Integer Equal Flows

Carol A. Meyers * Andreas S. Schulz †

Abstract

We examine an NP-hard generalization of the network flow problem known as the integer equal flow problem. The setup is the same as a standard network flow problem, with the added requirement that all arcs in given disjoint sets $R_1, R_2, \ldots, R_\ell$ must have equal flow. We show that this problem is not approximable within a factor of $2^{n(1-\epsilon)}$, for any $\epsilon > 0$, where n is the number of nodes. This result holds even if the cardinality of each arc set is 2. For the variant with a fixed number of arc sets, we show that the problem is solvable in polynomial time.

1 Introduction

The *equal flow problem* was first studied by Sahni [15] as a generalization of the traditional network flow problem. Its setup is similar to a standard maximum flow problem: we are given a graph $G = (N, A)$ with capacities u_a for all $a \in A$, and a designated source node s and sink node t. However, in addition we are also given sets $R_1, R_2, \ldots, R_\ell$ of disjoint groups of arcs, with the requirement that all arcs in the same set must carry the same amount of flow. We wish to send the maximum amount of flow from s to t subject to these constraints. The special case where $\ell = 1$ is known as the *simple equal flow problem*. (We can also define a *minimum cost flow* version, by assigning costs to each of the arcs and a set demand from s to t.)

Ahuja et al. [1] studied the minimum cost simple equal flow problem as a means of modeling a water resource system in Sardinia, Italy. They detailed several different methods of solving the problem, including a version of the network simplex algorithm and a parametric simplex method. More recently, Calvete [4] demonstrated a version of the network simplex algorithm for solving the general minimum cost equal flow problem. While it is possible to solve the general problem in polynomial time using the simplex method, her algorithm exploits the network structure and improves upon the running time.

The special case where all of the arc flows must be integral is known as the *integer equal flow problem*. Sahni [15] proved that the maximum flow version of this problem is NP-hard with a reduction from Non-Tautology. Later, Even, Itai, and Shamir [6] showed via a reduction from Satisfiability that the problem remains NP-hard even if the capacity of each arc is 1. Srinathan et al. [17] showed by a reduction from Exact Cover by 3-Sets that this problem also remains NP-hard if we further require that all arcs in a set R_i must originate from the same node.

Ali, Kennington, and Shetty [2] examined a special case of the integer equal flow problem where each arc set has cardinality 2. We refer to this as the *paired integer equal flow problem*. They developed a heuristic for solving the problem using Lagrangian relaxation and decomposition techniques. Larsson and Liu [11] later proposed a different heuristic algorithm, also based on Lagrangean relaxation.

The integer equal flow problem finds applications in several areas, including airline parts manufacturing [18] and crew scheduling [5, 16]. Feldman and Karger [7] show how the optimal decoding of certain Turbo codes can be accomplished using an integer equal flow problem. Srinathan et al. [17] describe a special case of the problem from supply chain management, where the flow on all arcs exiting a node other than the source is required to be the same. They give an approximation algorithm for the maximum flow version of this problem, which has a performance guarantee that is proportional to the degree of the source node.

*Lawrence Livermore National Laboratory, L-229, 7000 East Avenue, Livermore, CA 94550; meyers14@llnl.gov
†Massachusetts Institute of Technology, E53-361, 77 Massachusetts Avenue, Cambridge, MA 02139; schulz@mit.edu

Other problems that may be modeled as special cases of the integer equal flow problem include balanced network flow problems (see [8]) and certain problems in constraint programming [3]. Parmar [13] describes three variants of the integer equal flow problem arising in packet routing and network design, developing valid inequalities and branch-and-cut schemes. Glockner and Nemhauser [10] describe a dynamic network flow problem with random arc capacities that is also a special case of this problem.

In what follows, we address the approximability of the integer equal flow problem. We begin in Section 2 by presenting an LP formulation of the maximum equal flow problem, along with noting that the integrality gap can be very large. We then observe that the problem of determining whether a nontrivial feasible solution exists to the maximum integer equal flow problem is strongly NP-complete. This motivates our main result in Section 3, which is that no $2^{n(1-\epsilon)}$-approximation algorithm exists for the maximum integer equal flow problem for any fixed $\epsilon > 0$, even if a nontrivial flow is guaranteed to exist.

In Section 4 we extend this argument to show that this also holds for two related problems, the maximum paired integer equal flow problem and the uncapacitated minimum cost equal flow problem. For a special case where the number of sets that must have equal flow is fixed, we observe that this problem is solvable in polynomial time.

2 Problem Definition

An instance of the *maximum equal flow problem* is defined as follows. We are given a directed graph $G = (N, A)$ with special nodes s and t, and capacities $u_a \in \mathbb{Z}$ for all $a \in A$. In addition, we are given disjoint sets $R_1, R_2, \ldots, R_\ell \subseteq A$ of arcs, such that all arcs in the same set must have the same flow. We wish to send the maximum amount of flow from s to t under these conditions. This problem can be formulated as:

$$
\begin{array}{lll}
\max & v & \\
\text{s.t.} & \displaystyle\sum_{j:(s,j)\in A} x_{sj} - \sum_{j:(j,s)\in A} x_{js} = v & \\
& \displaystyle\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0 & \text{for all } i \in N \setminus \{s,t\} \\
& x_{i_1 j_1} = x_{i_2 j_2} & \text{for every pair } (i_1,j_1),(i_2,j_2) \in R_k,\ k = 1,\ldots,\ell \\
& 0 \le x_{ij} \le u_{ij} & \text{for all } (i,j) \in A
\end{array}
$$

The *maximum integer equal flow problem* is the same as above, except we constrain $x_{ij} \in \mathbb{N}$ for all arcs (i, j). This is also known as the *integral flow with homologous arcs* problem [9].

The integrality gap between optimal LP and IP solutions can be very large, even when the cardinality of each homologous set is 2, as in the following example:

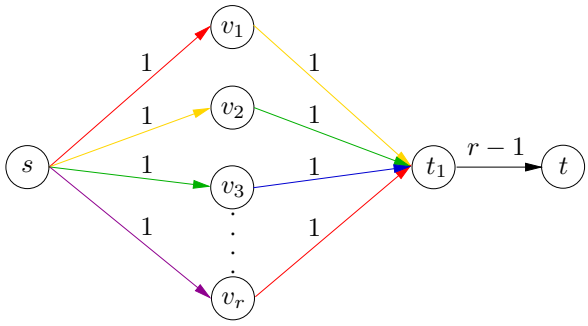

Figure 1: Example of a large gap in optimal LP and IP solutions

Here the homologous arc sets are $\{(s, v_{i+1}), (v_i, t_1)\}$ for $i = 1, \ldots, r-1$, and $\{(s, v_1), (v_r, t_1)\}$. The number on each arc represents its capacity. Note that by the way the homologous sets are constructed, all arcs (s, v_i)

are 'forced' to have equal flow, for all $i = 1, \ldots, r$. The optimal LP solution has a value of $r - 1$, which is achieved by sending $\frac{r-1}{r}$ units of flow along each of the arcs (s, v_i) and (v_i, t_1). The optimal IP solution has a value of 0, since there is no way to send any positive integral amount of flow along this network.

We can equivalently define the *minimum cost integer equal flow problem*, which has the same setup as a traditional minimum cost flow problem, but additionally contains sets $R_1, R_2, \ldots, R_\ell \subseteq A$ of arcs that must have equal flow.

Sahni [15] showed that the maximum integer equal flow problem is NP-hard, as mentioned in Section 1. Later, Garey and Johnson [9] observed that the modification of an Even et al. [6] construction shows that the problem is NP-hard even if the capacity of every arc is 1. Srinathan et al. [17] furthered this, showing that the problem remains NP-hard even if all capacities are 1 and all arcs in a homologous set originate from the same node. It is this construction that motivates several of our results.

3 Hardness of Approximation

Our hardness results are motivated by the following theorem. By 'nontrivial', we mean that some arc in the solution has positive flow (since the zero vector is always feasible). This theorem provides a strengthening of a reduction by Srinathan et al. [17] and shows that it is already strongly NP-hard to decide whether there exists a nontrivial solution.

Theorem 3.1 *The problem of determining whether an instance of the maximum integer equal flow problem has a nontrivial feasible solution is strongly NP-complete.*

Proof: First notice that this problem is in NP, since any nontrivial feasible solution can be taken as a certificate. We reduce from Exact Cover by 3-Sets, which is strongly NP-complete [9]. This problem is:

> **Instance:** A set $\mathcal{A} = \{a_1, \ldots, a_q\}$, such that q is divisible by 3, and a collection $S = \{S_1, \ldots, S_r\}$ of 3-element subsets of $\mathcal{A}$. (Without loss of generality, we can assume that $|\mathcal{A}| = |S|$ [14]; this makes some of our following proofs cleaner although the assumption is not strictly necessary.)
> **Question:** Does there exist a subcollection $S' \subseteq S$ such that each element of $\mathcal{A}$ occurs in exactly one member of S'?

Assume we are given an instance of the Exact Cover by 3-Sets problem, consisting of $\mathcal{A}$ and S. Construct an instance of the maximum integer equal flow problem as follows:

1. Create a source node s and a sink node t. Add q nodes $S_1, S_2, \ldots, S_q$, corresponding to elements of S, and q nodes $a_1, a_2, \ldots, a_q$, corresponding to elements of $\mathcal{A}$.

2. Add arcs: (s, S_i) for all i, of capacity 3.
   (S_i, a_j) if element a_j is contained in set S_i, of capacity 1.
   (a_j, t) for all j, of capacity 1.

3. Add homologous sets $\{(S_i, a_{i_1}), (S_i, a_{i_2}), (S_i, a_{i_3})\}$ for all i, where $S_i = \{a_{i_1}, a_{i_2}, a_{i_3}\}$, and the additional set $\{(a_1, t), (a_2, t), \ldots, (a_q, t)\}$.

For instance $\mathcal{A} = \{a_1, a_2, a_3, a_4, a_5, a_6\}$
$S = \{\{a_1, a_2, a_3\}, \{a_1, a_3, a_4\}, \{a_1, a_3, a_5\}, \{a_2, a_3, a_6\}, \{a_3, a_4, a_5\}, \{a_4, a_5, a_6\}\}$
the constructed graph is shown in Figure 2. Homologous arcs are colored the same, and all unlabeled arcs have capacity 1.

We claim that the answer to the Exact Cover by 3-Sets problem is 'yes' if and only if the maximum integer equal flow on the constructed instance has a nontrivial feasible solution. To see this, first note that if there exists an exact cover $\{S_1', S_2', \ldots, S_{\frac{q}{3}}'\}$, we can achieve a nontrivial feasible solution of value q by sending 3 units along each of the arcs (s, S_i'), and from there through each of the nodes $a_1, \ldots, a_q$ to t. Since each element a_j appears in exactly one of the sets S_i', none of the capacities will be violated.

Conversely, if there exists a nontrivial feasible solution, we claim that the value of the flow must be equal to q. This follows because all of the arcs (a_j, t) have capacity 1, and in a nontrivial solution they must

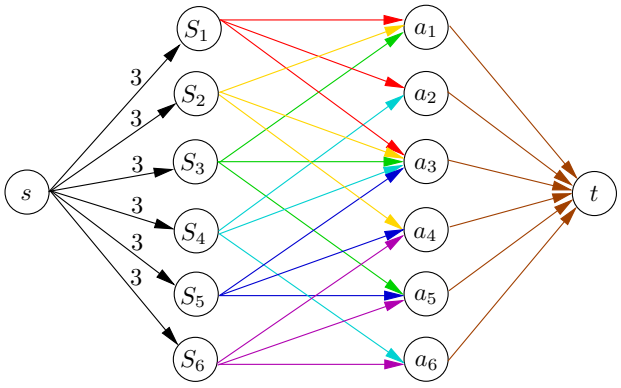

Figure 2: Constructed instance of the maximum integer equal flow problem

all have exactly 1 unit of flow by the homologous conditions. Moreover, by the homologous conditions on the arcs (S_i, a_j), this solution must send flow through exactly $\frac{q}{3}$ of the nodes S_i, and from there on through each of the nodes $a_1, \ldots, a_q$. By construction, this means that the set of nodes $\{S_1', S_2', \ldots, S_{\frac{q}{3}}'\}$ receiving positive flow must correspond to an exact cover by 3-sets. □

We can also apply this result to the case where all arc capacities are 1, by replacing each arc of capacity 3 with three arcs of capacity 1 and adding transshipment nodes as appropriate.

An extension of this argument provides us with our inapproximability result. In essence, we translate the problem of determining whether a nontrivial feasible solution exists into a problem of determining whether a solution of a certain cost exists. We then use the hardness of the first problem to induce a gap in the approximability of the second problem.

Theorem 3.2 *There is no $2^{n(1-\epsilon)}$-approximation algorithm for the maximum integer equal flow problem for any fixed $\epsilon > 0$, even if a nontrivial solution is guaranteed to exist, unless P=NP.*

Proof: Let $\epsilon > 0$ be given. We again reduce from Exact Cover by 3-Sets. Create the same instance of maximum integer equal flow as in the previous proof, and modify it as follows:

1. Let $k = \frac{(2q+3)}{\epsilon}$.
2. Create new nodes $t_1, t_2, \ldots, t_k, t_{k+1}$.
3. Add new arcs: (s, t_i) of capacity $2^{i-1}q$, for $i = 1, \ldots, k$.
   (s, t_{k+1}) of capacity 1.
   (t, t_1) of capacity q.
   (t_{i-1}, t_i) of capacity $2^{i-1}q$, for $i = 2, \ldots, k+1$.
4. Add homologous sets $\{(s, t_1), (t, t_1)\}$ and $\{(t_{i-1}, t_i), (s, t_i)\}$ for $i = 2, \ldots, k$.
5. Redefine the problem so that instead of a maximal $s - t$ flow, we now seek a maximal $s - t_{k+1}$ flow.

For our previous instance
$A = \{a_1, a_2, a_3, a_4, a_5, a_6\}$
$S = \{\{a_1, a_2, a_3\}, \{a_1, a_3, a_4\}, \{a_1, a_3, a_5\}, \{a_2, a_3, a_6\}, \{a_3, a_4, a_5\}, \{a_4, a_5, a_6\}\}$
the graph is as shown in Figure 3 (here, $q = 6$). Homologous arcs are colored the same, and the capacities in the original portion of the graph are unchanged.

By the same argument as in the previous proof, we see that if there is an exact cover by 3-sets, then the value of the maximum integer equal flow is greater than $2^k q$; if there is no exact cover, then the value of the maximum integer equal flow is equal to 1.

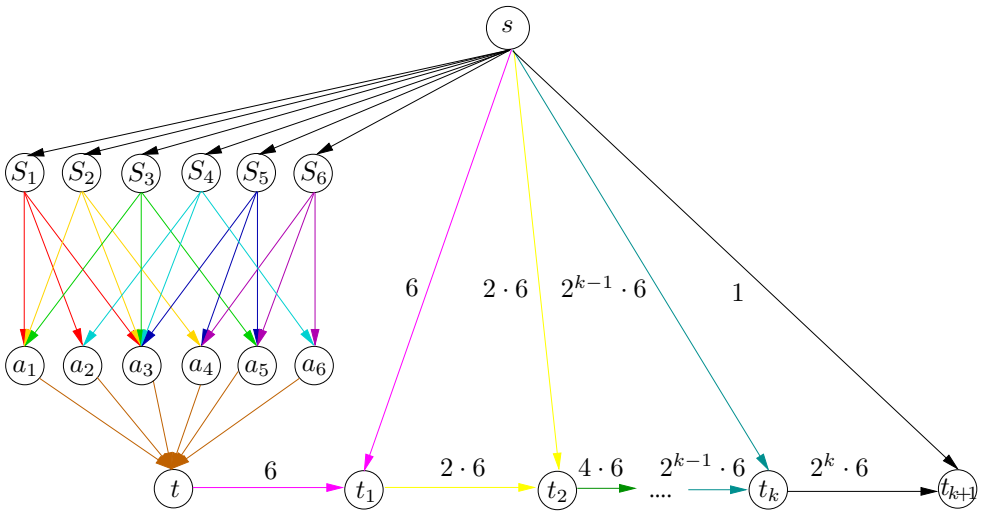

Figure 3: Extended construction of the maximum integer equal flow instance

We also have that $n = 2q + 3 + k = (2q + 3)\frac{1+\epsilon}{\epsilon}$, which implies that $k = n \cdot \frac{1}{1+\epsilon} > n(1 - \epsilon)$. Hence:

There is an exact cover by 3-sets	$\Rightarrow$	value of max integer equal flow is $> 2^{n(1-\epsilon)}$
There is no exact cover by 3-sets	$\Rightarrow$	value of max integer equal flow is 1.

Thus no $2^{n(1-\epsilon)}$ approximation algorithm exists, unless P=NP. □

4 Problem Variants

We now comment on several variants of the integer equal flow problem. The first variant we consider is the *paired* integer equal flow problem, in which all homologous arc sets have cardinality 2.

Theorem 4.1 *There is no $2^{n(1-\epsilon)}$-approximation algorithm for the maximum paired integer equal flow problem for any fixed $\epsilon > 0$, even if a nontrivial solution is guaranteed to exist, unless P=NP.*

Proof: We first claim that any of the homologous sets of size $k \geq 3$ used in the proof of Theorem 3.2 can be converted into collections of homologous sets of size 2, such that the equal flow conditions are still enforced and k new nodes are introduced. To see this, note that the only homologous sets of size greater than 2 used in the proof of Theorem 3.2 are those used in the original Exact Cover by 3-Sets gadget introduced in Theorem 3.1. These have the special structure that all arcs in a homologous set either originate from or end at a common node. A nearly identical construction may be used for both cases, so without loss of generality we consider the case where all homologous arcs end at the same node.

We replace homologous sets of the form $\{(v_1, v_{k+1}), (v_2, v_{k+1}, \ldots, (v_k, v_{k+1})\}$ with the collection of sets of size 2 shown in Figure 4. Here we have introduced k new nodes. The homologous pairs are $\{(v'_i, v_{k+1}), (v_{i+1}, v'_{i+1})\}$ for all $i = 1, \ldots, k-1$, and $\{(v'_k, v_{k+1}), (v_1, v'_1)\}$. We can verify by inspection that the $v_i - v_{k+1}$ flow must be the same for all i in both cases, by the way the homologous pairs are defined, so the constructions are equivalent.

We can now apply the same arguments as in Theorem 3.2 to establish the result. We note that the value of k must be (slightly) increased to compensate for a greater number of nodes in the original graph. □

The next variant we consider is that of the *minimum cost* integer equal flow problem. By extension, the hardness results we have presented thus far also apply to the capacitated version of this problem, since we can transform a maximum flow problem to a minimum cost flow problem using standard network techniques. Hence in what follows we consider the uncapacitated version.

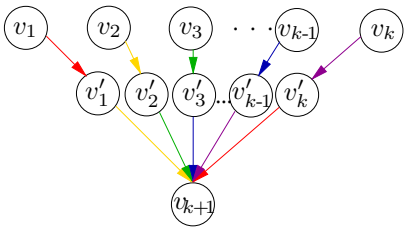

Figure 4: Transformed instance with k sets of size 2

Theorem 4.2 *The uncapacitated single source minimum cost integer equal flow problem is NP-hard, and no $2^{n(1-\epsilon)}$-approximation algorithm exists for any fixed $\epsilon > 0$, even if a nontrivial solution is guaranteed to exist, unless P=NP.*

Proof: We use the same construction as in Theorem 3.2, with the following modifications:

1. Remove the capacities on all of the arcs.
2. Assign the cost of arcs (s, t_{k+1}) and (a_q, t) to be 1. Give all other arcs zero cost.
3. Assign a supply of $2^k q$ units to s, and a demand of $2^k q$ units at t_{k+1}. Give all other nodes zero supply and demand.

We claim that if there is an exact cover by 3-sets, then the cost of the minimum cost integer equal flow is 1; if there is no exact cover, then the cost is greater than 2^k. To see this, first observe if there is an exact cover by 3-sets, then we can route the flow as in Theorem 3.2 and achieve a cost of 1. This is since 1 unit of flow will traverse arc (a_q, t), and all other flow will have a cost of zero.

If there is no exact cover by 3-sets, then there is no way that all of the arcs (a_1, t), (a_2, t), $\ldots$, (a_q, t) can simultaneously contain one unit of flow, by the way the graph is constructed. Moreover, there is no way that these arcs can simultaneously carry *more* than one unit of flow either; as the amount of flow on arc (t_i, t_{i+1}) must be double that of arc (t_{i-1}, t_i), the total amount of required flow would then exceed the available demand in the network. Thus arc (t, t_1) must have zero flow, and by extension all of the arcs (t_i, t_{i+1}) must also have zero flow. The only feasible flow is to send all $2^k q$ units of flow along the arc (s, t_{k+1}), which gives a cost of $2^k q > 2^k$.

Using a very similar anlaysis to that in Theorem 3.2, this implies that the problem is NP-hard and no $2^{n(1-\epsilon)}$-approximation algorithm exists for any $\epsilon > 0$, unless P=NP. □

Using the same techniques as in Theorem 4.1, we also obtain the following result.

Theorem 4.3 *The uncapacitated minimum cost paired integer equal flow problem is NP-hard, and there is no $2^{n(1-\epsilon)}$-approximation algorithm for any fixed $\epsilon > 0$, even if a nontrivial solution is guaranteed to exist, unless P=NP.*

We now address the problem where the number ℓ of homologous arc sets is fixed. Ahuja et al. [1] have shown that this problem is solvable in polynomial time when $\ell = 1$, but they did not address the complexity for greater values of ℓ. Our results are for the minimum cost flow version of the problem, though they also hold for the maximum flow version.

Theorem 4.4 *The minimum cost integer equal flow problem is solvable in polynomial time for any fixed number of homologous arc sets.*

Proof: Suppose the amount of supply at node i is $b(i)$. Let ℓ be fixed, and let y_k be the (common) amount of flow on the arcs in homologous set R_k. Our primary observation is that we can obtain the optimal amount of flow on the arcs in each of the homologous arc sets by solving the following mixed integer program:

$$
\begin{array}{lll}
\min & \displaystyle\sum_{(i,j)\in A} c_{ij}x_{ij} & \\
\text{s.t.} & \displaystyle\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = b(i) & i \in N \\
& x_{ij} = y_k & \text{for all } (i,j) \in R_k,\ k = 1,\ldots,\ell \\
& 0 \le x_{ij} \le u_{ij} & \text{for all } (i,j) \in A \\
& y_k \in \mathbb{Z} & \text{for all } k = 1,\ldots,\ell
\end{array}
$$

Given an optimal solution (x^*, y^*) to this problem, we can obtain an integral solution with the same objective function value as follows. First, we ensure that exactly y_k^* units of flow are sent along the arcs in R_k, using the following network transformation technique: if (i,j) is an arc in set R_k, we decrease the supply at node i by y_k^*, decrease the demand at node j by y_k^*, and set the new capacity of arc (i,j) to 0. Once these transformations have been performed, the resulting problem will be a minimum cost network flow problem on the remaining arcs, which we can then solve to give an integral optimal solution. This solution will have the same cost as the original, because network flow problems with integral data are always guaranteed to possess at least one integral optimal solution.

Hence if we can solve the above mixed integer program in polynomial time, we can solve the minimum cost integer equal flow problem in polynomial time. Since ℓ is fixed, this amounts to solving a mixed integer program with a fixed number of integer variables. Lenstra [12] has shown that such problems are solvable in polynomial time. □

Finally, we note that all of these results extend to a generalization of the equal flow problem known as the factor-α flow problem, first proposed by Larsson and Liu [11]. In this problem, we are given a graph $G = (N, A)$ and disjoint sets $R_1, \ldots, R_\ell$ of arcs. We want to find a flow such that for all $(i_1, j_1), (i_2, j_2) \in R_k$,

$$\frac{1}{\alpha} x_{i_2 j_2} \le x_{i_1 j_1} \le \alpha x_{i_2 j_2}$$

for some given $\alpha \ge 1$. The equal flow problem corresponds to the case when $\alpha = 1$.

The same arguments as in Theorem 3.2 establish that there is no $2^{n(1-\epsilon)}$-approximation algorithm for this problem, and the construction of Theorem 4.1 shows that this holds for the paired version as well. The version with a fixed number of homologous arc sets is solvable in polynomial time, via an extension of the arguments in Theorem 4.4, in which the flow on each arc $x_{ij} \in R_k$ is bounded between a lower bound y_k^{lb} and an upper bound αy_k^{lb}.

5 Conclusions

We have shown in Theorems 3.2 and 4.1 that the integer equal flow problem is not approximable within a factor of $2^{n(1-\epsilon)}$, for any fixed $\epsilon > 0$, even if the cardinality of each arc set is 2. This result holds not only for the maximum flow version of the problem, but also for the uncapacitated minimum cost flow version (Theorem 4.2) as well. The result was motivated by the observation (Theorem 3.1) that determining whether an instance of the problem has a nontrivial solution is NP-complete.

For the case where the number of homologous arc sets is constant, we have shown (Theorem 4.4) that the problem is solvable in polynomial time. This is due to the fact that mixed integer programs with a fixed number of integer variables can be solved efficiently.

We have also shown that the results may be extended to two related problems, the paired integer equal flow problem and the factor-α flow problem. In the first case, we were able to transform integer equal flow instances into paired integer equal flow instances; in the second case, we observed that the similar structure of the factor-α problem allowed us to apply the same constructions.

Acknowledgements. The authors wish to thank Jim Orlin for several helpful discussions, including an observation that led to a strengthening of Theorem 3.2.

References

[1] R. Ahuja, J. Orlin, G. Sechi, and P. Zuddas. Algorithms for the simple equal flow problem. *Management Science*, 45:1440–1455, 1999.

[2] A. Ali, J. Kennington, and B. Shetty. The equal flow problem. *European Journal of Operational Research*, 36:107–115, 1988.

[3] A. Bockmayr, N. Pisaruk, and A. Aggoun. Network flow problems in constraint programming. In *Proceedings of the Seventh International Conference on Principles and Practice of Constraint Programming*, number 2239 in Lecture Notes in Computer Science, pages 196–210, Paphos, Cyprus, 2001. Springer-Verlag.

[4] H. Calvete. Network simplex algorithm for the general equal flow problem. *European Journal of Operational Research*, 150:585–600, 2003.

[5] P. Carraresi and G. Gallo. Network models for vehicle and crew scheduling. *European Journal of Operational Research*, 16:139–151, 1984.

[6] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, 5:691–703, 1976.

[7] J. Feldman and D. Karger. Decoding turbo-like codes via linear programming. In *Proceedings of the Forty-Third Symposium on Foundations of Computer Science*, pages 251–260, Vancouver, Canada, 2002. IEEE Computer Society.

[8] C. Fremuth-Paeger and D. Jungnickel. Balanced network flows 1: A unifying framework for design and analysis of matching algorithms. *Networks*, 33:1–28, 1999.

[9] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, NY, 1979.

[10] G. Glocker and G. Nemhauser. A dynamic network flow problem with uncertain arc capacities: formulation and problem structure. *Operations Research*, 48:233–242, 2000.

[11] T. Larsson and Z. Liu. An efficient Lagrangean relaxation scheme for linear and integer equal flow problems. *Optimization*, 40:247–284, 1997.

[12] H. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983.

[13] A. Parmar. *Integer Programming Approaches for Equal-Split Network Flow Problems*. PhD dissertation, Georgia Institute of Technology, August 2007.

[14] K. Pillaipakkamnatt and V. Raghavan. On the Limits of Proper Learnability of Subclasses of DNF Formulas. *Machine Learning*, 25:237–263, 1996.

[15] S. Sahni. Computationally related problems. *SIAM Journal on Computing*, 3:262–279, 1974.

[16] F. Shepardson and R. Marsten. A Lagrangean relaxation algorithm for the two duty period scheduling problem. *Management Science*, 26:274–281, 1980.

[17] K. Srinathan, P. Goundan, M. Ashwin Kumar, R. Nandakumar, and C. Pandu Rangan. Theory of equal flows in networks. In *Proceedings of the Eighth International Computing and Combinatorics Conference*, number 2387 in Lecture Notes in Computer Science, pages 514–524, Singapore, 2002. Springer-Verlag.

[18] B. Verweij, K. Aardal, and G. Kant. On an integer multicommodity flow problem from the airplane industry. Technical Report UU-CS-1997-38, Utrecht University, Department of Computer Science, 1997.