INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Integral Symmetric 2-Commodity Flows

Aubin Jarry

N° 4622

Novembre 2002

THÈME 1



ISSN 0249-6399 ISRN INRIA/RR--4622--FR+ENG

INRIA
SOPHIA ANTIPOLIS

Integral Symmetric 2-Commodity Flows

Aubin Jarry*

Thème 1 — Réseaux et systèmes
Projet Mascotte

Rapport de recherche n° 4622 — Novembre 2002 — 20 pages

Abstract: Let $G = (V, A)$ be a symmetric digraph, and $\kappa : A \to \mathbb{N}$ be a symmetric capacity. Let $s_1, s_2, t_1, t_2 \in V$ and $v_1, v_2 \in \mathbb{N}$. An integral symmetric 2-commodity flow in G from (s_1, s_2) to (t_1, t_2) of value (v_1, v_2) is an integral 4-commodity flow from (s_1, t_1, s_2, t_2) to (t_1, s_1, t_2, s_2) of value (v_1, v_1, v_2, v_2). The Integral Symmetric 2-Commodity Flow Problem consists in finding a symmetric 2-commodity flow $(f_1, f_{-1}, f_2, f_{-2})$ from (s_1, t_1) to (s_2, t_2) of value (v_1, v_2) such that $\Sigma f_i \leq \kappa$. It is known that the Integral 2-Commodity Flow Problem is NP-complete for both directed and undirected graphs ([FHW80] and [EIS76]). We prove that the cut criterion is a necessary and sufficient condition for the existence of a solution to the Integral Symmetric 2-Commodity Flow Problem, and give a polynomial-time algorithm in that provides a solution to this problem. The time complexity of our algorithm is $6C_{flow} + O(|A|)$, where C_{flow} is the time complexity of your favorite flow algorithm (usually in $O(|V| \times |A|)$).

Key-words: disjoint paths, multiflow, commodity, flow, integer, symmetric, graph

* Aubin.Jarry@sophia.inria.fr

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Multiflots entiers symétriques à 2 commodités

Résumé : Soit $G = (V, A)$ un graphe orienté symétrique, et $\kappa : A \to \mathbb{N}$ une capacité symétrique. Soit $s_1, s_2, t_1, t_2 \in V$ et $v_1, v_2 \in \mathbb{N}$. Un multiflot entier symétrique à 2 commodités dans G de (s_1, s_2) vers (t_1, t_2) et de valeur (v_1, v_2) est un multiflot entier à 4 commodités de (s_1, t_1, s_2, t_2) vers (t_1, s_1, t_2, s_2) et de valeur (v_1, v_1, v_2, v_2). Le problème du multiflot entier symétrique à 2 commodités consiste à trouver un multiflot entier symétrique à 2 commodités $(f_1, f_{-1}, f_2, f_{-2})$ de (s_1, t_1) vers (s_2, t_2) et de valeur (v_1, v_2) tel que $\Sigma f_i \leq \kappa$. On sait que le problème du multiflot entier à 2 commodités est NP-complet dans les graphes orientés ou non ([FHW80] and [EIS76]). Nous prouvons que le critère de coupe est une condition nécessaire et suffisante pour ce problème, et nous donnons un algorithme en temps polynômial pour ce problème. La complexité de notre algorithme est $6C_{flow} + O(|A|)$, où C_{flow} est la complexité de l'algorithme de flot choisi (usuellement en $O(|V| \times |A|)$).

Mots-clés : chemins disjoints, multiflot, commodité, flot, entier, symétrique, graphe

1 Introduction

Given a graph $G = (V, A)$, a capacity $\kappa : A \to \mathbb{N}$ and a request set $R \in (V \times V \times \mathbb{N})$, the Multi-Commodity Flow Problem consists in finding $|R|$ flows corresponding to the request set and with respect to the capacity constraints on the graph. The Multi-Commodity Flow Problem has been widely studied, as it arises naturally from many classical problems such as routing problems. The Fractional Problem (allowing fractional flows) can be solved in polynomial time by using linear programming. However, the Integral Problem (not allowing fractional flows) is also of interest when we have non-splittable units of traffic, or non-splittable routes to find (e.g. for synchronous communications). The integral multi-commodity flow problem is NP-complete in the general case ([FHW80] and [Kar75]), and many variants have been studied. They divide themselves between NP-complete and tractable problems. One variant is the Disjoint Paths Problem which consists in finding $|R|$ disjoint paths corresponding to the request set (with $R \subset (V \times V)$). The other main variants depend on the structure of G or R or or both : whether G is directed or undirected, whether G or $G + R$ is planar, whether G or $G + R$ is Eulerian, etc... One can find a good survey in [Vyg94].

Our interest in this problem comes from routing problems in optical networks. Optical networks are best represented by symmetric digraphs. So the interconnection graph $G = (V, A)$ is symmetric directed (i.e. $(x, y) \in A \Rightarrow (y, x) \in A$), and the capacity function is symmetric (i.e. $\forall (x, y) \in A$, $\kappa(x, y) = \kappa(y, x)$). This topology applies also to many other telecommunication networks.

Since G is also Eulerian, the symmetric problem would appear to be a particular case of the directed Eulerian variant, except usually $G+R$ is assumed to be Eulerian (and not only G). On this variant, C.Nash-Williams proved in 1965 (one can find a proof in [Vyg94]) that with $|R| = 2$ the problem was polynomial, whereas J. Vygen proved in [Vyg95] that with $|R| = 3$ (and more) the problem was NP-complete. In the general directed case, the Disjoint Paths Problem is NP-complete with $|R| = 2$ ([FHW80]).

In this paper, we will more specifically study the problem with symmetric requests (so $G + R$ is indeed Eulerian, but $|R| = 4$), so one can observe that a symmetric digraph with symmetric requests has the same knowledge structure as an undirected graph with undirected requests, and that an undirected solution would perfectly fit for the symmetric problem. While this is true, the symmetric problem allows also more solutions and is more tractable : [EIS76] proved that the Integral Undirected 2-Commodity (i.e. $|R| = 2$) Flow Problem was NP-complete even with a value of 1 for the first commodity, whereas we will prove that the Integral Symmetric (i.e.

with a symmetric capacity and symmetric requests) 2-Commodity Flow Problem is polynomial.

In section 2 we introduce our notations, consider related works and introduce our problem. In section 3 we giving our algorithm and prove it, thus proving our main theorem (Theorem 5, stated at the end of section 2).

2 Standard notations and Related Works

2.1 Standard Notations

Let $G = (V, A)$ be a symmetric digraph, that is, for all $x, y \in V$, $(x, y) \in A \Rightarrow (y, x)$ in A. For all $x \in V$, we call $\Gamma(x)$ the set of all the vertices $y \in V$ such that $(x, y) \in A$.

Definition 1 (flow)
Let $s, t \in V$ and $k \in \mathbb{N}$. A flow f from s to t of value v is a function $f : A \to \mathbb{N}$ such that

- $\forall x \notin \{s, t\}, \Sigma_{y \in \Gamma(x)}(f(x, y) - f(y, x)) = 0$
- $\Sigma_{y \in \Gamma(s)}(f(s, y) - f(y, s)) = v$
- $\Sigma_{x \in \Gamma(t)}(f(x, t) - f(t, x)) = v$

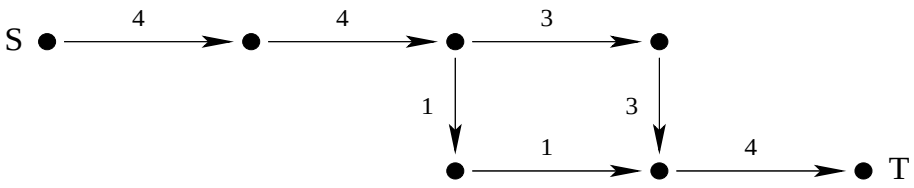


Figure 1: a flow from s to t of value 4

This definition of a flow allows the existence of loops. Thus we will speak of "flow without loops" when needed. Unless specified otherwise, the flow we consider are integral flows (they are not fractional). When we have several functions f_1, f_2 and $\kappa : A \to \mathbb{N}$ (for instance, f_1 and f_2 may be flows and κ a capacity) we will use the global notations :

- $f_1 \leq \kappa$ means that for all $a \in A$, $f_1(a) \leq \kappa(a)$.
- $f_1 + f_2 : A \to \mathbb{N}$ is the function defined by $\forall a \in A$, $(f_1 + f_2)(a) = f_1(a) + f_2(a)$.
- if $f_1 \leq \kappa$, $(\kappa - f_1) : A \to \mathbb{N}$ is the function defined by $\forall a \in A$, $(\kappa - f_1)(a) = \kappa(a) - f_1(a)$. Note that $f_2 \leq (\kappa - f_1)$ is equivalent to $(f_1 + f_2) \leq \kappa$.
- $|f_1| : A \to \mathbb{N}$ is the function defined by $\forall (x, y) \in A$, $|f_1|(x, y) = f_1(x, y) - \min(f_1(x, y), f_1(y, x))$.

Definition 2 (symmetric)
Let $f : A \to \mathbb{N}$. We say that f is symmetric if for all $(x, y) \in A$, $f(x, y) = f(y, x)$.

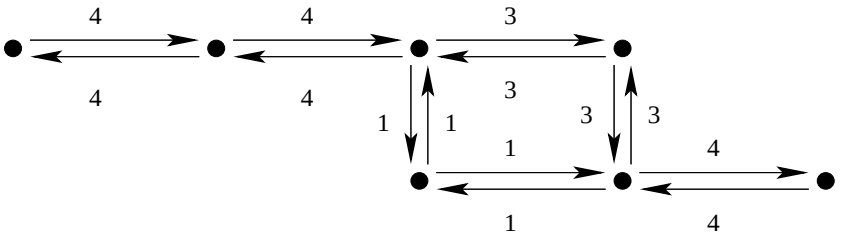


Figure 2: a symmetric function on A

We will study problems with a symmetric capacity κ.

Definition 3 (f^r, reverse)
Let $f : A \to \mathbb{N}$. The reverse function of f, $f^r : A \to \mathbb{N}$ is the function defined by $\forall (x, y) \in A$, $f^r(x, y) = f(y, x)$

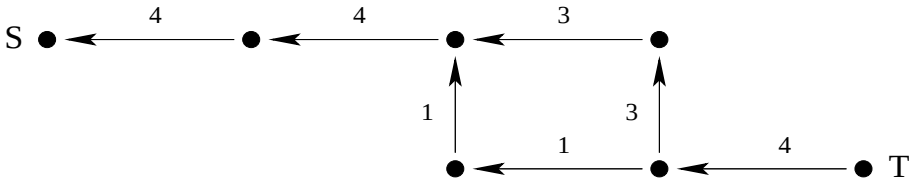


Figure 3: a reverse flow from t to s of value 4

If f is a flow from s to t of value v, then f^r is the reverse flow from t to s of value v.

Definition 4 (k-commodity flow)
For all $i \in \{1, ..k\}$, let f_i be a flow from s_i to t_i of value v_i. $(f_1, ..f_k)$ is a k-commodity flow from $(s_1, ..s_k)$ to $(t_1, ..t_k)$ of value $(v_1, ..v_k)$.

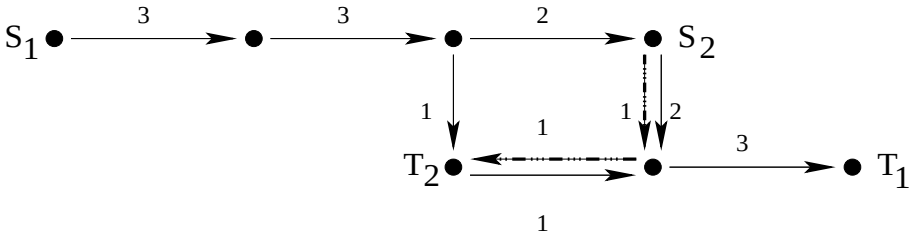


Figure 4: 2-commodity flow from (s_1, s_2) to (t_1, t_2) of value $(3, 1)$

Usually, the capacity constraint is $(\Sigma_i f_i) \leq \kappa$ for some $\kappa : A \to \mathbb{N}$.

2.2 Menger's theorem

Definition 5 (cut)
Let C be a subset of V with $C \neq \emptyset$ and $C \neq V$. C is also called a cut on G. Let $A(C) = (C \times (V \backslash C)) \bigcap A$. For every function $f : A \to \mathbb{N}$, we call $f(C)$ the sum $\Sigma_{(a \in A(C))}\ f(x, y)$.

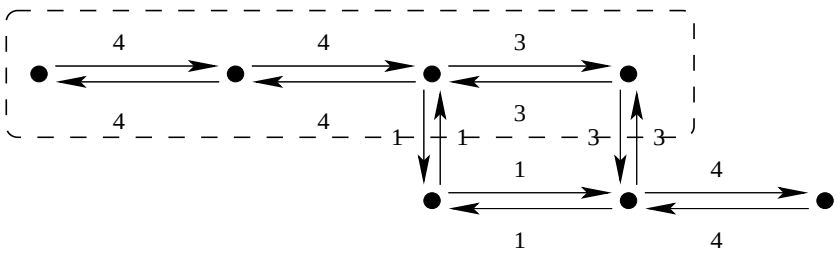


Figure 5: a cut of value 4

Definition 6 (cut criterion)
Let $\kappa : A \to \mathbb{N}$ be a capacity on G. Let $s_1, ..s_k, t_1, ..t_k \in V$. Let $v_1, ..v_k \in \mathbb{N}$. The cut criterion for $(\kappa, (s_1, t_1, v_1), .. (s_k, t_k, v_k))$ is : for all cut $C \subset V$, for all $I \subset \{1, ..k\}$ we have : $(\forall i \in I$, $s_i \in C$ and $t_i \notin C) \Rightarrow \kappa(C) \geq (\Sigma_{i \in I} v_i)$.

Usually we say that the cut criterion is true or false.

Theorem 1 (Menger) *The cut criterion is a necessary and sufficient condition for the existence of a solution to the Integral Flow Problem.*

Corollary 1 *The cut criterion is a necessary condition for the existence of a solution to the Integral k-Commodity Flow Problem.*

Proof
We suppose the existence of a k-commodity flow, and then prove that the cut criterion is true by adding all the commodities.
The input is :

- a capacity $\kappa : A \to \mathbb{N}$;
- the source and target of the k-commodity flow $(s_1, ..s_k)$, $(t_1, ..t_k) \in V^k$;
- the value of the k-commodity flow $(v_1, ..v_k) \in \mathbb{N}^k$;
- the k-commodity flow itself $(f_1, ..f_k)$ such that $\Sigma_i f_i \leq \kappa$;
- a cut $C \subset V$ and a set $I \subset \mathbb{N}$ such that $\forall i \in I$, $s_i \in C$ and $t_i \notin C$.

The cut criterion is whatever C and I, we must have $\kappa(C) \geq \Sigma_{i \in I} v_i$. In order to prove it, we will add all the flows $(f_i)_{i \in I}$. We call

- $S = \{s_i, i \in I\}$, $T = \{t_i, i \in I\}$ and $v = \Sigma_{i \in I} v_i$;
- $V' = \{S, T\} \bigcup \{\{x\}, x \in V \backslash (S \bigcup T)\}$, $A' = (V' \times V') \bigcap \mathcal{P}(\mathrm{A})$ and $G' = (V', A')$;
- $K : A' \to \mathbb{N}$ such that for all $a' \in A'$, $K(a') = \Sigma_{a \in a'} \kappa(a)$
- $F : A' \to \mathbb{N}$ such that for all $a' \in A'$, $F(a') = \Sigma_{a \in a'} \Sigma_{i \in I} f_i(a)$
- $C' = \{S\} \bigcup \{\{x\}, x \in C \backslash S\}$

Observe that $F \leq K$ is a flow from S to T of value v. Observe that $K(C') = \kappa(C)$. According to Theorem 1 the cut criterion is respected in G' for $(K, (S, T, v))$. Therefore, $\kappa(C) \geq v$. □

2.3 Complexity of some Integral Multi-Commodity Flow Problems

As we said in the introduction, our problem is a particular case of the general Integral Multi-Commodity Flow Problem. Since the result of Fortune, Hopcroft and Willie [FHW80] we know that the Disjoint Paths Problem with is NP-complete with only two requests ($|R| = 2$).

More specifically, we know that in Eulerian digraphs ($G + R$ is Eulerian), the problem is polynomial with 2 commodities ($|R| = 2$), but NP-complete with three ($|R| = 3$). In our problem, $|R| = 4$.

In undirected graphs, the Disjoint Paths Problem is polynomial with a bounded number of requests ($|R| \leq k$) [RS95], but the 2-Commodity Flow Problem (with $|R| = 2$) is NP-complete even if one of the requested flows should be of value 1 (with $R = \{(x, y, 1), (x', y', v)\}$) [EIS76].

In symmetric digraphs, we already know that the Disjoint Path Problem is NP-complete [Cha98] in general, but that it is polynomial with a bounded number of requests [Jar02]. For the Multi-Commodity Flow problem, one can see that the 2-Commodity Flow Problem is polynomial with a value of $(1, v)$ (with $R = \{(x, y, 1), (x', y', v)\}$) :

Theorem 2 *The 2-Commodity Flow Problem is polynomial in symmetric digraphs if the value is* $(1, v)$.

Proof
We assume that the cut criterion is true (it can be checked in polynomial time), then find the flow of value v, and last find the flow of value 1.
The input is :

- a symmetric capacity $\kappa : A \to \mathbb{N}$;
- the source and target of the requested 2-commodity flow (s_1, s_2), $(t_1, t_2) \in V^2$;
- the value of the requested 2-commodity flow $(1, v) \in \mathbb{N}^2$.

We first find a flow without loops $f_2 \leq \kappa$ from s_2 to t_2 of value v. If the cut criterion is true, then f_2 can be found using any polynomial time flow algorithm. Now we prove that the cut criterion is true for $((\kappa - f_2), (s_1, t_1, 1)$. Let $C \subset V$ be a cut such that $s_1 \in C$ and $t_1 \notin C$. Two cases are possible :

- if $s_2 \in C$ and $t_2 \notin C$, then $\kappa(C) \geq (1 + v)$. If $f_2(C) \geq v + 1$, then there is $x \in C$ and $y \notin C$ with $(x, y) \in A$ such that $f_2(y, x) \geq 1$ so $f_2(x, y) = 0$: $(\kappa - f_2)(C) \geq 1$

- otherwise, $\kappa(C) \geq 1$. If $f_2(C) \geq 1$, then there is $x \in C$ and $y \notin C$ with $(x, y) \in A$ such that $f_2(y, x) \geq 1$ so $f_2(x, y) = 0 : (\kappa - f_2)(C) \geq 1$

The cut criterion is true, so according to Theorem 1, we can find a flow $f_1 \leq (\kappa - f_2)$ from s_1 to t_1 of value 1. □

2.4 The 2-Commodity Flow Problem in a Symmetric Digraph

In this section, we consider a symmetric capacity $\kappa : A \to \mathbb{N}$ on G, two pair of vertices (s_1, t_1) and (s_2, t_2), and two positive integers v_1 and v_2.

We will see that the cut criterion is not a sufficient condition for the 2-Commodity Flow Problem, though it is necessary according to Theorem 1.

Theorem 3 *The cut criterion is not a sufficient condition for the existence of a solution to the Integral 2-Commodity Flow Problem.*

Proof
We give a counter-example (see Figure 6) :

- the vertex set is $V = \{s_1, s_2, t_1, t_2\}$;
- the arc set is $A = \{(s_1, s_2), (s_2, s_1), (s_2, t_1), (t_1, s_2), (s_1, t_2), (t_2, s_1), (t_2, t_1), (t_1, t_2)\}$;
- we consider the symmetric digraph $G = (V, A)$;
- the capacity on G is $\kappa : A \to \mathbb{N}$ such that

$$\begin{array}{ll} \kappa(s_1, s_2) = \kappa(s_2, s_1) = 1 & \kappa(s_2, t_1) = \kappa(t_1, s_2) = 1 \\ \kappa(s_1, t_2) = \kappa(t_2, s_1) = 3 & \kappa(t_2, t_1) = \kappa(t_1, t_2) = 1. \end{array}$$

The cut criterion is true for $(\kappa, (s_1, t_1, 2), (s_2, t_2, 2))$, but there is no 2-commodity flow from (s_1, s_2) to (t_1, t_2) of value $(2, 2)$. □

However even if the cut criterion is not necessary, another criterion stands out when we study the example given before : in this previous example, the cut $\{s_1, t_2\}$ was of value 2 though s_1 and t_2 are in the same side of this cut. So we introduce what we call the symmetric cut criterion :

Definition 7 (symmetric cut criterion)
Let $\kappa : A \to \mathbb{N}$ be a symmetric capacity. Let $s_1, t_1, s_2, t_2 \in V$. Let v_1 and v_2 be two positive integers. The symmetric cut criterion for $(\kappa, (s_1, t_1, v_1), (s_2, t_2, v_2))$ is the cut criterion for $(\kappa, (s_1, t_1, v_1), (t_1, s_1, v_1), (s_2, t_2, v_2), (t_2, s_2, v_2))$.

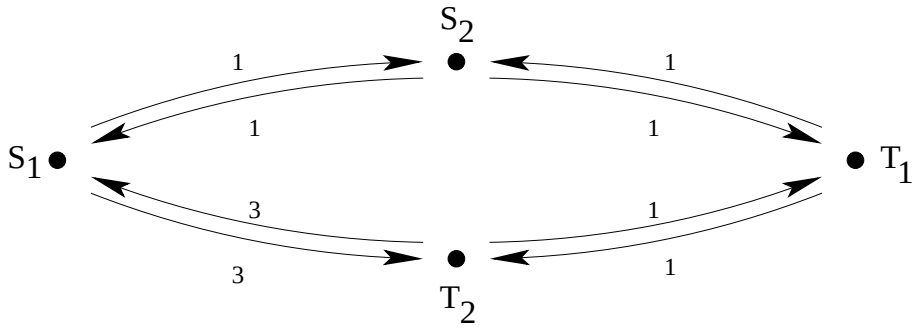


Figure 6: there is no 2-commodity flow from (s_1, s_2) to (t_1, t_2) of value $(2, 2)$

We will prove later on that this criterion is sufficient for the 2-Commodity Flow Problem, though it is not necessary :

Theorem 4 *The symmetric cut criterion is not necessary condition to have a solution to the 2-Commodity Flow Problem.*

Proof
We give a counter-example (see Figure 7) :

- the vertex set is $V = \{x, y\}$; the arc set is $A = \{(x, y), (y, x)\}$;
- we consider the symmetric digraph $G = (V, A)$;
- the capacity on G is $\kappa : A \to \mathbb{N}$ with $\kappa(x, y) = \kappa(y, x) = 1$;
- we define a flow $f \leq \kappa$ by $f(x, y) = 1$ and $f(y, x) = 0$.

The symmetric cut criterion is false for $(\kappa, (x, y, 1), (y, x, 1))$, but (f, f^r) is a 2-commodity flow from (x, y) to (y, x) of value $(1, 1)$. □

2.5 Symmetric 2-Commodity Flow

Since the symmetric cut criterion seems a bit strict for the 2-Commodity Flow Problem, we introduce the problem for which the symmetric cut criterion is a tight condition.

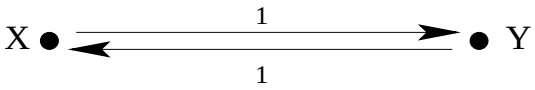


Figure 7: there is a 2-commodity flow from (x, y) to (y, x) of value $(1, 1)$

Definition 8 (symmetric 2-commodity flow)
Let $(f_1, f_{-1}, f_2, f_{-2})$ be a 4-commodity flow from (s_1, t_1, s_2, t_2) to (t_1, s_1, t_2, s_2) of value $(v_1, v_1,\ v_2, v_2)$. $(f_1, f_{-1}, f_2, f_{-2})$ is also called symmetric 2-commodity flow from (s_1, s_2) to (t_1, t_2) of value (v_1, v_2).

According to Theorem 1, the symmetric cut criterion is a necessary condition for the Symmetric 2-Commodity Flow Problem. In section 3 we will prove that it is also a sufficient condition :

Theorem 5 (symmetric 2-commodity flow) *The symmetric cut criterion is a necessary and sufficient condition for the existence of a solution to the Integral Symmetric 2-Commodity Flow Problem. A solution can be found in $6C_{flow} + O(|A|)$ steps.*

This theorem implies immediately that the symmetric cut criterion is a sufficient condition for the 2-Commodity Flow Problem (with a symmetric capacity).

Corollary 2 *The symmetric cut criterion is a sufficient condition for the existence of a solution to the 2-Commodity Flow Problem.*

3 Solution to the Symmetric 2-Commodity Flow Problem

In this section, we consider a symmetric capacity $\kappa : A \to \mathbb{N}$ on G, two pair of vertices (s_1, t_1) and (s_2, t_2), and two positive integers v_1 and v_2. The goal of this section is to prove Theorem 5 and to explain our algorithm (Algorithm 5, described at subsection 3.5). To proceed with our work, we need a preliminary lemma (lemma 1) and a subroutine (Algorithm 1) which is the purpose of the next subsection (subsection 3.1). Subsection 3.1 deals only with an integral flow, regardless of symmetry or any other property. Afterwards, (subsection 3.2) we give the first part of our algorithm (Algorithm 2 (first step)) which gives a flow for the first commodity which

does not alter the cut criterion of the second commodity. However, this intermediate flow is not part of the final solution. Then, in subsection 3.3 we compute the flows for the second commodity (Algorithm 3 (second step)). Eventually (subsection 3.4) we compute the flows for the first commodity (Algorithm 4 (third step)). Subsection 3.5 is a round up of the different parts of our algorithm.

3.1 A preliminary lemma on half flows

The purpose of this subsection is to prove that given a flow f of even value, we can split f in two flows g and g' (this means that $g+g'=f$) so that $2g \leq (f+1)$ and $2g' \leq (f+1)$. In other words, we compute a flow g such that $(f-1) \leq 2g \leq (f+1)$ (by 1, we mean the function such that $\forall a \in A$, $1(a)=1$).

Algorithm 1 (finding g such that $(f-1) \leq 2g \leq (f+1)$)
Input : $G=(V,A)$; $f : A \to \mathbb{N}$.
Variables : $f' : A \to \mathbb{N}$; $x \in V$.

- $\forall a \in A$, $f'(a) \leftarrow f(a)$
- *while there is* $(x_{start}, y_{start}) \in A$ *such that* $f'(x_{start}, y_{start})$ *is odd do*
  - $f'(x_{start}, y_{start}) \leftarrow f'(x_{start}, y_{start}) + 1$
  - $x \leftarrow y_{start}$
  - *while* $x \neq x_{start}$ *do*
    - *if there is* $y \in \Gamma(x)$ *such that* $f'(x,y)$ *is odd then*
      - $f'(x,y) \leftarrow f'(x,y) + 1$
      - $x \leftarrow y$
    - *else choose* $y \in \Gamma(x)$ *such that* $f'(y,x)$ *is odd*
      - $f'(y,x) \leftarrow f'(x,y) - 1$
      - $x \leftarrow y$
- $\forall a \in A$, ***define*** $g(a) = \frac{f'(a)}{2}$

Output : $g : A \to \mathbb{N}$.

Lemma 1 (half flow) *Let f be a flow from s to t of even value $2v$. There is a flow g from s to t of value v such that $(f-1) \leq 2g \leq (f+1)$. This flow can be computed by Algorithm 1 in $(O(|A|)$ steps.*

We divide the proof into two parts : first the algorithms completes and then the output is correct.

Proposition 1 *If f is a flow of even value, then Algorithm 1 completes in $O(|A|)$ steps.*

Proof
If f is a of flow of even value then for each vertex $x \in V$ there is an even number of vertices $y \in \Gamma(x)$ such that $f'(x,y)$ or $f'(y,x)$ is odd. This is true in particular for s and t because the value of f is even. Therefore, when the algorithm reaches a vertex x after changing an edge adjacent to x, it will find another edge adjacent to x to change, unless $x = x_{start}$. Moreover, at each step the number of edges $a \in A$ such that $f'(a)$ is odd decreases, so the number of steps is bounded by the cardinality of $\{a \in A, f(a) \text{ is odd}\}$. Thus the complexity is in $O(|A|)$. □

Proposition 2 *If f is a flow from s to t of even value $2v$, the output g of Algorithm 1 is a flow from s to t of value v such that $(f-1) \leq 2g \leq (f+1)$.*

Proof
At the beginning of each loop (from x_{start} to x_{start}), the flow equations are broken in x_{start}. At each step inside the loop, the flow equations are kept for each x encountered. At the end of each loop, the flow equations are restored for x_{start}. Therefore, at the end of the algorithm f' is a flow from s to t of value $2v$. Moreover, every edge is processed at most once, so f and f' differ by at most 1 on each edge. Thus $(f-1) \leq 2g \leq (f+1)$. □

3.2 Finding a flow allowing the second cut criterion

In this subsection, we describe an algorithm that finds a flow f such that the cut criterion is true for $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$.

Algorithm 2 (first step)
Input : $G = (V, A)$; $\kappa : A \to \mathbb{N}$; $s_1, t_1, s_2, t_2 \in V$; $v_1, v_2 \in \mathbb{N}$.

- *compute a flow $h \leq \kappa$ from s_2 to t_2 of value v_2*

Let h^r be the reverse flow from t_2 to s_2 of value v_2 (see definition 3).

- *compute a flow $g \leq (\kappa + h^r - h)$ from s_1 to t_1 of value v_1*
- *compute a flow $g' \leq (\kappa + h - h^r)$ from s_1 to t_1 of value v_1*

- *using* ***Algorithm 1****, compute a flow* f *from* s_1 *to* t_1 *of value* v_1 *such that* $2f \leq (|g + g'| + 1)$

Output : $f : A \rightarrow \mathbb{N}$.

Lemma 2 (first step) *Let* $\kappa : A \rightarrow \mathbb{N}$ *be a symmetric capacity. Let* $s_1, t_1,\ s_2, t_2 \in V$ *and* $v_1,\ v_2 \in \mathbb{N}$ *such that the symmetric cut criterion is true for* $(\kappa, (s_1, t_1, v_1), (s_2, t_2, v_2))$. *Then there is a flow* $f \leq \kappa$ *from* s_1 *to* t_1 *of value* v_1 *such that the cut criterion is true for* $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$. *This flow can be computed by Algorithm 2 in* $3C_{flow} + O(|A|)$ *steps.*

We divide the proof into two parts : first the algorithm completes, and then the output is correct.

Proposition 3 *If the symmetric cut criterion is true for* $(\kappa, (s_1, t_1, v_1), (s_2, t_2, v_2))$, *then Algorithm 2 completes in* $3C_{flow} + O(|A|)$ *steps.*

Proof
If the flows we look for exist, then any integer flow algorithm can find them.

- if the symmetric cut criterion is true, then the flow h does exist.
- in Ford and Fulkerson's algorithm, the capacity $(\kappa + h^r - h)$ is called the residual capacity once h has been computed. Since the cut criterion is true between $\{s_1, s_2\}$ and $\{t_1, t_2\}$, one can increment the flow between $\{s_1, s_2\}$ and $\{t_1, t_2\}$ by v_1 and so find a flow g from s_1 to t_1 of value v_1 with the residual capacity. That is $g \leq (\kappa + h^r - h)$.
- in the same manner, $(\kappa + h - h^r)$ is the residual capacity once removed the flow h^r, so g' can be found.
- According to lemma 1, f is found by Algorithm 1.

This algorithm computes 3 flows, calls Algorithm 1 one time and computes two capacity functions, so this algorithm takes $3C_{flow} + O(|A|)$ steps. □

Proposition 4 *If the symmetric cut criterion is true for* $(\kappa, (s_1, t_1, v_1), (s_2, t_2, v_2))$, *then the flow* f *given by Algorithm 2 is such that the cut criterion for* $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$ *is true.*

Proof
First, observe that $(g + g') \leq 2\kappa$, so we have $f \leq \kappa$. Before considering $(\kappa - f)$, we will give two lower bounds to the function $(\kappa - |g + g'|) : A \rightarrow \mathbb{Z}$. Those bounds will enable us to control $(\kappa - f)$ over a cut.

We computed g' in order to have $g' \leq (\kappa + h - h^r)$; thus we have $(\kappa - g') \geq (h^r - h)$ and this leads to $(\kappa - (g + g')) \geq (h^r - h - g)$. Now, consider $(x, y) \in A$ such that $g(y, x) > 0$:

- if $|g + g'|(x, y) = g'(x, y) - g(y, x)$, then $(\kappa - |g + g'|)(x, y) = \kappa(x, y) - g'(x, y) + g(y, x)$ so $(\kappa - |g + g'|)(x, y) \geq (h^r - h + g^r)(x, y)$.
- otherwise, $|g + g'|(x, y) = 0$, so $(\kappa - |g + g'|)(x, y) = \kappa(x, y)$. Since $g \leq \kappa + h^r - h$, we have $\kappa(y, x) \geq g(y, x) + h(y, x) - h(x, y)$, so $(\kappa - |g + g'|)(x, y) \geq (h^r - h + g^r)(x, y)$.

Since $g(x, y)$ and $g(y, x)$ can not be non null at the same time, we have $(\kappa - |g + g'|) \geq (h^r - h - g + g^r)$. By a symmetric argument, we have as well $(\kappa - |g + g'|) \geq (h - h^r - g' + g'^r)$.

With these bounds, we can now consider a cut $C \subset V$ with, for instance, $s_2 \in C$ and $t_2 \notin C$. We will prove that $(\kappa - f)(C) \geq v_2$. We know that $(\kappa - |g + g'|)(C) \geq (h - h^r - g' + g'^r)(C)$, so $(\kappa - |g + g'|)(C) \geq v_2 + (g'^r - g')(C)$.

- if $t_1 \in C$ or $s_1 \notin C$, then $g'(C) \leq g'^r(C)$ and $(\kappa - |g + g'|)(C) \geq v_2$ so $(\kappa - f)(C) \geq v_2$).
- otherwise, $s_1 \in C$ and $t_1 \notin C$ so $g'(C) = g'^r(C) + v_1$ and $(\kappa - |g + g'|)(C) \geq v_2 - v_1$. In this case we have also $|g + g'|(C) = |g + g'|^r(C) + 2v_1$ and $f(C) = f^r(C) + v_1$, so $f(C) \leq |g + g'|^r(C) + v_1$ which implies $f(C) \leq |g + g'|(C) - v_1$. Thus $(\kappa - f)(C) \geq v_2$

Therefore, the cut criterion is true for $((\kappa - f), (s_2, t_2, v_2))$. By a symmetric argument, it is also true for $((\kappa - f), (t_2, s_2, v_2))$. □

3.3 Finding two of the four flows

Although the cut criterion is true for $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$, it may not be sufficient to guarantee the existence of a flow h from s_2 to t_2 of value v_2 and a flow h' from t_2 to s_2 of value v_2 such that $(f + h + h') \leq \kappa$, as shown in Figure 8. That is why f is not part of the final solution.

However, f is useful to compute the final flows f_2 and f_{-2} for the second commodity.

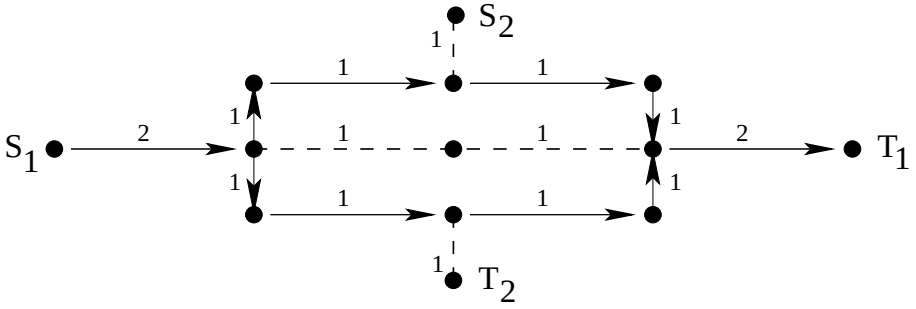


Figure 8: the cut criterion is true for $((\kappa - f), (s_2, t_2, 1), (t_2, s_2, 1))$

Algorithm 3 (second step)
Input : *$G = (V, A)$, $\kappa : A \to \mathbb{N}$; $s_2, t_2 \in V$; $v_2 \in \mathbb{N}$; $f : A \to \mathbb{N}$.*
Let f^r be the reverse flow from t_1 to s_1 of value v_1.

- *compute a flow $h \leq (\kappa - f)$ from s_2 to t_2 of value v_2*
- *compute a flow $h' \leq (\kappa - f^r)$ from s_2 to t_2 of value v_2*
- *using **Algorithm 1**, compute a flow f_2 from s_2 to t_2 of value v_2 such that $(|h + h'| - 1) \leq 2f_2 \leq (|h + h'| + 1)$.*

Let $f^r_{-2} = (|h + h'| - f_2)$. We call f_{-2} the reverse flow from t_2 to s_2 of value v_2.
Output : *$f_2 : A \to \mathbb{N}$; $f_{-2} : A \to \mathbb{N}$.*

Lemma 3 (second step) *Let $\kappa : A \to \mathbb{N}$ be a symmetric capacity. Let $s_1, t_1, s_2, t_2 \in V$ and $v_1, v_2 \in \mathbb{N}$. Let $f \leq \kappa$ be a flow from s_1 to t_1 of value v_1 such that the cut criterion is true for $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$. Then there are two flows f_2 from s_2 to t_2 of value v_2 and f_{-2} from t_2 to s_2 of value v_2 such that $(f_2 + f_{-2}) \leq \kappa$ and such that the cut criterion is true for $((\kappa - f_2 - f_{-2}), (s_1, t_1, v_1))$. These two flows can be computed by Algorithm 3 in $2C_{flow} + O(|A|)$ steps.*

We divide the proof into three parts : first the algorithm completes, then $(f_2 + f_{-2}) \leq \kappa$, and finally the cut criterion is true for $((\kappa - f_2 - f_{-2}), (s_1, t_1, v_1))$.

Proposition 5 *If the cut criterion is true for $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$, then Algorithm 3 completes in $2C_{flow} + O(|A|)$ steps.*

Proof
If the cut criterion is true for $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$, then the flows h and h' do exist. $|h + h'|$ is a flow from s_2 to t_2 of value $2 \times v_2$, so according to lemma 1, the flow f_2 can be computed by Algorithm 1. This algorithm computes 2 flows, calls Algorithm 1 and computes 4 functions so it completes in $(2C_{flow} + O(|A|))$ steps. □

Proposition 6 *The flows f_2 and f_{-2} given by Algorithm 3 are such that $(f_2 + f_{-2}) \leq \kappa$.*

Proof
We know that $2f_2 \leq (|h + h'| + 1)$ and $2f^r_{-2} \leq (|h + h'| + 1)$. Since $|h + h'| \leq \kappa$, we have $f_2 \leq \kappa$ and $f_{-2} \leq \kappa$. Moreover, $(f_2 + f^r_{-2}) = |h + h'|$ implies that $\forall (x, y) \in A$, if $f_2(x, y) > 0$, then $f^r_{-2}(y, x) = 0$, so $f_{-2}(x, y) = 0$. Thus $(f_2 + f_{-2}) \leq \kappa$. □

Proposition 7 *If $f \leq \kappa$ is a flow from s_1 to t_1 of value v_1, the flows f_2 and f_{-2} given by Algorithm 3 are such that the cut criterion for $((\kappa - f_2 - f_{-2}), (s_1.t_1, v_1))$ is true.*

Proof
Consider a cut $C \subset V$ such that $s_1 \in C$ and $t_1 \notin C$. First we will prove that $(|h + h'|(C) + |h + h'|(V \backslash C)) \leq (2\kappa(C) - 2v_1)$. This implies $((f_2 + f_{-2})(C) + (f_2 + f_{-2})(V \backslash C)) \leq (2\kappa(C) - 2v_1)$. Then we will prove that $(f_2 + f_{-2})(C) \leq (\kappa(C) - v_1)$.

We split $|h + h'|$ into two functions : g (the part related to h) and g' (the part related to h') so $g = \min(h, |h + h'|)$ and $g' = \min(h', |h + h'|)$. Observe that if $g(x, y) > 0$ then $g'(y, x) = 0$. This implies that $(g + g'^r) \leq (\kappa - f)$. Since h and h'^r are flows that go in opposite directions, we have $(h(C) - h(V \backslash C)) = (h'^r(V \backslash C) - h'^r(C))$. This implies $(g + g'^r)(C) = (g + g'^r)(V \backslash C)$. So we have $(g + g'^r)(C) + (g + g'^r)(V \backslash C) \leq 2(\kappa - f)(C)$, thus $|h + h'|(C) + |h + h'|(V \backslash C) \leq (2\kappa(C) - 2v_1)$.

Since $(f_2 + f^r_{-2}) = |h + h'|$, we have as well $(f_2 + f^r_{-2})(C) + (f_2 + f^r_{-2})(V \backslash C) \leq (2\kappa(C) - 2v_1)$, so $(f_2 + f_{-2})(C) + (f_2 + f_{-2})(V \backslash C) \leq (2\kappa(C) - 2v_1)$.

Like h and h', the flows f_2 and f_{-2} go in opposite directions, so $(f_2 + f_{-2})(C) = (f_2 + f_{-2})(V \backslash C)$. Therefore $(f_2 + f_{-2})(C) \leq (\kappa(C) - v_1)$. □

3.4 Finding the last two flows

Once f_2 and f_{-2} have been properly computed, the algorithm to find f_1 and f_{-1} is quite straightforward.

Algorithm 4 (third step)
Input : $G = (V, A)$, $\kappa : A \to \mathbb{N}$; $s_1, t_1 \in V$; $v_1 \in \mathbb{N}$; $f_2 : A \to \mathbb{N}$; $f_{-2} : A \to \mathbb{N}$.

- *compute a flow* $f_1 \leq (\kappa - f_2 - f_{-2})$ *from* s_1 *to* t_1 *of value* v_1

Let $f_{-1} = (\kappa - f_1 - f_2 - f_{-2})$
Output : $f_1 : A \to \mathbb{N}$ *;* $f_{-1} : A \to \mathbb{N}$.

Lemma 4 (third step) *If* (f_2, f_{-2}) *is a two-commodity flow from* (s_2, t_2) *to* (t_2, s_2) *of value* (v_2, v_2) *such that* $(f_2 + f_{-2}) \leq \kappa$*, and if the cut criterion is true for* $((\kappa - f_2 - f_{-2}), (s_1, t_1, v_1))$*, then Algorithm 4 completes in* $C_{flow} + O(|A|)$ *steps, and its output are two flows* f_1 *from* s_1 *to* t_1 *of value* v_1 *and* f_{-1} *from* t_1 *to* s_1 *of value* v_1 *such that* $(f_1 + f_{-1} + f_2 + f_{-2}) \leq \kappa$.

Proof
If the cut criterion is true for $((\kappa - f_2 - f_{-2}), (s_1, t_1, v_1))$, then the flow f_1 does exist. So Algorithm 4 takes $C_{flow} + O(|A|)$ steps. Now observe that the function $(\kappa - f_1 - f_2 - f_{-2}) : A \to \mathbb{N}$ is a flow from t_1 to s_1 of value v_1 (see definition 1), though it may have some loops (which could be easily removed in C_{flow} more steps). □

3.5 Connecting dots

Algorithm 5 (integral symmetric 2-commodity flow)
Input : $G = (V, A)$ *;* $\kappa : A \to \mathbb{N}$ *;* $s_1, t_1, s_2, t_2 \in V$ *;* $v_1, v_2 \in \mathbb{N}$.

- *using* ***Algorithm 2 (first step)****, compute a flow* f *from* s_1 *to* t_1 *of value* v_1 *such that the cut criterion is true for* $((\kappa - f), (s_2, t_2, v_2), (t_2, s_2, v_2))$
- *using* ***Algorithm 3 (second step)*** *and* f*, compute two flows* f_2 *from* s_2 *to* t_2 *of value* v_2 *and* f_{-2} *from* t_2 *to* s_2 *of value* v_2 *such that* $(f_2 + f_{-2}) \leq \kappa$ *and such that the cut criterion is true for* $((\kappa - f_2 - f_{-2}), (s_1, t_1, v_1))$.
- *using* ***Algorithm 4 (third step)****, compute two flows* f_1 *from* s_1 *to* t_1 *of value* v_1 *and* f_{-1} *from* t_1 *to* s_1 *of value* v_1 *such that* $(f_1 + f_{-1} + f_2 + f_{-2}) \leq \kappa$.

Output : $f_1 : A \to \mathbb{N}$ *;* $f_{-1} : A \to \mathbb{N}$ *;* $f_2 : A \to \mathbb{N}$ *;* $f_{-2} : A \to \mathbb{N}$.

Theorem 5 (symmetric 2-commodity flow). *The symmetric cut criterion is a necessary and sufficient condition for the existence of a solution to the Integral Symmetric 2-Commodity Flow Problem. A solution can be found by Algorithm 5 in* $6C_{flow} + O(|A|)$ *steps.*
Proof
According to corollary 1, the symmetric cut criterion is a necessary condition for the existence of a solution to our problem. The input is :

- a symmetric digraph $G = (V, A)$;
- a symmetric capacity $\kappa : A \to \mathbb{N}$;
- the source and the target (s_1, s_2), $(t_1, t_2) \in V^2$;
- the value $(v_1, v_2) \in \mathbb{N}^2$.

If the symmetric cut criterion is true for $(\kappa, (s_1, t_1, v_1), (s_2, t_2, v_2))$ then according to lemma 2, Algorithm 2 takes $3C_{flow} + O(|A|)$ steps ; according to lemma 3, Algorithm 3 takes $2C_{flow} + O(|A|)$ steps and according to lemma 4, Algorithm 4 takes $C_{flow} + O(|A|)$ steps ; so Algorithm 5 completes in $6C_{flow} + O(|A|)$ steps.

Moreover according to lemmas 2, 3 and 4 the 4-commodity flow $(f_1, f_{-1}, f_2, f_{-2})$ computed by Algorithm 5 is a symmetric 2-commodity flow from (s_1, s_2) to (t_1, t_2) of value (v_1, v_2) such that $(f_1 + f_{-1} + f_2 + f_{-2}) \leq \kappa$, so it is a solution to the problem. □

4 Open problems

We have proven that the Integral Symmetric 2-Commodity Flow Problem was polynomial. Now we would like to know more on the complexity of the Integral Symmetric Multi-Commodity Flow Problem (likely NP-complete), and on Integral Symmetric k-commodity Flow Problems, with $k > 2$.

If the requests are not symmetric, the complexity of Integral k Commodity Flow Problems in symmetric digraphs (with a symmetric capacity) is also open for $k > 1$.

References

[Cha98] Pascal Chanas. *Réseaux ATM : Conception et optimisation.* PhD thesis, Université de Grenoble, June 1998. France Télécom CNET.

[EIS76] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. Comput.*, 5(4):691–703, December 1976.

[FHW80] Steven Fortune, John Hopcroft, and James Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, (10):111–121, 1980.

[Jar02] A. Jarry. Disjoint Paths in Symmetric Digraphs. In *International Colloquium on Structural Information and Communication Complexity – SIROCCO*, pages 211–222, Andros, Greece, June 2002. Carleton.

[Kar75] R.M. Karp. On the complexity of combinatorial problems. *Networks*, (5):45–68, 1975.

[RS95] N. Robertson and P. D. Seymour. Graph Minors XIII. The Disjoint Paths Problem. *J. Combin. Theory Ser. B 63*, pages 65–110, 1995.

[Vyg94] Jens Vygen. Disjoint paths. Technical Report 94816, Research Institute for Discrete Mathematics, University of Bonn, February 1994. updated September 1998.

[Vyg95] Jens Vygen. NP-Completeness of Some Edge-Disjoint Paths Problems. *Discrete Applied Mathematics*, (61):83–90, 1995.



Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis 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 - 38330 Montbonnot-St-Martin (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