Robust Routing in Changing Topologies

Pascal Berthomé[1], Lynda Gastal[1] and Abdel Lisser[1]

Laboratoire de Recherche en Informatique (LRI), CNRS UMR 8623, Université Paris-Sud, 91405, Orsay-Cedex, France,
{Pascal.Berthome,Lynda.Gastal,Abdel.Lisser}@lri.fr

Résumé De nombreux problèmes d'optimisation dans les réseaux peuvent être modélisés par des problèmes de plus courts chemins. Ce problème a été largement étudié et peut être résolu en temps polynomial lorsque la configuration du réseau est fixe. Cependant, dans les applications réelles, le réseau peut être sujet à des fluctuations. Celles-ci peuvent être la conséquence de pannes mais aussi de paramètres économiques. En effet, les opérateurs de télécommunications peuvent être amenés à étendre ou réduire leur réseau.

Pour prendre en compte ces changements, nous modélisons cette situation par une séquence de réseaux, chacun représentant l'état du réseau à un instant t. L'objectif consiste alors à trouver une séquence de plus courts chemins, un par état. Cependant, changer un chemin peut induire des perturbations locales (déconnexion temporaire de certains clients). Pour les modéliser, nous introduisons deux coûts supplémentaires sur les liens, l'un correspondant au coût de désinstallation de ce lien (s'il était actif) et l'autre au coût d'installation (s'il n'était pas actif). Le plus court chemin consiste alors à trouver les plus courts chemins successifs qui minimisent la somme de chacun de ces chemins plus le coût des pertubations.

Nous montrons dans ce rapport que ce problème est NP-complet même lorsque l'on considère uniquement deux états du réseau et que le premier réseau est un chemin. Nous proposons une relaxation combinatoire et une relaxation semidéfinie du problème. Des résultats numériques sont donnés sur des réseaux générés aléatoirement.

Abstract In real life applications, network structures are subject to uncertainty due to arc failure or economical phenomena. In this paper, we introduce a variant of the robust shortest path in which uncertainty is also related to the network structure. Thus, network evolution is represented by scenarii. The problem consists in finding a path between a given pair of nodes for each network state. Moreover, changing an existing path has a cost corresponding to deallocation and re-allocation costs. We first show that this problem is NP-complete, then that it can be formulated as a 0-1 quadratic program. We give a combinatorial and a semidefinite relaxations. Numerical experiments are given on set of real life networks instances.

1 Introduction

Shortest path problem is one of the most fundamental problems in network designs area. However, in this problem, the network configuration is deterministic and known in advance, whereas risk and uncertainty both involve randomness. This assumptions make the static model not adequate to model real life problems. Indeed, cost coefficients or network structure could vary due to physical or economical phenomena. Problems under uncertainty are known as *robust optimization problems.*

The Robust Shortest Path problem has been studied widely the last decade. Three mains models for dealing with uncertainty have been proposed. In the first one, data is structured by taking weight as interval range defined by known upper and lower bound and without assuming any probability distribution ([9,13,14]). In the second one, uncertainty is modelled by means of a discrete scenario set ([4, 11, 22]). In this case, scenario represents a potential realization of the arc weights which occurs with a given probability level. Finally, in the last one, arc weights uncertainty is represented by fuzzy numbers and possibility models are used ([3, 5, 10, 12, 16–18]).

Typically, uncertainty in the network structure is led to the uncertainty in the arc state. In Robust Shortest Path problem, the uncertainty is only related to link weights and does not take into account the possible evolutions of the network structure itself. Telecommunication network is based on the market growing evolutions. Thus, it may increase and sometimes decrease when telecom companies sell a part of their networks. An existing path between two endpoints then may appear less profitable after adding new connections or nodes, and may have to be reconsidered. Further costs may appear during this evolution, especially for resource deallocation and re-allocation. In this paper, we introduce a variant of the robust shortest path in which the uncertainty is directly related to the network structure. Thus, the network evolution is represented by scenarii. In this model, routing consists in finding an optimal sequence of paths taking into account the network structure evolution. Moreover, changing existing path may create local perturbation which can be formulated in terms of cost. Thus, a sequence of paths will be optimal if it minimizes its total weight cost plus the price of the number of perturbations. We call this problem Robust Routing in Changing Topologies, *RRCT* for short.

In Section 2, we introduce the problem and show that it is NP-complete even with simplified assumptions in Section 3. We give a simple n-approximation, where n is the number of considered scenarii. In Section 4, we give its mathematical formulation, a combinatorial and a semidefinite relaxations. In Section 5, numerical results are given on set of real like networks instances. Finally, we draw a conclusion and address some open questions in Section 6.

2 The Robust Routing in Changing Topologies Problem

2.1 Definition

We consider a set $\mathcal{G}$ of s weighted digraphs $G_i = (V_i, E_i)$, where V_i denotes the set of vertices, and $E_i \subset V_i \times V_i$ the set of edges. Moreover, we assume that $V_i \subseteq V_{i+1}$ and $E_i \subset E_{i+1}$. To simplify, we say that $G_i \subset G_{i+1}$, *i.e.*, G_i is sub-graph of G_{i+1}. We define E as E_s and V as V_s for G_s.

Beside its weight, an edge has an installation cost and an uninstallation cost. Formally, we define $w : E \to \mathbb{N}$, $\iota : E \to \mathbb{N}$ and $\delta : E \to \mathbb{N}$ as the weight function, the installation cost function and the uninstallation cost function respectively. The problem considered is to find a path X_i in each graph G_i between two given vertices o and d (note that o and d belong to V_i for all i). This sequence of paths $(X_i)_{i\in\{1,\ldots,s\}}$, called X, has to minimize the total cost, *i.e.*, the transition plus weight costs. The transition cost is composed of uninstallation cost of the subset of arcs of $X_i \setminus X_{i+1}$ plus installation cost of the subset of arcs of $X_{i+1} \setminus X_i$.

Any cost function $c \in (w, \iota, \delta)$ on the arcs of G is clearly additive. We define $c(X) = \sum_{x\in X} c(x)$ for any set of arcs X.

Formally, we have to minimize the function:

$$\varphi(X) = \sum_{i=1}^{s} w(X_i) + \iota(X_1) + \sum_{i=1}^{s-1} (\delta(X_i \setminus X_{i+1}) + \iota(X_i \setminus X_{i+1})) \qquad (1)$$

To establish the complexity of this problem, we formulate it as a decision problem:

Robust Routing in Changing Topologies Decision Problem ($RRCTd$)

Instance:

s graphs $G_i = (V_i, E_i)$ such that $G_i \subset G_{i+1}$

$w : E \to \mathbb{N}$, $\iota : E \to \mathbb{N}$, $\delta : E \to \mathbb{N}$

two vertices o and d; and an integer b

Question:

Does there exist a sequence X of s paths between o and d such that $\varphi(X) \leq b$?

We call $RRCT$ the corresponding optimization problem. We give an example of this problem.

2.2 Example

In this section, we give an example of $RRCT$ problem with only two graphs. Let G_1 and G_2 be the graphs defined by $V_1 = \{o, a, d\}$, $E_1 = \{(oa), (ad), (od)\}$, $V_2 = \{o, a, b, d\}$ and $E_2 = \{(oa), (ad), (od), (ob), (bd)\}$. Each edge is defined by

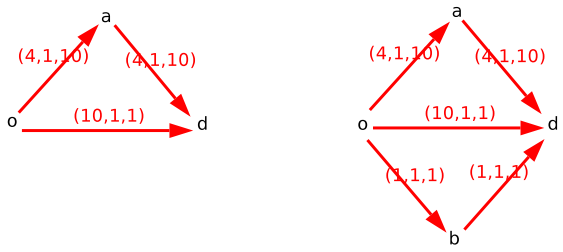

Figure 1. Example with 2 graphs

a triplet (w, ι, δ) representing its weight cost, its installation and uninstallation costs respectively. Thus, $(oa) = (ad) = (4, 1, 10)$, $(od) = (10, 1, 1)$ and $(ob) = (bd) = (1, 1, 1)$. Figure 1 illustrates these graphs.

For the sake of simplicity, we note C_1 the path $((od))$, C_2 the path $((oa), (ad))$ and C_3 the path $((ob), (bd))$. Let us evaluate the cost of each solution. If $X_1 = C_1$ and $X_2 = C_1$, $X_1 \setminus X_2 = X_1$ and $X_2 \setminus X_1 = X_2$. Then, the uninstallation cost is equal to $\delta(X_1 \setminus X_2) = \delta(X_1) = 1$. In the same way, installation cost is equal to $\iota(X_2 \setminus X_1) = \iota(X_2) = \iota((oa)) + \iota((ad)) = 2$. Thus $\varphi((X_1, X_2)) = \iota(X_1) + w(X_1) + \delta(X_1 \setminus X_2) + \iota(X_2 \setminus X_1) + w(X_2) = 22$. Each solution cost is reported in the table 1.

X_1	X_2	$w(X_1) + \iota(X_1)$	$\delta(X_1 \setminus X_2)$	$w(X_2)$	$\iota(X_2 \setminus X_1)$	$\varphi(X_1, X_2)$
C_1	C_1	11	0	10	0	21
C_1	C_2	11	1	8	2	22
C_1	C_3	11	1	2	2	16
C_2	C_1	10	20	10	1	41
C_2	C_2	10	0	8	0	18
C_2	C_3	10	20	2	2	34

Table 1. Costs of all solutions

We show that the optimal solution is (C_1, C_3) although the shortest path of graph G_1 is C_2. In the next section, we focus on the complexity of $RRCTd$ problem.

3 Problem Complexity

In this section, we focus on the complexity of $RRCTd$. We show that it is NP-complete even with only two scenarii and when the first graph is a path. The reduction is done from the Hamiltonian Circuit problem [6].

Theorem 1. *Problem $RRCTd$ is NP-complete even for a sequence of two graphs and when the first graph is a simple path.*

Proof. It is easy to see that $RRCTd$ belongs to NP. Indeed, the total cost of a given sequence of paths (X_1, X_2) is computed in polynomial time. To prove the NP-completeness, we reduce Hamiltonian Circuit problem into $RRCTd$ problem.

Hamiltonian Circuit Problem (HC)
Instance:
a graph $G = (V, E)$
Question:
Does G contain a Hamiltonian Circuit
(*i.e.* a simple circuit which passes through every vertex exactly once)?

Let $G = (V, E)$ be an instance of Hamiltonian Circuit problem. We derive two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ from G. Let n and m denote the number of vertices and the number of arcs of G.

The construction is depicted in Fig 2. We associate an arc s_k of weight 0 to each vertex v_k of V. Denote $head(s_k)$ and $tail(s_k)$, origin and tail of s_k respectively. For each $k \in \{1 \dots n\}$, we add an arc e_j of weight n from $head(s_k)$ to $tail(s_{k+1})$. Finally, we add an arc s'_1 of weight 0 such that $tail(s'_1) = head(e_n)$. We call o the head of s_1 and d the tail of s'_1. This achieves the construction of the graph G_1, which is a simple path.

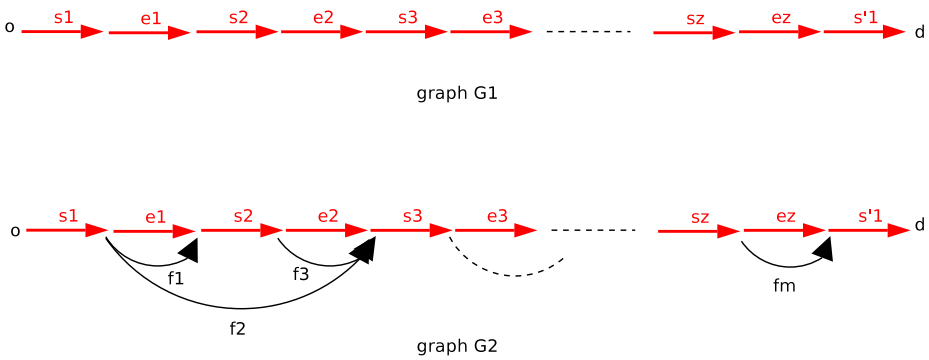

Figure 2. Scenarii for NP-completeness

The graph G_2 is obtained from G_1 by adding an arc f_i of weight 0 from $head(s_k)$ to $tail(s_\ell)$ if $\ell \neq 1$ and an arc linking vertex v_k to vertex v_ℓ exists in G. If $\ell = 1$, we add an arc f_i of weight 0 from $head(s_k)$ to $tail(s'_1)$. Let b be equal to n^2. All transition costs are equal to zero except the uninstallation cost of arc s_i which is equal to n^2.

	edge type		
	e_i	s_i	f_i
w	n	0	0
ι	0	0	0
δ	0	n^2	$\emptyset$

Table 2. Installation and uninstallation costs

Note that this construction can be done in polynomial time. Let (X_1, X_2) be a solution of the $RRCTd$ problem. According to (1), the cost of this solution is:

$$\varphi((X_1, X_2)) = w(X_1) + w(X_2) + \iota(X_1) + \delta(X_1 \setminus X_2) + \iota(X_2 \setminus X_1) \qquad (2)$$

As path X_1 is composed of n arcs s_k and n arcs e_j (recall that s_1 is repeated), its weight cost $w(X_1)$ is equal to n^2. Moreover, installation cost of arcs $(e_j)_{1 \leq j \leq n}$ and $(s_k)_{1 \leq k \leq n}$ is zero:

$$w(X_1) + \iota(X_1) = n^2$$

Let n_e be the number of arcs $(e_j)_{1 \leq j \leq n}$ used in the path X_2 and n_s be the number of arcs $(s_k)_{1 \leq k \leq n}$ not used in X_2. Thus, weight $w(X_2)$ is equal to $n_e \times n$. Installation cost of path X_2 is zero by construction and uninstallation cost is equal to $n_s n^2$ (recall that $\delta(s_i) = n^2$). Thus, we have:

$$w(X_2) + \delta(X_1 \setminus X_2) + \iota(X_2 \setminus X_1) = n_e n + n_s n^2$$

Therefore, the total cost of two paths sequence is given by:

$$\varphi((X_1, X_2)) = n_e n + (1 + n_s) n^2$$

This cost is less than or equal to b if and only if n_e and n_s are equal to zero. This means that path X_2 passes through every arcs $(s_k)_{1 \leq k \leq n}$ and no arc $(e_j)_{1 \leq j \leq n}$. Then, path X_2 is only composed of some arcs $(f_i)_{1 \leq i \leq m}$ and every arc $(s_k)_{1 \leq k \leq n}$.

If the instance of Hamiltonian Circuit Problem is satisfiable, we can find a path between s and t in graph G_2 using only arcs $(f_i)_{1 \leq i \leq m}$ and all arcs $(s_k)_{1 \leq k \leq n}$; arcs $(e_j)_{1 \leq j \leq n}$ are not used. Thus, the parameters n_e and n_s are equal to zero and the instance of $RRCTd$ is satisfiable.

Conversely, assume that the instance of problem $RRCTd$ is satisfiable. Then, there exists a path in G_2 using only subset of arcs $(f_i)_{1\leq i\leq m}$ and every arc $(s_k)_{1\leq k\leq n}$. We can see that the arcs used in X_2 gives a Hamiltonian Circuit in graph G. Indeed, it gives a path using only the arcs of the graph G and going through every vertex $(s_k)_{1\leq k\leq n}$ of graph G. □

Note that G_2 may not be simple. This can be easily avoided by adding new vertices in G_1, splitting the edges of type e into two, preserving the same global cost of the initial solution.

We consider the general case of sequence of s scenarii. One could consider large transition. In this case, if X_1^* denotes the shortest path of graph G_1, solution $S = (X_1^*, \ldots, X_1^*)$ is a feasible solution of $RRCT$ problem. Indeed, X_1^* belongs to each graph G_i for $i \in \{1, \ldots, s\}$. Moreover, S is optimal. This remark leads to the following lemma.

Lemma 1. *The solution S that consists in taking $(X_1^*, \ldots, X_1^*)$ provides a s-approximation for the RRCT problem.*

Proof. Let $\bar{S} = (X_1, \ldots, X_s)$ be the optimal solution of the $RRCT$ problem and $\varphi(\bar{S})$ its cost. Clearly, we have:

$$\varphi(\bar{S}) \geq w(X_1^*) + \iota(X_1^*)$$
$$\varphi(\bar{S}) \geq w(X_1^*)$$

The cost of the solution obtained by taking $(X_1^*, \ldots, X_1^*)$ is equal to:

$$\varphi(S) = s \times w(X_1) + \iota(X_1^*)$$
$$\varphi(S) \leq s \times \varphi(S_s)$$

□

Generally, this bound is tight. To illustrate it, we construct a sequence of s graphs in which all transition costs are equal to zero. The length of the shortest path of the first graph is equal to a constant k. The shortest path of the following graphs is always the same and has a length equals to 1. Thus, the cost of the solution obtained by this way is equal to sk. The optimal solution is composed of the shortest path of the first graph and the shortest path of the following graphs. Its cost is equal to $k + s - 1$.

4 Mathematical formulations

As shown before, $RRCTd$ is NP-complete. We focus on the associated optimization problem, called $RRCT$ problem. We first formulate $RRCT$ as a quadratic program. Then, we provide a combinatorial relaxation of $RRCT$. Finally, we propose a semidefinite relaxation of this problem. All these formulations will be used in Section 5.

4.1 Mathematical formulation of *RRCT*

In this section, we give a $0-1$ quadratic programming formulation of $RRCT$ problem.

Let x_{ij}^k be a binary variable equals to 1 if the arc from node i to j of graph G_k belongs to path X_k, and 0 otherwise. The objective function can be formulated as:

$$\varphi(X) = \overbrace{\sum_{(i,j)\in E_1}(w_{ij}+\iota_{ij})x_{ij}^1}^{\varphi_1} + \overbrace{\sum_{k=2}^{s}\sum_{(i,j)\in E_k} w_{ij}x_{ij}^k}^{\varphi_2} + \sum_{k=1}^{s-1}\left(\overbrace{\sum_{(i,j)\in E_k}\delta_{ij}x_{ij}^k(1-x_{ij}^{k+1})}^{\varphi_3} + \overbrace{\sum_{(i,j)\in E_{k+1}}\iota_{ij}x_{ij}^{k+1}(1-x_{ij}^k)}^{\varphi_4}\right) \quad (3)$$

where φ_1 corresponds to the weight cost plus the installation cost of path X_1; φ_2 is the weight cost of paths $(X_i)_{2\leq i\leq s}$; φ_3 is the uninstallation cost of the arcs of path X_k which are not in path X_{k-1}. In the same way, φ_4 is the installation cost of arcs of path X_{k+1} which are not in X_k. To express that the solution is composed by a path for each graph of the sequence, we define $F^k(u,v)$ the flow conservation constraints between nodes u and v in V_k by:

$$\begin{cases} F^k(u,v): \sum_{j:(i,j)\in E_k} x_{ij}^k - \sum_{j:(j,i)\in E_k} x_{ji}^k - b = 0 \\ where\ b = \begin{cases} 1\ if\ i=u, \\ -1\ if\ i=v, \forall i\in V_k \\ 0\ otherwise \end{cases} \end{cases}$$

Problem $RRCT$ can be formulated as:

$$(RRCT)\begin{cases} \min \varphi(X) \\ \text{s. t.} \\ \quad F^k(o,d) \quad \forall k\in\{1\dots n\} & (4) \\ \quad \sum_j x_{ij}^k \leq 1 \quad \forall i\in V_k \quad \forall k\in\{1\dots s\} & (5) \\ \quad x_{pq}^k F^k(o,p) \quad \forall p,q\in V_k \quad \forall k\in\{1\dots s\} & (6) \\ \quad x_{pq}^k F^k(q,d) \quad \forall p,q\in V_k \quad \forall k\in\{1\dots s\} & (7) \\ \quad x_{ij}^k\in\{0,1\} \quad \forall i,j\in V_k \quad \forall k\in\{1\dots s\} \end{cases}$$

Constraints (4) represent the flow conservation constraints. They ensure that for each scenario k in $\{1,\dots,n\}$ there is a path between o and d. Constraints (5) limit the degree of each node of the path to 1 to avoid potential cycles. Constraints (6) and (7) ensure that the obtained solution is acyclic. Indeed,

without these constraints, we show in the next section that the solution could contain independent cycles. We now provide a combinatorial relaxation of $RRCT$ problem.

4.2 Combinatorial Relaxation of *RRCT* problem

We call a combinatorial relaxation a new probem formulation composed of a subset of original problem constraints. It provides easily lower bounds for the original problem [15]. In this section, we present a combinatorial relaxation of $RRCT$ problem. Constraints (6) and (7) are removed from the model. We next show that it provides a relaxation of $RRCT$ problem. Let consider the following program :

$$(RRCTr)\begin{cases}\min \varphi(X) \\ \text{s. t.} \\ \quad F^k(o,d) \quad \forall k \in \{1 \dots s\} \\ \quad x_{ij}^k \in \{0,1\} \quad \forall i,j \in V_k \quad \forall k \in \{1 \dots s\}\end{cases} \tag{8}$$

Let's show in the following that $RRCTr$ only provides a relaxation of our problem. Let's consider $RRCT$ instance used in the proof of Theorem 1. It is composed by G_1 and G_2 which are simple path and general graph respectively. In this case, all $(x_{ij}^1)_{(ij)\in E_1}$ are known (all are equal to 1) and the objective function is not any more a quadratic function, but a linear function:

$\varphi(X_2) = \sum_{(i,j)\in E_1}(w_{ij}+\iota_{ij})x_{ij}^1+\sum_{(i,j)\in E_2} w_{ij}x_{ij}^2+\sum_{(i,j)\in E_1}\delta_{ij}x_{ij}^1(1-x_{ij}^2)+ \sum_{(i,j)\in E_2}\iota_{ij}x_{ij}^2(1-x_{ij}^1)$

$$\varphi(X_2) = \sum_{(i,j)\in E_1}(w_{ij}+\iota_{ij}) + \sum_{(i,j)\in E_2} w_{ij}x_{ij}^2 + \sum_{(i,j)\in E_1}\delta_{ij}(1-x_{ij}^2) + \sum_{(i,j)\in E_2}\iota_{ij}x_{ij}^2$$

Furthermore, all the constraints are linear and, since they represent flow constraints, the associated matrix is totally unimodular. Thus, the $RRCTr$ is polynomially solvable.

Let us consider now the $RRCT$ instance obtained from the proof of Theorem 1, where the graph considered for the hamiltonicity test is given by Figure 3. Obviously, this graph is not Hamiltonian. The optimal solution of $RRCTr$ problem is X_2 ={(5,6),(6,7),(7,8),(8,9),(9,10), (10,5) } ∪ { (1,2), (2,11), (11,12), (12,3), (3,4), (4,13), (13,14) } leading to the solution given by Fig 4.

According to the objective value and its expression in the original problem, graph G_2 would have been Hamiltonian, which is not true. Thus, $RRCTr$ is only a relaxation of $RRCT$ problem and its resolution can only provide lower bounds. Note that in this particular case, $RRCT$ solves the longest cycle of the graph G containing a given node (here a) whereas $RRCTr$ finds only a maximal

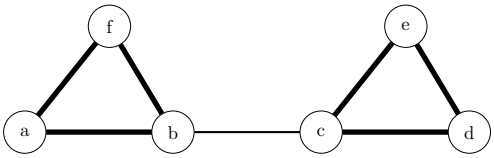

Figure 3. corresponding solution in graph G

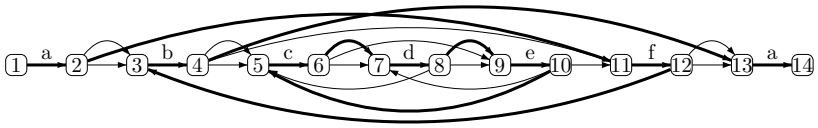

Figure 4. solution of $RRCTr$ in graph G_2

covering by cycles of G. Based on this, we note that the total cost of $RRCTr$ is equal to 36 and $RRCT$ total cost is equal to 144. It does not provide some constant factor of approximation for the $RRCT$ problem. However, we show in Section 4.3 that this relaxation may be useful.

4.3 Relationship between *RRCT* and *RRCTr*

It is easy to see that Constraints 6 and 7 are redundant if the optimal solution of $RRCTr$ problem does not contain any independant cycle. In this case, the $RRCTr$ problem is sufficient to solve the original problem.

Theorem 2. *If the optimal solution of RRCTr problem is composed of n paths between s and t, then this solution is optimal for RRCT problem.*

Proof. Let $\tilde{\mathrm{S}}$ be the optimal solution of $RRCTr$ and S the optimal solution of $RRCT$. It is easy to see that $\tilde{\mathrm{S}}$ is an upper bound of $RRCT$ since it is a feasible solution of $RRCT$ problem. It is also a lower bound of $RRCT$ since it is a feasible solution of $RRCTr$. □

Heuristically, if P_i denotes the part of this solution which is a path between s and t in graph G_i, and C_i denotes the part of the solution which is an independent cycle in graph G_i, the solution only composed of $(P_i)_{i\in 1\ldots n}$ is optimal if all the $(C_i)_{i\in 1\ldots n}$ are empty sets.

However, this algorithm often gives very bad lower bounds. Therefore, $RRCT$ can be solved either by linearizing its quadratic terms or by using semidefinite programing relaxation (SDP).

4.4 Semidefinite Relaxation of *RRCTr*

Semidefinite relaxation is a particular case of convex programming. It consists in optimizing a linear function subjet to linear constraints. Thus, it is a gener-

alization of linear programming in which the vector of variables is replaced by a semidefinite positive symmetric matrix.

The last decade, semidefinite programming made possible to obtain bounds to NP-hard problems in polynomial time. The seminal work of Goemans [7] propose a .87856-approximation algorithm for Max Cut problem based on semidefinite program. Recents studies show that semidefinite programs could be useful for solving quadratic 0-1 programming, as Quadratic Knapsack problem ([8]). For more details on the semidefinite programming, see [19, 20].

Solve a semidefinite program could be done by using interior point or spectral bundle method. The second way need to know a bound on the trace of the solution to converge polynomially. There exists free solvers using interior point method like [1, 2].

A semidefinite program is defined by:

$$SDP \begin{cases} \min C \bullet X \\ s.t. \\ \quad tr(A_i X) = b_i \ \forall i \in \{1, ..., m\} \\ \quad X \succeq 0 \end{cases}$$

where C, A_i and X are $n \times n$ matrices and X is symmetric. $X \succeq 0$ means X is positive semidefinite matrix *i.e.* $\forall z \in \mathbb{R}^n, z^t X z \geq 0$ (see [21] for more details). Operation $\bullet$ is the inner product of matrices and is defined by :

$$C \bullet X = \sum_{i=1}^{n} \sum_{i=1}^{n} C_{ij} X_{ij} = Tr(CX)$$

We proposed a SDP relaxation for solving $RRCTr$. We note $RRCT_{sdp}$ the corresponding problem.

Let first write our problem in its matricial form.

$$(RRCT) \begin{cases} \min \ x^t P x \\ s.t. \\ \quad A^k x = c^k \ \forall k \in \{1, ..., s\} \\ \quad x \in \mathbb{R}^{\sum_{k=1}^{s} n_k} \end{cases}$$

where $x = (x^1_{12}, \ldots, x^1_{n_1, n_1 - 1}, \ldots, x^s_{12}, \ldots, x^s_{n_s, n_s - 1})$, A^k is the incidence matrix of graph G_k and c^k is a vector of $\mathbb{R}^{n_k}$ is defined by:

$$c^k_i = \begin{cases} 1 & \text{if } i = o \\ -1 & \text{if } i = d \\ 0 & otherwise \end{cases}$$

and P is defined by:

$$p_{i,i} = \begin{cases} w_{ij} - \delta_{ij} & \text{if } i < n_1 \\ w_{ij} - \delta_{ij} - \iota_{ij} & \text{if } n_1 + 1 < i < \sum_{k=1}^{s-1} n_k \\ w_{ij} - \iota_{ij} & \text{if } i > \sum_{k=1}^{s-1} n_k \end{cases}$$

and

$$p_{i,j} = \frac{-\delta_{ij} - \iota_{ij}}{2} \; for \; \begin{cases} \sum_{k=1}^{l} n_k < i < \sum_{k=1}^{l+1} n_k \\ \sum_{k=1}^{l-1} n_k < j < \sum_{k=1}^{l} n_k \end{cases} \text{ for } 2 < l < s-1$$

We can now formulate the semidefinite relaxation of our problem. It is given as follows:

$$(RRCTr)_{sdp} \begin{cases} \min \; Tr(\bar{W}X) \\ s.t. \\ \; Tr(\bar{A}_k X) = c_i^k \; \forall 1 \leq k \leq s \\ \; X \succeq 0 \end{cases}$$

where $X = \begin{pmatrix} xx^t & x^t \\ x & 1 \end{pmatrix}$, $\bar{W} = \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix}$,

$\bar{A}_k$ is defined by:

$$\bar{A}_k = \begin{pmatrix} 0 & & & & & \\ & \ddots & & & & \\ & & 0 & & & \\ & & & B_k & & \\ & & & & 0 & \\ & & & & & \ddots \\ & & & & & & 0 \end{pmatrix}$$

and $diag(B^k) = (L_1(A^k), \ldots, L_{n_k}(A^k))$ where $L_i(A^k)$ is the i^{th} line of the matrix A^k.

Recall m_i and n_i are the number of edges and nodes of the graph G_i respectively. In the original problem, the number of variables is $\sum_{i=1}^{n} m_i$. We also have $\sum_{i=1}^{n} m_i$ flow conservation constraints. In the $(RRCT_{sdp})$ problem, the number of variables is equal to $(\sum_{i=1}^{n} m_i)^2$. The number of constraints is equal to $2\sum_{i=1}^{n} m_i$. This formulation is used in Section 5.

5 Numerical Results

In this section, we present the results of some numerical experiments. The framework of the experiment is detailed in Section 5.1. We present the results for the continuous relaxation in Section 5.2, and for the SDP relaxation in Section 5.3.

5.1 Networks instances

The experimental data instances consist of a growing sequence of real like graphs generated by the authors. The transition costs (installation and uninstallation) are constant for all edges of all the graphs. The graphs parameters are given in Table 3.

Moreover, we run our relaxations on problems composed of two and three graphs respectively. The problem, formulated as a quadratic program, is solved by CPLEX 8.1 for the continuous relaxation and by DSDP4.7 for semidefinite relaxation. The tests have been carried out on PC Intel Pentium 4 (CPU 2.60 GHz) using C++ code.

5.2 Continuous relaxation on Real Life Graphs

In this section, we test the influence of the graph size on the quality of the solutions. We focus on graphs of different sizes (see Table 3). Each instance is composed by a set of (two or three) increasing size graphs. For example, we associate net50 with net4, net10 and net20. Moreover, we calculate paths between different sources o and destinations d in order to have various examples. Finally, we run our relaxations on instances composed of two graphs and three graphs respectively.

Name	Nodes	Edges	Density (%)
net4	4	6	50
net10	10	45	50
net20	20	190	50
net50	50	1219	49.75
net100	100	4826	48.74
net150	150	10724	47.98
net200	200	18587	46.70
net250	250	28827	45.88

Table 3. Size of graphs

Results on the instances of two real life graphs. We test the quality of the solution obtained by a continuous relaxation of the integer quadratic program. Results are presented in Table 4. The first column gives the data instances. The second column corresponds to the average gap while the third column contains the maximal gap. Both gaps are obtained with respect to feasible integer solutions given by CPLEX. The last column presents the CPU Time. The results are given according to the maximal size of the two graphs. Recall that if the size of the largest graph is 50, this means that it has been associated with graphs of size 4, 10 and then 20.

Name of the largest graph	Average Gap (%)	Maximum Gap (%)	CPU Time (s)
net20	10.87	13.02	0.06
net50	9.42	16.15	0.91
net100	7.17	16.38	12.76
net150	5.76	16.43	117.53
net200	4.81	16.49	334.98
net250	4.13	16.53	839.96

Table 4. Average Gap and Maximum Gap with a continuous relaxation on two graphs

As illustrated in Table 4, the maximum gap for a continuous relaxation varies according to the maximal size of the two graphs. For real life graphs, the maximal gap goes from 13% up to 16.53%. The average gap decreases for large instances and goes from 10.87% down to 4.13%.

Results on instances of three real life graphs. We now test the impact of the number of graphs on the quality of the solution. These results are presented by Table 5. They are given according to the maximal size of the three graphs. As for instances of two graphs, we associate each graph to the other smaller graphs. For example, we put a graph of 50 nodes with graphs of 4 and 10 nodes, 4 and 20 nodes and finally 10 and 20 nodes. The columns of Table 5 give the average and the maximal gaps with the integer solution given by CPLEX and the CPU Time.

In the Table 5, we show that the maximum gap with a continuous relaxation varies according to the maximal size of the three graphs. For three real life graphs, the maximal gap goes from 11.06% up to 14.78%. The average gap decreases for large instances and goes from 8.89% down to 7.28%. Our results show that the maximal gap obtained for tests composed of two graphs is less than the one composed of three graphs.

Name of the largest graph	Average Gap (%)	Maximum Gap (%)	CPU Time (s)
net50	8.89	11.06	2.34
net100	8.51	14.50	40.16
net150	7.67	14.73	250.29
net200	7.28	14.78	1043.12

Table 5. Average Gap and Maximum Gap with a continuous relaxation on three real life graphs

5.3 Semidefinite relaxation on Real Life Graphs

The results obtained by a semidefinite relaxation of problem $RRCTr$ are given in Table 6.

Name of the largest graph	Average Gap (%)	Maximum Gap (%)	CPU Time (s)
net10	0.00	0.00	21.95
net20	0.53	3.21	520.90

Table 6. Average Gap and Maximum Gap with a semidefinite relaxation on two graphs

DSDP algorithm is based on interior point method and so, does not support a large number of variables. Here, DSDP can only solve instances of less than 50 nodes. However, we can note that the maximal gap is samll and the average gap is less than 1%, which out performs linear relaxation.

6 Conclusion

In this paper, we show that the Robust Routing in Changing Topologies problem is NP-complete and present a formulation based on 0-1 quadratic program as well as a combinatorial and a semidefinite relaxations of this problem. We have proposed numerical results on different graphs with up to 250 nodes.

However, some questions remain open. Indeed, the problem is polynomial when the transition costs are equal to zero or infinite. On the other hand, we have seen in Theorem 1 that the problem becomes difficult when the transition costs have a large amplitude. Further research are going on the impact of the transition functions on the complexity of the problem, e.g., when these functions are constant.

References

1. S.J. Benson and Y. Yinyu. DSDP5: Software for semidefinite programming. Technical Report ANL/MCS-P1289-0905, Mathematics and Computer Science Division,

Argonne National Laboratory, Argonne, IL, September 2005. Submitted to ACM Transactions on Mathematical Software.
2. B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11:613–623, 1999.
3. T. Chuang and J. Kung. The fuzzy shortest path a length and the corresponding shortest path in a network. *Computer Operations Research*, 32:1409–1428, 2005.
4. L.C. Dias and J.N. Climaco. Shortest path problems with partial informations: models and algorithms for detecting dominance. *European Journal of Operation Research*, 121:16–31, 2000.
5. D. Dubois and H. Prade. *Fuzzy sets and Systems: Theory and Applications.* New York, Academic Press, 1980.
6. M.R. Garey and D.S. Johnson. *Computers and Intractablility: A guide to the theory of NP-Completeness.* W.H. Freeman, 1979.
7. M.X. Goemans and D. P. Williamson. Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *J. Assoc. Comput. Mach.*, 42:1115–1145, 1995.
8. C. Helmberg, F. Rendl, and R. Weismantel. A semidefinite programming approach to the quadratic knapsack problem. *Journal of Combinatorial Optimization*, 4(2):197–215, 2000.
9. O.E. Karasan, M.C. Pinar, and H. Yaman. The robust shortes path problem with interval data. Technical report, Department of Industrial Engineering, Bilkent University, 06553 Bilkent, Ankara, Turkey, 2001.
10. C.M. Klein. Fuzzy shortest path. *Fuzzy Sets and Systems*, 39:27–41, 1991.
11. P. Kouvelis and G. Yu. Robust discrete optimisation and its applications. Kluwer Academic Publishers, 1997.
12. K. Lin and M. Chern. The fuzzy shortest path problem and its most vital arcs. *Fuzzy Sets and Systems*, 28:343–353, 1994.
13. R. Montemanni and L.M. Gambardella. An algorithm for the relative robust shortest path problem with interval data. Technical report, IDSIA-05-02,IDSIA,MannoLugano, Switzerland, February 2002.
14. R. Montemanni and L.M. Gambardella. An exact algorithm for the robust shortest path problem with interval data. *Computer Operations Research*, 31:1667–1680, 2004.
15. G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* J. Wiley, New York, 1988.
16. S. Okada. Fuzzy shortest path problems incorporating interactivity among paths. *Fuzzy Sets and Systems*, 142(3):335–357, 2004.
17. S. Okada and M. Gen. Fuzzy shortest path problem. *Computer and Industrial Engineering*, 27:465–468, 1994.
18. S. Okada and T. Soper. A shortest path problem on a network with fuzzy length. *Fuzzy Sets and Systems*, 109:129–140, 2000.
19. M. J. Todd. Semidefinite optimization. *Acta Numerica*, 10:515–560, 2001.
20. L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.
21. H. Wolkowicz, R. Saigal, and L. Vandengerghe. *Handbook of Semidefinite Programming, Theory, Algorithms, and Applications.* Kluwer Academic Publishers, 2000.
22. G. Yu and J. Yang. On the robust shortest path problem. *Computer and Operations Research*, 25:457–468, 1998.