A Branch and Price Algorithm for the k-splittable Maximum Flow Problem

Jérôme Truffot, Christophe Duhamel

A Branch and Price Algorithm for the k-splittable Maximum Flow Problem

Jérôme Truffot[1], Christophe Duhamel[2]
LIMOS, UMR 6158-CNRS
Université Blaise-Pascal, BP 10125, 63173 Aubière, France.

Research Report LIMOS/RR06-04

[1]jerome.truffot@isima.fr
[2]christophe.duhamel@isima.fr

Abstract

The Maximum Flow Problem with flow width constraints is a NP-hard problem. Two models are proposed: the first model is a compact node-arc model using two flow conservation blocks per path. For each path, one block defines the path while the other one send the right amount of flow on it. The second model is an extended arc-path model. It is obtained from the first model after a Dantzig-Wolfe reformulation. it is an extended model as it relies on the set of all the paths between the source and the sink nodes. Some symmetry breaking constraints are used to improve the model. A branch and price algorithm is proposed to solve the problem. The column generation reduces to the computation of a shortest path whose cost depends on weights on the arcs and on the path capacity. A polynomial time algorithm is proposed to solve this subproblem. Computational results are shown on a set of medium-sized instances to show the effectiveness of our approach.

Keywords: max flow, flow width, column generation, branch and price

Résumé

Nous nous intéressons au problème de la recherche d'un flot maximal dans un graphe avec une contrainte sur la largeur de flot. Cette contrainte rend le problème NP-difficile. Nous proposons un modèle compact impliquant deux blocs de conservation de flot pour chaque chemin. Le premier bloc permet de définir le chemin tandis que le second achemine la quantité de flot associée à ce chemin. Nous proposons ensuite un modèle étendu issu d'une reformulation de type Dantzig-Wolfe. Ce modèle est étendu dans la mesure où les variables sont indicées sur l'ensemble des chemins de la source au puits. L'utilisation de contraintes d'élimination de symétrie permet de renforcer le modèle. Nous mettons ensuite en place un mécanisme de branch and price pour résoudre le problème. La phase de génération de colonnes se ramène au calcul d'un plus court chemin dont le coût dépend de poids sur les arcs mais aussi de la capacité de ce chemin. Nous montrons que ce sous-problème peut néanmoins être résolu par un algorithme polynômial. Des résultats expérimentaux sont présentés sur un jeu d'instances de taille moyenne afin d'illustrer l'intérêt de cette approche.

Mots clés : flot maximal, largeur de flot, génération de colonnes, branch and price

Acknowledgements / Remerciements

This work was partialy supported by the french ACI-PRESTO project. The authors wish to thank professor Philippe Mahey for his helpful comments.

1 Introduction

In this paper, we consider a single-commodity k-splittable Flow Problem, a generalization of the Unsplittable Flow Problem (UFP) in which flow may use at most k paths from origin to destination (k=1 for the UFP). More precisely, we consider the k-splittable Maximal Flow Problem (KMFP) which consists in routing the maximum amount of flow, split between at most k paths. KMFP is NP-hard, as a generalization of the 2-splittable maximum flow problem [4] (which the 3-SAT problem can be reduced to).

This problem finds an application in telecommunication networks. New generation telecommunication networks (UMTS) allow the integration of Quality of Service (QoS) requirements on the traffic through routing protocols such as MPLS. An important feature of MPLS is its ability to set up traffic engineering mechanisms (MPLS-TE). For instance, MPLS-TE allows the traffic manager to put constraints on the end-to-end QoS. It also provides means to control the structure of the traffic for each customer by setting restrictions on the number of routes. The purpose of such restrictions is twice: first, keep a traffic structure as simple as possible and second, keep a low overall number of routes, while preserving a good end-to-end QoS.

This kind of restriction on the number of routes seems to be rather new in the literature even if there are strong connections to the UFP (unsplittable flow [2, 3, 5]) and disjoint paths [12]. Some previous works consider the problem of path number in multi-path routing but without a bounded number of paths ([7, 20]). To our knowledge, Baier *et al.* [4] were the first to introduce the k-splittable flow problem. They propose an approximation algorithm for the k-splittable maximum flow problem. Recently, Kolliopoulos [14] proved the existence of a (2,1)-approximation algorithm for a 2-splittable minimum cost flow problem. Martens and Skutella propose variants of k-splittable problem in [17] and length-bounded and dynamic k-splittable flows in [18]. Koch *et al.* [13] present approximation algorithms and complexity results for k-splittable flow problems.

In this work, we will focus on several formulations for the k-splittable maximum flow problem (KMFP) and then we apply a dedicated branch and price algorithm to compute the optimal solution. This paper will be organized as follows: in section 2, several mathematical formulations for KMFP will be presented. In section 3, the application of Branch and Price to solve the problem will be discussed. Section 4 will be devoted to numerical experiments, before conclusion are made in section 5.

2 Mathematical formulations

Let $G = (N, A)$ be a digraph where N is the set of n nodes and A is the set of m arcs. Each arc $a \in A$ is given a capacity $u_a \geqslant 0$. Let (s, t) be the origin-destination pair of the flow to route on G. Let H be the maximal number of elementary paths to carry out the traffic. The k-splittable Maximum Flow Problem (KMFP) is to find a maximum flow such that at most H paths are used.

Definition 1 (width)**.** *Let F be a feasible flow over G. The width $w(F)$ is the minimal number of routes such that the aggregation of the flow on each route will give exactly F.*

Given a flow F, the general question of computing its width is a NP-Hard problem (see [24]). However, it is polynomial on trivial cases like, for instance on disjoints paths. The width is treated a different way in the KMFP. The goal is not to compute its width, but rather to maintain its width under a given threshold.

In the following sections, we will propose three models for the problem. The first one is a basic arc-path formulation. As no efficient way to solve it were found, we describe each path as a

flow subproblem and define an arc-node formulation for the problem. By performing a Dantzig-Wolfe reformulation, we defined a new arc-path model on which we will apply our Branch and Price algorithm.

2.1 Basic arc-path formulation

The first model is based on the arc-path formulation. Let $\mathcal{P}$ be the potentialy exponential set of all the elementary $(s-t)$ paths. Let $u_p = \min_{a \in p}\{u_a\}$ be the capacity of path p and δ_a^p be the indicator vector that identifies which arc $a \in A$ belongs to path p. Let $x_p \geqslant 0$ be the flow variable on the path $p \in \mathcal{P}$ and $y_p \in \{0,1\}$ be the associated decision variable. Then the arc-path model is as follows:

$$
(\text{KMFP}_1)\begin{cases}
\max \displaystyle\sum_{p\in\mathcal{P}} x_p & & \\
\text{s.t.} & & \\
\quad \displaystyle\sum_{p\in\mathcal{P}} \delta_a^p x_p \leqslant u_a & \forall a \in A & (a) \\
\quad x_p - u_p y_p \leqslant 0 & \forall p \in \mathcal{P} & (b) \\
\quad \displaystyle\sum_{p\in\mathcal{P}} y_p \leqslant H & & (c) \\
\quad x_p \geqslant 0 & \forall p \in \mathcal{P} & (e) \\
\quad y_p \in \{0,1\} & \forall p \in \mathcal{P} & (f)
\end{cases} \tag{1}
$$

Constraints $1(a)$ are the capacity constraints. The link between flow and decision variables is made in $1(b)$ and $1(c)$ is the width constraint. Note that this model relies on a potentially exponential number of variables and constraints.

As will be shown in section 3.2, this model cannot be used an efficient way in a branch and bound scheme. Thus reformulations are needed.

2.2 Arc-node formulation

Rather than choosing H paths in $\mathcal{P}$, each one of this H paths can be described as a flow subproblem, using arc-node formulation. Then, we define a more compact model. This discretization imposes a classification on the H paths. For each path number $h = 1 \ldots H$, let $x_a^h \geqslant 0$ be the flow variable on the arc $a \in A$, let $y_a^h \in \{0,1\}$ be the associated decision variable and let $z_h \geqslant 0$ be the amount of flow. Let ω_v^- be the cocyle of node v incoming arcs and let ω_v^+ be the cocyle of node v outgoing arcs. Then the arc-node model can be stated as follows:

$$
(\text{KMFP}_2)\begin{cases}
\max \displaystyle\sum_{h=1}^{H} z_h & & & \\
\text{s.t.} & & & \\
\displaystyle\sum_{a\in\omega_v^-} x_a^h - \sum_{a\in\omega_v^+} x_a^h = \begin{cases} z_h & \text{if } v=t, \\ -z_h & \text{if } v=s, \\ 0 & \text{otherwise.} \end{cases} & & \forall h=1\dots H,\ \forall v\in N & (a) \\
\displaystyle\sum_{a\in\omega_v^-} y_a^h - \sum_{a\in\omega_v^+} y_a^h = \begin{cases} 1 & \text{if } v=t, \\ -1 & \text{if } v=s, \\ 0 & \text{otherwise.} \end{cases} & & \forall h=1\dots H,\ \forall v\in N & (b) \\
\displaystyle\sum_{h=1}^{H} x_a^h \leqslant u_a & & \forall h=1\dots H,\ \forall a\in A & (c) \\
x_a^h \leqslant u_a y_a^h & & \forall h=1\dots H,\ \forall a\in A & (d) \\
\displaystyle\sum_{a\in\omega_v^-} y_a^h \leqslant 1 & & \forall h=1\dots H,\ \forall v\in N & (e) \\
x_a^h \geqslant 0 & & \forall h=1\dots H,\ \forall a\in A & (f) \\
y_a^h \in \{0,1\} & & \forall h=1\dots H,\ \forall a\in A & (g) \\
z_h \geqslant 0 & & \forall h=1\dots H & (h)
\end{cases} \tag{2}
$$

This model involves two flow conservation blocks (namely constraints $2(a)$ and $2(b)$) for each path. $2(c)$ are the capacity constraints and $2(d)$ are the coupling constraints. Restrictions $2(e)$ are used to force each path h to be elementary that is, to prevent cycles to be connected to the path. Otherwise, one could not prevent the flow bifurcation illustrated in figure 1. Arcs a correspond to decision variables $y_a^h = 1$ and, for each one, the flow value x_a^h is reported. In such a situation, a cycle on path definition variables y^h may help define more than one flow path on variables x^h. Restrictions $2(d)$ do not prevent disconnected cycles, as shown in figure 2. However, since such cycles cannot lead to flow bifurcation, this situation does not need to be forbidden.

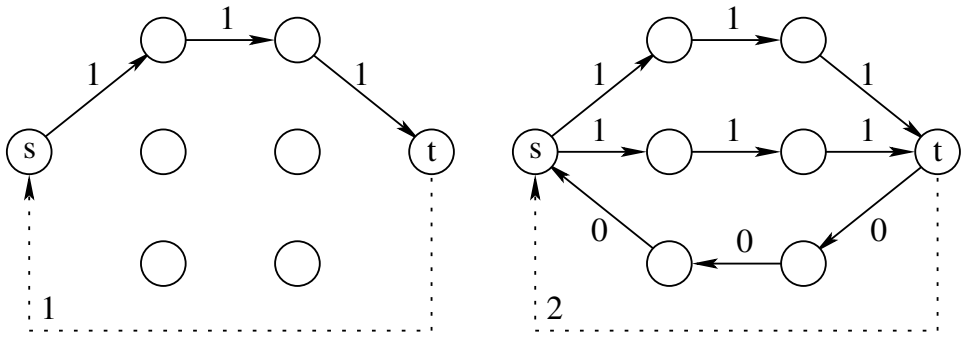


Figure 1: impact of a cycle in flow definition y^p over the flow variables x^p.

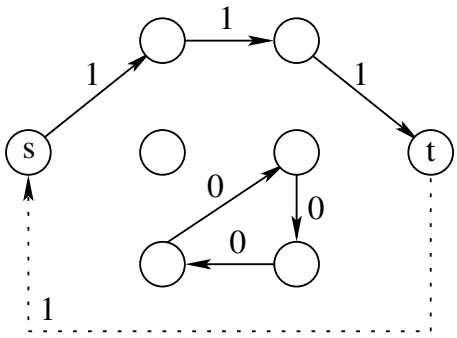


Figure 2: impact of a disconnected cycle over the flow variables x^p.

2.3 Arc-path reformulation

By performing a Dantzig-Wolfe reformulation of model (KMFP$_2$), a new arc-path model can be defined. It differs from model (KMFP$_1$) since it relies on a discretization on decision and flow variables. Let $x_p^h \geqslant 0$ be the flow of path p when p is used as path number h. Let $y_p^h \in \{0,1\}$ be the corresponding decision variable. This new model is as follows:

$$
(\text{KMFP}_3) \begin{cases} \max \displaystyle\sum_{h=1}^{H} \sum_{p \in \mathcal{P}} x_p^h & & \\ \text{s.t.} & & \\ \displaystyle\sum_{h=1}^{H} \sum_{p \in \mathcal{P}} \delta_a^p x_p^h \leq u_a & \forall a \in A & (a) \\ x_p^h - u_p y_p^h \leqslant 0 & \forall h = 1 \dots H,\ \forall p \in \mathcal{P} & (b) \\ \displaystyle\sum_{p \in \mathcal{P}} y_p^h \leqslant 1 & \forall h = 1 \dots H & (c) \\ & & \\ x_p^h \geqslant 0 & \forall h = 1 \dots H,\ \forall p \in \mathcal{P} & (d) \\ y_p^h \in \{0,1\} & \forall h = 1 \dots H,\ \forall p \in \mathcal{P} & (e) \end{cases} \tag{3}
$$

When compared to model (KMFP$_1$), model (KMFP$_3$) does not need any restriction on the number of active paths anymore since it is implicitely assumed throught the variable discretization. However, an additional assignment step is required for the path variables, through constraints $3(c)$. Note that the solution of model (KMFP$_3$) can always be translated into a solution of model (KMFP$_1$) using the following formulas:

$$
x_p = \sum_{h=1}^{H} x_p^h \quad \forall p \in \mathcal{P} \tag{4}
$$

$$
y_p = \max_{h=1 \dots H} y_p^h \quad \forall p \in \mathcal{P} \tag{5}
$$

In fact, the relashionship between those two models is even stronger, as shown below:

Property 1. *Models (KMFP$_1$) and (KMFP$_3$) are equivalent.*

Proof. Let $(\overline{x}_p^h, \overline{y}_p^h)$ be a fractional solution of model (KMFP$_3$). Let $x_p = \sum_h \overline{x}_p^h$ and $y_p = \sum_h \overline{y}_p^h$ be the aggregated variables. The solution (x_p, y_p) satisfies all the constraints of the model (KMFP$_1$).

Let $(\overline{x}_p, \overline{y}_p)$ be a fractional solution of model (KMFP$_1$). Let $x_p^h = \overline{x}_p/H$ and $y_p^h = \overline{y}_p/H$ be the disaggregated variables. The solution (x_p^h, x_p^h) satisfies all the constraints of the model (KMFP$_3$).

Thus $Conv$(KMFP$_3$) $= Conv$(KMFP$_1$). □

Now, when comparing to the model (KMFP$_2$), the situation is different:

Property 2. *Model (KMFP$_3$) is stronger than model (KMFP$_2$).*

Proof. Let $(\overline{x}_p^h, \overline{y}_p^h)$ be a fractional solution of model (KMFP$_3$). Let $x_a^h = \sum_p \delta_a^p \overline{x}_p^h$ and $y_a^h = \sum_p \delta_a^p \overline{y}_p^h$ be the projection on the arc variables. The solution (x_a^h, y_a^h) satisfies all the constraints of the model (KMFP$_2$).

The reverse does not hold, as any flow on the arc variables may be decomposed into a set of elementary paths and elementary cycles (see Ahuja et al [1]). Thus, as the model (KMFP$_2$) does not forbid cycles, many solutions from the model (KMFP$_2$) cannot be translated into solutions from the model (KMFP$_3$), see for instance the figure 2.

Therefore, the model (KMFP$_3$) is stronger than the model (KMFP$_2$). □

2.4 Improvements

One obvious drawback of model (KMFP$_3$) is its size. However, there is another one, closely related to the symmetry structure induced by the assignment of the decision variables [22]. Namely, assuming set $P = (p_1, p_2, \ldots, p_H)$ is an optimal set of paths for model (KMFP$_3$), any permutation of P gives an optimal solution. As any path can be set at any position in the routing solution, it can potentially lead to a big number of "identical" solutions. One way to break this variable symmetry is to introduce the so-called variable-ordering constraints. In our situation, stating that the path in position $h+1$ is required to have less flow than the path in position h is sufficient for most of the cases. However, this cannot break ties when, in the optimal solution, at least two paths carry the same amount of flow. The variable ordering is achieved through the following additional constraint:

$$\sum_{p \in \mathcal{P}} x_p^{h+1} - \sum_{p \in \mathcal{P}} x_p^h \leqslant 0 \quad \forall h = 1 \ldots H-1 \tag{6}$$

3 Branch and Price

One popular and efficient way to solve a MILP in extended formulation is to apply the branch and price scheme. It consists in embedding a column generation into a branch and bound framework. Branch and bound principle was presented by Land and Doig in [15]. Shortly before, Ford and Fulkerson ([10]) suggested column generation for multicommodity flow problem and Danzig and Wolfe ([9]) developed this idea in their well-known decomposition scheme. Finally, Barnhart *et al.* [6] and Vanderbeck and Wolsey [23] described generic algorithms for solving problems by integer programming column generation. Many applications are presented in the literature, as can be seen in the survey of Lübbecke and Desrosiers ([16]).

3.1 Column generation

The column generation is used to solve linear programs with a large number of variables (columns). It is based on the implicit knowledge of the whole set $\mathcal{X}$ of variables. At each iteration, it first solves a restricted master problem (RMP) and then solves a subproblem (SP) before updating the (RMP). The (RMP) consists in a restriction of $\mathcal{X}$ to a feasible subset of variables $S \subset \mathcal{X}$. Once the (RMP) has been solved to optimality on S, one has to check wether there exist improving variables in $\mathcal{X} \setminus S$ or not. This is done through the pricing procedure in (SP): using the dual information of the (RMP), the most violated reduced cost of a variable in $\mathcal{X} \setminus S$ is computed. If this reduced cost is improving, the associated variable is inserted into the (RMP) for subsequent iterations. Otherwise no improving variables exist, the column generation is stopped and the optimal solution of the (RMP) is the proven optimal solution of the problem.

In order to apply the branch and price technique on model (KMFP$_3$), it is first continuous relaxed. Then, it is simplified using the following property:

Property 3. *There is at least one optimal solution of the linear relaxation of (KMFP$_3$) where the coupling contraints are saturated.*

Proof. Let (x^*, y^*) be an optimal solution. Let $\overline{y_p^h} = x_p^{h*}/u_p$ if $u_p > 0$ and 0 otherwise, for each p in $\mathcal{P}$ and for each $h = 1 \dots H$. Then constraints $3(b)$ imply that $\overline{y_p^h} \leqslant y_p^{h*}$, for each p in $\mathcal{P}$ and for each $h = 1 \dots H$. So $\sum_{p\in\mathcal{P}} \overline{y_p^h} \leqslant \sum_{p\in\mathcal{P}} y_p^{h*} \leqslant 1$ for each $h = 1 \dots H$ (constraints $3(c)$). Then $(x^*, \overline{y})$ is a feasible solution and provides the same value than (x^*, y^*). Therefore, $(x^*, \overline{y})$ is an optimal solution too. □

Thus, coupling constraints can be dropped and the decision variables y can be replaced with the flow variables x. This leads to a simplified version of the linear relaxation, including the variable-ordering constraints:

$$
(\mathrm{LR}_3)\begin{cases}
\max \displaystyle\sum_{h=1}^{H}\sum_{p\in\mathcal{P}} x_p^h & & & \\
\text{s.t.} & & & \\
\quad \displaystyle\sum_{h=1}^{H}\sum_{p\in\mathcal{P}} \delta_a^p x_p^h \leqslant u_a & \forall a \in A & (a) \\
\quad \displaystyle\sum_{p\in\mathcal{P}} \frac{x_p^h}{u_p} \leqslant 1 & \forall h = 1 \dots H & (b) \\
\quad \displaystyle\sum_{p\in\mathcal{P}} x_p^{h+1} - \sum_{p\in\mathcal{P}} x_p^h \leqslant 0 & \forall h = 1 \dots H-1 & (c) \\
\quad x_p^h \geqslant 0 & \forall h = 1 \dots H \quad \forall p \in \mathcal{P} & (c)
\end{cases} \tag{7}
$$

Let respectively $\lambda_a \geqslant 0$, $\mu^h \geqslant 0$ and $\nu^h \geqslant 0$ be the dual variables associated to the primal constraints $7(a)$, $7(b)$ and $7(c)$ of the linear relaxation (LR$_3$). Then, the subproblem (SP) consists in finding a variable maximizing the reduced cost. (SP) can be decomposed into H subproblems (SP^h). Each of them reduces to an optimal cost elementary path problem for position h. The reduced cost is given by

$$
\overline{c_p^h} = 1 - \sum_{a\in A} \delta_a^p \lambda_a - \frac{\mu^h}{u_p} - (\nu^{h-1} - \nu^h) \tag{8}
$$

The computation of the maximal cost elementary path does not reduce to a simple shortest path problem, since the reduced cost involves a combination of two dual variables, λ and μ (the cost associated to the variable-ordering constraints only depend on the path position and not on the arcs of the path). Given a path position h, the subproblem is as follows:

$$(\text{SP}^h) \begin{cases} \min\ w = \sum_{a \in A} \lambda_a \delta_a^p + \mu^h / u_p - 1 + (\nu^{h-1} - \nu^h) \\ \text{s.t.} \\ \quad u_p = \min\{u_a, \delta_a^p = 1; a \in A\} \\ \quad \delta_a^p \in \mathcal{P} \end{cases} \tag{9}$$

More precisely, for any optimal path problem, Martin [19] defined the *weak optimality principle* as the fact that there is an optimal path made of optimal subpaths. He then showed that this principle is a necessary condition to the application of any labelling algorithm (for instance, the classical label setting / label correcting algorithms for the shortest path problem).

Unfortunately, our subproblem (SP) does not meet Martin's weak optimality principle, as illustrated in figure 3. For each arc a, the first number refers to its dual variables λ_a while the second one refers to its capacity u_a. The variable μ is supposed to be set to 4. The shortest path from node s to node v uses the arc a_2 since $\lambda_1 + \mu/c_1 = 5 > \lambda_2 + \mu/c_2 = 4$. The shortest path from node s to node t uses the path $a_2 - a_3$ since $\lambda_1 + \lambda_3 + \mu/c_1 = 5 < \lambda_2 + \lambda_3 + \mu/c_3 = 7$.

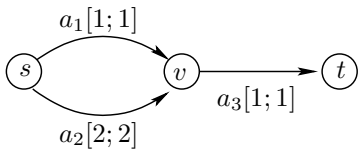


Figure 3: failure of the weak optimality principle

Thus, no labelling algorithm can be used to compute the optimal solution [19] and a specific algorithm has to be designed.

In (SP^h), the variables δ_a^p define the path p as they are the projection of variables y_p over the arcs while u_p defines its capacity. The optimal path p^* is a combination between the shortest path and the highest capacity path. Its computation can be done in the following way:

Property 4. *Algorithm 1 runs in polynomial time*

Proof. There is no negative cost arc since $\lambda_a \geqslant 0, \forall a \in A$. Thus, no negative cost cycle exists and any polynomial-time shortest path algorithm may be used. Next, at each iteration of the algorithm, at least one arc is removed from the set A'. Thus, there is at most m iterations, that is m computations of a shortest path. □

Since the columns might be used anywhere in the branching tree, they are inserted into a global pool. The column generation will stop as soon as $w^* \leqslant 0$.

3.2 Classical branchings

Since the column generation works on the linear relaxation of the initial problem, one has to perform branchings. The classical branching scheme of Dakin [8] cannot be applied as it works

Algorithm 1 optimal path computation for SP^h

```
let A' = A
let p* = ∅, u* = 0, w* = ∞
repeat
  compute a shortest path p from s to t on A'
  if p = ∅ then
    break
  else
    let u be its capacity, w its cost
    if w < w* then
      p* ← p
      u* ← u
      w* ← w
    end if
    A' ← A' \ {a ∈ A' | u_a ⩽ u}
  end if
until A' = ∅
return (p*, u*, w*)
```

on the original variables and as it is quite difficult to prevent the generation of a column that has already been forbidden in the current branch. Ryan and Foster [21] were among the first to propose a safe branching scheme ($(x = y) \vee (x \neq y)$). Instead of forcing or forbidding a decision variable, their idea relies on the fact that either two variables are set the same way or not.

More recently, Barnhart *et al.* [5] proposed a more efficient branching for the routing problems. It is based on the concept of node of divergence over the aggregated flow $x_a^h = \sum_p \delta_a^p x_p^h,\ \forall a \in A$. A node of divergence is a node $d \in N$ such that the aggregated flow is coming from a single arc and going out on several arcs, see figure 4. Given a position $h \leqslant H$, a divergence occurs when the flow decision variables y_p^h are fractional. On figure 4, those fractions are respectively $7/10$, $1/10$ and $2/10$. Thus, a fractional part of each path p is used.

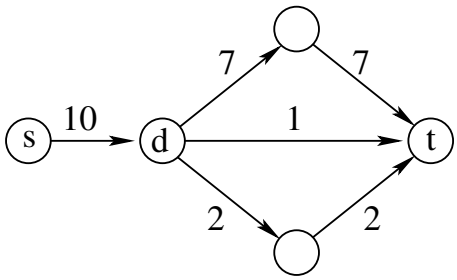


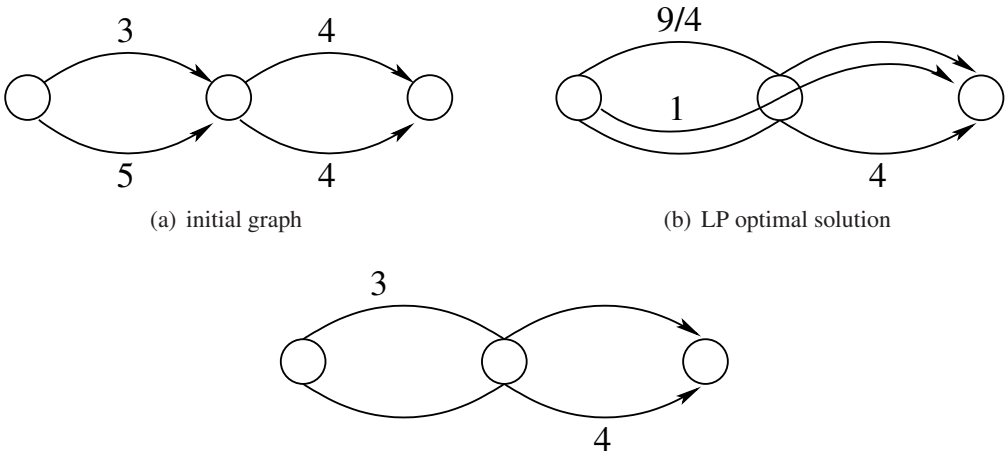
Figure 4: divergence on aggregated flow x^h on node d.

Let ω_d^+ be the cocycle of arcs going out of node d. Let ω_1 and ω_2 be a partition of ω_v^+ such that each set contains at least one arc carrying a positive amount of flow. Let $P_1 \subset \mathcal{P}$ and $P_2 \subset \mathcal{P}$ be the set of paths going through node d and using one arc of respectively ω_1 and ω_2. Since the flow on x^h has to be integer (that is, unsplittable), either it uses one arc in the set ω_1 or in the set ω_2. Then the following branching is valid:

$$(\sum_{p \in P_1} y_p^h = 0) \quad \text{vs.} \quad (\sum_{p \in P_2} y_p^h = 0)$$

For a better efficiency, the branching should be applied on the first node of divergence and the sets P_1 and P_2 should be built such that each sum is the most fractional.

It can be noted that this kind of branching cannot be easily applied to the first model (KMFP$_1$) as more than one path may be needed at the node of divergence. This situation is illustrated on figure 5. Figure 5(a) shows a graph whose capacities are reported. We wish to send the maximal amount of flow from s to t using at most $H = 2$ paths. Figure 5(b) illustrates the optimal relaxed solution. All the capacities are satisfied by the aggregated flow and the sum $\sum_{p \in \mathcal{P}} y_p = \sum_{p \in \mathcal{P}} x_p/u_p = 3/4 + 1/4 + 1 = 2$ satisfies the path limit H. Clearly, this solution uses three paths and no branching as defined previously can be done. The optimal integer solution is reported in figure 5(c).



(a) initial graph

(b) LP optimal solution

(c) IP optimal solution

Figure 5: divergence on aggregated flow x on node d.

Three paths are currently using node d and the limit is set to 2. However, the optimal solution uses two paths at d and no arc partition may help converging towards this solution.

3.3 Alternate branching

One of the shortcoming of the previous branching scheme is that it might require a careful partitioning procedure in order to be efficient. And even then, its efficiency will come from a combination of several successive branchings. We propose another branching scheme based on the number of path positions using an arc.

Let $a \in A$ be an arc and $S \subset \mathcal{H}_a$ be a subset of the path positions that use a. Then either all the path positions use a or at least one of them is not using a. More formaly, the branching is as follows:

$$(\sum_{h \in S} y_a^h = |S|) \quad \text{vs.} \quad (\sum_{h \in S} y_a^h \leqslant |S| - 1)$$

The first branch implies that all the flow for the positions in S goes through a. Then

$$(\sum_{h\in S} y_a^h = |S|) \quad \Rightarrow \quad (\sum_{h\in S}\sum_{p\in\mathcal{P}} x_p^h \leqslant u_a)$$

There is no equivalence as a linear combination of several paths may be used to route the flow at any path position in the relaxed solution – not only one path. Thus, this restriction is stronger than the capacity contraint $3(a)$:

$$\sum_{h\in S}\sum_{p\in\mathcal{P}} \delta_a^p x_p^h \leqslant \sum_{h\in S}\sum_{p\in\mathcal{P}} x_p^h \leqslant u_a$$

When switching back to the variables on the arcs, this branch becomes:

$$(\sum_{h\in S}\sum_{p\in\mathcal{P}} x_p^h \leqslant u_a) \quad \Leftrightarrow \quad (\sum_{h\in S}\sum_{a'\in\Omega_s^+} x_{a'}^h \leqslant u_a)$$

The other branch is equivalent to state that at least one of those path position should not route anything on a. It can be reformulated as follows

$$(\sum_{h\in S} y_a^h \leqslant |S|-1) \quad \Leftrightarrow \quad (\exists h \in S \mid y_a^h = 0)$$

If the subset S is reduced to a single path position, then the branching looks somewhat simpler:

$$(\sum_{a'\in\Omega_s^+} x_{a'}^h \leqslant u_a) \quad \text{vs.} \quad (y_a^h = 0)$$

The branching can only be done if there is a path position h such that the fractional flow is routing more than the capacity of an arc a used by h. Otherwise, a classical branching must be done. Even if the branching can be expressed in the arc-path formulation, it is better to use the node-arc formulation: as a branch leads to a new constraint in the RMP, a new dual variable will be added. This will alter the pricing subproblem and make it harder to solve. Thus, one solution is to use the node-arc formulation for the branching and link the node-arc variables with the arc-path variables. This is close to the Explicit Master formulation in the Robust Branch and Cut and Price (RBCP) proposed by Fukasawa et al. [11]. The RMP now looks like:

$$(\text{LR}_4)\begin{cases} \max \displaystyle\sum_{h=1}^{H}\sum_{p\in\mathcal{P}} x_p^h & & & \\ \text{s.t.} & & & \\ \quad \displaystyle\sum_{p\in\mathcal{P}} \delta_a^h x_p^h - x_a^h \leqslant 0 & \forall h = 1\dots H,\ \forall a \in A & (a) & \\ \quad \displaystyle\sum_{h=1}^{H}\sum_{p\in\mathcal{P}} \delta_a^p x_p^h \leqslant u_a & \forall a \in A & (b) & (10) \\ \quad \displaystyle\sum_{p\in\mathcal{P}} \frac{x_p^h}{u_p} \leqslant 1 & \forall h = 1\dots H & (c) & \\ \quad x_p^h \geqslant 0 & \forall h = 1\dots H \quad \forall p \in \mathcal{P} & (d) & \\ \quad x_a^h \geqslant 0 & \forall h = 1\dots H \quad \forall a \in A & (e) & \end{cases}$$

Then the branching will be done on the x_a^h variables while the column generation will work on the x_p^h variables. Those two kinds of variables are linked through the contraints $10(a)$.

3.4 Initialization

One crucial question that arises when implementing a column generation is the construction of the initial set of columns. For minimum cost flow problems, this may requires a carefull initialisation. The situation is different for the KMFP since even no path for each path position might lead to an initial feasible solution.

However, a better way to start is to compute an initial feasible solution, with respect to the width constraint. This can be done by using Baier *et al.* [4] approximation algorithm. It is based on an iterative insertion of a new path into the solution:

Algorithm 2 approximation algorithm for the KMFP

let q_a be the number of paths using arc a, initialy $q_a \leftarrow 0$
let P be the set of paths, initialy $P \leftarrow \emptyset$
let f be the flow carried by each path, initialy $f \leftarrow 0$
repeat
  let $G' = (N, A')$ be the residual graph with capacities $c_a = u_a/(1 + q_a)$
    backward arcs have capacities $c_a = f$
  compute a max capacity path p from s to t on G'
  let c be its capacity
  update P: $P \leftarrow P \cup \{p\}$
  for all backward arc $a \in p$ **do**
    update p and a path $p' \in P$ using arc a (deviation)
  end for
  update f: $f \leftarrow c$ for each path
until $p = \emptyset$ or $|P| = H$
return (P, f)

This algorithm runs in polynomial time. The set P of paths can then be used as the initial set of columns.

4 Numerical results

4.1 Instances

Two kinds of instances have been used to run our experiments. The first set of instances has been generated by the Transit Grid generator developped by G. Waissi [1]. The topology of those instances (see figure 6) looks close to the transportation networks and may be well-suited for studying the maxflow problem in the telecommunication networks as well. The second kind of topology in completely randomly generated: the edges are randomly chosen so as to have a connected network and the capacity of those edges is randomly chosen.

[1] `http://www.informatik.uni-trier.de/~naeher/Professur/research/generators/maxflow/tg/index.html`

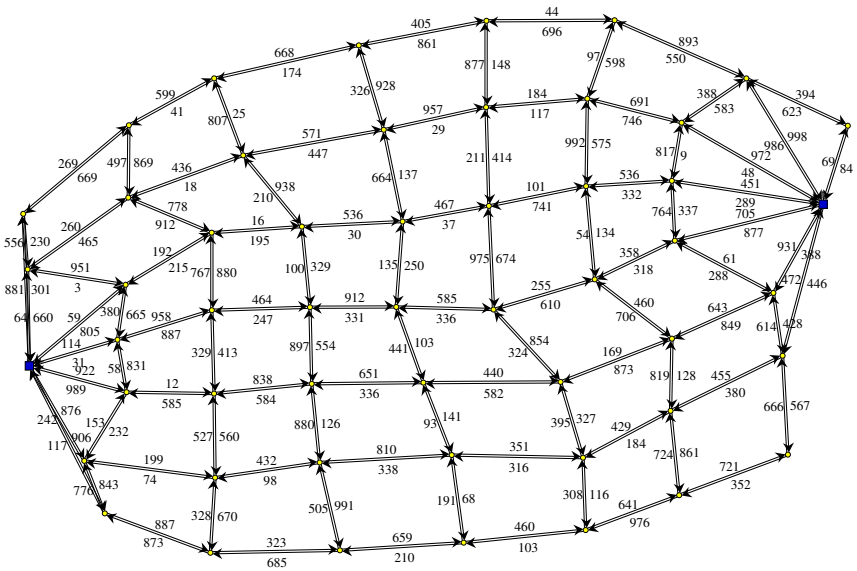


Figure 6: transit grid with 52 nodes and 198 arcs.

4.2 Results

The experiments were done on a Pentium 4 computer with 4Gb memory. The time limit has been set to 1 hour for each run.

For each table (table 1 to table 5) the column "Graph" gives the name of the instance, along with the number of nodes and the number of arcs. The width upper limit H is reported in the second column. The column z^* gives the optimal value, whenever available by at least one of our algorithms in the time limit. Otherwise, the sign "$\geqslant$" shows only a lower bound is known (the best integer found so far). The value of the solution obtained by the algorithm from Baier, Kölher et Skutella [4] is reported in column "BKS". As the CPU time is marginal compared to the other strategies, it is not reported.

Table 1 to table 3 report the CPU time of several strategies. Those strategies are:

C : the arc-node model ($KMFP_2$) is solved using CPLEX 8.0

BB : the arc-node model ($KMFP_2$) is solved using Barnhart's branching in a home-made branch and bound

BP : the extended model ($KMFP_3$) is solved using Barnhart's branching in a home-made branch and price

BP-V : similar to BP, except that the global pool strategy is used

BP-VP : similar to BP-V, except that the variable ordering is used

BP-VP2 : similar to BP-VP, except that the alternate branching is used

In the last two columns, we give the root gap from both the arc-node ($KMFP_2$) model and from the extended ($KMFP_3$) model. Thus the root gap stands for the relative gap between the fractional solution and the integer solution of the problem.

We can note that the gap is a lot better for the extended model. This is closely related to the problem with the subcycles illustrated in the figure 1: by relaxing the binary variables, the fractional solution may have some cycles on the flow definition variables y_a^h, for fixed values of h. Those cycles will help artificialy improve the amount of flow that can be routed. The extended model relies on elementary paths. Its fractional solution could create cycles on the path definition variables y_p^h, for fixed values of h. However, as the flow variables x_p^h are on the path too, no artificial improvement can be done. As the extended model has a smaller root gap than the arc-node model, this will explain why the "BP*" strategies always dominate the "C" and "BB" approaches in term of CPU time.

instance		values		CPU time (s)						gap (%)	
graph	H	z^*	BKS	C	BB	BP	BP-P	BP-VP	BP-VP2	$KMFP_2$	$KMFP_3$
tg10-2	1	389.00	389.00	0.01	0.01	**0.00**	**0.00**	**0.00**	**0.00**	25.13	0.00
12-38	2	557.00	557.00	0.18	0.93	0.21	0.11	0.06	**0.02**	26.59	11.98
	3	716.00	557.00	1.44	104.74	9.65	4.27	0.21	**0.02**	12.15	6.18
	4	815.00	716.00	0.03	7.48	**0.02**	0.21	0.06	0.07	0.00	0.00
	∞	815.00									
tg10-3	1	189.00	189.00	0.02	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	52.13	0.00
12-38	2	350.00	350.00	0.51	0.25	**0.01**	**0.01**	**0.01**	0.02	39.66	3.41
	3	466.00	442.00	3.04	21.82	0.40	0.09	0.05	**0.04**	19.66	6.92
	4	558.00	558.00	70.97	586.99	2.98	0.69	0.12	**0.07**	3.79	1.95
	5	580.00	558.00	0.11	0.04	**0.00**	**0.00**	0.01	0.04	0.00	0.00
	∞	580.00									
tg10-9	1	501.00	501.00	0.03	0.01	**0.00**	**0.00**	**0.00**	**0.00**	32.58	0.00
12-38	2	935.00	935.00	0.23	0.11	**0.00**	**0.00**	**0.00**	**0.00**	32.77	0.00
	3	1173.00	1051.00	1.95	9.08	0.51	0.15	0.06	**0.05**	22.68	1.84
	4	1356.00	1289.00	13.09	930.27	21.99	11.51	**0.23**	3.46	10.61	4.09
	5	1460.00	1364.00	288.64	-	-	1419.06	11.29	**0.54**	-	2.78
	6	1517.00	1364.00	5.52	82.11	3.57	**0.09**	5.88	0.68	0.00	0.00
	∞	1517.00									
tg20-2	1	385.00	385.00	0.02	0.13	**0.00**	**0.00**	**0.00**	**0.00**	18.43	0.00
22-72	2	643.00	643.00	0.38	112.14	**0.01**	**0.01**	**0.01**	**0.01**	24.62	0.00
	3	832.00	643.00	1.94	-	0.05	**0.01**	**0.01**	0.05	-	0.00
	4	853.00	832.00	0.38	2.43	1.26	**0.01**	104.60	0.22	0.00	0.00
	∞	853.00									
tg40-1	1	517.00	517.00	1.16	0.05	**0.01**	**0.01**	**0.01**	**0.01**	21.31	0.00
42-152	2	750.00	520.00	788.84	-	0.04	0.02	**0.01**	0.04	-	0.06
	3	908.00	750.00	-	-	2066.70	-	303.87	**20.18**	-	2.59
	4	994.00	918.00	-	-	-	-	-	**90.29**	-	-
	5	1004.00	918.00	28.09	-	0.17	**0.01**	183.20	-	-	0.00
	∞	1004.00									
tg40-5	1	487.00	487.00	0.23	0.07	**0.00**	0.01	0.01	0.01	22.70	0.00
42-152	2	828.00	828.00	2.87	-	23.29	17.87	4.19	**0.14**	-	4.94
	3	1062.00	828.00	-	-	-	-	137.41	**0.36**	-	0.28
	4	1078.00	1062.00	11.88	271.55	**0.02**	**0.02**	100.26	0.98	0.00	0.00
	∞	1078.00									
tg40-8	1	454.00	454.00	0.31	0.52	0.01	0.01	**0.00**	0.01	27.93	0.00
42-152	2	775.00	705.00	16.42	-	0.05	**0.01**	0.02	0.03	-	0.00
	3	991.00	858.00	-	-	1138.19	2148.15	332.33	**0.85**	-	2.56
	4	1085.00	1067.00	32.26	-	-	1739.14	**17.72**	-	-	0.00
	∞	1085.00									
tg40-10	1	142.00	142.00	1.71	26.32	**0.00**	**0.00**	**0.00**	0.01	72.28	0.00
42-152	2	278.00	278.00	-	-	**0.01**	**0.01**	**0.01**	0.02	-	0.00
	3	410.00	410.00	-	-	**0.01**	**0.01**	**0.01**	0.02	-	0.00
	4	509.00	509.00	-	-	**0.01**	**0.01**	**0.01**	0.03	-	0.00
	5	602.00	553.00	-	-	0.14	**0.01**	0.88	0.44	-	0.00
	6	691.00	642.00	-	-	0.06	**0.01**	1.43	0.91	-	0.00
	7	769.00	720.00	-	-	0.17	**0.01**	0.63	2.07	-	0.00
	8	804.00	769.00	22.75	-	0.29	**0.09**	-	9.21	-	0.00
	∞	804.00									

Table 1: CPU times for small transid grids.

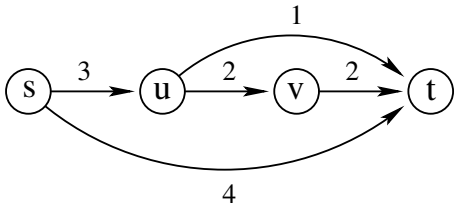
instance		values		CPU time (s)						gap (%)	
graph	H	z^*	BKS	C	BB	BP	BP-P	BP-VP	BP-VP2	$KMFP_2$	$KMFP_3$
tg50-2	1	336.00	336.00	12.39	0.56	0.01	**0.00**	0.01	0.01	63.77	0.00
52-198	2	652.00	652.00	-	-	1.78	0.26	0.36	**0.20**	-	1.69
	3	900.00	900.00	-	-	-	-	641.34	**4.52**	-	6.44
	4	1147.00	1134.00	-	-	-	-	818.97	**28.70**	-	4.09
	5	1342.00	1329.00	-	-	-	-	-	**138.51**	-	-
	6	$\geqslant$1394.00	1329.00	-	-	-	-	-	-	-	-
	∞	1719.00									
tg50-5	1	562.00	562.00	91.08	0.20	**0.00**	0.01	0.01	0.01	33.84	0.00
52-198	2	965.00	902.00	-	-	0.09	**0.02**	0.04	0.26	-	2.08
	3	1343.00	1087.00	-	-	1.23	0.21	**0.10**	0.80	-	1.85
	4	1596.00	1165.00	-	-	-	-	**83.80**	-	-	6.53
	5	$\geqslant$1781.00	1480.00	-	-	-	-	-	-	-	-
	∞	2507.00									
tg50-10	1	399.00	399.00	0.07	0.01	**0.00**	**0.00**	**0.00**	0.01	28.88	0.00
52-198	2	734.00	734.00	1.50	-	**0.01**	**0.01**	**0.01**	0.02	-	0.00
	3	899.00	734.00	219.69	-	**0.01**	**0.01**	**0.01**	0.09	-	0.00
	4	1031.00	734.00	-	-	**0.04**	**0.04**	0.13	0.18	-	0.00
	5	1153.00	899.00	-	-	0.45	**0.08**	0.18	1.51	-	0.00
	6	1270.00	899.00	-	-	1.34	**0.09**	0.62	3.62	-	0.00
	7	1376.00	1031.00	-	-	0.85	**0.06**	19.86	9.57	-	0.00
	8	1403.00	1153.00	-	-	4.57	**0.61**	452.23	35.66	-	0.00
	9	1430.00	1267.50	63.00	-	18.55	**0.54**	1922.68	23.48	-	0.00
	∞	1430.00									
tg60-3	1	776.00	776.00	0.24	0.03	**0.00**	0.01	0.01	0.01	4.53	0.00
62-242	2	1168.00	986.00	-	-	0.24	0.13	**0.06**	0.43	-	2.86
	3	1480.00	1216.00	-	-	584.87	106.12	**1.49**	3.68	-	3.06
	4	$\geqslant$1681.00	1528.00	-	-	-	-	-	-	-	-
	∞	2739.00									
tg60-6	1	347.00	347.00	-	1.81	0.01	0.01	**0.00**	0.01	52.55	0.00
62-242	2	676.00	676.00	-	-	**0.01**	**0.01**	**0.01**	0.02	-	0.00
	3	983.00	983.00	-	-	**0.01**	**0.01**	**0.01**	0.04	-	0.00
	4	1251.00	1208.00	-	-	40.00	1.66	2.34	**1.59**	-	0.00
	5	1510.00	1476.00	-	-	-	-	**15.10**	103.28	-	0.00
	6	$\geqslant$1512.00	1476.00	-	-	-	-	-	-	-	-
	∞	2070.00									
tg70-8	1	515.00	515.00	4.64	3.55	**0.01**	**0.01**	**0.01**	**0.01**	6.68	0.00
72-268	2	928.00	928.00	4.87	-	**0.01**	**0.01**	0.02	0.02	-	0.00
	3	1144.00	928.00	-	-	96.04	85.65	1.46	**0.51**	-	0.13
	4	1147.00	1144.00	51.03	-	1.94	**0.07**	-	-	-	0.00
	∞	1147.00									
tg80-1	1	549.00	549.00	-	3.47	**0.01**	**0.01**	**0.01**	**0.01**	27.31	0.00
82-322	2	984.00	984.00	-	-	56.39	12.00	7.79	**0.87**	-	5.02
	3	1411.00	1321.00	-	-	-	-	451.69	**1.54**	-	3.85
	4	$\geqslant$1589.00	1589.00	-	-	-	-	-	-	-	-
	∞	2797.00									
tg80-6	1	474.00	474.00	-	9.49	**0.01**	**0.01**	**0.01**	**0.01**	50.76	0.00
82-322	2	833.00	833.00	-	-	1.00	0.23	0.13	**0.10**	-	0.45
	3	1160.00	1139.00	-	-	167.95	332.63	41.21	**18.35**	-	1.77
	4	1429.00	1235.00	-	-	-	-	2474.14	**173.52**	-	2.98
	5	$\geqslant$1656.00	1480.00	-	-	-	-	-	-	-	-
	∞	2445.00									
tg100-2	1	530.00	530.00	-	7.27	**0.01**	**0.01**	**0.01**	**0.01**	42.76	0.00
102-400	2	1007.00	1007.00	-	-	0.02	**0.01**	0.02	0.02	-	0.00
	3	1407.00	1336.00	-	-	76.85	20.41	5.98	**0.46**	-	0.14
	4	1768.00	1664.00	-	-	-	-	**56.22**	1951.96	-	0.32
	5	$\geqslant$1711.00	1711.00	-	-	-	-	-	-	-	-
	∞	3519.00									
tg100-9	1	424.00	424.00	-	680.71	**0.01**	**0.01**	**0.01**	**0.01**	49.73	0.00
102-400	2	845.00	845.00	-	-	**0.02**	**0.02**	**0.02**	**0.02**	-	0.00
	3	1234.00	1199.00	-	-	-	2060.47	600.63	**0.55**	-	0.10
	4	1600.00	1570.00	-	-	-	-	-	**26.41**	-	-
	5	$\geqslant$1905.00	1905.00	-	-	-	-	-	-	-	-
	∞	3271.00									

Table 2: CPU times for medium transid grids.

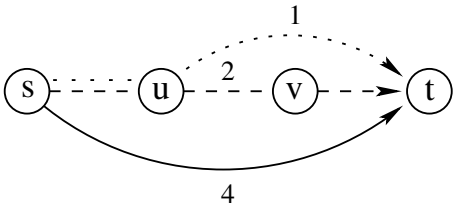
instance		values		CPU time (s)						gap (%)	
graph	H	z^*	BKS	C	BB	BP	BP-P	BP-VP	BP-VP2	$KMFP_2$	$KMFP_3$
random5-35	1	66.00	66.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	10.09	0.00
5-35	2	128.00	128.00	0.02	0.02	0.01	**0.00**	**0.00**	**0.00**	11.87	0.00
	3	182.00	182.00	0.02	0.54	**0.00**	**0.00**	**0.00**	**0.00**	9.08	0.00
	4	223.00	204.00	0.07	19.04	0.07	**0.02**	0.03	0.04	7.19	0.00
	5	262.00	243.00	0.05	221.62	0.09	**0.01**	0.05	0.08	5.59	0.00
	6	297.00	278.00	0.19	1186.40	0.22	**0.05**	0.09	0.12	4.15	0.00
	7	326.00	297.00	0.41	0.06	0.19	**0.03**	0.12	0.26	0.00	0.00
	∞	326.00									
random10-45	1	73.00	73.00	0.01	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	2.25	0.00
10-45	2	142.00	142.00	0.08	0.05	**0.00**	**0.00**	**0.00**	**0.00**	4.05	0.58
	3	209.00	209.00	0.53	0.52	0.03	**0.01**	**0.01**	0.04	5.43	0.57
	4	260.00	260.00	1.39	12.53	0.51	**0.09**	0.12	1.63	6.87	1.58
	5	306.00	306.00	7.26	147.83	6.87	1.04	**0.28**	2.85	7.42	1.83
	6	345.00	321.00	118.78	-	202.89	29.01	**0.93**	13.79	-	2.36
	7	381.00	360.00	1285.61	-	-	-	**4.81**	59.60	-	2.54
	8	413.00	368.00	-	-	-	-	**17.59**	379.34	-	2.93
	9	429.00	381.00	-	-	-	-	**2554.28**	-	-	6.21
	10	$\geqslant$417.00	417.00	-	-	-	-	-	-	-	-
	∞	498.00									
random15-60	1	86.00	86.00	0.01	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.00	0.00
15-60	2	163.00	163.00	0.08	0.02	**0.00**	**0.00**	**0.00**	0.01	0.61	0.00
	3	221.00	221.00	0.24	0.47	**0.00**	**0.00**	**0.00**	0.01	3.37	0.44
	4	248.00	229.00	1.97	28.51	5.43	1.26	**0.12**	0.76	7.62	2.53
	5	268.00	229.00	12.91	-	87.58	20.94	**0.92**	3.18	-	2.86
	6	287.00	229.00	72.99	-	-	2616.90	**2.07**	5.90	-	2.74
	7	295.00	229.00	1960.18	-	-	-	**46.63**	2080.02	-	3.97
	8	$\geqslant$301.00	256.00	-	-	-	-	-	-	-	-
	∞	310.00									
random20-140	1	81.00	81.00	0.01	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.46	0.00
20-140	2	158.00	158.00	0.02	0.02	**0.00**	**0.00**	**0.00**	**0.00**	0.30	0.00
	3	228.00	228.00	0.30	0.03	**0.00**	**0.00**	0.01	0.01	0.38	0.00
	4	253.00	235.00	14.90	-	91.92	323.10	7.48	**1.52**	-	1.81
	5	274.00	235.00	230.60	-	-	-	-	**4.80**	-	-
	6	294.00	236.00	-	-	-	-	-	**5.81**	-	-
	7	311.00	236.00	-	-	-	-	-	**19.25**	-	-
	8	319.00	236.00	-	-	-	-	-	**2042.98**	-	-
	9	$\geqslant$325.00	261.00	-	-	-	-	-	-	-	-
	10	327.00	261.00	725.65	-	49.66	**6.16**	-	-	-	0.00
	∞	327.00									

Table 3: CPU times for random digraphs.

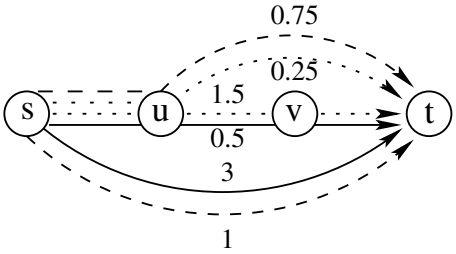
As a general point of view, the CPU time is decreasing when the strategy is improving (branch and price, adding the global pool, adding the variable ordering and using the alternate branching). However, on some instances, the strategies using the variable ordering seem to take a lot of time. This is especialy true when the width limit H is set to the maximal flow width (e.g. the width constraint becomes redundant). This may be explained by the dual variables ν^h associated to the variable ordering constraints. By the complementary slackness theorem, using the ν^h variables in the dual solution implies the associated variable ordering constraints are saturated. This means two successive paths positions will route the same amount of flow. In order to have the same amount of flow, the fractional primal solution might have to use several paths for those path positions. Then, the fractional solution when using variable ordering constraints is more likely to be more "splitted" than without those contraints. This phenomenon is illustrated in figure 7. The fractional solution 7(b) without VO constraints is using 3 path positions, routing respectively 4, 2 and 1 unit of flow on a single path. When using VO contraints, the fractional solution 7(c) still uses 3 positions, routing respectively 4.5, 1.75 and 1.75 units of flow. However, each position is now using 2 paths



(a) capacitated digraph



(b) without VO constraints, $H = 3$



(c) with VO constraints, $H = 3$

Figure 7: impact of the VO constraints on the fractional solution.

Tables 4 to 6 will help us further analyse the problem. For each instance, we report the rate of nodes in the branch and bound tree whose fractional solution saturates at least one VO contraint. This rate is a lot higher when using the VO contraints.

instances		values	VO saturation (%)		
graph	H	z^*	BP-P	BP-VP	BP-VP2
tg10-2	1	389.00	0.00	0.00	0.00
12-40	2	557.00	0.29	23.15	41.18
	3	716.00	0.04	65.53	71.43
	4	815.00	1.68	84.85	75.00
	∞	815.00			
tg10-3	1	189.00	0.00	0.00	0.00
12-40	2	350.00	0.00	58.82	40.00
	3	466.00	0.00	67.23	61.76
	4	558.00	0.00	77.25	84.44
	5	580.00	0.00	92.31	95.83
	∞	580.00			
tg10-9	1	501.00	0.00	0.00	0.00
12-40	2	935.00	0.00	0.00	100.00
	3	1173.00	0.76	71.79	74.19
	4	1356.00	0.30	74.60	81.08
	5	1460.00	1.09	89.79	94.63
	6	1517.00	1.00	99.98	96.23
	∞	1517.00			
tg20-2	1	385.00	0.00	0.00	0.00
22-80	2	643.00	0.00	0.00	100.00
	3	832.00	0.00	0.00	54.55
	4	853.00	0.00	79.04	86.49
	∞	853.00			
tg40-1	1	517.00	0.00	0.00	0.00
42-160	2	750.00	0.00	50.00	22.22
	3	908.00	0.12	10.91	28.37
	4	994.00	0.61	78.07	39.55
	5	1004.00	0.00	99.23	65.01
	∞	1004.00			
tg40-5	1	487.00	0.00	0.00	0.00
42-160	2	828.00	0.30	20.54	45.16
	3	1062.00	0.06	88.34	59.62
	4	1078.00	0.00	99.68	70.75
	∞	1078.00			
tg40-8	1	454.00	0.00	0.00	0.00
42-160	2	775.00	0.00	0.00	25.00
	3	991.00	0.28	27.36	34.67
	4	1085.00	0.43	96.09	75.41
	∞	1085.00			
tg40-10	1	142.00	0.00	0.00	0.00
42-160	2	278.00	0.00	0.00	100.00
	3	410.00	0.00	100.00	0.00
	4	509.00	0.00	100.00	100.00
	5	602.00	0.00	98.82	94.87
	6	691.00	0.00	99.63	94.29
	7	769.00	0.00	99.44	99.07
	8	804.00	3.03	100.00	83.60
	∞	804.00			

Table 4: saturation on VO constraints for small transid grids.

instances		values	VO saturation (%)		
graph	H	z^*	BP-P	BP-VP	BP-VP2
tg50-2	1	336.00	0.00	0.00	0.00
52-200	2	652.00	3.97	42.18	66.67
	3	900.00	1.81	81.08	69.54
	4	1147.00	0.47	90.73	86.72
	5	1342.00	1.71	97.17	87.22
	6	$\geqslant$1394.00	1.20	97.75	91.94
	∞	1719.00			
tg50-5	1	562.00	0.00	0.00	0.00
52-200	2	965.00	0.00	34.48	40.00
	3	1343.00	4.20	60.00	64.00
	4	1596.00	4.15	74.54	67.20
	5	$\geqslant$1781.00	1.14	94.92	74.61
	∞	2507.00			
tg50-10	1	399.00	0.00	0.00	0.00
52-200	2	734.00	0.00	0.00	100.00
	3	899.00	0.00	0.00	87.50
	4	1031.00	12.50	89.86	81.25
	5	1153.00	0.00	75.00	48.75
	6	1270.00	7.89	84.79	26.06
	7	1376.00	21.05	80.54	33.90
	8	1403.00	0.94	99.81	51.40
	9	1430.00	21.73	98.53	98.22
	∞	1430.00			
tg60-3	1	776.00	0.00	0.00	0.00
62-240	2	1168.00	0.00	26.09	40.48
	3	1480.00	0.06	31.54	48.05
	4	$\geqslant$1681.00	0.21	65.70	3.89
	∞	2739.00			
tg60-6	1	347.00	0.00	0.00	0.00
62-240	2	676.00	0.00	0.00	100.00
	3	983.00	0.00	0.00	100.00
	4	1251.00	3.48	84.70	36.90
	5	1510.00	3.92	57.07	17.75
	6	$\geqslant$1512.00	5.08	96.67	15.81
	∞	2070.00			
tg70-8	1	515.00	0.00	0.00	0.00
72-280	2	928.00	0.00	0.00	100.00
	3	1144.00	0.00	6.53	16.00
	4	1147.00	0.00	91.70	53.60
	∞	1147.00			
tg80-1	1	549.00	0.00	0.00	0.00
82-320	2	984.00	0.00	18.94	46.24
	3	1411.00	0.24	75.57	46.43
	4	$\geqslant$1589.00	1.99	93.54	91.87
	∞	2797.00			
tg80-6	1	474.00	0.00	0.00	0.00
82-320	2	833.00	0.00	19.30	40.00
	3	1160.00	0.52	33.12	18.95
	4	1429.00	1.46	81.85	46.63
	5	$\geqslant$1656.00	4.40	84.30	84.65
	∞	2445.00			
tg100-2	1	530.00	0.00	0.00	0.00
102-400	2	1007.00	0.00	0.00	100.00
	3	1407.00	0.03	27.02	52.38
	4	1768.00	0.58	38.91	20.22
	5	$\geqslant$1711.00	2.95	90.15	74.36
	∞	3519.00			
tg100-9	1	424.00	0.00	0.00	0.00
102-400	2	845.00	0.00	0.00	100.00
	3	1234.00	2.95	44.71	62.86
	4	1600.00	2.11	80.51	75.45
	5	$\geqslant$1905.00	7.60	95.20	56.36
	∞	3271.00			

Table 5: saturation on VO constraints for medium transid grids.

instances		values	VO saturation (%)		
graph	H	z^*	BP-P	BP-VP	BP-VP2
random5-35	1	66.00	0.00	0.00	0.00
5-35	2	128.00	0.00	0.00	100.00
	3	182.00	0.00	100.00	0.00
	4	223.00	18.52	82.98	87.50
	5	262.00	0.00	94.03	90.38
	6	297.00	14.10	93.14	90.48
	7	326.00	17.14	98.36	94.21
	∞	326.00			
random10-45	1	73.00	0.00	0.00	0.00
10-45	2	142.00	0.00	100.00	0.00
	3	209.00	34.48	88.24	92.31
	4	260.00	12.56	94.58	94.59
	5	306.00	12.62	98.11	98.22
	6	345.00	10.05	98.37	98.23
	7	381.00	9.45	99.57	98.63
	8	413.00	9.91	99.48	99.29
	9	429.00	13.79	99.78	98.98
	10	$\geqslant$417.00			99.85
	∞	498.00			
random15-60	1	86.00	0.00	0.00	0.00
15-60	2	163.00	0.00	0.00	100.00
	3	221.00	0.00	100.00	0.00
	4	248.00	0.95	67.27	89.11
	5	268.00	2.33	91.93	95.74
	6	287.00	4.14	99.38	98.87
	7	295.00	6.25	97.75	90.46
	8	$\geqslant$301.00	7.66	99.26	97.39
	9	$\geqslant$306.00	11.43	99.98	98.43
	10	310.00	13.04	100.00	99.99
	∞	310.00			
random20-140	1	81.00	0.00	0.00	0.00
20-140	2	158.00	0.00	0.00	100.00
	3	228.00	0.00	100.00	100.00
	4	253.00	2.10	69.25	83.81
	5	274.00	2.47	73.09	94.70
	6	294.00	25.74	99.96	95.67
	7	311.00	12.14	99.99	98.34
	8	319.00	13.57	97.92	96.49
	9	$\geqslant$319.00	13.54	99.58	99.79
	10	327.00	19.70	100.00	99.98
	∞	327.00			

Table 6: saturation on VO constraints for random digraphs.

5 Conclusion

The extended model (KMFP_3) has a clear advantage over the arc-node model (KMFP_2) as the root gap is much lower. Thus the branch and price may seem to be the best way to solve the problem. From our experiment, we were able to solve medium-sized instances for a limited width H. As H increases, the problem becomes harder to solve, until the width of the unrestricted maximum flow is reached. From a practical point of view, this is not really an issue as Internet Providers wish to work with small values – through the use of MPLS-TE for instance. Nevertheless, from a theoretical point of view, we believe our branch and price is limited by two factors: first, a lot of perturbations are introduced into the fractional solutions by the addition of VO constraints. As it was shown, the fractional solutions are using more paths for each path position. Therefore, more branchings need to be done to reach the integer solution. An alternate VO strategy that does not lead to such situation would really help the branch and price. Second, the two branching strategies shown in this work are lacking some efficiency. It would be interesting to find a stronger branching scheme, and maybe add some cuts in the LR_4 model as in the RBCP strategy. We are currently

working on both issues.

References

[1] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows - Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[2] Filipe Alvelos and José Manuel Vasconcelos Valério de Carvalho. Comparing Branch-and-price Algorithms for the Unsplittable Multicommodity Flow Problem. In *Proceedings of the International Network Optimization Conference*, pages 7–12, 2003.

[3] Alper Atamtürk and Deepak Rajan. On splittable and unsplittable capacitated network design arc-set polyhedra. *Mathematical Programming*, 92:315–333, 2002.

[4] Georg Baier, Ekkehard Köhler, and Martin Skutella. On the k-splittable flow problem. In *Proceedings of the 10th Annual European Symposium on Algorithms*, pages 101–113. Springer-Verlag, 2002.

[5] Cynthia Barnhart, Christopher A. Hane, and Pamela H. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2):318–326, 2000.

[6] Cynthia Barnhart, Ellis L. Johnson, George L. Nemhauser, Martin W.P. Savelsbergh, and Pamela H. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.

[7] James E. Burns, Teunis J. Ott, Johan M. de Kock, and Anthony E. Krzesinski. Path Selection and Bandwidth Allocation in MPLS Networks: a Non-Linear Programming Approach. In *ITCom Conference*, 2001.

[8] Robert J. Dakin. A tree-search algorithm for mixed integer programming problems. *The Computer Journal*, 8(3):250–254, 1965.

[9] George B. Dantzig and Philip Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.

[10] Lester R. Ford and Delbert R. Fulkerson. A suggested computation for maximal multicommodity network flow. *Management Science*, 5:97–101, 1958.

[11] R. Fukasawa, M. Poggi de Aragao, O. Porto, and E. Uchoa. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. In *INOC Conference*, pages 231–236, 2003.

[12] Jon M. Kleinberg. *Approximation algorithms for disjoint paths problems*. PhD thesis, Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 1996.

[13] Ronald Koch, Martin Skutella, and Ines Spenke. Approximation and complexity of k-splittable flows. In *WAOA*, pages 244–257, 2005.

[14] Stavros Kolliopoulos. Minimum-cost single-source 2-splittable flow. *Information Processing Letters*, 94:15–18, 2005.

[15] A. Land and A. Doig. An automatic method of solving discrete programming problems. *Econometrika*, 28(3):497–520, 1960.

[16] Marco E. Lübbecke and Jacques Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[17] Maren Martens and Martin Skutella. Flows on Few Paths: Algorithms and Lower Bounds. In *proceedings of ESA 2004*, pages 520–531, 2004.

[18] Maren Martens and Martin Skutella. Length-Bounded and Dynamic k-Splittable Flows. In *proceedings of the International Scientific Annual Conference Operations Research*, 2005.

[19] Ernesto Martins. The optimal path problem. *Investigação Operacional*, 19:43–60, 1999.

[20] Vahab S. Mirrokni, Marina Thottan, Huseyin Uzunalioglu, and Sanjoy Paul. A Simple Polynomial Time Framework To Reduced Path Decomposition in Multi-Path Routing. In *proceedings of INFOCOM 2004*, 2004.

[21] David M. Ryan and Brian A. Foster. An integer programming approach to scheduling. In Anthony Wren, editor, *Computer Scheduling of Public Transport : Urban Passenger Vehicle and Crew Scheduling*, pages 269–280. North-Holland Publishing Company, 1981.

[22] Hanif D. Sherali and J. Cole Smith. Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407, 2001.

[23] François Vanderbeck and Laurence A. Wolsey. An exact algorithm for ip column generation. *Operations Research Letters*, 19(4):151–159, 1996.

[24] Bénédicte Vatinlen. *Optimisation du routage dans les réseaux de télécommunication avec prise en compte de la qualité de service*. PhD thesis, University Paris VI, 2004.