# INTEGER NETWORK SYNTHESIS PROBLEM FOR HOP CONSTRAINED FLOWS

SANTOSH N. KABADI AND K.P.K. NAIR

Abstract. Hop constraint is associated with modern communication network flows. We consider the problem of designing an optimal undirected network with integer-valued edge-capacities that meets a given set of single-commodity, hop-constrained network flow value requirements. We present a strongly polynomial, combinatorial algorithm for the problem with value of hop-parameter equal to three when values of flow requirements are sufficiently large.

## 1. Introduction

In network flows, both analysis and synthesis (network design) problems are of great theoretical and practical significance, and considerable amount of results exist in both these areas [1, 7, 19, 23, 25]. The thrust of analysis is to reveal operational features of a given network while the focus of synthesis is to design an optimal network to meet stated requirements.

In the case of regular network flows without any additional constraints [7], significant literature exists on analysis and synthesis [1, 4, 10, 13, 14, 16, 19, 23, 25, 26]. Recently, there has been an interest in flows subject to additional constraints such as multipath flows [2, 6, 3, 12, 17, 18] and constraints on lengths of paths carrying flow, (these are popularly known as *hop constraints*) [5, 8, 11, 15, 20, 22]. The concept of hop-constrained flow is of relatively recent origin and is associated with communications networks which may include wavelength division multiplexing (WDM) [2]. In general, the smaller the maximum number of hops (edges) permitted on transmission paths, the higher is the quality of transmission. No doubt, with higher quality the sum of network edge capacities, and therefore the total edge cost of the network, will be higher.

In this paper, we consider the synthesis problem corresponding to single commodity hop-constrained network flows. We confine to minimizing the unweighted sum of edge capacities for meeting a given set of values of minimum flow requirements in an undirected network. In this case, a strongly polynomial algorithm for the continuous case is obtained by Gibbens and Kelley [8]. We consider the integer version of the problem. For the case of hop-parameter equal to three, we present a strongly polynomial, combinatorial algorithm when the values of flow requirements are sufficiently large.

In Section 2, we present notations, basic definitions, and statement of the problem. In Section 3 we present our result.

## 2. Notations, Basic Definitions and Statement of the Problem

Throughout, we consider only undirected graphs. We denote by $G = [N, E, u]$ an undirected, edge-capacitated network with node set $N = \{1, 2, \ldots, n\}$, edge set $E$, and edge-capacity function $u$ which assigns a non-negative capacity $u_{ij}$ to each edge $(i, j) \in E$. We denote by $K_n = [N, E_n]$ a complete, undirected graph on node set $N$. Throughout, we shall mean by a path, a simple path [1, 23]. We denote any path $p$ by its ordered sequence of nodes $(i_0, i_1, \ldots, i_k)$ and we denote by

*Key words and phrases.* Network Synthesis, Hop Constraints, Telecommunications .

The work is supported in part by research grants from the Natural Sciences and Engineering Research Council of Canada to both the authors.

$E^p$ the set of all the edges on the path $p$. For any pair of distinct nodes $s, t \in N$, and any positive integer $h$, (which we call the *hop-parameter*), we denote by $P^h_{st}$ the set of all paths $p$ from $s$ to $t$ in $G$ with $|E^p| \leq h$.

We consider the following **Hop-Constrained Network Synthesis Problem**: We are given two positive integers $h$ (the hop parameter), and $n$ (number of nodes), and a symmetric, nonnegative $n \times n$ matrix $R = \{r_{ij}\}$ with $r_{ii} = 0$ for each $i \in \{1, 2, \ldots, n\}$. Each $r_{ij}$ represents the minimum requirement of the value of $h$-hop-constrained flow from node $i$ to node $j$, (that is, flow from node $i$ to node $j$ using only paths in $P^h_{ij}$ such that the total flow on each edge is no more than its capacity). We wish to assign non-negative capacities $\{u_{ij} : (i, j) \in E_n\}$ to the edges of the complete graph $K_n = [N, E_n]$ so that each requirement $r_{ij}$ can be realized, non-simultaneously, and the sum of all edge capacities is minimum. In the integer version of the problem, we assume that $R$ is an integer matrix and require that all the edge capacities be integers.

For $h = 1$, the problem is trivial. (An optimal solution to the problem is $\{u_{ij} = r_{ij}\ \forall (i, j) \in E_n\}$.) For $h = n - 1$, strongly polynomial schemes for the continuous version are given in [21, 9], and strongly polynomial schemes for the integer version are obtained in [4, 26], (see also [16, 25]). In particular, Gomory and Hu [9] proved the following:

**Theorem 1.** *Let $\pi_i = \max_{j \neq i} r_{ij}\ \forall i \in N$. Then for $h = n - 1$, optimal objective function value for the continuous version of the synthesis problem is $\frac{1}{2}\sum_{i=1}^{n} \pi_i$.*

Obviously, for any given requirement matrix $R$, the optimal objective function value of each of the continuous and integer versions of the synthesis problem is a non-increasing function of $h$. Also, for any given $R$ and $h$, the optimal objective function value of the integer version of the problem is no smaller than that of the continuous version of the problem. Gibbens and Kelley [8] presented an algorithm for the continuous version of the problem with $h = 2$ that produces a feasible solution with objective function value equal to $\frac{1}{2}\sum_{i=1}^{n} \pi_i$. (Here, $\pi_i$'s are as defined in Theorem 1.) It follows from Theorem 1 that this solution is optimal for the continuous version of the problem with any $h \geq 2$. From this and Theorem 1, we also have the following.

**Lemma 1.** Let $\pi_i = \max_{j \neq i} r_{ij}\ \forall i \in N$. Then for any $h \geq 2$, $\frac{1}{2}\sum_{i=1}^{n} \pi_i$ is a lower bound on the optimal objective function value for the integer version of the synthesis problem.

The integer version of the case $h = 2$ is open and we conjecture that it is NP-hard. A non-trivial lower bound on the objective function value for this case when all the flow requirements are equal is given in [15].

In the next section, we consider the integer version of the problem with $R$ an integer matrix and $h = 3$. We present a strongly polynomial, combinatorial algorithm that produces an optimal solution to the problem when $\pi_i \geq n - 2\ \forall\ i \in N$. A weaker result for the case $\pi_i \geq n - 1\ \forall\ i \in N$ appeared in [15].

## 3. Results for Hop Parameter $h = 3$

In this section we assume that $R$ is an integer matrix and consider the integer version of the problem with $h = 3$. We give a strongly polynomial, combinatorial algorithm, (Algorithm *Hop-Synthesis*), that produces an optimal solution to the problem when $\pi_i \geq n - 2\ \forall\ i \in N$. Since the $(n - 1)$-hop case is well solved [4, 16, 25, 26]), we assume that $n > 4$.

Before giving a formal description of our algorithm, we briefly explain the main idea. This will facilitate understanding of the algorithm and the proof of its validity. Our algorithm outputs a network $G^* = [N, E_n, u^*]$ such that for any $i, j \in N$, we can send a 3-hop flow of value $\min\{\pi_i, \pi_j\} \geq r_{ij}$ between $i$ and $j$ in $G^*$ and $\sum\{u^*_e : e \in E\} = \lceil \frac{1}{2}\sum_{i=1}^{n} \pi_i \rceil$. It may be noted that it follows from Theorem 1 that such a network is optimum.

We first re-order the nodes such that $\pi_1 = \pi_2 \geq \pi_3 \geq \cdots \geq \pi_n$ and set $\pi^{(1)} = \pi$. For $v = 1, 2, \ldots,$ the algorithm successively peels off from the vector $\pi^{(v)}$, a non-negative vector $\Delta\pi^v$ such that the residual $\pi$-vector ($=\pi^{(v+1)} = \pi^{(v)} - \Delta\pi^v$) is non-negative with $\pi_1^{(v+1)} \geq \pi_2^{(v+1)} \geq \pi_3^{(v+1)} \geq \cdots \geq \pi_n^{(v+1)}$. We choose $\Delta\pi^1 = (n-2)p_1(1, 1, \ldots, 1)$, with $p_1 = \lfloor \frac{\pi_n}{(n-2)} \rfloor$. For each $v$, a network $G^v$ with edge capacities $\Delta u_{ij}^{(v)}$ is designed with sum of its edge-capacities no more than $\lceil \frac{1}{2} \sum_{i=1}^n \Delta\pi_i^v \rceil$; and the final network $G^*$, obtained by superposing all these networks, has total sum of edge capacities equal to $\lceil \frac{1}{2} \sum_{i=1}^n \pi_i \rceil$. However, each network $G^v$ does not necessarily satisfy the flow requirements corresponding to vector $\Delta\pi^v$. But the fact that $p_1 \geq 1$ allows us to design the network $G^1$ and the subsequent networks $G^2, \ldots$ in such a way that for each $v^0$, the network obtained by superposing $G^1, G^2, \ldots, G^{v^0}$ satisfies flow requirements corresponding to vector $\sum_{v=1}^{v^0} \Delta\pi^v$.

**Algorithm 1**: Hop-Synthesis

**Input**: An integer $n$ ($> 4$); an $n \times n$, integer, symmetric, nonnegative matrix $R = \{r_{ij}\}$ such that for each $i = 1, 2, \ldots, n$, $r_{ii} = 0$ and $\max_{j \neq i} r_{ij} \geq n - 2$;
**for** $i = 1, 2, \ldots, n$ set $\pi_i = \max_{j \neq i} r_{ij}$; permute $\pi$ so that $\pi_1 = \pi_2 \geq \pi_3 \geq \cdots \geq \pi_n$;
set $v = 1$, $\pi^{(1)} = \pi$, $n_1 = n$, $p_1 = \lfloor \frac{\pi_n^{(1)}}{(n-2)} \rfloor$; **for** $(i, j) \in E_n$ set $u_{ij} = 0$;
**if** $n$ is even **then**
  set $M = \{(1, 2), (3, 4), \ldots, (n-1, n)\}$; **for** $(i, j) \in E_n \setminus M$ set $\Delta u_{ij}^{(1)} = p_1$, $u_{ij} = u_{ij} + \Delta u_{ij}^{(1)}$;; **for** $(i, j) \in M$ set $\Delta u_{ij}^{(1)} = 0$; **for** $i = 1, 2, \ldots, n$ set $\pi_i^{(2)} = \pi_i^{(1)} - p_1(n-2)$;
**else**
  set $M^o = \{(1, 2), (1, 3), (2, 3), (4, 5), (6, 7), \ldots, (n-1, n)\}$; **for** $(i, j) \in E_n \setminus M^o$ set $\Delta u_{ij}^{(1)} = p_1$, $u_{ij} = u_{ij} + \Delta u_{ij}^{(1)}$; **for** $(i, j) \in M^o \setminus \{(1, 2), (1, 3), (2, 3)\}$ set $\Delta u_{ij}^{(1)} = 0$; set $\Delta u_{1,2}^{(1)} = \Delta u_{1,3}^{(1)} = \lceil \frac{1}{2} p_1 \rceil$, $\Delta u_{2,3}^{(1)} = \lfloor \frac{1}{2} p_1 \rfloor$; set $u_{1,2} = u_{1,2} + \Delta u_{1,2}^{(1)}$, $u_{1,3} = u_{1,3} + \Delta u_{1,3}^{(1)}$, $u_{2,3} = u_{2,3} + \Delta u_{2,3}^{(1)}$; **for** $i = 1, 2, \ldots, n$ set $\pi_i^{(2)} = \pi_i^{(1)} - p_1(n-2)$;
**end if**
**while** $\pi_3^{(v+1)} > 0$ **do**
  set $v = v + 1$, $n_v =$ the largest integer such that $\pi_{n_v}^{(v)} > 0$, $p_v = \lfloor \frac{\pi_{n_v}^{(v)}}{(n_v - 1)} \rfloor$;
  **if** $p_v > 0$ **then**
    **for** $(i, j) \in E_{n_v}$ set $\Delta u_{ij}^{(v)} = p_v$, $u_{ij} = u_{ij} + \Delta u_{ij}^{(v)}$; **for** $i = 1, 2, \ldots, n_v$ set $\pi_i^{(v+1)} = \pi_i^{(v)} - p_v(n_v - 1)$; **for** $i = n_v + 1, \ldots, n$ set $\pi_i^{(v+1)} = 0$;
  **else**
    set $y = \pi_{n_v}^{(v)}$;
    **if** $\pi_y^{(v)} = \pi_1^{(v)}$ **then** set $\alpha^v = 0$; **else** set $\alpha^v =$ the largest integer such that $\pi_{\alpha^v}^{(v)} > \pi_y^{(v)}$;
    **if** $\pi_y^{(v)} = \pi_{n_v}^{(v)}$ **then** set $\beta^v = n_v - 1$; **else** set $\beta^v =$ the largest integer such that $\pi_{\beta^v}^{(v)} = \pi_y^{(v)}$;
    set $S^{(v)} = \{1, 2, \ldots, \alpha^v, \beta^v - (y - \alpha^v - 1), \beta^v - (y - \alpha^v - 2), \ldots, \beta^v\}$;
    **for** $(i, j) \in E_{n_v}$, **if** $i \in S^{(v)}, j = n_v$ **then** set $\Delta u_{i,j}^{(v)} = 1$, $u_{i,j} = u_{i,j} + 1$; **else** set $\Delta u_{i,j}^{(v)} = 0$;
    **for** $i \in S^{(v)}$ set $\pi_i^{(v+1)} = \pi_i^{(v)} - 1$; set $\pi_{n_v}^{(v+1)} = 0$; **for** $i \in N - (S^{(v)} \cup \{n_v\})$ set $\pi_i^{(v+1)} = \pi_i^{(v)}$;
  **end if**
**end while**
**if** $\pi_1^{(v)} > 0$ and $np_1$ is even **then** increase $u_{1,2}$ by $\pi_1^{(v)}$;
**if** $\pi_1^{(v)} > 0$ and $np_1$ is odd **then** increase $u_{1,2}$ by $\pi_2^{(v)}$;
**for** $i = 1, 2, \ldots, n$ set $\pi_i^{(v+1)} = 0$; **for** $(i, j) \in E_n$ set $u_{i,j}^* = u_{i,j}$;
Output: $G^* = [N, E_n, u^*]$.

The following claims and lemmas will be useful in proving validity of the algorithm. The first claim is easily verified from the definition of set $S^{(v)}$ in the algorithm.

**Claim 1.** In any iteration $v$, if $\pi_1^{(v)} \geq \pi_2^{(v)} \geq \pi_3^{(v)} \geq \cdots \geq \pi_n^{(v)} \geq 0$ and $p_v = 0$, then for any $i, j \in N$ such that $i < j < n_v$, (i) if $\pi_i^{(v)} > \pi_j^{(v)}$ and $j \in S^{(v)}$, then $i \in S^{(v)}$; (ii) if $\pi_i^{(v)} = \pi_j^{(v)}$ and $i \in S^{(v)}$, then $j \in S^{(v)}$.

**Claim 2.** $\forall v,\ \pi_1^{(v)} \geq \pi_2^{(v)} \geq \pi_3^{(v)} \geq \cdots \geq \pi_n^{(v)} \geq 0$.

*Proof.* The result is obviously true for $v = 1$. Since each $\pi_i^{(1)}$ is decreased by the same amount $(= p_1(n-2) \leq \pi_n^{(1)})$, the result is true for $v = 2$. Suppose the result is true for all $v \leq k$ for some $k \geq 2$. If $\pi_3^{(k)} = 0$, then $\pi_i^{(k+1)} = 0\ \forall i$ and the result holds $v = k+1$. Suppose $\pi_3^{(k)} > 0$. If $p_k > 0$ then $\pi_i^{(k)}$ is decreased by the same amount $(= p_k(n_k - 1) \leq \pi_{n_k}^{(k)})$ for each $i \in N_k$ and so the result holds for $v = k+1$. If $p_k = 0$, then in iteration $k$, $\pi_i^{(k)}$ is decreased by one unit for each $i \in S^{(k)}$, and $\pi_{n_k}^{(k)}$ is decreased to zero. It therefore follows from the definitions of $n_k$ and Claim 1 that the result holds for $v = k+1$. The claim now follows using induction hypothesis. □

**Lemma 2.** Suppose that at some iteration $v^0$, $(\pi_i^{(v^0+1)} - \pi_j^{(v^0+1)}) > (\pi_i^{(v^0)} - \pi_j^{(v^0)})$ for some $i < j < n_{v^0}$. Then $(\pi_i^{(v')} - \pi_j^{(v')}) \in \{0, 1\}\ \forall v' \geq v^0$ such that $n_{v'} \geq j$. In particular, $(\pi_1^{(v)} - \pi_2^{(v)}) \in \{0, 1\}\ \forall v$.

*Proof.* Since $i < j < n_{v^0}$ and $(\pi_i^{(v^0+1)} - \pi_j^{(v^0+1)}) > (\pi_i^{(v^0)} - \pi_j^{(v^0)})$, it follows from the description of the algorithm that $p_{v^0} = 0$, $j \in S^{(v^0)}$ and $i \notin S^{(v^0)}$. This, together with claims 1 and 2, implies that $\pi_i^{(v^0)} = \pi_j^{(v^0)}$, and therefore, $(\pi_i^{(v^0+1)} - \pi_j^{(v^0+1)}) = 1$. Let $y$ be the smallest integer greater than $v^0$ such that $n_y > j$ and in iteration $y$, $\pi_i^{(y)}$ and $\pi_j^{(y)}$ decrease by different amounts. Then we must have $p_y = 0$ and it follows from Claim 1 that $i \in S^{(y)}$ and $j \notin S^{(y)}$. Thus, $\pi_i^{(y+1)} = \pi_j^{(y+1)}$. By repeating this process, we see that for any $v' > v^0$ such that $n_{v'} \geq j$, $(\pi_i^{(v')} - \pi_j^{(v')}) \in \{0, 1\}$.

Since $\pi_1^{(1)} = \pi_2^{(1)}$, it follows from the above that $(\pi_1^{(v)} - \pi_2^{(v)}) \in \{0, 1, \}\ \forall v$. □

**Lemma 3.** When $\pi_i \geq n - 2\ \forall\ i \in N$, the solution output by Algorithm *Hop-Synthesis*, if feasible, is an optimal solution to the given instance of the integer, 3-hop-constrained network synthesis problem.

*Proof.* In the first iteration, (steps (4-8) of the algorithm), the total increase in the capacities of edges equals $\lceil \frac{1}{2} p_1(n-2)n \rceil = \lceil \frac{1}{2}$(the total decrease in $\pi$-values in that step)$\rceil$.

In every other iteration $v$ with $\pi_3^{(v)} > 0$, the total increase in capacities of edges in that iteration is half the total decrease in the $\pi$-values in that step. If in the last iteration $v$, $\pi_1^{(v)} > 0$, then since $(\pi_1^{(v)} - \pi_2^{(v)}) = 0$ or 1, increase in the capacity of edge $(1, 2)$ in this iteration equals $\lceil \frac{1}{2}$(total decrease in $\pi$-values in this iteration)$\rceil$ if $np_1$ is even, and it equals $\lfloor \frac{1}{2}$(total decrease in $\pi$ -values in this iteration)$\rfloor$ if $np_1$ is odd. Since at the termination of the algorithm the $\pi$-values are all zero, it follows that the total sum of edge-capacities of the network $G^* = [N, E_n, u^*]$, output by the algorithm, is $\lceil \frac{1}{2} \sum_{i=1}^{n} \pi_i \rceil$. It now follows from Lemma 1 that the solution output by the algorithm, if feasible, is optimal. □

**Theorem 2.** *When $\pi_i \geq n - 2\ \forall\ i \in N$, Algorithm Hop-Synthesis produces an optimal solution to the given instance of the integer, 3-hop-constrained network synthesis problem in $O(n^3)$ time.*

*Proof.* The complexity of every iteration of the algorithm can be easily seen to be $O(n^2)$. It is also easy to see that $n \geq n_v > n_{v+2}\ \forall v$. Hence, the total number of iterations of the algorithm is no more than $2n$. The overall complexity of the algorithm is therefore $O(n^3)$.

Using Lemma 3, it is now sufficient to prove that the solution output by the algorithm is feasible. Thus, consider any $i, j \in N$, $i < j$. We shall show that we can send $\pi_j$ ($\geq r_{ij}$) units of 3-hop flow from $i$ to $j$ in $G^*$ as follows. Since we assume that $\pi_n \geq n-2$, $p_1 \geq 1$ and in the first iteration $\pi_j$ is decreased by $p_1(n-2)$. We shall show that we can send $p_1(n-2)$ units of 3-hop flow from $i$ to $j$ in the capacitated network $G^1 = [N, E_n, \Delta u^{(1)}]$. Then for every iteration, $v = 2, \ldots$ in which the value of $\pi_j$ is deceased, (except possibly the last such iteration), we shall show that we can send from $i$ to $j$ amount of flow equal to the decrease in the $\pi_j$ value in that iteration using only the edge capacities $\Delta u^{(v)}$ added in that iteration together with some of the still unused capacity in $G^1 = [N, E_n, \Delta u^{(1)}]$. Finally, in the last iteration in which the value of $\pi_j$ is deceased to zero, we shall again show that we can send the desired amount of additional flow from $i$ to $j$ using the additional edge capacities added in that iteration and the residual capacities of edges in $G^1 = [N, E_n, \Delta u^{(1)}]$. But in this case, under certain cases, we will slightly modify the flow of $p_1(n-2)$ units sent in the first iteration to accommodate the desired additional flow.

In the first iteration $\pi_j^{(1)}$ ($= \pi_j$) is decreased by $p_1(n-2)$. We shall first show that we can send $p_1(n-2)$ units of 3-hop flow from $i$ to $j$ in the capacitated network $G^1 = [N, E_n, \Delta u^{(1)}]$. (This flow will be slightly modified later on to get a total flow from $i$ to $j$ of $\pi_j$ units.)

**Case 1 ($n$ is even)**: If $\{(x,i),(y,j)\} \subseteq M$ for some $x \neq y \neq i$, then assign $p_1$ units of flow to each of the $(n-2)$ paths $\{(i,j)\} \cup \{(i,a,j) : a \in N \setminus \{i,j,x,y\}\} \cup \{(i,y,x,j)\}$. If $\{(i,j)\} \subseteq M$, then assign $p_1$ units of flow to each of the $(n-2)$ paths $\{(i,a,j) : a \in N \setminus \{i,j\}\}$.

**Case 2a ($n$ is odd and $3 < i < j$)**: If $\{(x,i),(y,j)\} \subseteq M^o$, for some $x \neq y \neq i$, then assign $p_1$ units of flow to each of the $(n-2)$ paths $\{(i,j)\} \cup \{(i,a,j) : a \in N \setminus \{i,j,x,y\}\} \cup \{(i,y,x,j)\}$. If $\{(i,j)\} \subseteq M^o$, then assign $p_1$ units of flow to each of the $(n-2)$ paths $\{(i,a,j) : a \in N \setminus \{i,j\}\}$.

**Case 2b ($n$ is odd, $i \in \{2,3\}$ and $j > 3$)**: Let $\{x\} = \{2,3\} \setminus \{i\}$ and $(y,j) \in M^o$. Assign a $p_1$ units of flow to each of the $(n-4)$ paths $\{(i,j)\} \cup \{(i,a,j) : a \in N \setminus \{i,j,1,x,y\}\}$, a $\lceil \frac{1}{2}p_1 \rceil$ units of flow to each of the paths $(i,1,j)$ and $(i,y,x,j)$, and a $\lfloor \frac{1}{2}p_1 \rfloor$ units of flow to each of the paths $(i,x,j)$ and $(i,y,1,j)$.

**Case 2c ($n$ is odd, $i = 1$ and $j > 3$)**: Let $(y,j) \in M^o$. Assign a $p_1$ units of flow to each of the $(n-4)$ paths $\{(1,j)\} \cup \{(1,a,j) : a \in N \setminus \{1,2,3,y,j\}\}$, a $\lceil \frac{1}{2}p_1 \rceil$ units of flow to each of the paths $(1,2,j)$ and $(1,3,j)$, and a $\lfloor \frac{1}{2}p_1 \rfloor$ units of flow to each of the paths $(1,y,2,j)$ and $(1,y,3,j)$.

**Case 2d ($n$ is odd, $i = 1$ and $j \in \{2,3\}$)**: Let $\{x\} = \{2,3\} \setminus \{j\}$. Assign a $p_1$ units of flow to each of the $(n-3)$ paths $\{(i,a,j) : a \in N \setminus \{1,2,3\}\}$, a $\lceil \frac{1}{2}p_1 \rceil$ units of flow to the path $(1,j)$, and a $\lfloor \frac{1}{2}p_1 \rfloor$ units of flow to the path $(1,x,j)$.

**Case 2e ($n$ is odd, $i = 2$ and $j = 3$)**: Assign a $p_1$ units of flow to each of the $(n-3)$ paths $\{(2,a,3) : a \in N \setminus \{1,2,3\}\}$, a $\lceil \frac{1}{2}p_1 \rceil$ units of flow to the path $(2,1,3)$, and a $\lfloor \frac{1}{2}p_1 \rfloor$ units of flow to the path $(2,3)$.

It may be noted that in each of the above cases, after sending the $p_1(n-2)$ units of 3-hop flow from $i$ to $j$ in $G^1$, there will be a residual capacity of at least one unit on all the edges in the subgraph of $G^1$ induced by node set $N \setminus \{i,j\}$ except possibly the set $M'$ defined below. This unused capacity will be used in later iterations to send desired additional flow.

In Case 1, $M' = \{(1,2),(3,4),\ldots,(n-1,n)\} \setminus \{(i,j)\}$ if $(i,j) \in M$ and $M' = \{(1,2),(3,4), \ldots,(n-1,n)\} \cup \{(x,y)\} \setminus \{(x,i),(y,j)\}$ otherwise. In Case 2a, $M' = \{(2,3),(4,5),\ldots,(n-1,n)\} \setminus \{(i,j)\}$ if $(i,j) \in M$ and $M' = \{(2,3),(4,5),\ldots,(n-1,n)\} \cup \{(x,y)\} \setminus \{(x,i),(y,j)\}$ otherwise. In Case 2b, $M' = \{(x,y),(4,5),\ldots,(n-1,n)\} \setminus \{(y,j)\}$. In Case 2c, $M' = \{(2,3),(4,5),\ldots,(n-1,n)\} \setminus \{(y,j)\}$, and in each of case 2d and 2e, $M' = \{(4,5),(6,7),\ldots,(n-1,n)\}$.

The set of iteration numbers, $v$ ($> 1$) with $n_v \geq j$ in which the value of $\pi_j^{(v)}$ is decreased can be partitioned into sets $\{V^1, V^{2b}, V^{2j}\}$, where $V^1 = \{v : n_{(v)} \geq j; p_v > 0\}$; $V^{2b} = \{v : p_v = 0; \{i,j\} \subseteq S^{(v)}\}$ and $V^{2j} = \{v : p_v = 0$; and either ($j \in S^{(v)}$ and $i \notin S^{(v)}$) or $j = n_v\}$.

For any $v \in V^1$, $\pi_j^{(v)}$ is decreased by $p_v(n_v - 1)$ in iteration $v$. We can send a total of $p_v(n_v - 1)$ units of 3-hop flow from $i$ to $j$ along the network $G^v = [N_v, E_{n_v}, \Delta u^{(v)}]$ by assigning $p_v$ units of flow to each of the $(n_v - 1)$ paths $\{(i,j)\} \cup \{(i,a,j) : a \in N_v \setminus \{i,j\}\}$.

For any $v \in V^{2b}$, the value of each of $\pi_i^{(v)}$ and $\pi_j^{(v)}$ is decreased by 1. Assign one unit of flow to the path $(i, n_v, j)$. This uses the unit capacities added to the edges in this path in this iteration.

Now let us consider the set $V^{2j}$. Let $V = \{v_1, v_2, \ldots, v_k\} \subseteq V^{2j}$ be all the elements of $V^{2j}$ with $v_1 < v_2 < \cdots < v_k$ and $n_{v_k} > j$. For each $v_\ell \in V$, the value of $\pi_j^{(v_\ell)}$ is decreased by 1 in iteration $v_\ell$. We thus have to send $k$ additional units of flow corresponding to the $k$ elements of $V$. We shall show below how this can be achieved and in doing so, we shall use some of the unused capacities on the edges of the subgraph of $G^1$ induced by node set $N \setminus \{i,j\}$ except the set $M'$ defined above.

If $k \leq 1$ then we set $Z = V$. If $k > 1$ then it follows from the proof of Lemma 2 that there exist $\{x_1, x_2, \ldots, x_{k-1}\}$ such that for each $\ell \in \{1, 2, \ldots, k-1\}$, $v_\ell < x_\ell < v_{\ell+1}$, $p_{x_\ell} = 0$, $i \in S^{(x_\ell)}$ and $j \notin S^{(x_\ell)}$.

If $k = 2$ then if $(n_{x_1}, n_{v_1}) \notin M'$, assign one unit of flow to the path $(i, n_{x_1}, n_{v_1}, j)$ and set $Z = \{v_2\}$; else, assign one unit of flow to the path $(i, n_{x_1}, n_{v_2}, j)$ and set $Z = \{v_1\}$.

If $k > 2$ then let $\alpha = \lceil \frac{k}{2} \rceil$. If $k$ is odd then assign an additional unit of flow to each of the paths $\{(i, n_{x_{\alpha+\ell-1}}, n_{v_\ell}, j),\ (i, n_{x_\ell}, n_{v_{\alpha+\ell}}, j) : \ell = 1, 2, \ldots, \alpha - 1\}$ and set $Z = \{v_\alpha\}$. If $k$ is even then assign an additional unit of flow to each of the paths $\{(i, n_{x_{\alpha+\ell-1}}, n_{v_\ell}, j) : \ell = 1, 2, \ldots, \alpha\} \cup \{(i, n_{x_\ell}, n_{v_{\alpha+\ell}}, j) : \ell = 1, 2, \ldots, \alpha - 1\}$ and set $Z = \{v_k\}$. This accounts for $k - |Z|$ units of additional flow, where $|Z| \leq 1$. If $V = V^{2j}$ and $Z = \emptyset$ then we are done.

If $V = V^{2j}$ and $Z = \{\gamma\}$ then we have to send just one more unit of flow corresponding to iteration number $\gamma$. By Claim 2, there exists $\mu < n_{v_k} \leq n_\gamma$ such that in some iteration $v$ ($> v_k$), capacity of edge $(i, \mu)$ is increased by at least one unit. If $(\mu, n_\gamma) \notin M'$ then assign an additional unit of flow to the path $(i, \mu, n_\gamma, j)$ and we are done. Suppose $(\mu, n_\gamma) \in M'$. In this case we shall choose a suitable path to which flow was assigned when we sent in the $p_1(n-2)$ units of flow corresponding to the first iteration and we shall alter the flow assignment to this path to accommodate, overall, one more unit of flow. For this, we shall consider separately the various cases encountered in sending flow corresponding to the first iteration.

In Case 1, if there exists $\beta \in N \setminus \{i, j, x, y, \mu, n_\gamma\}$, then reduce the flow already assigned to path $(i, \beta, j)$ by one unit and assign an additional unit of flow to each of the paths $(i, \beta, n_\gamma, j)$ and $(i, \mu, \beta, j)$. If no such $\beta$ exists, then since $n \geq 5$, $i, j, x, y$ must all be distinct. Reduce the flow assigned to path $(i, y, x, j)$ in the first iteration by one unit and assign an additional unit of flow to each of the paths $(i, y, n_\gamma, j)$ and $(i, \mu, x, j)$.

In each of the cases 2a-2e, reduce flow assigned to path $(i, \sigma, j)$ in the first iteration by one unit and assign an additional unit of flow to each of the paths $(i, \sigma, n_\gamma, j)$ and $(i, \mu, \sigma, j)$, where in cases 2a, 2b and 2e, $\sigma = 1$; in case 2c, $\sigma = 2$ and in case 2d, we can choose any $\sigma \in \{3, 4, \ldots, n\} \setminus \{\mu, n_\gamma\}$.

Suppose $V^{2j} = V \cup \{t\}$, where $n_t = j$. Then in iteration $t$, $p_t = 0$ and the value of $\pi_j^{(t)}$ is reduced to zero. If $i \in S^{(t)}$ then assign an additional unit of flow to the path $(i, j)$. Let $Y = S^{(t)} - \{i\}$. If $Y = \emptyset$ send an additional $|Z|$ units of flow from $i$ to $j$ as in the previous case and we are done.

Suppose $Y \neq \emptyset$. Then since by Claim 2, $\pi_j^{(v)} \leq \pi_i^{(v)}$, there exists $T \subseteq \{1, 2, \ldots, n_{v_k} - 1\} \setminus \{i, j\}$ such that after iteration number $v_k$, edges $\{(i, a) : a \in T\}$ are assigned additional capacities that add up to at least $|Y| + |Z|$. IF $|Z| = 1$ then by Corollary 2 and Lemma 2, there exists at most one element $a' \in T$ such that $a' > n_t$ and if such an element $a'$ exists, then the additional capacity assigned to edge $(i, a')$ after iteration number $v_k$ is one unit. Choose any $b \in Y$, assign an additional unit of flow to path $(i, a', b, j)$ and delete $a'$ from $T$ and $b$ from $Y$. Let $Z = \{\gamma\}$. Choose any $a \in T$, (obviously $a < n_t$), assign an additional unit of flow to the path $(i, a, n_\gamma, j)$, delete $\gamma$ from $Z$, reduce

the residual capacity of the edge$(i, a)$ by one unit and if it is zero then delete $a$ from $T$. Now we have to send an additional $|Y|$ units of flow from $i$ to $j$.

For each $a \in Y \cap T$, assign one unit of flow to the path $(i, a, j)$, set $Y = Y \setminus \{a\}$, reduce residual capacity of the edge $(i, a)$ by one unit and if it becomes zero then set $T = T \setminus \{a\}$. We thus have $Y \cap T = \emptyset$.

If $|T| > 1$, then let $T' \subseteq T$ and $Y' \subseteq Y$ be largest sets such that for each $a \in T'$, there exists $b \in Y'$ such that $(a, b) \in M'$. If $|T'| > 1$ then by Konig's theorem (citeLov Theorem 1.4.18), there exists a matching $\theta : T' \to Y'$ such that for any $a \in T'$, $(a, \theta(a)) \notin M'$. For each $a \in T'$, assign an additional unit flow to the path $(i, a, \theta(a), j)$, delete $\theta(a)$ from $Y$ and $Y'$ , delete $a$ from $T'$, reduce the residual capacity of the edge $(i, a)$ by one unit and if it becomes zero, then set $T = T \setminus \{a\}$. Thus, we have $|T'| \leq 1$.

If $T' = \{a\}$ and $Y \setminus Y' \neq \emptyset$ then arbitrarily choose $b \in Y \setminus Y'$, assign an additional unit flow to the path $(i, a, b, j)$, set $Y = Y \setminus \{b\}$, reduce the residual capacity of the edge $(i, a)$ by one unit and if it becomes zero, then delete $a$ from $T$ and $T'$. By repeating this process, we get $T' = \emptyset$ or $Y \setminus Y' = \emptyset$.

If $T' = \emptyset$, then arbitrarily choose $a \in T,\ b \in Y$, assign an additional unit flow to the path $(i, a, b, j)$, set $Y = Y \setminus \{b\}$, reduce the residual capacity of the edge $(i, a)$ by one unit and if it becomes zero, then set $T = T \setminus \{a\}$. Repeat this process until $Y = \emptyset$.

If $Y \setminus Y' = \emptyset$, then say $Y = Y' = \{b\}$. If there exists $a \in T \setminus T'$then assign an additional unit flow to the path $(i, a, b, j)$, and we are done. Suppose $T = T' = \{a\}$. Then nodes $i,\ a,\ b$ are distinct and less than $j$. Hence, in the first iteration of the algorithm, cases $2d$ and $2e$ cannot occur. In each of the cases 1, $2a$ and $2b$, there exists a node $w \in N \setminus \{i, j, a, b\}$ such that each of the edges $(w, a)$ and $(w, b)$ has an unused capacity of at least one unit in $G^1$. Reduce the flow already assigned to the path $(i, b, j)$ by one unit and assign an additional flow of one unit to each of the paths $(i, w, b, j)$ and $(i, a, w, j)$. Now, let us consider case case $2c$. If $n \geq 7$, then there exists a node $w \in N \setminus \{i, j, a, b, y\}$ such that edges $(a, w)$ and $(b, w)$ each has at least one unit of unsued capacity in $G^1$, (where $y$ is as defined in case $2c$). Reduce the flow already assigned to path $(i, w, j)$ by one unit and assign an additional flow of one unit to each of the paths $(i, w, b, j)$ and $(i, a, w, j)$.

If $n = 5$, then $\{a, b\} = \{2, 3\}$. If there is unused residual capacity on the edge $(2, 3)$ in $G^1$, then send one unit of flow along the path $(i, a, b, j)$ . Else, if there is no unused residual capacity on the edge $(2, 3)$ in $G^1$, then $p_1$ must equal 1, which implies that there is one unit of unused residual capacity on the edge $(i, y)$. (Here, $y$ is as defined in case $2c$.) We also have an unused residual capacity on each of the edges $(y, a)$ and $(y, b)$ in $G^1$. Assign an additional flow of one unit to the path $(i, y, b, j)$.

This proves the theorem. □

The complexity of the above algorithm can be reduced to $O(n^2)$ as follows: Let $V^4$ be the set of iteration numbers during which the algorithm performs Step 4. Instead of updating the edge capacities $u_{ij}$ as per Step 4 during each of the iterations in $V^4$, we can store the data $\{v, n_v, p_v\}$ corresponding to each such iteration $v$ and perform the corresponding updating of the values $u_{ij}$ simultaneously in the end as follows: let $v_1 < v_2 < \cdots < v_k$ be an arrangement of elements of $V^4$ in increasing order. Set $U = p_{v_1}$ and for each edge $(i, j)$ with $i < j$ and $n_{v_2} < j \leq n_{v_1}$ increase the capacity of the edge by $U$. Now update $U = U + p_{v_2}$ and increase the capacity of each edge $(i, j)$ with $i < j$ and $n_{v_3} < j \leq n_{v_2}$ by $U$. Repeat the process. All this takes $\mathrm{O}(n^2)$ time. We thus have the following theorem.

**Theorem 3.** *When $\pi_i \geq n - 2\ \forall\ i \in N$, an optimal solution to the given instance of the integer, 3-hop-constrained network synthesis problem can be obtained in $O(n^2)$ time.*

## 4. Conclusion

Consider the case $h = 3$, $n = 5$ and $\pi_1 = \pi_2 = \pi_3 = \pi_4 = \pi_5 = 2$. Then $\left\lceil \frac{1}{2} \sum_{i=1}^{n} \pi_i \right\rceil = 5$. In this case $\pi_n = \pi_5 = 2 = (n-3)$. In an optimal solution to the integer version of the corresponding synthesis problem, if edges with positive capacities form a tree then obviously, capacity of each edge will have to be 2 and so the objective function value will be $2 \times (4) = 8 > 5$. If the number of edges with positive capacity is at least $n = 5$, then to achieve the objective function value of 5, these edges must form a Hamiltonian cycle and each must have a capacity of 1. But it is easy to see that such a solution is not feasible. Hence, the optimal objective function value has to be strictly greater than $\left\lceil \frac{1}{2} \sum_{i=1}^{n} \pi_i \right\rceil$. We however conjecture the following:

**Conjecture 1**: When $n > 5$ and $\pi_i \geq n - 3 \ \forall \ i \in N$, optimal objective function value of the given instance of the integer, 3-hop-constrained network synthesis problem is $\left\lceil \frac{1}{2} \sum_{i=1}^{n} \pi_i \right\rceil$.

**Conjecture 2**: For any integer $2 < h < n$, when $\pi_i \geq n - h + 1 \ \forall \ i \in N$, optimal objective function value of the given instance of the integer, $h$-hop-constrained network synthesis problem is $\left\lceil \frac{1}{2} \sum_{i=1}^{n} \pi_i \right\rceil$.

## References


[1] R. Ahuja, T. L. Magnanti and J. B. Orlin. *Network Flows: Theory, Algorithms and Applications*, Prentice-Hall, Englewood, Cliffs, NJ, 1993.

[2] Y. P. Aneja, R. Chandrasekaran, S. N. Kabadi and K. P. K. Nair. Flows over Edge-Disjoint Mixed Multipaths and Applications, *Discrete Applied Mathematics*, 155(15), 1979-2000, 2007.

[3] R. Chandrasekaran, K.P.K. Nair, Y.P. Aneja and S.N. Kabadi. Multi-terminal Multipath Flows: Synthesis, *Discrete Applied Mathematics*, 143, 182-193, 2004.

[4] W. Chow and H. Frank. Survivable Communication Networks and the Terminal Capacity Matrix, *IEEE Trans. Circuit Theory*, CT-17, 192-197, 1970.

[5] G. Dahl. Notes on Polyhedra Associated with Hop-Constrained Paths, *Operations Research Letters*, 25, 97-100, 1999.

[6] D. Du and S. N. Kabadi. An Improved Algorithm for decomposing Arc Flows into Multipath Flows, *Operations Research Letters*, 34, 53-57, 2006.

[7] L.R. Ford, and D.R. Fulkerson. *Flows in Networks*, Princeton University Press, 1962.100

[8] R. J. Gibbens and F. P. Kelly. Dynamic Routing in Fully Connected Networks, *IMA Journal of Mathematical Control and Information*, 7, 77-111, 1990.

[9] R.E. Gomory and T.C. Hu. Multi-terminal network flows, *Journal of SIAM*, 9, 551-570, 1961.

[10] R. Hassin and A. Levin. Synthesis of 2-Commodity Flow Networks, *Mathematics of Operations Research*, 29 (2), 280-88, 2004.

[11] A. Itai, Y. Perl and Y. Shiloach. The Complexity of Finding Maximum Disjoint Paths with Length Constraints, *Networks*, 12, 277-286, 1982.

[12] S. N. Kabadi, R. Chandrasekaran, K.P.K. Nair and Y. P. Aneja. Integer Version of the Multipath Flow Network Synthesis Problem, *Discrete Applied Mathematics*, 156(18), 3376-99, 2008.

[13] S. N. Kabadi, R. Chandrasekaran and K.P.K. Nair. 2-Commodity Integer Network Synthesis Problem, *Algorithmic Operations Research*, to appear.

[14] S. N. Kabadi, J. Yan, D. Du and K. P. K. Nair. Integer Exact Network Synthesis Problem, *SIAM Journal on Dicrete Mathematics*, 23(1), 136-54, 2008.

[15] S. N. Kabadi, J. Kang, R. Chandrasekaran and K. P. K. Nair. Hop-Constrained Network Flows: Analysis and Synthesis, *working paper, Faculty of Administration, University of New Brunswick*, October, 2003.

[16] S. N. Kabadi and R. Sridhar. Peeling Algorithm for integral Network Synthesis, *working paper, Faculty of Administration, University of New Brunswick*, January, 1996.

[17] W. Kishimoto. A method for obtaining the maximum multi-route flows in a network, *Networks*, 27, 279-291, 1996.

[18] W. Kishimoto and M. Takeuchi. On $m$-route flows in a network, *IEICE Trans.* J-76-A , 1185-1200, 1993 (in Japanese).

[19] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*, Algorithms and Combinatorics 21, Springer-Verlag, New York, 2002.
[20] L. Lovasz, V. Neumann-Lara and M. Plummer. Mengerian Theorems for Paths of Bounded Length, *Periodica Mathematica Hungerica*, 9, 4, 269-276, 1978.
[21] W. Mayeda. Terminal and branch capacity matrices of a communication Net, *IRE Transactions on Circuit Theory*, CT-7, 261-269, 1960.
[22] S. T. McCormick. The Complexity of Max Flow and Min Cut with Bounded-Length Paths, *working paper, Faculty of Commerce and Business Administration, University of British Columbia*, May, 2001.
[23] K. G. Murty. *Network Programming*, Prentice Hall, Englewood Cliffs, New Jersey, 1992.
[24] L. Lovasz and M. D. Plummer. *Matching Theory*, Annals of Discrete Mathematics, 29, North-Holland, New York, 1986.
[25] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, Algorithms and Combinatorics 24, Springer-Verlag, New York, 2003.
[26] R. Sridhar and R. Chandrasekaran. Integer Solution of Synthesis of Communication Network. *Mathematics of Operations Research*, 17, 3, 581-585, 1992.

Faculty of Business Administration, University of New Brunswick, Fredericton, New Brunswick, Canada
*E-mail address*: kabadi@unb.ca

Faculty of Business Administration, University of New Brunswick, PO Box 4400, Fredericton, N.B., Canada E3B5A3
*E-mail address*: nairk@unb.ca