Two- and Three-index formulations of the Minimum Cost Multicommodity *k*-splittable Flow Problem

Mette Gamst   Peter Neergaard Jensen   David Pisinger   Christian Plum

Department of Computer Science, University of Copenhagen
Universitetsparken 1, 2100 Copenhagen Ø, Denmark
{gamst, petgaard, pisinger, plum}@diku.dk

Abstract

The Multicommodity Flow Problem (MCFP) considers the efficient routing of commodities from their origins to their destinations, subject to capacity restrictions and edge costs. This paper studies the $\mathcal{NP}$-hard Minimum Cost Multicommodity k-splittable Flow Problem in which each commodity may use at most *k* paths between its origin and its destination. The problem has applications in transportation problems where a number of commodities must be routed, using a limited number of transportation units at each destination. Based on a three-index formulation from (Truffot et al., 2005) we present a new two-index formulation for the problem, and solve both formulations through branch-and-price. The three-index algorithm by Truffot et al. is improved by introducing a heuristic method to reach a feasible solution by eliminating some symmetry. A novel branching strategy for the two-index formulation is presented, forbidding subpaths in the branching children. The proposed heuristic for the three-index algorithm improves the performance, but the three-index algorithm is still outperformed by the two-index algorithm, both with respect to running time and to the number of solved test instances.

Keywords*: mixed integer programming, k-splittable, Multicommodity Flow, Branch-and-Price*

1 Introduction

We consider the $\mathcal{NP}$-hard Minimum Cost, k-splittable variant of the Multicommodity Flow Problem (MCFP). A MCFP consists of a network with capaciteted edges, and of a set of commodities, and the goal is to either minimize the total cost of sending all flow of the commodities, or to maximize the total amount of flow sent for all the commodities. The MCFP can be formulated as a linear programming problem and is thus polynomial [1]. Often, however, extra conditions must be satisfied, making the problem $\mathcal{NP}$-hard. An example of such a condition is, that all flow for each commodity must be sent via just one path. This problem is denoted the *Unsplittable* MCFP, and was introduced and proved $\mathcal{NP}$-hard by Kleinberg [5]. Yet another practically relevant condition is an upper bound on the number of paths used by a commodity. This is called the Multicommodity *k*-splittable Flow Problem (MCkFP). We consider the Minimum Cost MCkFP, which for instance is relevant in the transportation sector or in telecommunication context.

Barnhart et al. [4] considered the Minimum Cost Unsplittable MCFP. They presented a branch-and-price-and-cut algorithm with a new branching rule allowing new columns to be generated effectively. The Multicommodity k-splittable Flow Problem (MCkFP) was introduced and proved to be $\mathcal{NP}$-hard by Baier et al. [3], who also presented approximation algorithms for the Single- and Multicommodity k-splittable Flow Problems. Truffot et al. [8, 10] used branch-and-price to solve the Maximum MCkFP. The pricing problem is a shortest path problem solvable in polynomial time. Truffot et al. [9] also introduced the Minimum Cost MCkFP. A three-index model for the problem was solved using a branch-and-price algorithm. The algorithm is closely related to the one presented in [10].

The Minimum Cost MCkFP is represented by a directed graph, $G = (V, E)$, where V is the set of vertices and E the set of edges. An edge $e \in E$ has weight $c_e \geq 0$, and capacity $u_e > 0$. The set of commodities is denoted L. Commodity $l \in L$ has source s_l and destination t_l, an amount to be shipped F^l and an upper bound on the number of used routes k^l.

The main contribution of this paper is to compare various formulations of the *Minimum Cost* MCkFP when solved through branch-and-price. Truffot et al. [8] introduced both a two-index and a three-index formulation, but discarded the two-index formulation due to complications in the branching strategy. We present and compare branch-and-price algorithms for both formulations, add a heuristic for the three index-model, and introduce a new branching strategy for the two-index model.

2 Three-index model

Let P^l be the set of possible paths for commodity l. The variable x_p^{hl} denotes the amount of flow on path p for the h'th path of commodity l. The binary variable y_p^{hl} decides whether path p for the h'th path of commodity l is to be used or not. The model is:

$$
\text{(MIP1)} \quad \begin{cases}
\min \quad \sum_{l \in L} \sum_{h=1}^{k^l} \sum_{p \in P^l} c_p x_p^{hl} & \\
\text{s.t.} \quad \sum_{l \in L} \sum_{h=1}^{k^l} \sum_{p \in P^l} \delta_e^p x_p^{hl} \leq u_e & \forall e \in E \quad (1) \\
x_p^{hl} - u_p y_p^{hl} \leq 0 & \forall l \in L, h = 1, \ldots, k^l, \forall p \in P^l \quad (2) \\
\sum_{p \in P^l} y_p^{hl} \leq 1 & \forall l \in L, h = 1, \ldots, k^l \quad (3) \\
\sum_{h=1}^{k^l} \sum_{p \in P^l} x_p^{hl} = F^l & \forall l \in L \quad (4) \\
x_p^{hl} \geq 0 & \forall l \in L, h = 1, \ldots, k^l, \forall p \in P^l \\
y_p^{hl} \in \{0, 1\} & \forall l \in L, h = 1, \ldots, k^l, \forall p \in P^l
\end{cases}
$$

The objective function minimizes the total cost. Constraint (1) is a capacity constraint, in which δ_e^p indicates whether or not edge e is used by path p. In (2), u_p denotes the capacity constraint on path p, which is defined as $u_p = min\{u_e \mid e \in p\}$, hence (2) forces every decision variable, y_p^{hl}, to be set, if there is flow on the corresponding path, x_p^{hl}. Constraint (3) ensures, that at most one path is used as the h'th path of a commodity l, and finally (4) ensures that all commodities are shipped.

The model is relaxed into an LP-model: first the binary variables y_p^{hl} are LP-relaxed to $0 \leq y_p^{hl} \leq 1$. From (2) and (3) we have that: $x_p^{hl}/u_p \leq y_p^{hl} \leq 1$. We set y_p^{hl} to its lower bound, which leaves the following model:

$$
\text{(LP2)} \quad \begin{cases}
\min \quad \sum_{l \in L} \sum_{h=1}^{k^l} \sum_{p \in P^l} c_p x_p^{hl} & \\
\text{s.t.} \quad \sum_{l \in L} \sum_{h=1}^{k^l} \sum_{p \in P^l} \delta_e^p x_p^{hl} \leq u_e & \forall e \in E \quad (5) \\
\sum_{p \in P^l} \frac{x_p^{hl}}{u_p} \leq 1 & \forall l \in L, h = 1, \ldots, k^l \quad (6) \\
\sum_{h=1}^{k^l} \sum_{p \in P^l} x_p^{hl} = F^l & \forall l \in L \quad (7) \\
x_p^{hl} \geq 0 & \forall l \in L, h = 1, \ldots, k^l, \forall p \in P^l
\end{cases}
$$

Model *(LP2)*, causes symmetry in the solution space, as the h-index may result in equivalent solutions being treated as different solutions. To eliminate some of this symmetry, the following *variable ordering* constraint is added to

(MIP1):

$$\sum_{p\in P^l} x_p^{(h+1)l} - \sum_{p\in P^l} x_p^{hl} \leq 0, \quad \forall l \in L, h = 1,\dots,k^l - 1 \tag{8}$$

The reduced cost is calculated. Let $\pi_e \leq 0$ be the dual of (5), $\lambda^{hl} \leq 0$ the dual of (6), $\sigma^l \in \mathbb{R}$ the dual of (7) and $\omega^{hl} \leq 0$ the dual of (8). Furthermore let $\bar{\omega}^{hl} = \omega^{hl}, h = 1$, $\bar{\omega}^{hl} = \omega^{hl} - \omega^{(h-1)l}, h = 2, \dots k^l - 1$ and $\bar{\omega}^{hl} = -\omega^{(h-1)l}, h = k^l$. Even though the primal model only consists of one variable type, the dual formulation has three constraints because of the symmetry constraint (8). The reduced costs are:

$$\sum_{e\in E} \delta_e^p (c_e - \pi_e) - \frac{\lambda^{hl}}{u_p} + \sigma^l + \bar{\omega}^l \quad \forall l \in L, h = 1,\dots,k^l, \forall p \in P^l \tag{9}$$

For each pair of values (h, l) the task is to find a path $p \in P^l$ which has negative reduced cost in (9). If the value for u_p is known in advance, then the problem can be recognized as a shortest path problem defined in costs $(c_e - \pi_e) \geq 0$, which can be solved in polynomial time using e.g. Dijkstra's algorithm [1]. The path capacity u_p can take on at most $|E|$ different values; for each of the $|E|$ values of u_p the shortest path problem is solved on a graph, where edges with $u_e < u_p$ are removed.

The chosen branching scheme seeks to reach a solution, where at most k^l paths is used for each commodity l. Let *the first divergence node* of a commodity be defined as the node, to which all flow of the h'th path of the commodity is following the same route, and from which the flow is using two or more routes. For the h'th path of commodity l, the strategy is based on dividing all edges going out from the first divergence node, into two subsets. The two resulting subsets of outgoing edges are disjoint and balanced. Now, the branching strategy generates two branching children, in which each of the subsets of outgoing edges is forbidden. In this manner, the branching strategy eventually ensures, that at most one route is used for the h'th path of the commodity.

To decrease the running time of the branch-and-price algorithm, we suggest a simple heuristic method to reach a feasible solution by eliminating some symmetry in the relaxed LP-model. As x_p^{hl}/u_p will not always be binary, the relaxed model may allow several paths to be used as the h'th path of commodity l, and identical paths may take on different h values for a commodity l. To reach a feasible solution faster, the heuristic seeks to eliminate these issues by performing the following steps: **1:** For a commodity, several identical paths have different values of h. The paths are merged and assigned a single value of h. **2:** More than one path is used for a single value of h for a commodity. Each path is assigned a unique value of h, if possible.

3 Two-index model

A two-index formulation of the Minimum Cost MCkFP, without the use of h-indices is presented:

$$\textit{(MIP3)} \quad \begin{cases} \min & \sum_{l\in L}\sum_{p\in P^l} c_p x_p^l \\ \text{s.t.} & \sum_{l\in L}\sum_{p\in P^l} \delta_e^p x_p^l \leq u_e \quad \forall e \in E & (10) \\ & x_p^l - u_p y_p^l \leq 0 \quad \forall l \in L, \forall p \in P^l & (11) \\ & \sum_{p\in P^l} y_p^l \leq k^l \quad \forall l \in L & (12) \\ & \sum_{p\in P^l} x_p^l = F^l \quad \forall l \in L & (13) \\ & x_p^l \geq 0 \quad \forall l \in L, \forall p \in P^l \\ & y_p^l \in \{0,1\} \quad \forall l \in L, \forall p \in P^l \end{cases}$$

Here x_p^l is the total flow of commodity l on path p, and the corresponding variable y_p^l is set, if and only if, commodity l has flow on path p. The remaining variables have the same meaning as in the three-index model. The model is similar to *(MIP1)*, only constraint (12) has been added to limit the number of used paths for commodity l

to k^l. The problem is relaxed in the same manner as the three-index model, i.e. we replace y_p^l with x_p^l/u_p getting:

$$
(LP4) \quad \begin{cases}
\min & \sum_{l \in L} \sum_{p \in P^l} c_p x_p^l \\
\text{s.t.} & \sum_{l \in L} \sum_{p \in P^l} \delta_e^p x_p^l \leq u_e \quad \forall e \in E & (14) \\
& \sum_{p \in P^l} \frac{x_p^l}{u_p} \leq k^l \quad \forall l \in L & (15) \\
& \sum_{p \in P^l} x_p^l = F^l \quad \forall l \in L & (16) \\
& x_p^l \geq 0 \quad \forall l \in L, \forall p \in P^l
\end{cases}
$$

Let π_e, λ^l and σ^l be the dual variables for equations (14), (15) and (16) in *(LP4)*. The reduced cost for a path $p \in P^l$ for a commodity $l \in L$ is given by:

$$\sum_{e \in E} \delta_e^p (c_e - \pi_e) - \frac{\lambda^l}{u_p} + \sigma^l \tag{17}$$

The reduced cost is similar to that for the three-index model, hence the pricing problem is solved using Dijkstra's algorithm for the shortest path problem for each pair of values (h, l) and for each of the at most $|E|$ values of u_p.

The branching strategy for the three-index model cannot be reused, because the h-indices are omitted. A new strategy is developed, which considers the paths emanating from the first divergence node for each commodity. If the number of emanating paths is greater than k^l, then branching is necessary. The number of edges with positive flow going out of the divergence node may be smaller than the number of paths emanating from the node. Thus, it does not suffice to forbid the use of an edge. Rather, the branching strategy must forbid the use of a subpath. For each emanating path, the strategy finds the smallest sequence of edges, which makes the path unique. That is, the strategy seeks to minimize the size of the forbidden subpath. The number of branching children is $k^l + 1$, in which forbidden edge sequences are evenly distributed such that each branching child contains at least one forbidden edge sequence. No feasible solution is omitted from the combined solution space of the branching children. A feasible solution can use at most k^l of the subpaths we consider in a branch and each of these is forbidden in exactly one of the $k^l + 1$ branching children. Any valid solution will therefore be valid in at least one of these branching children, where its k^l used subpaths are forbidden in the remaining k^l branching children. The solution space of the branching children is not necessarily disjoint, which may result in degeneracy problems, since a solution can exist in several branching children, which must thus be explored.

The branching strategy necessitates some changes to the pricing problem. When solving the shortest path problem, we need to ensure that we do not use the forbidden edge sequences. The shortest path problem with forbidden paths is a polynomial problem and can be solved using a modified k-shortest path algorithm [11].

4 Computational Results

The described branch-and-price algorithms for the two models are tested on an Intel Pentium 4, 3.00 GHz machine with 2 GB RAM. The algorithms have been implemented using the framework `COIN` [7] with ILOG CPLEX 9.1 as LP-solver. We have through preliminary results decided to use strong branching, where all possible branching candidates are generated. A best-first search strategy is used in the branch-and-bound tree. Computations regarding selection of branching candidate and branching child are handled by `COIN`. The number of paths priced in per iteration to $0.5 \cdot |L| \cdot k$ for the three-index algorithm and to $0.5 \cdot |L|$ for the two-index algorithm.

The algorithms are tested on four types of problems: The randomly generated Carbin instances (`bl` and `bs`) [2], the `grid` instances formed as grids and `planar` instances simulating problems arising in telecommunication [6]. Three different values for k has been tested: $k = 2, 3$ and 10. When $k = 1$ the problem becomes the unsplittable MCFP, where more specialized algorithms are developed [4]. For large values of k, the problem becomes the linear MCFP.

Problem, k	**Heur.**	**Time**	**Tree size**	**Depth**	**Gap**	**UB**
bl03, 2	no	225.06	>34000	75	0.23	15836.0
bl03, 2	yes	225.38	>34000	75	0.23	15836.0
bl03, 3	no	2.98	317	49	0.00	15799.0
bl03, 3	yes	0.44	1	0	0.00	15799.0
bl03, 10	no	2.17	63	31	0.00	15799.0
bl03, 10	yes	0.13	1	0	0.00	15799.0
bs03, 2	no	0.59	125	28	0.00	16488.0
bs03, 2	yes	0.47	97	25	0.00	16488.0
bs03, 3	no	0.17	29	14	0.00	16488.0
bs03, 3	yes	0.02	1	0	0.00	16488.0
bs03, 10	no	1.31	61	25	0.00	16488.0
bs03, 10	yes	0.08	1	0	0.00	16488.0

Table 1: Results for the three-index algorithm with and without the proposed heuristic. The column `H. Time` denotes the time spent in the heuristic.

Name	k	# instances	3-index		2-index	
			A.Mean	**Opt.**	**A.Mean**	**Opt.**
bl	2	11	5.06	6/11	1.90	11/11
bl	3	11	0.43	10/11	0.21	11/11
bl	10	11	0.87	11/11	0.22	11/11
bs	2	11	41.66	3/11	0.32	9/11
bs	3	11	37.95	8/11	0.32	11/11
bs	10	11	1.08	11/11	0.27	11/11
planar	2	5	117.92	4/5	3.09	5/5
planar	3	5	2.58	4/5	2.75	5/5
planar	10	5	267.40	5/5	15.13	5/5
grid	2	7	1.40	4/7	0.24	5/7
grid	3	7	0.09	5/7	0.73	7/7
grid	10	7	7.00	7/7	1.31	7/7

Table 2: The number of test instances solved to optimality with the 3-index and 2-index algorithms, for various k values. **A.Mean** is the average mean time in seconds calculated over those instances solved to optimality by both algorithms.

First off, results of computational evaluations of the branch-and-price algorithm for the three-index model with and without the heuristic are showed in Table 1. The running times are improved significantly by including the proposed heuristic, as this gives a smaller search tree. Hence, the heuristic is included in the remaining tests.

Next, we compare the two branch-and-price algorithms with each other, see Table 2. For $k = 2$, the three-index algorithm shows diffulty in solving many instances, whereas the two-index algorithm has much greater success. The latter also has better running times for instances, both algorithms can solve. Both algorithms, however, fail for larger instances. For $k = 3$ and $k = 10$ both algorithms perform well with respect to the number of solved instances, but the branch-and-price algorithm for the two-index model has better running times.

The running times reflect the complexity of the corresponding problem instances and used algorithms. Whenever the value of k exceeds some threshold value, the running time for solving the instance decreases. The reason for this is that at some point, k does not impose a constraint on the problem, i.e., the instance corresponds to the linear MCFP. The value of k has greater impact on the three-index algorithm. When k takes on a value greater than the mentioned threshold, the running time of the three-index algorithm increases, because columns are generated for each $i = 1, \ldots, k$, and are priced into the master problem. Generating columns and solving a larger master problem is time consuming. The same is obviously not the case for the two-index algorithm.

The three-index algorithm is capable of solving instances with up to 2239 commodities, 850 edges and 150 nodes (planar_{150}), and 400 commodities, 1520 edges and 400 nodes ($\text{grid}_{400:1520:400}$) for $k = 10$, and instances with up to 532 commodities, 1085 edges and 100 nodes (planar_{100}) for $k = 2$. The two-index algorithm solves instances with up to 2239 commodities, 850 edges and 150 nodes (planar_{150}) and 400 commodities, 1520 edges and 400 nodes (grid$_{400:1520:400}$) for $k = 10$, and instances with up to 2239 commodities, 850 edges and 150 nodes (planar$_{150}$) for $k = 2$. Also, the three-index algorithm is capable of solving about 76% of the test instances to optimality, while the two-index has solved just over 96% of the test instances to optimality. Hence, for the far majority of the problem instances the two-index algorithm outperforms the three-index algorithm, both with respect to time spent and to the number of instances solved to optimality. We conclude, that this is partly due to the extra h-index in the three-index model causing symmetry in the solution space, and partly due to the three-index algorithm having k times as many variables as the two-index algorithm.

5 Conclusions

In this paper we have presented a branch-and-price algorithm for the MCMCkFP, which outperforms existing methods. The new branch-and-price algorithm is based on a mathematical formulation, which unlike previous formulations omits a symmetry inducing index for each of the k paths per commodity. Hence, we have named our formulation the two-index model, while the existing model is a three-index model. The two-index model has parallelly been suggested for the Maximum Flow MCkFP by Truffot et al. [9], but they discarded the model because it complicates branching. We have presented a branching strategy for the model, which ensures that the pricing problem can be solved efficiently. The branching strategy and the algorithm for the resulting pricing problem can be directly used on the Maximum Flow problem.

Furthermore, we have introduced a heuristic for the three-index branch-and-price algorithm which transforms certain fractional solutions into feasible solutions. Though the heuristic boosts the performance of the three-index algorithm, it is still outperformed by the two-index algorithm both with respect to time usage and to the number of solved instances. The three-index algorithm including the proposed heuristic has solved 76% of the problem instances to optimality, where the two-index has solved 96% of the problem instances to optimality.

Acknowledgement
We would like to thank GlobalConnect A/S for their support of this work.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orling. *Network flows*. Prentice-Hall Inc., 1993.

[2] F. Alvelos and J. M. Valério de Carvalho. Comparing branch-and-price algorithms for the unsplittable multicommodity flow problem. In *International Network Optimization Conference*, pages 7–12, 2003. http://pessoais.dps.uminho.pt/falvelos/mcf_instances/carbin.zip.

[3] G. Baier, E. Köhler, and M. Skutella. On the k-splittable flow problem. In *10th Annual European Symposium on Algorithms*, pages 101–113, 2002.

[4] C. Barnhart, C. A. Hane, and P. A. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48:318–326, 2000.

[5] J. M. Kleinberg. *Approximation algorithms for disjoint paths problems*. PhD thesis, MIT, Cambridge, MA, 1996.

[6] T. Larsson and Di Yuan. An augmented lagrangian algorithm for large scale multicommodity routing. *Computational Optimization and Applications*, 27:187–215, 2004.

[7] R. Lougee-Heimer. The common optimization interface for operations research. *IBM Journal of Research and Development*, 47:57–66, 2003.

[8] J. Truffot and C. Duhamel. A branch and price algorithm for the k-splittable maximum flow problem. *Discrete Optimization*, 5(3):629–646, 2008.

[9] J. Truffot, C. Duhamel, and P. Mahey. Branch & price pour le problème du multiflot k-séparable de coût minimal. In *LIMOS, UMR 6158 - CNRS, ROADEF'05*, 2005.

[10] J. Truffot, C. Duhamel, and P. Mahey. Using branch-and-price to solve multicommodity *k*-splittable flow problems. In *Proceedings of International Network Optimization Conference (INOC)*, 2005.

[11] D. Villeneuve and G. Desaulniers. The shortest path problem with forbidden paths. *European Journal of Operational Research*, 165(1):97–107, 2005.