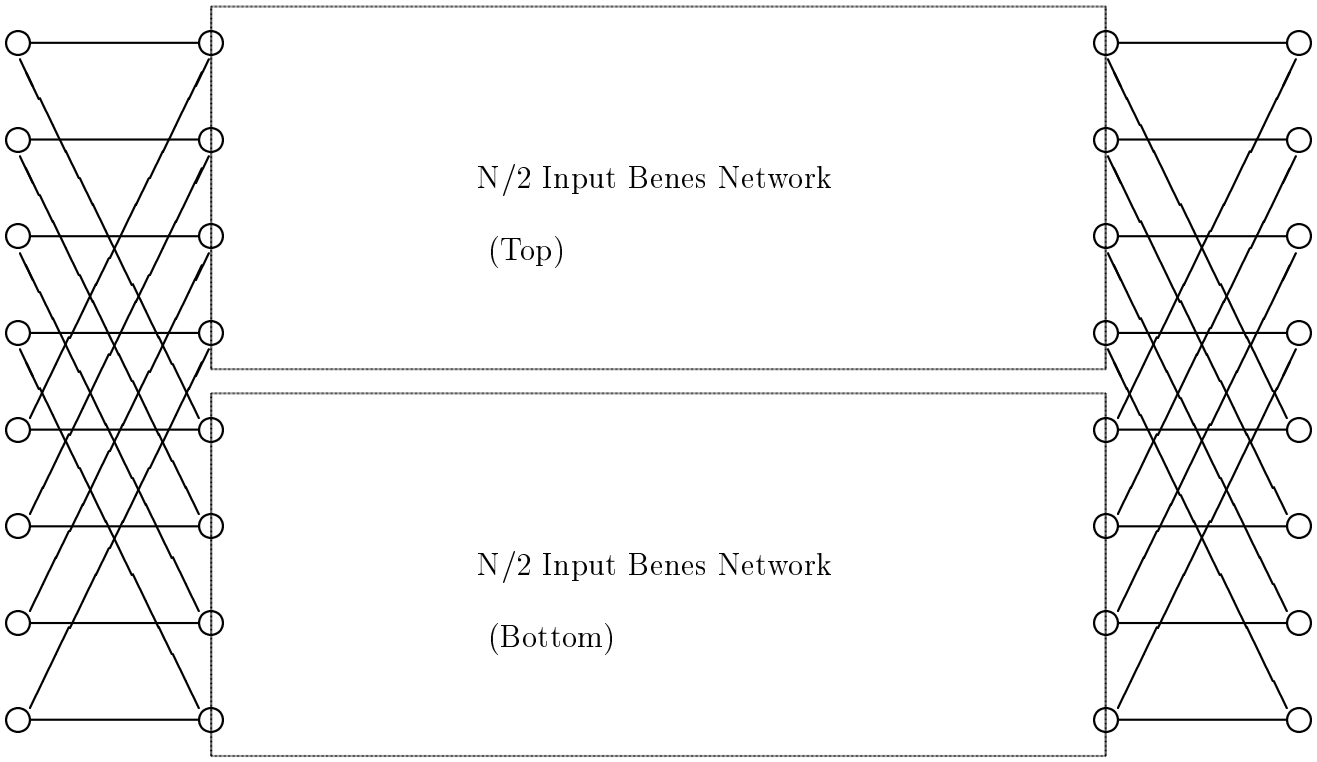

Figure 1: Structure of an N input Benes Network

In the permutation routing problem each processor i holds a single packet which it wishes to send to processor $\pi(i)$, where π is a permutation, i.e. $\pi(i) = \pi(j)$ iff $i = j$. We consider the offline version of the problem, i.e. we assume there is a central coordinating processor which knows π completely, and which can do lots of computation and decide how the paths for the packets should be selected. Although our original packet routing model disallowed such a central coordinator, the offline version is nevertheless interesting because (1) In some cases we might know π well in advance of running the algorithm, e.g. if our computations involve FFTs, then we might know that the bit-reverse permutation is needed. In that case it makes sense to spend time before hand planning the paths. (2) It is theoretically interesting to find out whether offline computation can help us do things faster than would be suggested by the deterministic oblivious routing bound.

Main result of this note: On a hypercube, for any permutation π, it is possible to assign paths to packets such that maximum congestion is 2, and further, the movement along the paths is normal. In fact all packets can be routed in time $2 \log N$, on an N node hypercube.

We will prove the result using a network invented by Benes. The connection to the hypercube will be immediate.

1 Benes Network

A Benes Network is simply a Butterfly network on N inputs (denoted B_N) connected "in series" following a reverse butterfly network on N inputs (denoted B'_N). In particular, the node in row i column $n = \log N$ of B'_N is merged with the node in row i column 0 of B_N. The network thus has N rows and $2n+1$ columns. Nodes in column 0 are called inputs, and nodes in column $2n$ the outputs. An alternative description of the network stresses its heirarchical structure: an N input Benes can be made by (i) taking two $N/2$ input Benes networks (one called top and the other bottom), (ii) connecting the corresponding inputs with 2 input Butterflies, (iii) connecting corresponding output with 2 input Butterflies as shown in Figure 1.

Theorem 1 (Benes) *Given any permutation π over $[0, N-1]$, it is possible to establish paths from input i to output $\pi(i)$ such that the paths for different i are node disjoint.*

Note that by coalescing every row into a single node we get a hypercube, and thus paths in the Benes give paths on the hypercube. The paths will traverse hypercube dimensions first from the most significant to the least significant, and then from the least significant to the most significant, and will clearly have congestion 2 at most.

Proof of Theorem 1: The proof is by induction over N. The base case, $N = 2$ is obviously true. Assume that the theorem is true for $N/2$ input Benes networks, i.e. we can establish node disjoint paths in an $N/2$ input Benes network for any permutation on $N/2$ inputs. Given this assumption we will show that the theorem holds for the N input Benes as well, completing the induction.

We first show how to ensure node disjointness in levels 1 and $2n-1$. The inductive hypothesis will then enable us to ensure node disjointness within the top and the bottom sub-networks.

Let $b(i) = \frac{N}{2} \oplus i$. Node disjointness in level 1 requires that if the path starting at input i is connected through the top Benes then the path starting at $b(i)$ must be connected through the bottom, and vice versa. Node disjointness in level $2n-1$ requires that if the path ending at output $\pi(i)$ (starting at i) is connected through the top, then the one ending at $b(\pi(i))$ (starting at $\pi^{-1}(b(\pi(i)))$) must be connected through the bottom.

These conditions can be stated as a coloring problem. We have an N vertex graph in which each vertex corresponds to an input. We have two colours, "top" and "bottom" with which to colour the inputs. Each node i in the graph has two edges. The first is to $b(i)$, this models the constraint that paths starting at i and $b(i)$ cant both go through the top, or both through the bottom. Likewise the second edge is to $\pi^{-1}(b(\pi(i)))$.

We now show that this graph is two-colourable. For this we must show that the graph has no odd cycles. First we classify edges into two kinds: edge (i, j) will be called a b-edge if $j = b(i)$; it will be called a ρ-edge if $j = \pi^{-1}(b(\pi(i)))$. Note that this classification is consistent: if $j = b(i)$, then $i = b(j)$, and also if $j = \pi^{-1}(b(\pi(i)))$, then $i = \pi^{-1}(b(\pi(j)))$. If an odd cycle exists in the graph, we would have to have 2 edges of the same kind in succession. But we know that every vertex is incident on only a single b-edge and a single ρ-edge. Thus odd cycles cannot exist and the graph is two colourable.

It is easy to find a two colouring. This is especially easy because the above graph has degree two – each node has only one b-edge and one ρ-edge. Thus the graph is made up of disjoint cycles. So we pick any vertex, assign colours alternately in the cycle to which it belongs. This is repeated for all cycles.

Thus we can find a colouring of the graph. The colouring gives us a way to assign the paths starting at each input to either the top or bottom Benes network. With this assignment, each Benes is assigned $N/2$ paths; further each of these paths is required to start at a unique input of the Benes and end at a unique output. Using the induction hypothesis we can ensure that the paths get assigned in a node disjoint manner even inside the top and bottom Benes, thus completing the proof. ■

Exercises

Adapt the above results to construct low congestion paths for permutation routing on other networks besides the Benes, i.e. hypercube, Butterfly, shuffle-exchange, multidimensional mesh.