Egerváry Research Group
on Combinatorial Optimization

Technical reportS

TR-2008-12. Published by the Egerváry Research Group, Pázmány P. sétány 1/C, H–1117, Budapest, Hungary. Web site: www.cs.elte.hu/egres. ISSN 1587–4451.

Strongly polynomial time solvability of integral and half-integral node-capacitated multiflow problems

Gyula Pap

May 2008

Strongly polynomial time solvability of integral and half-integral node-capacitated multiflow problems ⋆

Gyula Pap ⋆⋆

Abstract

We show the strongly polynomial time solvability of the problem of finding a maximum half-integral node-capacitated multiflow, and of the problem of finding a maximum integral node-capacitated multiflow.

1 Introduction

We consider an undirected graph $G = (V, E)$, a set $A = \{a_1, a_2, \cdots, a_k\} \subseteq V$ of terminals, and a vector $c \in \mathbb{N}^V$ of node-capacities. A multiflow is a set of flows x_{ij} between pairs of terminals $a_i, a_j, i < j$, and we define its size by the sum of the size of these flows. The maximum fractional node-capacitated multiflow problem is to find a maximum size multiflow that does not exceed the capacity of any node $v \in V$, i.e. the total traffic of the flows x_{ij} through v is no more than $c(v)$. A rigorous definition is given below. The (half-)integral version of this problem is that, in addition, we require that the flows are (half-)integral. We remark that, by the path-decomposition of flows, multiflow problems are equivalent with the respective A-paths packing problems, which are detailed below.

A min-max formula for all three problems is already known: a min-max formula for the fractional version comes from LP duality, for the half-integral version comes from the existence of a half-integral LP optimum, and the integral version – the most interesting of all – is solved by Mader's [14] min-max formula. Note that the special case of $V = A$ is b-matching, thus Mader's Formula contains for example, the Berge-Tutte formula for non-bipartite matching; thus Mader's formula is involved with parity.

The algorithmic complexity of all three problems (fractional, integral, half-integral) has been addressed in the literature. A strongly polynomial time algorithm for the fractional problem follows from Frank and Tardos' [6] strongly polynomial version of the ellipsoid method to solve a $0, \pm 1$ LP (an LP where the constraint matrix only has $0, \pm 1$ entries, but the bounding vector and the objective vector are arbitrary). Note

⋆Research is supported by OTKA grant K60802 and by the MTA-ELTE Egerváry Research Group of the Hungarian Academy of Sciences.

⋆⋆MTA-ELTE Egerváry Research Group (EGRES), Dept. of Operations Research, Eötvös University, Pázmány P. s. 1/C, Budapest, Hungary H-1117. e-mail: gyuszko@cs.elte.hu

that the LP does not define a half-integral polytope, so the existence of a half-integral optimum is not trivial, and is not easy to find just using the LP. A polynomial time algorithm for the integral problem in case of "small" capacities follows from Lovász' linear matroid matching theory [12] with Schrijver's [20] linear matroid matching representation. This also implies a polynomial time algorithm for the half-integral problem with small capacities. The polynomial time solvability of the integral problem with arbitrary node-capacities is known from a proximity lemma of the author [15], and that also implies a polynomial time for the half-integral problem with arbitrary capacities. A better (polynomial) running time to solve solve the half-integral problem is achieved by Babenko and Karzanov [1] with capacity scaling. None of these approaches implies a strongly polynomial time algorithm for neither the integral, nor for the half-integral node-capacitated multiflow problem. Note that there are further variations, and special cases, that by themselves have attracted the attention of the combinatorial optimization community. A variation is concerned with edge-capacities instead of node-capacities. Keijsper, Pendavingh and Stougie [10] showed that the integral edge-capacitated multiflow problem is solvable in polynomial time (not strongly!). Ibaraki, Karzanov, Nagamochi [9] implies that the half-integral edge-capacitated problem is solvable in strongly polynomial time. This, together with the author's proximity lemma approach [15] implies that the integral edge-capacitated version is solvable in strongly polynomial time.

Thus, to the best knowledge of the author, the above mentioned two versions (the half-integral and the integral node-capacitated multiflow problem) have not yet been shown to be solvable in strongly polynomial time, only their polynomial time solvability is known. To show their strongly polynomial time solvability, in this paper, we use a different approach to those cited above. We introduce the notion of region-constrained multiflows for which we prove a half-integrality result, implying that a maximum half-integral region-constrained node-capacitated multiflow may be constructed in strongly polynomial time. Then we solve the maximum half-integral node-capacitated multiflow by reducing it to the region-constrained version, and thus obtain a strongly polynomial time algorithm. Here we use Frank and Tardos' strongly polynomial version of the ellipsoid method for $0, \pm 1$ LP to construct the regions used in the reduction. Finally, we find a maximum integral node-capacitated multiflow by using a half-integral optimum and the proximity lemma approach of [15].

We remark that the algorithms presented in this paper are not satisfactory in terms of running time, since at some places, we require a subroutine to solve a $0, \pm 1$ LP, which by Frank and Tardos [6] is solvable in strongly polynomial time via ellipsoid method and diophantine approximation. On the other hand, the construction of our algorithm for the half-integral and integral problem is quite tricky, since we need the introduction of the notion of region-constrained multiflows. The construction of the two algorithms (one for the half-integral and another for the integral node-capacitated problem) is based on the following principles: the proximity lemma for integral node-capacitated multiflows [15], a reduction of half-integral node-capacitated multiflows to region-constrained half-integral node-capacitated multiflows, and a half-integrality property of the region-constrained node-capacitated multiflow polytope.

A survey of further related results we will be given below, but next we recall the definitions.

2 Definitions

As input we are given an undirected graph $G = (V, E)$, a fixed subset $A \subseteq V$ of nodes, and a vector $c : V \to \mathbb{N} = \{0, 1, 2, \cdots\}$. Denote $A = \{a_1, a_2, \cdots, a_k\}$. Nodes a_i are called *terminals*. For $v \in V$, the value $c(v)$ is called the *capacity* of node v.

There are two equivalent ways to formulate the problem, and first we take on the formulation by considering A-paths. A path is called an *A-path* if its ends are two distinct terminals, and its internal nodes are in $V - A$. A *node-capacitated fractional packing* of A-paths is an assignment of non-negative "fractional multiplicities" to the A-paths such that for every node v the sum over all A-paths traversing v does not exceed its capacity $c(v)$. More formally, let $\mathcal{P}$ denote the set of all A-paths, and for a node $v \in V$, let $\mathcal{P}_v$ denote the set of paths $P \in \mathcal{P}$ such that $v \in V(P)$. A node-capactitated fractional packing is a function $m : \mathcal{P} \to \mathbb{R}_+$ subject to the capacity constraints $m(\mathcal{P}_v) \leq c(v)$. The *size* of a node-capatitated fractional packing of A-paths is defined as the sum of fractional multiplicities of all A-paths. A node-capacitated fractional packing of A-paths is called a *node-capacitated integral packing* if all multiplicities are integers. The formal LP representation of the node-capacitated fractional packing problem is the following, over vectors $m : \mathcal{P} \to \mathbb{R}$:

$$\max \mathbf{1} \cdot m \quad \text{subject to} \tag{1}$$

$$m \geq 0 \tag{2}$$

$$m(\mathcal{P}_v) \leq c(v) \qquad \text{for } v \in V \tag{3}$$

Note that the number of variables might be exponential. The *node-capacitated fractional packing problem* is that, given G, A, c as above, find a node-capacitated fractional packing of maximum size $\nu^*(G, A, c)$. The *node-capacitated integral packing problem* is that, given G, A, c as above, find a node-capacitated integral packing of maximum size $\nu(G, A, c)$.

Second, we consider an equivalent formulation of this problem by considering multiflows. Suppose we are given G, A, c as above. Let $G' = (V, E')$ denote the digraph obtained from G by replacing the edges $uv \in E$ by the two arcs uv and vu. A *multiflow* is a family $x = \{x_{ij}\}$ of (fractional) flows in G' from a_i to a_j, where i, j runs through the pairs $1 \leq i < j \leq k$, and is subject to the capacity contraints. A flow x_{ij} is assumed to have zero flow into nodes of $A - a_j$ and zero flow out of nodes of $A - a_i$. The *traffic of x_{ij} through a given node $v \in V$* is defined by $\mu_{x_{ij}}(v) := x_{ij}(\delta^{in}(v))$ for $v \neq a_i$, and $\mu_{x_{ij}}(a_i) := x_{ij}(\delta^{out}(a_i))$, and the *traffic of x through a given node $v \in V$* is defined by $\mu_x(v) := \sum_{i<j} \mu_{x_{ij}}(v)$. The *capacity constraint* for a given node $v \in V$ is written as $\mu_x(v) \leq c(v)$ for $v \in V$. The size of a flow is $size(x_{ij}) := x_{ij}(\delta^{in}(a_j))$, and the *size of a multiflow* is $size(x) = \sum_{i<j} size(x_{ij})$. The "condensed" LP representation of the

multiflow problem is the following:

$$\max size(x) \quad \text{subject to} \tag{4}$$
$$x \text{ is a multiflow with respect to } G, A \tag{5}$$
$$\mu_x(v) \le c(v) \qquad \text{for } v \in V, \tag{6}$$

which is the abbreviation of the following "full" LP representation:

$$\max \sum_{i<j} x_{ij}(\delta^{in}(a_j)) \quad \text{subject to} \tag{7}$$
$$x_{ij}(\delta^{in}(a_l)) = 0 \qquad \text{for } i<j, l \ne j \tag{8}$$
$$x_{ij}(\delta^{out}(a_l)) = 0 \qquad \text{for } i<j, l \ne i \tag{9}$$
$$x_{ij}(\delta^{in}(v)) - x_{ij}(\delta^{out}(v)) = 0 \qquad \text{for } i<j, v \in V-A \tag{10}$$
$$\sum_{i<j} x_{ij}(\delta^{in}(v)) \le c(v) \qquad \text{for } v \in V-A \tag{11}$$
$$\sum_{i<l} x_{il}(\delta^{in}(a_l)) + \sum_{j>l} x_{lj}(\delta^{out}(a_l)) \le c(a_l) \qquad \text{for } 1 \le l \le k. \tag{12}$$

A multiflow x is called an integral multiflow if all entries $x_{ij}(uv)$ are integers. The maximum fractional node-capacitated multiflow problem (FNCMP) is that, given G, A, c as above, find a fractional multiflow of maximum size, that is, solve the LP. The integral node-capacitated multiflow problem (INCMP) is that, given G, A, c as above, find a integral multiflow of maximum size, that is, solve the IP. Observe that, by the path-decomposition of flows, the maximum size of a fractional (resp. integral) multiflow is equal to the maximum size of a frational (resp. integral) packing of A-paths, thus the maximum is equal to $\nu^*(G, A, c)$ (resp. $\nu(G, A, c)$).

The half-integral version of the above problem, denoted by HNCMP, is to solve (7)-(12) under the additional constraint that x is half-integral. We remark that, the LP (7)-(12) does have half-integral optimum, which thus is an optimum solution of both FNCMP and HNCMP. The ellipsoid method may be used to generate an optimum basic solution, a vertex of the polytope. However, unluckily, the polytope (8)-(12) itself is not half-integral, that is, it may have vertices that are not half-integral, as is show in an example below. Thus an optimum vertex generated by the ellipsoid method will not necessarily be half-integral, thus to solve HNCMP we need a more sophisticated approach, and the same goes for INCMP.

2.1 An example that the multiflow polytope is not half-integral

Basically, the vertex of any $0, 1$-coefficient polytope arises as the vertex of (8)-(12). A concrete example is given as follows. Let $G = (V, E)$ be the graph of a cycle of length $k \ge 4$, with a pendant edge incident with every node of the cycle. Let A be the set of nodes not in the cycle, that is the other end of the pendant edges. Thus $V = A \cup A'$, where $A = \{a_1, \cdots, a_k\}$, $A' = \{a'_1, \cdots, a'_k\}$, $E = \{a_i a'_i : i = 1, \cdots, k\} \cup \{a_i a_j : j \equiv i + 1 \,(mod\, k)\}$. The node-capacity is defined by $c \equiv 1$. Now we define x_{ij} to be all zero if $j \not\equiv i+2$, and $x_{ij} := \frac{1}{k-1} P_i$ if $j \equiv i+2$, where P_i denote the "long" directed path from a_i to a_j (i.e. $|V(P_i)| = k+1$). Then x is a feasible multiflow, since the total traffic of every node in A' is equal to its capacity 1, and the total traffic through nodes in A is equal to $\frac{2}{k-1}$. To see that x is in fact a vertex of the polytope (8)-(12), notice that all the values of x are zero except for those corresponding to a path P_i. Thus the constraints that are tight with respect to x are only solved by a linear combination of the flows P_i, say P_i is taken with multiplicity λ_i in this combination. Also the node-capacity of the nodes in A' is tight, since exactly $k-1$ of those paths traverse one of them. The only combination λ_i that makes each of these node-capacities tight is $\lambda_i = \frac{1}{k-1}$. Thus we have proved that x is the only solution of the system of equations given by the system of tight inequalities, which shows that x is a vertex of the LP (7)-(12) that is not half-integral.

2.2 The twelve versions

Note that there is an *edge-capacitated version*, where we are looking for a multiflow subject to edge-capacities (instead of node-capacities) and the capacity of an edge imposes an upper bound on the total traffic through that edge. Thus the maximum fractional edge-capacitated multiflow problem (FECMP) is given by an LP similar to (4)-(6). Also note that the most looked-at version of both the edge-capacitated, and the node-capacitated problems is the special case of edge-disjoint-, and node-disjoint A-paths, which is equivalent to the special case of all-one capacities. Mader's [14] problem of node-disjoint $\mathcal{A}$-paths (where $\mathcal{A}$ is a partition of A) is equivalent to node-capacitated A-paths with the capacities infinity for the terminals and capacities one for the non-terminals. Thus the variations of multiflow and A-path problems considered in this survey are given by the following three parameters:

1. fractional (F), half-integral (H) or integral (I),
2. edge-capacities (E) or node-capacities (N),
3. all-one capacities (1) or arbitrary capacities (C) (always assumed integral).

Note that these three parameters are becoming more and more general from left to right, thus the most general case is that of finding a maximum integral multiflow subject to arbitrary node-capacities (INCMP). A total of 12 variations are possible. A thirteenth version is the problem of internally node-disjoint A-paths considered by Mader [14], which is equivalent with node-capacity one for non-terminals, and node-capacity infinity (or equivalently, $|V|$) of terminals. This version is denoted by IN1'MP. Next we survey the known results on these variations that are relevant to the topic of this paper. These citations are organized primarily with respect to the technique, and not chronologically.

2.3 Min-max results

We remark that the four fractional versions may be solved by the linear programming description (7)-(12) or similar, which is a $0, \pm 1$ LP, and thus solvable in strongly polynomial time by combining the ellipsoid method with the diophantine approximations of Frank and Tardos [6]. The remaining eight versions may not be solved by the LP directly. A min-max formula for HE1MP and HECMP was given by Lovász and Cherkassky, and a min-max formula for HN1MP and HNCMP follows from a result in the book of Vazirani [22]. These min-max results for the four half-integral multiflow problems are based on the existence of a half-integral dual optimum of an LP. The four integral multiflow problems are even more tricky than the half-integral ones, since parity comes into play, and non-bipartite matching is in fact a special case of all four integral multiflow problems. A min-max formula for the four integral problems has been given by Mader [14, 13], which thus generalizes the Berge-Tutte formula for non-bipartite matching. Further results: Sebő and Szegő [21] proved a Gallai-Edmonds-type structural description, that implies that there is a "canonical", that is a uniquely defined dual optimum, and such that this canonical dual optimum is invariant over isomorphisms of the graph with al the terminals fix points. Chudnovsky et al. [4, 5] and then the author [17, 19] considered the problem of disjoint A-paths subject to an algebraic constraint, that is, the disjoint non-zero A-path problem and the disjoint non-returning A-path problem, respectively.

2.4 Algorithmic results

The known algorithmic results are depicted below, where P stands for polynomial time and SP stands for strongly polynomial time. We remark that in some of the below cited papers the complexity result is not stated explicitly, but follows easily, and thus it is more than fair to say that the complexity result is due to that paper. For the notation H,I,E,N,1,C, see above.

	H	I
E1	P Lovász [11] or Cherkassky [3]	P Lovász [12] with Schrijver [20]
EC	SP Ibaraki-Karzanov-Nagamochi [9]	P / SP Keijsper et al. [10]: P Ibaraki et al. [9] with Pap [15]: SP
N1	P Lovász [12] with Schrijver [20] or Pap [16], or Babenko [2]	P N1: Gallai [7] N1': Lovász [12] with Schrijver [20]
NC	P via ellipsoid, structure, rounding (SP: this paper)	P Pap [15] or Babenko-Karzanov [1] (SP: this paper)

The results in the table are explained below:

Around 1972, Lovász [11] and Cherkassky [3] independently proved a min-max formula for IE1MP, with the eulerian assumption that all non-terminals have even degree. The result says that the minimum $a, A-a$ cuts are sufficient to determine the maximum. This implies a min-max formula for FECMP, and the maximum attains with a half-integral multiflow, which thus implies a min-max formula for HECMP. Since minimum cuts may be determined in polynomial time, this also implies a polynomial time algorithm to find a IECMP, under the eulerian assumption. A polynomial time algorithm follows to find a FECMP, and the maximum obtained is half-integral. By Frank and Tardos' Theorem 3.1, the ellipsoid method implies a strongly polynomial time algorithm for FECMP, but the maximum obtained is not necessarily half-integral. This, by rounding, implies a polynomial (but not strongly polynomial) time algorithm to solve HECMP.

As opposed to Lovász' and Cherkassy's result, without the eulerian assumption, minimum $a, A - a$ cuts do not determine the optimum of IE1MP, IECMP, which can be seen by $V = \{v_1, v_2, v_3, v_4\}, E = \{v_1v_2, v_1v_3, v_1v_4\}, A = \{v_2, v_3, v_4\}$. Thus, to solve IE1MP or IECMP, we need a more sophisticated method, and in fact, parity comes into play. In 1978, Mader [13] solved this by proving a min-max formula for IE1MP, without the eulerian assumption. Parity is essential in Mader's formula. On the other hand, IN1MP, which also contains non-bipartite matching as a special case, is shown by Gallai [7] to be solvable via a reduction to non-bipartite matching. Mader also considered so-called internally disjoint A-path problem, which is equivalent with maximum integral multiflow subject to node-capacity one for the non-terminals and infinity for the terminals. Let us refer to this problem by IN1'MP. Thus, for IN1'MP, Mader [14] proved a min-max formula (see Theorem 4.2 below). Though non-bipartite matching is a special case, and parity of "components" is essential, neither of Mader's formulas is known to reduce to non-bipartite matching. Mader observed that – by constructing the exponential sized capacity-expanded graph – these min-max formulas imply a min-max formula for IECMP and INCMP. Mader's original proof is non-algorithmic, and thus the construction of a polynomial time algorithm remained an open question for quite a while.

A polynomial time algorithm for IN1'MP and IE1MP follows from Lovász' linear matroid matching algorithm [12], and Schrijver's construction showing that it reduces to linear matroid matching. Some other combinatorial algorithms for IN1'MP, and generalizations have been given recently [5, 4, 15, 17, 19]. By applying any algorithm for IN1'MP in the capacity-expanded graph one obtains an exponential algorithm to solve INCMP and IECMP. Note that, for "small" capacities ($poly(|V|)$), this exponential algorithm becomes polynomial.

HN1'MP may be solved, by a reduction to IN1'MP, via linear matroid matching. A different solution follows from the author's [16, 18] result showing the existence of a special optimum, such that the components of its support are A-paths and so-called odd A-cycles. Thus we obtain a polynomial time algorithm as follows: We determine the optimum value by the linear programming description. Then we delete edges as long as the deletion does not decrease the optimum value. Finally we obtain a graph that is the collection of disjoint A-paths and odd A-cycles; there the optimum solution is uniquely determined and easy to construct. Another different solution of HN1'MP (best running time) has been given by Babenko [2], and that solution is also based on odd A-cycles (which he called odd stars).

Ibaraki, Karzanov, Nagamochi constructed a strongly polytime algorithm to IECMP, under the eulerian assumption that the sum of capacities of edge incident with a non-terminal is assumed to be even. This implies a strongly polynomial time algorithm for HECMP. Their algorithm is based on a reduction to the three terminal case, and then solving the three terminal case by the avarage of two integral circulations. This, together with a proximity lemma of the author [16, 15] implies a strongly polynomial time algoritm for HECMP.

Keijsper, Pendavingh and Stougie [10] proved a polyhedral description that, via ellipsoid method, implies a polynomial time algorithm to solve IECMP. This approach only implies a polynomial time algorithm.

3 On the definition of polynomial and strongly polynomial running time

The point in this paper is the construction of polynomial and strongly polynomial time algorithms. It's true that these notions from complexity theory are essential to combinatorial optimization, but nevertheless, there is some ambiguity about the precise definition. Thus here we recall the definition of these notion based on [20]. The ambiguity is about what operations are allowed to be used in the algorithm, and how do we bound the size of numbers occurring while running the algorithm.

3.1 Polynomial running time

In a *polynomial time algorithm*, the input is given by n bits. The algorithm is allowed to perform *logical operations* with these bits, which are counted as one step in the running time. An algorithm is called polynomial time if a polynomial of n is proved to be an upper bound on the number of operations of the algorithm. An issue, which often is neglected, is the representation of rational numbers in a polynomial time algorithm. Consider a problem with rational numbers in the input, and assume they are given in binary digits of their numerator and denominator. Arithmetic operations such as addition, subtraction, multiplication, division, maximum, minimum, comparison of rationals may be performed in polynomial time. What often is neglected in the analysis of an algorithm is the potentially growing size of numbers. Note that the well-known algorithm to determine the decimal digits of 2^k – by using the binary digits of k, and iteratively squaring 2 – is not a polynomial time algorithm, since the output number 2^k has around $(\log_{10} 2)k$ decimal digits, whereas the input only has $\log_2 k$! On the other hand, the algorithm only needs a polynomial number of arithmetic operations. Thus, to prove that some algorithm dealing with numbers has in fact polynomial running time, one has to bound the size of numbers occurring while performing the algorithm. This is not completely trivial even for Gaussian elimination, so be careful!

3.2 Strongly polynomial running time

In a *strongly polynomial time algorithm*, the input consists of n integer numbers, and m bits. Typically, the bits are there to specify a graph, and the numbers are there to specify capacities. Operations allowed to be performed within a strongly polynomial time algorithm are the *arithmetic operations* addition, subtraction, multiplication, division, and the comparison of numbers, and *logical operations* of bits, each of which will count as one step in the computation of running time. Of a strongly polynomial time algorithm we require two things, one of which is a $poly(n + m)$ upper bound on the number of steps (arithmetic or binary) performed by the algorithm. The other property is a polynomial bound on the growth of numbers, defined as follows. Consider, as input numbers, n rational numbers, and let N denote the sum of their sizes. Then, while running the algorithm, a lot of numbers occur, and let N' denote the sum of their sizes. Then the *factor of growth of numbers* is defined as N'/N. We require is that there is a $poly(n + m)$ general upper bound on the growth of numbers. Thus, in the definition of a strongly polynomial time algorithm, we require a $poly(n + m)$ upper bound on the number of arithmetic and logical operations, and the factor of growth. The point in this is that without the second property, the algorithm to compute 2^k would be strongly polynomial. This algorithm is not regarded as strongly polynomial time, since the factor of growth is exponential. Also note that rounding down a rational number – which is a polynomial time algorithm using the Euclidian algorithm – is not strongly polynomial time either. On the other hand, with the given definition, a strongly polynomial time algorithm always implies a polynomial time algorithm for the case when the input numbers are integers given in binary digits.

3.3 $0, \pm 1$ linear programming in strongly polynomial time

Often, in papers claiming a strongly polynomial time algorithm, the technical proof of bounding the growth of numbers is omitted, but this gap is easily filled in most cases, and is left to the reader. A notable exception is linear programming, where a polynomial time algorithm is known via Khachiyan's ellipsoid method, but constructing a strongly polynomial time algorithm remains an open question. A partial solution has been given by Frank, Tardos [6] who, using a diophantine approximation scheme, proved that an LP with a $0, \pm 1$ constraint matrix, and arbitrary bounding and objective vector, may be solved in strongly polynomial time. The technique to prove this result is to get rid of the "big" numbers in b, c, and prove that there is another LP with "small" numbers, that is essentially equivalent with the original LP. This "small" LP is then solved via ellipsoid method, or your favorite LP solver.

Theorem 3.1 (Frank, Tardos [6]). *There is a strongly polynomial time algorithm, achieving a running time $poly(n + m)$, to find an optimum basic solution of the LP* $\max cx : Ax \leq b$*, assuming* $A \in \{-1, 0, 1\}^{n \times m}$*,* $c \in \mathbb{Z}^n$*, and* $b \in \mathbb{Z}^m$*.*

Frank and Tardos' algorithm itself will be used as a subroutine in some algorithms presented below, and in some other algorithms, the idea of Frank and Tardos' proof – reducing the problem to an instance with small numbers – will be essential.

3.4 Parity of input numbers

In some capacitated combinatorial optimization problems, such as the integral b-matching problem, the parity of the input numbers plays a role. Find, for example, a maximum b-matching in K_3, where b is defined as the constant-β-vector on the node set of K_3, for some $\beta \in \mathbb{N}$. The only maximum b-matching will be the constant-$\lfloor \frac{1}{2}\beta \rfloor$-vector on the edge set of K_3. Unluckily, rounding down is not considered an arithmetic operation in a strongly polynomial time algorithm. To determine this output number, one would at least have to know the parity of β. Thus, in this sense, the maximum integral b-matching problem may not be solved in strongly polynomial time. On the other hand, why don't we simply assume the knowledge of the parity of input numbers? More precisely, consider the problem of finding a maximum integral b-matching, assuming an input consisting of the graph, vector b, and the parity vector of b. Under this assumption, Gerards [8] constructed a strongly polynomial time algorithm to find a maximum integral b-matching. That approach of Gerards – the Proximity Lemma – has in fact motivated a method used in this paper. In view of these considerations, the input of the maximum integral multiflow problem subject to node-capacities should be given as a graph $G = (V, E)$, a set $A \subseteq V$, a vector $c \in \mathbb{N}^V$, and a set $O \subseteq V$ satisfying $O = \{v : v \in V, c(v) \text{ is odd }\}$. Under this assumption, the vector $\lfloor \frac{1}{2}c \rfloor$ follows easily.

4 Mader's min-max formula

In our quest for a strongly polynomial time algorithm, we must rely on the well-known result of Mader claiming a min-max formula for node-disjoint A-paths. The fact that this min-max formula is true will be essential in the proof of correctness of our algorithms.

Let $G = (V, E)$ be an undirected graph as above, and consider a set $A \subseteq V$ of terminals, and $c \in \mathbb{N}^V$ of node-capacities. The main result on node-capacitated A-paths is the node-capacitated integral version of Mader's min-max formula [14], i.e. a formula for INCMP, which determines the maximum size of a node-capacitated integral packing in terms of the minimum of the value of a so-called A-partition. An *A-partition* is a family $\mathcal{X} = \{X_0; X_1, \cdots, X_k\}$ of disjoint subsets of V such that $a_i \in X_0 \cup X_i$ for $i = 1, \cdots, k$. Let $X := \bigcup_{i=1}^k X_i$. Let $\mathcal{K}$ denote the set of those components of the subgraph $G - X_0 - \bigcup_{i=1}^k E[X_i]$ that are not isolated nodes, i.e. that have at least two nodes. Members of $\mathcal{K}$ are called *components* (with respect to $\mathcal{X}$). The *border of a component* K is defined as $V(K) \cap X$. The *value* of A-partition $\mathcal{X}$ is defined as the sum of the capacity of X_0, plus half the capacities of the borders rounded down, that is,

$$val(G, c, \mathcal{X}) := c(X_0) + \sum_{K \in \mathcal{K}} \left\lfloor \frac{1}{2} c(V(K) \cap X) \right\rfloor \qquad (13)$$

The point of this definition is that the value of an A-partition provides an upper bound on $\nu(G, A, c)$:

Claim 4.1.

$$\nu(G, A, c) \leq val(G, c, \mathcal{X}) \tag{14}$$

Proof. To see this, consider an integral packing m. Let $\mathcal{P}_0$ be the set of those A-paths that traverse at least one node in X_0, and let $\mathcal{P}_1$ be the set of those A-paths disjoint from X_0. Clearly, $\mathcal{P} = \mathcal{P}_0 \cup \mathcal{P}_1$, and $m(\mathcal{P}_0) \leq b(X_0)$. Now notice that for all $P \in \mathcal{P}_1$ there is a component K such that $|V(P) \cap V(K) \cap X| \geq 2$, i.e. P hits the border of K twice. Let us assign to P one of these components, say K_P. The capacity constraints for the nodes in $V(K) \cap X$, and the integrality of m imply that $\sum_{P:K_P=K} m(P) \leq \lfloor \frac{1}{2} c(V(K) \cap X)) \rfloor$. Finally we get that $m \cdot \mathbf{1} = m(\mathcal{P}_0) + \sum_{K \in \mathcal{K}} \sum_{P:K_P=K} m(P) \leq val(G, c, \mathcal{X})$, implying (14).

Conversely, Mader proved that the minimum value of an A-partition is equal to $\nu(G, A, c)$. Note that, assuming the knowledge of the parity of input numbers (entries of vector c), the value of an A-partition may be determined in strongly polynomial time. Thus, Mader's following theorem indicates that there may be a strongly polynomial time algorithm to solve INCMP, which to the best knowledge of the author, remained an open question, and which will be solved in this paper. Mader's Formula is essential to this paper, and will be used in the proof of correctness of the algorithms.

Theorem 4.2 (Mader [14], node-capacitated version, INCMP)**.** *Let $G = (V, E)$ be an undirected graph, consider a set $A \subseteq V$, and a vector $c : V \to \mathbb{N}$. Then*

$$\nu(G, A, c) = \min val(G, c, \mathcal{X}) \tag{15}$$

where the minimum is taken over A-partitions $\mathcal{X}$.

Mader's uncapacitated formula claims the special case of Theorem 4.2 for G', A', c' such that $c'(v) = 1$ for $v \in V' - A'$ and $c'(v) = +\infty$ for $v \in A'$. (This is the case of IN1'MP.) In this special case, $\nu(G', A', c')$ is equal to the maximum number of internally disjoint A'-paths, i.e. paths not sharing a node in $V' - A'$.

Proof. (Sebő, Szegő, [21], reduction to Mader's uncapacitated min-max formula IN1'MP.) We apply the uncapacitated to the expanded graph $G' = G^c$ defined as follows. Construct G^c from G by splitting every node $v \in V$ into $c(v)$ identical copies, and for all edges $uv \in E$ introducing an edge between all pairs of copies of the two nodes u, v, and finally, introducing $A' := \{a' : a \in A\}$ and joining a' by an edge with all other copies of a. Clearly, $\nu(G', A', c') = \nu(G', A', c')$. Sebő and Szegő [21] proved a Gallai-Edmonds-type structural description of a "canonical" A-partition in Mader's uncapacitated min-max formula, that is, a dual optimum that is invariant over isomorphisms of the graph with all terminals fix points. Let $\mathcal{X}'$ be this canonical A-partition. Clearly, $A' \cap X_0 = \emptyset$. Moreover, note that permutations of the copies of a node are isomporphisms, thus, by Sebő-Szegő, if a copy of a node is in some set $X_i \in \mathcal{X}'$, then each of its fellow copies must also be in X_i. Now define $\mathcal{X}$ to be the image of $\mathcal{X}'$ in G, i.e. X_i be the set of nodes with each of its copies being in X_i'. It is quite easy to see that $val(G, \mathcal{X}, c) = val(G', \mathcal{X}', c')$, completing the proof of Theorem 4.2.

We remark that there is a polynomial time algorithm to solve the special case of IN1'MP (the uncapacitated version), by applying Lovász' linear matroid matching algorithm; for further results, see the above survey. Lovász' algorithm, by applying it to the expanded graph G', implies an algorithm for "small" ($poly(|V|)$) capacities. A strongly polynomial time algorithm is given later in this paper.

Theorem 4.2 may also be used to prove the following min-max formula on the fractional problem. The fractional multiflow problem may be thought of as an LP, and thus, LP duality already provides a min-max formula. A strenghtening of LP duality is claimed by the following theorem, a half-integrality result on multiflows, which of course is equivalent with a half-integrality result on node-capacitated packing. For this, consider a vector $w : V \to \mathbb{R}$. w is called a *fractional cover* if $0 \le w \le 1$, and for all A-paths P, $w(V(P)) \ge 1$ holds. For a cover w, $c \cdot w$ is an upper bound on the size of a multiflow. LP-duality implies that, conversely, the maximum size of a multiflow is equal to the minimum of $w \cdot b$ taken over all covers w. A strengthening of this is that the minimum of $w \cdot c$ attains with a *half-integral cover*, that is $w : V \to \{0, \frac{1}{2}, 1\}$. And moreover, the maximum size of a multiflow attains with a *half-integral multiflow*, that is, a multiflow x such that $x_{ij} : E' \to \frac{1}{2}\mathbb{N}$.

Theorem 4.3 (Mader [14], node-capacitated fractional/half-integral version, FNCMP, HNCMP)**.** *Let $G = (V, E)$ be an undirected graph, consider a set $A \subseteq V$, and a vector $c : V \to \mathbb{N}$. Then*

$$\nu^*(G, A, c) = \min w \cdot c \tag{16}$$

where the minimum is taken over fractional covers. Moreover, the minimum is taken over a half-integral cover, and $\nu(G, A, c)$ attains with a half-integral multiflow.

Proof. (Reduction to Theorem 4.2.) Instead of the fractional problem with respect to G, A, c, we consider the integral problem with respect to $G, A, 2c$. Let $\mathcal{X}$ be an A-partition such that $\nu(G, A, 2c) = val(G, 2c, \mathcal{X})$. Note that, since $2c$ is an even vector, one can omit rounding the capacity of the borders, that is, $val(G, 2c, \mathcal{X}) = c(X_0) + \sum_{K \in \mathcal{K}} \frac{1}{2} c(V(K) \cap X)$. Define $w(v) := 1$ if $v \in X_0$, and $w(v) := \frac{1}{2}$ if $v \in V(K) \cap X$ for some $K \in \mathcal{K}$, and $w(v) := 0$ otherwise. Then $2c \cdot w = val(G, 2c, \mathcal{X}) = \nu(G, A, 2c)$. Note that, by multiplying an integral multiflow with respect to $G, A, 2c$ by onehalf one obtains a half-integral multiflow with respect to G, A, c. This implies that $\nu(G, A, c) = \frac{1}{2}\nu(G, A, 2c)$. Finally we get that $w \cdot c = \frac{1}{2}\nu(G, A, 2c) = \nu(G, A, c)$, implying (16), and note that this also implies the assertions on half-integrality.

A polynomial time algorithm for this half-integral problem in case of "small" ($poly(|V|)$) capacities follows from the integral version, Lovász' algorithm. A polynomial (but not strongly polynomial) time algorithm for arbitrary capacities follows from the proximity lemma approach of [15], and a better polynomial running time algorithm is given by Babenko and Karzanov [1] via capacity scaling. A strongly polynomial time algorithm is given in this paper, and that will also be usedto solve the integral problem.

5 Region-constrained multiflows

A goal of this paper is to solve HNCMP, that is to find a maximum half-integral multiflow subject to node-capacities, and to do this, we will use a subroutine to solve another problem where we impose additional constraints by so-called regions. We remark that this region-contraint is related to, but quite different from the "island" constraint of Babenko [2] and Banenko-Karzanov [1].

Assume as input a graph $G = (V, E)$, a set $A = \{a_1, \cdots, a_k\} \subseteq V$ of terminals, and a vector $c : V \to \mathbb{N}$ of capacities. Assume, moreover, that we are given disjoint sets $R_i \subseteq V$ for $i = 1, \cdots, k$ such that $a_i \in R_i$. These sets R_i are called *regions*. Let $R := \cup_i R_i$, and $Q := V - R$. The *region constraint* says, informally, that a flow x_{ij} from a_i to a_j is zero on all edges entering R_i or $V - R_j$, and is zero between any pair of nodes in $V - R_i - R_j$. Formally, the *region constraint* is (23) and (24), which is added to the LP formulation (7)-(12) of node-capacitated multiflows, and thus we obtain the following LP description of the fractional region-constrained node-capacitated multiflow problem FRNCMP:

$$\max \sum_{i<j} x_{ij}(\delta^{in}(a_j)) \quad \text{subject to} \tag{17}$$

$$x_{ij}(\delta^{in}(a_l)) = 0 \qquad \text{for } i < j, l \neq j \tag{18}$$

$$x_{ij}(\delta^{out}(a_l)) = 0 \qquad \text{for } i < j, l \neq i \tag{19}$$

$$x_{ij}(\delta^{in}(v)) - x_{ij}(\delta^{out}(v)) = 0 \qquad \text{for } i < j, v \in V - A \tag{20}$$

$$\sum_{i<j} x_{ij}(\delta^{in}(v)) \leq c(v) \qquad \text{for } v \in V - A \tag{21}$$

$$\sum_{i<l} x_{ij}(\delta^{in}(a_l)) + \sum_{j>l} x_{lj}(\delta^{out}(a_l)) \leq c(a_l) \qquad \text{for } 1 \leq l \leq k, \tag{22}$$

$$x_{ij}(\delta^{in}(R_i)) = x_{ij}(\delta^{out}(R_j)) = 0 \qquad \text{for } i < j, \tag{23}$$

$$x_{ij}(uv) = 0 \qquad \text{for } i < j, u, v \in V - R_i - R_j, uv \in E \tag{24}$$

Thus FRNCMP, the problem to find a maximum fractional node-capacitated region-constrained multiflow, is given by (17)-(24). The half-integral version is denoted by HRNCMP. The idea behind the definition is that, firstly, HRNCMP may be solved in strongly polynomial time, and secondly, our original problem HNCMP reduces to HRNCMP by cleverly specifying the regions. Note that the region-constraint is not monotone in the sense that adding or deleting a node from a region may change a multiflow from one that satisfies the region contraint to one that does not, or vice versa. Also note that the region-constraint is different from the "island-constraint" of Babenko [2], Babenko-Karzanov [1], since there they only require the equivalent of (23). We remark that FRNCMP is strongly polynomial time solvable by Theorem 3.1.

Next we show that HRNCMP is solvable in strongly polynomial time, and in fact show that FRNCMP admits a half-integral optimum, and one can be found in strongly polynomial time. Intuitively, the region-constraint makes the multiflow much easier, since the flows x_{ij} will not get "entangled". Hence one has to specify the pre-flow starting in a terminal inside its region, and then connect these pre-flows by edges between the regions R_i, R_j, and edges from R_i to Q. (A pre-flow is something like a flow, but instead of flow conservation, it has nonnegative flow excess at every node other than its source.) A more detailed investigation of this idea will be used in the proof of the following theorem, claiming a strongly polynomial time algorithm for HRNCMP.

Theorem 5.1. *HRNCMP may be solved in strongly polynomial time.*

Proof. To solve the HRNCMP problem, we consider the polytope defined by (25)-(31) given below.

For all i we introduce directed edges uv and vu for all $uv \in E[R_i]$, and denote the set of these directed edges by E'. Let, moreover, E'' denote the set of edges uv such that $u \in R_i$, $v \in R_j$ for some $i \neq j$; and let E''' denote the set of edges between a node of R and a node of Q. Thus we obtain a mixed graph $G' = (V; E', E'' \cup E''')$, where E' are directed edges and $E'' \cup E'''$ are undirected edges. We consider the above sketched approach, that is we consider pre-flows in $E'[R_i]$ with source a_i that are connected by values assigned with the undirected edges. We denote by $\delta^{un}(v)$ the set of undirected edges incident with v, and denote by $\delta^{in}(v)$, resp. $\delta^{out}(v)$ the set of directed edges entering, resp. leaving v. Let, for $v \in V, U \subseteq V$, $\delta^{un}(v, U)$ denote the set of undirected edges with one endpoint equal to v and the other endpoint in U. We consider the following LP, where $y : E' \cup E'' \cup E''' \to \mathbb{R}$:

$$
\begin{aligned}
y &\geq 0 && && (25)\\
y(\delta^{in}(v)) - y(\delta^{out}(v)) - y(\delta^{un}(v)) &= 0 && \text{for } v \in R, && (26)\\
y(\delta^{in}(v)) &\leq c(v) && \text{for } v \in R - A, && (27)\\
y(\delta^{in}(a_i)) &= 0 && \text{for } 1 \leq i \leq k, && (28)\\
y(\delta^{out}(a_i)) + y(\delta^{un}(a_i)) &\leq c(a_i) && \text{for } 1 \leq i \leq k, && (29)\\
y(\delta^{un}(v)) &\leq 2c(v) && \text{for } v \in Q, && (30)\\
y(\delta^{un}(v, R_i)) - y(\delta^{un}(v, R - R_i)) &\leq 0 && \text{for } v \in Q, 1 \leq i \leq k. && (31)
\end{aligned}
$$

The point in this LP is that $y|_{E'[R_i]}$ are pre-flows starting in a_i, and $y|_{E'' \cup E'''}$ are the connections between these pre-flows. Here (26) is in place instead of the conservation rule, (27), (29) and (30) is for the capacity constraints, and (31) is to make sure that the flow arriving in $v \in Q$ from a_i may be spread over the flows from the remaining a_j's, $j \neq i$.

We claim that the polytope defined by (25)-(31) is half-integral, i.e. all its vertices are half-integral vectors. For this we need some properties of the fractional b-matching polytope of graphs. Recall that fractional b-matchings are given in an undirected graph, where b is a non-negative integer vector on the set of vertices. A fractional b-matching is a non-negative real vector on the set of edges that satisfies the degree-bound given by b, that is, the sum over the set of edges incident with a node is less than or equal to the value of b of that node. An important fact is: ($*$) The polytope defined by these inequalities is known to be half-integral, i.e. all its vertices are half-integer vectors. Another important property is the following: ($**$) If x is a vertex of the fractional b-matching polytope, and $x(uv)$ is not an integer, then the degree-bound associated with u (and v) is tight with respect to x. We will use these propoerties in the proof of the following Claim.

Claim 5.2. *Polytope* (25)-(31) *is half-integral. Moreover, if y is a vertex of the polytope* (25)-(31)*, and $x(qr)$ is not integral for some $q \in Q$, then the inequality* (30) *or* (31) *is tight for $v = q$.*

We prove this claim by showing that every vertex y of the polytope (25)-(31) is equivalent with a vertex of a projection a b-matching polytope. We construct a different b-matching polytope for every vertex y. Formally, the proof goes as follows. Consider a vertex y of polytope (25)-(31). To prove that y is half-integral, we only have to care about an independent system of tight inequalities, and thus obtain a system of equations of which y is the unique solution. We assume that all tight non-negativity constraints are added to our independent system.

Consider a fixed node $q \in Q$. Then (31) may only be tight for at most two i's. Moreover, if it is tight for $i \neq j$, then $y(\delta^{un}(q, R_l)) = 0$ for all $l \neq i, j$, thus the two equations for i, j are not independent, so at most one of the inequalities (31) is in our independent system.

We construct an auxiliary graph $H_0 := (V_0, E_0)$ and $b_0 \in \mathbb{N}^{V_0}$, where we will consider the fractional b_0-matching polytope. We start the construction from the empty graph, and add nodes and edges. For every node $v \in R_i$, we add nodes $v', v'' \in V_0$, add an edge $v'v'' \in E_0$, for every edge $uv \in E[R_i]$, we add edges $u'v'', u'v'' \in E_0$, and define $b_0(v') := b_0(v'') := c(v)$. For an edge $st \in E, s \in R_i, t \in R_j, i \neq j$, we add an edge $s''t'' \in E_0$. For a node $q \in Q$, we consider the two cases depending on whether (31) is tight for some i or not. If (31) is tight for i, then we introduce two nodes $q_1', q_2' \in V_0$, add an edge $q_1'q_2' \in E_0$, add edges $r''q_1' \in E_0$ for $rq \in E, r \in R_i$, add edges $r''q_2' \in E_0$ for $rq \in E, r \in R - R_i$, and define $b_0(q_1') := b_0(q_2') := c(q)$. If (31) is not tight, then we introduce one new node $q' \in V_0$, add edges $r''q' \in E_0$ for $rq \in E, r \in R$, and define $b_0(q') := c(q)$. Thus we have defined $H_0 = (V_0, E_0)$ and $b_0 \in \mathbb{N}_0^V$.

A fractional b_0-matching in H_0 is defined by $y_0(u'v'') := y(uv)$, $y_0(r''q') := y(rq)$, $y_0(r''q_1') := y(rq)$, $y_0(r''q_2') := y(rq)$, and $y_0(v'v'')$ (resp. $y_0(q_1'q_2')$) is defined such that v'' (resp. q_1' and q_2') becomes tight. Thus, all the vertices q_1', q_2' are tight, the vertices v', v'' are tight for $v \in R - A$, and the vertices a_i'' are also tight. If there was another fractional b_0-matching y_0' such that all these inequalities are tight, then that, by reversing this construction, implies another solution y' of the (25)-(31) such that the tight inequalities for y are also tight for y'. Since y is the unique solution of those tight inequalities, another y' may not exist, thus y_0' may not exist, thus y_0 is a vertex of the fractional b_0-matching polytope of H_0. Thus, by property $*$, y_0 is half-integral, impying that y is half-integral. Moreover, by property $**$, we get that if q' is incident with a non-integer value of y_0, then inequality (30) or (31) must be tight. This completes the proof of the Claim.

The relationship of region-constrained multiflows and solutions of this LP is given by the following claim.

Claim 5.3. *Polytope* (25)-(31) *is the projection of the polytope* (18)-(24). *In particular, a region-constrained node-capacitated multiflow x (a solution of* (17)-(24)*) may be converted into a solution y of* (25)-(31) *such that $size(x) = \frac{1}{2}\sum_i y(\delta^{un}(a_i) \cup \delta^{out}(a_i))$, in strongly polynomial time, and vice versa. The conversion maintains (half-)integrality.*

To see that a region-constrained multiflow x with respect to G, A, R_i, c can be converted into a solution $y = y_x$ of the LP (25)-(31). Consider the flow $x_i(uv)$ from a_i to $A - a_i$ that is defined by the sum of flows $x_{ij}(uv)$ $(j < i)$ and reverse flows $x_{ji}(vu)$ $(j < i)$, where $uv \in E$. Then a solution of the LP is given by $y_x(e) := x_i(e)$ for $e \in E'[A_i]$, $y_x(uv) := \sum_{i,j} x_{ij}(\{uv, vu\})$ for $e \in E'' \cup E'''$. To see that the vector y is a solution of the LP (25)-(31), note that $y \geq 0$, (26) follows from the flow conservation for x and the region-constraint, (27), (29), (30) follow from the capacity constraints for x, and (31) follows from the conservation rule and the region constraint implying that every flow contributing to $\delta^{un}(v, A_i))$ will also contribute to $\delta^{un}(v, A - A_i)$. Thus a region-constrained multiflow x may be converted into a solution $y = y_x$ of (25)-(31). Notice, moreover, that $size(x_i) = y_x(\delta^{un}(a_i) \cup \delta^{out}(a_i))$ and thus, $size(x) = \frac{1}{2}\sum_i y_x(\delta^{un}(a_i) \cup \delta^{out}(a_i))$.

We show that, conversely, a solution y of (25)-(31) may be converted into a multiflow $x = x_y$ that satisfies the region- and the capacity-constraints, and the objective values share the same relation as above, that is, $size(x_y) = \frac{1}{2}\sum_i y(\delta^{un}(a_i) \cup \delta^{out}(a_i))$. The construction of $x = x_y$ goes as follows.

Let y be a solution of (25)-(31). Then, for all $1 \leq i \leq k$, let y_i be the pre-flow in R_i starting in a_i defined by $y_i := y|_{E'[R_i]}$. Thus y_i is a pre-flow with source a_i and non-negative flow excess $excess_i(v) := y_i(\delta^{in}(v)) - y_i(\delta^{out}(v)) = y(\delta^{un}(v)) \geq 0$, where $v \in R_i$ (see (26)). These pre-flows will have to be connected by the using the values of y on the undirected edges to make sure that we get a multiflow.

Denote $F_i := \delta^{un}(R_i, V - R_i)$. Then, to distribute the pre-flow y_i among the edges in F_i, we have to decompose pre-flow y_i into flows $y_{i,e}$ for $e \in F_i$, that is, we require that $y_i = \sum_{e \in F_i} y_{i,e}$, and that $y_{i,e}$ is a flow from a_i to r, where $e = rq, r \in R_i, q \in V - R_i$, and $size(y_{i,e}) = y(e)$. The existence and the algorithmic construction of this decomposition follows from the path-decomposition property of flows, and it also implies that if y is (half-)integral, then all flows $y_{i,e}$ may be assumed to be (half-)integral. Let $\overline{y}_{i,e}$ denote the reverse of the flow $y_{i,e}$, that is, $\overline{y}_{i,e}(ab) := y_{i,e}(ba)$ for all $ab \in E$.

Now consider an edge $st \in E''$ such that $s \in R_i, t \in R_j$, say $i < j$. Then let $x_e := y_{i,e} + y(e)\chi_e + \overline{y}_{j,e}$. It is easy to see that x_e is a flow from a_i to a_j. Since y is half-integral, we get that x_e is half-integral.

For a node $q \in Q$, consider the values on the adjacent edges $\delta^{un}(q)$, that is, consider the vector $y|_{\delta^{un}(q)}$. Inequality (31) implies that $y|_{\delta^{un}(q)}$ is a nonnegative combination of the vectors $\chi_{\{e,f\}}$, where $e \in \delta^{un}(q, R_i), f \in \delta^{un}(q, R_j)$ for some $i \neq j$, that is,

$$y|_{\delta^{un}(q)} = \sum_{e \in \delta^{un}(q,R_i), f \in \delta^{un}(q,R_j), i \neq j} \beta_{e,f}\chi_{\{e,f\}} \tag{32}$$

where $\beta_{e,f} \geq 0$ is defined for all $e \in \delta^{un}(q, R_i), f \in \delta^{un}(q, R_j), i \neq j$. By Claim 5.2, these numbers $\beta_{e,f}$ may be chosen half-integral. Then, for every pair e, f of edges such that $e \in \delta^{un}(q, R_i), f \in \delta^{un}(q, R_j), i \neq j$, we define a flow $x_{e,f} := y_e + \beta_{e,f}\chi_e\{e, f\} + \overline{y}_f$.

For $i < j$, we define

$$x_{ij} := \sum_{e \in \delta^{un}(R_i, R_j)} x_e + \sum_{q \in Q} \sum_{e \in \delta^{un}(q,R_i), f \in \delta^{un}(q,R_j)} x_{e,f}. \tag{33}$$

Since $x_e, x_{e,f}$ are flows from a_i to a_j, we get that x_{ij} is a flow from a_i to a_j, too, thus $x_y := x$ is a multiflow. Inequality (27) implies the node-capacity constraint for nodes in $R - A$, (29) (resp. (30)) implies the node-capacity constraint for nodes in A (resp. Q). The equation $size(x_y) = \frac{1}{2}\sum_i y(\delta^{un}(a_i) \cup \delta^{out}(a_i))$ follows from the construction. This proves the Claim.

We solve HRNCMP in strongly polynomial time as follows. By Theorem 3.1 of Frank and Tardos, we find an optimum basic solution of (25)-(31) with respect to the objective max $\frac{1}{2}\sum_i y_x(\delta^{un}(a_i) \cup \delta^{out}(a_i))$. By the second Claim, y is half-integral. By the first Claim, we can construct an optimum half-integral solution $x = x_y$ of FRNCMP, which, since it is half-integral, is an optimum solution of HRNCMP.

6 A strongly polynomial time algorithm for HNCMP

Consider the problem of finding a maximum half-integral multiflow subject to arbitrary node-capacities. The input is the graph $G = (V, E)$, the set $A = \{a_1, \cdots, a_k\} \subseteq V$ of terminals, and the vector $c : V \to \mathbb{N}$ of capacities. The goal is to find a maximum half-integral multiflow $x = \{x_{ij}\}$ subject to G, A, c. Recall that $\nu^*(G, A, c)$ denotes the maximum size of a fractional (or equivalently, half-integral) multiflow, and those multiflows attaining this maximum are called maximum multiflows. A multiflow x is called a *shortest maximum multiflow* if it attains the minimum of $\min\{\mathbf{1} \cdot x : x \text{ is a maximum multiflow}\}$.

The polytope of node-capacitated multiflows has a $0, \pm 1$ description, thus, by Frank and Tardos' Theorem 3.1, one can optimize the linear objective $size(\cdot)$ over the polytope of node-capacitated multiflows, and determine $\nu^*(G, A, c)$. Next, adding the constraint that $size(x) = \nu^*(G, A, c)$, one obtains a $0, \pm 1$ description of the polytope of maximum size node-capacitated multiflows. This implies that, by Theorem 3.1, we can determine the minimum of $\mathbf{1} \cdot x$ over the polytope of maximum size node-capacitated multiflows, i.e. find a shortest maximum multiflow. By adding the constraint that $\mathbf{1} \cdot x$ is equal to the shortest one, we obtain a $0, \pm 1$ description of the polytope of shortest maximum multiflows. Thus we proved the following.

Claim 6.1. *The polytope of shortest maximum fractional node-capacitated multiflows has a $0, \pm 1$ LP description, and can be constructed in strongly polynomial time.*

Further we claim that we may assume without loss of generality that for every terminal a_i there is a shortest maximum multiflow with positive flow from or to a_i. To see that we can make this assumption, note that for all i we can test whether there is a shortest maximum multiflow x such that $\mu_x(a_i) > 0$. Testing this property of a_i is done by maximizing $\mu_x(a_i)$ over the polytope of shortest maximum multiflows by Theorem 3.1. If the property does not hold for a_i, then we delete a_i, which will not change the polytope of shortest maximum multiflows. By repeating this procedure, we may assume that

$$\text{for all } i, \text{ there is a shortest maximum multiflow such that } \mu_x(a_i) > 0. \tag{34}$$

By Therorem 3.1, the maximum of $\mu_{x_{lm}}(v)$ over the polytope of shortest maximum multiflows may be determined in strongly polynomial time. We are interested in testing whether this maximum is positive for some v, l, m. Thus we can construct the following sets S_i in strongly polynomial time:

$$R_i := \{v : v \in V, \text{ there is a shortest maximum multiflow } x \text{ and } i \in \{l,m\} \text{ such that } \mu_{x_{lm}}(v) > 0, \text{ and there is no shortest maximum multiflow } x \text{ and } i \notin \{l,m\} \text{ such that } \mu_{x_{lm}(v)} > 0\}. \tag{35}$$

The above condition (34) implies that $a_i \in S_i$. These sets R_i are the *regions*, which are disjoint, and $a_i \in R_i$ holds for all i.

We claim that the maximum of HNCMP attains with a multiflow satisfying the region-constraints with respect to R_i.

Claim 6.2. *The maximum of HNCMP with respect to G, c, A is equal to the maximum of HRNCMP with respect to G, c, A, R_i.*

Proof. Consider a shortest maximum multiflow x. We want to prove that x satisfies the region-constraints with respect to R_i.

First assume for contradiction that x fails to satisfy constraint (23). By symmetry, assume that $x_{ij}(\delta^{in}(R_i)) > 0$. This implies that there is a path P_1 from a_i to a_j, and a positive ϵ such that $x_{ij} \geq \epsilon\chi_{P_1}$ and $|\delta^{in}(R_i) \cap P_1| > 0$. Say $vz \in P_1$ such that $v \in V - R_i, z \in R_i$. By the definition of R_i, there is a shortest maximum multiflow x', and a path P_2 from a_l to a_m such that $i \notin \{l, m\}$ and $x'_{lm} > \epsilon'\chi_{P_2}$ where $\epsilon' > 0$, and such that $v \in V(P_2)$. Assume, by symmetry, that $j \neq l$. Then another shortest maximum multiflow x'' is defined from $\frac{1}{2}(x+x')$ by replacing $\epsilon''\chi_{P_1}$ and $\epsilon''\chi_{P_2}$ by $\epsilon''\chi_{P_3}$ and $\epsilon''\chi_{P_4}$, where $P_3 := a_iP_1vP_2a_m$ and $P_4 := a_lP_2vP_1a_j$, and $\epsilon'' := \frac{1}{2}\min\{\epsilon, \epsilon'\}$. The construction implies that x' is a shortest maximum multiflow. Note that $v \in V(P_4)$, thus, by definition (35), $v \notin R_i$, a contradiction. Thus x satisfies (23).

Second assume for contradiction that x fails to satisfy (24). Thus there is a shortest maximum multiflow x' and a path P_1 from a_i to a_j and $\epsilon > 0$ such that $x'_{ij} \geq \epsilon\chi_{P_1}$, and there is an edge $vz \in P_1$ such that $v, z \in Q$. By definition, there is a shortest maximum multiflow x', and there are two paths P_2 and P_3 such that P_2 is from a_l to a_m and $v \in V(P_2)$, and P_3 is from a_r to a_s and $z \in V(P_3)$, and $x'_{lm} \geq \epsilon'\chi_{P_2}, x'_{rs} \geq \epsilon'\chi_{P_3}, \epsilon' > 0$, and $i \notin \{l, m\}, j \notin \{r, s\}$. We define a multiflow x'' from $\frac{1}{2}(x+x')$ such that we replace $\epsilon''\chi_{P_1}$, $\epsilon''\chi_{P_2}$, and $\epsilon''\chi_{P_3}$, by $\frac{1}{2}\epsilon''\chi_{P_2}$, $\frac{1}{2}\epsilon''\chi_{P_3}$, $\frac{1}{2}\epsilon''\chi_{P_4}$, $\frac{1}{2}\epsilon''\chi_{P_6}$, $\frac{1}{2}\epsilon''\chi_{P_6}$, and $\frac{1}{2}\epsilon''\chi_{P_7}$, where $P_4 := a_iP_1vP_2a_l, P_5 := a_iP_1vP_2a_m$, $P_6 := a_jP_1vP_3a_r$, $P_7 := a_jP_1vP_3a_s$, and $\epsilon'' := \frac{1}{2}\min\{\epsilon, \epsilon'\}$. Then x' satisfies the capacity constraints, $size(x'') = size(\frac{1}{2}(x + x'))$, and $\mathbf{1} \cdot x'' = \mathbf{1} \cdot \frac{1}{2}(x + x') - \epsilon''$. A contradiction with the fact that x and x' were chosen as shortest maximum multiflows, which proves the Claim.

As a corollary of the above results on region-constrained multiflows, we get that there is a strongly polynomial time algorithm to solve HNCMP.

Theorem 6.3. *There is a strongly polynomial time algorithm to find a maximum half-integral node-capacitated multiflow.*

Proof. The algorithm goes as follows, using the results proved above. Construct the regions R_i defined in (35). By Theorem 5.1, solve the HRNCMP, i.e. find a maximum half-integral region-constrained node-capacitated multiflow x with respect to G, A, c, R_i. This, by Claim 6.2, implies that x is a maximum half-integral node-capacitated multiflow, a solution of HNCMP.

7 A strongly polynomial time algorithm for INCMP

Our goal in this section is to prove that INCMP is solvable in strongly polynomial time. The input for INCMP is a graph $G = (V, E)$, a set $A \subseteq V$, a vector $c \in \mathbb{N}^V$, and the set $Odd \subseteq V$ of nodes $v \in V$ for which $c(v)$ is odd. To see why we need to assume the knowledge of the set Odd, refer to the above section on the parity of input numbers. Note that $c - \chi_{Odd}$ is an even vector. Thus we solve, by Theorem 6.3, HNCMP with respect to the capacity vector $c' := \frac{1}{2}(c - \chi_{Odd})$, and obtain a maximum half-integral multiflow x with respect to c'. Then $2x'$ will be an integral multiflow with respect to capacity c, and since $val(G, c, \mathcal{X}) \leq 2val(G, c', \mathcal{X}) - |V|$ for any A-partition $\mathcal{X}$, we get the following.

Corollary 7.1. *We can construct in strongly polynomial time an integral multiflow subject to G, A, c of size at least $\nu(G, A, c) - |V|$.*

The remaining part of this section is presented in term of integral node-capacitated packing of A-paths, which is a problem equivalent with INCMP, as is shown in an earlier section. By Corollary 7.1, we get a multiflow x of size at least $\nu(G, A, c) - |V|$. Recall that a flow can be decomposed into flows along paths such that the number of paths is at most the number of edges of the graph. The number of flows in x is $\binom{k}{2}$, thus we get the following. This decomposition into paths can be constructed in strongly polynomial time. (Recall that $\mathcal{P}$ denotes the set of A-paths in G.)

Corollary 7.2. *We can construct in strongly polynomial time an integral node-capacitated packing $m : \mathcal{P} \to \mathbb{N}$ subject to G, A, c such that $\mathbf{1} \cdot m \geq \nu(G, A, c) - |V|$, and such that $|supp(m)| \leq \binom{k}{2}|E|$.*

Here we remark that, while we perform operations with packings, we will always be able to maintain a packing of support no bigger than $\binom{k}{2}|E|$. In the algorithm, a packing m, is assumed to be given by specifying the elements of its support, and the values associated with elements of the support. Thus, operations with packings will be performed in strongly polynomial time.

An *augmentation procedure* is a strongly polynomial time algorithm that, given G, A, c and an integral packing m, returns either a bigger integral packing, or a proof of its maximality. Clearly, by starting with the near-maximum packing of Corollary 7.2, and repeating an augmentation procedure at most $|V|$ times, we obtain a maximum packing and a proof of its maximality. Thus we get the following.

Corollary 7.3. *If there is a strongly polynomial time augmentation subroutine for integral node-capacitated packing of A-paths, then there is a strongly polynomial time algorithm for INCMP.*

Hence the remainder of this section is devoted to the construction of an augmentation procedure. That construction is based on the reduction to a "small" instance, that is we show that augmentation of m with respect to G, A, c is equivalent with augmentation of m' with respect to G, A, c', where c' is "small", i.e. $c'(V) \leq poly(|V|)$. This "small" instance can be solved by Lovász' linear matroid matching algorithm in polynomial time. This idea to prove the strongly polynomial time solvabity of INCMP is motivated by Gerards' [8] Proximity Lemma for b-matching, which he used in the construction of a strongly polynomial time algorithm for maximum (weight) b-matching. (He in fact proved a Proximity Lemma for weighted b-matching.) The construction of the equivalent "small" instance is based on a Proximity Lemma for A-paths, and we remark that the proof of this Proximity Lemma is different from Gerards' proof of the Proximity Lemma for b-matching. To prove that Proximity Lemma, we need the following "complementary slackness" type characterization of a pair of an optimum primal and optimum dual solution.

The "complementary slackness" condition is a necessary and sufficient condition for a pair of a primal and dual solution to be optimal. Recall that $\mathcal{P} = \mathcal{P}(A)$ denotes the set of all A-paths, and $\mathcal{P}_v = \mathcal{P}_v(A)$ denotes the set of all A-paths that traverse a node $v \in V$. For packing $m : \mathcal{P}(A) \to \mathbb{N}$, let $c_m(v) := c(v) - m(\mathcal{P}_v)$ denote the remaining capacity of node v, which is a non-negative integer.

Lemma 7.4. *Let $m : \mathcal{P} \to \mathbb{N}$ be an integral packing, and $\mathcal{X}$ an A-partition. Then $\mathbf{1} \cdot m = val(G, c, \mathcal{X})$ if and only if all the following conditions hold:*

1. $c_m(v) = 0$ *for every node* $v \in X_0$,
2. $m(P) = 0$ *for every path such that* $|V(P) \cap X_0| \geq 2$,
3. $c_m(v) \leq 1$ *for every node* $v \in X \cap V(K)$ *in a component* $K \in \mathcal{K}(\mathcal{X})$,
4. $c_m(v) = 0$ *or* $c_m(v') = 0$ *for any pair of nodes* $v \neq v', v, v' \in X \cap V(K)$ *in the same component* $K \in \mathcal{K}(\mathcal{X})$,
5. $m(P) \leq 1$ *for every path such that* $|V(P) \cap V(K) \cap X| \in \{1, 3\}, K \in \mathcal{K}(\mathcal{X})$,
6. $m(P) = 0$ *or* $m(P') = 0$ *for any pair of paths* $P \neq P', |V(P) \cap V(K) \cap X| \in \{1, 3\}, |V(P') \cap V(K) \cap X| \in \{1, 3\}$ *for the same component* $K \in \mathcal{K}(\mathcal{X})$,
7. $m(P) = 0$ *for every path such that* $|V(P) \cap V(K) \cap X| \geq 2$ *and* $|V(P) \cap V(K') \cap X| \geq 2$ *for two distinct components* $K \neq K', K, K' \in \mathcal{K}(\mathcal{X})$,

8. *$m(P) = 0$ for every path such that $v \in V(P)$ for some $v \in X_0$ and $|V(P) \cap V(K) \cap X| \geq 2$ for some $K \in \mathcal{K}(\mathcal{X})$.*

Proof. Let $\mathcal{P}_0$ denote the set of all A-paths such that $X_0 \cap V(P) \neq \emptyset$, and let $\mathcal{P}_1 = \mathcal{P} - \mathcal{P}_0$ denote the set of all A-paths such that $X_0 \cap V(P) = \emptyset$. Moreover, for a component $K \in \mathcal{K}(\mathcal{X})$, we denote by $\mathcal{P}_K$ the set of all A-paths such that $X_0 \cap V(P) = \emptyset$, and $|X \cap V(K) \cap V(P)| \geq 2$. This implies that $\mathcal{P}_1 = \cup \mathcal{P}_K$. Thus we get the following inequalities:

$$
\begin{aligned}
\mathbf{1} \cdot m &= m(\mathcal{P}_0) + m(\mathcal{P}_1) && (36)\\
&\leq c(X_0) + m(\mathcal{P}_1) && (37)\\
&\leq c(X_0) + \sum_{K \in \mathcal{K}(\mathcal{X})} m(\mathcal{P}_K) && (38)\\
&\leq c(X_0) + \sum_{K \in \mathcal{K}(\mathcal{X})} \lfloor \frac{1}{2} c(X \cap V(K)) \rfloor && (39)\\
&= val(G, c, \mathcal{X}). && (40)
\end{aligned}
$$

Now, assuming that $\mathbf{1} \cdot m = val(G, c, \mathcal{X})$, we get that all those inequalities hold with equality. Equality in (37) implies condition 1 and 2. Equality in (38) implies condition 7. Equality in (39) implies that $\sum_{v \in V(K) \cap X} c_m(v) \leq 1$, which implies condition 3 and 4. Since an A-path in $\mathcal{P}_K$ traverses at least two nodes in $V(K) \cap X$, equality in (39) also implies that all the capacity of $V(K) \cap X$ except for a gap of at most one must be saturated by paths that traverse exactly two nodes in $V(K) \cap X$ and no node in X_0, except for at most one path P such that $m(P) = 1$ and $|V(P) \cap X \cap V(K)| \in \{1, 3\}$. This implies condition 5, 6 and 8. Thus we proved all the conditions. Conversely, if we assume that a pair $m, \mathcal{X}$ satisfies the above conditions 1-8, we get that the support of m consists only of paths using exactly one node of X_0, saturating all nodes of X_0, and the remaining paths in the support traverse at most three nodes of a border, and at least two nodes of at most one border, and nearly saturate all borders, thus implying that $\mathbf{1} \cdot m = val(G, c, \mathcal{X})$, which completes the proof.

The idea of the proximity lemma is that, if the size of an integral packing m is not maximum, then to obtain a bigger integral packing, m has to be changed only a by little, i.e. there is a bigger integral packing in a small neighborhood of m. This small neighborhood is defined such that we don't have to decrease a positive entry of m by more than 2, and no entry does have to be increased by more than 2. We define another integral packing m' in a way that reflects the structure of m, that is, an entry of m' is 0, 1, or 2 if and only if that entry of m is 0, 1, or at least 2, respectively. Clearly, m' can be constructed in strongly polynomial time:

$$
m'(P) := \begin{cases} 0 & \text{if } m(P) = 0 \\ 1 & \text{if } m(P) = 1 \\ 2 & \text{if } m(P) \geq 2 \end{cases} \tag{41}
$$

Note that the cardinality of the support of m, and thus of m', is at most $\binom{k}{2}|E|$, thus $\mathbf{1} \cdot m' \leq 2\binom{k}{2}|E|$, i.e. m' is a "small" integral packing.) Now let us we define

$$c'(v) := \min\{m'(\mathcal{P}_v) + 2, c(v) - m(\mathcal{P}_v) + m'(\mathcal{P}_v)\}. \tag{42}$$

Note that $c'(v) \leq m'(\mathcal{P}_v) + 2$ implies that

$$c'(v) \leq 2\binom{k}{2}|E| + 2,$$

i.e. c' is a "small" node-capacity vector. The definition of c' is constructed so that we get the following Proximity Lemma, that has already been sketched in [15], but here we include the full proof for completeness. The lemma basically says that augmentation of m with respect to G, A, c is equivalent with augmentation of m' with respect to G, A, c.

Lemma 7.5 (Proximity Lemma, [15])**.** *Suppose we are given G, A, c and an integral packing m, and we defined c' and m' as above. Then m is a maximum packing with respect to G, A, c if and only if m' is a maximum packing with respect to G, A, c'. In particular, the following two assertions hold:*

1. *If m' is not maximal, then m is not maximal. In particular, if m'' is an integral packing with respect to G, A, c' such that $\mathbf{1} \cdot m'' > \mathbf{1} \cdot m'$, then $m''' := m - m' + m''$ defines an integral packing with respect to G, A, c such that $\mathbf{1} \cdot m''' > \mathbf{1} \cdot m$.*

2. *If m' is maximal, then m is maximal. In particular, if $\mathcal{X}$ is am A-partition such that $\mathbf{1} \cdot m' = val(G, c', \mathcal{X})$, then $\mathbf{1} \cdot m = val(G, c, \mathcal{X})$*

Proof. Since $c'(v) \leq c(v) - m(\mathcal{P}_v) + m'(\mathcal{P}_v)$, we get the following,

$$m'''(\mathcal{P}_v) = m(\mathcal{P}_v) - m'(\mathcal{P}_v) + m''(\mathcal{P}_v) \leq m(\mathcal{P}_v) - m'(\mathcal{P}_v) + c'(v) \leq c(v),$$

which proves the first assertion.

Next assume that $\mathbf{1} \cdot m' = val(G, c', \mathcal{X})$. This implies that conditions 1-8 of Lemma 7.4 hold with respect to G, A, c', m'. Note that all those conditions are concerned only with $m'(P)$ being equal to 0 or at most 1, and with $c_{m'}(P)$ being equal to 0 or at most 1. Note that $m'(P) = 0$ (resp. $m'(P) = 1$) implies that $m(P) = 0$ (resp. $m(P) = 1$), and $c_{m'}(v) = 0$ (resp. $c_{m'}(v) = 1$) implies that $c_m(v) = 0$ (resp. $c_m(v) = 1$). Thus we get that $\mathbf{1} \cdot m = val(G, c, \mathcal{X})$, implying the second assertion.

8 Conclusion

We constructed a strongly polynomial time algorithm for HNCMP (maximum half-integral node-capacitated multiflow), which is summarized as follows. The algorithm first constructs regions R_i, which is done by applying Frank and Tardos' algorithm for $0, \pm 1$ linear programming two times (to determine the polytope of shortest maximum multiflows), and another $\binom{k}{2}|V|$ times (to determine the maximum of $\mu_{x_{lm}}(v)$ over the polytope of shortest maximum multiflows). Then we solve HRNCMP (maximum half-integral region-constrained node-capacitated multiflow) by applying once again Frank and Tardos' algorithm for $0, \pm 1$ linear programming. The multiflow obtained this way is an optimum solution of HNCMP. The algorithm uses $O(k^2|V|)$ times the algorithm of Frank and Tardos based on the ellipsoid method.

We constructed a strongly polynomial time algorithm for INCMP (maximum integral node-capacitated multiflow), which goes as follows. By using the parity of the node-capacities, we round down to the nearest even node-capacities. INCMP can be solved for even node-capacities by applying HNCMP, and thus we obtain a near-optimal integral multiflow. We convert the near-optimal integral multiflow into a near-optimal integral packing m. Near optimality means that we have to augment m at most $|V|$ times to reach optimality. An augmentation subroutine is given by reducing to c', m', and solving INCMP with respect to c', m' with Lovász' linear matroid matching algorithm. The bottleneck in bounding the running time comes from solving HNCMP, where we need $O(k^2|V|)$ times the elipsoid method.

We remark that the Proximity Lemma may also be used to construct a polynomial time algorithm for INCMP via capacity scaling.

References

[1] M. Babenko, A.V. Karzanov, *A scaling algorithm for maximum node-capacitated multiflow problem*, ESA 2008, accepted

[2] M. Babenko, *A Fast Algorithm for Path 2-Packing Problem*, Proceedings of Computer Science - Theory and Applications, Second International Symposium on Computer Science in Russia, CSR 2007, Ekaterinburg, Russia, September 3-7, 2007, [eds. Volker Diekert, Mikhail V. Volkov, Andrei Voronkov] Lecture Notes in Computer Science, Springer Berlin / Heidelberg, LNCS Volume 4649, 70-81.

[3] B.V. Cherkassky, *Mnogopolyusnye dvukhproduktovye zadachi*, [Russian: Multiterminal two commodity problems], in: Issledovaniya po Diskretnoi Optimizatsii [Russian: Studies in discrete optimization] (A.A. Friedman, ed.), Izdatel'stvo "Nauka", Moscow, 1976, 261–289.

[4] M. Chudnovsky, J.F Geelen, B. Gerards, L. Goddyn, M. Lohman, P. Seymour, *Packing non-zero A-paths in group-labeled graphs*, to appear in the Journal of Combinatorial Theory Ser. B.

[5] M. Chudnovsky, W.H. Cunningham, J.F. Geelen, *An algorithm for packing non-zero A-paths in group-labeled graphs*, submitted, (2004)

[6] A. Frank, É. Tardos, *An application of simultaneous diophantine approximation in combinatorial optimization*, Combinatorica **7** (1987) 40-65.

[7] T. Gallai, *Maximum-minimum Sätze und verallgemeinerte Faktoren von Graphen*, Acta Mathematica Academiae Scientarim Hungaricae 12 (1961) 131-137.

[8] A.M.H. Gerards, *Matching*, in: Network Models (M.O. Bal, T.L. Magnanti, C.L. Monma, G.L. Nemhauser, eds.) [Handbooks in Operations Research and Management Science, Vol 7] Elsevier, Amsterdam, 1995

[9] T. Ibaraki, A.V. Karzanov, H. Nagamochi, *A fast algorithm for finding a maximum free multiflow in an inner Eulerian network and some generalizations*, Combinatorica, **18**/1, (January, 1998), 61-83.

[10] J.C.M. Keijsper, R.A. Pendavingh, L. Stougie, *A linear programming formulation of Mader's edge-disjoint paths problem*, J. of Combinatorial Theory Ser. B **96** (2006) 159-163.

[11] L. Lovász, *On some connectivity properties of Eulerian graphs*, Acta Mathematica Academiae Scientarum Hungaricae **28** (1976) 129-138.

[12] L. Lovász, *Selecting independent lines from a family of lines in a space*, Acta Sci. Math. Szeged, **42**(1-2) (1980), 121-131.

[13] W. Mader, *Über die Maximalzahl kreuzungsfreier A-Wege*, Archiv der Mathematik **30** (1978) 325-336.

[14] W. Mader, *Über die Maximalzahl kreuzungsfreier H-Wege*, Archiv der Mathematik (Basel) **30** (1978) 387-402.

[15] G. Pap, *Some New Results on Node-Capacitated Packing of A-Paths*, Proceedings of ACM STOC '07, June 11-13, 2007, San Diego, California, 599-604.

[16] G. Pap, *A constructive approach to matching and its generalizations*, Ph.D. thesis, Eötvös University, Budapest, 2007

[17] G. Pap, *Packing non-returning A-paths*, Combinatorica, Volume 27, Issue 2, (March 2007), 247-251.

[18] G. Pap, *Alternating paths revisited IV: packings and 2-packings of A-paths*, EGRES Technical Report, TR-2005-15

[19] G. Pap, *Packing non-returning A-paths algorithmically*, Discrete Mathematics, Volume 308, Issue 8, (April 2008) 1472-1488.

[20] A. Schrijver, *Combinatorial Optimization*, Springer, 2003, pages 1292, 1459.

[21] A. Sebő, L. Szegő, *The path-packing structure of graphs*, in: Integer programming and combinatorial optimization, D. Bienstock, G. Nemhauser, eds., 256–270, Lecture Notes in Comput. Sci., 3064, Springer, Berlin, 2004.

[22] V.V. Vazirani, *Approximation algorithms*, Springer, Berlin, 2001.