Routing a Permutation in the Hypercube by Two Sets of Edge-Disjoint Paths

Qian-Ping Gu* Hisao Tamaki†

Abstarct: *Consider a hypercube regarded as a directed graph, with one edge in each direction between each pair of adjacent nodes. We show that any permutation on the hypercube can be partitioned into two partial permutations of the same size so that each of them can be routed by edge-disjoint directed paths. This result implies that the hypercube can be made rearrangeable by virtually duplicating each edge through time-sharing (or through the use of two wavelengths in the case of optical connection), rather than by physically adding edges as in previous approaches. When our goal is to route as many source-destination pairs of the given permutation as possible by edge-disjoint paths, our result gives a 2-approximate solution which improves previous ones.*

Keywords: Permutation routing, edge-disjoint paths, algorithm, circuit-switched networks.

1 Introduction

Over years, the Boolean n-cube, or hypercube, has been one of the popular topologies for multiprocessor systems. Several commercial machines with hypercube topology have been built and a huge amount of research work, both theoretical and practical, has been done on various aspects of the hypercube. One of the most challenging open problems in the theory of interconnection networks is the *rearrangeability* of the hypercube. Let G be a directed graph with the set of input nodes $I \subseteq V(G)$ and the set of output nodes $O \subseteq V(G)$, $|I| = |O|$. We say G is *rearrangeable* if for any one-to-one mapping ρ from I onto O, we can construct edge-disjoint directed paths in G, one from each v to $\rho(v)$. In this paper, we concentrate on the case where $I = O = V(G)$ and hence ρ in the above definition is a permutation. There has been a great deal of work on the rearrangeability of various networks [3, 6, 9, 11, 12, 13, 15, 16, 17]. Szymanski [16] studied the rearrangeability of the n-dimensional hypercube H_n and conjectured that H_n is rearrangeable for every n. Here, and throughout the paper, we regard H_n as a directed graph with one edge for each direction between each pair of adjacent nodes. He proved his conjecture for $n \leq 3$ but left it open for $n \geq 4$. In fact, he gave a stronger conjecture that H_n is rearrangeable even if each input-output pair is required to be routed by a shortest path; a counterexample was later found by Lubiw [11] to this version of the conjecture.

The rearrangeability of a network is closely related to the *circuit-switched* routing capability of the network [16, 10]. Circuit-switched routing is one of the common routing models and are used to support simultaneous communications across multiprocessor parallel and telecommunication systems. In this routing model, a path is dedicated to each source-destination pair of the communication request and the data is pipelined through the path. This is in contrast to to *packet-switched routing* where each packet sequentially traverses a path from its source to destination and occupies each edge of the path one by one. The collection of paths in circuit switched routing must be edge-disjoint so as to allow parallel data transfer between every source destination pair. If we want to deal with any set of source-destination pairs specified by a permutation, rearrangeability is the property we are asking for.

An encouraging fact has long been known: it follows from the classic routing method of Beneš[3] (see also Lubiw[11] and Leighton's textbook[10]) that the hypercube is rearrangeable if each edge is doubled, i.e., each adjacent pair of nodes has two directed edges in each direction. In fact, such a doubled hypercube can provide edge-disjoint paths for two arbitrary permutations simultaneously.

Call a directed graph k-*rearrangeable*[1] if any

*The Department of Computer Software, University of Aizu, Aizu-Wakamatsu, Fukushima 965-80 Japan (qian@u-aizu.ac.jp)

†Tokyo Research Laboratory, IBM Japan, Yamatoshi, Kanagawa 242 Japan (htamaki@trl.ibm.co.jp).

[0]This research was partially supported by The Founding of Group Research Projects at The University of Aizu

[1]Lubiw[11] used the term 2-rearrangeable in a different meaning: G is 2-rearrangeable in her sense if the graph in which each edge of G is doubled is rearrangeable.

permutation can be partitioned into k partial permutations so that each of them can be routed by edge-disjoint paths. Our main result is that the hypercube is 2-rearrangeable. Despite the superficial similarity of this result to the above mentioned rearrangeability of the doubled hypercube (both are a factor of 2 away from the true rearrangeability, in a loose sense), we remark that our result is by no means an obvious consequence of the Beneš routing method, although we do build on it.

To see the strength of our result in a practical sense, consider the following approach of "virtually" realizing the rearrangeability of the hypercube without physically doubling the edges, by time-sharing for example. We partition the continuous time into unit intervals, alternately colored blue and red. We let each physical edge serve as a blue virtual edge in a blue time interval and as a red virtual edge in a red time interval. Our result allows us to partition any given permutation into a blue partial permutation and a red partial permutation so that each partial permutation can be routed using the edges of its color, making the "virtually doubled" hypercube rearrangeable. Note that the Beneš routing itself is not sufficient for this type of virtual doubling, because a path provided by the Beneš routing on the virtually doubled hypercube may contain both red and blue edges and hence may not be functional as the whole path at any given time. Similarly, our result implies the rearrangeability of an optically connected hypercube with 2 wavelengths (in the "general switch model" of Aggarwal *et al*[1]), whereas the direct application of the Beneš routing suffers from a similar problem and needs $O(n)$ wavelengths. Aumann and Rabani improved the upper bound of $O(n)$ to constant 16, where they model the hypercube as an undirected graph [2].

Our result also has an important application on the problem of determining (or approximating) the maximum number of distinguished node pairs in a graph that can be simultaneously connected via edge-disjoint paths. This problem has recently been brought into focus in the context of admission control in high-speed networks and of routing in all-optical networks [1, 14, 8, 7, 2]. An *a-approximation algorithm* for such a maximization problem is guaranteed to connect at least m/a of the given pairs by edge-disjoint paths, where m is the maximum number of pairs (from the given set of pairs) that can be connected by edge-disjoint paths. Kleinberg and Tardos[7] give a constant-approximation algorithm for a class of graphs that includes the 2-dimensional mesh. Our result immediately implies an efficient 2-approximation algorithm for this problem on H_n. (The direct application of Beneš routing gives a $(2n-1)$-approximation.)

Choi and Somani [6] recently made a significant progress in a related direction, by constructing a rearrangeable hypercube with considerably smaller number of edges than the doubled hypercube: they only need to double the edges in one fixed dimension. Their method directly implies that the hypercube is 4-rearrangeable. The result of Aumann and Rabani mentioned before uses a similar idea [2]. We borrow the innovative use by Choi and Somani of the Beneš routing as one of the key ingredients in our result.

The rest of this paper is divided into three sections. Section 2 gives basic definitions. Section 3 starts with the description of the routing methods on which our result builds and then describes our routing method that establishes the 2-rearrangeability of the hypercube. We conclude the paper with some remarks in Section 4.

2 Definitions

In this paper, the n-dimensional hypercube, denoted by H_n, is the directed graph on node set $V(H_n) = \{0,1\}^n$ such that there is an edge from $u \in V(H_n)$ to $v \in V(H_n)$ if and only if u and v differ exactly in one bit position. Figure 1 shows H_3. If u and v differ only in the ith bit position, $1 \le i \le n$, then we say the edge from u to v (and from v to u) is in dimension i. For $n \ge 1$, the 0-subcube of H_n, denoted by H^0_{n-1}, is defined to be the subgraph of H_n induced by the set of nodes whose first bit is 0. Define similarly the 1-subcube of H_n. Thus, H^0_{n-1} and H^1_{n-1} are both isomorphic to H_{n-1} and are connected to each other by edges of H_n in dimension 1.

The following *projection mappings* will be used frequently. Mapping $\pi : V(H_n) \to V(H_{n-1})$ is defined by $\pi(a_1a_2\ldots a_n) = a_2\ldots a_n$, i.e., it removes the first bit of the n-tuple identifying the node. Mapping $\pi_i : V(H_n) \to V(H^i_{n-1})$, $i = 0,1$, is defined by $\pi_i(a_1a_2\ldots a_n) = ia_2\ldots a_n$, i.e., it fixes the first bit of the n-tuple to i.

A *routing request* on a directed graph G is a multi-set R of ordered pairs of nodes of G. For each pair (u,v) in a routing request, u is called the *source* and v the *destination* of the pair. A routing request R is said to be h–k if each node appears in R at most h times as a source and at most k times as a destination. A routing request on G is called a *partial permutation* if it is 1–1;

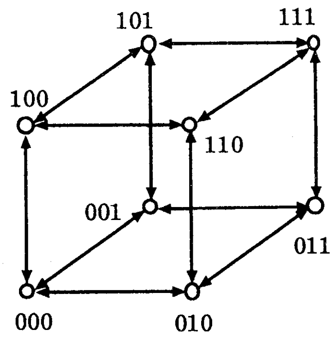

Figure 1: A 3-dimensional hypercube H_3.

a *permutation* if it is 1–1 and has exactly $|V(G)|$ pairs.

Given a routing request R on G, a *routing of* R is a multi-set P of directed paths of G that contains exactly one path from u to v for each pair $(u, v) \in R$. The *congestion* of an edge e in routing P is the number of paths of P that contain e. The *congestion of a routing* P is the maximum congestion over all edges. A routing P with congestion 1 is said to be *edge-disjoint*. We say that a directed graph G is *k-rearrangeable* if any 1–1 routing request on G can be partitioned into k routing requests each of which has an edge-disjoint routing. We say G is rearrangeable if it is 1-rearrangeable.

3 Routing methods: old, recent and new

3.1 Beneš routing

The classical routing method of Beneš[3], when applied to the hypercube, gives a routing with congestion 2 of an arbitrary 2–2 routing request. According to Lubiw[11], this application of the method of Beneš to the hypercube is folklore. Since our routing result builds on this scheme, we describe it in some detail here.

We first extend the projection mappings π, π_0, and π_1 to a routing request R by letting $\pi(R) = \{(\pi(u), \pi(v)) \mid (u, v) \in R\}$, and similarly for π_i. Here, the set notation must be interpreted as that for a multi-set, so that $(\pi(u), \pi(v))$ and $(\pi(u'), \pi(v'))$ are distinct elements of $\pi(R)$ even if they are identical, as long as (u, v) and (u', v') are distinct elements of R. The following simple lemma is central to the application of the method of Beneš to the hypercube.

Lemma 1 *Any 2–2 routing request on H_n can be partitioned into two 1–1 routing requests on H_n.*

Proof: We may regard a routing request on H_n as the set of edges in a bipartite multigraph between two copies of $V(H_n)$. It is well-known that the edge set of a bipartite multigraph with maximum degree 2 can be partitioned into two matchings (see, for example, Berge[4], Chapter 12, Theorem 2). □

Given a 2–2 routing request R on H_n, a *Beneš routing* of R is recursively constructed as follows. If $n = 0$, in which case R must consist of at most two occurrences of the pair (v_0, v_0) where v_0 is the unique node of H_0, the routing is unique and trivial: it contains at most two occurrences of an empty path from v_0 to v_0. Suppose $n \geq 1$. Using Lemma 1, partition R into two 1–1 routing requests R_0 and R_1. The idea is to route the pairs in R_0 through the 0-subcube of H_n and the pairs in R_1 through the 1-subcube. For this purpose, recursively obtain a Beneš routing P_i of the projection $\pi_i(R_i)$, $i = 0, 1$, noting that it is a 2–2 routing request on H^i_{n-1}. In our routing of R, each pair $(u, v) \in R$ is given the following path. Suppose $(u, v) \in R_i$. The path for (u, v) consists of three parts, the *prefix*, *recursive part*, and *suffix*. The prefix is empty if $u = \pi_i(u)$; otherwise it is an edge in dimension 1 from u to $\pi_i(u)$. The recursive part is the path from $\pi_i(u)$ to $\pi_i(v)$ in the subcube H^i_{n-1} given by P_i. The suffix is empty if $v = \pi_i(v)$; otherwise it is an edge in dimension 1 from $\pi_i(v)$ to v. We note that a Beneš routing of given R is not unique in general, due to the freedom of choice in partitioning R into R_0 and R_1.

Figure 2 shows a Beneš routing of the following 2–2 routing request R on H_2.

$$\begin{aligned} R \;=\; & \{(00,10), (00,01), (01,11), (01,10), \\ & (10,00), (10,00), (11,01), (11,11)\} \end{aligned}$$

The routing is diagramed in the form of the *Beneš network*: each row corresponds to a hypercube node, a diagonal line segment corresponds to a hypercube edge while a horizontal line segment indicates that the path stays on the same node, and the boxes by dotted lines correspond to the subproblems generated by the recursive process.

Each path of a Beneš routing consists of two parts: the *forward part* that consists of the edges included as the prefix (in all the recursive steps) and the *backward part* that consists of the edges included as the suffix. We call the node at which the forward part connects to the backward part the *intermediate destination* of the path. The intermediate destination of a path appears in the central column in the Beneš network representation. The following theorem is intuitively clear from the above example and easy to prove by induction.

Theorem 2 *Let P be any Beneš routing of any 2–2 routing request on H_n. Then, each edge of H_n appears in the forward part of at most one path of P and in the backward part of at most one path of P. Thus, the congestion of P is at most 2. Moreover, each node of H_n is the intermediate destination of at most two paths of P.*

3.2 Choi-Somani routing

The routing method of Choi and Somani[6], given any 1–1 routing request, constructs a routing in which every edge except in dimension 1 has congestion at most 1 and every edge in dimension 1 has congestion at most 2. Roughly speaking, their method deals with the 1–1 routing request on H_n as a 2–2 routing request on H_{n-1} and use the Beneš routing. To avoid the congestion of 2 on every edge, their key idea is to map the forward part of the Beneš routing paths to the 0-subcube of H_n and the backward part to the 1-subcube. More formally, let (u,v) be a pair in the given 1–1 routing request R on H_n and let p denote the path from $\pi(u)$ to $\pi(v)$, given by a Beneš routing of the 2–2 routing request $\pi(R)$ on H_{n-1}. Let p_f and p_b be the forward and the backward parts of p, respectively, and v_p be the intermediate destination of p. Then, the path for (u,v) in the Choi-Somani routing of R consists of five parts, the *prefix, forward part, bridge, backward part,* and *suffix.* The prefix is empty if u is in H^0_{n-1}; otherwise it is the edge from u to $\pi_0(u)$. The forward part is the isomorphic image of p_f in H^0_{n-1}, from $\pi_0(u)$ to $0v_p$. The bridge is the edge from $0v_p$ to $1v_p$. The backward part is the isomorphic image of p_b in H^1_{n-1}, from $1v_p$ to $\pi_1(v)$. Finally, the suffix is empty if v is in H^1_{n-1}; otherwise it is the edge from $\pi_1(v)$ to v.

The following theorem is a straightforward consequence of the properties of a Beneš routing (Theorem 2) and the fact on a Choi-Somani routing that (1) each edge from H^1_{n-1} to H^0_{n-1} is used at most once as a prefix and at most once as a suffix and (2) each edge from H^0_{n-1} to H^1_{n-1} is used at most twice as a bridge.

Theorem 3 *In any Choi-Somani routing of any 1–1 routing request on H_n, the congestion of each edge in dimension 1 is at most 2 and the congestion of each edge in other dimensions is at most 1.*

3.3 2-rearrangeability of the hypercube

A straightforward consequence of the Choi-Somani routing method is that the hypercube is 4-rearrangeable. To see this, suppose a 1–1 routing request R on H_n is given and partition R into $R[0]$ and $R[1]$, where $R[i]$ denotes the collection of pairs whose sources are in the i-subcube of H_n, $i = 0, 1$. Consider a Choi-Somani routing of $R[0]$. Since each path starts in H^0_{n-1}, the congestion of each edge from H^1_{n-1} to H^0_{n-1} in this routing is at most 1. The edges in the reverse direction, i.e., from H^0_{n-1} to H^1_{n-1} may have congestion 2, but we can easily partition $R[0]$ into two subsets so that the congestion is 1 for each subset. Similarly, $R[1]$ can be partitioned into two subsets (using a Choi-Somani routing in which the roles of the two subcubes are interchanged) each of which has an edge-disjoint routing. Thus, we have a partition of R into 4 subsets, each of which can be routed by edge-disjoint paths.

Here is the idea for our result. When we apply the Choi-Somani routing method to $R[0]$ in the above method, it uses a Beneš routing, which is designed for a general 2–2 routing request, to deals with $\pi(R[0])$ on H_{n-1}. However, $\pi(R[0])$ is in fact a 1–2 routing request. Our approach is to design a variant of the Beneš routing method specialized for 1–2 routing requests and to substitute it in the Choi-Somani scheme. By doing so, we hope to exploit the special property to obtain a routing of $R[0]$ with congestion 1.

The following lemma, similar to Lemma 1, is the key to such a variant of the Beneš routing method.

Lemma 4 *Any 1–2 routing request R on H_n can be partitioned into two routing requests R_0 and R_1 so that $\pi(R_i)$ for each i is a 1–2 routing request on H_{n-1}.*

Proof: Given a 1–2 routing request R on H_n, construct a bipartite multi-graph G_R between $V(H_{n-1})$ and $V(H_n)$ by drawing an edge between $\pi(u)$ and v for each pair $(u,v) \in R$. Then, the maximum degree of G_R is 2 and therefore the edge set of G_R can be partitioned into two matchings. Since the pairs in R are in one-to-one correspondence with the edges of G_R, this partition induces a partition of R into two subsets R_0 and R_1. For each node w of H_{n-1}, there is at most one pair (u,v) in R_0 such that $\pi(u) = w$ and at most two pairs (u,v) such that $\pi(v) = w$, because R_0 corresponds to a matching of G_R. Therefore, $\pi(R_0)$ is a 1–2 routing request. Similarly, $\pi(R_1)$ is also a 1–2 routing request. □

Given a 1–2 routing request R on H_n, a *1–2 Beneš routing* of R is recursively constructed as follows. The construction is similar to the one for a Beneš routing described previously. The base case $n = 0$ is trivial. Suppose $n \geq 1$. Using Lemma 4, partition R into two routing requests R_0 and R_1 so that $\pi(R_i)$ is a 1–2 routing request for $i = 0, 1$. Recursively obtain a 1–2 Beneš routing P_i of $\pi_i(R_i)$, $i = 0, 1$, on H^i_{n-1}. The path for each $(u, v) \in R$ consists of the prefix, recursive part, and suffix, exactly in the same way as before. We list their description again for the convenience of the reader. Suppose $(u, v) \in R_i$. The prefix is empty if u is in H^i_{n-1}; otherwise it is an edge in dimension 1 from u to $\pi_i(u)$. The recursive part is the path from $\pi_i(u)$ to $\pi_i(v)$ in H^i_{n-1} given by P_i. The suffix is empty if v is in H^i_{n-1}; otherwise it is an edge in dimension 1 from $\pi_i(v)$ to v.

The forward part, backward part, and intermediate destination of a path in a 1–2 Beneš routing are defined similarly as before. Figure 3 shows a 1–2 Beneš routing of the following 1–2 routing request R on H_2.

$$R = \{(00, 01), (01, 11), (10, 00), (11, 01)\}$$

For a comparison, Figure 4 shows a general Beneš routing of the same 1–2 routing request R.

As is seen from the above example, an advantage of a 1–2 Beneš routing over a general Beneš routing applied to a 1–2 routing request, is that at most one path, rather than two, chooses each node of H_n as its intermediate destination.

Lemma 5 *Let P be any 1–2 Beneš routing of any 1–2 routing request on H_n. Then, each edge of H_n appears in the forward part of at most one path of P and in the backward part of at most one path of P. Moreover, each node of H_n is the intermediate destination of at most one path of P.*

Theorem 6 *H_n is 2-rearrangeable.*

Proof: Let R be an arbitrary 1–1 routing request on H_n. Let $R[i]$, $i = 0, 1$, denote the collection of pairs in R that have their source in the i-subcube. Let P_0 be a 1–2 Beneš routing of $\pi(R[0])$ on H_{n-1}. Let (u, v) be a pair in $R[0]$, let p be the path from $\pi(u)$ to $\pi(v)$ in P_0, and let p_f and p_b the forward and backward part of p respectively. As in Choi-Somani routing, p_f is mapped to the isomorphic path p'_f in H^0_{n-1}, p_b is mapped to the isomorphic path p'_b in H^1_{n-1}, with a bridge edge connecting the last node of p'_f to the first node of p'_b: call the resulting path p'. The congestion of each bridge edge is at most 1 due to Lemma 5. The path from u to v is completed by adding a suffix (i.e., an edge from H^1_{n-1} to H^0_{n-1}) to p', if necessary. Since no prefix is needed, we obtain a routing of $R[0]$ with congestion 1. By symmetry, we can construct a routing with congestion 1 of $R[1]$. □

Remark: The above description of the construction of our routing can readily be translated into an efficient algorithm. Since the partition in Lemma 4 can be computed in linear time (bicoloring the edge set of a bipartite graph with maximum degree 2), the entire algorithm runs in $O(N \log N)$ time, the same complexity as for the Beneš routing, where $N = 2^n$ is the number of nodes of H_n. See Waksman[17], Nassimi and Sahni[12], and Carpinelli and Oruc[5] for algorithmic issues in Beneš-type routing.

4 Concluding Remarks

Szymanski conjectured that the n-dimensional hypercube H_n is rearrangeable for every n and proved his conjecture for $n \leq 3$. We proved in this paper that H_n is 2-rearrangeable for all n, i.e., every permutation on H_n can be partitioned into two partial permutations each of which can be routed by edge-disjoint paths. However, the conjecture of Szymanski is still open for $n \geq 4$ (although a rather brute force proof may work for $n = 4$). Some directions of further research include:

1. Find a special (yet general enough to be interesting) class of permutations, every member of which is routable by edge-disjoint paths. One such example can be found in Sprague and Tamaki[15].

2. Find some upper bound f on the size of the 1–1 routing request on H_n such that any R with $|R| \leq f$ can be routed by edge-disjoint paths. We conjecture that $f = c|V(H_n)|$, for some constant c, is such an upper bound. Note that this is a weaker conjecture than Szymanski's but appears considerably harder to prove than the constant-rearrangeability.

3. Our result implies a 2-approximation algorithm for the problem of routing as many pairs as possible from a given 1–1 routing request. Find a $(1+\epsilon)$-approximation algorithm for some $\epsilon < 1$.

References

[1] A. Aggarwal and A. Bar-Noy and D. Coppersmith and R. Ramaswami and B. Schieber and M. Sudan. Efficient Routing and Scheduling Algorithms for Optical Networks. *Proc. of the 5th ACM-SIAM Symposium on Discrete Algorithms*, pages 412–423, 1994.

[2] Y. Aumann and Y. Rabani. Improved bounds for all optical routing. In *Proc. of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 567–576, 1995.

[3] V.E. Benes. *Mathematical Theory of Connecting Networks and Telephone Traffic.* Academic Press, 1965.

[4] C. Berge. *Graphs and Hypergraphs.* North-Holland, 1973.

[5] J.D. Carpinelli and A. Y. Oruc Applications of Matching and Edge-Coloring Algorithms to Routing in Clos Networks *Networks*, 24:319–326, 1994.

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

[7] J. Kleinberg and E. Tardos. Disjoint paths in densely embedded graphs. To appear in *Proc. of the 36th IEEE Symposium on Foundation of Computer Science*, 1995.

[8] J. Kleinberg and E. Tardos. Approximations for the disjoint paths problem in high diameter planar networks. In *Proc. of the 27th Annual ACM Symposium on the Theory of Computing (STOC'95)*, pages 26–35, 1995.

[9] K.Y. Lee. On the rearrangeability of $2\log_2 n) - 1$ stage permutation networks. *IEEE Trans. on Comput.*, C-34:412–425, 1985.

[10] F.T. Leighton. *Introduction to Parallel Algorithms and Architecture: Arrays·Trees·Hypercubes.* Morgan Kaufmann, 1992.

[11] A. Lubiw. Counterexample to a Conjecture of Szymanski on Hypercube Routing. *Information Processing Letters*, 29:57-61, 1990.

[12] D. Nassimi and S. Sahni. Parallel algorithms to set up the benes permutation network. *IEEE Trans. on Comput.*, C-31:148–154, 1982.

[13] N. Pippenger and G. Lin. Fault-tolerant circuit-switching networks. In *Proc. of the 4th ACM Symposium on Parallel Algorithms and Architectures (SPAA'92)*, pages 229–235, 1992.

[14] P. Raghavan and E. Upfal Efficient Routing in All-Optical Networks In *Proc. of the 26th ACM Symposium on Theory of Computing*, pages 134–143, 1994

[15] A.P. Sprague and H. Tamaki. Routings for involutions of a hypercube. *Discrete Applied Math.*, 48:175–186, 1994.

[16] T. Szymanski. On the permutation capability of a circuit-switched hypercube. In *Proc. Inthernational Conference on Parallel Processing (I)*, pages 103–110, 1989.

[17] A. Walksman. A permutation network. *J. ACM*, 15:159–163, 1968.

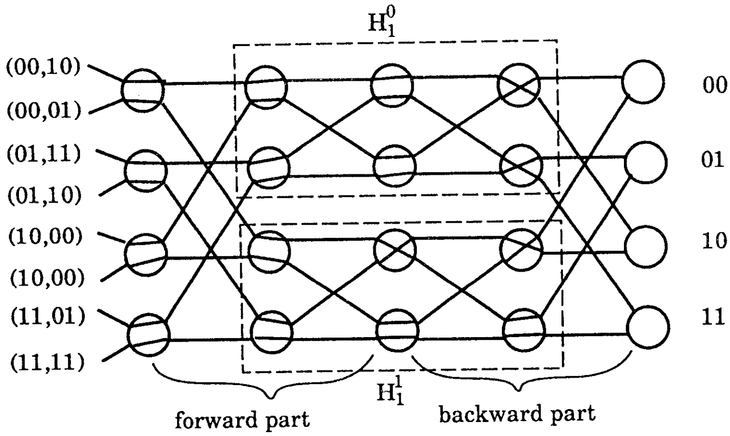

Figure 2: A Beneš routing for the 2–2 routing request on H_2.

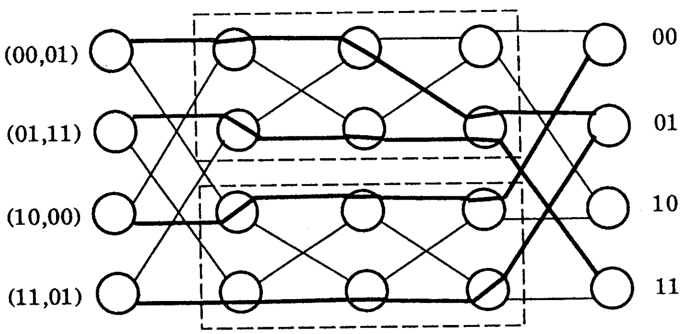

Figure 3: A 1–2 Beneš routing for the 1–2 routing request in H_2.

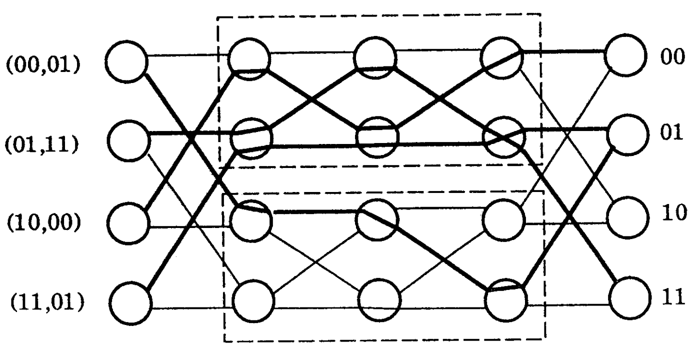

Figure 4: A general Beneš routing for the 1–2 routing request in H_2.