Theory of Equal-Flows in Networks

K. Srinathan*, Pranava R. Goundan, M.V.N. Ashwin Kumar,
R. Nandakumar**, and C. Pandu Rangan

Department of Computer Science and Engineering, Indian Institute of Technology,
Madras, Chennai - 600036, India
{ksrinath,prg,mvnak}@cs.iitm.ernet.in,rangan@iitm.ernet.in

Abstract. The Maximum-Flow problem is a classical problem in combinatorial optimization and has many practical applications. We introduce a new variant of this well known Maximum-Flow problem, viz., the Maximum-Equal-Flow problem,wherein, for each vertex (other than the source) in the network, the actual flows along the arcs emanating from that vertex are constrained to be equal and integral. Surprisingly, unlike the Maximum-Flow problem that is known to admit a polynomial time solution, we prove that the Maximum-Equal-Flow problem is NP-Hard. Nevertheless, we provide an approximation algorithm for the Maximum-Equal-Flow problem. We develop a new (analogous) theory for Equal-Flows in networks and also illustrate the Maximum-Equal-Flow equivalents of the fundamental results in flow theory.

1 Introduction

Consider the problem of a company manufacturing a product P using certain resources $R_1, R_2, \ldots, R_n$. Each resource is in turn produced by other companies using other resources. Hence, for every product, it is possible to obtain a directed graph (network), wherein each node represents a product and a dircted arc (i, j) represents that the product i requires j as a resource in it's manufacture. Hence, in each such network, the *source* will be the final product P to be made and nature will be th e *sink*, providing all the required basic raw materials. But there may also be supply constraints on the availibility of resources for each of the products, which can be captured as capacity constraints on the flow in each of the corresponding arcs of the network. Such a graph is called a "Bill of Demand" in operations management terminology. The problem of finding the maximum number of units of the product P that a company can produce subject to these supply constraints, given the input requirements for each of the products, can be modeled mathematically as a problem similar to that of the Maximum-Flow in the "Bill of Demand" network, wherein each node distributes it's integral inflow among it's out-neighbours as per a fixed pre-specified ratio. Evidently, this problem cannot be solved using the algorithms for Maximum-flow as they do not guarantee that the out-flows will be distributed as per the ratio. Interestingly,

* Financial support from Infosys Technologies Limited, India, is acknowledged.
** Currently at University of Chicago. E-mail: nanda@cs.uchicago.edu

O.H. Ibarra and L. Zhang (Eds.): COCOON 2002, LNCS 2387, pp. 514–524, 2002.

the above ratio-bound flow problem can be abstracted to the MAXI MUM-EQUAL-FLOW problem[1] in an analogous network. This could be done by applying the "conservation of units" principle to each node. Equivalently, we can use different measurement units to capture the required ratio for each of the out-arcs. Hence, the solution to the MAXIMUM-EQUAL-FLOW problem in the resulting network with capacity constraints in the modified units would give the optimal number of units of the final product P the company can make, subject to the supply constraints of other companies.

Informally, the EQUAL-FLOW problem in a network is a "uniform" view of the classical flows problem in a network, in the sense that in the former, each node distributes its in-flow equally and in integral quantities to all its neighbours unlike the latter, wherein each node is "random" and could distribute its in-flow in any manner that it wished. Hence, in situations like in the above problem of product maximization, where the maintenance of "equanimity" among the neighbours and the integrality constraint is of a higher priority than just a maximization of the overall flow, the "maximum-flow" sought for is in fact the maximum-equal-flow. In this work, we systematically study the problem of EQUAL-FLOWS in networks. We show that the MAXIMUM-EQUAL-FLOW problem is NP-hard, unlike the MAXIMUM-FLOW problem. We give the combinatorial and linear programming structure of this problem and design approximate algorithms for it. Furthermore, we illustrate the maximum-equal-flow equivalents of the fundamental results in flow theory viz., the *decomposition* theorem, the *augmenting path* theorem, and the *max-flow min-cut* theorem.

2 The MAXIMUM-EQUAL-FLOW Problem

We begin by introducing the well-known MAX-FLOW problem. Consider a network $N = (V, A, s, t, c)$ where (V, A) is a digraph with two specified nodes s (the *source*) and t (the *sink*). For each arc (i, j) the capacity $c(i, j)$, a positive integer, is given. A *flow* in N is an assignment of nonnegative integer value $f(i, j) \leq c(i, j)$ to each arc (i, j) such that for each node other than s and t, the sum of the flows of the incoming arcs is equal to the sum of flows along the outgoing arcs. The value of flow f is the sum of flows in the arcs leaving s. The problem of determining the maximum possible flow for s to t, subject to these constraints, is well known as the MAX-FLOW problem. A polynomial time algorithm can be obtained by building upon a fundamental fact about networks, known as the *max-flow min-cut theorem* [2,1]. Now consider the following variation of the MAX-FLOW problem. The flow function is integral and such that for all $i \in V$ and $(i, j_1), (i, j_2) \in A$, $j_1 \neq j_2$, we have $f(i, j_1) = f(i, j_2)$. In other words, the flow emanating out of a vertex is same in all the outgoing arcs (from that vertex, other than the source) and of an integral amount. Surprisingly, this problem is NP-hard (see Section 4) even though a polynomial time algorithm exists for the MAX-FLOW problem.

[1] By EQUAL-FLOW, we mean that each node distributes its in-flow equally to all its neighbours, without violating the integrality constraint of the actual flows.

Definition 1 (Maximum-Equal-Flow).
Instance*: A network $N = (V, A, s, t, c)$, positive integer K.*
Question*: Does there exist an equal-flow of value at least K in the network N?*

The algorithm for the MAX-FLOW problem would not work here. To see why the MAX-FLOW algorithm would not work in our case, let us have a brief look at the algorithm in [4]. If we want to find out if a given flow f is optimal, we have to check if a flow f' of value greater then f exists. If such a flow f' exists, then $\Delta f = f' - f$ is itself a positive flow. But Δf may have arcs with negative flows. This can be viewed as a positive flow in the reverse direction. This is equivalent to saying that Δf is a flow in a derived network $N(f) = (V, A', s, t, c')$, where $A' = \{A - \{(i,j) : f(i,j) = c(i,j)\}\} \cup \{(i,j) : (j,i) \in A, f(j,i) > 0\}$. So, finding if f is optimum is same as deciding if $N(f)$ has a positive flow, very similar to determining if a path exists between s and t. Hence, this can be done in polynomial time. Thus, starting with a network N of zero flow, we can repeatedly augment the flow. When no positive flow exists, we have arrived at the optimal. This algorithm seems to work even for the MAX-EQ-FLOW. But finding just a positive flow Δf is not enough; in fact we need a *positive equal path*. That is, the positive path should be such that *all* its "branches" have positive paths as well. Hence in the worst case, an analogous algorithm may end up potentially examining a super-polynomial number of paths. An even more grave problem is the fact that the non-existence of a positive flow path in the derived network implies that the current flow is the maximum (see Theorem 10). An analogous sufficiency condition may not always hold for positive equal flows (see Theorem 11). In the sequel (see Section 4), we show that the MAX-EQ-FLOW problem is in fact NP-hard. Nevertheless, in line with the efforts of [2], we develop a theory for equal-flows in networks that helps design approximation algorithms for the MAX-EQ-FLOW problem.

3 Equal Flow Theory

In this section, we state and prove the equal-flow equivalents of the classic flow decomposition, augmenting path and the max-flow min-cut theorems.

Observation 1 *For any (arbitrary) given network $N = (V, A, s, t, c)$, there exists a network $N' = (V, A, s, t, c')$ (i.e, a renaming of the capacities of the arcs in N) where $c'(x) = c'(y)$ for any two arcs x and y emanating from the same vertex (that is, $\exists \alpha, \beta \in V$ such that $x = (v, \alpha), y = (v, \beta)$), such that every maximum equal-flow through N is a maximum equal-flow through N'.*

Definition 2. *A network $N = (V, A, s, t, c)$ is said to be an* EQ-NETWORK *if the capacity function $c : A \to Z_+$, is such that for each vertex $v \in V$, all the arcs emanating from v have equal capacities.*

Observation 2 *Any* EQ-NETWORK *$N = (V, A, s, t, c)$ can be more succinctly represented as $N = (V, A, s, t, c')$ where $c' : V \to Z_+$, that is, each vertex can be* *assigned a capacity instead of all its out-going arcs assigned the* same *capacity. Hereafter (due to Observation 1), we will use* Networks *to mean* Eq-Networks*.*

3.1 Feasibility of Equal-Flows

This section involves the determination of answer to the following very basic question: *Does there exist any feasible (non-zero) equal-flow in the given network?* We initially assume the edge capacities to be infinity and answer the question and later study the effects of bounding the edge flows.

Observation 3 *Given a network N, there exists an ordering of the vertices such that $(v_i, v_j) \in A \Rightarrow i < j$ iff N contains no cycles.*

Observation 4 *For any network N without cycles, with infinite edge capacities, there exists a non-zero equal-flow iff there exists a non-zero (classical) flow.*

Proof: Obviously if an equal-flow exists, then a classical flow exists. We provide an incremental construction of an equal-flow if a classical flow exists. From Observation 3, we know that the vertices can be ordered such that no back edges occur. Since classical flow exists, there exists a positive path from the source to the sink. Consider one such positive path, say $\mathcal{P}$. Now construct an equal-fractional-flow as follows. Assign a flow of 1 through the first edge of the path $\mathcal{P}$ and 0 to all the other edges from the source. We give below the transition from i^{th} vertex in the ordering to the $(i+1)^{th}$. Since there are no back edges from the $(i+1)^{th}$ vertex v_{i+1}, if v_{i+1} has a non-zero in-flow, then divide it equally among all its successors and proceed to the next vertex in the ordering. By this procedure, we have a valid equal-fractional-flow, which can be converted to an equal-flow by multiplying the individual flows by a suitable integer. □

Definition 3 (Splits). *We define a split $P \subseteq A$ of a network $N = (V, A, s, t, c)$ to be the set of arcs of the form $(V_1, V \backslash V_1)$ or $(V \backslash V_1, V_1)$ for some $V_1 \subseteq V$.*

Definition 4 (Forbidden Splits). *A split $P \subseteq A$ with arcs of the form $(X, V \backslash X)$ or $(V \backslash X, X)$, $X \subseteq V$, is said to be p-forbidden if the following properties hold, for any path p from s to t in N: (1) X or $V \backslash X$ contains both s and t. (2) All the arcs of P are similarly oriented from the set containing both s and t to its complement set of vertices. (3) For the path p from s to t, there exists a vertex v in the other set (complement to the one containing both s and t), such that there is a path q from s to v, with the first arc of p and q being the same.*

Theorem 5. *For an arbitrary network N with infinite edge capacities, there exists a non-zero equal-flow only if there exists a path p from s to t in N such that N contains no p-forbidden splits.*

Proof: Let there exist a non-zero equal-flow in N. On the contrary, assume that every path p from s to t in N contains a p-forbidden split P such that every arc in P is of the form $(Y, V \backslash Y)$ or $(V \backslash Y, Y)$. Without loss of generality, we can assume that both s and t are in Y. Since P is a forbidden split, by definition there exists a path q from s to $v \in V \backslash Y$ such that first arc of p and q is the same. Let u be the last common vertex between p and q. Since there is a positive all-equal-flow through u, a non-zero flow enters $V \backslash Y$ but cannot reach the sink (condition 2 of Definition 4), contradicting the flow conservation property. □
The existence of an equal-flow through networks with bounded capacities requires the satisfaction of a few more conditions (in addition to the one mentioned in Theorem 5) that will be elaborated in the sequel.

Definition 5. *Given a network $N = (V, A, s, t, c)$, define the set $\mathcal{S} \subseteq A$ to be the set of all arcs that are emanating from the source, that is $\mathcal{S} = \{a \mid a \in A, a = (s, v)$ for some $v \in V\}$. We call the set $\mathcal{S}$ as the set of* source arcs *in N.*

Definition 6. *The* Crossing-Index *and the* Thickness-Index *of an arc $a \in A$ with respect to a source arc $a_s \in \mathcal{S}$ in a network $N = (V, A, s, t, c)$ is defined to be the values of the numerator and the denominator respectively, of the fractional flow (expressed as an irreducible fraction) through the arc a generated (see Subsection 3.2) by an unit flow across a_s, all the arc capacities being infinity. We denote the crossing-index (and the thickness-index) of an arc $a \in A$ with respect to a source arc $a_s \in \mathcal{S}$ by $\mathcal{C}(a, a_s)$ (and $\mathcal{T}(a, a_s)$ respectively).*

Observation 6 [Feasibility Condition] *For an arbitrary network N with infinite edge capacities, there exists a non-zero all-equal-flow iff there exist a source arc a_s and a terminating arc[2] a_t such that $\mathcal{C}(a_t, a_s) > 0$.*

3.2 Feasible Values of Equal-Flows

In this section, we seek to answer to the following: (1)*Does there exist a feasible equal-flow in the given network of a stated magnitude (less than the value of the maximum-equal-flow)?* (2) *What are the characteristics of feasible equal-flows?*

Definition 7. *Basic-Equal-Fractional flow for a given network (if it exists) is the flow generated by assigning a flow of* 1 *to exactly one of the source arcs.*

Observation 7 *Any network contains at most $|\mathcal{S}|$ Basic-Equal-Fractional flows.*

We now give an algorithm to generate a Basic-Equal-Fractional flow w.r.t a source arc a_s, if it exists. Let $V = \{s, v_1, v_2, \ldots, v_n, t\}$ be the vertices of the network. Without loss of generality, we can assume that $a_s = (s, v_1)$. Assign variables f_i to denote the fractional flow in each of the outgoing arcs from the

[2] Terminating arcs are those that are in-coming arcs to the sink in the network.

vertex v_i, $1 \leq i \leq n$. Note that f_1 is a known linear function[3] of the variables $f_1, f_2, \ldots, f_n$ and hence by applying the flow conservation equation at the node v_1, we can eliminate the variable f_1. Similarly at node v_i we can eliminate f_i. Hence if a flow exists, all the variables will be eliminated at v_n, leading to the values of the fractional flows through each of the arcs. Note that the fractional flow (expressed as an irreducible fraction $\frac{n_a}{d_a}$) through an arc a is such that $n_a = \mathcal{C}(a, a_s)$ and $d_a = \mathcal{T}(a, a_s)$. We denote by $\mathcal{B}(a_s)$ the Basic-Equal-Fractional Flow generated by $a_s \in \mathcal{S}$.

Definition 8. *An Atomic-Equal-Flow is an integer valued equal-flow that cannot be expressed as a sum of smaller integer valued equal-flows.*

Lemma 1. *Any Atomic-Equal-Flow is the sum of an integral number of Basic-Equal-Fractional Flows.* □

Definition 9. *The* THICKNESS *of a source arc a_s, $\mathcal{J}(a_s)$, is the least integer such that the flow $\mathcal{J}(a_s) \cdot \mathcal{B}(a_s)$ is integral. Furthermore, the flow $\mathcal{J}(a_s) \cdot \mathcal{B}(a_s)$ is an Atomic-Equal-Flow. And, $\mathcal{J}(a_s) = LCM(\mathcal{T}(a, a_s)), \forall a \in A, \mathcal{T}(a, a_s) \neq 0$.*

3.3 Decomposition

Since any Equal-flow through N is (trivially) a flow through N, the classical decomposition theorem holds for Equal-flows too. In the sequel, we present a more tailor-made decomposition theorem for Equal-flows in networks.

Theorem 8 (Equal-Flow Decomposition). *Any integral equal-flow in a network $N = (V, A, s, t, c)$ can be expressed as a sum of atomic-equal-flows.* □

Theorem 9. *Given m irreducible fractions $\frac{n_1}{d_1}, \frac{n_2}{d_2}, \ldots, \frac{n_m}{d_m}$, and an expression $E = \sum_{i=1}^{m} \lambda_i \frac{n_i}{d_i}$, ($\lambda_i$'s are integers), E is an integer only if λ_i is an integral multiple of* $LCM\left[d_i, \left(\frac{\prod_{i=1}^{m} d_i}{d_i}\right)\right] \Big/ \left(\frac{\prod_{i=1}^{m} d_i}{d_i}\right)$.

PROOF: Note that $E = \sum_{i=1}^{m} \lambda_i \frac{n_i}{d_i} = \sum_{j=1}^{m} \left(n_j \lambda_j \frac{\prod_{i=1}^{m} d_i}{d_j} \right) / \prod_{i=1}^{m} d_i$. We require E to be an integer. We denote $\prod_{i=1}^{m} d_i$ by M. Let $\frac{M}{d_i} \cdot n_i \cdot \lambda_i \equiv b_i \ (mod\ M)$. Now, $n_i \cdot \lambda_i$ is an integer $\Longrightarrow \frac{M}{d_i} \Big| b_i$, $\left(b_i \text{ is an integral multiple of } \frac{M}{d_i}\right)$. E is an integer implies that the sum of b_i's ("remainders") is an integral multiple of M. Let us denote $r_i = \sum_{i=1}^{m} b_i - b_i$. $\Longrightarrow M | (r_i + b_i) \Rightarrow d_i | (r_i + b_i)$. It can be seen that $d_i | r_i$ (since each b_j, $j \neq i$ is a multiple of d_i) $\Longrightarrow d_i | b_i$. Since b_i is a multiple of d_i and $\frac{M}{d_i}$, it is also a multiple of $LCM\left(d_i, \frac{M}{d_i}\right)$. It follows that λ_i is a multiple of $LCM\left(d_i, \frac{M}{d_i}\right) \Big/ \frac{M}{d_i}$. Hence the theorem. □

[3] $\delta_{out}(v_i) \cdot f_i = \sum_{(j,i) \in A} f_j$ where δ_{out} denotes the outdegree.

Corollary 1. *Let $\mathcal{S} = \{a_{s_1}, a_{s_2}, \ldots, a_{s_t}\}$ be the set of source arcs in the network. The actual equal-flow through any source arc a_{s_i} is an integral multiple of*

$$\mathcal{G}(a_{s_i}) = \underset{\forall a \in A}{LCM} \left\{ LCM \left(\mathcal{T}(a, a_{s_j}), \frac{\prod_{j=1}^{t} \mathcal{T}(a, a_{s_j})}{\mathcal{T}(a, a_{s_j})} \right) \Big/ \frac{\prod_{j=1}^{t} \mathcal{T}(a, a_{s_j})}{\mathcal{T}(a, a_{s_j})} \right\} \square$$

3.4 Augmenting

Theorem 10 (Classical). *A flow is maximum iff there is no augmenting path.*

Since the non-existence of an augmenting path is both necessary and sufficient for the flow to be maximum, we have an algorithm to find the maximum flow through networks. Unfortunately, the corresponding theorem for equal-flows holds for maximal flows and not maximum flows (for which only the sufficiency holds). Hence an analogous algorithm would not always work.

Definition 10 (Augmenting Atomic-Equal-Flow). *An atomic-equal-flow $\mathcal{A}$ is said to be augmenting for a given equal-flow $\mathcal{F}$ if the equal-flow $\mathcal{A} + \mathcal{F}$ does not exceed the capacity in any of the arcs in the network.*

Observation 11 (Equal-Flow Augmenting) *Any equal-flow F is maximum only if there is no augmenting atomic-equal-flow.*

3.5 Max-Equal-Flow Min-Equal-Cut

In this section, we provide a theorem similar to the max-flow min-cut theorem for classical flows through networks. The results of this section, though not as elegant as their classical flow counterparts, help us design approximate algorithms for the maximum-equal-flow problem which is shown to be NP-hard in the sequel (Section 4). From Lemma 1 and Observation 8, we know that every arc $a \in A$ in N with the set of source arcs $\{a_{s_1}, \ldots, a_{s_t}\}$, is associated with a $t-tuple$, $\frac{n_1}{d_1}, \frac{n_2}{d_2}, \ldots, \frac{n_t}{d_t}$, of fractions[4] such that if $\lambda_1, \ldots, \lambda_t$ are the flows in each of the t source arcs respectively, then the flow through the arc a is $\sum_{i=1}^{t} \lambda_i \cdot \frac{n_i}{d_i}$. We use this fact to define the capacity of an equal-cut in the network.

Definition 11 (Equal-Cut). *An equal-cut is a split C where the arcs in C are of the form $V_1, V \backslash V_1$ or $V \backslash V_1, V_1$ and $s \in V_1$ and $t \in V \backslash V_1$.*

Definition 12 (Capability of an Equal-Cut). *The flow L_a through an arc $a \in A$ is a linear combination of the flows (λ_i's) through the source arcs. The flow is negative if the arc is into V_1, else it is positive. The flow through the equal-cut is the sum of the flows through all the arcs in the equal-cut, which is also a linear combination L_c of the λ_i's. The capability of an equal-cut is defined as the maximum value that L_c can take subject to each L_a being less than its capacity $c(a)$. Surprisingly, we will prove that any equal-cut has the same capability.*

[4] In fact, $n_i = \mathcal{C}(a, a_{s_i})$ and $d_i = \mathcal{T}(a, a_{s_i})$.

Theorem 12. *The Maximum-equal-flow through the network is equal to the capability of any equal-cut.*

Proof: The max-equal-flow is less than or equal to any equal-cut. Since, otherwise the actual flow of at least one arc is greater than its capacity. Furthermore, the max-equal-flow is greater than or equal to the capability of any equal-cut, because the flow through any equal-cut defines a valid equal-flow in the sense that it does not violate the capacity constraints of any arc in the network. $\square$

Corollary 2. *All equal-cuts have the same capability and this is equal to the maximum-equal-flow through the network.*

4 Max-Eq-Flow is NP-Hard

We give a polynomial reduction of the known NP-complete problem, viz. Exact Cover By 3-Sets [3] to that of the Max-Eq-Flow problem.

Definition 13. Exact Cover By 3-Sets (X3C)
Instance *: Set X with $|X| = 3q$ and a collection C of 3-element subsets of X.*
Question *: Does C contain an exact cover for X, i.e., a subcollection $C' \subseteq C$ such that every element of X occurs in exactly one member of C'?*

Theorem 13. Max-Eq-Flow *is NP-Hard.*

Proof : We construct an instance of the Max-Eq-Flow problem from the given instance X (such that $|X| = 3q$) and C of the *Exact Cover by 3-Sets* problem as follows. Consider the network $N = (V, A, s, t, c)$, where $V = L \cup R \cup \{s, t\}$, where $|L| = |C|$, $|R| = |X|$ and there exist *bijective mappings* $h_l : L \to C$ and $h_r : R \to X$. The set of arcs is defined as $A = \{(\ell, r) \mid \ell \in L, r \in R, h_r(r) \in h_\ell(\ell)\} \cup \{(s, \ell) \mid \forall \ell \in L\} \cup \{(r, t) \mid \forall r \in R\}$. The capacity function $c : A \to Z$, where Z is the set of positive integers, is defined as $c(a) = \begin{cases} 3 \text{ if } a = (s, \ell) \in A, \text{ for some } \ell \in L. \\ 1 \text{ otherwise.} \end{cases}$.
Now the instance of the Max-Eq-Flow problem is: $N = (V, A, s, t, c)$, $|X|$, that is: *Is there an equal-flow of value $\geq |X|$ in N?*
Claim: If there exists an equal-flow of size $|X|$ in N, then there exists a $C' \subseteq C$ such that every element of X occurs in exactly one member of C'.
Proof: From the construction of the network N, since each vertex in R has an out-flow capacity of at most 1 (to the sink t), it is evident that exactly $3q$ vertices in R should be involved with non-zero actual equal-flows to produce an overall equal-flow of $3q$. Let $r_1, r_2, \ldots, r_{3q}$ be $3q$ vertices in R such that each of the arcs (r_i, t), $1 \leq i \leq 3q$ has a flow of 1 through it. Moreover, since the out-flow from each of the above $3q$ vertices is 1, it is essential that their in-flow was 1 too. Therefore, for each vertex r_i,$1 \leq i \leq 3q$, we have exactly one vertex $\ell_i \in L$ such that the actual flow in the arc (ℓ_i, r_i) is equal to 1. Also, there are exactly[5] q such vertices $\ell_1, \ell_2, \ldots, \ell_q$. We will now show that the sets $h_\ell(\ell_1), h_\ell(\ell_2), \ldots, h_\ell(\ell_q)$ are

[5] Due to the equal-flow constraint at ℓ_i $\forall i$

mutually disjoint sets in C and hence a solution to the given X3C problem. On the contrary, assume that there exists an element $r \in R$, such that $h_r(r) \in h_\ell(\ell_i)$ and $h_r(r) \in h_\ell(\ell_j)$, for some i, j, $1 \leq i, j \leq 3q$, $i \neq j$. This means that A contains the two arcs a_i and a_j, namely, $a_i = (\ell_i, r) \in A$ and $a_j = (\ell_j, r) \in A$. Since the actual flow is constrained to be an equal-flow of value $3q$, and it is already known that there exist arcs (ℓ_i, r_i) and (ℓ_j, r_j) with actual flows of 1 in them, it is necessary that the actual flows in the arcs a_i and a_j are respectively equal to 1 too (otherwise, it violates the equal-flow property at the nodes ℓ_i and ℓ_j respectively). But then the actual in-flow into the vertex r now stands at least 2 which contradicts the fact that the actual in-flow of any node $r \in R$ is ≤ 1.
Claim: If there exists no equal-flow of size $3q$ in N, then there exists no $C' \subseteq C$ such that every element of X occurs in exactly one member of C'.
Proof: On the contrary, let there exist q mutually disjoint sets, $c_1, c_2, \ldots, c_q$ in C. Then, in the network $N = (V, A, s, t, c)$ constructed as illustrated earlier, there exist q vertices $S = \{\ell_1, \ell_2, \ldots, \ell_q\}$, $S \subseteq L$ such that $h_\ell(\ell_i) = c_i$, $1 \leq i \leq q$. Let the actual flows $f : A \to Z$, where Z is the set of positive integers, to be:

$$f(a) = \begin{cases} 3 \text{ if } a = (s, \ell_i) \in A, \ell_i \in S, 1 \leq i \leq q. \\ 1 \text{ if } a = (\ell_i, r), \ell_i \in S \text{ for some } r \in R. \\ 1 \text{ if } a = (r, t), r \in R, \text{ such that } f((\ell, r)) = 1 \text{ for some } \ell \in S. \\ 0 \text{ otherwise.} \end{cases}$$

Since for any $r \in R$, there can exist at most one $\ell \in S$ such that $f((\ell, r)) = 1$ (otherwise $c_1, c_2, \ldots, c_q$ cannot be mutually disjoint sets), it is clear that the above assignment gives an actual equal-flow of value $3q$. □

5 Approximating the Max-Eq-Flow

We know that every flow can be expressed as the sum of atomic-equal-flows which themselves can be expressed as the sum of basic-equal-fractional-flows. In other words, any equal-flow through the network can be generated with the knowledge of the flows in the source arcs. We provide an elegant methodology to arrive at the flows in the source arcs that almost maximize the flow through the network, using certain combinatorial observations and linear programming.

5.1 Integer Linear Programming Formulation

From Theorem 1, it is clear that the flow through a source arc a_{s_i} can be expressed as a integral multiple of $\mathcal{G}(a_{s_i})$. So, let the flow through a_{s_i} be $x_i \cdot k_i$. Moreover, the flow through each arc $a_j \in A$ is expressible as a linear combination of the x_i's (see Subsection 3.5), say, $\sum_{a_{s_i} \in \mathcal{S}} k_{ji}.x_i$. Therefore, the maximum equal-flow problem is to: $Maximize$ $\sum_{a_{s_i} \in \mathcal{S}} k_i x_i$ Subject to $\sum_{a_{s_i} \in \mathcal{S}} k_{ji}.x_i \leq c(a_j) \forall a_j \in A$ where the x_i's are integers and the k_i's and k_{ji}'s are known precomputable constants.

5.2 Algorithm

Since in the worst case, solving the above ILP is computationally hard, we relax the condition of the x_i's being integers, and solve the resulting LP. We proceed to construct an all-integer based solution set by studying a sufficient condition on the actual flows through the source arcs in N to be integers.

Theorem 14. *It is sufficient to use*

$$k_i = \mathcal{K}(a_{s_i}) = \underset{\forall a \in A}{LCM} \left\{ \frac{LCM\left(\mathcal{T}(a, a_{s_j}), \frac{\prod_{j=1}^{t} \mathcal{T}(a,a_{s_j})}{\mathcal{T}(a,a_{s_j})}\right)}{\frac{\prod_{j=1}^{t} \mathcal{T}(a,a_{s_j})}{\mathcal{T}(a,a_{s_j})}} \times (1 + \Psi(a,i)) \right\}$$

in the integer linear programming formulation above, so that the solution to the ILP gives the maximum equal-flow in the corresponding network, where $\Psi(a,i)$ is as defined in the proof below.

PROOF: From Theorem 9 we know that given m irreducible fractions $\frac{n_1}{d_1}, \ldots, \frac{n_m}{d_m}$, and an expression $E = \sum_{i=1}^{m} \lambda_i \frac{n_i}{d_i}$, ($\lambda_i$'s are integers), E is an integer only if λ_i is an integral multiple of $g_i = LCM\left[d_i, \left(\frac{\prod_{i=1}^{m} d_i}{d_i}\right)\right] \Big/ \left(\frac{\prod_{i=1}^{m} d_i}{d_i}\right)$. But the above condition is not sufficient for E to be an integer. We give a procedure to find a sufficient condition for E to be an integer. First compute the g_i's for all $1 \le i \le m$. Get a modified set of fractions, viz. $\frac{n_1 g_1}{d_1}, \frac{n_2 g_2}{d_2}, \ldots, \frac{n_m g_m}{d_m}$ and express them as irreducible fractions[6], say, $\frac{n_1}{r_1}, \frac{n_2}{r_2}, \ldots, \frac{n_m}{r_m}$. Note that in general, not all the r_i's need to be 1 (in which case E is an integer). Let $I^{(i)} = \{i_1, i_2, \ldots, i_q\}$ be the set of all the indices such that $r_{i_1} = r_{i_2} = \cdots = r_{i_q} = r_i$. Hence, it is necessary that $\mathcal{N}_i = \sum_{j=1}^{q} n_{i_j}$ be a multiple of r_i. Let $\mathcal{N}_i = w r_i + s$ for some non-negative integers $w, s < r_i$. Let g be the least positive integer such that g can be expressed as a linear combination of the n_{i_j}'s with non-negative integer coefficients $\alpha_{i_1}, \ldots, \alpha_{i_q}$ such that $s + g$ is a multiple of r_i. Therefore, setting $\lambda_i = g_i(1 + \alpha_i)$ guarantees that E is an integer. In the case of the max-equal-flow in networks, define $\Psi(a,i) = \alpha_i$ when using for each arc in A, $n_i = \mathcal{C}(a, a_{s_i})$ and $d_i = \mathcal{T}(a, a_{s_i})$. The theorem follows. □

Note that the value of m^{lp} (the maximum equal-fractional-flow) is unaltered by the changing of the k_i's. Using Theorem 14 results in the solution set for the flow through each source arcs to be $x_i^{suf} = \left\lfloor \frac{x_i^{lp}}{\mathcal{K}(a_{s_i})} \right\rfloor \mathcal{K}(a_{s_i})$. The total flow in this case is $m^{suf} = \left(\sum_{i=1}^{t} \left\lfloor \frac{x_i^{lp}}{\mathcal{K}(a_{s_i})} \right\rfloor \mathcal{K}(a_{s_i})\right) \le m^{lp}$. From Theorem 14, it is clear that the feasible solution set of x_i^{suf}s (certainly) results in an integer equal-flow.

Theorem 15. *Given n positive integers $\lambda_1, \ldots, \lambda_n$ and n real numbers $r_1, \ldots, r_n$ such that* $(\min_{i=1}^{n} r_i) > 0$ *and* $(\max_{i=1}^{n} r_i) \ge 1$, $\frac{\sum_{i=1}^{n} \lambda_i \lfloor r_i \rfloor}{\sum_{i=1}^{n} \lambda_i r_i} > \frac{\lambda_{min}}{(n+1)\lambda_{max}}$, *where* $\lambda_{max} = \max_{i=1}^{n} \lambda_i$ *and* $\lambda_{min} = \min_{i=1}^{n} \lambda_i$.

[6] Note that the numerator remains the same.

Proof: Left Hand Side =

$$\frac{\sum_{i=1}^{n}\lambda_i\lfloor r_i\rfloor}{\sum_{i=1}^{n}\lambda_i r_i} \geq \frac{\sum_{i=1}^{n}\lambda_{min}\lfloor r_i\rfloor}{\sum_{i=1}^{n}\lambda_{max} r_i} \geq \frac{\lambda_{min}}{\lambda_{max}}\frac{\sum_{i=1}^{n}\lfloor r_i\rfloor}{\sum_{i=1}^{n} r_i} \geq \frac{\lambda_{min}}{\lambda_{max}}\left(1-\frac{\sum_{i=1}^{n} frac(r_i)}{\sum_{i=1}^{n} r_i}\right)$$

where $frac(r_i) = r_i - \lfloor r_i\rfloor$. Now, let $\sum_{i=1}^{n} frac(r_i) = q$. This, along with the fact that $(\max_{i=1}^{n} r_i) \geq 1$ would imply that $\sum_{i=1}^{n} r_i \geq q+1$. Therefore, $\frac{\lambda_{min}}{\lambda_{max}}\left(1-\frac{\sum_{i=1}^{n} frac(r_i)}{\sum_{i=1}^{n} r_i}\right) \geq \frac{\lambda_{min}}{\lambda_{max}}\left(1-\frac{q}{q+1}\right) \geq \frac{\lambda_{min}}{(q+1)\lambda_{max}}$. Since $q < n$, the theorem follows. ■

Corollary 3. *The above algorithm outputs a flow* m^{suf} *such that* $\frac{m^*}{m^{suf}} < (t+1)\frac{\max_{i=1}^{t}\mathcal{K}(a_{s_i})}{\min_{i=1}^{t}\mathcal{K}(a_{s_i})}$, *where* m^* *is the maximum-equal-flow, since* $m^{lp} \geq m^*$.

6 Conclusion

The problem of Equal-Flow in a network is of immense theoretical and practical importance. In the absence of integrality constraints, this problem can be solved efficiently. However, there are practical problems (like the "Bill of Demand" problem discussed earlier) where the integrality constraint cannot be dispensed with. Surprisingly, integral Maximum-Equal-Flow, unlike integral Maximum-flow turns out to be NP-Hard. Nevertheless, we provide an approximation algorithm, whose quality is proportional to the degree of the source node, to solve the problem.

References

1. P. Elias, A. Feinstein, and C. E. Shannon. Note on maximum flow through a network. *IRE Trans. Information Theory, IT-2*, 1956.
2. L. R. Ford and D. R. Fulkerson. *Flows in Networks.* Princeton University Press, Princeton, N. J., 1962.
3. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.
4. C. H. Papadimitriou. *Computational Complexity.* Addison-Wesley Publiction Company, 1994.