There are various ways to define a hypercube. Here is one. The hypercube Q_k on k dimensions has 2^k nodes each labelled by a distinct k bit string, so that all the different k bit strings get used up. Then there is an edge from node u to node v if the labels of u and v differ in exactly 1 bit.

Another definition stresses the heirarchical structure. The hypercube Q_0 is simply a single vertex. The hypercube Q_k is obtained by taking two copies of Q_{k-1} and joining corresponding nodes in the two copies. It should be clear that this is the same as the first definition.

Yet another definition uses the notion of graph products.

1 Product Graphs

The product $G \Box H$ of graphs G, H is defined to have the vertex set $V(G \Box H) =$ the cartesian product $V(G) \times V(H)$, and there is an edge from (g, h) to (g', h') in $G \Box H$ iff (a) $g = g'$ and h, h' are neighbours in H, or (b) $h = h'$ and g, g' are neighbours in G.

Clearly, $|V(G \Box H)| = |V(G)| \cdot |V(H)|$.

Here is a way to visualize a $G \Box H$. Consider $|V(G)|$ rows, each containing $|V(H)|$ vertices. Label the rows by labels of vertices $V(G)$, and the columns by Vertices $V(H)$. On the vertices in each row, put down a copy of H, with h in row labelled h. Similarly, in each column, put down a copy of G. This then is the product $G \Box H$.

Here is an example. Let $G = P_2$, the path on 2 vertices, just an edge. Then $P_2 \Box P_2$ is simply a square. $(P_2 \Box P_2) \Box P_2$ is a cube, in fact Q_3, and so on.

It is customary to say that g is the first coordinate of vertex (g, h) of $G \Box H$, and h the second.

Lemma 1 *$\Box$ is commutative and associative.*

Proof: Need to show that $G \Box H$ is isomorphic to $H \Box G$. Recall that X is isomorphic to Y if there exists a bijection f from $V(X)$ to $V(Y)$ s.t. (x, x') is an edge in X iff $(f(x), f(x'))$ is an edge in Y. The vertices in $G \Box H$ are

(g, h) and those in $H \square G$ are (h, g) where $g \in V(G), h \in V(H)$. We use $f((g, h)) = (h, g)$.
There is an edge from (g, h) to (g', h') in $G \square H$
$\Leftrightarrow$ either (a) $g = g'$ and h, h' are neighbours in H, or (b) $h = h'$ and g, g' are neighbours in G.
$\Leftrightarrow$ There is an edge from (h, g) to (h', g') in $H \square G$.
$\Leftrightarrow$ There is an edge from $f((g, h)) = (h, g)$ to $f((g', h')) = (h', g')$ in $H \square G$.
Thus f is the required isomorphism. Associativitity is similar. $\blacksquare$

Let $D(G)$ denote the diameter of G, and $d(u, v)$ the length of the shortest path from u to v.

Lemma 2 $D(G \square H) = D(G) + D(H)$

Proof: Say an edge $((g_1, h_1), (g_2, h_2))$ is of type G if $h_1 = h_2$, and of type H if $g_1 = g_2$.

Consider any pair of vertices (g, h) and (g', h') in $G \square H$. We will show that there must exist a shortest path between them in which all G edges occur before all H edges. The Lemma will then follow.

Consider a shortest path from (g, h) to (g', h') in which an H edge $e_i = ((a, b), (a, c))$ is followed by a G edge $e_{i+1} = ((a, c), (d, c))$. Clearly, $(b, c) \in E(H)$, and $(a, d) \in E(G)$. Thus, $G \square H$ must also contain the edges $e'_i = ((a, b), (d, b))$, and $e'_{i+1} = ((d, b), (d, c))$. If we replace e_i, e_{i+1} by e'_i, e'_{i+1}, P continues to remain a shortest path. But in this operation we have moved one H edge beyond a G edge. Continuing in this way, we can ensure that all G edges, if any, precede all H edges, if any.

Let P be one such shortest path, i.e. it is made of segments P_1 and P_2 consisting respectively of entirely G and H edges. Clearly P_1 must end at (g', h), and must stay within a column as per the above visualization. Thus the length of P_1 is exactly the distance from g to g' in G. Similarly, the length of P_2 is at most the distance from h to h' in H. Thus the length of P is at most $D(G) + D(H)$, with equality arising when the distance from g to g' in G is $D(G)$, and that from h to h' in H is $D(H)$. $\blacksquare$

Paths which first travel through the column and then through the row (or vice versa), "correcting coodinate in succession", will be called a *canonical paths.*

1.1 Multiple products

If $G = G_1 \square G_2 \square \cdot \square G_k$, then its vertices will have k coordinates, the ith coordinate being a vertex from G_i. Vertices of G differing in the ith coordinate will be said to differ in the i dimension. The notion of canonical paths again naturally generalizes. Also $D(G) = \sum_i D(G_i)$.

1.2 Implications for the hypercube

Clearly the hypercube Q_k has diameter k, and to go from one vertex to another vertex, we can use a canonical path, correcting the dimensions in say, a least significant to most significant order.

Note that because of associativity of $\square$, a hypercube on $k = k_1 + k_2$ dimensions can be viewed as a product of hypercubes on k_1 and k_2 dimensions.

2 Symmetries of the hypercube

It may be obvious that all vertices of a hypercube are symmetrical. This is formalized using the notion of a *automorphism*. An automorphism is simply an isomorphism from a graph to itself.

It will be convenient to think of vertices as k bit strings. It will also be useful to have the notation $u \oplus v$ which denotes the bit-wise exclusive or of u and v. Note that $\oplus$ is associative and commutative, and that the all 0s string is the identity and each element is its own inverse. Further, suppose u, v are neighbours. Thus they must differ in just one bit. Say it is the ith least significant bit. Then we may write $u = v \oplus 2^i$.

We will show that there is an automorphism f on Q_k that maps any vertex u to any vertex v. Consider $f(x) = x \oplus w$ where $w = (u \oplus v)$.

1. $f(x) = f(y) \Rightarrow x \oplus w = y \oplus w \Rightarrow x \oplus w \oplus w = y \oplus w \oplus w \Rightarrow x = y$. Thus f is a bijection.

2. $f(u) = u \oplus w = u \oplus u \oplus v = v$. Thus f maps u to v as required.

3. Consider a vertex x and its neighbour $x \oplus 2^i$. Then we have $f(x \oplus 2^i) = x \oplus 2^i \oplus w = x \oplus w \oplus 2^i = f(x) \oplus 2^i$. Thus $f(x \oplus 2^i)$ and $f(x)$ are also neighbours across dimension i.

Thus f is the required automorphism.

Exercises

1. Show that there is an automorphism of the hypercube that maps any given edge to any given edge.

2. In the graph colouring problem, we are required to assign a colour to each vertex such that adjacent vertices in the graph get different colours. The minimum number of colours which can be used to colour a graph G satisfying the above requirement is said to be its chromatic number, denoted $\omega(G)$. Show that $\omega(G \square H) = \max(\omega(G), \omega(H))$. Observe this for the hypercube, and produce a colouring for Q_3 using $\omega(Q_3)$ colours. Prove it in general. Hint: Start by colouring one row of $G \square H$ as per the optimal colouring of H. Derive the colours for the next row in some systematic manner.

3. In the *permutation routing problem* each processor sends a message to a unique processor. Consider this on the hypercube Q_3 and suppose that the canonical path is used: correct bits from lsb to msb. Suppose processor i sends a message to $i + 5 \bmod 8$. Draw the paths. The maximum number of messages travelling on each link in any single direction is said to be its congestion. Observe that the congestion is at most 1. This phenomenon persists for other hypercubes and all cyclic shift like permutations, but you are not expected to prove this, yet.

4. Consider a $P_n \square P_n \square P_n$, also called an $n \times n \times n$ 3 dimensional processor array. Consider the permutation routing problem and suppose the canonical path is used: first correct the x coordinate, then the y coordinate, then the z coordinate. Show that there exists a way to assign destinations such that some link gets $\theta(n^2)$ congestion for some direction. The congestion is important because it is a lower bound on the time to finish message transmission – since only one message can go through a link at any step (assume messages are 1 word long). Since the network diameter is $3n-3$ we might expect that message movement should finish in $O(n)$ time for all permutation problems. However, this exercise shows that $\Omega(n^2)$ time might well be needed.