Theory Comput. Systems **31**, 63–66 (1998)

Theory of Computing Systems

Comments on the Oblivious Routing Algorithm of Kaklamanis, Krizanc, and Tsantilas in the Hypercube

F. K. Hwang[1] and Y. C. Yao[2]

[1]Department of Applied Mathematics, National Chiao Tung University, Hsinchu, Taiwan, Republic of China

[2]Institute of Statistical Science, Academia Sinica, Taipei, Taiwan, Republic of China

Abstract. Kaklamanis, Krizanc, and Tsantilas (1991) gave an asymptotically optimal oblivious algorithm for many–one routing in hypercubes. It is shown that their argument needs to be modified in order for the algorithm to attain the asymptotic lower bound. They also applied the algorithm to permutation routing via the many–one and one–many routing phases. They claimed to save a factor of two by proposing to divide the packets into halves, routing the first half forward and the second half backward in the bit sequence. It is shown that this idea cannot reduce the total number of steps by a factor of two.

Let $V = \{0, 1\}^d$ be the set of nodes of a d-dimensional hypercube where there is a link from a node u to another node v if their Hamming distance equals one (hence also a link from v to u). At the beginning, each node v contains a packet with destination $\pi(v)$. The problem is to route the $N = 2^d$ packets to their destinations. Under the multiport model, a node can send and receive up to d packets simultaneously, but a link can carry only one packet at a time. A routing algorithm is called *oblivious* if the path of a packet depends only on its origin and destination. When all the N destinations $\pi(v)$, $v \in V$, are distinct, the routing is a *permutation routing*; otherwise, it is a *many–one routing*. In this note we are only concerned with oblivious routing algorithms, and omit the word “oblivious” hereafter.

It is easily seen that $\lceil (N-1)/d \rceil$ is a (worst-case) lower bound on the number of steps for many–one routing. Kaklamanis, Krizanc, and Tsantilas [2] (to be referred to as KKT hereafter) gave an asymptotically optimal algorithm based on the result that a

d-cube with even d can be decomposed into d Hamiltonian circuits (or, more precisely, $d/2$ pairs of Hamiltonian circuits). Their idea is to break up the d-cube routing into a k-cube routing and a $(d-k)$-cube routing with k an even number $\leq d$. By choosing $k = \log d$ ($\log = \log_2$), they argued that the number of steps required for their algorithm is $(1+o(1))N/d$ as $d \to \infty$ (hence attaining the lower bound asymptotically). More precisely, for an even dimension d, KKT's many–one routing algorithm consists of two phases: In phase (i) a many–one routing is performed in each sub-k-cube (with the last $d-k$ bits fixed) for a suitably chosen even number k. Such a routing can be done in 2^{k-1} steps by traversing a Hamiltonian circuit. At the end of phase (i), as many as 2^k packets can cohabit at a node. In phase (ii) a routing is performed in each sub-$(d-k)$-cube (with the first k bits fixed) by evenly distributing the packets at each node to the $d-k$ Hamiltonian circuits and then running each Hamiltonian circuit at most $\lceil 2^k/(d-k)\rceil$ times. The total number of steps equals $f_d(k) = 2^{k-1} + \lceil 2^k/(d-k)\rceil 2^{d-k}$. It is readily seen that $f_d(\log d)$ does not attain the asymptotic lower bound $(1+o(1))N/d$ as $d \to \infty$, as claimed by KKT. However, observe that

$$2^{k-1} + \frac{2^k}{d-k}2^{d-k} \leq f_d(k) \leq 2^{k-1} + \left(\frac{2^k}{d-k}+1\right)2^{d-k}.$$

If $k = k(d)$ is chosen in such a way that $k - \log d \to \infty$ and $k/d \to 0$ as $d \to \infty$ (e.g., $k = 2\lfloor \log d \rfloor$), then both the lower bound and the upper bound of $f_d(k)$ are $(1+o(1))2^d/d$, so that $f_d(k) = (1+o(1))N/d$ (as $d \to \infty$).

Borodin and Hopcroft [1] introduced the idea of a two-phase permutation routing on the d-cube: Let $d = d_1 + d_2$, $d_1, d_2 > 0$. In the first phase perform many–one routing in each of the 2^{d_2} d_1-cubes where the nodes of each d_1-cube have the same last d_2 bits; in the second phase perform one–many routing in each of the 2^{d_1} d_2-cubes. In general, d_1 and d_2 should be about equal so as to minimize the number of steps. Applying KKT's many–one routing algorithm to the two-phase permutation routing in the d-cube, the number of steps required equals $(4+o(1))\sqrt{N}/d$. KKT argued that "we can save a factor of two in the running time for permutation routing by dividing the packets into halves, where, in the many–one routing phase, the first half is routed using the first collection of subhypercubes while, simultaneously, the second half is routed using the second collection; and similarly for the one–many routing phase." It is shown below that this idea cannot reduce the running time by a factor of two. (However, it does save a factor of two in the many–one routing phase, so that the total number of steps required equals $(3+o(1))\sqrt{N}/d$.)

Let the packets be divided into two parts H_0 and H_1, and correct each packet in H_0 (H_1) "forward" ("backward") as follows. Fix (even) d and $k < d/2$. In the many–one routing phase, each packet in H_0 (H_1) corrects the first (last) $d/2$ bits using KKT's many–one routing algorithm (which consists of two subphases (i) and (ii)); and in the one–many routing phase, each packet in H_0 (H_1) corrects the last (first) $d/2$ bits using the reverse of KKT's many–one routing algorithm (which also consists of two subphases (i) and (ii)). Then at the end of subphase (i) of the many–one routing phase, at most 2^{k-1} H_0-packets (H_1-packets) can cohabit at a node (if, for example, H_0 consists of those packets with origins $u = (u_{(1)}, \ldots, u_{(d)})$ satisfying $u_{(1)} + \cdots + u_{(d)} = 0 \pmod 2$).

Thus, the number of steps for the many–one routing phase equals

$$2^{k-1} + \left\lceil \frac{2^{k-1}}{d/2-k} \right\rceil 2^{d/2-k} = (1+o(1))\frac{\sqrt{N}}{d} \qquad (\text{as} \quad d \to \infty)$$

if k is suitably chosen (e.g., $k = 2\log d$). Should we correct all the packets forward, then the number of steps for the many–one routing phase would equal $(2+o(1))\sqrt{N}/d$. Thus, a factor of two is saved. However, a division which works for the many–one routing phase may not work for the one–many routing phase, i.e., at the end of subphase (i) there may be more than 2^{k-1} H_0-packets (H_1-packets) at some node. Indeed, in the following we show that for any (oblivious) division of the packets into two parts H_0 and H_1, it cannot happen that, at the end of subphase (i) in both the many–one and one–many routing phases, there are at most 2^{k-1} H_0-packets (H_1-packets) at each node.

An oblivious division of packets into two parts is determined by a set $S \subset \{(u, v) : u, v \in \{0,1\}^d\}$ in the following sense. For a given permutation π (i.e., $\pi(u)$ denotes the destination of the packet with origin u), those packets satisfying $(u, \pi(u)) \in S$ are corrected forward while the others are corrected backward (i.e., H_0 consists of those packets satisfying $(u, \pi(u)) \in S$). For $u \in \{0,1\}^d$, write $u = (u_1, u_2, u_3)$ where u_1, u_2, u_3 consist, respectively, of the first k bits, the middle $d-2k$ bits, and the last k bits of u. We say that S is (d,k)-*good* if, for every permutation π and every node $w \in \{0,1\}^d$, we have

(1) $|\{(u,\pi(u)) \in S : u_2 = w_2, u_3 = w_3, \pi(u)_1 = w_1\}| \le 2^{k-1}$,
(2) $|\{(u,\pi(u)) \in S : u_3 = w_3, \pi(u)_1 = w_1, \pi(u)_2 = w_2\}| \le 2^{k-1}$,
(3) $|\{(u,\pi(u)) \in S^c : u_1 = w_1, u_2 = w_2, \pi(u)_3 = w_3\}| \le 2^{k-1}$,
(4) $|\{(u,\pi(u)) \in S^c : u_1 = w_1, \pi(u)_2 = w_2, \pi(u)_3 = w_3\}| \le 2^{k-1}$.

Note that the cardinality of the set on the left side of (1) (resp. (3)) is the number of H_0-packets (resp. H_1-packets) at node w at the end of subphase (i) of the many–one routing phase, while the cardinality of the set on the left side of (2) (resp. (4)) is the number of H_0-packets (resp. H_1-packets) at node w at the end of subphase (i) of the one–many routing phase.

Proposition 1. *For d even and $k < d/2$, there is no (d,k)-good S.*

To establish the proposition, we need the following lemma, the proof of which is straightforward and is omitted.

Lemma.

(i) *For an $n \times n$ bipartite graph with more than $n^2/2$ edges, there exists a matching of size $> n/2$.*
(ii) *For an $n \times n$ bipartite graph with exactly $n^2/2$ edges, there exists a matching of size $\ge n/2$.*

Proof of the Proposition. Suppose S is (d,k)-good. For $a_2, b_2 \in \{0,1\}^{d-2k}$ and $a_3, b_1 \in$

$\{0,1\}^k$, let

$$S(a_2,a_3,b_1,b_2) = \{(u,v)\in S : u_2=a_2, u_3=a_3, v_1=b_1, v_2=b_2\},$$

$$S^*(a_2,a_3,b_1,b_2) = \{(u_1,v_3)\in\{0,1\}^k\times\{0,1\}^k : ((u_1,a_2,a_3),(b_1,b_2,v_3))\in S\}.$$

Clearly, $(u_1,v_3)\in S^*(a_2,a_3,b_1,b_2)$ if and only if $((u_1,a_2,a_3),(b_1,b_2,v_3))\in S(a_2,a_3,b_1,b_2)$. Noting that $S^*(a_2,a_3,b_1,b_2)$ can be naturally identified with the edge set of a $2^k\times 2^k$ bipartite graph, if $|S^*(a_2,a_3,b_1,b_2)| > 2^{2k-1}$, then, by the lemma, there exist $2^{k-1}+1$ pairs $(u_1^{(i)}, v_3^{(i)}) \in S^*(a_2,a_3,b_1,b_2)$ such that the $u_1^{(i)}$'s are all distinct and the $v_3^{(i)}$'s are all distinct. For a permutation π satisfying $\pi(u_1^{(i)},a_2,a_3) = (b_1,b_2,v_3^{(i)})$, $i=1,\ldots,2^{k-1}+1$, condition (1) is violated with $w=(b_1,a_2,a_3)$. Thus $|S(a_2,a_3,b_1,b_2)| = |S^*(a_2,a_3,b_1,b_2)| \le 2^{2k-1}$ for all $a_2,b_2\in\{0,1\}^{d-2k}$ and $a_3,b_1\in\{0,1\}^k$, from which it follows that $|S|\le 2^{2k-1}\cdot(2^{d-2k})^2\cdot(2^k)^2 = 2^{2d-1}$. The same argument can be applied to show $|S^c|\le 2^{2d-1}$. However, $|S\cup S^c| = 2^{2d}$, implying $|S|=|S^c|=2^{2d-1}$ and hence $|S^*(a_2,a_3,b_1,b_2)| = 2^{2k-1}$ for all $a_2,b_2\in\{0,1\}^{d-2k}$ and $a_3,b_1\in\{0,1\}^k$. Consider $S^*(\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0})$. By part (ii) of the lemma, there exist 2^{k-1} pairs $(u_1^{(i)},v_3^{(i)})\in S^*(\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0})$ such that the $u_1^{(i)}$'s are all distinct and the $v_3^{(i)}$'s are all distinct. Since $S^*(\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0}) = 2^{2k-1} > 2^{k-1}\times 2^{k-1}$, there must exist a pair $(u_1^*,v_3^*)\in S^*(0,0,0,0)$ such that either $u_1^*\notin\{u_1^{(1)},\ldots,u_1^{(2^{k-1})}\}$ or $v_3^*\notin\{v_3^{(1)},\ldots,v_3^{(2^{k-1})}\}$. We consider these two cases separately.

Case (i): $u_1^*\notin\{u_1^{(1)},\ldots,u_1^{(2^{k-1})}\}$. Write $u_1^{(2^{k-1}+1)} = u_1^*$ and $v_3^{(2^{k-1}+1)} = v_3^*$, so that $u_1^{(i)}$, $i=1,\ldots,2^{k-1}+1$, are all distinct. Consider $S^*(\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{1})$, which, by part (ii) of the lemma, contains 2^{k-1} pairs $(u_1'^{(i)}, v_3'^{(i)})$, $i=1,\ldots,2^{k-1}$, such that the $u_1'^{(i)}$'s are all distinct and the $v_3'^{(i)}$'s are all distinct. Pick a $u_1^{(j)}\in\{u_1^{(1)},\ldots,u_1^{(2^{k-1}+1)}\}\backslash\{u_1'^{(1)},\ldots,u_1'^{(2^{k-1})}\}$. Consider a permutation π satisfying $\pi(u_1'^{(i)},\mathbf{0},\mathbf{0}) = (\mathbf{0},\mathbf{1},v_3'^{(i)})$, $i=1,\ldots,2^{k-1}$, and $\pi(u_1^{(j)},\mathbf{0},\mathbf{0}) = (\mathbf{0},\mathbf{0},v_3^{(j)})$. It is easily checked that condition (1) is violated with $w=(\mathbf{0},\mathbf{0},\mathbf{0})$.

Case (ii): $v_3^*\notin\{v_3^{(1)},\ldots,v_3^{(2^{k-1})}\}$. Write $v_3^{(2^{k-1}+1)} = v_3^*$ and $u_1^{(2^{k-1}+1)} = u_1^*$, so that $v_3^{(i)}$, $i=1,\ldots,2^{k-1}+1$, are all distinct. Consider $S^*(\mathbf{1},\mathbf{0},\mathbf{0},\mathbf{0})$, which, by part (ii) of the lemma, contains 2^{k-1} pairs $(u_1'^{(i)}, v_3'^{(i)})$, $i=1,\ldots,2^{k-1}$, such that the $u_1'^{(i)}$'s are all distinct and the $v_3'^{(i)}$'s are all distinct. Pick a $v_3^{(j)}\in\{v_3^{(1)},\ldots,v_3^{(2^{k-1}+1)}\}\backslash\{v_3'^{(1)},\ldots,v_3'^{(2^{k-1})}\}$. Consider a permutation π satisfying $\pi(u_1'^{(i)},\mathbf{1},\mathbf{0}) = (\mathbf{0},\mathbf{0},v_3'^{(i)})$, $i=1,\ldots,2^{k-1}$, and $\pi(u_1^{(j)},\mathbf{0},\mathbf{0}) = (\mathbf{0},\mathbf{0},v_3^{(j)})$. Now, condition (2) is violated with $w=(\mathbf{0},\mathbf{0},\mathbf{0})$. This completes the proof. ☐

References

[1] A. Borodin and J. E. Hopcroft, Routing, merging and sorting on parallel models of computation, *J. Comput. System Sci.* 30 (1985), 130–145.

[2] C. Kaklamanis, D. Krizanc, and T. Tsantilas, Tight bounds for oblivious routing in the hypercube, *Math. Systems Theory* 24 (1991), 223–232.

Received January 1995, *and in final form May* 1997.