Routing on a Hypercube

Myron Liu

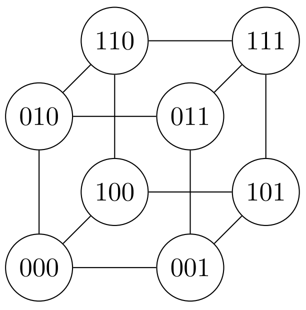
February 5-10, 2015

1 Problem Definition

First, we define a hypercube. A hypercube is a graph $G = \{V, E\}$ with the following properties:

- $|V| = N \equiv 2^n$ nodes such that each node $v \in V$ can be labeled as $v = \{0,1\}^n$.
- $v - v' \in E$ if and only if the bit representation of v and v' differ in only one index. Consequently, each $v \in V$ has degree n, and so $|E| = 2^n n/2$.

The familiar example of a 3d hypercube is



Now, consider the following problem:

- Initially, there is one packet at each node. Let the starting node of packet i be denoted by s_i.
- Let destinations $\{d_i\}$ be a permutation of $\{s_i\}$ such that packet i travels from s_i to d_i.
- How can we route the packets such that with high probability, all packets reach their destinations in $\mathcal{O}(n)$ time?

For the transit time, we introduce the notion of congestion/queue. At any point in time, only one packet can traverse a given edge. The queue for a particular edge is the number of packets that simultaneously wish to cross that edge. Then for any particular packet, its worse case transit time corresponds to the scenario in which it is placed at the end of every queue it enters. Letting the sequence of queues for packet i be denoted by $q_{i,1}, q_{i,2}, ..., q_{i,m}$, we can therefore bound the maximum transit time for packet i as $T_i \leq \sum_{j=1}^{m} q_{i,j}$. It suffices, then, to bound the queue time.

2 An Unsatisfactory Deterministic Algorithm

A naive approach would be to use the following deterministic *bit-fixing* algorithm:

- Let the k^{th} bit of $\llcorner s_i \lrcorner$ be denoted by $\llcorner s_i \lrcorner_k$ (similarly for d_i).
- Step through successive bits in $\llcorner s_i \lrcorner$ and $\llcorner d_i \lrcorner$ and modify $\llcorner s_i \lrcorner$ until it matches $\llcorner d_i \lrcorner$. Do this for each packet i in parallel.
- In other words, if $\llcorner s_i \lrcorner_k \neq \llcorner d_i \lrcorner_k$ then invert $\llcorner s_i \lrcorner_k$ by traversing the edge of the hypercube that corresponds to this bit-inversion.

This deterministic algorithm can suffer on some pathological instances of the problem. Consider the following examples:

- Suppose that $n = even$.
- Suppose the subset of packets such that $\llcorner s_i \lrcorner = \{0,1\}^{n/2} | \{0\}^{n/2}$ has corresponding destinations $\llcorner d_i \lrcorner = \{0\}^{n/2} | \{0,1\}^{n/2}$.
- By our deterministic algorithm, all $2^{n/2}$ of such s_i will reach the hub node $\{0\}^{n/2} | \{0\}^{n/2}$.

- Although, they may not arrive at the hub simultaneously (due to being held up in queues at previous steps), we consider the best case scenario in which they all reach the hub simultaneously. For all we care, we can assume that they all reach the hub instantaneously. We will see that even with this idealization, the maximum transit time can be very long.
- Each corresponding d_i will have its first 1 in any of $n/2$ bit positions.
- Therefore, the average queue length at this intermediate hub node will be $(2^{n/2} - 1)/(n/2)$
- One such packet's time spent in this hub's queue must exceed $(2^{n/2} - 1)/(n/2)$, and therefore, so must its total transit time.
- This long transit time constitutes a pathological case. We will see that introducing randomness makes this scenario highly unlikely.

3 Towards a Randomized Algorithm

As aforementioned, we can avoid pathological cases by introducing randomness into the bit-fixing algorithm. Here is one way to achieve this. Introduce a random set of intermediate destinations $\{r_i\}$, where, r_i need not be distinct from r_j. Having done so, we now execute the deterministic bit-fixing algorithm along the path $s_i \to r_i \to d_i$ for each packet i. We will call $s_i \to r_i$ phase 1 and $r_i \to d_i$ phase 2. For simplicity, we will assume that no packet begins phase 2 before all other packets finish phase 1. Clearly, this sort of waiting is not optimal, but we will show that even with such a constraint the algorithm is fast with high probability. Furthermore, we only need to perform the analysis on phase 1, the analysis applies identically to phase 2.

In analyzing this randomized algorithm, it is helpful to look at the problem from another viewpoint. Rather than selecting r_i randomly at the beginning, we can view r_i as being set dynamically. Since the bits are fixed sequentially, we view the random process as a coin flip at each subsequent bit.

Similarly, rather than focus on any one specific packet, we take the general approach of considering any potential path P in phase 1. Let the nodes along P be v_i with e_i being the the edge $v_i \to v_{i+1}$. The bit being fixed on e_i we shall denote by D_i (note that $D_i < D_j$ for $i < j$ by definition). Notice that there are only n bits to fix, so i runs from 0 to m with $m \leq n$. Graphically P is:

$$\begin{tikzcd}
\boxed{v_0} \arrow[r] & \boxed{v_1} \arrow[r] & \boxed{v_2} \arrow[r] & \boxed{v_3} \arrow[r, dash] & \cdots \arrow[r] & \boxed{v_i} \arrow[r, "z_i"] & \boxed{v_{i+1}} \arrow[r, dash] & \cdots \arrow[r] & \boxed{v_{m-1}} \arrow[r] & \boxed{v_m}
\end{tikzcd}$$

Here, we have labeled each edge e_i with z_i: the total number of packets crossing e_i in phase 1 (not necessarily simultaneously). We will call $T_P = \sum_{i=0}^{m} z_i$ the load on P. Our goal, is to show that $\Pr[t_P \geq 30n] \leq 2^{-cn}$ for some constant c. The reason for seeking this result is that for any packet, its transit time on p (call it t_P) is clearly upper bounded by load T_P. More specifically, if some packet k travels exactly along P, then the time it takes for packet k to complete phase 1 is no more than T_P.

Consider a packet k traversing the hypercube. Maintain an incrementer B_k that keeps track of which bit of packet k will be fixed next. We say that a packet k is *active* at node v_i in P if it arrives at v_i such that $B_k \leq D_i$. That is, upon reaching v_i, packet k will traverse e_i. It is easy to see that if packet k is ever active on P, then the moment it leaves P, it never becomes active again. After all, packet k would only leave P if it differs on some first bit D_j. By the bit-fixing algorithm, bit D_j of $v_{\ell > j}$ is identical to bit D_j of v_j, and so v_ℓ will be at odds with packet k as well *i.e.* packet k is exiled from P forever.

Let H be the number of distinct packets active along P. Let H_k be an indicator variable for packet k with:

$$H_k = \begin{cases} 1 & \text{if packet } k \text{ is active at some point on } P \\ 0 & \text{otherwise} \end{cases}$$

and $H = \sum_{k=1}^{N} H_k$. Note that the H_k are independent, since the trajectory of packet k depends entirely on r_k, and the set of r_k are chosen independently. What is $E[H]$, or more laxly, how can we bound $E[H]$ from above, such that we can apply the Chernoff bound? Let's consider the number of packets active at v_i. There are up to $2^{D_i - 1}$ packets that can be active at v_i. This can be seen using the following illustration:

$v_i =$	$a_1, ..., a_{D_i - 1}$	a_{D_i}	$a_{D_i + 1}, ..., a_n$
$v_{i+1} =$	$a_1, ..., a_{D_i - 1}$	b_{D_i}	$a_{D_i + 1}, ..., a_n$

where a_j are the values of the bits of v_i; b_j is the D_i^{th} bit value of v_{i+1}, which is the only place where v_{i+1} could potentially deviate from v_i. We see then that only the packets that have initial location s_k that are consistent with bits D_i onward (inclusive) of v_i can be active at v_i. Such packets have the chance of executing a random sequence of edge traversals such that it merges onto path P before bit D_i is to be fixed. This gives the $2^{D_i - 1}$ packets that could potentially be active at v_i, since we can choose the corresponding starting locations by fixing the first $D_i - 1$ bits to any sequence $\{0, 1\}^{D_i - 1}$.

Now, we need the probability that such a packet will actually be active at v_i. Recall our perspective of the random assignment r_k as a sequence of coin tosses. The probability that a sequence of tosses will be consistent with v_i is $2^{-(D_i-1)}$. Then the expected number of active packets at v_i is simply the product $2^{D_i-1}2^{-(D_i-1)} = 1$. Hence, the expected number of *distinct* packets on P is bounded by $E[H] \leq m \cdot 1 \leq n$. Applying Chernoff Bound, we have:

$$Pr[H \geq 6n \geq 6E[H]] \leq 2^{-6n}$$

Now look upon the load T_P. Is the probability of the load being large also small?

$$\begin{aligned}\Pr[T_P \geq 30n] =& \Pr[T_P \geq 30n | H < 6n] \Pr[H < 6n] + \\ & \Pr[T_P \geq 30n | H \geq 6n] \Pr[H \geq 6n] \\ \leq & \Pr[T_P \geq 30n | H < 6n] + \\ & \Pr[H \geq 6n] \\ \leq & \Pr[T_P \geq 30n | H < 6n] + \\ & 2^{-6n}\end{aligned}$$

How can we bound $\Pr[T_P \geq 30n | H < 6n]$? Recall our observation that the moment a once-active packet leaves P, it can nevermore become active. With this it is easy to see that the probability that a packet (that is at some point active on P) is active at v_i is no more than $1/2$; "no more" because the random coin flips for this packet must also be such that it is active at $v_{i-1}, v_{i-2}, ..., v_{\text{first active on } P}$. This is good, we want the probability of packets being inactive at v_i to be high, since this will decrease the load.

Our goal is to have packets leave P. Let's relax our assumptions to something slightly less than ideal, and assume that the probability of a packet being inactive at v_i is very close to $1/2$. The probability that the set of active packets puts a total load greater than $30n$ can be bounded in the following way. We view the process of packets crossing edges in P and leaving P (forever), as tosses of an unbiased coin $36n$ times. Each time we get tails, one of the active packets crosses an edge of P, thus increasing the load by one. Each time we get heads, one of the packets leaves P, and we rejoice. Of course, we are conditioning on $H < 6n$ (or more laxly $H \leq 6n$), so heads that appear beyond the $6n^{th}$ occurence don't make much sense in this regard. But it matters not. What we are after is a lower bound on the probability that fewer than $6n$ packets leave P in these $36n$ trials; if all $6n$ packets leave, then the total number of edge traversals will be fewer than $36n - 6n = 30n$. Since we took the pessimistic assumption that the probability of a packet being inactive at each step is negligibly larger than $1/2$, the probability of less than $6n$ heads is indeed less than the actual quantity we are after.

$$\begin{aligned}& \Pr[T_P \geq 30n | H < 6n] \\ \leq & \Pr[T_P \geq 30n | H \leq 6n] \\ \leq & \Pr[N_{head} \leq 6n]\end{aligned}$$

We can bound this last expression using Chernoff bound. Using $E[N_{head}] = 18n$ and $6n = (1-\delta = 1/3)E[N_{head}]$ we have:

$$\Pr[T_P \geq 30n | H < 6n] \leq \Pr[N_{head} \leq 6n] \leq e^{-4n} \leq 2^{-3n-1}$$

And so...

$$\begin{aligned}\Pr[T_P \geq 30n] &\leq \Pr[T_P \geq 30n | H < 6n] + 2^{-6n} \\ &\leq 2^{-3n-1} + 2^{-6n} \\ &\leq 2^{-3n}\end{aligned}$$

Finally, we need to consider all paths P. Clearly, there are 2^{2n} possible paths, since fixing the combination s_i, r_i determines the P. By union bound, the probability that at least one of the loads T_P exceeds $30n$ is no more than $2^{2n} \Pr[T_P \geq 30n] \leq 2^{-n}$ which is small. So it is highly unlikely that any packet will take more than $30n$ time-steps to complete phase 1.

This same analysis applies to phase 2. The only difference is that in phase 2, we don't have the condition that initially there is exactly one packet at each location. However, note that in nowhere in the analysis did we rely on this assumption, so it is also true that with high probability, all packets complete phase 2 within $30n$ time steps. Now, phase 1 and phase 2 are correlated in some way, but we can be pessimistic and say that the probability of failure in either phase 1 or phase 2 is no more than $2^{-n} + 2^{-n} = 2^{1-n}$ which is still small. So ultimately we have shown that by introducing randomness, the bit-fixing algorithm routes packets along a hypercube in linear time with high probability. ■