- While $R \neq P$ do
  1. Pick $(s,t) \in P - R$.
  2. Route (s,t) through B^0, i.e. route from s to s^0 and from t to t^0.
     Set $R := R \cup \{(s,t)\}$.
  3. Find $(s',t') \in P$, such that $t' = \text{Pair}(t)$ (then $(s',t') \notin R$). Route (s',t') through B^1.
     Set $R := R \cup \{(s,t)\}$.
  4. Find $(s,t) \in P$ such that $s = \text{Pair}(s')$. If $(s,t) \notin R$ then goto 2.

Since for each t there is exactly one $\text{Pair}(t)$, and for s' exactly one $\text{Pair}(s')$, it follows that the procedure completes with each request routed exactly once. By construction, the induced patterns on B^0 and B^1 are sparse. This is because we see to it that if (s,t) are routed through, say B^0, then both $(\text{Pair}(s),t')$ and $(s',\text{Pair}(t))$ are routed through B^1. Thus, by definition, $\bar{s^0}$ and $\bar{t^0}$, are not used in the routing. Thus, the pattern on B^0 is sparse. Similarly for B^1.

Thus, we proved that any sparse pattern can be routed using sparse paths. I.e. at each layer, the set of nodes used for routing is sparse. Thus, if u is used then $\bar{u}$ is not used. If two edges, e and $\bar{e}$, are crossing edges, and $e = (u,v)$ then $\bar{e} = (\bar{u},v')$. Thus, an sparse routing uses at most one of any pair of crossing edges. It thus remains to show how, given a full permutation pattern P, to split it into two sub-permutation each of which is sparse. This is done in analogy to the inductive procedure presented above. (The full permutation P can be viewed as an sparse pattern on a network of one dimension larger. The details are omitted). ∎

works. *IEEE Trans. Comm.*, 1993.

[BT91] K. Bala and T.E. Stern. Algorithms for routing in a linear lightwave network. In *Proceedings of INFOCOM*, pages 1–9, 1991.

[Ben65] V. Beněs. *Mathematical Theory of Connecting Networks and Telephone Traffic*, Academic Press, New York, 1965.

[Che90] K.W. Cheng. Accousto-optic tunable filters in narrowband wm networks. *IEEE JSAC*, 8:1015–1025, 1990.

[GGK+93] J. Garay, I. Gopal, S. Kutten, Y. Mansour, and M. Yung. Efficient on-line call control algorithms. In *Proceedings of the 2nd Annual Israel Conference on Theory of Computing and Systems*, 1993.

[GVY93] N. Garg, V.V. Vazirani, and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees, with applications to matching and set cover. In *Proceedings of ICALP '93*, pages 64–75.

[KG92] M. Kevacevic and M. Gerla. Rooted routing in linear lightwave networks. In *Proceedings of INFOCOM*, pages 39–48, 1992.

[Kle94] J. Kleinberg. Private communication.

[KLPS90] B. Korte, L. Lovász, H.J. Promel, and A. Schrijver, editors. *Paths, flows, and VLSI-layout.* Springer-Verlag. 1990.

[Kun91] M. Kunde. Concentrated regular data streams on grids: sorting and routing near to the bisection bound. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science*, 1991, pages 141–150.

[LR88] F.T. Leighton and S. Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In *Proceeding of the 29th Annual IEEE Symposium on Foundations of Computer Science*, pages 422–431, 1988.

[Pan92] R.K. Pankaj. *Architectures for Linear Lightwave Networks.* PhD thesis, MIT, 1992.

[PS93] G.R. Pieris and G.H. Sasaki. A linear lightwave Beněs network. *IEEE/ACM Trans. on Networking*, 1993.

[RU94] P. Raghavan and U. Upfal. Efficient routing in all optical networks. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 133–143, 1994.

[Sch93] A. Schrijver. Complexity of disjoint paths problems in planar graphs. In *Proceedings of ESA '93.*

[SM93] M. Settembre and F. Matera. All optical implementations of high capacity TDMA networks. *Fiber and Integrated Optics*, 26:8–18, 1993.

[Wak68] A. Waksman. A permutation network. *Journal of the ACM*, 15(1):159–163, January 1968.

Appendix

Proof of Lemma 5.1: For a node $u = (i, x)$, denote $\bar{u} = (i, x^{|i|})$. Let X be a set of nodes. We say that X is *sparse* if

$$u \in X \Rightarrow \bar{u} \notin X \ .$$

Say a communication pattern $P = \{(s_i, t_i)\}$ is *sparse* if the sets $\{s_i\}$ and $\{t_i\}$ are both sparse. A set of paths is *sparse* if the set of nodes used in the paths is sparse. We first show how to route any sparse pattern using sparse paths.

The proof proceeds by induction of d, the dimension of the network (B_d is the network with $2d+1$ layers). For $d = 1$ by inspection. Assume for $d-1$, we prove for d. Let P be a communication pattern for B_d. W.l.o.g. assume $|P| = 2^{d-1}$. Thus, for each u of the first layer either $u \in \{s_i\}$, or $\bar{u} \in \{s_i\}$. Similarly, for v of the last layer and the set $\{t_i\}$. If one removes layers d and $-d$ of B_d, then one is left with two copies of B_{d-1}, an *upper* copy and a *lower* copy, denoted B^0 and B^1. We split P into two subsets P_0 and P_1. Subset P_0 is routed through B^0, and P_1 through B^1. We guarantee that the induced patterns on B^0 and B^1 are both sparse. By induction, we can then complete the routing of the patterns on their respective networks.

For $x = (x_1, \ldots, x_d) \in \{0,1\}^d$ denote $x||0 = (x_1, \ldots, x_{d-1}, 0)$ and $x||1 = (x_1, \ldots, x_{d-1}, 1)$. For $(s, t) = ((-d, x), (d, y))$ denote $s^b = (-d+1, x||b)$ and $t^b = (d-1, x||b)$, $b = 0, 1$. Consider the set $P = \{(s_i, t_i)\}$. For s_i denote

$$\text{Pair}(s_i) \stackrel{\text{def}}{=} s_j \text{ s.t. } \exists t_j \text{ with } (s_j, t_j) \in P \text{ and } s_j^0 = \bar{s_i^0} \ .$$

Since P is sparse, there is at most one $s_j \in \{s_i\}$ such that $s_j^0 = \bar{s_i^0}$. Since P is of maximal size, there is always one such s_j. Thus, the function Pair$(\cdot)$ is well defined. Similarly, for t_i let

$$\text{Pair}(t_i) \stackrel{\text{def}}{=} t_j \text{ s.t. } \exists s_j \text{ with } (s_j, t_j) \in P \text{ and } t_j^0 = \bar{t_i^0} \ .$$

The routing of P to B^0 and B^1 is performed using the following procedure:

- Set $R := \emptyset$.

realize R_{00} using six wavelengths, and R_{01} using two additional wavelengths. A symmetric argument allows to realize R_{11} with six additional wavelengths, and R_{10} with two additional wavelengths, thus proving the theorem.

Denote $H_0 = \{x \in H : x_0 = 0\}$ and $H_1 = \{x \in H : x_0 = 1\}$. Consider R_{01}. To realize this set, we first embed the $(2d-1)$-layer Beněs network in the hypercube, with the nodes of H_0 as the input nodes of the network, and those of H_1 as the output nodes. The embedding is straight forward: the first d layers are mapped to the corresponding nodes in H_0 and the last d layers mapped to H_1. Note that the nodes in the middle layer of B are mapped twice. For each such node, there is a single edge connecting its two copies. Edges of B directly translate to H, either as corresponding edges in H_0 and H_1, or else both ends of the edge are mapped to the same node. For the middle layer, the edges that connect it to the previous layer are mapped to edges in H_0. The edges that connect it to the next layer are mapped to edges in H_1. This embedding has edge congestion two, because every pair of crossing edges of the Beněs network is mapped to a single edge of the hypercube.

On the Beněs network, any input-output permutation can be routed using node disjoint paths, and any 2-relation can be routed using edge disjoint paths. In order to overcome the edge congestion two in the above embedding, we require a somewhat different routing scheme on the Beněs network, provided by Lemma 5.1. We show there that any permutation can be partitioned into two sub-permutations, each of which can be routed on the Beněs network so that at most one of every pair of crossing edges is used.

We route R_{01} on the hypercube using the embedded Beněs network by applying Lemma 5.1 (notice that the lemma applies trivially to a sub-permutation). R_{01} is thus partitioned into two *batches*, each routed using a separate wavelength. Since at most one path in each batch passes through any middle layer node in the Beněs network, we can use the edge connecting the two copies of such a node to cross the two parts of the hypercube. This completes the determination of the routes for the R_{01} connections.

The R_{00} connections are routed as follows. First, complement the most significant bit of each destination (so it becomes a 1). Then, route all connections to these new destinations using the same method as for R_{01}. Now, the routes still have to connect to the correct destination. Only the most significant bit has to be corrected. For this, the edges connecting H_1 to H_0 are used again. It is easy to see that for each of the two batches, any path may intersect at most two other paths. (All the intersections occur on the edges connecting H_0 and H_1. A path may intersect some other path on its way across the Beněs network, and another path when it corrects the most significant bit of its destination.) Therefore, the paths in each batch can be colored using 3 colors (wavelengths), so that no two paths with the same color share an edge. ■

Acknowledgements

We are indebted to Tom Leighton for the idea of reducing realization to throughput. We thank Boaz Patt-Shamir for illuminating discussions.

References

[ABC+94] A. Aggarwal, A. Bar-Noy, D. Coppersmith, R. Ramaswani, B. Schieber, and M. Sudan. Efficient routing and scheduling algorithms for optical networks. In *Proceedings of the 5th Annual ACM-SIAM Symposuim on Discrete Algorithms*, pages 412–423, 1994.

[AAP93] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput competitive on-line routing. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, November 1993.

[ABFR94] B. Awerbuch, Y. Bartal, A. Fiat, and A. Rosén. Competitive non-preemptive call control. In *Proceedings of 5th ACM-SIAM Symposium on Discrete Algorithms*, 1994.

[AGLR94] B. Awerbuch, R. Gawlick, F.T. Leighton, and Y. Rabani. On-line admission control and circuit routing for high performance computing and communication. To appear in *Proceedings of FOCS '94*.

[BH92] R.A. Barry and P.A. Humblet. Bounds on the number of wavelengths needed in WDM networks. In *LEOS'92 Summer Topical Mtg. Digest*, pages 114–127, 1992.

[BH93] R.A. Barry and P.A. Humblet. On the number of wavelengths and switches in all optical net-

these wavelengths there must be at least one which routes at least $\frac{k/\log n}{O(1)}$ requests. ∎

Higher dimension arrays are treated similarly. Again, the full array is divided into boxes of equal side-length, for lengths $1, 2, 4, \ldots, n^{1/d}$. The main technical problem is constructing a small crossbar between neighboring boxes. We do this using a routing algorithm of Kunde [Kun91] for multi-dimensional arrays. A straight-forward implementation would give an additional factor of $2^{O(d \log d)}$ for the d-dimensional array. With some extra effort, this factor can be reduced to $2^{O(d)}$. The details are deferred to the full version of the paper.

4.2 From Throughput to Realization

Provided a procedure to approximate the throughput, the following algorithm obtains a full realization:

- Set $A = R$
- While $A \neq \emptyset$
  1. Realize an $\Omega(1/\log n)$ fraction of the maximum throughput of A using one wavelength. Denote the set thus realized by A'.
  2. $A := A - A'$.

THEOREM 4.2. *Assume that R can be realized using k wavelengths, then the above algorithm realizes R using at most $O(k \log |R| \log n)$ wavelengths.*

Proof: Since R can be realized with k wavelengths, so can any subset of R. Thus, for any subset $A \subseteq R$, there must be one wavelength which routes at least $1/k$ of the requests of A. Thus, for each subset A of R, the maximum throughput of A is at least $|A|/k$. The procedure from the previous section provides an $O(\log n)$ approximation for the throughput. Thus, at each round of the algorithm, the size of A decreases by at least a $\frac{1/k}{\Omega(\log n)}$ factor. Hence, after at most $O(k \log |R| \log n)$ iterations all requests are realized. ∎

5 Routing on the Hypercube

By using the standard embedding of the Beněs network in the hypercube, one can route any permutation with edge congestion four. However, since every path may intersect as many as $\Theta(\log n)$ other paths, it may be required to use that many wavelengths. In order to obtain a realization of any permutation with a constant number of wavelengths, we use a different embedding of the Beněs network.

THEOREM 5.1. *Any permutation on the hypercube can be realized using $O(1)$ wavelengths.*

For the proof, we need a technical lemma. Let $B_d = (V, E)$ be the $2d+1$ layer Beněs network, with

$$V = \left\{(i, x) : i \in \{-d, \ldots, 0, \ldots d\}, x \in \{0,1\}^d\right\},$$

and $E_B = E_S \cup E_D$,

$$E_S = \{((i, x), (i+1, x)) : -d \leq i < d\}$$

$$E_D = \{((i, x), (i-1, x^i)) : 1 \leq i \leq d\}$$
$$\cup \{((i, x), (i+1, x^{|i|})) : -d \leq i \leq 0\},$$

(where x^j is the string x with the j-*th* bit complemented). We call the nodes $\{(i, x)\}$ the i-*th* layer of the network. The set E_S we call *straight* edges, and E_D *diagonal* edges. For each layer, we group the diagonal edges leaving the layer in pairs. Edge $((i, x), (i+1, x^j))$ is paired with $((i, x^j), (i, x))$ (i.e. the pairs match edges which connect the same pair of rows). The edges of a pair we call *crossing edges.*

LEMMA 5.1. *Let B be a Beněs network, and let P be a permutation pattern, with sources in the first layer of B, and destinations in the last. Then, P can be efficiently separated into two sub-permutation patterns P_0 and P_1 such that each P_b can be efficiently routed in B using node disjoint paths which use at most one edge of each pair of crossing edges.*

The proof appears in the appendix.

Proof of Theorem 5.1: Consider a d-dimensional hypercube, H. Use the standard representation of the nodes of H, as binary d-tuples. Let R be a permutation pattern for H. For a node x, let x_0 be the first bit in the representation of x. Partition R into four sub-permutations, R_{00}, R_{01}, R_{10}, and R_{11}, where $R_{ab} = \{(x, y) \in R : x_0 = a,\ y_0 = b\}$. Each R_{ab} is a sub-permutation on $(d-1)$ tuples. We show how to

t_i, and finally, within T, to v_i on the boundary (see Figure 2). Call u_i the *inlet* node of the path, and v_i the *outlet* node. We use the path P_i to determine the route between s_i and t_i in the original graph G. Specifically, within S the route follows the corresponding segments of P_i, from s_i to u_i. Similarly, within T, from v_i to t_i. Since f is a flow, by definition, the path segments within S and T, for all $(s_i, t_i) \in A$, are edge disjoint. Hence, they can all use the same wavelength. It thus remains to determine the path segments between the outlets and the inlets.

Let $B(S,T)$ be the set of edges of distance at most $2 \cdot 2^\ell$ from $S \cup T$. The region $B(S,T) - (S \cup T)$ forms a crossbar structure between the set of nodes on the boundary of S and those on the boundary of T. Thus, all inlet-outlet pairs can be routed simultaneously within $B(S,T)$ using edge disjoint paths. ∎

LEMMA 4.2. *Assume k requests of $R(S,T)$ can be routed using edge disjoint paths. Then there exists an integral flow f, for the corresponding graph $X_{S,T}$, such that $|f| = k$.*

Proof: Let $A \subseteq R(S,T), |A| = k$, be a set of requests that can be routed with edge disjoint paths. Let $\mathcal{P}$ be the set of paths for routing A, with $P_i \in \mathcal{P}$ routing $(s_i, t_i) \in A$. The routes of $\mathcal{P}$ are edge disjoint. In particular, the segments within S and T are disjoint. Accordingly, we construct the following flow in $X_{S,T}$. For each $P_i \in \mathcal{P}$ put a unit flow on the path segment within S, and a unit flow on the segment within T. Let the flow run from the boundary of S to the s_i, and from t_i to the boundary of T. In addition, put a unit flow on the edge (s_i, t_i). This is a legal flow of size $|A| = k$. ∎

COROLLARY 4.1. *For any S and T, one can route efficiently the maximum throughput of $R(S,T)$ using edge disjoint paths.*

Proof: Solve the corresponding integer max-flow problem and use Lemma 4.1 to find the routes. By Lemma 4.2 this is a maximum set. ∎

The same holds also for set of requests from S to U, the square 2^ℓ columns to the right of S (see Figure 1). In a similar manner we can also route the maximum throughput for the set of requests between S and its diagonal neighboring square, denoted by V, and to the square W, on the other diagonal.

Consider the set $\mathcal{S}_\ell$ of all squares in the *ℓ-th* level division. For each $S \in \mathcal{S}_\ell$ let T_S, U_S, V_S, W_S, be the 2^ℓ above, 2^ℓ to the right, left diagonal and right diagonal squares, respectively. Set $\Gamma(S) = \{T_S, U_S, V_S, W_S\}$. Consider the set (of sets)

$$\mathcal{R}_\ell = \{R(S,X) : S \in \mathcal{S}_\ell, X \in \Gamma(S)\} .$$

Set $R_\ell = \bigcup_{R(S,X) \in \mathcal{R}_\ell} R(S,X)$. We call R_ℓ the *ℓ-th level requests.*

LEMMA 4.3. *Let k be the maximum number of requests in R_ℓ which can be routed simultaneously using edge disjoint paths. At least k requests of R_ℓ can be realized efficiently using $O(1)$ wavelengths.*

Proof: For each $R(S,X) \in \mathcal{R}_\ell$ we can efficiently route the maximum throughput using edge disjoint paths, and hence one wavelength. Denote the set thus routed by $A(S,X)$. The routes for $A(S,X)$ remain within a $2 \cdot 2^\ell$ distance from $S \cup X$. Thus, the routes for $A(S,X)$ overlap with the routes of at most a constant number of other sets $A(S',X') \subseteq R(S',X')$, $R(S',X') \in \mathcal{R}_\ell$. Hence, all sets $A(S,X)$ can be realized simultaneously using at most $O(1)$ wavelengths. Let $M \subseteq R_\ell$ be the maximum set of requests which can routed with one wavelength. Clearly,

$$\left| \bigcup_{S \in \mathcal{S}_\ell, X \in \Gamma(S)} A(S,X) \right| \geq |M| = k .$$

∎

We thus obtain:

THEOREM 4.1. *Let G be an n node torus and R a communication pattern for G. Assume that k requests of R can be routed simultaneously using edge disjoint paths. Then, one can deterministically and efficiently (in polynomial time) find a set $A \subseteq R$, $|A| = \Omega(k/\log n)$ and edge disjoint routes for all requests in A.*

Proof: There are $\log \sqrt{n} = O(\log n)$ divisions $\mathcal{S}_\ell$. Each request $(s,t) \in R$ appears in one and only one of the request sets R_ℓ. Thus, there must be at least one set R_ℓ, for which the maximum throughput of R_ℓ is at least $k/\log n$. By Lemma 4.3, the maximum throughput of the set can be efficiently realized with a constant number of wavelengths. Of

other communication paths. Thus, using greedy wavelength assignment, $O(\log^4 n/\beta^2)$ wavelengths suffice to realize all the requests. ∎

4 Bounded Dimension Arrays

Let G be an n node d dimensional array, and let $R = \{(s_1, t_1), \ldots, (s_m, t_m)\}$ be a set of communication requests for G. Define the *maximum throughput* of R to be the maximum subset $A \subseteq R$ which can be realized concurrently with one wavelength, i.e. using edge disjoint paths. First we present an $O(\log n)$ approximation algorithm for the maximum throughput. We then show that this yields an $O(\log n \log |R|)$ approximation algorithm for the number of wavelengths necessary to realize the entire set R.

4.1 Edge Disjoint Paths

First we present the algorithm for the two dimensional case. For simplicity we describe the algorithm for the torus. The modification for the mesh is straight forward.

Let G be an $\sqrt{n} \times \sqrt{n}$ torus. For $\ell = 0, 1, \ldots, \log\sqrt{n}$, divide G into squares of side-length 2^ℓ. Consider one such division and a square S in the division, as depicted in Figure 1. Let T be

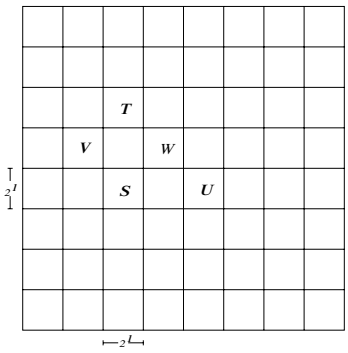

Figure 1: A division, a sub-mesh S and its neighboring area.

the square 2^ℓ rows above S, as depicted in Figure 1. Let $R(S,T)$ be the set of requests with one endpoint in in S and the other in T. W.l.o.g. assume all sources are in S and all destination in T. We now show how to route the maximum throughput of $R(S,T)$ with edge-disjoint paths.

Let $S = (V_s, E_s)$ and $T = (V_t, E_t)$. Consider the graph $X = X_{S,T} = (V_x, E_x)$ with $V_x = V_s \cup V_t$, and $E_x = E_s \cup E_t \cup R(S,T)$ (we use $R(S,T)$ both as a set of requests and as a set of edges). The graph X is the union of the graphs S and T, with an additional edge drawn between the source and sink of each communication request in $R(S,T)$. Consider the following integer (single commodity) max-flow problem on X. Direct all the edges of $R(S,T)$ (in X) *from S to T*. All other edges are undirected. Assign capacity one to all edges. Let the boundary nodes of S be source nodes for the flow, and boundary nodes of T the sink nodes (this can be done by adding a source node s connected with infinite capacity edges to all boundary nodes of S, and a sink node t connected with infinite capacity edges to all boundary nodes of T).

LEMMA 4.1. *Let f be an integer flow in X and $|f|$ be the amount of flow in f. Given f one can efficiently find a realization for $|f|$ requests of $R(S,T)$ using edge-disjoint paths. The paths remain within a distance of $2 \cdot 2^\ell$ from $S \cup T$.*

Proof: Flow f defines edge disjoint paths in X. Any such path uses exactly one edge of the form (s_i, t_i). Moreover, each such edge is used at most once. Let A be the set of edges of the from (s_i, t_i) used in f. Then $|A| = |f|$. The set A naturally corresponds to a set of communication requests. We show how to realize all requests in A using one wavelength.

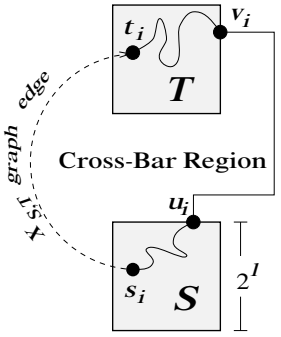

Figure 2: The paths from S to T.

Consider a path $P_i \in f$, which uses the edge (s_i, t_i). Path P_i starts at u_i, on the boundary of S, continues within S to s_i, then through (s_i, t_i) to

R is said to be an *h-relation*, if no node is a source or a sink to more than h requests. A 1-relation is a *permutation* (note that in this formulation, a permutation need not be complete). Suppose R is an h-relation. Then, R can be partitioned into $O(h)$ permutations. Thus, a realization of any permutation in ω wavelengths, provides a realization for R in $O(h \cdot \omega)$ wavelengths. We note that in the worst case, one cannot hope to do better for an h-relation than to realize it as $O(h)$ permutations, but in particular instances this may not be the best solution.

Let R be a communication pattern, and let ω_{opt} be the least number of wavelengths necessary for realizing R. A realization of R is said to be an *α-approximation*, if it uses ω wavelengths, with $\frac{\omega}{\omega_{opt}} = O(\alpha)$.

We say that an algorithm is *efficient* if it terminates in polynomial time.

The *edge expansion* of a graph G, denoted $\beta(G)$, is the minimum over all subsets of nodes that contain at most half the nodes in G, of the ratio of the number of edges that have exactly one endpoint in that subset to the size of the subset.

3 Bounded Degree Networks

In this section we present a near optimal realization of permutations in arbitrary bounded-degree networks. Consider a network $G = (V, E)$, and let β be the edge expansion of G. Let R be a permutation communication pattern for G. Consider the following multicommodity flow problem. For each $r_i = (s_i, t_i) \in R$ there is a unit demand for commodity i from source s_i to sink t_i. Assign unit capacity to each edge. An integer multi-flow for this problem provides paths for realizing R. The *congestion*, c, of the flow is the maximum number of paths using an edge. The *dilation*, d, of the flow is the maximum length of a path from source to sink. Leighton and Rao [LR88, Theorem 2], show how to obtain an integer flow for the above problem, with maximum congestion and dilation both $O(\log n/\beta)$. Use these paths for routing the requests. Consider the conflict graph C, with a node for each path and an edge between any two paths that share an edge. The graph C has degree at most cd, and can thus be colored with at most $cd + 1 = O(\log^2 n/\beta^2)$ colors in a greedy fashion. The colors determine the wavelengths. We thus obtain:

PROPOSITION 3.1. *For any bounded degree graph G, and permutation pattern R, the pattern R can deterministically and efficiently be realized using $\omega = O(\log^2 n/\beta^2)$ wavelengths.*

This almost matches the $\Omega(1/\beta^2)$ lower bound established in [RU94].

3.1 Local Path Selection

The above algorithm necessitates a full global control for determining the paths and the wavelengths assignment. With an additional $O(\log^2 n)$ factor a more local algorithm can be obtained.

Let $B = (V_B, E_B)$ be a wrapped butterfly graph on N nodes, with $3n \geq N \geq n$. Consider a mapping f of the nodes of B onto the nodes of G, with the property that at most 3 nodes of B map to any particular node of G. By a corollary from [LR88, Theorem 2], any such f can be efficiently extended to an embedding of the edges of B in G, such that the maximum dilation and congestion of the embedded paths are both $O(\log n/\beta)$. Determining the embedding is performed once, off-line, when setting up the network. Given such an embedding, future communication patterns are routed as if on B, using the fixed embedding. Routing on B (a butterfly) is performed using random two phase routing. This allows the source processor to determine the path locally. Once the path is determined, the wavelength is determined by picking a wavelength which is not already used along the path. We note that choosing the wavelength still requires some coordination between requests.

PROPOSITION 3.2. *With the above algorithm, with high probability, any permutation is realized using at most $O(\log^4 n/\beta^2)$ wavelengths, and all paths are determined locally at the source processors.*

Proof: Each path on the butterfly is of length $O(\log n)$ and with high probability meets at most $O(\log n)$ other paths. Each edge of the butterfly is mapped to a path in G of length $O(\log n/\beta)$. This path overlaps with the paths assigned to at most $O(\log n/\beta)$ other edges. Thus, in total, the length of a communication path assigned to a request is at most $O(\log^2 n/\beta)$ and meets with $O(\log^2 n/\beta)$

Our result for the hypercube does not explicitly use flow techniques. However, it continues the theme of relying on a solution to the maximum disjoint paths problem. We use the classical result of Beněs [Ben65, Wak68] that constructs a rearrangeable network, i.e. a network where every permutation can be routed along edge disjoint paths. We use a non-standard embedding of the Beněs network onto the hypercube, and a variation on the routing algorithm for the Beněs network, to get the $O(1)$ wavelengths result.

Previous related work. Optical routing in arbitrary networks was considered by Raghavan and Upfal [RU94]. They prove an existential $\Omega(\beta^{-2})$ lower bound on the number of wavelengths necessary to realize a permutation. For the upper bound, [RU94] present an algorithm which routes any permutation in $O\left(\log n^2/\log^2 \lambda\right)$ wavelengths with high probability, where λ is the second largest eigenvalue (in absolute value) of the transition matrix of the standard random walk on G. In terms of the edge expansion, the algorithm obtains a worst case performance of $O(\log^2 n/\beta^4)$, with $O(\log^2 n/\beta^2)$ obtained for some graphs. For degree d arrays, [RU94] present an algorithm with an $O(dn^{1/d}/\log n)$ worst case performance. The randomization of the algorithm results in worst case performance for all patterns. For trees, [RU94] present a 3/2 approximation.

Pankaj [Pan92] considers routing in hypercube based networks. For the hypercube, the shuffle exchange, and the deBruijn networks, he shows that routing can be achieved with $O(\log^2 n)$ wavelengths. Aggrawal, Bar-Noy, Coppersmith, Ramaswani, Schieber and Sudan [ABC+94], show that $O(\log n)$ wavelengths are sufficient for the routing in this class.

Lower bounds where considered in [BH92, BH93], [PS93], and [Pan92]. Pankaj [Pan92], proves a $\Omega(\log n)$ worst case lower bound on the number of wavelength necessary for routing a permutation in a bounded degree network. Barry and Humblet [BH92, BH93] give bounds for routing in passive (switchless) networks. An almost matching upper bound is presented in [ABC+94]. Peiris and Sasaki [PS93], consider bounds for elementary switches. The connection between packet routing and optical routing is discussed in [ABC+94].

A problem related to ours is that of integral mutlicommodity flow [LR88, GVY93]. (See also [KLPS90, Sch93] and references therein.) This problem has also been discussed in the context of on-line algorithms [GGK+93, AAP93, ABFR94, AGLR94].

Jon Kleinberg has informed us that he and Eva Tardos have obtained an improved bound of $O(\log n)$ to the approximation of the number of rounds required to realize a communication pattern on bounded dimension arrays [Kle94].

2 Terminology and Definitions

Let $G = (V, E)$ be a graph. A *communication request*, r, on G is a pair $r = (s, t) \in V \times V$. Node s is called the *source* of r, and t the *sink*. A *communication pattern* is a (multi-)set of requests. A request is *realized* by determining a *route*, from source to sink, and sending the message along the route, using a fixed *wavelength*. At any given time, distinct routes sharing the same edge must use different wavelengths. Let R be a communication pattern. We are interested in realizing all requests in R. Realizing R may be performed in *rounds*, with separate requests realized in different rounds. All rounds share the same bound on the number of wavelengths, but there are no further restriction on path selection and wavelengths assignment across rounds. In conclusion, a realization assigns to each request a triplet: (i) a route (ii) a wavelength, and (iii) a round. Denote the number of distinct wavelengths used in realizing R by ω, and the total number of rounds by T. We are interested in minimizing T and ω. Consider a realization which uses ω_1 wavelengths and T_1 rounds. Given such a realization, it is easy to construct another realization which uses at most $\omega_1 \cdot T_1$ wavelengths, and completes in one round. Conversely, it is also possible to construct a realization with one wavelength, and $\omega_1 \cdot T_1$ rounds. In fact, any realization with $\omega \cdot T = O(\omega_1 \cdot T_1)$, can easily be obtained. Thus, from now on, we discuss only the number of wavelengths necessary for one round, and omit explicit reference to T.

Consider a communication pattern R. A single node may participate in several requests. The set

RU94]. *Generalized switches*, on the other hand, are capable of switching incoming steams based on their wavelength [Che90, ABC+94, RU94]. Using acousto-optic filters, the switch splits the incoming signals to the different steams associated with the various wavelengths, and may direct them to separate outputs. In both cases, on any given link different messages must use different wavelengths. The switch may not route to the same link different messages which are using the same wavelength.

We are interested to allow communication between nodes of the network. A *communication request* is a *(source, destination)* pair of nodes. A *communication pattern* is a set of communication requests. A communication pattern is *realized* by assigning wavelengths to the messages, and setting the switches in accordance. Generally, the length of a path is an insignificant factor. The important measures are the number of wavelengths that the system must be able to handle, and the number of *rounds* necessary to complete the transfer of all messages. Algorithmically, the challenge is to devise a method which, given a communication pattern, finds a wavelength assignment to the messages and a setting for the switches, which minimizes these measures.

This work. In this paper we focus on the routing problem with generalized switches. Our results also give improved results for the elementary switches. We consider the case where all nodes of the network are occupied by sending and receiving terminals, as well as switches. In this work we present a near optimal realization algorithm for bounded degree networks. In addition we present improved realization algorithms for routing in bounded dimension arrays and for hypercubes. We obtain:

1. An algorithm which for any bounded degree graph G and permutation pattern R, gives a realization of R in one round using $O(\log^2 n/\beta^2)$ wavelengths, where β is the edge expansion of G, and n the number of nodes. This almost matches the existential lower bound of $\Omega(1/\beta^2)$ of [RU94].

2. For any bounded dimension array, any given number of wavelengths, and pattern R, an algorithm which realizes R using at most $O(\log n \log |R| \cdot T_{opt}(R))$ rounds, where $T_{opt}(R)$ is the minimum number of rounds necessary to realize R.

3. For the hypercube, an algorithm which realizes any permutation pattern using $O(1)$ wavelengths in one round.

(In Section 2 we show that in all results the role of rounds and wavelengths is interchangeable).

Note that the array result (item 2), gives a *per-instance* approximation. Previous results for arrays obtain a performance which is good only with regards to the worst-case pattern. No better performance is guaranteed for "easier" patterns. For example, a given pattern on the mesh may be realizable using single round, but the previous algorithms will still necessitate $O(\sqrt{n})$ rounds. Our algorithm, in contrast, approximates the number of rounds *per-instance*. For each pattern R the algorithm produces a realization which approximates the number of rounds necessary to realize the given pattern, to within an $O(\log |R| \log n)$ factor. The only previously known approximation algorithm is for trees, for which [RU94] give a 3/2 approximation algorithm.

Methods. Previously is was argued that flow techniques are ill-suited for tackling the optical routing problem [RU94]. We show to the contrary. The result for general networks is directly derived from results in integral multicommodity flow of [LR88].

The problem of approximating the number of wavelengths required to realize a communication pattern can be reduced to the problem of finding the maximum number of edge disjoint paths between source-destination pairs. That is, if one can approximate the number of requests in a pattern R that can be connected by edge disjoint paths, then one can also approximate the number of rounds required to realize R. The second approximation entails an additional $O(\log |R|)$ factor. Our per-instance approximation for arrays is derived from an $O(\log n)$ approximation algorithm for the maximum edge disjoint paths problem on arrays. This approximation algorithm may be of independent interest. The algorithm uses (integral) single commodity flow.

Improved Bounds for All Optical Routing

(Preliminary Version)

Yonatan Aumann* Yuval Rabani†

Abstract

We consider the problem of routing in networks employing all optical routing technology. In these networks, messages travel in optical form and switching is performed directly on the optical signal. By using different *wavelengths*, several messages may use the same edge concurrently, However, messages assigned the *same* wavelength must use *disjoint paths*, or else be routed at separate rounds. No buffering at intermediate nodes is available. Thus, routing in this setting entails assigning wavelengths, paths, and time slots for the different messages.

For arbitrary bounded degree networks, we show that any permutation can be routed efficiently in one round using at most $O(\log^2 n/\beta^2)$ wavelengths, where β is the edge expansion of the network. This improves a quadratic factor on previous results, and almost matches the $\Omega(1/\beta^2)$ existential lower bound. We consider two of the more popular architectures for parallel computers. For bounded dimension arrays we give the first *per-instance* approximation algorithm. Given a limited number of wavelengths and a set of messages to be routed, the algorithm approximates to within polylogarithmic factors the optimal number of rounds necessary to route all messages. Previous results for arrays give only worst–case performance. Finally, we show that on the hypercube any permutation can be routed using only a constant number of wavelengths. The previous known bound was $O(\log n)$.

1 Introduction

Motivation. Optical communication technology allows for very high data transmission rates, exceeding those of traditional electronical technology by several orders of magnitude. Optics is thus emerging as a key technology in state-of-the-art communication networks, and are expected to dominate many applications. The high data transmission rate is achieved, in part, using *wavelength division multiplexing (WDM)*. By WDM, multiple data streams may be transferred concurrently along the same fiber-optic, with the different streams assigned separate *wavelengths*. The corresponding input and output terminals are modulated to omit and receive the signal on the prescribed wavelength. In large scale networks switching must be allowed. In order to retain the high data transmission rate it is necessary that the switching is performed directly on the optical signal, without translation into electronic form (see [SM93]). Such optical switches are currently in development. Optical switches do not modulate the wavelengths of the signals passing through them. Rather, the switch directs the incoming wave to one or more of its outputs. Buffering is generally not available in these networks. Thus, optical communication introduces a new routing environment, with distinct characteristics. By nature, packet-routing algorithms are ill-designed for this setting. It is thus necessary to devise new algorithms, and algorithmic methodology, for optical network communication.

The Model. An optical network consists of *nodes*, interconnected by point-to-point fiber optic links. Each of the fiber-optic links supports a given number of *wavelengths*. The nodes may be occupied either by terminals, switches, or both. Terminals send and receive signals. Switches direct the input signals to one or more of the output links. Several types of optical switches exist (or are in development), An *elementary switch* is capable of directing the signals coming along each of its input links to one or more of the outputs. The elementary switch cannot, however, differentiate between the different wavelengths coming along the same link. Rather, the entire signal is directed to the same output(s) [BT91, KG92, ABC+94,

*MIT Laboratory for Computer Science, Cambridge, MA, 02138, aumann@theory.lcs.mit.edu. Supported by a Wolfson postdoctoral fellowship and DARPA contract N00014-92-J-1799.

†Work done while at MIT Laboratory for Computer Science, supported by ARPA/Army contract DABT63-93-C-0038. Present address: Department of Computer Science, University of Toronto, Toronto, Ontario M5S 1A4, rabani@cs.toronto.edu.