College of Computer Science
Northeastern University
COM 3810: Distributed Network Algorithms

Lecture 4
April 17, 2002
Scribe: Guolong Lin

Multicommodity Flow

Lecture Outline:

- Analysis of *Awerbuch-Leighton's* edge balancing algorithm [AL93].
- Network design problems for provisioning virtual private networks (VPNs)

1 Multicommodity flow

We are given an undirected graph $G = (V, E)$, in which there are capacity constraints on each edge $c(e)$. We have K commodity flows. Each flow is between a source s_i and a sink t_i, $1 \leq i \leq K$. Flow i has demand d_i. The problem is to find flow paths which meet all the commodity demands d_i subject to the constraints of edge capacities, if possible. Alternatively, find the largest fraction μ such that μd_i amount of commodity i, for each i, can be shipped from s_i to t_i, subject to capacity constraints. Our notation can be summarized as follows.

- V : set of vertices (nodes).
- E : set of edges.
- M : number of edges, ie, $|E|$.
- K : number of commodities.
- s_i, t_i : source and sink for commodity i.
- $c(e)$: capacity of edge e.
- d_i : demand for commodity i.

2 Awerbuch-Leighton's algorithm

Awerbuch and Leighton propose an elegant local-control algorithm for the above problem. The algorithm is based on a simple "edge balancing" technique. Each node maintains K kinds of queues, one for each commodity. For each commodity i, there are δ_i queues, one per incident edge, at each node i, where δ_i is the number of incident edges of i. The algorithm attemps to balance each commodity i across each edge $e = (u \rightarrow v)$ if the queue size at u is larger than that at v. More formally, the node n has some *potential* associated with each queue of commodity i:

$$\phi_i = \sum_{1 \leq j \leq \delta_i} (\frac{q_i}{d_i})^2$$

where q_i is the size of each queue at node n for commodity i. Remember that there are δ_i such queues, all have the same queue size after "rebalancing" (see later for details).

Intuitively, each packet has some "potential" equal to its height h (think of gravity), then the sum of potential of all the packets in a queue with height q_i is $1 + 2 + \cdots + q_i \simeq \frac{1}{2}q_i^2$, normalized by $(1/2)d_i^2$, we get definition of ϕ_i shown above. The normalization is needed to keep high-demand commodities from swamping out lower-demand commodities.

The algorithm operates by trying to minimize the **global potential** (sum of ϕ_i over all commodity $i \in K$ and all nodes $n \in V$). So naturally, we want to "push" packets across each edge from "tall" queue to "short" queue.

2.1 Algorithm in detail

Phase 1: *Add new flow to each sources.* Add $(1+\epsilon)d_i\delta_i^{-1}$ units of flow to each of the δ_i queues for commodity i at the source s_i. In total, we add $(1+\epsilon)d_i$ units of commodity i.

Phase 2: *Push flow across each edge so as to balance the queues as much as possible.* For each edge e, define

$$\Delta_i(e) = q_i(tail(e)) - q_i(head(e))$$

If we push f_i units of commodity i across e, the decrease of **global potential** is

$$2f_i(\Delta_i(e) - f_i)d_i^{-2} \tag{1}$$

We want to maximize the total decrease on edge e

$$\sum_{1 \le i \le K} 2f_i(\Delta_i(e) - f_i)d_i^{-2}$$

subject to the constraints:

$$f_i \ge 0 \text{ for all } 1 \le i \le K, \tag{2}$$
$$\sum_{1 \le i \le K} f_i \le c(e), \tag{3}$$
$$f_i = 0 \text{ if } c(e) \le \epsilon\, d_i/M. \tag{4}$$

Note that the last constraint indicates that commodity i does not flow through any edge whose capacity is at most $\varepsilon d_i/M$. Imposing such a condition only reduces the demand that can be routed for each commodity by an ε factor. We impose this condition for technical reasons and will use this aspect of the algorithm in our analysis.

The reader is referred to [AL93] on the details for computing f_i.

Phase 3: *Remove flow from sinks.* Zero out all queues for commodity i at sink t_i $(1 \le i \le K)$.

Phase 4: *Rebalance at nodes.* Reallocate each commodity within each node so that the queues for commodity i ar all equal within each node.

2.2 Analysis of the algorithm

2.2.1 High level description

We are going to show that this algorithm is stable, or

Theorem 1 *(A-L Algorithm is stable)* *At any time, the commodities remaining in the network are bounded provided that there exists feasible flows to route $(1+2\epsilon)d_i$ units of commodity $i, (1 \leq i \leq K)$ without using any edge of small capacity $c(e) \leq \epsilon d_i/M$.*

This is shown by bounding the **global potential** of the network. Before going into technical details, let's briefly discuss the logic of the analysis an a high level first. The **global potential** function is given by

$$\phi = \sum_{e \in E} \sum_{1 \leq i \leq K} \left(\frac{q_i(head(e))}{d_i}\right)^2 + \left(\frac{q_i(tail(e))}{d_i}\right)^2.$$

The logic of the argument is as follows: if the global potential is small at the beginning of a round, then it does not increase by more than a small amount; if the potential is large at the beginning of a round, then it decreases during the round. Combining these two, we will be able to show that ϕ is bounded.

More specifically, for each commodity i, we will obtain a queue height q_i^* (in terms of ϵ, K, L, M, d_i), such that whenever there is a queue for commodity i with height q_i^*, then the potential drops. This is argued under two cases (q_i here refers only to the queue size for commodity i at source s_i):

- Case 1:

$$\sum_{1 \leq i \leq K} \frac{q_i}{d_i} > X_1.$$

- Case 2:

$$\sum_{1 \leq i \leq K} \frac{q_i}{d_i} \leq X_1.$$

We also obtain an upper bound X_2 on the increase of potential at each round.

Then, the **global potential** at any time is bounded by:

$$2MK(q_i^*/d_i)^2 + X_2.$$

2.2.2 More details

In what follows, we fill in the missing details (analysis of Cases 1 and 2) and find out the values of q_i^*, X_1, X_2, etc.

For the four phases of *Awerbuch-Leighton* algorithm, only **Phase 1** increases potential, the other three decrease it. During Phase 1, each of those δ_i queues at s_i increases in size from q_i to $q_i + \frac{(1+\epsilon)d_i}{\delta_i}$ for $i \in [1, K]$. The total increase in potential is thus

$$\sum_{1 \leq i \leq K} \delta_i \left[\left(\frac{q_i + (1+\epsilon)d_i\delta_i^{-1}}{d_i}\right)^2 - \left(\frac{q_i}{d_i}\right)^2\right]$$

$$= \sum_{1 \le i \le K} \left[\frac{2(1+\epsilon)q_i}{d_i} + \frac{(1+\epsilon)^2}{\delta_i} \right]$$

$$= 2(1+\epsilon) \sum_{1 \le i \le K} q_i d_i^{-1} + (1+\epsilon)^2 \sum_{1 \le i \le K} \delta_i^{-1}$$

$$\le 2(1+\epsilon) \sum_{1 \le i \le K} q_i d_i^{-1} + (1+\epsilon)^2 K. \tag{5}$$

During Phase 2, since (our assumption) there exists a feasible solution with demands $\{(1+2\epsilon)d_i\}$ for which no commodity i uses an edge e with capacity $c(e) \le \frac{\epsilon d_i}{M}$, we consider the potential drop due to this feasible solution. The potential drop using the A-L algorithm will be at least as big because it tries to maximize the drop at each round.

In the feasible solution, define L to be the length of the longest flow path, and let $g_i(e)$ be the amount of commodity i being pushed across edge e. The drop in potential is easily computed as

$$\frac{2g_i(e)(\Delta_i(e) - g_i(e))}{d_i^2}.$$

And the **global potential** drop is (refer to [AL93] for detailed derivation)

$$\sum_{e \in E} \sum_{1 \le i \le K} \frac{2g_i(e)(\Delta_i(e) - g_i(e))}{d_i^2}$$

$$\cdots\cdots$$

$$\ge 2(1+2\epsilon) \sum_{1 \le i \le K} \frac{q_i}{d_i} - 2(1+2\epsilon)^2 KL + 2(1+2\epsilon)(1+\epsilon) \sum_{1 \le i \le K} \delta_i^{-1}. \tag{6}$$

Phases 3 and 4 can not increase the potential. So combining equations 5 and 6, the overall change in potential during each round is at most

$$-2\epsilon \sum_{1 \le i \le K} \frac{q_i}{d_i} + 2(1+2\epsilon)^2 KL.$$

Thus, the potential decreases if

$$\sum_{1 \le i \le K} \frac{q_i}{d_i} > (1+2\epsilon)^2 \epsilon^{-1} KL.$$

Up to now, we have analyzed Case 1 and obtained the value for X_1:

$$X_1 = (1+2\epsilon)^2 \epsilon^{-1} KL. \tag{7}$$

Now we are ready to analyze Case 2 by assuming

$$\sum_{1 \le i \le K} \frac{q_i}{d_i} \le (1+2\epsilon)^2 \epsilon^{-1} KL$$

we are going to show that the potential drops if any particular queue is larger than q_i^*, whose value will be obtained later on.

Before that, we want to solve the upper bound X_2. From the analysis (equation 5) of Phase 1, as well as our assumption above (if the assumption is not true, no potential increase at all), we know that the increase in potential at each round (due to new flow) is at most

$$2(1+\epsilon)(1+2\epsilon)^2\epsilon^{-1}KL + (1+\epsilon)^2K \leq 2(1+2\epsilon)^3\epsilon^{-1}KL.$$

Thus, we get the upper bound promised previously:

$$X_2 = 2(1+2\epsilon)^3\epsilon^{-1}KL. \tag{8}$$

Let's now consider some queue for commodity i at node v_0 that has height q_i^*. We can push a flow of magnitude $\frac{\epsilon d_i}{M}$ from $v_0 \to \cdots \to s_i \to \cdots \to t_i$ (remember that we don't use small capacity edges). Summing the hop-wise potential drops (Equation 1), we get the potential drop of at least (note that queue size at v_1 is q_i^*, and 0 at t_i)

$$\frac{2g(q_i^* - 2Ng)}{d_i^2},$$

where $g = \frac{\epsilon d_i}{M}$. Again, we are using the argument that A-L algorithm tries to maximize the potential drop at Phase 2, thus the drop is at least as large as any feasible flow's drop.

Hence, the potential will decrease during the round if

$$\frac{2g(q_i^* - 2Ng)}{d_i^2} > X_2$$

or

$$\frac{q_i^*}{d_i} > (1+2\epsilon)^3\epsilon^{-2}KLM + \frac{2N\epsilon}{M}. \tag{9}$$

So, we have analyzed Case 2 and got values for X_2 as well as q_i^*.

2.2.3 Wrapping Up

We have proved that whenever there is a queue for commodity i with height q_i^* satisfying Equation 9, then the potential of the system drops during the round. Note that the particular argument for the potential drop depends on the value of $\sum_{1\leq i\leq K}\frac{q_i}{d_i}$ (Case 1 or 2). We have also given an upper bound X_2 at Equation 8. This means that the maximum potential in the system at any time is at most

$$\phi_{max} \leq 2MK(\frac{q_i^*}{d_i})^2 + X_2.$$

The amount of commodity i in the system at any time is at most W_i where

$$2M(\frac{W_i}{2Md_i})^2 \leq \phi_{max}.$$

Using relevant equations obtained previously, we are able to bound W_i for any $i \in [1, K]$. This completes the proof that A-W algorithm is stable.

2.2.4 Comments

- The reader is encouraged to read the original paper [AL93] for more details.
- This lecture note is prepared by combining my personal class note and my understanding of the paper [AL93], so various presentation details may be a bit different from what you saw in class, but the overall idea is the same.
- There is another tighter bound analysis by using *exponential* (instead of quadratic) potential function ϕ in another paper [AL94] by the same authors.

3 Network design for provisioning VPNs

The multicommodity flow problem we studied is a network design problem, where we are given source-destination pairs with associated demands and the goal is to identify a set of routes or flows for these demands that satisfy the capacity constraints of the network. One can thus use a solution to such a problem to reserve appropriate bandwidth on the relevant links so as to set up a virtual private network (VPN) among the source-destination pairs.

We now consider another interesting variant of the VPN provisioning problem in which we are not given the demands between every pair of terminals in the VPN; instead, we are given a single number for each terminal that is an upper bound on the traffic associated with the terminal. Such a model is referred to as the *hose model*; the model discussed in the preceding paragraph, in which we are given demands for each pair, is referred to as the *pipe model*.

We start with a *symmetric* version of the problem in which we assume that the communication between any two nodes is bidirectional. We are given an undirected graph $G = (V, E)$, with a subset of the nodes as terminals and the remaining nodes as routers. Each terminal i has a symmetric bandwidth requirement (upper bound) of $f(i)$. The goal is to identify paths between terminal pairs and reserve bandwidth x_e for any edge e on the paths such that any traffic pattern that satisfies the bandwidth constraint $f(i)$ can be routed, and $\sum_e c(e)x_e$ is minimized, where $c(e)$ is the unit cost of reserving edge e's bandwidth.

We will consider two versions of the above problem. One is $SymT$, where we require that the set of edges on which bandwidth is reserved forms a tree. The second is $SymG$, in which we allow the set of edges on which bandwidth is reserved to be an arbitrary subset of E. We will discuss $SymT$ and $SymG$ next week.

References

[AL93] Baruch Awerbuch and Frank Thomson Leighton. A simple local-control approximation algorithm for multicommodity flow. In *IEEE Symposium on Foundations of Computer Science*, pages 459–468, 1993.

[AL94] Baruch Awerbuch and Frank Thomson Leighton. Improved approximation algorithms for the multi-commodity flow problem and local competitive routing in dynamic networks. In *ACM Symposium on Theory of Computing*, pages 487–496, 1994.