SAND game is equivalent to the original Shapley network design game. Indeed, expressions (7.5) and (7.6) confirm the results already demonstrated in [14]: $PoA = k$ and $PoS = \mathcal{H}_k$. In the SAND game, for increasing α values, the upper bound on the PoS decreases, and tends to 1 for $\alpha \to \infty$ (then also $PoS \to 1$). This can be easily explained since for $\alpha \to \infty$ the *social* component of objective function (7.1) is predominant. In this limiting behavior, all users share the same utility function, which is the second term of expression (7.1); in such a situation, the social network optimum (i.e., the network with the minimum total cost) is obviously also a Nash equilibrium, since the objective function of each single player coincides with the social network cost (multiplied by α).

On the other hand, our lower bound on the PoA increases when α increases. This corresponds to the (quite counter-intuitive) fact that, in some cases, more socially-aware users can design less efficient networks. This happens for example in the simple instance of Figure 7.1. Such inefficiency is due to the myopic decision criterion: each player only considers the effect of its own choice (i.e., changing the selected path), without considering eventual future decisions from the other players.

However, it can be observed from expression (7.10) that the $RPoA$ of the SAND game strictly decreases for increasing α values. For large α values (notably, for $\alpha \to \infty$), the $RPoA$ is less than k. Therefore, if the SAND game is played starting

from an empty network, worst-case scenarios like that illustrated in Figure 7.1 are eliminated.

Finally, our simulations show that the Nash equilibria reached in the SAND game are, in average, consistently better than those achieved by the Shapley game, as we will show in Section 7.5.

7.4 The Network Administrator-Driven Socially-Aware Network Design game

We now illustrate a variation of the SAND game, named the Network Administrator-Driven SAND (NAD-SAND) game, where a network administrator plays before the users, and his aim is to drive them to the best Nash equilibrium possible.

Since computing the optimal Stackelberg strategy is NP-hard, we present in this Chapter a simple strategy that achieves consistent performance improvements. Such approach is implemented via the following heuristic:

1. Given the network topology, the network administrator solves a generalized Steiner Tree problem [62], determining the minimum-cost subnetwork such that the source/destination nodes of each player are connected by a path. Let E^{opt} be the set of edges belonging to such optimal subnetwork.

2. The network administrator chooses all links belonging to E^{opt}, thus offering to share eventually their cost with the other players. Therefore, using the notation introduced in Section 7.1, after this step we have $x_e = 1, \forall e \in E^{opt}$ (that is, the network administrator has already chosen all links that are *optimal* from a social point of view).

3. At this point, all the k users play the SAND game, each trying to optimize its own objective function, which is the same of expression (7.1).

The reader is referred to Appendix A.2 for a brief overview of the Steiner Tree problems in networks.

The rationale behind the proposed NAD-SAND game is the following: the network administrator tries to motivate all players to use the links that belong to the socially optimal solution by sharing their cost with network users. We will show in the next Section that such heuristic is very effective, and permits to obtain dramatic performance improvements with respect to the SAND game.

We observe that the first step of the NAD-SAND game involves solving an NP-Complete problem. However, several efficient heuristics and approximation algorithms have been proposed to solve such problem in a reasonable computation time [62,

63, 64]. In the numerical results presented in the next Section we were able to compute exactly the minimum cost generalized Steiner tree using a simple ILP formulation, and solving it with the CPLEX solver [47].

Finally, we observe that as $\alpha \to \infty$, the NAD-SAND game always reaches the minimum cost network since for each player the cost of choosing any link that does not belong to the minimum-cost subnetwork (i.e., to E^{opt}) has an exceedingly large cost. As a consequence, $PoA \to 1$ as $\alpha \to \infty$.

We now derive a lower bound on the Price of Anarchy for the NAD-SAND game.

Proposition 4 *In the NAD-SAND game, a lower bound on the Price of Anarchy (PoA) is given by the following expression:*

$$PoA \geq \frac{k}{2(1+\alpha)} \tag{7.12}$$

for $\alpha \leq \frac{k}{2} - 1$.

Proof: Let us consider the network scenario illustrated in Figure 7.2, where each of the k players must connect the source node s_i to the common destination node t. Link $\nu \to t$ has a cost equal to C, with $2 \leq C \leq k$.

The minimum cost generalized Steiner Tree is in this case formed by all links $s_i \to \nu$ (with cost zero) and by link $\nu \to t$, so that its cost is equal to C. Hence, the network administrator will choose all these links, which constitute the E^{opt} set.

In the optimal outcome, each player i chooses the $s_i \to \nu \to t$ path, and the cost of the formed network is C.

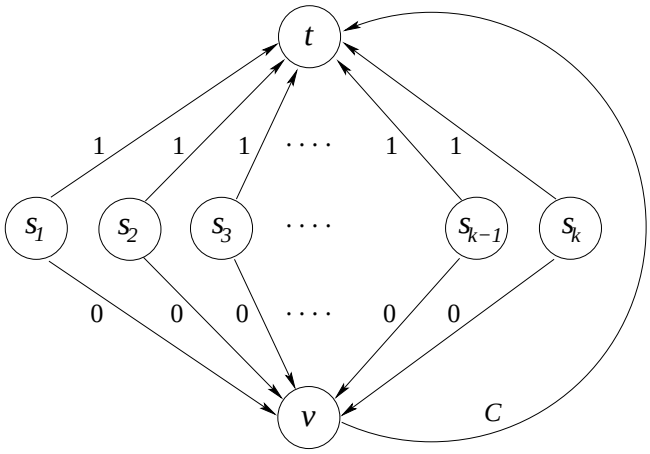


Figure 7.2: Parallel link topology: bound on the Price of Anarchy for the NAD-SAND game.

The NAD-SAND game, however, admits another Nash equilibrium. Let us suppose that initially all k users are routed over the direct path, $s_i \rightarrow t$, with total cost k. It can be easily seen that if $\alpha \leq \frac{C}{2} - 1$, no user has a gain to deviate and choose the $s_i \rightarrow \nu \rightarrow t$ path, with cost equal to C.

Then, the cost of the network is k. Now if $C = 2(1 + \alpha)$, the above inequality is satisfied, and we obtain that the Price of Anarchy is at least $\frac{k}{C} = \frac{k}{2(1+\alpha)}$.

□

7.5 Numerical Results

In this Section, we report the results obtained by the proposed Socially-Aware Network Design (SAND) and Network Administrator - Driven SAND (NAD-SAND) games in random network topologies as well as in real ISP topologies, comparing them both to the Shapley network design game [14] and to the social optimum. To this end, we consider both randomly generated network instances and real ISP topologies mapped by the Rocketfuel tool [40, 41]. Random networks are obtained using a custom generator as well as a degree-based generator (BRITE [44, 45]) to create topologies with a power law distribution of node degrees.

In all cases, we start from the empty network and we apply iteratively the best response algorithm illustrated in Section 7.2, until a Nash equilibrium is reached.

For each scenario we report the total network cost obtained by the proposed games for different α values, as well as the optimal network cost (the ILP column), obtained formulating the generalized Steiner Tree problem [62] with an Integer Linear Program (ILP), using AMPL [46], and solving it with the CPLEX solver [47]. Solving such problem provides the least-cost network topology that connects all source/destination pairs, thus representing a term of comparison for the efficiency of the equilibria reached by our proposed games.

Finally, we evaluated numerically the Price of Anarchy and the Price of Stability in all our simulation settings, and we found that they are, respectively, always less than 1.78 and 1.56, hence consistently lower than the bounds provided in Section 7.3.

Full-Mesh Topologies

We first consider full-mesh network topologies with 50 nodes randomly distributed on a 1000×1000 square area and 20 players (source/destination pairs). The cost of each link is equal to its length, and the numerical results, averaged over 20 random extractions, are reported in Table 6.1.

The SAND game permits to obtain network equilibria with a cost significantly

lower (approximately 13%) than that obtained with the Shapley network design game (i.e., the SAND row with $\alpha = 0$), even though there is still room for improvements, as demonstrated by the gap existing between the SAND game and the optimal cost (the ILP column).

This gap is filled by the NAD-SAND game, which achieves consistently cheaper Nash equilibria for all α values (up to more than 30%), including the $\alpha = 0$ case, thus representing a very effective approach also when applied to the original Shapley network design game. Furthermore, when α is sufficiently large (≥ 10), the NAD-SAND game reaches exactly the optimum (i.e., the generalized Steiner Tree cost).

Table 7.1: Full-Mesh topology with 50 nodes randomly distributed on a 1000 $\times$ 1000 area and 20 players: average network costs for the SAND and the NAD-SAND games. The optimal network cost is also reported. The cost of each link is set equal to its length.

Game	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 50$	$\alpha = 100$	$\alpha = 1000$	**ILP**
SAND	6956.17	6124.62	6037.53	6043.82	6050.94	6046.66	**4214.63**
NAD-SAND	5718.86	4707.48	4214.63	4214.63	4214.63	4214.63	

Random Topologies

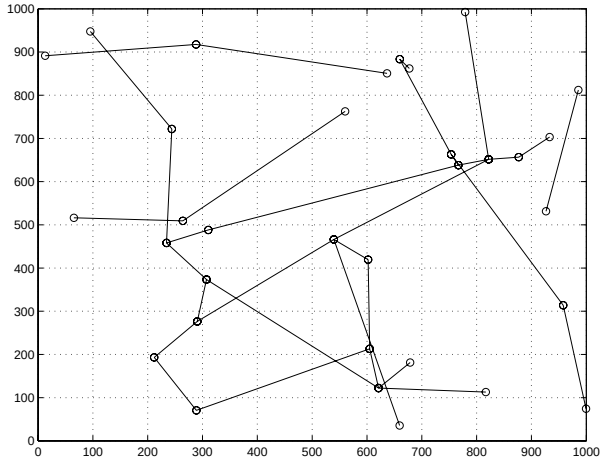
To generate random network instances, we have implemented a topology generator which considers a square area with edge equal to 1000, and randomly extracts the position of N nodes, uniformly distributed on the square area.

As for the network links, which can be bought by players to connect their end-points, we consider two alternative choices:

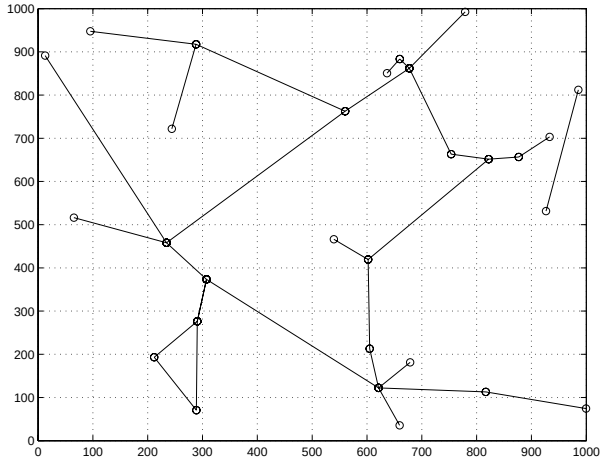
- Random geometric graphs: links exist between any two nodes located within a range R. The link cost is set equal to its length.
- Random network model "Uniform": a given number of links, L, is generated between randomly extracted couples of nodes. The cost of each link is uniformly distributed in the 0 to C range (C being the maximum link cost).

Given a random network, 20 random selections of $k = 20$ source/destination couples are considered.

Figure 7.3 shows the networks resulting at the Nash equilibria with the SAND game in a random geometric graph scenario with 50 nodes, range $R = 500$, 20 source/destination pairs, and two different α values (viz., $\alpha = 0$ and $\alpha = 50$).



(a) $\alpha = 0$



(b) $\alpha = 50$

Figure 7.3: Nash equilibria obtained by the SAND game in a random geometric network with 50 nodes, $R = 500$, 20 source/destination pairs, and different α values ($\alpha = 0$ and $\alpha = 50$).

We observe that for $\alpha = 50$ the topology is much closer to a tree-like topology than that obtained by the Shapley network design game ($\alpha = 0$). This is reflected in the total network cost, which is equal to 7046.3 for $\alpha = 0$ and to 5336.2 for $\alpha = 50$, thus resulting in a gain of more than 24%.

Table 6.2 illustrates the results obtained in the same random network scenario of Figure 7.3, with 50 nodes, range $R = 500$ and 20 source/destination pairs (players). The results are averaged on 20 source/destination random selections, and also on 5

random topologies. The Table reports the total cost of the network planned by the SAND and NAD-SAND games, as well as the optimal network cost. Also in this scenario, the SAND game achieves improved equilibria with respect to the Shapley network design game (up to 13.2%). The NAD-SAND game outperforms the SAND game, further decreasing the planned network cost, and reaches, for $\alpha \geq 10$, the social optimum.

Table 7.2: Random geometric graphs; random networks with 50 nodes, $R = 500$ and 20 players: average network costs for the SAND and the NAD-SAND games. The optimal network cost is also reported.

Game	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 50$	$\alpha = 100$	$\alpha = 1000$	**ILP**
SAND	6567.73	6074.57	5708.95	5724.18	5736.17	5706.09	**4213.82**
NAD-SAND	5645.44	4675.13	4213.82	4213.82	4213.82	4213.82	

Table 6.3 shows the numerical results obtained using Random network model "Uniform" in networks with 100 nodes, $L = 2000$ links, maximum link cost $C = 100$ and 20 source/destination pairs. The total network costs are illustrated in the Table for both the SAND and NAD-SAND games, while the ILP column reports the optimal network cost. Although there is still room for improvement, in this scenario the SAND game approaches the ILP bound, and consistently improves the quality of network equilibria with respect to the $\alpha = 0$ case.

Also in this case, the NAD-SAND game performs better than the SAND game, lowering the overall network cost of more than 11%, reaching the socially optimal outcome for $\alpha \geq 50$.

Table 7.3: Random network model "Uniform"; random networks with 100 nodes, 2000 links, maximum link cost $C = 100$, 20 players: average network costs for the SAND and the NAD-SAND games. The optimal network cost is also reported.

Game	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 50$	$\alpha = 100$	$\alpha = 1000$	**ILP**
SAND	314.69	294.18	296.38	295.09	295.57	296.48	**263.25**
NAD-SAND	295.85	274.38	263.70	263.25	263.25	263.25	

Power Law topologies

We further considered BRITE, a degree-based topology generator, which was used to generate power law topologies; the Barabási − Albert model was adopted with default parameters provided by BRITE.

We generated two different power-law network scenarios: one with 100 nodes and average node degree equal to 3, which contained about 300 links; the other with 300 nodes and average node degree equal to 5, which contained about 1500 links. The cost of each link was set equal to its length.

Table 6.4 reports the corresponding numerical results, including the optimal network cost obtained solving the ILP model.

In this case, the Nash equilibria reached by the Shapley network design game are already close to the optimum. This is mainly due to the fact that power law topologies, which result in short characteristic path lengths and heavy clustering, are the result of social concerns that eventually bring efficiency to the network design (even though not explicitly expressed in the underlying routing protocols design).

The SAND and NAD-SAND games, however, permit to improve the efficiency of such equilibria, finally obtaining the minimum cost network provided by the ILP model.

Table 7.4: Power law topologies generated using BRITE: average network costs for the SAND and the NAD-SAND games. The optimal network cost is also reported.

	100 nodes						
Game	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 50$	$\alpha = 100$	$\alpha = 1000$	**ILP**
SAND	15980.20	15684.28	15596.31	15593.69	15593.43	15602.21	**15314.14**
NAD-SAND	15479.98	15314.14	15314.14	15314.14	15314.14	15314.14	

	300 nodes						
Game	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 50$	$\alpha = 100$	$\alpha = 1000$	**ILP**
SAND	22580.82	22312.27	22224.65	22210.97	22210.55	22210.55	**21927.41**
NAD-SAND	22052.06	21987.68	21927.41	21927.41	21927.41	21927.41	

Real ISP topologies

Finally, we consider three real ISP topologies mapped using Rocketfuel [40, 41], listed in Table 6.5, with an increasing number of nodes and links. Table 6.6 shows the results obtained in such topologies, that is, the total network costs as well as the optimal network costs obtained solving the ILP model. The link costs are those provided by Rocketfuel, and we performed 20 random selections of 20 source/destination pairs.

In the small-size Telstra topology, the equilibria found by the SAND game (as well as those of the Shapley network formation game) are very close to the optimum,

Table 7.5: Rocketfuel-inferred ISP topologies: number of network nodes and links.

Network	Location	Nodes	Links
Telstra	AU	108	306
Sprintlink	US	141	748
Abovenet	US	315	1944

which is reached by the NAD-SAND game for $\alpha \geq 50$. As for the other ISP topologies, the SAND game diminishes the total network costs, and approaches the optimal ILP solution, while the NAD-SAND game always plans cheaper networks, achieving the optimal outcome already for relatively small α values.

Table 7.6: Rocketfuel topologies, 20 source/destination pairs: average network costs for the SAND and the NAD-SAND games. The optimal network cost is also reported.

Network	Game	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 50$	$\alpha = 100$	$\alpha = 1000$	**ILP**
Telstra	SAND	165.55	164.55	163.15	163.10	163.10	163.10	**157.95**
	NAD-SAND	163.25	160.70	158.55	157.95	157.95	157.95	
Sprintlink	SAND	286.25	282.60	280.90	280.30	280.30	280.30	**255.60**
	NAD-SAND	270.30	263.35	256.85	255.65	255.60	255.60	
Abovenet	SAND	301.30	292.65	287.45	287.45	287.05	287.05	**261.55**
	NAD-SAND	282.95	272.50	262.10	261.55	261.55	261.55	

A final interesting point that we would like to mention is the resilience of the proposed network design game solutions to link failures. In particular, for the three ISP scenarios described above we individuated the most congested link (i.e., the one used by the largest number of players), dropped it and recomputed new equilibria. It turns out that the new network costs in all the cases are at most 2% higher than the costs of the original networks.

Chapter 8

Conclusion

8.1 Discussion and Concluding Remarks

In this thesis, we addressed two alternative approaches for the design of overlay networks: (1) a centralized network optimization approach, which is based on the Service Overlay Network (SON) paradigm, and (2) a fully distributed approach, where the overlay network is designed by a large number of independent actors who seek to selfishly optimize their own utility.

We first investigated the topology design problem for Service Overlay Networks in terms of deciding the number and location of the overlay nodes to be deployed, the assignment of users to access overlay nodes, the traffic routing, the capacity reserved on each overlay link as well as the optimal subset of end-users to be covered in order to maximize the SON operator's profit.

To this end, we proposed efficient SON design models, based on mathematical programming, which take into account the individual requirements of the end-users, the connectivity between overlay nodes and the management of the traffic flows. Our proposed models can solve two different problems: the minimization of the overall network installation cost while ensuring full coverage of all end-users, and the maximization of the SON profit by choosing which users to serve based on the expected gain and budget constraints specified by the SON operator.

Furthermore, we introduced a broad set of efficient heuristics that obtain near-optimal solutions for large-scale instances in a reasonable computation time.

As for the fully distributed approach, in this thesis we also proposed two novel socially-aware network design games. In the first game we incorporated a socially-aware component in the users' utility functions, while in the second game we used additionally a Stackelberg (leader-follower) approach to improve efficiency.

We provided numerical results of the proposed models and heuristics for the SON

design problem on a set of realistic and large-scale instances, and discussed the effect of different parameters on the characteristics of the planned networks. We showed that in the considered network scenarios the proposed heuristics perform close to the optimum with a short computing time.

We further measured the performance of the proposed overlay network formation games in several network scenarios, including real ISP topologies, and we showed how they outperform classical network formation games (like the Shapley network design game), often obtaining the socially optimal outcome.

Numerical results demonstrated that (1) introducing some incentives to make users more socially-aware is an effective solution to achieve stable and efficient networks in a distributed way, and (2) the proposed Stackelberg approach permits to achieve dramatic performance improvements, obtaining almost always the socially optimal network, which could be designed by a central authority. Hence, we conclude that the proposed approaches can be very effective to design efficient overlay networks.

8.2 Future Research Issues

Several research issues are worth pursuing, and are discussed in the following:

- As for the proposed VLSN search heuristic, we observe that in our work, given a local optimum obtained either using local or tabu search, we applied only one iteration of VLSN to obtain a new solution that is both cost-decreasing and far from the local optimum. It may be interesting to investigate the effect of performing several iterations of VLSN on the local optimum obtained by the local/tabu search instead of one iteration, in a variable neighborhood search-like way. Another future issue could be to incorporate strategic oscillations [65, 66] into the local search and tabu search approaches accepting also non-feasible solutions in order to correct greedy bad behaviors. Within this framework, one promising issue can be to use VLSN for computing feasible allocation solutions. Finally, it may be interesting to use the so-called *Variable Neighborhood Search* [67, 68], which consists in starting from a given solution and applying iteratively two procedures (*shaking* and *local search*) until a given stopping condition is satisfied. The shaking procedure aims to give a reasonable initial solution while local search attempts to improve the initial solution yielding a local optimum.

- As for the distributed overlay network design approach, it would be interesting to (1) investigate the coalition formation game among overlay users and (2)

test the proposed network formation games in real testbed scenarios (like for example the One-Lab platform) in order to gather a feedback on real network behavior.

References

[1] Z. Duan, Z.-L. Zhang, and Y. Hou, "Service overlay networks: SLAs, QoS, and bandwidth provisioning," *IEEE/ACM Transactions on Networking*, pp. 870–883, vol. 11, no. 6, December 2003.

[2] Z. Li and P. Mohapatra, "QRON: QoS-aware Routing in Overlay Networks," *IEEE Journal on Selected Areas in Communications*, pp. 29–40, vol. 22, no. 1, January 2004.

[3] L. Subramanian, I. Stoica, H. Balakrishnan, and R. H. Katz, "OverQoS: Offering Internet QoS Using Overlays," in *Proceedings of the 1st Workshop on Hot Topics in Networks HotNets-I*, October 2002.

[4] J. Touch and S. Hotz, "The X-Bone," in *Proceedings of the third Global Internet Mini-Conference*, pp. 75–83, 1998.

[5] H. Tran and T. Ziegler, "A design framework towards the profitable operation of service overlay networks," *Computer Networks*, pp. 94–113, vol. 51, 2007.

[6] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," in *IETF RFC 3031*, January 2001.

[7] S. L. Vieira and J. Liebeherr, "Topology Design for Service Overlay Networks with Bandwidth Guarantees," in *the 12th IEEE International Workshop on Quality of Service, IWQoS*, pp. 211–220, Montreal, Canada, June 2004.

[8] Z. Li and P. Mohapatra, "On investigating overlay service topologies," *Computer Networks*, pp. 54–68, vol. 51, 2007.

[9] J. Han, D. Waston, and F. Jahanian, "Topology Aware Overlay Networks," in *Proc. IEEE Infocom'05*, Miami, FL, 13-17 March 2005.

[10] J. Fan and M. Ammar, "Dynamic Topology Configuration in Service Overlay Networks: A Study of Reconfiguration Policies," in *Proceedings IEEE Infocom'06*, Barcelona, Spain, April 2006.

[11] S. Shi and J. Turner, "Placing servers in overlay networks," in *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS) 2002*, San Diego, CA, July 2002.

[12] B. Vleeschauwer, F. Turck, B. Dhoedt, and P. Demeester, "On the construction of QoS enabled overlay networks," in *Proceedings of the Fifth International Workshop on Quality of future Internet Services (QofIS04)*, pp. 164–173, Barcelona, Spain, October 2004.

[13] S. Roy, H. Pucha, Z. Zhang, Y. Hu, and L. Qiu, "Overlay Node Placement: Analysis, Algorithms and Impact on Applications," in *Proceedings of the 27th International Conference on Distributed Computing Systems*, Toronto, Canada, June 2007.

[14] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden, "The price of stability for network design with fair cost allocation," in *Proc. of the 45th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 295–304, Rome, Italy, October 17-19, 2004.

[15] E. Anshelevich, A. Dasgupta, E. Tardos, and T. Wexler, "Near-optimal network design with selfish agents," in *Proc. of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 511–520, San Diego, CA, USA, 2003.

[16] H.-L. Chen and T. Roughgarden, "Network design with weighted players," in *Proc. of the 18th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA'06)*, Cambridge, MA, USA, July 30-August 2, 2006.

[17] S. Albers, "On the value of coordination in network design," in *Proc. of the 19th annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pp. 294–303, San Francisco, CA, USA, 2008.

[18] A. Epstein, M. Feldman, and Y. Mansour, "Strong equilibrium in cost sharing connection games," in *Proc. of the 8th ACM conference on Electronic commerce*, pp. 84–92, San Diego, CA, USA, June 2007.

[19] J. Chen, "A note on solution of the capacitated single allocation hub location problem," *International Journal of Applied Management Science*, vol. 1, no. 2, no. 2, pp. 198–216, 2008.

[20] G. Smaragdakis, V. Lekakis, N. Laoutaris, A. Bestavros, J. Byers, and M. Roussopoulos, "EGOIST: overlay routing using selfish neighbor selection," in *Proceedings of the 2008 ACM CoNEXT Conference*, ACM New York, NY, USA, 2008.

[21] E. Koutsoupias and C. Papadimitriou, "Worst-case equilibria," in *Proc. of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, pp. 404–413, Trier, Germany, 4-6 March 1999.

[22] R. Boorstyn and H. Frank, "Large-Scale Network Topological Optimization," *IEEE Transactions on Communications*, pp. 29–47, vol. 25, no. 1, January 1977.

[23] M. Pioro and D. Medhi, *Routing, Flow, and Capacity Design in Communication and Computer Networks*. Morgan Kaufmann Publishers, 2004.

[24] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection," in *Proceedings IEEE Infocom'02*, pp. 1190–1199, vol.3, New York, NY, June 2002.

[25] A. Kershenbaum, P. Kermani, and G. Grover, "MENTOR: an algorithm for mesh network topological optimization and routing," *IEEE Transactions on Communications*, pp. 503–513, vol. 39, no. 4, April 1991.

[26] A. Hills, "Large-Scale Wireless LAN Design," *IEEE Communications Magazine*, pp. 98–107, vol. 39, no. 11, November 2001.

[27] E. Amaldi, A. Capone, M. Cesana, and F. Malucelli, "Optimization Models for the Radio Planning of Wireless Mesh Networks," in *Proceedings of Networking 2007*, Atlanta, GA, USA, 14-18 May 2007.

[28] L. Zhou and A. Sen, "Topology Design of Service Overlay Network with a Generalized Cost Model," in *Proceedings of IEEE Global Telecommunications Conference, GLOBECOM*, pp. 75–80, November 2007.

[29] A. Sen, L. Zhou, B. Hao, B. Shen, and S. Ganguly, "On Topological Design of Service Overlay Networks," in *Proceedings of the thirteenth International Workshop on Quality of Service, IWQoS*, pp. 54–68, 2005.

[30] S. Shi and J. Turner, "Multicast routing and bandwidth dimensioning in overlay networks," *IEEE Journal on Selected Areas in Communications*, pp. 1444–1455, vol. 20, no. 8, October 2002.

[31] D. Pompili, C. Scoglio, and L. Lopez, "Multicast algorithms in service overlay networks," *Computer Communications, accepted for publication, August 2007*.

[32] C. Yeo, B. Lee, and M. Er, "A survey of application level multicast techniques," *Computer Communications*, pp. 1547–1568, vol. 27, no. 15, September 2004.

[33] S. Fahmy and M. Kwon, "Characterizing Overlay Multicast Networks and their Costs," *IEEE/ACM Transactions on Networking*, pp. 373–386, vol. 15, no. 2, April 2007.

[34] R. Ahuja, Ö. Ergun, J. Orlin, and A. Punnen, "A survey of very large-scale neighborhood search techniques," *Discrete Applied Mathematics*, pp. 75–102, vol. 123, 2002.

[35] R. Ahuja, J. Orlin, and D. Sharma, "Very large-scale neighborhood search," *International Transactions in Operational Research*, pp. 301–317, vol. 7, 2000.

[36] L. Hogg and N. Jenning, "Variable sociability in agent-based decision making," *Lecture Notes in Computer Science*, pp. 305–318, vol. 1757/2000, 2000.

[37] A. Azad, E. Altman, and R. El-Azouzi, "From altruism to non-cooperation in routing games," *INRIA Technical Report*, 18 October 2008.

[38] H. Hamacher, M. Labbé, S. Nickel, and T. Sonneborn, "Adapting polyhedral properties from facility to hub location problems," *Discrete Applied Mathematics*, vol. 145, no. 1, no. 1, pp. 104–116, 2004.

[39] R. Raghavan and C. Thompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, pp. 365–374, vol. 7, no. 4, 1987.

[40] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *Proc. of ACM SIGCOMM 2002*, Pittsburgh, PA, USA, August 2002.

[41] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," in *Proc. of the 2nd Internet Measurement Workshop*, Marseille, France, November 2002.

[42] "GT-ITM: Modeling Topology of Large Internetworks." Available at http://www.cc.gatech.edu/projects/gtitm/.

[43] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an Internetwork," in *Proceedings IEEE Infocom'96*, pp. 594–602, vol.2, San Francisco, CA, March 2006.

[44] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: an approach to universal topology generation," in *Proc. of MASCOTS 2001*, Cincinnati, OH, USA, August 2001.

[45] A. Medina, I. Matta, and J. Byers, "On the origin of power-laws in Internet topologies," *ACM Communications Review*, pp. 18–28, vol. 30, no. 2, April 2000.

[46] "AMPL: a modeling language for mathematical programming." available at http://www.ampl.com.

[47] ILOG Optimization Products, "ILOG CPLEX." available at http://www-01.ibm.com/software/integration/optimization/cplex/.

[48] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, pp. 509–512, vol. 286, no. 5439, 1999.

[49] G. Carello, "Hub Location Problems in Telecommunication Networks." PhD thesis, Politecnico di Torino, Italy, February 2004.

[50] R. Ahuja, J. Orlin, S. Pallottino, M. Scaparra, and M. Scutellà, "A multi-exchange heuristic for the single source capacitated facility location problem," *Management Science*, pp. 749–760, vol. 50, no. 6, June 2004.

[51] R. Ahuja, J. Orlin, and D. Sharma, "Multi-exchange neighborhood search algorithms for the capacitated minimum spanning tree problem," *Mathematical Programming*, pp. 71–97, vol. 91, 2001.

[52] P. Thompson and J. Orlin, "Theory of cyclic transfers." Working paper, Operations Research Center, MIT, 1989.

[53] G. Carello, F. D. Croce, M. Ghirardi, and R. Tadei, "Solving the Hub location problem in telecommunication network design: A local search approach," *Networks*, pp. 94–105, vol. 44, no. 2, 2004.

[54] S. Martello and P. Toth, *Knapsack Problems - Algorithms and Computer Implementations*. Wiley and Sons, 1990.

[55] F. Glover, "Tabu Search. Part I," *Orsa Journal on Computing*, pp. 190–206, vol. 1, 1989.

[56] F. Glover, "Tabu Search. Part II," *Orsa Journal on Computing*, pp. 4–32, vol. 2, 1990.

[57] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, "Algorithmic game theory," *Chapter 19, Network Formation Games*, pp. 487–516, Cambridge University Press, 2007.

[58] J. NASH, "Equilibrium points in n-person games," in *Proceedings of the National Academy of Sciences*, pp. 4849, vol. 36, 1950.

[59] A. Fabrikant, C. Papadimitriou, and K. Talwar, "On the complexity of pure equilibria," in *Proc. 37th Annuals Symp. on Theory of Computing (STOC)*, 2004.

[60] D. Monderer and L. Shapley, "Potential games," *Games and Economic Behavior*, pp. 124143, vol. 14(1), May 1996.

[61] R. W. Rosenthal, "A class of games possessing pure-strategy Nash equilibria," *International Journal of Game Theory*, pp. 65–67, vol. 2, no. 1, 1973.

[62] B. Awerbuch, Y. Azar, and Y. Bartal, "On-line generalized Steiner problem," *Theoretical Computer Science*, pp. 313–324, vol. 324, no. 2–3, September 2004.

[63] A. Agrawal, P. Klein, and R. Ravi, "When Trees Collide: An Approximation Algorithm for the Generalized Steiner Problem on Networks," *SIAM Journal on Computing*, pp. 440–456, vol. 24, no. 3, June 1995.

[64] M. X. Goemans and D. P. Williamson, "A general approximation technique for constrained forest problems," *Proc. of the 3rd Annual ACM-SIAM Symposium on Discrete algorithms*, pp. 307–316, 1992.

[65] K. Dowsland, "Nurse scheduling with tabu search and strategic oscillation," *European Journal of Operational Research*, pp. 393–407, vol. 106, no. 2–3, 1998.

[66] R. Aringhieri and M. Dell'Amico, "Comparing metaheuristic algorithms for the SONET network design problems," *Journal of Heuristics*, pp. 35–57, vol. 11, 2005.

[67] N. Mladenović and P. Hansen, "Variable neighborhood search," *Computers and Operations Research*, pp. 1097–1100, vol. 24, 1997.

[68] P. Hansen and N. Mladenović, "Variable neighborhood search: Principles and applications," *European Journal of Operational Research*, pp. 449–467, vol. 130, 2001.

[69] Z. Drezner and H. Hamacher, *Facility Location: Applications and Theory*. Springer, 2002.

[70] J. Campbell, "Integer programming formulations of discrete hub location problems," *European Journal of Operational Research*, pp. 387–405, vol. 72, 1994.

[71] A. Ernst and M. Krishnamoorthy, "Efficient algorithms for the uncapacitated single allocation p-hub median problem," *Location Science*, pp. 139–154, vol. 4, 1996.

[72] A. Ernst and M. Krishnamoorthy, "Exact and heuristic algorithms for the uncapacitated multiple allocation p-hub median problem," *European Journal of Operational Research*, pp. 100–112, vol. 104, no. 1, 1998.

[73] A. Ernst and M. Krishnamoorthy, "Solution algorithms for the capacitated single allocation hub location problem," *Annals of Operations Research*, pp. 141–159, vol. 86, 1999.

[74] J. Ebery, M. Krishnamoorthy, A. Ernst, and N. Boland, "The capacitated multiple allocation hub location problem: Formulations and algorithms," *European Journal of Operational Research*, pp. 614–631, vol. 120, no. 3, 2000.

[75] G. Mayer and B. Wagner, "HubLocator: an exact solution method for the multiple allocation hub location problem," *Computers and Operations Research*, pp. 715–739, vol. 29, no. 6, 2002.

[76] M. Labbé, H. Yaman, and E. Gourdin, "A branch and cut algorithm for hub location problems with single assignment," *Mathematical Programming*, pp. 371–405, vol. 102, no. 2, 2005.

[77] B. Wagner, "An exact solution procedure for a cluster hub location problem," *European Journal of Operational Research*, pp. 391–401, vol. 178, no. 2, 2007.

[78] J. Chen, "A hybrid heuristic for the uncapacitated single allocation hub location problem," *Omega, Elsevier*, pp. 211–220, vol. 35, no. 2, 2007.

[79] I. Rodríguez-Martín and J. Salazar-González, "Solving a capacitated hub location problem," *European Journal of Operational Research*, pp. 468–479, vol. 184, no. 2, 2008.

[80] J. Elias, "Neighborhood based heuristics for the Hub Location Problem," *Politecnico di Milano, Technical Report # 2008.19*, September 2008.

[81] H. Yaman, "Concentrator location in telecommunication networks." PhD thesis, Universit Libre de Bruxelles, Belgium, 2002. Available at http://smg.ulb.ac.be/.

[82] H. Yaman and G. Carello, "Solving the hub location problem with modular link capacities," *Computers and Operations Research*, pp. 3227–3245, vol. 32, no. 12, 2005.

[83] J. Klincewicz, "Hub location in backbone/tributary network design: a review," *Location Science*, pp. 307–335, vol. 6, 1998.

[84] R. Karp, "Reducibility among combinatorial problems," *Complexity of computer computations*, vol. 43, pp. 85–103, 1972.

[85] P. Winter, "Steiner problem in networks: a survey," *Networks*, vol. 17, no. 2, no. 2, pp. 129–167, 1987.

[86] N. Alon and Y. Azar, "On-line Steiner trees in the Euclidean plane," *Discrete and Computational Geometry*, vol. 10, no. 1, no. 1, pp. 113–121, 1993.

[87] J. Westbrook and D. Yan, "Greedy algorithms for the on-line Steiner tree and generalized Steiner problems," in *Proceedings of the Third Workshop on Algorithms and Data Structures*, pp. 622–633, 1993.

[88] M. Imase and B. Waxman, "Dynamic Steiner tree problem," *SIAM Journal on Discrete Mathematics*, vol. 4, pp. 369–384, 1991.

Appendix A

A.1 Overview of the Hub Location Problem

The Hub Location problem (HLP) is a well known optimization problem which finds large applicability in many sectors, such as air transportation, rapid transit, postal networks, service network design, trucking and telecommunications. Such problem involves locating hub facilities and assigning non-hub nodes to these facilities.

Hub Location problems consist of two kinds of nodes: the non-hub nodes and hubs. The non-hub nodes represent origins and destinations which must exchange flow but cannot be directly connected. Therefore, non-hub nodes are directly connected to hubs which collect the flow from the origins and route it through other hubs towards the destinations.

Different variants of HLPs are described in the literature. These Hub Location problems differ for the capacity constraints, the constraint on the number of hubs or the assignment requirements. A recent and less recent survey on Hub Location problems and on proposed algorithms can be found in [69] and in [70], respectively.

Several works have addressed different kinds of Hub Location problems [71, 72, 73, 74, 75, 76, 77, 78, 79].

The Uncapacitated Single and Multiple Allocation p-Hub Median problems are addressed in [71, 72]. In these versions, it is required to locate p uncapacitated hubs amongst n nodes that exchange traffic in a network. Non-hub nodes are to be allocated either uniquely ([71]) or multiply ([72]) to the chosen p hubs. There is no cost for establishing a hub. The objective is to minimize the total transportation costs in the network.

The same authors have also tackled in [73] the Capacitated Single Allocation Hub Location problem (CSAHLP) which is similar to the Hub Location problem considered in [80]. CSAHLP is an NP-hard problem (see [69, 81] for the proof for special cases). The authors have developed heuristic algorithms for solving CSAHLP based on simulated annealing and random descent.

In [74, 75], the Capacitated and Uncapacitated Multiple Allocation Hub Location problems are investigated. A shortest-path based heuristic algorithm is proposed in [74] to deal with the Capacitated Multiple Allocation Hub Location problem and such heuristic is then incorporated as an upper bound in a linear-programming based branch-and-bound solution procedure. A branch-and-bound procedure is introduced in [75] for the Uncapacitated Multiple Allocation Hub Location problem.

Moreover, polyhedral properties of Hub Location problems with single assignment are studied in [76] and a branch-and-cut algorithm is introduced to solve the Quadratic Capacitated Hub Location problem with single assignment.

The Uncapacitated Single Allocation Hub Location problem (USAHLP), with the hub-and-spoke network structure, is studied in [78]. To resolve USAHLP, two approaches are introduced: the first one determines the upper bound for the number of hubs and the second one is a hybrid heuristic which is based on the simulated annealing method, tabu list and improvement procedures.

In [79], the authors address a Capacitated Hub Location problem in where the arcs connecting the hubs are not assumed to create a complete graph. A mixed integer linear programming formulation is presented and two decomposition-technique based branch-and-cut algorithms are described for small size instances. Also, a heuristic approach based on a linear programming relaxation of the mixed integer model is proposed for larger instances.

Finally, many works have been published on similar problems related to telecommunication networks [49, 53, 81, 82]. A survey on general Backbone/Tributary Network Design Problems can be found in [83].

A.2 Overview of the Generalized Steiner and Steiner Tree Problems

The *Generalized Steiner Problem* (GSP) is defined as follows. We are given a graph with non-negative weights and a set of pairs of vertices. The algorithm has to construct minimum weight subgraph such that the two nodes of each pair are connected by a path. This problem [63, 64] has received a lot of attention in combinatorial optimization, networking, and distributed computing communities.

Agrawal et al. and Goemans et al. [63, 64] have shown a polynomial-time $2(1-(1/n))$-approximation algorithm. However, these algorithms are inapplicable in either on-line or distributed environments.

The special case of the GSP problem where all pairs of some subset of vertices have to be connected is the *Steiner tree problem*. It is one of the most notorious NP-hard problems [84, 85]. The problem has been studied in a series of papers

including [86, 87, 88].

A.3 AMPL: A Modeling Language for Mathematical Programming

AMPL is an algebraic modeling language for linear and nonlinear optimization problems, in discrete or continuous variables. Developed at Bell Laboratories, AMPL allows to formulate optimization models and examine solutions, while the computer manages communication with an appropriate solver. AMPL's flexibility and convenience make it ideal for rapid prototyping and model development, while its speed and control options make it an efficient choice for repeated production runs. For more details, the reader can refer to [46].

List of Acronyms

AMPL	*A Modeling Language for Mathematical Programming*
BRITE	*Boston university Representative Internet Topology gEnerator*
CEN	*Cyclic Exchange Neighborhood*
CS	*Candidate Site*
DN	*Destination Node*
FC-UAR	*Full Coverage User Assignment and Routing*
FCSD	*Full Coverage SON Design*
GT-ITM	*Georgia Tech Internetwork Topology Models*
H-FCSD	*Heuristic to solve the Full Coverage SON Design Problem*
H-PMSD	*Heuristic to solve the Profit Maximization SON Design Problem*
ILP	*Integer Linear Programming*
IP	*Internet Protocol*
ISP	*Internet Service Provider*
I-VLS-TS	*Iterative Very Large-Scale Tabu Search*
LP	*Linear Programming*
MCSD	*Minimum Cost SON Design*
MILP	*Mixed Integer Linear Programming*
MPLS	*Multiprotocol Label Switching*
NAD-SAND	*Network Administrator-Driven Socially-Aware Network Design*
NE	*Nash Equilibrium*
NP	*Non-deterministic Polynomial-time*
PM-UAR	*Profit Maximization User Assignment and Routing*
PMSD	*Profit Maximization SON Design*
PoA	*Price of Anarchy*
PoS	*Price of Stability*
PSAN	*Polynomial Size Allocation Neighborhood*
PSLN	*Polynomial Size Location Neighborhood*
QoS	*Quality of Service*
RPoA	*Reachable Price of Anarchy*
SAND	*Socially-Aware Network Design*

SON	*Service Overlay Network*
TP	*Test Point*
TS-MCSD	*Very Large-Scale Neighborhood Search Heuristic for the MCSD Problem*
VLSN	*Very Large Scale Neighborhood*

List of Figures

List of Tables