

Routing Reconfiguration/Process Number: Networks with Shared Bandwidth.

David Coudert — Dorian Mazauric — Nicolas Nisse

N° 6790

January 2009

Thème COM



*Rapport
de recherche*



Routing Reconfiguration/Process Number: Networks with Shared Bandwidth.

David Coudert^{*†}, Dorian Mazauric^{*†}, Nicolas Nisse^{*†}

Thème COM — Systèmes communicants
Projets Mascotte

Rapport de recherche n° 6790 — January 2009 — 12 pages

Abstract:

In this paper, we address the problem of scheduling the switching of a set of connection requests one after the other from current routing to another pre-determined routing. We propose a model that handles requests using only a constant fraction of the bandwidth of a link, thus generalizing the model proposed in [2, 4] for WDM networks. Our main result is the proof that the problem of deciding whether it exists a scheduling of the rerouting of connection requests without traffic interruption is NP-complete even if requests use the third of the bandwidth of a link. Note that the problem is polynomial when the bandwidth of a link cannot be shared [2].

Key-words: Connection oriented networks, Reconfiguration.

This work was partially funded by Région PACA and by European project IST FET AEOLUS.

* MASCOTTE, INRIA, I3S, CNRS, Univ. Nice Sophia, Sophia Antipolis, France.

† `firstname.lastname@sophia.inria.fr`

Reconfiguration de routage/ process number: prise en compte du partage de la bande passante

Résumé :

Dans cet article; nous étudions le problème de reconfiguration qui consiste à modifier des connexions de manière successive pour passer du routage courant à un autre routage pré-calculé. Nous proposons un modèle qui prend en compte les requêtes utilisant une fraction de la bande passante d'un lien. Ceci généralise le modèle proposé pour les réseaux WDM dans [2, 4]. Nous prouvons principalement la NP-complétude du problème qui consiste à décider si il existe un ordonnancement pour le reroutage des requêtes sans aucune interruption du trafic. Notons que ce problème est NP-complet sitôt que les requêtes utilisent un tiers de la bande passante des liens, alors qu'il peut être résolu en temps linéaire lorsque la bande passante des liens ne peut être partagée.

Mots-clés : Réseaux en mode connecté, Reroutage

1 Introduction

In connection oriented networks such as WDM (Wavelength Division Multiplexing), SONET (Synchronous optical networking), or MPLS (Multiprotocol Label Switching) networks, the traffic pattern evolves constantly due to the variation of the demands (on-demand TV, mobile Internet, peer-to-peer). In this context, new connection requests are routed greedily using available resources without interfering with the routing of pre-established connections. Ending connections are removed similarly. However, such policy leads to a poor usage of resources and so higher blocking probability: new connection requests might be rejected although the network has enough resources to serve all the traffic, up to the rerouting of some existing connections (see Fig. 1). Therefore, it is important to regularly reconfigure the routing of established connections in order to optimize the usage of network resources.

The routing reconfiguration problem consists in two phases: (i) the determination of a new routing improving the usage of network resources, and (ii) the migration of connections from current routes to new routes. In this paper, we concentrate on the second phase that consists in switching existing connections one after the other from the current routing to a new pre-computed routing. Thus, our study is independent of the destination routing and its computation is not considered here.

To switch an established connection from a route to another, one has to ensure that destination resources are available. For instance, in Fig. 1, connection d must be switched before connection a . To model all dependencies between connections in the reconfiguration phase, we use the notion of *dependency digraph* proposed in [4] for WDM networks. In such digraph, a vertex corresponds to a request and an arc from u to v indicates that connection v must be switched before connection u . Clearly, when the dependency digraph is a directed acyclic graph (DAG), the scheduling of the switchings is straightforward, starting from the leaves. However, cyclic dependencies may exist and so the dependency digraph may contain cycles. In such cases, reconfiguration is feasible only if we allow to temporarily interrupt some connections in order to break the dependency cycles. The objective is thus to minimize the number of connections that will be simultaneously interrupted. Initially, this problem has been studied in [4] with an heuristic algorithm. Later, the network reconfiguration problem has been defined in terms of *process number* [2], an analogous of cops-and-robber games [5, 3].

More formally, given a network modeled by a digraph $G = (V, E)$ and an instance \mathcal{I} of connection requests $r = (x, y) \in V^2$, a *routing* \mathcal{R} is a set of directed paths associated to the requests, one directed path from x to y in G for each request $(x, y) \in \mathcal{I}$. So $\mathcal{R}(r)$ is the route of request r in G . The *routing reconfiguration problem* consists in switching connection requests one after the other from an initial routing \mathcal{R}_1 to a precomputed routing \mathcal{R}_2 in such a way that the smallest number of requests are simultaneously interrupted. An important assumption is that, once a request has been (re)routed, it cannot be moved anymore. In this model, each link of G has capacity one, i.e., each directed link may be used by at most one request. In [2], the dependency digraph $D = (W, A)$ of \mathcal{R}_1 and \mathcal{R}_2 is defined as follows. W is the set of requests $r \in \mathcal{I}$ with different routes in \mathcal{R}_1 and \mathcal{R}_2 . Moreover, there is an arc from request u to request v if there exists a link e of G that

belongs to both paths $\mathcal{R}_2(u)$ and $\mathcal{R}_1(v)$. The *process number* of D , denoted by $\text{pn}(D)$, is a digraph invariant that equals the smallest number of requests that have to be simultaneously interrupted during the reconfiguration phase. In particular, $\text{pn}(D) = 0$ if and only if D is a DAG. Using this formulation, polynomial-time algorithms have been designed to decide whether a digraph has process number 1 or 2 [2]. However, the problem of computing the process number of a digraph has been shown NP-complete and difficult to approximate [2]. Also, heuristic algorithms based on the process number have been proposed in the context of WDM networks [1].

However, this model considers that a connection request uses all the bandwidth of a link (e.g. wavelength) and it is too limited to handle cases in which a request uses only a fraction of the bandwidth of a link (e.g. MPLS, SONET and so traffic grooming). For example, suppose that all requests have bandwidth requirement 1 and that links have capacity 2. Moreover, let us assume that a link e is used by only one request u in R_1 and will be used by two requests v and w in the new routing R_2 . The model of [2, 4] requires the extra knowledge of which one of v or w will use the unit of bandwidth that will be released by u . But this information is not always known in advance and may even be irrelevant. Indeed, since the bandwidth of link e is shared, this is a scheduling decision to know which one of v or w has to wait until the rerouting of u before being allowed to use the capacity released by u on the link e . Also, the model has to be extended to integrate such a situation.

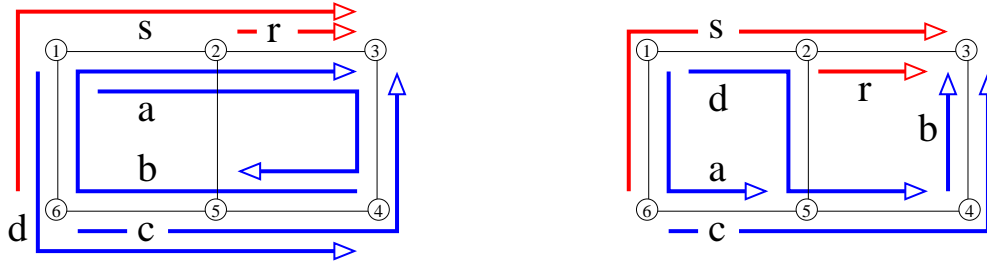
In this paper, we propose a model generalizing [2, 4] to handle cases in which a request uses only a fraction of the bandwidth of a link. We present the modeling in Section 2. Then, in Section 3, we show that the problem of deciding whether there exists a scheduling of the rerouting without traffic interruption is NP-complete. Finally, we conclude this paper in Section 4 with a list of open problems.

2 Modeling

As said in the introduction, in this paper we generalize the routing reconfiguration problem studied in [1, 2, 4] to the case in which each connection request uses a fixed fraction of the bandwidth of a link. Also, in the following, we consider that connection requests have bandwidth requirement 1 and that each link e of the network has capacity $c(e) \in \mathbb{N}$. Thus, [1, 2, 4] consider the case $c(e) = 1$ for any link e . Multiple connection requests between two nodes of the network are allowed and considered as distinct requests.

Given a network modeled by a digraph $G = (V, E)$ with some capacity function $c : E \rightarrow \mathbb{N}$ over the arcs of G and an instance \mathcal{I} of connection requests $r = (x, y) \in V^2$. A *routing* \mathcal{R} is *valid* if for any arc $e \in E$, e belongs to at most $c(e)$ paths in \mathcal{R} .

The (generalized) *routing reconfiguration problem* consists in switching connection requests one after the other from an initial valid routing \mathcal{R}_1 to a precomputed valid routing \mathcal{R}_2 . A request r can be switched from its initial route in \mathcal{R}_1 to its new route in \mathcal{R}_2 only when the required capacity is available along the path. Clearly, some requests may have to be switched before r to release some capacity, and these scheduling constraints are modeled by a *dependency multi-digraph*.



(a) Routing \mathcal{R}_1 of requests a, b, c, d . The new requests r and s cannot be satisfied.

(b) Routing \mathcal{R}_2 satisfying requests a, b, c, d, r, s .

Figure 1: Fig. 1(a) is a grid where links have capacity two in each direction. New requests r and s cannot be accepted, although the routing of Fig. 1(b) is possible.

The dependency multi-digraph $D = (W, A)$ with labeled arcs is built as follows. There is a vertex v_r in W for each request $r \in \mathcal{I}$ with different routes in \mathcal{R}_1 and \mathcal{R}_2 (requests that must be switched). Furthermore, to model the available capacity $q(e)$ of link $e \in E$ in routing \mathcal{R}_1 ($q(e)$ equals $c(e)$ minus the number of requests using e in \mathcal{R}_1), there is a vertex in W for each unit of capacity available on link e . Let $\mathcal{V}(e) = \{v_e^1, \dots, v_e^{q(e)}\}$ be the set of these $q(e)$ vertices. We denote $\mathcal{V} = \cup_{e \in E} \mathcal{V}(e)$ the set of such vertices that we call *virtual vertices*. Now, there is an arc labeled e from vertex u to vertex v if there is a link e of G that belongs to $\mathcal{R}_2(u)$ and v is a virtual vertex in $\mathcal{V}(e)$, or if $e \in \mathcal{R}_2(u) \cap \mathcal{R}_1(v)$ with v corresponding to a request. In other words, all arcs leaving a vertex u with label e represent the possible resources that may be used by request u to traverse the link e in the final routing.

For example, in Fig. 1 the routing of requests a, b, d is different in \mathcal{R}_2 than the routing in \mathcal{R}_1 , thus allowing to route requests r and s in \mathcal{R}_2 . The dependency multi-digraph D , represented in Fig. 2, has vertex set $\{v_a, v_b, v_d, v_{1,6}, v_{4,3}, v_{2,5}, v'_{2,5}\}$, where v_a corresponds to request a and $v_{i,j}$ corresponds to a free unit of capacity on link (i, j) (2 units of free capacity on link $(2, 5)$). There are arcs with same label in D from v_a to both v_d and $v_{1,6}$ since, on link $(1, 6)$, Request a can use in \mathcal{R}_2 either the free unit of capacity, $v_{1,6}$, or the unit of capacity that will be released by d if it is rerouted before a .

Now, we can see from Fig. 2(a) that request b can be switched at any time since its single out-neighbor is a free unit of capacity. Furthermore, request d needs a free unit of capacity on link $(2, 5)$, so either $v_{2,5}$ or $v'_{2,5}$, and it needs a unit of capacity on link $(1, 2)$ where both units of capacity are currently used by requests a and b . Thus either request a or b must be switched before d .

Let us continue with some simple remarks.

- The out-degree of a virtual vertex in D is 0 (see Fig. 2(a)).

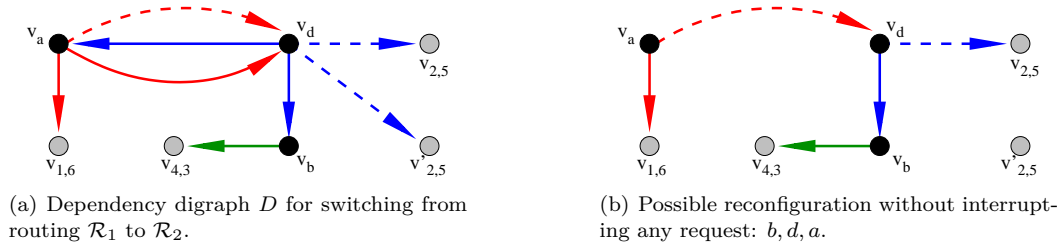


Figure 2: Dependency digraph D and possible scheduling. Arcs with same color represent dependencies on link e and so arcs with label e . Virtual vertices in grey.

- For any link $e \in E(G)$, let D_e be the subgraph of D induced by all arcs with label e . Thus, $D_e = (X \cup Y, F)$ is a complete bipartite digraph from the requests that will use e in \mathcal{R}_2 (the vertices in X) to the requests that use e in \mathcal{R}_1 and the corresponding virtual vertices (if any) (the vertices in Y). Note that $|X| \leq |Y|$. We call such a graph a *nice bipartite digraph*.

The main difficulty of the problem comes from the fact that the final routing \mathcal{R}_2 only gives the links used by the requests but not the unit of capacity. Therefore, we have some choice when performing the scheduling of the reconfiguration phase.

More precisely, when moving a request r to $\mathcal{R}_2(r)$ and for any $e \in \mathcal{R}_2(r)$, we must decide which unit of capacity of e will be used by r . This corresponds to choose the out-arc of r labeled e that will be “used” in the subgraph D_e .

In other words, for any $e \in E(G)$, a reconfiguration gives a maximal matching of D_e (recall that the vertex set of D_e is denoted $X \cup Y$). Indeed, for any label of the arcs, exactly one out-arc with this label is chosen for any request in X , and two arcs with same label cannot be incident to the same vertex of Y because it would have meant that two requests use the same resource to traverse the link e . Reciprocally, for any edge $e \in E(G)$ (i.e., for any label of the arcs of D), let M_e be a maximal matching of D_e . Then, the subgraph of D induced by $\bigcup_{e \in E(G)} M_e$ corresponds to a reconfiguration compatible with the routing \mathcal{R}_2 and the network link capacities. For instance, Fig. 2(b) represents a choice of such a matching in the conflict digraph depicted in Fig. 2(a).

The above discussion leads us to our main definition. Let D be a multi digraph with labeled edges, such that the edges labeled with same label e induce a nice bipartite digraph D_e (we say that D is labeled in a *nice way*). For any label e , let \mathcal{M}_e be the set of all maximal matching in D_e . Let \mathcal{D} be the set of all digraphs that can be obtained by choosing one maximal matching M_e in \mathcal{M}_e for every label e and by considering the subgraph induced by $\bigcup_e M_e$.

Definition 1. *The generalized process number of D , denoted by $\text{gpn}(D)$, equals the minimum of the process number of D' over all D' in \mathcal{D} .*

By definition, the generalized process number of a dependency multi-digraph of two routings \mathcal{R} and \mathcal{Q} equals the smallest number of requests that must be simultaneously interrupted during the reconfiguration phase from \mathcal{R} to \mathcal{Q} .

Note that, the generalized process number is an invariant of multi-digraph with labeled arcs. In particular, if all arcs of D receive distinct labels (this occurs when all links of G have capacity one), the generalized process number of D equals its process number [1, 2]. Indeed, in this case, any subgraph D_e induced by the arcs of D with same label e ($e \in E(G)$) consists of a single edge, and, thus $\mathcal{D} = \{D\}$.

3 Some results

We first give a simple relationship between the routing reconfiguration problem in networks where links have some capacity, and the same problem in networks where all edges have capacity one.

Let $G = (V, E)$ be a directed graph with some capacity function $c : E \rightarrow \mathbb{N}$ over the arcs of G . We set G^- be the same digraph in which all arcs have only capacity one. We consider the routing reconfiguration problem from a valid routing \mathcal{R} to another valid routing \mathcal{Q} in G , such that both routings are the superimposings of elementary routings. More precisely, let $\mathcal{R} = \cup_{i \leq p} \mathcal{R}_i$ and $\mathcal{Q} = \cup_{i \leq p} \mathcal{Q}_i$ such that, for any $i \leq p$, \mathcal{R}_i and \mathcal{Q}_i are two valid routings in G^- relative to the same set of requests. Let D be the dependency multi-digraph of \mathcal{R} and \mathcal{Q} , and, for any $i \leq p$, let D_i be the dependency digraph of \mathcal{R}_i and \mathcal{Q}_i . In this case, the next proposition gives an easy upper bound on the generalized process number. However, as shown in Fig. 3, the process number of such a multi-digraph D may be strictly less than this upper bound.

Proposition 2. *The generalized process number of D is at most $\max_{i \leq p} \text{pn}(D_i)$.*

Sketch of the Proof. It is sufficient to remark that the disjoint union of all D_i , $i \leq p$, is a subgraph of D that can be obtained by fixing some particular maximal matchings, i.e., $\cup_{i \leq p} D_i \in \mathcal{D}$. The result then easily follows Definition 1 and the fact that the process number of a graph is the maximum of the process number of its connected components [2]. \square

The definition of the generalized process number implies that, for any multi digraph D labeled in a nice way, $\text{gpn}(D) = 0$ if there is a maximal matching M_e for any label e , such that the subgraph induced by $\cup_e M_e$ is a DAG.

Theorem 3. *Let D be a multi digraph labeled in a nice way. Then, the problem to decide whether $\text{gpn}(D) = 0$ is NP-complete.*

Proof. The problem clearly belongs to NP because $\text{gpn}(D) = 0$ if and only if there is a maximal matching M_e for any label e , such that the subgraph induced by $\cup_e M_e$ is a DAG.

The proof reduces 3-SAT to this problem. Let F be a 3-SAT-Formula with variables x^1, \dots, x^n and clauses C^1, \dots, C^m . From F we construct a simple $3n + 6m$ -nodes digraph

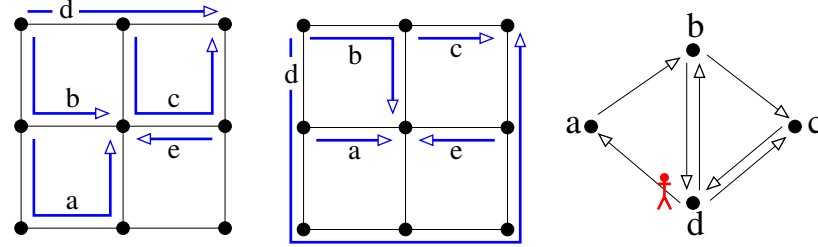
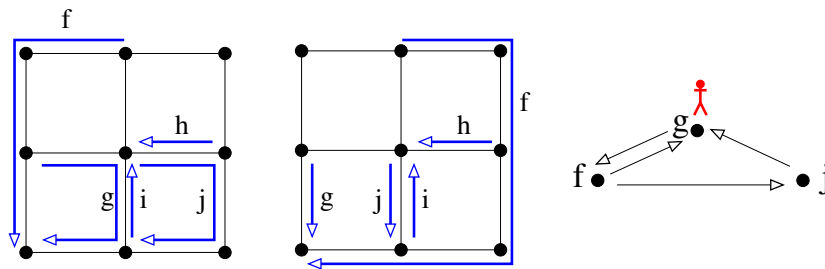
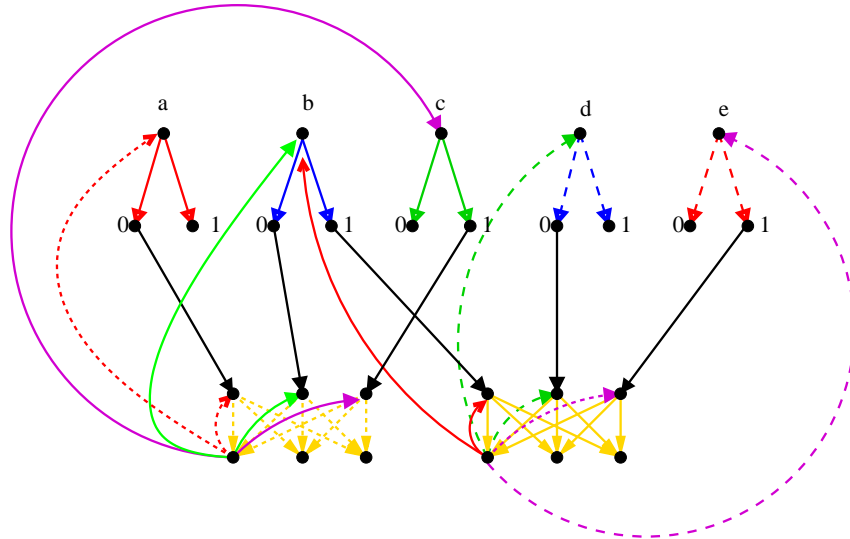
(a) Initial and final routings \mathcal{R}_1 and \mathcal{Q}_1 and their dependency digraph(b) Initial and final routings \mathcal{R}_2 and \mathcal{Q}_2 and their dependency digraph

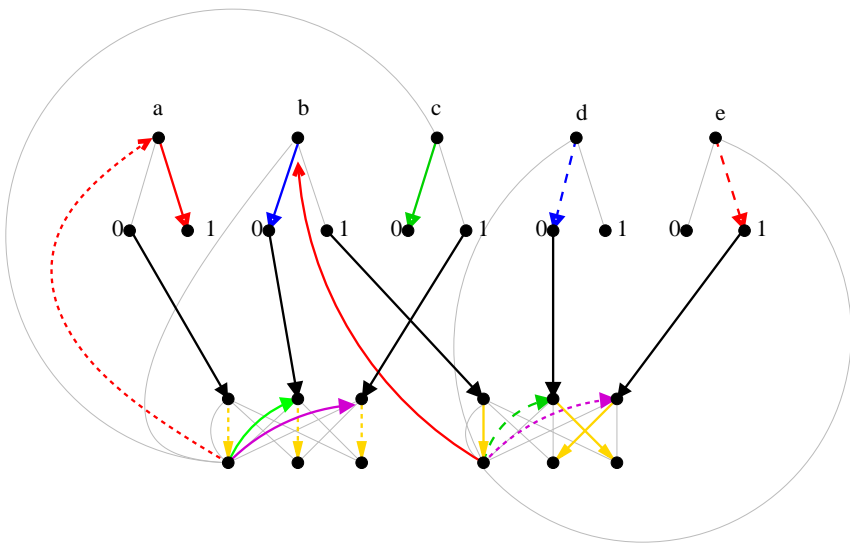
Figure 3: Figs. 3(a) and 3(b) represent two instances of the routing reconfiguration problem in a grid in which all arcs have capacity one. Both associated dependency digraph have process number ≥ 1 (because cycles exist). However, if we consider the reconfiguration from $\mathcal{R}_1 \cup \mathcal{R}_2$ to $\mathcal{Q}_1 \cup \mathcal{Q}_2$ in the same grid where all arcs have capacity 2, it can be done without interrupting any request by switching the requests in the following order: j, b, a, f, d, g .

D with labeled edges such that, for any label, the subgraph induced by the edges with this label is either a star with maximum out-degree three or $K_{3,3}$. We construct D as follows (see Fig. A.4(a) for an example):

1. for each variable x^i in F add the three vertices v^i, v_0^i and v_1^i ;
2. for any $i \leq n$, choose an unused label and add two arcs with this label from v^i to v_0^i and to v_1^i . Intuitively, any maximal matching in the subgraph induced by these two arcs represents an assignment of the variable x^i ;
3. for each clause C^j in F add the six vertices $a_1^j, a_2^j, a_3^j, c^j, c_0^j$ and c_1^j . Intuitively, a_1^j, a_2^j and a_3^j will be associated to the variables occurring in C^j , and c^j will represent the value of C^j ;



(a) Reduction of the formula $(a \vee b \vee \neg c) \wedge (\neg b \vee d \vee \neg e)$.



(b) Possible set of matchings inducing a DAG. The arcs in grey have been “removed”.

Figure 4: Reduction of a 3-SAT-Formula to our problem (Fig. A.4(a)) and set of matchings forming a DAG and so the corresponding satisfying assignment (Fig. A.4(b)).

4. for any $j \leq m$, choose an unused label and add all arcs with this label from a_1^j, a_2^j and a_3^j to c^j, c_0^j and c_1^j . Note that c_0^j and c_1^j are used only to ensure that the subgraph obtained with this label is a nice bipartite digraph;
5. for any $j \leq m$, let us consider a variable x^i that appears in C^j . Choose an unused label and add an arc either from v_0^i to a_1^j if there is an occurrence of x^i in C^j , or from v_1^i to a_1^j if there is an occurrence of \bar{x}^i in C^j . We say that we *associate* x^i with a_1^j . In a similar way, using distinct labels, let us associate the second variable occurring in C^j with a_2^j , and the third variable with a_3^j ;
6. for any $j \leq m$ and for any variable x^i that appears in C^j and associated to a_ℓ^j , $\ell \in \{1, 2, 3\}$, choose an unused label and add an arc from c^j to v^i and from c^j to a_ℓ^j .

Remark that for any arc e defined in step (v), e is the single arc with its label and so any maximal matching contains e .

Let F be a formula that cannot be satisfied. Let D be the digraph obtained from F by the construction above. For any label e of the arcs of D , let us choose an arbitrary maximal matching M_e of the subgraph induced by the edges with label e . We prove that the subgraph H of D induced by $\bigcup_e M_e$ contains a cycle. We consider the assignment β such that for any variable x^i , we set $\beta(x^i) = \text{true}$ if H contains the arc (v^i, v_1^i) and $\beta(x^i) = \text{false}$ otherwise (H contains the arc (v^i, v_0^i)). Since F cannot be satisfied, there is $j \leq m$, such that $\beta(C^j) = \text{false}$. In H , the vertex c^j is the head of one arc and w.l.o.g., we assume that $(a_1^j, c^j) \in E(H)$. Let x^i be the variable occurring in C^j that is associated to a_1^j . We prove the result by assuming that C^j contains the occurrence of \bar{x}^i , the other case is similar. By definition of D , $(v_1^i, a_1^j) \in E(D)$ and it is the only arc with its label. Thus $(v_1^i, a_1^j) \in E(H)$. Since $\beta(C^j) = \text{false}$, thus $\beta(x^i) = \text{true}$ and $(v^i, v_1^i) \in E(H)$. To conclude, it is sufficient to remark that either (c^j, v^j) or (c^j, a_1^j) is an arc in H . Hence, H contains a cycle: (c^j, a_1^j, c^j) or $(c^j, v^i, v_1^i, a_1^j, c^j)$.

Let β be a satisfying assignment for F . We show that, for each label, we can choose a maximal matching with this label such that the subgraph induced by these edges is acyclic. For each $i \leq n$, choose the arc (v^i, v_1^i) if $\beta(x^i) = \text{true}$, and (v^i, v_0^i) otherwise. For any $j \leq m$, let us consider the clause C^j . There exists $i \leq n$ such that, either $\beta(x^i) = \text{true}$ and there is an occurrence of x^i in C^j , or $\beta(x^i) = \text{false}$ and there is an occurrence of \bar{x}^i in C^j (such an integer i exists because β is a satisfying assignment for F). The variable x^i plays a particular role for the clause C^j because it is a variable whose value implies $\beta(C^j) = \text{true}$. W.l.o.g., let us assume that x^i is associated to a_1^j in D . We choose (a_1^j, c^j) , (a_2^j, c_0^j) and (a_3^j, c_1^j) and we choose (c^j, v^i) , (c^j, a_2^j) and (c^j, a_3^j) . See Fig. A.4(b) for an example.

We now prove that the resulting subgraph H of D is acyclic.

We first prove that no cycles pass through v^i for any variable x^i . Note that v^i has out-degree one in H , and H contains either (v^i, v_1^i) or (v^i, v_0^i) , w.l.o.g., $(v^i, v_0^i) \in E(H)$. By construction of H , this means that $\beta(x^i) = \text{false}$. If v_0^i is a leaf (i.e., has out-degree 0), no cycles pass through v^i . Otherwise, let us consider any out-neighbour a_ℓ^j of v_0^i , for some

$j \leq m, \ell \in \{1, 2, 3\}$. We prove that the single out-neighbour of a_ℓ^j has out-degree 1 and it is not v^i , thus no cycles pass through v^i . Indeed, (v_0^i, a_ℓ^j) belongs to $E(D)$ only if there is an occurrence of x^i in C^j (step (v) of construction of D). Since $\beta(x^i) = false$, the value of x^i does not imply that $\beta(C^j) = true$ and, by construction of H , $(a_\ell^j, c^j) \notin E(H)$. Therefore, the unique out-neighbour of a_ℓ^j is either c_0^j or c_1^j that have out-degree 0. Furthermore for any $i \leq n$ the single in-neighbour of v_0^i (and of v_1^i) in H , if any, is v^i . Thus no cycles pass through v_p^i .

By construction of D , for any $j < k \leq m$, there are no arcs between a vertex in $\{a_1^j, a_2^j, a_3^j, c^j, c_0^j, c_1^j\}$ and a vertex in $\{a_1^k, a_2^k, a_3^k, c^k, c_0^k, c_1^k\}$. Hence, if there is a cycle in H , there must be a $j \leq m$, such that the vertices of this cycle belong to $\{a_1^j, a_2^j, a_3^j, c^j, c_0^j, c_1^j\}$. Recall that, in H , there are three independent arcs from a_1^j, a_2^j, a_3^j to c^j, c_0^j, c_1^j , two arcs from c^j to 2 vertices in $\{a_1^j, a_2^j, a_3^j\}$ and no other arcs from and to vertices belonging to the previous set. W.l.o.g., let us assume that the single in-neighbour of c^j in H is a_1^j . By construction of H , the 2 previous arcs from c^j go to a_2^j and a_3^j . Thus it cannot be a cycle with vertices in $\{a_1^j, a_2^j, a_3^j, c^j, c_0^j, c_1^j\}$. This concludes the proof that H is acyclic, and the proof of Theorem 3. \square

Theorem 3 implies that the problem of deciding whether there exists a scheduling of the rerouting of connection requests without traffic interruption is NP-complete. Moreover, this problem is NP-complete even if requests use the third of the bandwidth of a link. Indeed, the multi-digraph with labeled arcs that we build when reducing 3-SAT to this problem has the following property: for any label, the set of edges with this label induces a subgraph that is either a star with maximum out-degree 3 or a $K_{3,3}$. That is, we reduce 3-SAT to some instances of the routing reconfiguration problem in a network the links of which have capacity at most three.

4 Open problems

Many questions remain open in this study. In particular, is it always possible to reduce the problem to the case with capacity one, i.e., when $\text{pn}(D) = 0$, can we decompose the problem into sub-problem with capacity 1 that can be solved without interruption? What is the complexity of deciding whether it exists a scheduling of the rerouting of connection requests without traffic interruption when links' capacity is at most 2. The characterization of dependency multi-digraphs for which the number of requests that must be simultaneously interrupted can be determined in polynomial time is also an important issue. Finally, efficient heuristic algorithms to address practical instances are needed.

Acknowledgments

This work has been partially supported by région PACA, and European project IST FET AEOLUS.

References

- [1] D. Coudert, F. Huc, D. Mazaauric, N. Nisse, and J-S. Sereni. Routing reconfiguration/process number: Coping with two classes of services. In *13th Conference on Optical Network Design and Modeling (ONDM)*, Braunschweig, Germany, February 2009. IEEE. to appear.
- [2] D. Coudert and J-S. Sereni. Characterization of graphs and digraphs with small process number. Research Report 6285, INRIA, September 2007.
- [3] F. Fomin and D. Thilikos. An annotated bibliography on guaranteed graph searching. *Theor. Comput. Sci.*, 399(3):236–245, 2008.
- [4] N. Jose and A.K. Somani. Connection rerouting/network reconfiguration. *Design of Reliable Communication Networks*, pages 23–30, October 2003.
- [5] M. Kirousis and C.H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47(2):205–218, 1986.



Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
<http://www.inria.fr>
ISSN 0249-6399