How to Implement Giga-Networks

A Problem Paper

Equipment – cables and couplers – in the 10 gigabaud range have now been commercially available for a couple of years. Existing net implementations are, as far as I know, either "stars" (having a centre like a telephone exchange) or "rings" (joining the stations into a cycle). Neither of those solutions is very satisfactory in terms of reliability and efficiency. A star centre must be some huge powerful equipment, and if it goes down, the whole net goes down. In a ring, throughput capacity is rather poorly utilized (under any really occuring distribution of transfer claims), delays can happen to be long, and there is again a non-trivial problem of salvaging the rest of the net in case a station fails.

For nets in the megabauds range, many a technique has been developped to overcome such inconveniences; those techniques are based on allowing for collisions, collision detection and collision-resolving algorithms. But we better forget all the old ideas for at least two reasons: (a) present high-speed equipment is glass fibre, where reading is destructive (not absolutely, but it can only be shared among a few), so not everybody can listen to the same transmission; (b) a technique involving end-to-end signal propagation time for transmission approval seems not even desirable because not only transfer is faster but distances can be much greater, so end-to-end time may be utterly large as compared with transmission times.

Network Graphs

Vertices: the stations. Edges: the (immediate) transfer links between stations; links are assumed capable of transfer both ways (which is not a compelling assumption, the opposite being equally senseful and workable; nonetheless, this is today's lesson). - All graphs below will be invariant under permutation of vertices: homogeneous nets will be considered.

v: the number of vertices.

e: the number of edges.

d: the diameter, that is, the maximum distance, distance being the minimum path length between a pair of vertices.

a: the average distance. (It might be argued that the zero distance between a vertex and itself is not to be included in the count since a station does not communicate with itself. Well, then multiply by $\frac{v}{v-1}$.)

In a ring, e equals v, d is $\lfloor \frac{v}{2} \rfloor$, a is $\frac{\lceil \frac{v}{2} \rceil \times \lfloor \frac{v}{2} \rfloor}{v}$.

On the other hand, ~~joining~~ when every vertex pair is joined by an edge, that is, in the complete graph, e is $\frac{v(v-1)}{2}$, d is 1, and everyone can talk to everyone simultaneously; an ideal yet expensive setup.

The point is that there are reasonable arrangements in between: a comparatively small number of edges can drastically reduce d and a, and improve upon possibilities of simultaneous usage and of bypassing possibly damaged parts of the net.

As a first idea consider the n-dimensional cube. It has $v = 2^n$, $e = n2^{n-1} = \frac{\log_2 v}{2} v$, $d = n = \log_2 v$, $a = \frac{n}{2} = \frac{\log_2 v}{2}$. Thus, compared with the ring, a logarithmic "cabling extra-cost rate" $\left(\frac{\log_2 v}{2}\right)$ reduces distances to logarithmic size. (In the complete graph, the extra-cost rate is $\frac{v-1}{2}$ to reach $d=1$.) To prevent any pair of stations from communicating, at least n stations must be fully occupied (maintaining at least $n-1$ simultaneous communications each) or defective.

Now a cube is an incredible cobweb. It is certainly not a welcome architecture for cabling some scattered network. We would like graphs that can be "bundled": their edges can be ordered along a cycle with no (or at least not much) looping and zigzagging, so they can form a single cable to run around.

Consider, e.g., the following graph on $v = 2^n$ vertices. Number the vertices $0, \dots, v-1$. Now connect the vertices by edges in (cyclical) order. (That is, make a ring.) Next connect all odd vertices in order into a ring, and similarly all even vertices into another ring. Then build four rings of the residue classes mod 4. Continue so over powers of two, stopping at some 2^k $(k \leq n-1)$. – This is a bundled graph by construction. Call it the "(n,k) bundle".

Each station has k neighbours in either direction. The cable consists of $2^{k+1}-1$ fibres. $e = (k+1)2^n$. $d = 2^{n-1-k} + \lfloor \frac{k}{2} \rfloor$. $a = 2^{n-k-2} + \frac{k}{3} - \frac{1}{18} + \frac{(-1)^k}{9 \cdot 2^{k+1}}$ (awful).

(Case $k = n-1$ has twice the number of edges of the cube but has a better d and a. Lower values of k are asymptotically inferior to the cube; still, they offer very acceptable architectures for networks of realistic sizes.)

Example. Take $n = 8$, $v = 2^n = 256$.

Graph	e	d	a
Complete	32640	1	0.99609375
Cube	1024	8	4
Bundle –			
$k = 0$ (ring)	256	128	64
$k = 1$	512	64	32.25
$k = 2$	768	33	16.625
$k = 3$	1024	17	8.9375
$k = 4$	1280	10	5.28125
$k = 5$	1536	6	3.375
$k = 6$	1792	5	2.9453125
$k = 7$	2048	4	2.777348$^+$

Note that $k = 3$ and 4 yield fair mediocre networks with 15- and 31-fibre cables, resp., sharply reduced distances, and a lot of possible parallelities and bypasses, while buffering for transit packets is not very expensive (transit route multiplicity being not higher than the number of neighbours, k).

No claim is made that the above graphs are in any respect optimal. In particular, a similar but triadic construction ought to be examined. – Could we find optimal graphs? How to define optimality?

Routing Strategies

As the path ~~between~~ from a station to another is no longer unique, we have to choose. It is conceivable to have some arbitrating instance or negotiating procedure that assigns a path to a subsequent communication. This might be senseful in particular cases; it is unpleasant as a general method: with such speeds, time and transfers needed for the decision would possibly be more than enough to do the communication itself. For similar reasons, not even purely local decision techniques are desirable if based on full information concerning network state. What we need is decision procedures based on purely local, or not very remote, or very-seldom-very-remote, information.

What we want to optimize is, first of all, network throughput and transmission delay, more accurately, their expectation values and maybe worst cases. These values and optima trivially exist for each given finite sequence of claims and for any finite set of such sequences. I guess their existence can be proved under more general assumptions. It would be preferable to have decision procedures that are not very sensitive of claim distribution. (In collision-resolving networks, the best access strategies are asymptotically insensitive.)

The intuitively best procedure I could find to date (applicable to all of the above nets) is the following. To make the next step in building a path to an addressee, select from your immediate neighbours that are nearer to the addressee than you are and are able to further or to accept, as the case is, the neighbour ~~which~~ (one of them) ~~that~~ which is nearest to the addressee. If none, the failed. – Distances are easily computed from station numbers. The state information used is obviously fairly local.

This procedure has not yet been duly analyzed. (I feel an exact treatment is possible, like with collision-resolving access.)

Remarks.

– We resign even in cases where delivery is possible. Instead, we may drop the qualification "nearer to the addressee", replacing it by "not yet in the path" or at least using a step counter. This complicates handling. Since early resignation is likely to occur in heavy-load periods, maybe it is even advantageous. Hard to tell, for lack of formulae.

– We might buffer and wait for better times under certain conditions, e.g., when momentarily no "best possible" step or "nearer to the addressee" step is available. This can be combined with a time-out.

– If a sequence of transmissions or a period of communication is planned, does it pay off to preserve, or to try to preserve, a path once established? Most probably not.

– Defective stations lengthen distances; that's all right. Missing (non-existent) stations do the same. So if the number of existing stations is not an exact power of two, the graph should, and presumably could, be adjusted to their number. How?