Competitive Analysis of Minimal Oblivious Routing Algorithms on Hypercubes

Tzuoo-Hawn Yeh and Chin-Laung Lei

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R. O. C.

Abstract

We study the performance of oblivious routing algorithms that follow minimal (shortest) paths, referred to as *minimal oblivious routing algorithms* in this paper, using *competitive analysis* on a d-dimensional, $N = 2^d$-node hypercube. We assume that packets are injected into the hypercube arbitrarily and continuously, without any (e.g., probabilistic) assumption on the arrival pattern of the packets. Minimal algorithms reduce the total load in the network in the first place and they preserve locality. First we show that the well known deterministic oblivious routing algorithm, namely, the *greedy routing algorithm*, has competitive ratio $\Omega(N^{1/2})$. Then we show a problem lower bound of $\Omega(N^{\log_2(5/4)}/\log^5 N)$. We also give a natural randomized minimal oblivious routing algorithm whose competitive ratio is close to the problem lower bound we provide.

1 Introduction

Routing algorithms are classified as either *oblivious* or *adaptive* strategies [9, 17]. A *deterministic* routing algorithm is oblivious if the path of a packet depends only on its source and target. For a *randomized* oblivious routing algorithm, the path of a packet depends on a probability distribution which is a function of the source and target of the packet. Oblivious routing algorithms are *distributed* (They process packets with local information. It is not necessary to know, for example, the current global network status.), *on-line* (They process packets without knowing future packets.), and *memoryless*, all of which are good properties for distributed systems. Another natural requirement of routing is to keep a routing algorithm from using paths that are longer than the distance of the source and target of a packet. This kind of routing algorithms are referred to as the *minimal routing algorithms* in this paper. Minimal routing algorithms works very well in some simple inputs that preserve locality. For example, consider the input that each node in the hypercube sends a packet to each of its neighbors. A minimal routing algorithm processes this input simply by sending every packet over the edge the packet covers. The network load is kept to a minimum. Non-minimal routing algorithms may route packets along long paths. This destroys the locality in this input and the resulting network load may be high. For these reasons, in this paper we concentrate on the analysis of *minimal oblivious routing algorithms*.

The *greedy routing algorithm* is a prime example of a deterministic minimal oblivious routing algorithm. Given a packet from source node s to target node t, the greedy routing algorithm corrects the bits that differ between s and t in increasing dimension order.

It is known [17] that there are permutations that the greedy routing algorithm needs $\Omega(\sqrt{N})$ steps to route. While in the average, the greedy algorithm performs well. In particular, if every node has $m \geq \log N$ packets initially that are destined for random targets, the greedy routing algorithm routes this random input in $O(m)$ steps with high probability.

In [27], Stamoulis and Tsitsiklis analyzed the greedy routing algorithm on a model that each node generates packets according to a Poisson process of rate λ. Each packet is to be sent to a target of distance k from its source with probability $p^k(1-p)^{d-k}$. They showed that the network is stable as long as $\rho < 1$, where $\rho = \lambda p$ is the load factor of the network. They also showed that the average delay T per packet satisfies $T \leq \frac{dp}{1-\rho}$.

In [28], Tsai and Shin showed some simulation results for routing on hypercubes. Various scheduling policies that decide which packet to forward first if multiple packets are contending for a link are compared. The performance of a routing policy called Progressive Adaptive that modifies the greedy routing algorithm are evaluated against a centralized path selection algorithm. The transient behaviors of these scheduling and routing policies are measured by, giving random input at time 0, how long it takes to complete the delivery of all packets. They found that the Largest Remaining Bandwidth First scheduling policy combined with the adaptive Progressive Adaptive routing algorithm to be effective in many conditions.

In the following, we introduce the input model, the cost measure, and the analysis technique used in this paper. We also contrast our models to those used in the related works.

Two fundamental models are frequently employed in constructing inputs for the analysis of routing algorithms. The first model is the *permutation input* model [9, 32, 10, 13, 23, 24, 16, 1]. In a permutation input, each node in the hypercube is the source and the target of exactly one packet. Another input model is the *probabilistic input* model [27], in which packets are injected into the system according to some probability distribution. As an alternative to the two models of inputs mentioned above, in this paper, we assume that the packets are injected into the network arbitrary and continuously without any (e.g., probabilistic) assumption on the arrival patterns of the inputs.

The cost measure of interest is the *congestion* of the network. A routing algorithm has congestion C if the maximum number of packets that travel through any edge is C. The congestion is a good performance indication in the packet switching model. In fact, the congestion is a lower bound on the number of time steps, called the *work-span*, to deliver all packets to their targets, under the constraint that at most one packet can cross an edge at a single time step. The congestion can also upper bound the work-span, as the progresses made in [18, 19, 20, 21, 25] show. Given a set of paths whose congestion is C and whose dilation, the length of the longest path, is D, the algorithm in [19], for example, schedules the paths (decides which packet to go first if multiple packets are contending for an edge) that the resulting work-span is $O(C+D)$. The congestion is also an important parameter in the circuit switching model, in which a set of packets can be transmitted simultaneously if their paths are edge-disjoint. So the congestion is a lower bound on how many steps to route all the packets.

Traditionally, worst case analysis [16, 32, 10, 13], average case analysis [17, 27], worst expected case analysis [29, 30], and simulation [28] are used to report the performance of routing algorithms. In this paper, however, we use *competitive analysis* [26] as a tool to evaluate the performance of minimal oblivious routing algorithms. Competitive analysis has been used successfully in many *on-line* problems [2, 8, 7, 11, 12, 15, 22, 31]. An on-line algorithm is said to be α competitive, or has competitive ratio α, if, for any request sequence, (cost of the on-line algorithm) $\leq \alpha \times$ (cost of the optimal off-line algorithm) + (some additive constant), in which the additive constant is independent of the request sequence. The competitive ratio reports the loss in performance due to on-line decision making against the optimal off-line algorithm, which is referred to as the *adversary* in this paper and in the literature [8]. The notion of competitiveness for on-line algorithms can be applied to comparing the performance of algorithms in a distributed setting, where a distributed algorithm that makes decisions locally is compared to an adversary that makes decisions with global views. Competitive analysis of problems in a distributed environment can be found in [3, 4, 5, 6]. For randomized algorithms, we assume that the adversaries are *oblivious* as was defined in [8].

The organization and the main results of this paper are summarized in the following. Let d be the number of hypercube dimensions and let $N = 2^d$ be the number of nodes. In Section 2, the notations and definitions useful in this paper are defined. In Section 3, we show that the competitive ratio of the greedy routing algorithm is $O(N^{1/2})$. We modify the transpose permutation [17] and show that it is bad for the greedy routing algorithm while at the same time some off-line algorithm can route this partial permutation with small congestion. In Section 4, we provide a lower bound $\Omega(N^{\log_2(5/4)}/\log^5 N)$ of competitive ratio for randomized minimal oblivious algorithms. Note that $\log_2(5/4)$ is about 0.3219. In Section 5, we show that a randomized variant of the greedy routing algorithm is close to our supplied problem lower bound. Given a packet, this randomized routing algorithm complements the different bits between the source and target of the packet in random order. The competitive ratio of the algorithm is within a factor of $O(\log^3 N)$ from the lower bound. We have to emphasize that the upper and lower bounds we provide are strong since the adversaries also use minimal paths. We are not comparing minimal oblivious routing algorithms against adversaries that are allowed to use non-minimal paths, in which case such adversaries may be too strong to provide comprehensive competitive ratios. The orders of the lower and upper bounds are derived in Section 6.

2 Preliminaries

A d-dimensional *hypercube* is a directed graph with node set $\mathbb{V}$ and edge set $\mathbb{E}$. There are 2^d nodes in $\mathbb{V}$ and they are numbered from 0 to $2^d - 1$. We use d-bit binary encodings of the numbers to denote nodes. Let $b_{d-1}b_{d-2}\cdots b_0$ be a node, we call b_i, $0 \le i \le d-1$, the ith bit (or the ith dimension) of the node. If two nodes u and v differ in precisely one dimension, then there are two edges connecting them, one from u to v and another from v to u, denoted as (u, v) and (v, u), respectively. If u and v differ in dimension x, (u, v) is also denoted as $u \xrightarrow{(x)}$. Note that $|\mathbb{E}| = |\mathbb{V}| \cdot d$. When $d = 0$, a hypercube is a single node. When $d = 1$, a hypercube consists of two nodes and two edges connecting them. No routing is necessary on these cases, so in this paper we assume that $d \ge 2$.

A *path* τ of length l from node s to node t is a sequence of adjacent edges $e_1 = (v_0, v_1) = (s, v_1)$, $e_2 = (v_1, v_2)$, $\ldots$, $e_l = (v_{l-1}, v_l) = (v_{l-1}, t)$ that connecting s and t. We call e_i, $1 \le i \le l$, the ith edge on the path, and we call v_i, $0 \le i \le l$, the ith node on the path. The path is a *minimal path* (or *shortest path*) if l equals the number of bits that differ between s and t. Assume that the end-point nodes of e_i differ in bit x_i, we write the path as the sequence $[\,x_1, x_2, \ldots, x_l\,]$ when s and t are clear from the context. A *packet* δ from node s to node t is denoted as $\langle s, t\rangle$. We call s and t the source and target nodes, respectively. A path for δ is a path connecting s and t. The length of δ is the length of a shortest path from s to t. We say two packets $\delta_1 = \langle s_1, t_1\rangle$ and $\delta_2 = \langle s_2, t_2\rangle$ are *equivalent* if $s_1 = s_2$ and $t_1 = t_2$; otherwise we say δ_1 and δ_2 are non-equivalent. Let $SE(\delta, i)$ be the set of edges $\{e \,|\, e$ is the ith edge on some shortest path for δ.$\}$ An *input* σ of size m is a multi-set of m packets $\{\delta_1, \delta_2, \ldots, \delta_m\}$. Let $\sigma^k = \{\delta_1\delta_1\cdots\delta_1\delta_2\delta_2\cdots\delta_2\cdots\delta_m\delta_m\cdots\delta_m\}$, the input where each packet in σ is repeated k times.

Given an input σ and an algorithm A that assigns paths to the packets in σ, we define the *load* on an edge $e \in \mathbb{E}$, denoted as $\text{load}_{\mathsf{A}}(\sigma, e)$, to be the number of packets whose paths cross e. (Without loss of generality, we assume that the paths are simple paths. That is, the path for a packet crosses an edge at most once.) We define the *cost* (or *congestion*) of A on σ, denoted as $\text{cost}_{\mathsf{A}}(\sigma)$, to be $\max_{e\in\mathbb{E}} \text{load}_{\mathsf{A}}(\sigma, e)$, the maximum load among the edges.

Consider an input $\sigma = \{\delta_1\delta_2\cdots\delta_m\}$. A *deterministic oblivious* routing algorithm is one that the path of a packet $\delta = \langle s, t\rangle \in \sigma$ depends only on s and t. A *randomized oblivious* routing algorithm is one that the path of a packet $\delta = \langle s, t\rangle$ is chosen randomly from a set of paths whose random distribution depends on s and t. The *oblivious routing problem* is that of designing oblivious routing algorithms whose competitive ratios are minimized, where we say an algorithm A is c competitive (or we say A has competitive ratio c) if there is a constant β such that, for any input σ,

$$\mathrm{cost}_{\mathsf{A}}(\sigma) \le c \cdot \mathrm{cost}_{\mathsf{ADV}}(\sigma) + \beta. \quad (1)$$

ADV stands for the adversary, an off-line optimal routing algorithm. When A is a randomized algorithm, $\mathrm{cost}_{\mathsf{A}}(\sigma)$ in the inequality should be read as $\exp(\mathrm{cost}_{\mathsf{A}}(\sigma))$, the expectation of A's cost on σ.

3 The Greedy Routing Algorithm

Given a packet $\langle s, t\rangle$, the greedy routing algorithm Greedy complements the different bits between s and t in some fixed order, which we assume in this paper is in increasing dimension order without loss of generality.

Theorem 1. *The competitive ratio of* Greedy *is* $\Omega(\sqrt{N})$.

Proof. Let m be an integer. Consider the following input σ. For each node s whose bit 0 is 1, bits 1 to $\lfloor d/2\rfloor - 1$ are arbitrary, and bits $\lfloor d/2\rfloor$ to $d-1$ are zeros, send m packets to the node t whose bits are s rotated left $\lfloor d/2\rfloor$ bits. All packets cross the edge $e_0 = 0^d \xrightarrow{(\lfloor d/2\rfloor)}$. So $\mathrm{cost}_{\mathsf{Greedy}}(\sigma) \ge \mathrm{load}_{\mathsf{Greedy}}(\sigma, e_0) = m \cdot 2^{\lfloor d/2\rfloor - 1}$.

The following strategy processes σ via shortest paths with cost m: For any packet $\delta = \langle s, t\rangle \in \sigma$, send it to t by complementing the bits that differ between s and t in *decreasing* dimension order. Consider two packets $\delta_1 = \langle s_1, t_1\rangle$ and $\delta_2 = \langle s_2, t_2\rangle$ that $s_1 \ne s_2$ and therefore $t_1 \ne t_2$. Let v_1 and v_2, respectively, be intermediate nodes on the paths for δ_1 and δ_2, respectively. If v_i is a node on the first half of the path for δ_i (while we are traversing bits $d-1$ to $\lfloor d/2\rfloor$), bits 0 to $\lfloor d/2\rfloor - 1$ of v_i and s_i are the same. Similarly, if v_i is a node on the second half of the path for δ_i (while we we are traversing bits $\lfloor d/2\rfloor - 1$ to 0), bits $\lfloor d/2\rfloor$ to $d-1$ of v_i and t_i are equal. Note that bits 0 to $\lfloor d/2\rfloor - 1$ of s_1 and s_2 are different and bits $\lfloor d/2\rfloor$ to $d-1$ of t_1 and t_2 are different. So for any v_1 and v_2 mentioned above, $v_1 \ne v_2$, and therefore the cost of the strategy on σ is m and $\mathrm{cost}_{\mathsf{ADV}}(\sigma) \le m$. As a result, the competitive ratio of Greedy is at least $\frac{m \cdot 2^{\lfloor d/2\rfloor - 1} - \beta}{m}$. When m is large, β is negligible, and the competitive ratio of Greedy is at least $\frac{m\cdot 2^{\lfloor d/2\rfloor -1}}{m} = 2^{\lfloor d/2\rfloor - 1} \ge 2^{d/2-2} = \Omega(2^d) = \Omega(\sqrt{N})$. □

4 A Problem Lower Bound

For $0 \le i \le \lfloor (d-3)/2\rfloor$, we call bits $2i$ and $2i+1$ of a node the ith *buddy* of that node. Given d, the dimension of the hypercube, and an integer k where $0 \le k \le \lfloor (d-1)/4\rfloor$, let $ST_{k/d}$ be the set of packets that a packet $\delta = \langle s, t\rangle \in ST_{k/d}$ if the following three conditions are met.

ST Condition 1. The distance from s to t is $2k+1$ and the edge $(0^d, 10^{d-1}) \in SE(\delta, k+1)$.

ST Condition 2. If d is even, bits $d-2$ of s and t are both 0.

ST Condition 3. For $0 \le i \le \lfloor (d-3)/2\rfloor$, if the ith buddy of s is 01, then the ith buddy of t is 10; if the ith buddy of s is 10, then the ith buddy of t is 01; if the ith buddy of s is 11, then the ith buddy of t is 00; if the ith buddy of s is 00, then the ith buddy of t is either 00 or 11.

By ST Condition 1, a packet $\langle s, t\rangle \in ST_{k/d}$ has the following properties: (1) Bits $d-1$ of s and t are 0 and 1, respectively; (2) s has k one bits and t has $k+1$ one bits; (3) if bit i of s is 1, then bit i of t is 0; and (4) if bit i of t is 1, then bit i of s is 0.

For any integer k, $0 \le k \le \lfloor (d-1)/4\rfloor$, and any edge $e = (u, v) \in \mathbb{E}$, let $\sigma_{e,k/d}$ be the set of packets defined recursively as follows:

Canonical Condition 0. If $k = 0$, $\sigma_{e,k/d} = \{\langle u, v\rangle\}$.

Canonical Condition 1. If $u = 0^d$ and $v = 10^{d-1}$, then $\sigma_{e,k/d} = \{\langle s, t\rangle \mid \langle s, t\rangle \in ST_{k/d}$ and $\langle s', t'\rangle \in \sigma_{e',k'/d'}\}$, where s', t', e', … , are computed by the following procedure. (Procedure HALF)

- Input: $(d, k, e = (u, v), \delta = (s, t))$. Output: $(d', k', e' = (u', v'), \delta' = (s', t'))$.

- Assume that buddies x_0, x_1, ... , x_{j-1} of s are 00 or 11 buddies, $0 \le x_0 < x_1 < \cdots < x_{j-1} \le \lfloor (d-3)/2 \rfloor$. By ST Condition 3, buddies x_0, x_1, ... , x_{j-1} of t are 00 or 11 buddies, too, and these are the only 00 or 11 buddies of t.
- $s' = 0q_{j-1}q_{j-2}\cdots q_0$, where bit $q_i = 0$ or 1, respectively, if buddy x_i of s is 00 or 11, respectively.
- $t' = 1r_{j-1}r_{j-2}\cdots r_0$, where bit $r_i = 0$ or 1, respectively, if buddy x_i of t is 00 or 11, respectively.
- $d' = 1 + j$; k' = the number of one bits in s'; and $e' = (u', v') = (0^{d'}, 10^{d'-1})$.

Canonical Condition 2. If u and v differ in bit $d-1$, then $\sigma_{e,k/d} = \{\langle s \oplus u, t \oplus v\rangle \mid \langle s, t\rangle \in \sigma_{(u\oplus u, v \oplus u),k/d}\} = \{\langle s \oplus u, t \oplus v\rangle \mid \langle s, t\rangle \in \sigma_{(0^d,10^{d-1}),k/d}\}$. The symbol $\oplus$ denotes the bitwise exclusive or operator.

Canonical Condition 3. If u and v differ in bit k, $0 \le k < d-1$, then $\sigma_{e,k/d} = \{\langle \mathrm{rol}(s, k+1), \mathrm{rol}(t, k+1)\rangle \mid \langle s, t\rangle \in \sigma_{(\mathrm{ror}(u,k+1),\mathrm{ror}(v,k+1)),k/d}\}$, where $\mathrm{rol}(n, k)$ is n rotated left k bits and $\mathrm{ror}(n, k)$ is n rotated right k bits.

Any two distinct packets in the set $\sigma_{e,k/d}$ are non-equivalent. Furthermore,

Lemma 1. *If $\delta_1 = (s_1, t_1)$ and $\delta_2 = (s_2, t_2)$ are two distinct packets in $\sigma_{e,k/d}$, then $s_1 \ne s_2$ and $t_1 \ne t_2$.*

Proof. By Canonical Conditions 2 and 3, one can see that the construction of $\sigma_{e,k/d}$ for a general e is by some fixed complementation and/or rotation that depends on e of the bits of the endpoints of each of the packets in $\sigma_{(0^d,10^{d-1}),k/d}$. So in the proof we focus on the edge $e = (u, v) = (0^d, 10^{d-1})$ without loss of generality.

We prove the lemma by induction on d. When $2 \le d \le 4$, k must be 0. By Canonical Condition 0, $\sigma_{e,k/d}$ is $\{\langle u, v\rangle\}$. When $5 \le d \le 8$, k can be 0 or 1. If $k = 0$, then $\sigma_{e,k/d} = \{\langle u, v\rangle\}$. See Table 1 for the packets in $\sigma_{e,1/d}$, for $5 \le d \le 8$. It's clear that the lemma holds in these simple cases.

$\sigma_{e,1/5}$	$\langle 00001, 10010\rangle$, $\langle 00010, 10001\rangle$, $\langle 00100, 11000\rangle$, $\langle 01000, 10100\rangle$.
$\sigma_{e,1/6}$	$\langle 000001, 100010\rangle$, $\langle 000010, 100001\rangle$, $\langle 000100, 101000\rangle$, $\langle 001000, 100100\rangle$.
$\sigma_{e,1/7}$	$\langle 0000001, 1000010\rangle$, $\langle 0000010, 1000001\rangle$, $\langle 0000100, 1001000\rangle$, $\langle 0001000, 1000100\rangle$, $\langle 0010000, 1100000\rangle$, $\langle 0100000, 1010000\rangle$.
$\sigma_{e,1/8}$	$\langle 00000001, 10000010\rangle$, $\langle 00000010, 10000001\rangle$, $\langle 00000100, 10001000\rangle$, $\langle 00001000, 10000100\rangle$, $\langle 00010000, 10100000\rangle$, $\langle 00100000, 10010000\rangle$.

Table 1: The packets contained in $\sigma_{e,1/d}$, for $5 \le d \le 8$.

Assume that the lemma holds when $d < d_0$, in the following we prove that the lemma holds when $d = d_0$.

Assume that the outputs of Procedure HALF on the inputs $(d, k, e, \langle s_1, t_1\rangle)$ and $(d, k, e, \langle s_2, t_2\rangle)$ are $(d'_1, k'_1, e'_1, \langle s'_1, t'_1\rangle)$ and $(d'_2, k'_2, e'_2, \langle s'_2, t'_2\rangle)$, respectively.

Consider the case where $s_1 = s_2$. By HALF, $s'_1 = s'_2$, $d'_1 = d'_2$, $k'_1 = k'_2$, and $e'_1 = e'_2$. Packets $\langle s'_1, t'_1\rangle$ and $\langle s'_2, t'_2\rangle$ belong to the same input $\sigma_{e'_1,k'_1/d'_1}$. By the induction hypothesis, $t'_1 = t'_2$. By ST Condition 3, the 01 and 10 buddies of t_1 and t_2 match. By HALF, the 00 and 11 buddies of t_1 and t_2 match. So $t_1 = t_2$ and δ_1 and δ_2 are equivalent, a contradiction.

Consider the case where $t_1 = t_2$. By HALF, $t'_1 = t'_2$, $d'_1 = d'_2$, $k'_1 = k'_2$, and $e'_1 = e'_2$. Packets $\langle s'_1, t'_1\rangle$ and $\langle s'_2, t'_2\rangle$ belong to the same input $\sigma_{e'_1,k'_1/d'_1}$. By the induction hypothesis, $s'_1 = s'_2$. By ST Condition 3, the 01 10 buddies of s_1 and s_2 match. By HALF, the 00 and 11 buddies of s_1 and s_2 match. So $s_1 = s_2$ and δ_1 and δ_2 are equivalent, a contradiction. □

The following lemma gives the size of $\sigma_{e,k/d}$.

Lemma 2. *Given d and k, $0 \le k \le \lfloor (d-1)/4 \rfloor$, and an edge $e \in \mathbb{E}$, $\frac{|\sigma_{e,k/d}|}{\binom{d-1}{k}} \ge (3/4)^{\log_2(d-1)}$.*

Proof. By the definition of $\sigma_{e,k/d}$, $|\sigma_{e_1,k/d}| = |\sigma_{e_2,k/d}|$ for any two edges e_1 and e_2. So without loss of generality we focus on the edge $e = (u, v) = (0^d, 10^{d-1})$.

We prove the lemma by induction on d. When $2 \leq d \leq 4$, k must be 0. In this case, $|\sigma_{e,k/d}| = 1$, $\binom{d-1}{k} = 1$, and $(3/4)^{\log_2(d-1)} \leq 1$, so the lemma holds. Assume that the lemma holds when $d < d_0$. In the following we consider the case where $d = d_0$.

Define S_1 to be the set of nodes whose distance from u is k and whose bit $d-1$ is 0. S_1 is exactly the set of source nodes of the packets mentioned in ST Condition 1. If d is even, define S_2 to be the set of nodes in S_1 whose bits $d-2$ is 0; if d is odd, define $S_2 = S_1$. S_2 is exactly the set of source nodes of the packets that satisfy ST Conditions 1 and 2. Define S_3 to be the set of nodes $\{s \,|\, \langle s,t\rangle \in \sigma_{e,k/d}$ for some $t.\}$ S_3 is exactly the set of source nodes of the packets that satisfy ST Conditions 1 and 2 and Canonical Condition 1. By Lemma 1, $|S_3| = |\sigma_{e,k/d}|$. If d is odd, $S_2 = S_1$; otherwise, $|S_2| = \binom{d-2}{k}$. So, $\frac{|S_2|}{|S_1|} \geq \frac{\binom{d-2}{k}}{\binom{d-1}{k}} \geq \frac{d-1-k}{d-1} \geq 3/4$.

We partition S_2 into groups that two nodes s_1 and s_2 in S_2 are in the same group if (1) the 01 and 10 buddies of s_1 and s_2 match, and (2) the numbers of 11 buddies of s_1 and s_2 are equal. Note that the numbers of 00 buddies of two nodes in a group are also equal. We also partition S_3 into groups in the same way. Now consider a group F of S_2 and let $G = F \cap S_3$. Note that G is a group of S_3 and is exactly the set of source nodes of packets in $\sigma_{e',k'/d'}$, where k' is the number of 11 buddies of any node in G, $d'-1-k'$ is the number of 00 buddies, and $e' = (0^{d'}, 10^{d'-1})$. Therefore, $|G| = |\sigma_{e',k'/d'}|$. By the way we partition S_2, $|F| = \binom{d'-1}{k'}$. Therefore, $\frac{|G|}{|F|} = \frac{|\sigma_{e',k'/d'}|}{\binom{d'-1}{k'}} \geq (3/4)^{\log_2(d'-1)} \geq (3/4)^{\log_2 \frac{d-1}{2}} \geq (3/4)^{\log_2(d-1)-1}$, by the induction hypothesis and the fact that $(d'-1) \leq \frac{d-1}{2}$. This ratio holds for any group of S_2 and S_3 thus partitioned, therefore, $\frac{|S_3|}{|S_2|} \geq (3/4)^{\log_2(d-1)-1}$. As a result, $\frac{|\sigma_{e,k/d}|}{\binom{d-1}{k}} = \frac{|S_3|}{|S_1|} = \frac{|S_3|}{|S_2|} \cdot \frac{|S_2|}{|S_1|} \geq (3/4)^{\log_2(d-1)-1} \cdot (3/4) = (3/4)^{\log_2(d-1)}$. □

We say $h = h_{d-1}, h_{d-2}, \ldots, h_0$ is a *reorder* if $h_{d-1}, h_{d-2}, \ldots, h_0$ is a permutation of $d-1, d-2, \ldots, 0$. We call h_i, $0 \leq i \leq d-1$, the ith digit of reorder h. Define v/h, where $v = b_{d-1}b_{d-2}\cdots b_0$ is a node in $\mathbb{V}$, to be the node $b_{h_{d-1}}b_{h_{d-2}}\cdots b_{h_0}$. Define δ/h, where $\delta = \langle s,t\rangle$ is a packet, to be the packet $\langle s/h, t/h\rangle$. Define σ/h, where $\sigma = \{\delta_1\delta_2\cdots\delta_m\}$ is an input, to be the input $\{\delta_1/h, \delta_2/h, \ldots, \delta_m/h\}$.

Consider an edge $e = u \xrightarrow{(x)}$. Let H_e be the set of all reorders whose digit x is x. We say that an input σ is a *bad input* with respect to e if there are reorders $h_k \in H_e$, $0 \leq k \leq \lfloor (d-1)/4 \rfloor$, that $\sigma = \bigcup_{k=0}^{\lfloor (d-1)/4 \rfloor} \sigma_{e,k/d}/h_k$.

Lemma 3. *For any minimal oblivious routing algorithm* A*, there is an edge* $\hat{e}$ *and a bad input* $\hat{\sigma}$ *with respect to* $\hat{e}$ *that* $\exp(\mathrm{load}_{\mathsf{A}}(\hat{\sigma}, \hat{e})) \geq (3/4)^{\log_2(d-1)} \cdot \sum_{k=0}^{\lfloor (d-1)/4 \rfloor} \frac{(d-1)!\cdot k!}{(d-1-k)!\cdot(2k+1)!}$.

Proof. We use an averaging argument on all edges and all bad inputs. Let $\sigma(e)$ be the bad input with respect to e that has the worst expected load on e.

$$\begin{aligned}
\exp(\mathrm{load}_{\mathsf{A}}(\hat{\sigma}, \hat{e})) &\geq \frac{1}{|\mathbb{E}|}\sum_{e\in\mathbb{E}} \exp(\mathrm{load}_{\mathsf{A}}(\sigma(e), e)) \\
&\geq \frac{1}{|\mathbb{E}|}\sum_{e\in\mathbb{E}} \sum_{k=0}^{\lfloor (d-1)/4 \rfloor} \Bigg(\frac{1}{|H_e|} \\
&\qquad \cdot \sum_{h\in H_e} \exp(\mathrm{load}_{\mathsf{A}}(\sigma_{e,k/d}/h, e))\Bigg) \\
&= \frac{1}{|\mathbb{E}|}\sum_{e\in\mathbb{E}} \sum_{k=0}^{\lfloor (d-1)/4 \rfloor} \Bigg(\frac{1}{|H_e|} \\
&\qquad \cdot \sum_{h\in H_e} \sum_{\delta\in\sigma_{e,k/d}} \exp(\mathrm{load}_{\mathsf{A}}(\{\delta/h\}, e))\Bigg).
\end{aligned}$$

Let $I_{e,k}$ be the set of input sets $\{\{\delta\} \,|\, \delta$ is a packet of length $2k+1$ and $e \in SE(\delta, k).\}$ The inputs mentioned in

$$\sum_{h\in H_e} \sum_{\delta\in\sigma_{e,k/d}} \exp(\mathrm{load}_{\mathsf{A}}(\{\delta/h\}, e))$$

are in $I_{e,k}$ and there are $|\sigma_{e,k/d}| \cdot |H_e|$ of them. Because the summation is over all reorders in H_e, the inputs in $I_{e,k}$ are symmetrical in how many times they appear in the summation, and, therefore,

$$\sum_{h\in H_e}\sum_{\delta\in\sigma_{e,k/d}} \exp(\mathrm{load}_{\mathsf{A}}(\{\delta/h\},e)) = \frac{|\sigma_{e,k/d}|\cdot|H_e|}{|I_{e,k}|}\sum_{\sigma\in I_{e,k}}\exp(\mathrm{load}_{\mathsf{A}}(\sigma,e)).$$

Then,

$$\exp(\mathrm{load}_{\mathsf{A}}(\sigma_{\hat{e}},\hat{e})) \ge \frac{1}{|\mathbb{E}|}\sum_{k=0}^{\lfloor(d-1)/4\rfloor}\sum_{e\in\mathbb{E}}\Bigg(\frac{|\sigma_{e,k/d}|}{|I_{e,k}|} \cdot \sum_{\sigma\in I_{e,k}}\exp(\mathrm{load}_{\mathsf{A}}(\sigma,e))\Bigg),$$

by Lemma 2,

$$\ge \frac{1}{|\mathbb{E}|}\sum_{k=0}^{\lfloor(d-1)/4\rfloor}\sum_{e\in\mathbb{E}}\Bigg(\frac{(3/4)^{\log(d-1)}\cdot\binom{d-1}{k}}{\binom{d-1}{k}\cdot\binom{d-1-k}{k}} \cdot\sum_{\sigma\in I_{e,k}}\exp(\mathrm{load}_{\mathsf{A}}(\sigma,e))\Bigg)$$
$$= \sum_{k=0}^{\lfloor(d-1)/4\rfloor}\Bigg(\frac{(3/4)^{\log(d-1)}}{|\mathbb{E}|\cdot\binom{d-1-k}{k}} \cdot\sum_{e\in\mathbb{E}}\Bigg(\sum_{\sigma\in I_{e,k}}\exp(\mathrm{load}_{\mathsf{A}}(\sigma,e))\Bigg)\Bigg).$$

Let J_k be the set of input sets $\{\{\delta\}\,|\,\delta$ is a packet of length $2k+1\}$, $\sum_{e\in\mathbb{E}}\left(\sum_{\sigma\in I_{e,k}}\exp(\mathrm{load}_{\mathsf{A}}(\sigma,e))\right) = \sum_{\sigma\in J_k}\left(\sum_{e\in SE(\sigma,k+1)}\exp(\mathrm{load}_{\mathsf{A}}(\sigma,e))\right) = \sum_{\sigma\in J_k}1 = |J_k|$. Therefore,

$$\exp(\mathrm{load}_{\mathsf{A}}(\sigma_{\hat{e}},\hat{e})) \ge \sum_{k=0}^{\lfloor(d-1)/4\rfloor}\left(\frac{(3/4)^{\log(d-1)}}{|\mathbb{E}|\cdot\binom{d-1-k}{k}}\cdot|J_k|\right)$$
$$= \sum_{k=0}^{\lfloor(d-1)/4\rfloor}\left(\frac{(3/4)^{\log(d-1)}}{|\mathbb{E}|\cdot\binom{d-1-k}{k}}\cdot|\mathbb{V}|\cdot\binom{d}{2k+1}\right)$$
$$= \sum_{k=0}^{\lfloor(d-1)/4\rfloor}\frac{(d-1)!\cdot k!\cdot(3/4)^{\log(d-1)}}{(d-1-k)!\cdot(2k+1)!}. \qquad \square$$

Lemma 4. *For any edge e and any bad input σ with respect to e,* $\mathrm{cost}_{\mathsf{ADV}}(\sigma)\le d$.

Proof. Recall that $\sigma = \bigcup_{k=0}^{\lfloor(d-1)/4\rfloor}\sigma_{e,k/d}/h_k$, where h_k, $0\le k\le\lfloor(d-1)/4\rfloor$, is an arbitrary reorder in H_e. We will show that $\mathrm{cost}_{\mathsf{ADV}}(\sigma_{e,k/d}/h_k)=1$. So $\mathrm{cost}_{\mathsf{ADV}}(\sigma)\le\lfloor(d-1)/4\rfloor+1$, and the lemma follows.

Note that any reorder h_k can not fool the adversary, that is, if there is a strategy for $\sigma_{e,k/d}$ that $\mathrm{cost}_{\mathsf{ADV}}(\sigma_{e,k/d})=1$, then it is easy to convert the strategy to another for $\sigma_{e,k/d}/h_k$ that $\mathrm{cost}_{\mathsf{ADV}}(\sigma_{e,k/d}/h_k)=1$. Therefore, without loss and generality, we focus on the input $\sigma_{e,k/d}$. Also note that $\sigma_{e,k/d}$ is obtained by complementing (as in Canonical Condition 2) and rotating (as in Canonical Condition 3) of $\sigma_{(0^d,10^{d-1}),k/d}$. So without loss of generality, we focus on the edge $e=(0^d,10^{d-1})=0^d\xrightarrow{(d-1)}$ and the input $\sigma_{e,k/d}$, for any k, $0\le k\le\lfloor(d-1)/4\rfloor$.

In the following we show how to construct a path for each packet in $\sigma_{e,k/d}$ recursively such that (1) the paths are edge-disjoint; and (2) dimension $d-1$ is crossed last in the paths.

Consider a packet $\delta=(s,t)\in\sigma_{e,k/d}$. Let the output of Procedure HALF on (d,k,e,δ) be $(d',k',e',\delta'=(s',t'))$. Assume that bit i, $0\le i\le d'-2$, of s' corresponds to buddy $g(i)$ of s. Remember that buddies $g(i)$'s of s are 00 or 11 buddies. Assume that the path constructed for δ' is $[\,p_1,p_2,\ldots,p_{2k'},d'-1\,]$ and that buddies $f(1)$, $f(2)$, …, $f(l)$ of s are 01 or 10 buddies, $f(1)<f(2)<f(3)<\cdots<f(l)$. The path we assign to δ is $[\,2f(1),2f(1)+1,2f(2),2f(2)+1,\ldots,2f(l),2f(l)+1,2g(p_1),2g(p_1)+1,2g(p_2),2g(p_2)+1,\ldots,2g(p_{2k'}),2g(p_{2k'})+1,d-1\,]$. For convenience, the path is divided into three parts. The first part path is

$$[\,2f(1),2f(1)+1,2f(2),2f(2)+1,\ldots,2f(l),2f(l)+1\,]$$

the second part path is

$$[\,2g(p_1),2g(p_1)+1,2g(p_2),2g(p_2)+1,\ldots,2g(p_{2k'}),2g(p_{2k'})+1\,],$$

00000001 →	00000000 →	00000010 →	10000010
00000010 →	00000011 →	00000001 →	10000001
00000100 →	00000000 →	00001000 →	10001000
00001000 →	00001100 →	00000100 →	10000100
00010000 →	00000000 →	00100000 →	10100000
00100000 →	00110000 →	00010000 →	10010000

Table 2: Paths constructed for packets in $\sigma_{e,1/8}$.

and the third part path is $[\,d-1\,]$. In the first part path, 01 and 10 buddies of s change to 10 and 01 buddies, respectively. In the second part path, 11 buddies of s change to 00 buddies, and some 00 buddies of s change to 11 buddies, based on the path for δ'. In the third part path, dimension $d-1$ is crossed.

We prove by induction on d that the paths thus constructed for packets in $\sigma_{e,k/d}$ satisfy the two properties that (1) the paths are edge-disjoint and (2) dimension $d-1$ is crossed last in every path. When $2 \le d \le 4$, k must be 0. $\sigma_{e,k/d} = \{\langle u, v\rangle\}$. When $5 \le d \le 8$, k can be 0 or 1. If $k = 0$, then $\sigma_{e,k/d} = \{\langle u, v\rangle\}$. When $k = 1$, the packets in $\sigma_{e,k/d}$ is listed in Table 1. It is easy to see that there are paths that satisfy the two properties. Take $d = 8$ and $k = 1$ as an example, the paths that fulfill the properties are listed in Table 2. The induction basis is thus established.

Assume that for $d < d_0$ and any k, the constructed paths satisfy the two properties. Now consider the input $\sigma_{e,k/d}$ where $d = d_0$. By the way we construct the paths for packets in $\sigma_{e,k/d}$, the second property that dimension $d-1$ is crossed last is always true. We prove that the paths are edge-disjoint by proving the following claims.

Claim 1. *The first part paths of the packets in $\sigma_{e,k/d}$ are edge-disjoint.*

Consider a packet $\delta = (s,t) \in \sigma_{e,k/d}$ and assume that its first part path is $[\,f_1, f_2, \dots, f_{2l}\,]$, $0 \le f_1 < f_2 < \cdots < f_{2l} \le d-1$. Define $n_x(\delta)$, $0 \le x \le d$, to be $s + [\,f_1, f_2, \dots, f_i\,]$, where i is the largest number such that $f_i < x$.

If the 0th buddy of δ is 00 or 11, then $n_2(\delta) = n_0(\delta) = s$. If the 0th buddy of δ is 01 (or 10, respectively), then the packet $\tilde{\delta} = (\tilde{s}, \tilde{t}) \in \sigma_{e,k/d}$, where $\tilde{s}$ is s except that the 0th buddy of $\tilde{s}$ is 10 (01, respectively), is the *complement* packet of δ in that $n_2(\delta) = n_0(\tilde{\delta}) = \tilde{s}$ and $n_2(\tilde{\delta}) = n_0(\delta) = s$. As a result $\bigcup_{\delta\in\sigma_{e,k/d}} n_0(\delta) = \bigcup_{\delta\in\sigma_{e,k/d}} n_2(\delta)$, and there is a one-to-one correspondence between the nodes in $\bigcup_{\delta\in\sigma_{e,k/d}} n_0(\delta)$ and in $\bigcup_{\delta\in\sigma_{e,k/d}} n_2(\delta)$.

Now consider two distinct packets $\delta_1 = (s_1, t_1)$ and $\delta_2 = (s_2, t_2)$ in $\sigma_{e,k/d}$. By Lemma 1, $s_1 \ne s_2$, that is, $n_0(\delta_1) \ne n_0(\delta_2)$. By the discussion above, $n_2(\delta_1) \ne n_2(\delta_2)$. We can extend the discussion above and say that $n_i(\delta_1) \ne n_i(\delta_2)$, for any even i.

Assume that the first part paths of δ_1 and δ_2 share an common edge $(\bar{u}, \bar{v})$, where $\bar{u}$ and $\bar{v}$ differ in bit x. Then it must be that $\bar{u} = n_x(\delta_1) = n_x(\delta_2)$ and $\bar{v} = n_{x+1}(\delta_1) = n_{x+1}(\delta_2)$. When x is even, $n_x(\delta_1) = n_x(\delta_2)$ implies that $\delta_1 = \delta_2$, a contradiction. When x is odd, $x+1$ is even, $n_{x+1}(\delta_1) = n_{x+1}(\delta_2)$ implies $\delta_1 = \delta_2$, a contradiction. So the first part paths of δ_1 and δ_2 are edge-disjoint.

Claim 2. *The second part paths of the packets in $\sigma_{e,k/d}$ are edge-disjoint.*

Consider a packet $\delta = (s,t) \in \sigma_{e,k/d}$. Assume that the first and second part paths of δ is $[\,f_1, f_2, \dots, f_{2l}\,]$ and $[\,g_1, g_2, \dots, g_{2j}\,]$, respectively, for some l and j. Define $q_i(\delta) = s + [\,f_1, f_2, \dots, f_{2l}\,] + [\,g_1, g_2, \dots, g_i\,]$. $q_0(\delta) = n_d(\delta)$. By the proof of Claim 1, for any two distinct packets δ_1 and δ_2, $q_0(\delta_1) \ne q_0(\delta_2)$.

We partition nodes into groups so that two nodes v_1 and v_2 are in the same group if (1) their 01 buddies match; and (2) their 10 buddies match. From $q_i(\delta)$ to $q_{i+2}(\delta)$, for any even i, some 00 buddy is changing to 11 buddy or some 11 buddy is changing to 00. So $q_i(\delta)$ and $q_{i+2}(\delta)$ are in the same group. And,

Observation 1. *For any two distinct packets δ_1 and δ_2 in $\sigma_{e,k/d}$, where $q_0(\delta_1)$ and $q_0(\delta_2)$ are in different groups. Then, for any even i and j, $q_i(\delta_1)$ and $q_j(\delta_2)$ are in different groups and, therefore, $q_i(\delta_1) \ne q_j(\delta_2)$.*

For any even i, the number of one bits in $q_i(\delta)$ minus k is even and the number of one bits in $q_{i+1}(\delta)$ minus k is odd. So,

Observation 2. *For any two packets δ_1 and δ_2 in $\sigma_{e,k/d}$, $q_i(\delta_1) \ne q_j(\delta_2)$, for any even i and odd j.*

Now consider two distinct packets δ_1 and δ_2 in $\sigma_{e,k/d}$. Assume that the second part paths of δ_1 and δ_2 share a common edge $(\bar{u}, \bar{v})$. We prove the claim by contradiction by considering the following two cases:

Case 1. $q_0(\delta_1)$ and $q_0(\delta_2)$ are in the same group.
Let $(d'_1, k'_1, e'_1, \delta'_1)$ and $(d'_2, k'_2, e'_2, \delta'_2)$ be the outputs of Procedure HALF on the inputs (d, k, e, δ_1) and (d, k, e, δ_2), respectively. Because $q_0(\delta_1)$ and $q_0(\delta_2)$ are in the same group, the source nodes of δ_1 and δ_2 are in the same group. So $d'_1 = d'_2$, $k'_1 = k'_2$, $e'_1 = e'_2$, and δ'_1 and δ'_2 are in the same input $\sigma_{e'_1,k'_1/d'_1}$. Consider the following sub-cases:

- $\bar{u} = q_i(\delta_1) = q_j(\delta_2)$ and $\bar{v} = q_{i+1}(\delta_1) = q_{j+1}(\delta_2)$ for some even i and j.
  In this sub-case $q_{i+2}(\delta_1) = q_{j+2}(\delta_2)$, because this is how the second part paths of δ_1 and δ_2 are constructed. This implies that δ'_1 and δ'_2 share some common edge, a contradiction to the induction hypothesis.
- $\bar{u} = q_i(\delta_1) = q_j(\delta_2)$ and $\bar{v} = q_{i+1}(\delta_1) = q_{j+1}(\delta_2)$ for some odd i and j.
  In this sub-case $q_{i-1}(\delta_1) = q_{j-1}(\delta_2)$. This implies that δ'_1 and δ'_2 share some common edge, a contradiction to the induction hypothesis.
- $\bar{u} = q_i(\delta_1) = q_j(\delta_2)$ for some even i and odd j.
  By Observation 2, this is impossible.
- $\bar{u} = q_i(\delta_1) = q_j(\delta_2)$ for some odd i and even j.
  By Observation 2, this is impossible.

Case 2. $q_0(\delta_1)$ and $q_0(\delta_2)$ are in different groups.
Assume that $\bar{u} = q_i(\delta_1) = q_j(\delta_2)$ and $\bar{v} = q_{i+1}(\delta_1) = q_{j+1}(\delta_2)$, for some i and j. When both i and j are even, the equality $q_i(\delta_1) = q_j(\delta_2)$ contradicts Observation 1. When both i and j are odd, the equality $q_{i+1}(\delta_1) = q_{j+1}(\delta_2)$ contradicts Observation 1. When i is even and j is odd or i is odd and j is even, the equality $q_i(\delta_1) = q_j(\delta_2)$ contradicts Observation 2.

Claim 3. *Given two packets δ_1 and δ_2 not necessarily distinct, the first part path of δ_1 and the second part path of δ_2 are edge-disjoint.*

The nodes in the first part path have k, $k+1$, or $k-1$ one bits. Consider any edge $e_1 = (n_1, n_2)$ in the first part path of δ_1. If n_1 has k one bits, then it changes some 01 buddy to 00 or changes some 10 buddy to 11 and becomes n_2. If n_1 has $k+1$ one bits, then it changes some 11 buddy to 01. If n_1 has $k-1$ one bits, then n_1 changes some 00 buddy to 10. On the other hand, Consider any edge $e_2 = (n_3, n_4)$ in the second part path of δ_2. If n_3 has k one bits, then it changes some 00 buddy to 01 or changes some 11 buddy to 10. If n_3 has $k+1$ one bits, then it changes some 01 buddy to 11. If n_3 has $k-1$ one bits, then it changes some 10 buddy to 00. All three cases in the second part path of δ_2 are different from those in the first part path of δ_1. So $e_1 \neq e_2$.

Claim 4. *Given two distinct packets δ_1 and δ_2, the path of δ_1 and the third part path of δ_2 are edge-disjoint.*

Bit $d-1$ of the target node of δ_2 is 1, which is different from all the intermediate nodes in the first part paths and the second part paths of δ_1. By Lemma 1, the targets of δ_1 and δ_2 are distinct. The claim follows.

By the four claims, we know that the paths we construct for the packets in $\sigma_{e,k/d}$ are edge-disjoint. The lemma follows. ☐

Theorem 2. *For any minimal oblivious routing algorithm A, the competitive ratio of A is at least*

$$\frac{(3/4)^{\log_2(d-1)}}{d} \cdot \sum_{k=0}^{\lfloor (d-1)/4 \rfloor} \frac{(d-1)! \cdot k!}{(d-1-k)! \cdot (2k+1)!}.$$

Proof. Let m be a large integer. Consider the bad input $\hat{\sigma}$ and the edge $\hat{e}$ mentioned in Lemma 3. Let $\sigma = (\hat{\sigma})^m$. Because A is an oblivious algorithm and by Lemma 3,

$$\begin{aligned}\exp(\text{load}_{\mathsf{A}}(\sigma, \hat{e})) &= m \cdot \exp(\text{load}_{\mathsf{A}}(\hat{\sigma}, \hat{e})) \\ &\geq m \cdot (3/4)^{\log_2(d-1)} \\ &\quad \cdot \sum_{k=0}^{\lfloor (d-1)/4 \rfloor} \frac{(d-1)! \cdot k!}{(d-1-k)! \cdot (2k+1)!}.\end{aligned}$$

Therefore,

$$\begin{aligned}&\exp(\text{cost}_{\mathsf{A}}(\sigma)) \\ &\quad = \exp(\max_{e \in \mathbb{E}}(\text{load}_{\mathsf{A}}(\sigma, e))) \\ &\quad \geq \exp(\text{load}_{\mathsf{A}}(\sigma, \hat{e})) \\ &\quad \geq m \cdot (3/4)^{\log_2(d-1)} \\ &\qquad \cdot \sum_{k=0}^{\lfloor (d-1)/4 \rfloor} \frac{(d-1)! \cdot k!}{(d-1-k)! \cdot (2k+1)!}.\end{aligned}$$

On the other hand, by Lemma 4, $\text{cost}_{\mathsf{ADV}}(\sigma) \leq m \cdot \text{cost}_{\mathsf{ADV}}(\sigma) = m \cdot d$. By the definition of competitiveness in Equation 1, the competitive ratio of A is at least $\frac{\exp(\text{cost}_{\mathsf{A}}(\sigma)) - \beta}{\text{cost}_{\mathsf{ADV}}(\sigma)}$. When m is large, β is negligible. The theorem follows. □

5 A Randomized Minimal Oblivious Routing Algorithm

In this section we provide a randomized minimal oblivious routing algorithm called MinRand and analyze its competitive performance. Consider an arbitrary input of m packets $\sigma = \{\delta_1 \delta_2 \cdots \delta_m\}$. MinRand processes each packet in σ independently. Let $\delta = \langle s, t \rangle$ be a packet in σ. MinRand chooses a minimal path for δ randomly and uniformly from all possible minimal paths connecting s and t. We analyze the competitive performance of MinRand in the following. First, it's easy to see that the following lemma is true.

Lemma 5. *Let δ be a packet and let e be an edge that $e \in SE(\delta, i_e)$ for some i_e. The expected load of δ on e is* $\exp(\text{load}_{\mathsf{MinRand}}(\{\delta\}, e)) = \frac{1}{|SE(\delta, i_e)|}$.

Theorem 3. *The competitive ratio of* MinRand *is at most*

$$\alpha_{\mathsf{MinRand}} = \sum_{k=0}^{d-1} \frac{9 \cdot d! \cdot k!}{(d-1-k)! \cdot (2k+1)!}.$$

Proof. First we show that for any input σ and any edge e, $\exp(\text{load}_{\mathsf{MinRand}}(\sigma, e)) \leq \frac{\alpha_{\mathsf{MinRand}}}{6} \cdot \text{cost}_{\mathsf{ADV}}(\sigma)$. Let edge $e_0 = 0^d \xrightarrow{(0)}$. Note that all hypercube edges are symmetrical. For any input σ and any edge e, there is an input σ' that $\exp(\text{load}_{\mathsf{MinRand}}(\sigma', e_0)) = \exp(\text{load}_{\mathsf{MinRand}}(\sigma, e))$ and $\text{cost}_{\mathsf{ADV}}(\sigma') = \text{cost}_{\mathsf{ADV}}(\sigma)$. Therefore, without loss of generality we focus on the edge $e_0 = 0^d \xrightarrow{(0)}$ in the following. Further, we assume that σ contains no packets that could never cross e_0, because such packets have no effect on the expected load on e_0. For any packet $\delta \in \sigma$, assume that $e_0 \in SE(\delta, i_\delta)$, for some i_δ.

We define the *weight* of an edge $e = (u, v)$ with respect to e_0, denoted as $w(e)$, as follows. If bit 0 of u is 0 and u has k one bits, then define $w(e) = \frac{1}{\binom{2k+1}{k} \cdot (k+1)}$; else if bit 0 of v is 1 and v has $k+1$ one bits, then define $w(e) = \frac{1}{\binom{2k+2}{k+1} \cdot (k+1)}$; otherwise define $w(e) = 0$. Consider a packet $\delta = (s, t)$ in σ. Define the *cumulative weight* of ADV on δ as $W_{\mathsf{ADV}}(\sigma) = \sum_{e \in \tau(\delta)} w(e)$, where $\tau(\delta)$ is the path taken by ADV for the packet δ. We argue in the following that $W_{\mathsf{ADV}}(\delta) \geq \exp(\text{load}_{\mathsf{MinRand}}(\delta, e)) = \frac{1}{|SE(\delta, i_\delta)|}$.

Let $\delta = (s, t)$, k_1 be the distance from s to node 0^d, and k_2 be the distance from $0^{d-1}1$ to t. It's easy to see that

(1) bit 0 of s is 0 and s has k_1 one bits;

(2) bit 0 of t is 1 and t has $k_2 + 1$ one bits;

(3) the distance from s to t is $k_1 + k_2 + 1$;

(4) $i_\delta = k_1 + 1$; and

(5) $|SE(\delta, i_\delta)| = \binom{k_1 + k_2 + 1}{k_1} \cdot (k_2 + 1)$.

Now consider the case where $k_1 \leq k_2$. No matter which path ADV takes for δ, it must crosses an outgoing edge of s. Call this edge e_1. Then,

$$\begin{aligned}W_{\mathsf{ADV}}(\delta) \geq w(e_1) &= \frac{1}{\binom{2k_1+1}{k_1}\cdot(k_1+1)}\\ &\geq \frac{1}{\binom{k_1+k_2+1}{k_1}\cdot(k_1+1)} \geq \frac{1}{\binom{k_1+k_2+1}{k_1}\cdot(k_2+1)}\\ &= \frac{1}{|SE(\delta,i_\delta)|}.\end{aligned}$$

Consider the other case that $k_1 > k_2$. In this case δ must crosses an incoming edge of t. Call this edge e_2. Then,

$$\begin{aligned}W_{\mathsf{ADV}}(\delta) \geq w(e_2) &\geq \frac{1}{\binom{2k_2+2}{k_2+1}\cdot(k_2+1)}\\ &\geq \frac{1}{\binom{k_1+k_2+1}{k_2+1}\cdot(k_2+1)} \geq \frac{1}{\binom{k_1+k_2+1}{k_1}\cdot(k_2+1)}\\ &= \frac{1}{|SE(\delta,i_\delta)|}.\end{aligned}$$

In both cases,

$$W_{\mathsf{ADV}}(\delta) \geq \frac{1}{|SE(\delta,i_\delta)|}. \qquad (2)$$

Let $I = \{\delta \,|\, e_0 \in SE(\delta,i)$, for some $i.\}$, the set of packet δ that e_0 is in some minimal path for δ. Let $m(\delta)$ be the number of packets in σ that are equivalent to δ. Because $\mathsf{MinRand}$ is an oblivious algorithm,

$$\begin{aligned}&\exp(\text{load}_{\mathsf{MinRand}}(\sigma,e_0))\\ &\qquad= \sum_{\delta\in I}\exp(\text{load}_{\mathsf{MinRand}}(\{\delta\}^{m(\delta)},e_0)).\end{aligned}$$

By Lemma 5,

$$\exp(\text{load}_{\mathsf{MinRand}}(\{\delta\}^{m(\delta)},e_0)) \leq \frac{m(\delta)}{|SE(\delta,i_\delta)|}.$$

Therefore,

$$\begin{aligned}\exp(\text{load}_{\mathsf{MinRand}}(\sigma,e_0)) &\leq \sum_{\delta\in I}\frac{m(\delta)}{|SE(\delta,i_\delta)|}\\ &= \sum_{\delta\in\sigma}\frac{1}{|SE(\delta,i_\delta)|}\\ &\leq \sum_{\delta\in\sigma}W_{\mathsf{ADV}}(\delta), \qquad \text{by Equation 2,}\\ &= \sum_{e\in E}(\text{load}_{\mathsf{ADV}}(\sigma,e)\cdot w(e))\\ &\leq \sum_{e\in E}(\text{cost}_{\mathsf{ADV}}(\sigma)\cdot w(e))\\ &\leq \text{cost}_{\mathsf{ADV}}(\sigma)\sum_{k=0}^{d-1}\Bigg(\frac{\binom{d-1}{k}\cdot d}{\binom{2k+1}{k}\cdot(k+1)}\\ &\qquad + \frac{\binom{d-1}{k}\cdot d}{\binom{2k+2}{k+1}\cdot(k+1)}\Bigg)\\ &= \text{cost}_{\mathsf{ADV}}(\sigma)\sum_{k=0}^{d-1}\frac{3\cdot d!\cdot k!}{2\cdot(d-1-k)!\cdot(2k+1)!}.\end{aligned}$$

Generally, the derivation is true for any edge in the hypercube, as a result, for any edge $e \in \mathbb{E}$,

$$\begin{aligned}&\exp(\text{load}_{\mathsf{MinRand}}(\sigma,e))\\ &\quad\leq \text{cost}_{\mathsf{ADV}}(\sigma)\sum_{k=0}^{d-1}\frac{3\cdot d!\cdot k!}{2\cdot(d-1-k)!\cdot(2k+1)!}.\end{aligned}$$

Let

$$R = 6\cdot\text{cost}_{\mathsf{ADV}}(\sigma)\sum_{k=0}^{d-1}\frac{3\cdot d!\cdot k!}{2\cdot(d-1-k)!\cdot(2k+1)!}.$$

Consider any edge $e \in \mathbb{E}$, by Chernoff's bound [14, 17],

$$P[\,\text{load}_{\mathsf{MinRand}}(\sigma,e) \geq R\,] \leq 2^{-R}.$$

Then, for any edge $e_1, e_2, \cdots \in \mathbb{E}$,

$$\begin{aligned}&P[\,\exists e,\ \text{load}_{\mathsf{MinRand}}(\sigma,e) \geq R\,]\\ &\quad= P[\,\text{load}_{\mathsf{MinRand}}(\sigma,e_1) \geq R\\ &\qquad\quad \text{or } \text{load}_{\mathsf{MinRand}}(\sigma,e_2) \geq R \text{ or } \cdots\,]\\ &\quad\leq P[\,\text{load}_{\mathsf{MinRand}}(\sigma,e_1) \geq R\,]\\ &\qquad + P[\,\text{load}_{\mathsf{MinRand}}(\sigma,e_1) \geq R\,] + \cdots\\ &\quad\leq |\mathbb{E}|\cdot 2^{-R}.\end{aligned}$$

It is easy to see that

$$\begin{aligned}&\exp(\mathrm{cost}_{\mathsf{MinRand}}(\sigma))\\ &\quad\le R + P[\,\exists e,\ \mathrm{load}_{\mathsf{MinRand}}(\sigma,e)\ge R\,]\\ &\qquad\cdot(\text{The max value of }\mathrm{cost}_{\mathsf{MinRand}}(\sigma))\\ &\quad\le R + |\sigma|\cdot|\mathbb{E}|\cdot 2^{-R}.\end{aligned}$$

Note that

$$\begin{aligned}R &= 6\cdot \mathrm{cost}_{\mathsf{ADV}}(\sigma)\cdot\sum_{k=0}^{d-1}\frac{3\cdot d!\cdot k!}{2\cdot(d-1-k)!\cdot(2k+1)!}\\ &\ge 6\cdot\frac{|\sigma|}{|\mathbb{E}|}\cdot 1,\end{aligned}$$

and

$$|\sigma|\cdot|\mathbb{E}|\cdot 2^{-R}\le\frac{|\sigma|\cdot|\mathbb{E}|}{2^{6\cdot|\sigma|/|\mathbb{E}|}}.$$

As $|\sigma|$ grows, $|\sigma|\cdot|\mathbb{E}|$ grows slower than $2^{6\cdot|\sigma|/|\mathbb{E}|}$, so $|\sigma|\cdot|\mathbb{E}|\cdot 2^{-R}=O(1)$. In particular, when $|\sigma|\ge N^2$, $|\sigma|\cdot|\mathbb{E}|\cdot 2^{-R}\le\frac{2^d\cdot d\cdot 2^{2d}}{2^{6\cdot 2^d/d}}<1$. As a result, when $|\sigma|\ge N^2$, $\exp(\mathrm{cost}_{\mathsf{MinRand}}(\sigma))\le R+1$; when $|\sigma|<N^2$, $\exp(\mathrm{cost}_{\mathsf{MinRand}}(\sigma))\le|\sigma|<N^2$. Now we are ready to say that,

$$\begin{aligned}\exp(\mathrm{cost}_{\mathsf{MinRand}}(\sigma))&\le R+N^2\\ &=\left(\sum_{k=0}^{d-1}\frac{9\cdot d!\cdot k!}{(d-1-k)!\cdot(2k+1)!}\right)\\ &\qquad\cdot\mathrm{cost}_{\mathsf{ADV}}(\sigma)+N^2.\end{aligned}$$

The theorem follows. □

6 Comparison of the Lower and Upper Bounds

Let

$$\phi(k)=\frac{(d-1)!k!}{(d-1-k)!(2k+1)!}.$$

The lower bound given in Section 4 is

$$LB=\frac{(3/4)^{\lg(d-1)}}{d}\cdot\sum_{k=0}^{\lfloor\frac{d-1}{4}\rfloor}\phi(k).$$

The upper bound given in Section 5 is

$$UB=9d\cdot\sum_{k=0}^{d-1}\phi(k).$$

Lemma 6.

$$k\ge\frac{d-7}{5}\iff\frac{\phi(k+1)}{\phi(k)}\le 1.$$

Proof. $\frac{\phi(k+1)}{\phi(k)}=\frac{(k+1)(d-1-k)}{(2k+2)(2k+3)}=\frac{2-1-k}{2(2k+3)}$. Therefore, $\frac{\phi(k+1)}{\phi(k)}\le 1\iff d-1-k\le 2(2k+3)\iff k\ge\frac{d-7}{5}$. □

Lemma 7.

$$\sum_{k=0}^{d-1}\phi(k)\le 16\cdot\sum_{k=0}^{\lfloor\frac{d-1}{4}\rfloor}\phi(k).$$

Proof. Let

$$U=\sum_{k=0}^{d-1}\phi(k)=\sum_{k=0}^{\lceil\frac{d-7}{5}\rceil-1}\phi(k)+\sum_{k=\lceil\frac{d-7}{5}\rceil}^{d-1}\phi(k),$$

$$L=\sum_{k=0}^{\lfloor\frac{d-1}{4}\rfloor}\phi(k)=\sum_{k=0}^{\lceil\frac{d-7}{5}\rceil-1}\phi(k)+\sum_{k=\lceil\frac{d-7}{5}\rceil}^{\lfloor\frac{d-1}{4}\rfloor}\phi(k).$$

Let $n(a,b)=b-a+1$, the number of integers between a and b, inclusive. Then $n(\lceil\frac{d-7}{5}\rceil,d-1)\le\frac{4d+7}{5}$, and $n(\lceil\frac{d-7}{5}\rceil,\lfloor\frac{d-1}{4}\rfloor)\ge\frac{d+3}{20}$. As a result,

$$\frac{n(\lceil\frac{d-7}{5}\rceil,d-1)}{n(\lceil\frac{d-7}{5}\rceil,\lfloor\frac{d-1}{4}\rfloor)}\le\frac{16d+28}{d+3}\le 16. \tag{3}$$

Therefore,

$$\begin{aligned}U&=\sum_{k=0}^{\lceil\frac{d-7}{5}\rceil-1}\phi(k)+\sum_{k=\lceil\frac{d-7}{5}\rceil}^{d-1}\phi(k)\\ &\le 16\cdot\sum_{k=0}^{\lceil\frac{d-7}{5}\rceil-1}\phi(k)+\sum_{k=\lceil\frac{d-7}{5}\rceil}^{d-1}\phi(k),\end{aligned}$$

by Lemma 6 and Eq. (3),

$$\le 16 \cdot \sum_{k=0}^{\lceil\frac{d-7}{5}\rceil-1} \phi(k) + 16 \cdot \sum_{k=\lceil\frac{d-7}{5}\rceil}^{\lfloor\frac{d-1}{4}\rfloor} \phi(k)$$
$$= 16 \cdot L. \qquad \square$$

Theorem 4.

$$\frac{UB}{LB} = O(d^3).$$

Proof.

$$\frac{UB}{LB} = \frac{9d^2}{(3/4)^{\lg(d-1)}} \cdot \frac{\sum_{k=0}^{d-1}\phi(k)}{\sum_{k=0}^{\lfloor\frac{d-1}{4}\rfloor}\phi(k)}$$
$$\le \frac{9d^2}{(3/4)^{\lg(d-1)}} \cdot 16 = O(d^3). \qquad \square$$

Theorem 5. *The order of the lower bound for minimal oblivious routing is*

$$\Omega(\frac{(5/4)^d}{d^5}).$$

Proof. By Sterling's formula, there are three constants x_0, c_1, and c_2 that when x is at least x_0,

$$c_1\sqrt{2\pi x}(\frac{x}{e})^x \le x! \le c_2\sqrt{2\pi x}(\frac{x}{e})^x.$$

A lower bound for $\sum_{k=0}^{\lfloor(d-1)/4\rfloor}\phi(k)$ is derived in the following. Let $d' = d-1$,

$$\sum_{k=0}^{\lfloor\frac{d-1}{4}\rfloor}\phi(k) \ge \phi(\lfloor\frac{d-1}{5}\rfloor)$$
$$= \left.\frac{(d-1)!(k)!}{(d-1-k)!(2k+1)!}\right|_{k=\lfloor\frac{d-1}{5}\rfloor}$$
$$= \left(\frac{1}{(k+1)(d-1-k)(2k+1)} \cdot \frac{(d-1)!}{(d-2-k)!} \cdot \frac{(k+1)!}{(2k)!}\right)\Bigg|_{k=\lfloor\frac{d-1}{5}\rfloor}$$
$$\ge \frac{1}{d^3} \cdot \frac{c_1\sqrt{2\pi d'}(\frac{d'}{e})^{d'}}{c_2\sqrt{2\pi(4d'/5)}(\frac{(4d'/5)}{e})^{4d'/5}} \cdot \frac{c_1\sqrt{2\pi(d'/5)}(\frac{d'/5}{e})^{d'/5}}{\cdot c_2\sqrt{2\pi(2d'/5)}(\frac{2d'/5}{e})^{2d'/5}}$$
$$= (\frac{c_1}{c_2})^2 \cdot \sqrt{\frac{5}{8}} \cdot \frac{1}{d^3} \cdot (\frac{5}{4})^{d'}.$$

As a result, our lower bound for minimal oblivious routing is

$$\frac{(3/4)^{\lg(d-1)}}{d} \cdot \sum_{k=0}^{\lfloor\frac{d-1}{4}\rfloor}\phi(k)$$
$$\ge \frac{(3/4)^{\lg(d-1)}}{d} \cdot (\frac{c_1}{c_2})^2 \cdot \sqrt{\frac{5}{8}} \cdot \frac{1}{d^3} \cdot (\frac{5}{4})^{d'}$$
$$= \Omega(\frac{(5/4)^d}{d^5}). \qquad \square$$

References

[1] W. Aiello, F. Leighton, B. Maggs, and M. Newman. Fast algorithms for bit-serial routing on a hypercube. *Mathematical Systems Theory*, 24:253–271, 1991.

[2] S. Albers. A competitive analysis of the list update problem with lookahead. *Theoretical Computer Science*, 197:95–109, 1998.

[3] B. Awerbuch and Y. Azar. Local optimization of global objectives: Competitive distributed deadlock resolution and resource allo-

cation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 240–249, 1994.

[4] B. Awerbuch, Y. Bartal, and A. Fiat. Competitive distributed file allocation. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing (STOC)*, pages 164–173, 1993.

[5] B. Awerbuch, S. Kutten, and D. Peleg. Competitive distributed job scheduling. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing (STOC)*, pages 571–580, 1992.

[6] Y. Bartal, A. Fiat, and Y. Rabani. Competitive algorithms for distributed data management. *Journal of Computer and System Sciences*, 51(3):341–358, 1995.

[7] Y. Bartal and S. Leonardi. On-line routing in all-optical networks. *Theoretical Computer Science*, 221:19–39, 1999.

[8] S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Wigderson. On the power of randomization in on-line algorithms. *Algorithmica*, 11:2–14, 1994.

[9] A. Borodin and J. Hopcroft. Routing, merging, and sorting on parallel models of computation. *Journal of Computer and System Sciences*, 30:130–145, 1985.

[10] S. Choi and A. Somani. Rearrangeable circuit-switched hypercube architectures for routing permutations. *Journal of Parallel and Distributed Computing*, 19:125–130, 1993.

[11] R. El-Yaniv, R. Kaniel, and N. Linial. Competitive optimal on-line leasing. *Algorithmica*, 25:116–140, 1999.

[12] A. Fiat, D. Foster, H. Karloff, Y. Rabani, Y. Ravid, and S. Vishwanathan. Competitive algorithms for layered graph traversal. *SIAM Journal on Computing*, 28:447–462, 1998.

[13] Q. Gu and H. Tamaki. Routing a permutatin in the hypercube by two sets of edge disjoint paths. *Journal of Parallel and Distributed Computing*, 44:147–152, 1997.

[14] T. Hagerup and C. Rub. A guided tour of Chernoff bounds. *Information Processing Letters*, 33:305–308, 1990.

[15] S. Irani. Coloring inductive graphs on-line. *Algorithmica*, 11:53–72, 1994.

[16] C. Kaklamanis, D. Krizanc, and T. Tsantilas. Tight bounds for oblivious routing in the hypercube. *Mathematical Systems Theory*, 24:223–232, 1991.

[17] F. Leighton. *Intorduction to Parallel Algorithms and Archietectures: Arrays, Trees, Hypercubes.* Morgan Kaufmann Publishers, 1992.

[18] F. Leighton, B. Maggs, and S. Rao. Packet routing and job-shop scheduling in $O(\text{congestion} + \text{dilation})$ steps. *Combinatorica*, 14(2):167–186, 1994.

[19] F. Leighton, B. Maggs, and A. Richa. Fast algorithms for finding $O(\text{congestion} + \text{dilation})$ packet routing schedules. *Combinatorica*, 19(3):375–401, 1999.

[20] F. Meyer auf der Heide and B. Vöcking. Shortest-path routing in arbitrary networks. *Journal of Algorithms*, 31:105–131, 1999.

[21] R. Ostrovsky and Y. Rabani. Universal $O(\text{congestion} + \text{dilation} + \log^{1+\epsilon} n)$ local control packet switching algorithms. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing (STOC)*, pages 644–653, 1997.

[22] S. Plotkin. Competitive routing of virtual circuits in ATM networks. *IEEE Journal of Selected Areas in Communications*, 13(6):1128–1136, 1995.

[23] M. Ramras. Congestion-free optimal routings of hypercube automorphisms. *SIAM Journal on Discrete Mathematics*, 10(2):201–208, 1997.

[24] M. Ramras. Congestion-free routings of linear complement permutations. *SIAM Journal on Discrete Mathematics*, 11:487–500, 1998.

[25] C. Scheideler and B. Vöcking. Universal continuous routing strategies. *Theory of Computing Systems*, 31(4):425–449, 1998.

[26] D. Sleator and R. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28:202–208, 1985.

[27] G. Stamoulis and J. Tsitsiklis. The efficiency of greedy routing in hypercubes and butterflies. In *Proceedings of the 3rd Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 248–259, 1991.

[28] B. Tsai and K. Shin. Combined routing and scheduling of concurrent communication traffic in hypercube multicomputers. In *IEEE 16th International Conference on Distributed Computing Systems*, pages 150–157, 1996.

[29] L. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11:350–361, 1982.

[30] L. Valiant and G. Brebner. Universal schemes for parallel communication. In *Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 263–277, 1981.

[31] T.-H. Yeh, C.-M. Kuo, C.-L. Lei, and H.-C. Yen. Competitive analysis of on-line disk scheduling. *Theory of Computing Systems*, 31:491–506, 1998.

[32] A. Youssef. Off-line permutation scheduling on circuit-switched fixed routing networks. In *IEEE 4th Symposium on the Frontiers of Massively Parallel Computation*, pages 389–396, 1992.