Lecture 17

Network Coding Bounds

November 29th, 2004
Lecturer: Kamal Jain
Notes: Daniel Lowd

17.1 Network Coding

Start with a directed acyclic graph (DAG) with a single sender and many receivers, where each receiver has k-edge connectivity from the sender. We assume that each edge has unit capacity, say, 1 bit/second.

If our initial DAG does not meet this criterion, we can easily construct one that does: first, divide edge capacities by their greatest common divisor, then replace each c-capacity edge with c unit-capacity edges. Our unit capacities are no longer in the same units as before, but this is not a problem: the relative capacities have been maintained.

Under these conditions, each recipient can receive k bits/second from the sender.

Today, we prove two bounds on q, the field size of the bits sent. For ordinary, Boolean bits, $q = 2$, but we may need a larger field size to handle conflicts among the paths shared by different receivers.

First, we will prove the upper bound $q = O(\text{ \# receivers})$. Second, we will demonstrate the lower bound $q = \Omega(\sqrt{\text{ \# receivers }})$.

17.1.1 Upper bound

For ease of discussion, we consider a DAG with one sender and three receivers: r (red), b (blue), and g (green). However, all methods demonstrated will apply to an arbitrary number of receivers. In this DAG, there are k edge-disjoint paths from the sender to r, from the sender to b, and from the sender to g. We name the bits being sent by the receiver $b_1, b_2, \ldots, b_k$.

Theorem 17.1. *The field size q is $O(k$ \# receivers$)$.*

We first state and proof an algebra theorem that will be helpful in fixing this upper bound:

Theorem 17.2. *Consider a vector space V of dimension d over $GF(q)$. This vector space cannot be written as a union of fewer than or equal to q proper subspaces.*

Proof. Represent V as the union of p subspaces:

$$V = V_1 \cup V_2 \cup \cdots \cup V_p$$

Since $p > 1$, and 0 is an element of each subspace, we must double-count it at least once. Therefore, the total number of elements in the union is at most: $pq^{(d-1)} - 1$. (The $(d-1)$ exponent is because these are proper subspaces.) This expression can never be smaller than the size of V, which is q^d:

$$pq^{(d-1)} - 1 \geq q^d$$

This implies the following strict inequality:

$$pq^{d-1} > q^d$$

Solving for p, we find that $p > q$. □

We now proceed to prove an upper bound on q:

Proof. Initially, we assign one bit to each path to r, one bit to each path to b, and one bit to each path to g. Of course, since the paths to each receiver may overlap, the assignments may be inconsistent: one link of a path shared by r and b may simultaneously be assigned b_1 and b_2, for example. The following algorithm will resolve these conflicts until consistent assignments have been found for each edge, while maintaining the property that each receiver receives all the information sent.

We draw a k by k matrix for each receiver. Each row corresponds to one of the k paths from the sender to the receiver. Each column corresponds to one of the k bits being sent. For example, if $k = 4$ and the third row of r's matrix is the vector $[0\ 1\ 2\ 0]$, then the sender will send the vector $b_2 + 2b_3$ along the third edge-disjoint path to r. Note that as long as this matrix is non-singular, the receiver can reconstruct all k bits.

At the outset, before any conflicts are resolved, each matrix is the identity: one bit for each path. We wish to resolve all conflicts between receivers while maintaining non-singularity.

Using the original DAG, we can list the edges in a topologically sorted order, so that if edge a appears before edge b in any path, then a appears before b in the sort. Our algorithm will resolve conflicts, edge by edge, in this order. Once all conflicts have been resolved for a particular edge, we never need to reconsider that edge.

To resolve a conflict on a particular edge, we assume that each of up to k receivers has a different vector they wish to send along that edge. WLOG, let us focus on the receivers r, b, and g, and assume that the edge in question is along the kth path for each. Name their initial vectors v_r^k, v_b^k, and v_g^k. What requirements will we place on these vectors?

We first require that each receiver receives all information, i.e.:

$$\begin{aligned} v_r^1, v_r^2, v_r^3, \ldots, v_r^k &= (GF(q))^k \\ v_b^1, v_b^2, v_b^3, \ldots, v_b^k &= (GF(q))^k \\ v_g^1, v_g^2, v_g^3, \ldots, v_g^k &= (GF(q))^k \end{aligned}$$

We want to find a new vector $v = v'^k_r = v'^k_b = v'^k_g$, such that:

1. Each receiver still has complete information, using the new vector. Specifically,

$$\begin{aligned} v_r^1, v_r^2, v_r^3, \ldots, v'^k_r &= (GF(q))^k \\ v_b^1, v_b^2, v_b^3, \ldots, v'^k_b &= (GF(q))^k \\ v_g^1, v_g^2, v_g^3, \ldots, v'^k_g &= (GF(q))^k \end{aligned}$$

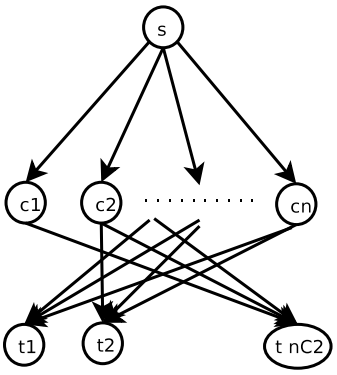

Figure 17.1: Network topology requiring a field size q that is $\Omega(\sqrt{t})$. Each receiver t_i is connected to all but two of the n intermediate nodes.

2. The information to pass along this edge can be constructed from the information on the incoming paths. Specifically,
$$v \in \text{lin span}(v_b^k, v_b^k, v_g^k)$$

We now apply Theorem 17.2 to show that a valid vector does in fact exist. Let $L = \text{lin span}(v_r^k, v_b^k, v_g^k)$. Let L_r refer to the set of objectionable vectors in L, according to receiver r. "Objectionable vectors" are those that are linear combinations of r's other vectors, and thus reduce the rank of r's matrix:

$$L_r = \{v \in L | v \in \text{lin span}(v_r^1, v_r^2, \ldots, v_r^{k-1})\}$$
$$L_r = L \cap \text{lin span}(v_r^1, v_r^2, \ldots, v_r^{k-1})$$

Clearly, L_r is a space since it's the intersection of two vector spaces. Furthermore, it's a proper subspace of L since $v_r^k \notin L$, but $v_r^k \in L$.

Choose field size q such that $L \neq L_r \cup L_b \cup L_g (\cup \cdots \cup L_t)$. By the Theorem 17.2, this must be true for all $q \geq t$, the number of receivers (which is the maximum number of subspaces in the union).

□

The running time for this algorithm scales with the number of flows.

17.1.2 Lower bound

Theorem 17.3. *The field size q is $\Omega(\sqrt{t})$.*

Proof. Construct a graph as in Figure 17.1 with a single sender, n intermediate nodes, and $t = \binom{n}{2}$ receivers. From the sender to each intermediate edge, there is an edge c_i. Each receiver t_j receives information from all but 2 of the n intermediate nodes, so $k = n - 2$.

Let $C = \{c_1, c_2, c_3, \ldots, c_n |$ possible sequences that can be observed on n edges $\}$. C is a code with minimum distance 3. By the perfect code bound, $q^k[1 + (q-1)n] \leq q^{k+2}$. q^k is the number of non-overlapping balls, $1 + (q-1)n$ is the volume per ball, and q^{k+2} is the total volume of the space.

From this, we can derive: $1 + (q-1)n \leq q^2$, so $q \geq n - 1$. Recall that $t = \binom{n}{2}$, so $q = \Omega(\sqrt{t})$. □

With a single sender, this bound is optimal. With multiple senders, it is unknown whether or not this bound is optimal.

17.2 Introduction to Lattices

Recall the following previous result: given a convex set S such that

1. S has a strong separation oracle.
2. S has a ball of radius ϵ.
3. S is contained in an ellipsoid E (given) of volume R

we can then find a point inside S, in time polynomial in dimension and $\log(R/\epsilon)$.

But what if we lack the elipsoid E, or the ball of radius ϵ? Given a convex set S such that

1. S has a strong separation oracle.
2. S There is a rational point in S with encoding length ϕ.

we can then find a point inside s in time polynomial in dimension and ϕ.

Furthermore, there also exists an optimal solution with small encoding length to the linear program:

$$\begin{array}{c} \min \sum_i cx_i \\ \forall j \sum_i a_{ij}x_i \geq b_j \end{array}$$

This solution can be found by using cosets divided by the determinant, for the corner point solution.

Definition 17.1. A *lattice* is an integer span $\{\vec{a_1}, \vec{a_2}, \ldots, \vec{a_n}\} = \{z_1\vec{a_1} + z_2\vec{a_2} + \cdots + z_n\vec{a_n} | z_i \in \mathcal{Z}\}$, where each $\vec{a_i}$ is a rational vector.

The requirement that each $\vec{a_i}$ is a rational vector is necessary – otherwise, the span could be dense. For example, the single-dimensional span of $\{1, \sqrt{2}\}$ is dense.

Definition 17.2. If $L = \text{lin span}\{b_1, \ldots, b_m\}$ and $b_1, \ldots, b_m$ are all linearly independent, then $b_1, \ldots, b_m$ are called a *basis* of L.

In one dimension, the basis is simply the greatest common divisor of the rationals $a_1, \ldots, a_n$. Therefore, the basis can be seen as a generalization of the gcd.

Lemma 17.4. *Given a lattice in d-dimension by vectors $\vec{a_1}, \ldots, \vec{a_n}$, we can find the basis of L, $\vec{b_1}, \ldots, \vec{b_m}$, in polynomial time.*

Because gcd is a special case, this algorithm is not strongly polynomial.

If the set isn't linearly independent, then you can find $x_1, \ldots, x_n \in \mathcal{Z}$ such that $\sum_i x_i\vec{a_i} = 0$. If $x_1 = \pm 1$, then $\vec{a_1}$ is a linear combination of $\vec{a_2}, \ldots, \vec{a_n}$. But this may not be true, such as in the example $\{6, 10\}$.

We will prove this lemma next time.