CS 388R: Randomized Algorithms, Fall 2021 September 28

Lecture 10: Routing

Prof. Eric Price *Scribe: Lucas Gretta, Aditya Parulekar*

NOTE: THESE NOTES HAVE NOT BEEN EDITED OR CHECKED FOR CORRECTNESS

1 Overview

In the last lecture we talked about limited independence, universal and pairwise independent hash families, and concluded our discussion of balls and bins.

In this lecture we discuss a special case of the routing problem.

2 Routing

Suppose we have a network through which we want to pass messages. In particular, each node i might have a message for another node j, and we need to send messages through the edges of the graph in such a way that at each timestep, at most one message passes through any given edge. How do we send the messages so that we minimize the time T taken for all messages to reach their destination?

We look at the case where the graph is the hypercube. The hypercube graph is defined as follows: There are $N = 2^n$ nodes, and each node is labeled by n bits. There is an edge between two nodes if their label differs by exactly one bit. For $n = 2$, this graph is the square, for $n = 3$, it is a cube, and so on.

Further, we only consider the case where the required routing is a permutation: that is, for any node x, we want to route a message to $\pi(x)$, where π is some permutation.

2.1 Bounds on Routing

We attempt to establish some lower bounds on routing

- $T \geq n$, as if we send x to $\neg x$ it takes $\geq n$ steps to reach ignoring congestion, so overall it must take $\geq n$ steps to send all messages.
- An idea is to look at the "total amount of work." The average distance in a random permutation is $n/2$ in expectation, so $Nn/2$ total work in expectation, and we can do at most Nn work per round, so it takes $\geq 1/2$ steps, which is absolutely useless!

Its not that clear how to get better bounds.

3 Bit Fixing Algorithm

Lets fix routes for each send-receive pair, once we have fixed these routes then at each timestep we can run through each message, and move it along the edge specified by the predetermined route, delaying if we already sent a message through this edge on this timestep.

To determine these routes, for each send-receive pair $(x, \pi(x))$, we find the first bit that differs between x and $\pi(x)$, and "correct" it by sending x along the edge corresponding to flipping that bit.

Analysis

It turns out that this algorithm does fairly poorly in the worst case.

Consider the send-receive pair

$$\overbrace{X}^{n/2-1 \text{ bits}}\ 00\ \overbrace{000\ldots0}^{n/2-1 \text{ bits}} \longrightarrow \overbrace{000\ldots0}^{n/2-1 \text{ bits}}\ 11\ \overbrace{X}^{n/2-1 \text{ bits}}$$

where X has exactly $n/4$ 1s.

There are $\binom{n/2-1}{n/4} \approx \frac{2^{n/2}}{\sqrt{n}} = \sqrt{\frac{N}{n}}$ such pairs. Note that all of these pairs pass through the edge

$$\overbrace{0\ldots0}^{n \text{ bits}} \longrightarrow \overbrace{000\ldots0}^{n/2-1 \text{ bits}}\ 10\ \overbrace{000\ldots0}^{n/2-1 \text{ bits}}$$

Therefore in the worst case $T \gtrsim \sqrt{\frac{N}{n}}$, which is quite bad.

In fact in general no deterministic algorithm can do better than this by more than a constant factor, though the proof for this is outside the scope of this course.

4 Adding Randomization

One of the more natural approaches is to randomize the order of the bit fixing, but it turns out that this takes $2^{\Omega(n)}$ timesteps in the worst case. (To see this, shrink X from the above example to about $n/10$ bits, then while we are exponentially unlikely to fix all the first X bits before any others, there are also exponentially many such pairs and it turns out that $2^{\Omega(n)}$ will need to go through the same edge).

We will try another natural approach: reducing to the **average case** behavior.

- Send each message to a random destination (not nesc. a permutation)
- Use this to solve the worst case π.

4.1 Analysis

Let $L(e) = \#$ of paths that use edge e (we treat each edge as directed). To find $\mathbb{E}[L(e)]$ note that by symmetry for every e this value must be the same, so $\mathbb{E}[L(e)] =$ "average total load". Let D_v denote the length of the path starting from v.

$$\begin{aligned}\mathbb{E}[\sum_{e\in E} L(e)] &= \sum_{v\in V} \mathbb{E}[D_v] \\ &= \sum_{v\in V} \frac{n}{2} \\ &= \frac{Nn}{2}\end{aligned}$$

And so $E[L(e)] = \frac{Nn}{2Nn} = \frac{1}{2}$

Let $Y_x = 1_{x\to\pi(x)\text{uses } e}$. Then $L(e) = \sum_{x\in V} Y_x$ Note that all the Y's are independent as we choose the destinations independently, and so we can use a Chernoff bound to bound $L(e)$ (multiplicative, as our probabilities are small)!

$$\begin{aligned}\mathbb{P}[L(e) \geq (1+\varepsilon)\,\mathbb{E}[L(e)] &\leq \exp(-\frac{\min(\epsilon,\epsilon^2)}{3}\,\mathbb{E}[L(e)]) \\ &\leq \exp(-\frac{\min(\epsilon,\epsilon^2)}{6})\end{aligned}$$

In fact we will use a slightly nicer version, that is for $t > 2e\,\mathbb{E}[X], \mathbb{P}[X \geq t] \leq 2^{-t}$ (where X is the sum of independent variables in $[0,1]$)

So when $t > e$, we get $\mathbb{P}[L(e) \geq t] \leq 2^{-t}$, and so if we set $t = O(n)$, we can use a union bound over the nN edges to say w.h.p. No edge has more than $O(n)$ paths going through it.

Therefore, a path can be delayed at most $O(n)$ per edge, for $O(n^2)$ total time to solve the routing problem. Note that this is an exponential improvement over the deterministic version.

However, the $O(n^2)$ bound seems a little loose, as if we delay for a message it seems like we shouldn't delay for the same message again, as it is "ahead". in the next section we will improve this bound to $O(n)$, what we showed to be optimal earlier.

4.2 Improved bound

Let p_x denote the path taken from x to its random destination. Then, we define the set of "overlapping messages":

$$S_x = \{y \mid p_x \cap p_y \neq \emptyset\}$$

Lemma 1. *The time taken for the message of x to traverse the path p_x is at most $n + |S_x|$.*

The proof for this was omitted due to time constraints, but follows from analyzing the "lags" that any set of messages could suffer from traversing a common path, and is given in the book.

Now, we know that

$$\begin{aligned}\mathbb{E}[|S|] &\leq \mathbb{E}\left[|\{P_j : \exists e, e \in p_x \cap e \in p_y\}|\right] \\ &= \sum_{i=1}^{k} \mathbb{E}[L(e)] \\ &= \frac{k}{2} \leq \frac{n}{2}\end{aligned}$$

Using a particular form of the Chernoff bound, we know that

$$\mathbb{P}[|S_i| \geq t] \leq 2^{-t}, \forall t \geq 2e \cdot \frac{n}{2}$$

So, we have

$$\mathbb{P}[|S_i| \geq cn] \leq \frac{1}{N^c}$$

Now, using the above lemma, we have that

$$\mathbb{P}[\text{time taken by message i} \geq (c+1)n] \leq \frac{1}{N^c}$$

and by the union bound,

$$\mathbb{P}[\text{time taken by all messages} \geq (c+1)n] \leq \frac{1}{N^{c-1}}$$

which means that with high probability, we have that our time taken is $O(n)$.

4.3 Reduction to Average case

Now how do we use this to solve the worst case? For each send-receive pair $(x, \pi(x))$ we first send it to a random intermediate destination $f(x)$. Then we know with high probability it takes $O(n)$ steps to send x to $f(x)$, and in fact the analysis above is reversible, so it takes $O(n)$ steps to send $f(x)$ to $\pi(x)$, and therefore w.h.p we solved routing in $O(n)$ time!