# Theory of Network Communication

Dr. Christian Scheideler

Johns Hopkins University, Sept. 9 2002

## 1 Introduction

The goal of this lecture is to give an introduction to the state of the art in the theory of network communication. It is a widely accepted fact that algorithmic advances in the area of computer science are only useful to society if they are based on models that truthfully reflect the restrictions and requirements of the corresponding applications. This is certainly also true for network communication. For example, in general any infrastructure connecting processing units with each other may be called a network, but certain infrastructures such as a dedicated line between any pair of nodes are certainly unrealistic, because they are too expensive to build. Thus, messages may have to traverse several units to reach their destination, causing (among other problems) route selection and scheduling problems. Also, we will not just study network communication out of context, but we will also look at various topics that require or support efficient network communication such as distributed sorting, load balancing, data management, or design strategies for overlay networks.

In this section we start with an introduction to graph theory followed by a simple routing example. Afterwards, we will speak shortly about various communication layers and modes and give an introduction to network flows and parameters.

### 1.1 Graph theory

A *graph* $G = (V, E)$ consists of a set of *nodes* (or *vertices*) $V$ and a set of *edges* (or *arcs*) $E$. Often, we will set $n = |V|$ (the size of $V$) and $m = |E|$. The *size* of $G$ is defined as the number of nodes it contains. For all $v, w \in V$, $(v, w)$ denotes a *directed* edge from $v$ to $w$, and $\{v, w\}$ denotes an *undirected* edge from $v$ to $w$. $G$ is called *undirected* if $E \subseteq \{\{v, w\} \mid v, w \in V\}$ and *directed* if $E \subseteq \{(v, w) \mid v, w \in V\}$. Unless explicitly mentioned, we assume for the rest of this lecture that $G$ is undirected.

A sequence of contiguous edges in $G$ is called a *path*. The *length* of the path is defined as the number of edges it contains. A path is called *node-simple* if it visits every node in $G$ at most once. Similarly, it is called *edge-simple* (or *simple* for short) if it contains every edge in $G$ at most once. $G$ is called *connected* if, for any pair of nodes $v, w \in V$, there is a path in $G$ from $v$ to $w$. We call a path a *cycle* if it starts and ends at the same node. The *girth* of a graph $G$ is defined as the length of the shortest cycle $G$ contains. $G$ is called a *tree* if it is connected and contains no cycle. A graph $T = (V', E')$ is called a *spanning tree* of $G$ if $V' = V$, $E' \subseteq E$, and $T$ is a tree. $G$ is called *bipartite* if its node set can be partitioned into two node sets $V_1$ and $V_2$ such that $E \subseteq \{\{v, w\} \mid v \in V_1,\ w \in V_2\}$.

For any pair of nodes $v, w \in V$, let $\delta(v, w)$ denote the *distance* of $v$ and $w$ in $G$, that is, the length of a shortest path from $v$ to $w$. The *diameter* $D$ of $G$ is defined as $\max\{\delta(v, w) \mid v, w \in V\}$. If $\{v, w\} \in E$ then $v$ is called a *neighbor* of $w$. For any subset $U \subseteq V$, the *neighborhood* of $U$ is defined as

$$\Gamma(U) = \{v \in V \setminus U \mid \exists u \in U : \{u, v\} \in E\} .$$

The number of neighbors of $v$ is called the *degree* of $v$ and denoted by $d_v$. The degree of $G$ is defined as $d = \max\{d_v \mid v \in V\}$. If all nodes in $G$ have the same degree, then $G$ is called *regular*.

A family of graphs $\mathcal{G} = \{G_n \mid n \in \mathbb{N}\}$ has degree $d(n)$ if for all $n \in \mathbb{N}$ the degree of $G_n$ is $d(n)$. If it is clear to which family a graph belongs, we say that this graph has constant (or bounded) degree if and only if its family has constant degree.

A *network* is specified by a graph $G = (V, E)$ with edge capacities given by a function $c : E \rightarrow \mathbb{R}^+$. Given a graph $G$ with capacities $c$, let the capacity of a node $v \in V$ be defined as

$$c(v) = \sum_{w \in V} c(\{v, w\})$$

and the capacity of any node set or edge set $U$ be defined as $c(U) = \sum_{u \in U} c(u)$.

## 1.2 A simple example

*Routing* is the process of sending information in a network from a source to a destination. Let us start with a simple routing example in a network of uniform edge capacities. We assume that information has to be sent through the network in the form of *packets* of uniform size. The nodes work in a *synchronous* way, that is, all nodes start their next step at the same time. In each time step a vertex $v$ is able to perform the following tasks:

- create new packets (if any),
- send out packets,
- receive incoming packets, and
- absorb packets whose destination is $v$.

We will not count internal computations. A packet needs one time step to cross an edge and each edge can transport at most one packet in each direction at a time.

A *cycle* is a graph $G = (V, E)$ with

$$E = \{\{v_i, v_{i+1}\} : i \in \{1, \dots, n-1\}\} \cup \{\{v_n, v_1\}\} .$$

Suppose that we are given a set of $m$ packets represented as source-destination pairs $(s_i, d_i)$, $i \in \{1, \dots, m\}$. Initially, all packets are stored at their sources. In order to send the packets to their destinations, the nodes choose the *shortest path* and use the *furthest-to-go* (or FTG) rule. FTG means that if two packets contend to use the same link in the same direction at the same time step, then the one that has the most edges to go is preferred. If several packets have the same number of edges to go, then the one of maximum index $i$ wins. For this strategy we can show the following result.

**Theorem 1.1** *For any set of $m$ source-destination pairs our strategy needs at most $m + \lfloor n/2 \rfloor - 1$ time steps to send all packets to their destinations.*

**Proof.** Let $P = (p_1, \ldots, p_m)$ denote the set of all packets, and let the *rank* $\mathrm{rank}_v(p_i)$ of packet $p_i$ at node $v$ be defined as the number of edges $p_i$ still has to traverse from $v$ plus $i/(m+1)$. According to our preference rules it must hold for any two packets $p, q$ at some node $v$ where $p$ is preferred against $q$ that

$$\mathrm{rank}_v(p) > \mathrm{rank}_v(q) .$$

We will need this observation later.

Now, let us use a proof method called *backwards analysis*. We start with the last packet that reached its destination. Let this packet be called $q_1 \in \{p_1, \ldots, p_m\}$, and let its destination be called $v_0$. We follow $q_1$ backwards in time till it was delayed by some other packet, say $q_2$, at some node $v_1$. Let $\ell_1$ be the number of edges traversed from $v_0$ to $v_1$. We continue to follow $q_2$ backwards in time until it was delayed by some other packet, say $q_3$, at some node $v_2$. Let $\ell_2$ be the number of edges traversed from $v_1$ to $v_2$. In general, we will follow packet $q_i$ backwards in time until it was delayed by some other packet, say $q_{i+1}$, at some node $v_i$. Let $\ell_i$ be the number of edges traversed from $v_{i-1}$ to $v_i$. If we reach a packet $q_i$ that has not been delayed by any other packet, we simply follow it backwards in time until it reached its source node, called $v_i$.

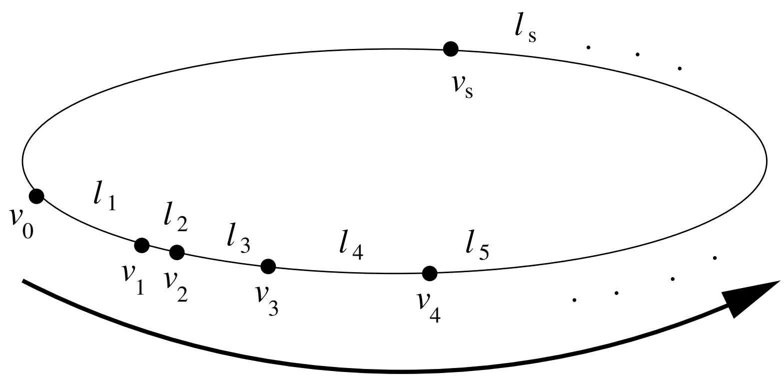


Figure 1: A possible outcome of the backwards analysis.

Assume now that we touched $s$ packets $q_1, \ldots, q_s$ in the course of this argument. Using our observation above, we can state the following three facts:

1. $\mathrm{rank}_{v_0}(q_1) \geq 0$
2. $\mathrm{rank}_{v_i}(q_{i+1}) > \mathrm{rank}_{v_i}(q_i)$
3. $\mathrm{rank}_{v_i}(q_i) = \mathrm{rank}_{v_{i-1}}(q_i) + \ell_i$

We will use these facts to prove the following lemmas.

**Lemma 1.2** $\sum_{i=1}^{s} \ell_i \leq \lfloor n/2 \rfloor$.

**Proof.** Using facts (1) to (3), we obtain that

$$\text{rank}_{v_s}(q_s) \geq \sum_{i=1}^{s} \ell_i$$

However, since $\text{rank}_{v_s}(q_s)$ must be smaller than $\lfloor n/2 \rfloor + 1$ due to a maximum path length of $\lfloor n/2 \rfloor$ and since the $\ell_i$ are integers, the lemma follows. $\sqcap$

**Lemma 1.3** $s \leq m$

**Proof.** Due to fact (2) and the fact that if $\text{rank}_v(p) > \text{rank}_v(q)$ for two packets $p$ and $q$ then this relationship also holds for all other nodes visited by $p$ and $q$, it holds that no packet can appear more than once in the backwards argument. Hence, $s \leq m$. $\sqcap$

Obviously, the total runtime of our strategy is equal to the time covered by the backwards argument. Since the number of time steps covered by this argument is exactly $\sum_{i=1}^{s} \ell_i + s - 1$, the two lemmas above imply a runtime of at most

$$\lfloor n/2 \rfloor + m - 1 \ .$$

$\sqcap$

Theorem 1.1 is worst case optimal, since a routing problem can easily be set up that requires $m + \lfloor n/2 \rfloor - 1$ steps (see Figure 2).

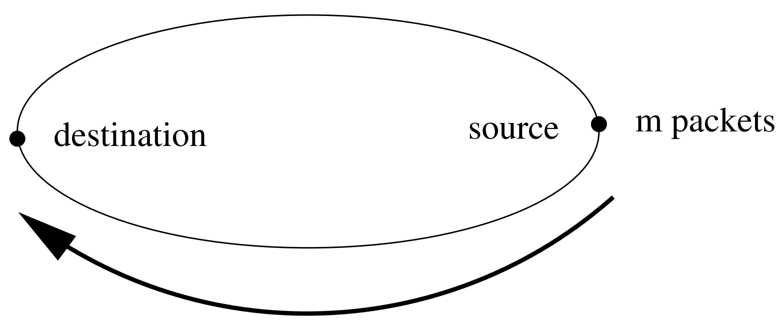


Figure 2: A worst case situation.

## 1.3 Communication layers and modes

The simple example above demonstrated how to efficiently communicate in a cycle. It also demonstrates that several components have to be specified to be able to design and analyze strategies for network communication: a path selection rule, and a rule for scheduling packets along the selected paths. (In our example above the path selection rule was "use the shortest path" and the scheduling rule was FTG.) These rules are quite general and could also be applied to other networks than just the cycle. Such rules are therefore also called *universal*. We will have a closer look at these rules later on in the lecture.

Network communication can sometimes be a very complex task and is therefore usually divided into several layers in practice:

- *Medium access control* or *link layer*: handles the exchange of information between neighbors and the scheduling (i.e. a preference rule for the packets). May also handle topology control.

- *Network layer*: handles the path selection.

- *Transport layer*: handles the logical end-to-end transport of information, including rate- and admission control.

For the first half of this lecture we will assume that we are given a network of fixed point-to-point connections, and we only have to deal with the problem of routing information through this network. Several communication modes are possible:

- *Gathering*: all messages have the same destination.

- *Unicasting*: each message has a single, arbitrary destination.

- *Anycasting*: each message has a set of destinations but only has to reach any one of them.

- *Multicasting*: each message has a set of destinations and has to reach all of them.

- *Broadcasting*: each message has to reach all of the destinations.

We will mostly concentrate on unicasting, i.e. point-to-point communication problems. As we will see at the end of this section, network flow results help us to get a first idea of how to measure the ability of a network to support unicast traffic.

## 1.4 Single commodity flows

In a *single commodity flow problem* we have a network $G = (V, E)$ with positive edge capacities specified by a function $c : E \to \mathbb{R}^+$. One of the nodes is designated as the *source* $s$ and one as the *sink* or *target* $t$. The objective is to route as much flow as possible from the source to the sink without violating the capacity constraints. The maximum amount of flow that can be so routed is called the *max-flow*. The *min-cut* is the minimum amount of capacity that needs to be removed from the network in order to disconnect the source from the sink. One of the earliest and most important advances in the field of network flows was the result by Ford and Fulkerson [2, 3] that for any instance of the single commodity flow problem the min-cut is equal to the max-flow. To clarify this result we introduce some notation.

A *flow* from a source $s$ to a sink $t$ is a function $f : E \to \mathbb{R}^+$ with the property that

- for all $v \in V \setminus \{s, t\}$, $\sum_u f(u, v) = \sum_w f(v, w)$ (also called *flow conservation principle*), and

- $\sum_v f(s, v) = \sum_v f(v, t)$.

$f$ is called *feasible* if $f(e) \leq c(e)$ for all $e \in E$. The *(absolute) flow value* or *throughput* of a flow $f$ is defined as $t_f = \sum_v f(s, v)$. The maximum flow value reachable by a feasible flow is called *max-flow* and defined here as $T$.

For every $U \subseteq V$ the edge set $(U, \bar{U})$ denotes the set of all edges that link a node in $U$ to a node in $\bar{U}$. $(U, \bar{U})$ is also referred to as a *cut* of the network since the removal of these edges separates $U$ from the rest of the network. The capacity of a cut $(U, \bar{U})$ is defined as

$$c(U, \bar{U}) = \sum_{e \in (U, \bar{U})} c(e) \ .$$

The min-cut $C$ of a single commodity flow problem with source-sink pair $(s, t)$ is equal to

$$\min_{U \subset V : (s,t) \in U \times \bar{U}} c(U, \bar{U}) \ .$$

**Theorem 1.4 (Min-cut max-flow theorem [2])** *For any instance of a single commodity flow problem, $C = T$, where $C$ is the min-cut and $T$ is the max-flow of the problem. Finding such a cut can be done in polynomial time.*

Certainly, the max-flow can be at most the min-cut, because for every flow with value $f$ and every cut $(U, \bar{U})$ separating $s$ and $t$, $f \leq c(U, \bar{U})$. To prove equality is more involved and omitted here.

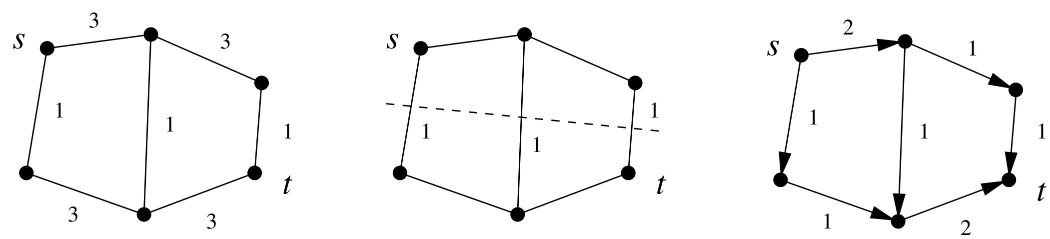


Figure 3: A 1-commodity flow problem (a) for which the min-cut (b) and max-flow (c) are both 3.

## 1.5 Multicommodity flows

Next we study multicommodity flows. Consider an undirected graph $G = (V, E)$ with an assignment of non-negative capacities to the edges, $c : E \rightarrow \mathbb{R}^+$. A *multicommodity flow* instance on $G$ is a set of ordered pairs of vertices $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$. Each pair $(s_i, t_i)$ represents a *commodity* with source $s_i$ and target $t_i$. The objective is to maximize the amount of flow traveling from the sources to the corresponding destinations, subject to the capacity constraints. The problem comes in two flavors. In the first, called the *maximum throughput* problem, the total flow, summed over all commodities, is to be maximized. The second is called the *maximum concurrent flow* problem. Here, for each commodity $(s_i, t_i)$ a non-negative demand $d_i$ is specified. The objective is to maximize the *fraction* of the demand that can be shipped simultaneously for all commodities. Both the maximum throughput problem and the maximum concurrent flow problem can be solved in polynomial time using linear programming.

Given a multicommodity flow instance together with demands, the *cut ratio* of a cut $(U, \bar{U})$ is defined as

$$R_U = \frac{c(U, \bar{U})}{d(U, \bar{U})} \quad \text{where} \quad d(U, \bar{U}) = \sum_{(s_i, t_i) \in (U \times \bar{U}) \cup (\bar{U} \times U)} d_i$$

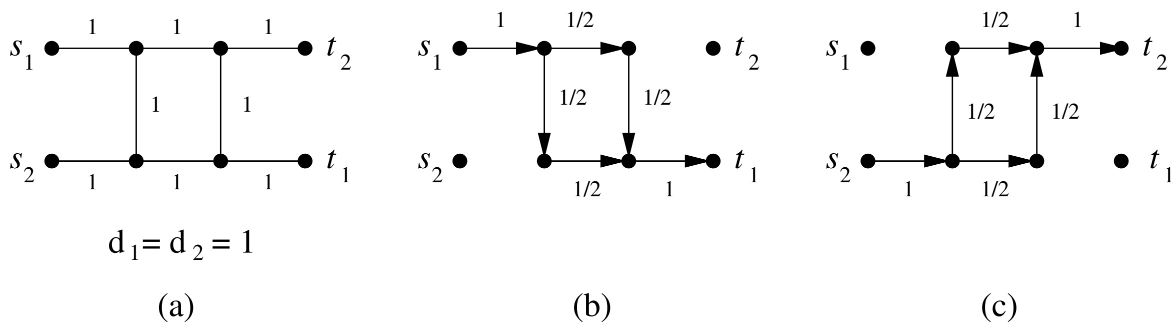


Figure 4: Solution to a 2-commodity flow problem (a). The routing of the first commodity is shown in (b) and the second commodity is shown in (c).

and the *min-cut ratio* is defined as $R = \min_{U \subseteq V} R_U$.

For every multicommodity flow $h$ consisting of flow $h_i$ for each commodity $i$, the *concurrent flow value* of $h$ is defined as $\min_i t_{h_i}/d_i$. The *concurrent max-flow* $f$ of a multicommodity flow instance $I$ is defined as the maximum over all concurrent flow values of feasible multicommodity flow solutions for $I$.

**Theorem 1.5 (Min-cut-ratio concurrent-max-flow theorem [1])** *For any instance of a concurrent multicommodity flow problem with $k$ commodities,*

$$\Omega\left(\frac{R}{\log k}\right) \leq f \leq R \ ,$$

*where $f$ is the concurrent max-flow and $R$ is the min-cut ratio of the problem. Finding such a cut can be done in random polynomial time.*

**Proof.** We only show that $f \leq R$. Consider any cut $(U, \bar{U})$ and let $i_1, i_2, \ldots, i_r$ denote the commodities whose source and target are separated by this cut. Since all flows for these commodities must cross $(U, \bar{U})$, we know that

$$\sum_{j=1}^{r} f \cdot d_{i_j} \leq c(U, \bar{U}) \ .$$

Since $\sum_{j=1}^{r} d_{i_j} = d(U, \bar{U})$, this means that

$$f \leq \frac{c(U, \bar{U})}{d(U, \bar{U})}$$

and therefore that the concurrent max-flow is upper bounded by the min-cut ratio. $\sqcap$

Also specific classes of concurrent multicommodity flow problems have been studied. In a *product multicommodity flow problem* (PMFP) we associate a non-negative *weight* $\pi(u)$ with each node $u \in V$. The demand for the commodity between nodes $u$ and $v$ is then set to be $\pi(u) \cdot \pi(v)$. Let $\mathcal{P}$ denote the subset of nodes for which $\pi$ is nonzero and let $p = |\mathcal{P}|$. Without loss of generality, we will assume that $\sum_{u \in V} \pi(u) = p$ (if not, scale $\pi$ correspondingly). Then the min-cut ratio for a PMFP is equal to

$$\min_{U \subseteq V} \frac{c(U, \bar{U})}{2\pi(U) \cdot \pi(\bar{U})} \ ,$$

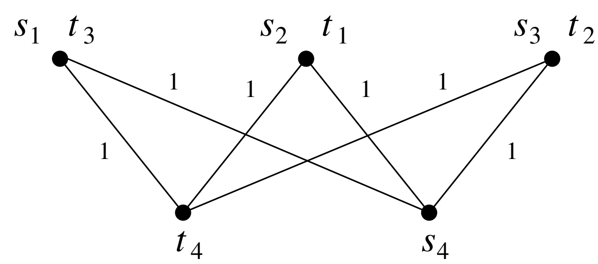


Figure 5: The Okamura-Seymour [7] example of a 4-commodity flow problem for which the concurrent max-flow is 3/4 and the min-cut-ratio is 1. In this example, all demands are one. The max-flow is attained by routing $1/4$ unit of commodity 4 on each of the three paths between $s_4$ and $t_4$ and $3/8$ units of commodity $i$ on each of the two paths between $s_i$ and $t_i$ for $1 \leq i \leq 3$.

where $\pi(U) = \sum_{u \in U} \pi(u)$.

**Theorem 1.6 ([6])** *For any PMFP with $p$ nodes of nonzero $\pi$,*

$$\Omega\left(\frac{R}{\log p}\right) \leq f \leq R \, ,$$

*where $f$ is the concurrent max-flow and $R$ is the min-cut ratio of the PMFP. Furthermore, a flow solution with such an $f$ can be found that only uses paths of maximum length*

$$O\left(\frac{\max\{\gamma, c(V)/p\} \log p}{p \cdot R}\right) \quad \textit{where} \quad \gamma = \max_{u \in V : \pi(u) > 0} \frac{c(u)}{\pi(u)} \, .$$

For the maximum throughput problem a different notion then the min-cut ratio is useful – that of the *minimum multicut*. A *multicut* is a subset of edges $E' \subseteq E$ whose removal disconnects all source-destination pairs. The *capacity* of a multicut $E'$ is the sum of capacities of the edges in $E'$. The minimum multicut value $C$ is defined as the minimum capacity over all multicuts separating all source-destination pairs. ($C$ can be seen as a generalization of the min-cut for single commodity flows.)

The *throughput* of a multicommodity flow $h$ with flow $h_i$ for each commodity $i$ is defined as $t_h = \sum_i t_{h_i}$. The *maximum throughput* $T$ of a multicommodity flow instance $I$ is the maximum throughput over all feasible multicommodity flows for $I$.

**Theorem 1.7 (Min-multicut max-throughput theorem [4])** *For any instance of a maximum multicommodity flow problem with $k$ commodities,*

$$\Omega\left(\frac{C}{\log k}\right) \leq T \leq C \, ,$$

*where $C$ is the minimum multicut value and $T$ is the maximum throughput of the problem. Such a multicut can be found in polynomial time.*

Thus, also for multicommodity flow problems the minimum cut is always closely related to the maximum flow. This will be useful when comparing various parameters that have been suggested to study the routing performance of networks.

## 1.6 Routing parameters

What are now suitable parameters to measure the ability of a network to support routing? A very popular parameter has been the *edge expansion* of a network. We define it here in a more general form than it is usually used:

**Definition 1.8** *Given a network $G = (V, E)$ with a non-negative capacity function $c$, the* edge expansion $\alpha$ *of $G$ is defined as*

$$\alpha = \min_{U \subseteq V} \frac{c(U, \bar{U})}{\min\{c(U), c(\bar{U})\}} .$$

But how well is this parameter able to capture the ability of a network to route unicast traffic? For this we need some suitable network flow parameter. Suppose that we are given a network with topology $G = (V, E)$ and edge capacities specified by a non-negative function $c$. Then each node $v \in V$ can inject and absorb at most $c(v)$ flow. Thus, it would be desirable to solve any multicommodity flow problem in which each node $v$ is the origin and destination of a demand of at most $c(v)$ as good as possible. This can be studied with the help of so-called balanced multicommodity flow problems. A *balanced multicommodity flow problem* (BMFP) is a multicommodity flow problem in which the sum of the demands of the commodities originating and the commodities terminating in a node $v$ is equal to $c(v)$ for every $v \in V$.

In order to measure how well BMFPs can be handled by a network, we introduce the flow number. Suppose we have a network $G = (V, E)$ with arbitrary non-negative edge capacities. Given a concurrent multicommodity flow problem with feasible solution $\mathcal{S}$, let the *dilation* $D(\mathcal{S})$ of $\mathcal{S}$ be defined as the length of the longest flow path in $\mathcal{S}$ and the *congestion* $C(\mathcal{S})$ of $\mathcal{S}$ be defined as the inverse of its concurrent flow value (i.e., the congestion says how many times the edge capacities would have to be increased in order to satisfy the demands of all commodities when using the same set of paths). Let $\mathcal{B}$ be the PMFP in which $\pi(v) = c(v)/\sqrt{c(V)}$ for every node $v$, that is, each pair of nodes $(v, w)$ has a commodity of demand $c(v) \cdot c(w)/c(V)$. Note that $\mathcal{B}$ is also a BMFP.

**Definition 1.9 ([5])** *The* flow number $F(G)$ *of a network $G$ is defined as the minimum over all feasible solutions $\mathcal{S}$ of $\mathcal{B}$ of* $\max\{C(\mathcal{S}), D(\mathcal{S})\}$.

In the case that there is no risk of confusion, we will simply write $F$ instead of $F(G)$. Note that the flow number of a network is invariant to a scaling of the capacities.

The flow number can be computed exactly via linear programming in polynomial time. Another advantage of the flow number is that, as shown by the next theorem, it can be applied to much more general multicommodity flow problems than just the one that defines it.

**Theorem 1.10** *For any network $G$ with flow number $F$ and any instance $I$ of the BMFP for $G$, there is a feasible solution for $I$ with congestion and dilation at most $2F$.*

**Proof.** The idea is to decompose $I$ into two multicommodity flow problems: for every commodity $i$ with source $s_i$ and destination $t_i$, the first problem $I_1$ has commodities $i_u$ from $s_i$ to $u$ for all $u \in V$ with demands $d_{i_u} = d_i \cdot c(u)/c(V)$, and the second problem $I_2$ has commodities $i'_u$ from $u$ to $t_i$ for all $u \in V$ with demands $d_{i'_u} = d_i \cdot c(u)/c(V)$. For every commodity $i$ from the original problem, the total demand of corresponding commodities in $I_1$ is $d_i$ and is $d_i$ in $I_2$ as well. Moreover, for every node $u \in V$ the fraction of the demand of commodity $i$ directed to $u$ in $I_1$ is equal to the fraction of commodity $i$ leaving $u$ in $I_2$.

Both of the flow problems $I_1$ and $I_2$ are PMFPs with $\pi(v) = c(v)/\sqrt{c(V)}$ for every node $v$, because for any pair $v, w \in V$, the total demand of the commodities with source $v$ and destination $w$ in $I_1$ is equal to

$$\sum_{i:\, s_i=v} \frac{d_i \cdot c(w)}{c(V)} = \frac{c(v) \cdot c(w)}{c(V)} = \pi(v) \cdot \pi(w) \ ,$$

and in $I_2$ is equal to

$$\sum_{i:\, t_i=w} \frac{d_i \cdot c(v)}{c(V)} = \frac{c(v) \cdot c(w)}{c(V)} = \pi(v) \cdot \pi(w) \ .$$

Therefore, both $I_1$ and $I_2$ represent the multicommodity flow problem $\mathcal{B}$ underlying the definition of the flow number. Thus, both $I_1$ and $I_2$ have a feasible solution with congestion and dilation at most $F$. Hence, the original problem $I$ has a feasible solution with congestion and dilation at most $2F$, which proves the claim. $\sqcap$

With techniques similar to those used in the proof of Theorem 5.0.3 in [8] one can also prove the following result.

**Theorem 1.11** *On average over all BMFPs $I$, the minimum $\max\{C(\mathcal{S}), D(\mathcal{S})\}$ over all feasible solutions $\mathcal{S}$ of $I$ is $\Omega(F)$.*

Hence, the flow number truthfully captures the problem of routing BMFPs in networks. Using Theorem 1.10, we prove another powerful result, called *Shortening Lemma*, that shows that the flow number allows to convert arbitrary multicommodity flow solutions into solutions with short flow paths.

**Theorem 1.12 (Shortening Lemma [5])** *Suppose we are given a network with flow number $F$. Then, for any $\epsilon \in (0, 1]$ and any feasible solution $\mathcal{S}$ to an instance of the concurrent multicommodity flow problem with a concurrent flow value of $f$, there exists a feasible solution with concurrent flow value $f/(1 + \epsilon)$ that uses paths of length at most $2 \cdot F(1 + 1/\epsilon)$.*

**Proof.** Given a solution $\mathcal{S}$ as above, let $\mathcal{S}' \subseteq \mathcal{S}$ consist of all paths from $\mathcal{S}$ that are longer than $L$, for $L = 2 \cdot F/\epsilon$. We are going to shorten the paths in $\mathcal{S}'$ at the cost of slightly decreasing the satisfied demand of each commodity.

For a path $p \in \mathcal{S}'$ between $s_p$ and $t_p$, let $a_{p,1} = s_p, a_{p,2}, \cdots, a_{p,L}$ denote its first $L$ nodes and $b_{p,1}, \cdots, b_{p,L-1}, b_{p,L} = t_p$ its last $L$ nodes and let $f_p$ be the size of the flow along $p$. Then the set $\mathcal{U} = \bigcup_{p \in \mathcal{S}'} \bigcup_{i=1}^{L} \{a_{p,i}, b_{p,i}, f_p\}$ is (a subset of) an instance of the BMFP. By Theorem 1.10, there exists a feasible solution $\mathcal{P}$ to $\mathcal{U}$ with flow value at least $1/(2F)$ consisting of paths of length at most $2F$. We are going to combine the initial and final parts of the long paths in $\mathcal{S}'$ with these “shortcuts” in $\mathcal{P}$ to obtain the desired short solution.

First, decrease the flows along all paths $p \in \mathcal{S}$ by a factor of $1/(1 + \epsilon)$ so that we have room to accommodate new, short paths for the paths in $\mathcal{S}'$. These short paths are constructed in the following way:

For every path $p \in \mathcal{S}'$, we replace $p$ by $L$ flow systems $S_{p,i}$, $i = 1, \cdots, L$. Each flow system $S_{p,i}$ consists of two parts:

1. the flow paths between $a_{p,i}$ and $b_{p,i}$ in $\mathcal{P}$ corresponding to the request $\{a_{p,i}, b_{p,i}, f_p\}$ from $\mathcal{U}$, now with a flow of $f_p/(L(1+\epsilon))$, and

2. $f_p/(L(1+\epsilon))$ units of flow between $a_{p,1}$ and $a_{p,i}$ along $p$, and $f_p/(L(1+\epsilon))$ units of flow between $b_{p,i}$ and $b_{p,L}$ along $p$.

For each $i$, the length of each path in the subsystem $S_{p,i}$ is at most $L + 2 \cdot F$, and $f_P/(L(1+\epsilon))$ units of flow are shipped along each path system $S_{p,i}$. Summed over all $i = 1 \dots L$, we have $f_p/(1+\epsilon)$ units of flow between $s_p = a_{p,1}$ and $t_p = b_{p,L}$, which is as high as the original flow through $p$ reduced by $1/(1+\epsilon)$. Hence, we can replace $p$ by the systems $S_{p,i}$ without changing the amount of flow from $s_p$ to $t_p$.

Now, it holds for every edge $e$ that the flow traversing $e$ due to the paths in $\mathcal{S}$ is at most $c(e)/(1+\epsilon)$, and due to the shortcuts in $\mathcal{P}$ is at most

$$\sum_{p \in \mathcal{P}\colon e \in p} \frac{f_p}{L(1+\epsilon)} \le \frac{2F}{L(1+\epsilon)} \cdot c(e) = \frac{\epsilon \cdot c(e)}{1+\epsilon} ,$$

since

$$\sum_{p \in \mathcal{P}\colon e \in p} \frac{f_p}{2F} \le c(e) .$$

Thus, the flows in $\mathcal{S}$ and $\mathcal{P}$ sum up to at most $c(e)$ for an edge $e$. Therefore, the modification yields a feasible solution satisfying the desired properties. $\sqcap$

## 1.7 Flow number vs. edge expansion

Now we are ready to compare the flow number with the edge expansion and are therefore able to answer how well the edge expansion allows to capture the ability of a network to support routing. Using Theorem 1.6 we can prove the following result.

**Theorem 1.13** *For any network $G$ with edge expansion $\alpha$ it holds for its flow number $F$ that*

$$\alpha^{-1} \le F \le O(\alpha^{-1} \log n) .$$

**Proof.** First we prove that $F \ge \alpha^{-1}$. Let $f$ be the concurrent max-flow of the problem $\mathcal{B}$ used for the definition of $F$. Then it holds according to the proof of Theorem 1.5 that for any set $U$,

$$f \le \frac{c(U, \bar{U})}{d(U, \bar{U})} .$$

For $\mathcal{B}$ it holds that

$$d(U, \bar{U}) = \sum_{(u,v) \in (U \times \bar{U}) \cup (\bar{U} \times U)} \frac{c(u) \cdot c(v)}{c(V)} = \frac{2c(U) \cdot c(\bar{U})}{c(V)} .$$

We distinguish between two cases. If $c(U) \ge c(V)/2$, then $c(U) \cdot c(\bar{U})/c(V) \ge c(\bar{U})/2$. Thus,

$$f \le \frac{c(U, \bar{U})}{2 \cdot c(\bar{U})/2} = \frac{c(U, \bar{U})}{\min\{c(U), c(\bar{U})\}} .$$

If $c(\bar{U}) \geq c(V)/2$, then $c(U) \cdot c(\bar{U})/c(V) \geq c(U)/2$ and therefore

$$f \leq \frac{c(U,\bar{U})}{2 \cdot c(U)/2} = \frac{c(U,\bar{U})}{\min\{c(U), c(\bar{U})\}} \ .$$

Hence, in both cases,

$$f \leq \frac{c(U,\bar{U})}{\min\{c(U), c(\bar{U})\}}$$

and therefore $f \leq \alpha$ or $1/f \geq \alpha^{-1}$. Using the fact that $F \geq 1/f$ it follows that $F \geq \alpha^{-1}$.

Next we show that $F = O(\alpha^{-1} \log n)$. Consider the PMFP $\mathcal{B}$ and let $R$ be the min-cut ratio for it. Recall the notation used in Theorem 1.6. We require there that $p = \sum_{u \in V} \pi(u)$. In our case, $\sum_{u \in V} \pi(u) = \sum_{u \in V} c(u)/\sqrt{c(V)} = \sqrt{c(V)}$, but since $F$ is invariant to scaling we can scale the capacities so that $\sqrt{c(V)} = n$ without changing $F$. Using the definitions of the minimum cut-ratio and the weighted edge expansion it holds that

$$R = \min_{U \subseteq V} \frac{c(U,\bar{U})}{2\pi(U) \cdot \pi(\bar{U})} = \min_{U \subseteq V} \frac{c(U,\bar{U})}{2c(U) \cdot c(\bar{U})/c(V)} \geq \min_{U \subseteq V} \frac{c(U,\bar{U})}{2\min\{c(U), c(\bar{U})\}} = \alpha/2$$

because $c(U) \cdot c(\bar{U})/c(V) \leq \min\{c(U), c(\bar{U})\}$. Furthermore, we have $\gamma = c(u)/\pi(u) = \sqrt{c(V)}$ and $c(V)/p = \sqrt{c(V)}$. Thus, according to Theorem 1.6 there is a solution to the PMFP $\mathcal{B}$ such that $L = O((\sqrt{c(V)} \log n)/(nR)) = O((\log n)/R) = O(\alpha^{-1} \log n)$ and $f = \Omega(R/\log n) = \Omega(\alpha/\log n)$, which implies the desired $F = O(\alpha^{-1} \log n)$. $\sqcap$

The relationship between $F$ and $\alpha$ is tight, since there are examples in which $F = \Theta(\alpha^{-1})$ and $F = \Theta(\alpha^{-1} \log n)$ [5].

## 1.8 Modeling point-to-point communication by network flows

Now that we introduced several parameters, the question is how useful they are to study problems such as the routing problem we considered for the cycle above. That is, imagine we have a set of messages we want to send through the network, how much do the parameters tell us about the time that would be necessary for this?

An easy way to approach the problem of sending information in a network is to simply treat information as a continuous flow. This will allow to forget about scheduling problems and therefore to concentrate on path selection and rate control problems. To illustrate this situation, imagine there is a message of size $s$ that has to be sent along a flow path of length $\ell$ of value $f$. Then message pieces of size $f$ can be sent along the flow path in a pipelined fashion, giving a total number of

$$\lceil s/f \rceil + \ell - 1 \tag{1}$$

time steps to transmit the message. In general, we can prove the following theorem.

**Theorem 1.14** *Let $G$ be a graph with flow number $F$ and non-negative edge capacities given by $c$. Suppose that information can be sent through the network as a continuous flow. Then for any routing problem in which each node $v$ has at most $d(v)$ data to transmit and receive, at most*

$$O\left(F \cdot \max_{v} \frac{d(v)}{c(v)}\right)$$

*time is necessary to send all the data to their destinations.*

**Proof.** Any routing problem of the kind stated in the theorem can be represented as a set of source-destination pairs $(s_i, t_i)$ with demands $d_i$ that have the property that for every node $v$ the sum of the demands with source resp. destination $v$ is at most $d(v)$. Now, let $q = \max_v d(v)/c(v)$. If we divide every $d_i$ by $q$, then we arrive at a multicommodity flow problem with demands $d'_i$ so that the total demand originating from and destined for a node $v$ is at most $c(v)$. Hence, we have a subset of a BMFP that, according to Theorem 1.10, has a flow solution $\mathcal{S}$ with congestion and dilation at most $2F$. This means that every pair $(s_i, t_i)$ of demand $d'_i$ has a flow of value at least $d'_i/(2F)$ shipped along paths of length at most $2F$ that obey the capacity constraints. Hence, using (1), it takes at most $4F$ time steps to ship all the demands $d'_i$. Raising the $d'_i$ now back to $d_i$ gives the claimed time bound. $\sqcap$

As we will see later, there are networks of constant degree and uniform edge capacities with $F = O(\log n)$. Hence, these networks can get remarkably close to a best possible transmission time of $\max_v d(v)/c(v)$.

# References


[1] Y. Aumann and Y. Rabani. An $O(\log k)$ approximate min-cut max-flow theorem and approximation algorithm. *SIAM Journal on Computing*, 27(1):291–301, 1998.

[2] L. Ford and D. Fulkerson. Sur le probléme des courbes gauches en topologie. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[3] L. Ford and D. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, 1962.

[4] N. Garg, V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM Journal on Computing*, 25:235–251, 1996.

[5] P. Kolman and C. Scheideler. Improved bounds for the unsplittable flow problem. In *ACM/SIAM Symp. on Discrete Algorithms (SODA)*, 2002.

[6] T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46(6):787–832, 1999.

[7] H. Okamura and P. Seymour. Multicommodity flows in planar graphs. *Journal of Combinatorial Theory B*, 31:75–81, 1981.

[8] C. Scheideler. *Universal Routing Strategies for Interconnection Networks*, volume 1390 of *Lecture Notes in Computer Science*. Springer, 1998.