Chapter

Topological Properties and Dynamic Programming Approach for Designing the Access Network

Franco Robledo, Pablo Romero, Pablo Sartor, Luis Stábile and Omar Viera

Abstract

A wide area network (WAN) can be considered as a set of sites and a set of communication lines that interconnect the sites. Topologically a WAN is organized in two levels: the backbone network and the access network composed of a certain number of local access networks. Each local access network usually has a treelike structure, rooted at a single site of the backbone and connected users (terminal sites) either directly to this backbone site or to a hierarchy of intermediate concentrator sites which are connected to the backbone site. The backbone network has usually a meshed topology, and this purpose is to allow efficient and reliable communication between the switch sites that act as connection points for the local access networks. This work tackled the problem of designing the Access Network Design Problem (ANDP). Only the construction costs, e.g., the costs of digging trenches and placing a fiber cable into service, are considered here. Different results related to the topological structure of the ANDP solutions are studied. Given the complexity of the ANDP (the problem belongs to the NP-hard class), recurrences to solve it are proposed which are based on Dynamic Programming and Dynamic Programming with State-Space Relaxation methodology.

Keywords: topological design, access network, dynamic programming with state-space relaxation

1. Introduction

Telecommunication networks have become strategic resources for private- and state-owned institutions, and its economic importance continuously increases. There are series of recent tendencies that have a considerable impact on the economy evolution such as growing integration of networks in the productive system, integration of different services in the same communication system, and important modification in the telephone network structure. Such evolutions accompany a significant growth of the design complexity of these systems. The integration of different sorts of traffics and services and the necessity of a more accurate management of the service quality are factors that make this type of systems very hard to design, to dimension, and therefore to optimize. This situation is aggravated with a very high competitiveness context in an area of critical strategic importance.

The conception of a WAN is a process in which dozens of sites with different characteristics require to be connected in order to satisfy certain reliability and performance restrictions with minimal costs. This design process involves the terminal site location, the concentrator location, the backbone (central network or kernel) design, the routing procedures, as well as the lines and nodes dimensioning. A key aspect on WAN design is the high complexity of the problem, as much in its globality as in the principal subproblems in which it is necessary to decompose it. Due to the high investment levels, a cost decrease of very few percentage points while preserving the service quality results in high economic benefits.

Typically, a WAN network global topology can be decomposed into two main components: the access network and the backbone network. These components have very different properties, and consequently they introduce specific design problems (although they are strongly interdependent). On the one hand, this causes complicated problems (particularly algorithmic ones); on the other hand, it leads to stimulating and difficult research problems.

A WAN access network is composed of a certain number of access subnetworks, having treelike topologies; and the flow concentration nodes allow to diminish the costs. These integrated flows reach the backbone which has a meshed topology, in order to satisfy security, reliability, vulnerability, survivability, and performance criteria. Consequently, the backbone is usually formed by high-capacity communication lines such as optic fiber links.

Modeling a WAN design by means of the formulation of a single mathematical optimization problem is very intricate due to the interdependence of its large amount of parameters. Therefore the design of a WAN is usually divided into different subproblems [1–4]. A good example of a possible decomposition approach for the WAN design process is the following [5]:

1. Access and backbone network topologies design. Specific knowledge about the cost of laying lines between different network sites (terminals, concentrators, and backbone) is assumed. Frequently, these costs are independent of the type of line that will effectively be installed since they model the fixed one-time costs (cost of digging trenches in the case of optic fiber, installing cost, placing a fiber cable into service). A high percentage from the total construction network budget is spent in this phase [6].

2. Dimensioning of the lines that will connect the different sites of the access and backbone networks and the equipment to be settled in the mentioned sites.

3. Definition of the routing strategy of the flow on the backbone network.

This work focuses on phase (1) of the decomposition of a WAN design process. More precisely, it deals with the topology planning process concerning the access network. Due to the NP-hard nature of the problem and even though there exist some results, there is still room for improving industrial practices applied today. In this sense, the authors believe it is of strategic importance to design powerful quantitative analysis techniques, potentially easy to integrate into tools. Combinatorial optimization models are introduced that formally define the topological design of the access networks. Moreover, different results related to the topological structure are introduced. Finally, different algorithms are proposed for the topological design which are based on Dynamic Programming and Dynamic Programming with State-Space Relaxation methodology.

2. A model for a WAN design

In this section, a model for the design of a WAN is introduced. The model tries to show the most essential aspects which are considered when designing access and backbone networks. In this model, some parameters are not considered: the operation probability of the lines and equipment, the number of equipment ports, and the memory capacity of the equipment. The objective is to design a WAN with the smallest possible installation cost, so that the constraints are satisfied.

In what follows, the data of the model are presented as well as its formalization as a combinatorial optimization problem on weighted graphs. The goal is to find the optimal topology that satisfies the imposed constraints to the access and backbone networks. **Figure 1** shows an example of a wide area network. The information available for each type of equipment (switch and concentrator) and each type of connection line, as well as the line laying, is the following:

- E_a is the set of types of connection lines available. Furthermore $\forall e \in E_a$ the following data are given:
    - c_e is the cost by kilometer of the line type e. Here the laying cost is not included.
    - v_e is the speed in Kbits/s of the line type e.
- K is the set of types of concentrator equipment available. Furthermore $\forall k \in K$ the following data are given:

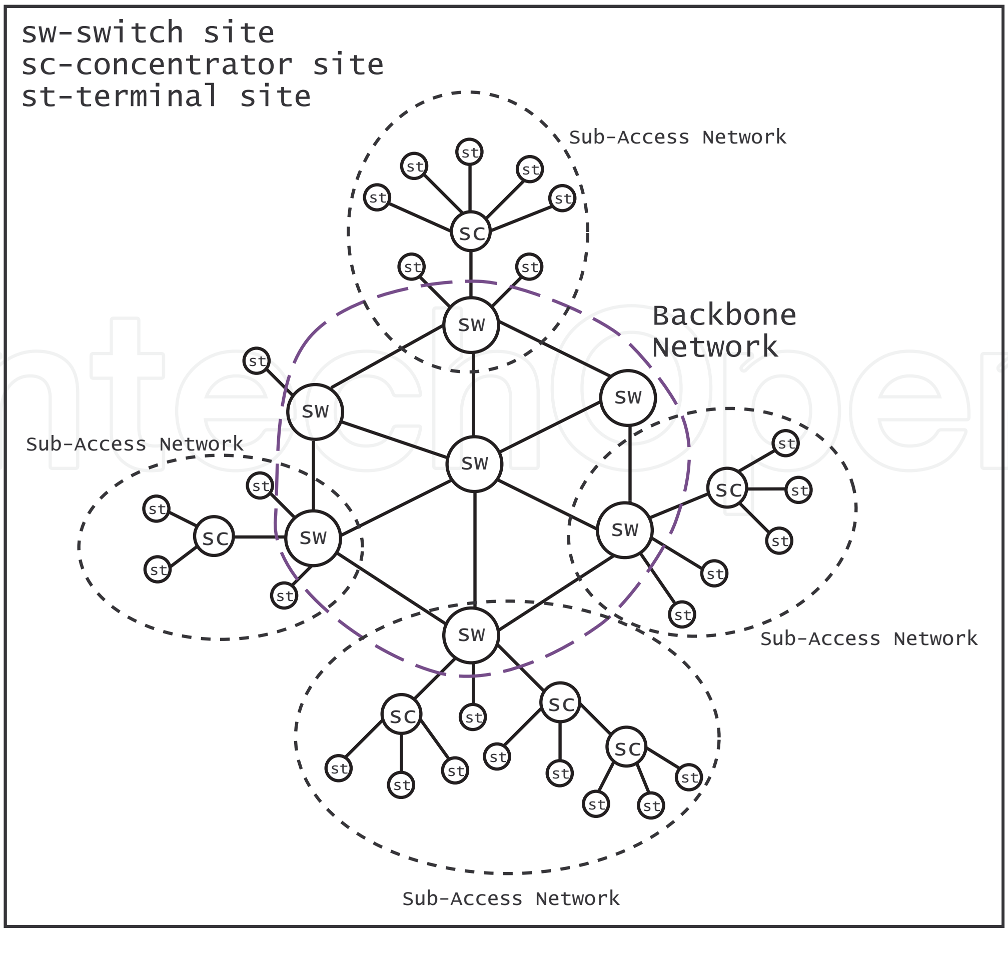

Figure 1.
WAN example.

   - c_k is the installation cost of the concentrator type k.
   - v_k is the speed in Kbits/s of the concentrator type k.

- W is the set of types of switch equipment available. Furthermore $\forall w \in W$ the following data are given:
   - c_w is the installation cost of the switcher type w.
   - v_w is the speed in Kbits/s of the switcher type w.

- $C = F_{cost}(L) = \{c_{ij} = \textit{direct connection costs between the sites } i,j; \forall i \in S, \forall j \in S_C \cup S_D\}$; this matrix gives us, for a site of S and a site of $S_C \cup S_D$, the cost of laying a line among them. When the direct connection among both places is not possible, we assume that $c_{ij} = \infty$.

In terms of graph theory, a model for the design of a WAN, based on the problem, is presented as follows. Some notation is introduced next, that is then used to formally define the problem.

- $E_1 = \{(i,j); \forall i \in S_T, \forall j \in S_C \cup S_D / d_{ij} < \infty\}$ is the set of feasible connections between a terminal site and a concentrator or switch site.

- $E_2 = \{(i,j); \forall i \in S_C, \forall j \in S_C \cup S_D / d_{ij} < \infty\}$ is the set of feasible connections between a concentrator site and a switcher or another concentrator site.

- $E_3 = \{(i,j); \forall i \in S_D, \forall j \in S_D / d_{ij} < \infty\}$ is the set of feasible connections between two switch sites.

- $E = E_1 \cup E_2 \cup E_3$ is the set of all feasible connections on the WAN.

- $D_{S_T} = \{D_{t_i}, t_i \in S_T\}$, where • D_{t_i} is the set of terminal nodes which demand connections with $t_i \in S_T$.

- $V_{S_T} = \{v_{i,j}\}_{i,j \in S_T}$ is the traffic demand matrix.

Definition 1 (WANDP—wide area network design problem). Let $G = (S, E)$ be the graph of feasible connections on the WAN. The wide area network design problem $(S, E, K, W, E_a, C, D_{S_T}, V_{S_T})$ consists in finding a subnetwork of G of minimum cost which satisfies the following points:

1. The backbone network topology must be at least 2-node-connected.
2. The access and backbone networks must be able to support the demand of connection and traffic required by the terminal sites.

Given the complexity of the WANDP, to facilitate its solution, the topological design problem is divided into three subproblems:

1. The Access Network Design Problem
2. The backbone network design problem (BNDP)

3. The routing (or flow assignment) and capacity assignment problem (RCAP)

The remainder of this work concentrates only in the first problem (ANDP).

3. Access Network Design Problem

The Access Network Design Problem is defined as follows.

Definition 2 (ANDP—Access Network Design Problem). Let $G_A = (S, E_1 \cup E_2)$ be the graph of feasible connections on the access network and C the matrix of connection costs defined previously. The Access Network Design Problem $(S, E_1 \cup E_2, C)$ consists in finding a subgraph of G_A of minimum cost such that $\forall i \in S_T;$ there exists a path from i to some site $j \in S_D$ of the backbone network.

Notation 1. Γ_{ANDP} denotes the space of feasible solutions of ANDP$(S, E_1 \cup E_2, C)$ that do not have any cycle and with an output only toward the backbone network $\forall t \in S_T$. These have forest topology as we illustrate in **Figure 2**.

In order to define these problems in terms of graph theory, the following notation is introduced:

- S_T is the set of terminal sites (clients) to be connected to the backbone.
- S_C is the set of feasible concentrator sites of the access network. On each one of these sites, an intermediate server equipment might be placed. From this one, a trunk line is laid toward the backbone or other concentrator site.
- S_D is the set of feasible switch sites of the backbone network. On each one of these sites, a powerful server might be placed and, from it, connection lines toward other backbone server equipment.
- $V = S_T \cup S_C \cup S_D$ are all the feasible sites of the WAN network.
- $A = \{a_{ij}\}_{i,j \in V}$ is a matrix which gives for any pair of sites $i, j \in V$, the cost $a_{ij} \geq 0$ of laying a line between them. When the direct connection between i and j is not possible, we define $a_{ij} = \infty$.

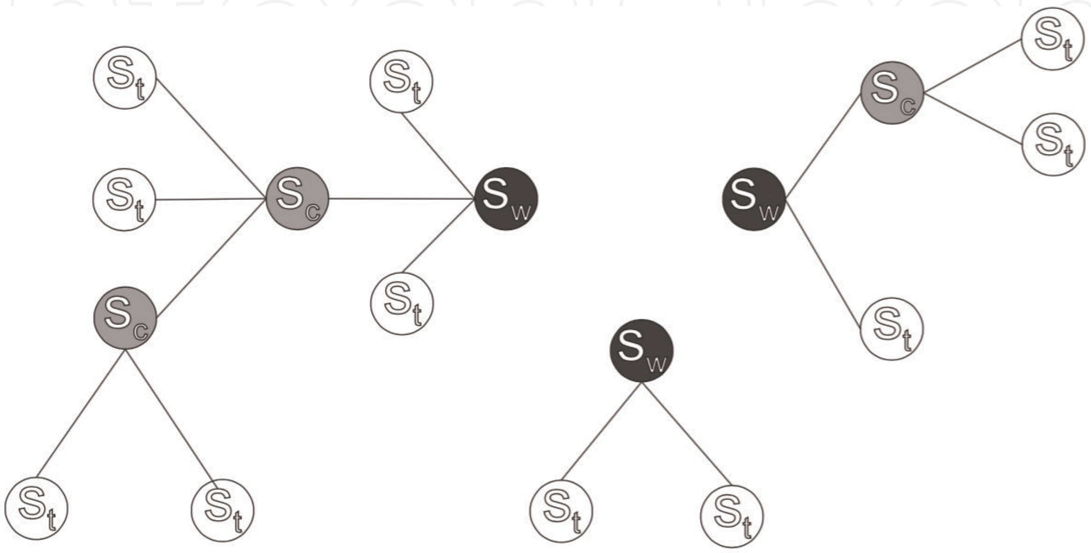

Figure 2.
A feasible solution of ANDP.

- $U = \{(i,j) | i, j \in V, a_{ij} < \infty\}$ is the set of all the feasible connections between the different sites of the WAN network.

- $G = (V, U)$ is the simple graph which models every node and feasible connection of the WAN.

The General Access Network Design Problem (GANDP) consists of finding a minimum-cost subgraph $H \subset G$ such that all the sites of S_T are communicated with some node of the backbone. This connection can be direct or through intermediate concentrators. The use of terminal sites as intermediate nodes is not allowed; this implies that they must have degree one in the solution.

The GANDP is here simplified by collapsing the backbone into a fictitious node and given the name of "Access Network Design Problem." The equivalence between both problems, GANDP and ANDP, as well as the NP-hardness of the ANDP, is proved in [7].

This work concentrates on the ANDP with the objective of proposing a new approach for solving this problem. We study different results related to the topological structure of the ANDP solutions. In particular we present results that characterize the topologies of the feasible solutions of an ANDP instance. The following proposition shows the topological form of the feasible solutions of ΓANDP for a given ANDP instance.

Proposition 1. Given an ANDP with associated graph $G_A = (S, E_1 \cup E_2)$ and matrix of connection costs C. If the subnetwork $H = \left(S_T \cup \overline{S}, \overline{E}\right)$ (with $\overline{S} \subseteq S_C \cup S_D$ and $\overline{E} \subseteq E_1 \cup E_2$) is an optimal solution of ΓANDP, it is composed of a set of disjoint trees $H = \{H_1, \ldots, H_m\}$ that satisfy:

1. $\forall H_l \in H, \exists j \in S_D \; unique \; / j \in H_l$

2. $\forall H_l \in H, \exists \; a \; subset \; S_T^l \subset S_T, S_T^l \neq \varnothing_{\overline{S_T^l}} \subseteq NODES(H_l)$

3. $\bigcup_{l=1}^{m} S_T^l = S_T$

Proof. Trivial.

The following propositions present results that characterize the structure of the global optimal solution.

Proposition 2. Let ANDP $(S, E_1 \cup E_2, C)$ be a problem where $s_c \in S_C, \bar{s} \in S_C \cup S_D$ and $s \in S_T \cup S_C$ such that $\{(s, s_c), (s_c, \bar{s})\} \subset E_1 \cup E_2$ and $\exists s_w \in S_D / c_{s,s_w} < c_{s,s_c} + c_{s_c,\bar{s}}$. Then, if $T_A \in$ ΓANDP is a globally optimal solution, it is fulfilled that $g(s_c) \geq 3$ in T_A, $\forall s_c \in T_A, s_c \in S_C$.

Proof. Let us suppose that there exists $T_A \in$ ΓANDP global optimal solution such that $\exists s_c \in T_A$ a concentrator site with $g_{s_c} < 3$ in T_A. If $g(s_c) = 1;$ then s_c is a pendant in T_A; therefore, eliminating this, a feasible solution of smaller cost would be obtained. This is a contradiction; hence, $g(s_c) \neq 1$. If $g(s_c) = 2$, let $\bar{s} \in S_C \cup S_D$ be the site adjacent to s_c in T_A which its output site is toward the backbone network. Let $s \in S_T \cup S_C$ be the other adjacent site in T_A. Considering the network $H = (T_A\{s_c\}) \cup \{(s, s_w)\}$, where $s_w \in S_D$ satisfies $c_{s,s_w} < c_{s_c,\bar{s}}$, it is fulfilled:

$$COST(H) = COST(T_A) - c_{s,s_c} - c_{s_c,\bar{s}} + c_{s,s_w} < COST(T_A) \qquad (1)$$

Furthermore, it is easy to see that $H \in$ ΓANDP. Hence, this implies that H is a better feasible solution compared with T_A. This is a contradiction, entailing that $g(s_c) \geq 3$ in T_A, as required and completing the proof.

Proposition 3. Given an ANDP $(S, E_1 \cup E_2, C)$ such that for any three sites (s_1, s_2, s_3), with $s_1 \in S_T \cup S_C$, $s_2 \in S_C$ and $s_3 \in S_C \cup S_D$, the strict triangular inequality is satisfied, i.e., $c_{s_1, s_k} < c_{s_i, S_j} + c_{s_j, s_k}$, $i, j, k \in \{1, 2, 3\}$. Then, if $T_A \in \Gamma_{\text{ANDP}}$ is a globally optimal solution, it is fulfilled that $g(s_c) \geq 3$ in T_A, $\forall s_c \in T_A$, $s_c \in S_C$.

Proof. As in the previous proposition, let us suppose that there exists $T_A \in \Gamma_{\text{ANDP}}$ global optimal solution such that $\exists s_c \in T_A$, a concentrator site with $g(s_c) < 3$ in T_A. Clearly $g(s_s)$ must be different to 1. Now, let us consider the case $g(s_c) = 2$ inT_A. Let s_1, s_2 be the adjacent sites to s_c in T_A. By hypothesis $c_{s_1, s_2} < c_{s_1, s_c} + c_{s_c, s_2}$. Considering the network $\overline{T_A} = (T_A\{s_c\}) \cup \{(s_1, s_2)\}$, a feasible solution is found, and moreover

$$COST\left(\overline{T}_A\right) = COST(T_A) - c_{s_1, s_c} - c_{s_c, s_2} + c_{s_1, s_2} < COST(T_A) \tag{2}$$

This is a contradiction; therefore, $g(s_c) \geq 3$ in T_A, hence completing the proof.

The next section presents algorithms applied to the ANDP$^{(\leq k)}$ with $k \in \{1, 2\}$. A way of computing the global optimal solution cost of it using the Dynamic Programming approach is obtained. Considering that the ANDP$^{(\leq 1)}$ is a NP-hard problem, we obtain lower bounds to the global optimal solution cost by Dynamic Programming with State-Space Relaxation in polynomial time.

4. Algorithms applied to the ANDP

This chapter presents the Dynamic Programming approach as alternative methodology to find a global optimal solution cost for the ANDP$^{(\leq 1)}$ and ANDP$^{(\leq 2)}$. After we introduce the Dynamic Programming with State-Space Relaxation as a method to obtain lower bounds for the original problem.

4.1 Dynamic Programming

Proposition 4. Given an ANDP $(S, E_1 \cup E_2, A)$, the cost of a global optimal solution of $\varGamma^{\leq 1}_{ANDP}$ is given by$f_{\left(S_T, Z, A^Q\right)}$, with$f_{(.,.,.)}$ defined by the following expression of Dynamic Programming:

$$f_{S_C}\left(S_T, Z, A^Q\right) = \left\{ \begin{array}{lr} \min\limits_{s_t \in S_T} \left\{ \begin{array}{c} COST(s_t, Z) + f_{S_C}\left(S_T\{s_t\}, Z, A^Q\right), \\ \min\limits_{s_c \in S_C} \left\{ \begin{array}{c} COST(s_t, s_c) + COST(s_c, Z) + \\ f_{S_C}\left(S_T\{s_t\}, Z, A^{Q \cup \{(s_c, Z)\}}\right) \end{array} \right\} \end{array} \right\} & if S_T \neq \varnothing \\ \qquad\qquad\qquad 0 & otherwise \end{array} \right\} \tag{3}$$

where $COST(s, Z) = \min_{z \in S_D} \{COST(s, z)\}$, $(s, Z) = \text{argmin}_{z \in S_D} \{COST(s, z)\}$ and the matrix of connection costs $A^Q = \{a_{i,j}\}_{i,j \in E_1 \cup E_2}$ is defined by

$$a_{i,j} = \left\{ \begin{array}{cc} COST(i, j) & if (i, j) \notin Q \\ 0 & otherwise \end{array} \right. \tag{4}$$

Proposition 5. Given an ANDP $(S, E_1 \cup E_2, A)$, the cost of a global optimal solution of $\varGamma^{\leq 2}_{ANDP}$ is given by$f_{\left(S_T, Z, A^Q\right)}$, with$f_{(.,.,.,.)}$ defined by the following expression of Dynamic Programming

$$f_{S_C}\left(S_T, Z, A^Q\right) = \begin{cases} \min\limits_{s_t \in S_C} \left\{ \begin{array}{c} COST(s_t, Z) + f_{S_C}\left(S_T\{s_t\}, Z, A^Q\right), \\ \min\limits_{s_c \in S_C} \left\{ \begin{array}{c} COST(s_t, s_c) + COST(s_c, Z) + \\ f_{S_C}\left(S_T\{s_t\}, Z, A^{Q \cup \{(s_c, Z)\}}\right) \end{array} \right\}, \\ \min\limits_{\left(s_c^u, s_c^v\right) \in E_2} \left\{ \begin{array}{c} COST\left(s_t, s_c^u\right) + \\ COST\left(s_c^u, s_c^v\right) + COST\left(s_c^v, Z\right) + \\ f_{S_C}\left(S_T\{s_t\}, Z, A^{Q \cup \left\{\left(s_c^u, s_c^v\right), \left(s_c^v, Z\right)\right\}}\right) \end{array} \right\} \end{array} \right\} & if S_T \neq \emptyset \\ 0 & otherwise \end{cases} \quad (5)$$

where $COST(s, Z) = \min_{z \in S_D} \{COST(s, z)\}$, $(s, Z) = \operatorname{argmin}_{z \in S_D} \{COST(s, z)\}$ and the matrix of connection costs $A^Q = \left\{a_{i,j}\right\}_{i,j \in E_1 \cup E_2}$ is defined by

$$a_{i,j} = \begin{cases} COST(i,j) & if\,(i,j) \notin Q \\ 0 & otherwise \end{cases} \quad (6)$$

4.2 Dynamic programming with state-space relaxation

In order to find a lower bound of $f_{S_C}\left(S_T, Z, A^Q\right)$, the Dynamic Programming with State-Space Relaxation is now applied. It is a general relaxation procedure applied to a number of routing problems [8]. The motivation for this methodology stems from the fact that very few combinatorial optimization problems can be solved by Dynamic Programming alone due to the dimensionality of their state-space. To overcome this difficulty, the number of states is reduced by mapping the state-space associated with a given Dynamic Programming recursion to a smaller cardinality space. This mapping, denoted by g, must associate to every transition from a state S_1 to a state S_2 in the original state-space, a transition $g(S_1)$ to $g(S_2)$ in the new state-space. To be effective, the function g must give rise to a transformed recursion over the relaxed state-space which can be computed in polynomial time. Furthermore, this relaxation must generate a good lower bound for the original problem.

With the aim of illustrating this methodology, we present this approach in the context of the minimization of the total schedule time for the Traveling Salesman Problem with Time Window (TSPTW), after we apply it to the Dynamic Programming recursion presented in Proposition 5. The objective of the TSPTW is to find an optimal tour where a single vehicle is required to visit each of a given set of locations (customers) exactly once and then return to its starting location. The vehicle must visit each location within a specified time window, defined by an earliest service start time and latest service start time. If the vehicle arrives at a service location before the earliest service start time, it is permitted to wait until the earliest service start time is reached. The vehicle conducts its service for a known period of time and immediately departs for the location of the next scheduled customer. Assume that the time constrained path starts at fixed time value a_o. Define $F(S, i)$ as the shortest time it takes for a feasible path starting at node o, passing through every node of $S \subseteq N$ exactly once, to end at node $i \in S$. Note that optimization of the total arc cost would involve an additional dimension to account for the arrival time at a node. The function $F(S, i)$ can be computed by solving the following recurrence equations:

$$F(S, j) = \min_{(i,j) \in E} \left\{ F(S - \{j\}, i) + t_{ij} \middle| i \in S - \{j\} \right\} \forall S \subseteq N, j \in S \quad (7)$$

The recursion formula is initialized by

$$F(\{j\},j)=\begin{cases}\max\{a_j,a_o+t_{oj}\} & \mathit{if}\,(o,j)\in E\\ +\infty & \mathit{otherwise}\end{cases}\tag{8}$$

The optimal solution to the TSPTW is given by

$$\min_{j\in N}\{F(N,j)+t_{jd}\}\tag{9}$$

Note that Eq. (7) is valid if $a_j\le F(S,j)\le b_j$. If however $F(S,j)<a_j$, then $F(S,j)=a_j$; if $F(S,j)>b_j$, $F(S,j)=\infty$. Equations (7) and (9) define a shortest path algorithm on a state graph whose nodes are the states (S,i) and whose arcs represent transitions from one state to another. This algorithm is a forward Dynamic Programming algorithm where at step s, with $s=1,\dots,n+1$, a path of length s is generated. The state (S,i) of cost $F(S,i)$ are defined as follows: S is an unordered set of visited nodes and i is the last visited node, $i\in S$.

Several alternatives for the mapping g have been suggested [9]. Here is presented the shortest r-path relaxation, i.e., $g(S)=r=\sum_{i\in S}r_i$, where $r_i\ge 1$ is an integer associated with node $i\in N$; then $g(S\{i\})=g(S)-r_i$. Define $R=\sum_{i\in S}r_i$. Hence the transformed recursion equations are

$$F(r,j)=\min_{(i,j)\in E}\{F(r-r_j,i)+t_{ij}|r-r_j\ge r_i\},r\in\{1,\dots,R\},j\in N\tag{10}$$

Recursion (10) holds if $a_j\le F(r,j)\le b_j$. Otherwise, if $F(r,j)<a_j$, then $F(r,j)=a_j$; if $F(r,j)>b_j$, $F(r,j)=\infty$. The recursion formula is initialized by

$$F(\{j\},j)=\begin{cases}\max\{a_j,a_o+t_{oj}\} & \mathit{if}\ (o,j)\in E\ \mathit{and}\ q=q_j\\ +\backslash\mathit{infty} & \mathit{otherwise, for}\ q\in\{1,\dots,Q\},j\in N\end{cases}\tag{11}$$

The lower bound is given by

$$\min_{j\in N}\{F(R,j)+t_{jd}\}\tag{12}$$

The complexity of the bounding procedure is $O(n^2\times Q)$ for a n-node problem. Now, we present this approach in the context of finding a “good” lower bound for the solution of ANDP$^{(\le 2)}$. The following proposition gives a lower bound for the $f_{S_C}(S_T,Z,A^Q)$ presented in Proposition 5 (the optimum value of the ANDP$^{(\le 2)}$).

Proposition 6. Given an ANDP $(S,E_1\cup E_2,C)$, a lower bound of $f_{S_C}(S_T,Z,A^Q)$ is derived from the following expression of Dynamic Programming with State-Space Relaxation

$$g_{S_C}(r,Z,A^Q)=\begin{cases}\min\limits_{s_t^i\in S_T}\left\{\begin{matrix}COST(s_t^i,Z)+g_{S_C}(r-r_i,Z,A^Q),\\ \min\limits_{s_c^j\in S_C}\left\{\begin{matrix}COST\left(s_t^i,s_c^j\right)+COST\left(s_c^j,Z\right)+\\ g_{S_C}\left(r-r_i,Z,A^{Q\cup\{(s_c^j,Z)\}}\right)|r-\hat{R}-r_i\ge r_j\end{matrix}\right\},\\ \min\limits_{\left(s_c^{ju},s_c^k\right)\in E_2}\left\{\begin{matrix}COST\left(s_t^i,s_c^j\right)+COST\left(s_c^j,s_c^k\right)+COST(s_c^k,Z)+\\ g_{S_C}\left(r-r_i,Z,A^{Q\cup\{(s_c^j,s_c^k),(s_c^k,Z)\}}\right)|\\ r-\hat{R}-r_i\ge r_j+r_k\end{matrix}\right\}\end{matrix}\right\} & \mathit{if}\ S_T\ne\varnothing\\ 0 & \mathit{otherwise}\end{cases}\tag{13}$$

where $1 \le r_i \le R$ is an integer associated with the site $i \in S_T \cup S_C$, $R = \sum_{i \in S_T \cup S_C} r_i$, $\hat{R} = \sum_{j \in S_C} r_j$ and the matrix of connection costs $A^{Q} = \left\{a_{i,j}\right\}_{i,j \in E_1 \cup E_2}$ is defined by

$$a_{i,j} = \begin{cases} COST(i,j) & \text{if } (i,j) \notin Q \\ 0 & otherwise \end{cases} \tag{14}$$

The lower bound is given by $g_{(R,Z,A^{\varnothing})}$.

5. Computational results

This section presents the experimental results obtained with the recursions of above. The algorithms were implemented in ANSI C. The experimental results were obtained in an Intel Core i7, 2.4 GHz, and 8 GB of RAM running under a home PC. The recursions presented in Propositions 4 and 5 were applied to the ANDP$^{(\le 1)}$ and the ANDP$^{(\le 2)}$, respectively, whereas the recursion presented in Proposition 6 was applied to ANDP$^{(\le 2)}$. They were tested using a large test set, by modifying the Steiner Problem in Graphs (SPG) instances from SteinLib [10]. This library contains many problem classes of widely different graph topologies. Most of the problems were extracted from these classes: C, MC, X, PUC, I080, I160, P6E, P6Z, and WRP3. The SPG problems were customized, transforming them into ANDP instances by means of the following changes. For each considered problem:

1. The terminal node with greatest degree was chosen as the z node (modeling the back- bone).

2. The Steiner nodes model the concentrator sites, and the terminal nodes model the terminal sites.

3. All the edges between terminal sites were deleted (as they are not allowed in feasible ANDP solutions).

Moreover, if the resulting topology was unconnected, the problem instance was discarded. Let us notice that since in the ANDP the terminals cannot be used as intermediate nodes (which implies also that edges between pairs of terminals are not allowed), the cost of a SPG optimum is a lower bound for the optimum of the corresponding ANDP. Therefore they are for ANDP$^{(\le k)}$ with $k \in 1 \ldots 2$.

Table 1 shows the results obtained by applying the recurrences presented in Propositions 4 and 5. In each one of them, the first column contains the names of the original SteinLib classes with the name of the customized instance. The entries from left to right are:

- The size of the selected instance in terms of number of nodes, edges, and terminal sites, respectively

- A lower bound for the optimal cost; the SPG optimum cost (LB_{SPG})

- c^1_{opt} and c^2_{opt} where c^k_{opt} is the cost of the best feasible solution found in $\Gamma^{(\le k)}_{ANDP}$

- The gap of the cost for the best feasible solution of $\Gamma^{(\le k)}_{ANDP}$ with respect to the lower bound $LB^{(k)}_{SPG}$ with $k \in \{1, 2\}$ $\left(LB\_GAP^{(k)}_{SPG}\right)$

Set	Name	|V|	|E|	|T|	LB_{SPG}	c_{opt^1}	c_{opt^2}	$LBGAP_{SPG^{(1)}}$	$LBGAP_{SPG^{(2)}}$
I080	i080-001	80	120	6	1787	∞	2187		22.38%
I080	i080-011	80	350	6	1479	∞	1499		1.35%
I080	i080-012	80	350	6	1484	∞	1497		0.88%
I080	i080-013	80	350	6	1381	∞	1383		0.14%
I080	i080-014	80	350	6	1397	∞	1505		7.73%
I080	i080-111	80	350	8	2051	∞	2159		5.27%
I080	i080-112	80	350	8	1885	2201	1887	16.76%	0.11%
I080	i080-113	80	350	8	1884	∞	1884		0%
I080	i080-114	80	350	8	1895	∞	2099		10.77%
I080	i080-115	80	350	8	1868	2174	1969	16.38%	5.41%
I080	i080-233	80	160	16	4354	∞	4564		4.82%
I160	i160-011	160	812	7	1677	∞	1875		11.81%
I160	i160-012	160	812	7	1750	∞	1891		8.06%
I160	i160-013	160	812	7	1661	∞	1862		12.10%
I160	i160-014	160	812	7	1778	∞	1991		11.98%
I160	i160-015	160	812	7	1768	2281	1864	29.02%	5.43%
PUC	cc3-4p	64	288	8	2338	∞	2553		9.20%
PUC	cc3-4u	64	288	8	23	∞	25		8.70%
Average								20.72%	7.01%

Table 1.
Results obtained by applying Dynamic Programming to c_{opt}^1 and c_{opt}^2.

The $LB\_GAP_{SPG}^{(k)}$ is computed as

$$LB\_GAP_{SPG}^{(k)} = 100 \times \frac{c_{opt}^k - LB_{SPG}}{LB_{SPG}}. \tag{15}$$

Feasible solutions were obtained here only for i080-112, i080-115, and i160-015 with $k = 1$ because, as can be seen, the cost is finite. The optimal values of the SPG instances (LBSP G) provided lower bounds for the optimal values of the ANDP (therefore to ANDP(≤k) with $k \geq 0$), considering that in the ANDP generation process, all the connections between terminal nodes were deleted and further that ANDP's feasible solution space is more restrictive than of SPG. The experimental results obtained for c_{opt}^1 have an average gap with respect to the lower bound of 20.72%. Increasing k to 2 (applying the recursion presented in Proposition 5), feasible solutions were obtained for all the testing networks, and the experimental results obtained have an average gap with respect to the lower bound of 7.01%.

It can be proved that (it is out of the scope of this chapter) increasing k, the following inequality is fulfilled:

$$\frac{c_{opt}^{k-1}}{c_{opt}^{k}} \leq 1 + \text{floor}\left(\frac{n_C}{k}\right) \cdot \left(\frac{1}{k + n_T}\right) \cdot \left(\frac{c_{max}}{c_{min}} - 1\right) \tag{16}$$

Table 2 shows the results obtained. Despite the bound was not good in these cases (due the heterogeneity of costs of the lines), it can help us in some cases to answer the following question: how much can be saved with a higher k?

Name	n_T	n_C	c_{min}	c_{max}	$\frac{c_{opt^1}}{c_{opt^2}}$	$1+floor\left(\frac{n_c}{2}\right)\left(\frac{1}{2+n_T}\right)\left(\frac{c_{max}}{c_{min}}-1\right)$
i080-112	7	72	85	209	1.166401	5.997385619
i080-115	7	72	86	302	1.1004114	10.325581395
i160-015	6	153	86	300	1.223712	23.639534884

Table 2.
Relation between optimal solutions of ANDP$^{(\leq 1)}$ *and* ANDP$^{(\leq 2)}$.

Set	Name	|V|	|E|	|T|	c_{opt^2}	$t_{c_{opt^2}}$	$LB_{SSR^{(2)}}$	$t_{LB_{SSR^{(2)}}}$	$LBGAP_{SSR^{(2)}}$
I080	i080-001	80	120	6	2187	0	1698	0	28.8%
I080	i080-011	80	350	6	1499	6.04	1307	0.27	14.69%
I080	i080-012	80	350	6	1497	5.33	1486	0.16	0.74%
I080	i080-013	80	350	6	1383	8.20	1000	0.92	38.3%
I080	i080-014	80	350	6	1505	4.89	1211	0.25	24.28%
I080	i080-111	80	350	8	2159	3.09	1982	0.45	8.93%
I080	i080-112	80	350	8	1887	1812	1501	7.52	25.72%
I080	i080-113	80	350	8	1884	1809	1591	393.8	18.42%
I080	i080-114	80	350	8	2099	44.81	1988	6.65	5.58%
I080	i080-115	80	350	8	1969	479.8	1496	15.41	31.62%
I080	i080-233	80	160	16	4564	361.1	3997	6.75	14.19%
I160	i160-011	160	812	7	1875	45.67	1399	2.17	34.02%
I160	i160-012	160	812	7	1891	8.83	1502	1.13	25.9%
I160	i160-013	160	812	7	1862	6.58	1381	1.81	34..83%
I160	i160-014	160	812	7	1991	6.06	1783	0.86	11.67%
I160	i160-015	160	812	7	1864	70.28	1793	6.21	3.96%
PUC	cc3-4p	64	288	8	2553	79.37	2177	2.54	17.27%
PUC	cc3-4u	64	288	8	25	80.04	21	5.18	19.05%
Average								19.89%	

Table 3.
Lower bounds obtained to ANDP$^{(\leq 2)}$ *by applying Dynamic Programming with State-Space Relaxation.*

Table 3 shows the results obtained by applying the recursion presented in Proposition 6. As before the first column contains the names of the original SteinLib classes with the name of the customized instance. The entries from left to right are:

- The size of the selected instance in terms of number of nodes, edges, and terminal sites, respectively
- The cost of a global optimal solution of $\Gamma^{(\leq 2)}_{ANDP}\left(c^2_{opt}\right)$
- The execution time, in seconds, for $c^2_{opt}\left(t_{c^2_{opt}}\right)$
- A lower bound for the cost of a global optimal solution of $\Gamma^{(\leq 2)}_{ANDP}$ by applying Dynamic Programming with State-Space Relaxation (presented in Proposition 6) $(LB^{(2)}_{SSR})$

- The execution time, in seconds, for $LB_{SSR}^{(2)}\left(t_{LB_{SSR}^{(2)}}\right)$
- The gap of the cost for a global optimal solution of $\Gamma_{ANDP}^{(\leq 2)}\left(c_{opt}^{2}\right)$ with respect to the lower bound $LB_{SSR}^{(2)}$; $LB\_GAP_{SSR}^{(2)}$

The $LB\_GAP_{SSR}^{(2)}$ is computed as

$$LB\_GAP_{SSR}^{(2)} = 100 \times \frac{c_{opt}^{2} - LB^{(2)}{}_{SSR}}{LB^{(2)}{}_{SSR}} \tag{17}$$

In general, the gaps related to the lower bounds were low. The $r_i$to each terminal site and concentrator site were distinct integers chosen from $\{1, \dots |S_T \cup S_C|\}$. This lower bound can be increased by modifying the state-space through the application of subgradient optimization to r_i. As future work, it is possible to incorporate the method for a better choice of r_i.

It can be noticed that the execution times of computing global optimal solution costs were much longer than using Dynamic Programming with State-Space Relaxation.

6. Conclusions

The implementation of the algorithms was tested on a number of different problems with heterogeneous characteristics. In particular, a set of ANDP instances transforming 18 SPG instances extracted from SteinLib was built. The optimal values for the selected SPG instances are lower bound for the corresponding ANDP. The solutions found by the algorithm were, in average, 21% and 7% lower than the mentioned bounds in ANDP$^{(\leq 1)}$ and ANDP$^{(\leq 2)}$, respectively. It is reasonable supposing that the gaps related to the global optimum of the ANDP instances be even lower since the feasible solutions of the ANDP that are also feasible solutions of the original SPG, but not reciprocally. In this sense, remember that in any ANDP instance generated, all the edges between pairs of terminal nodes were deleted (because in our ANDP such connections are not allowed) having the additional constraint that the terminal nodes must have degree one in the solution.

Besides, a Dynamic Programming with State-Space Relaxation algorithm was developed which can give a lower bound in polynomial time. The average gaps with respect to the global optimal solution costs were lower than 20%.

Notice that, as expected, the execution times of the proposed algorithms are strongly dependent on the number of sites, edges, and terminal sites. To sum up, as far as the authors are concerned, the results obtained with the recurrences above are very good, considering that computing the global optimal solution of an ANDP$^{(\leq 2)}$ is a NP-hard problem.

Author details

Franco Robledo[1], Pablo Romero[1], Pablo Sartor[2]*, Luis Stábile[1] and Omar Viera[1]

1 Instituto de Computación, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay

2 Departamento de Análisis de Decisiones, IEEM Business School, Universidad de Montevideo, Montevideo, Uruguay

*Address all correspondence to: psartor@um.edu.uy

References

[1] Xie S, Ouyang Y. Reliable service systems design under the risk of network access failures. Transportation Research Part E: Logistics and Transportation Review. 2019;**122**(1): 1-13. DOI: 10.1016/j.tre.2018.11.002

[2] Lee Y, Kim Y, Park G. An access network design problem with end-to-end QoS constraints. Omega. 2014; **48**(1):36-48

[3] Zhang J, Sun X, Wandelt S. HUBBI: Iterative network design for incomplete hub location problems. Computers and Operations Research. 2019;**104**(1): 394-414. DOI: 10.1016/j.cor.2018.09.011

[4] Ljubic I, Putz P, Salazar-González J. A MIP-based approach to solve the prize-collecting local access network design problem. European Journal of Operational Research. 2014;**253**(3): 727-739

[5] Priem M, Priem F. Ingénierie des WAN (text in French). Paris: Dunod InterEditions; 1999

[6] Stoer M. Design of Survivable Networks. Vol. 1532. Berlin Heidelberg: Springer-Verlag; 1992

[7] Robledo F. GRASP heuristics for a Wide Area Network design [PhD thesis]. Universidad de la República Oriental del Uruguay and Université de Rennes I; 2005

[8] Christofides N, Mingozzi A, Toth P. State-space relaxation procedures for the computation of bounds to routing problems. Networks. 1981;**11**(2): 145-164. DOI: 10.1002/net.3230110207

[9] Mingozzi A. State space relaxation and search strategies in dynamic programming. In: Proceedings of the 5th International Symposium on Abstraction, Reformulation and Approximation. London, UK: Springer-Verlag; 2002. p. 51. Available from: http://portal.acm.org/citation.cfm?id=645848.758271

[10] Koch T, Martin A, Voß S. SteinLib: An updated library on Steiner tree problems in graphs. Technical Report ZIB-Report 00-37. Berlin: Konrad-Zuse-Zentrum für Informationstechnik Berlin; 2000. Available from: http://elib.zib.de/steinlib