A DONKEY BRIDGE (PONS ASINORUM)

June 1989

Opening Remarks

Suppose we have a network with flat addresses (that is, global addresses with no discernible structure to them), and we tolerate (maybe packet losses but) no duplications and no order changes (inversions) in the sequence of packets received by a particular station from a particular station. (I am not suggesting that this is the right way to do things. This is a way to do things.)

We might want to connect, on the lowest level possible, networks having this property into networks having this property. (Which, again, may make sense.) We might want to do that in an efficient way (whatever that means).

After (and before) deadly debates, one-and-a-half techniques emerged to be considered as standard solutions for IEEE (and leading implementors): the "spanning tree" and the "source routing".

"Spanning tree" is ripe and ready. As long as the system of bridges itself is a tree, or almost, there is no problem. When not, packet routing starts to be increasingly pessimal, duly counterbalanced by hardware standing idle in the wings while hoping for a breakdown of concurrent equipment in a very Miltonian spirit ("they also serve who only stand and wait"). Nevertheless, it works, requiring, in the particular stations, no knowledge (and no additional coding) about the configuration and reconfiguration of the network they are part of.

"Source routing", on the other hand, is ripe and unready. It enables optimal routing (according to some criteria), and provides information as to "how to". The lesser problem is that the method proposed (by IBM people) to get at that information implies a combinatorial explosion; this could have been avoided (albeit not in the proposed representation). Still, even destultified, it remains a strain on a net bigger than moderately-sized, and it requires, on the part of the particular stations, awareness of network state (with appropriate extra coding added), all that in a time scale which is really demanding if a customer wants real, as contrasted with promised, network efficiency.

We could dwell upon comparative merits till exhausted but, fortunately, we are not forced to decide upon. The following two statements are true. (i) There are methods of automatic shortest-path routing (where "shortest path" can be defined rather arbitrarily), with practically no more administrative overhead than in the "spanning tree" and no Miltonian hardware, requiring no awareness of anything on the part of the involved stations. (ii) Such methods, and strict "spanning tree", and strict "source routing" (and anything in between) can be implemented, if reasonably coded, in a way that the same physical bridges can handle all of them. Reasonable coding means simply that different types of routing demands can be formally distinguished.

An Algorithm

C o n f i g u r a t i o n

A **network** is built out of **stations** grouped into **local nets** that are joined by **bridges** in an arbitrary manner (it may even fall apart). The bridges themselves can (but needn't) be regarded as stations on the local nets they join.

Two-headed bridges will be assumed. Bridges with more heads may be fitted into the network. A direct working description would require obvious, but tedious, re-wording; the best way to think of them is perhaps in terms of as many bridges as they have pairs of heads, and then economize upon that concept.

The network **graph** is the graph with vertices the local nets and edges the bridges. The network **diameter**, d, is the diameter of that graph.

The network is permitted to change (be reconfigured or suffer failures). Its actual configuration is called its **state**.

H e l l o S i g n a l s

For every station, a **Hello signal** is issued every T secs. (T might be in the order of tens.) On behalf of the stations, the signals are issued by the bridges attached to their local net: for stations on one side, signals go to the other side. (With the bridges giving the signals, no extra coding is needed in the stations themselves. - The way bridges can detect the existence of a station is different in different local net implementations but, as far as I know, there **is** a method everywhere.)

The signal carries the **station address**, a **serial number** and a **hop count**. The serial number is incremented station-wise for each signal issued (cyclically of course; a byte is more than sufficient). The hop count is initially set 1, and incremented any time the signal is forwarded.

As a fact of matter, no separate station signals are needed; what is needed is separate signals for local nets, with common serial number and hop count, and enumerating the station addresses concerned.

Bridges are free to combine and recombine any number of signals into packets, whether or not for the same local net. (This reduces network load.)

Signal packets shall contain the address of the immediate sender (the issuing or forwarding bridge).

S i g n a l   F o r w a r d i n g

The following is done for signals of each particular station (or local net).

Whenever a bridge receives a signal with a new serial number (from either side), it neglects signals with an earlier number and forwards (with incremented hop count) the new signal and possible subsequent signals with the same number toward the opposite side provided their (initial) hop count is smaller than that of any signal received from either side.

Forwarding starts only U secs after the arrival of the first signal, with the "best" signal at **that** time. (This is to sieve out spurious signals in order to reduce network load. If U is well chosen, unnecessary forwarding will never occur, or at least become a rarity. - For the right choice of U, notice that it is to absorb time differences in signal-propagation along different shortest paths. Those differences arise from the "scheduling uncertainty": the discrepancy between the moment a bridge **should** transmit and **does** transmit. This of course is a stochastic quantity, but under present hardware conditions it may be expected to be in the milliseconds order. A shortest path is at most d-long. On the other hand, U ought to be significantly shorter than T; to have it short is, moreover, an end in itself. So U might be in the order of 0.5. - With a good U, no serial number is really needed. But it is cheap, and may prove useful in less homogeneous networks, see below.)

Maximum Hello-propagation time is dU secs. (Bridges can find out d from the accumulated hop counts except at configuration / reconfiguration; in fact, they don't need to know.)

C o m p e t e n c e

Competence of a bridge concerning a particular station is the decision, made by the bridge, whether to forward data packets addressed to that station and, if so, in which direction.

To decide upon, the signals (with the last serial number) having minimum hop count are evaluated. The initial issuer of a signal assumes hop count zero on the station's side.

(a) If the minima on the two sides are equal, the bridge is incompetent. (It joins two local nets having the same distance to the station, that is, it is "perpendicular" to the paths.)

(b1) If there is no signal on one side, the bridge is competent for data packets coming from that side, to be forwarded to the other side. (The bridge is the only candidate.)

(b2) If the larger minimum is larger than the smaller plus 1, competence is in direction from the larger towards the smaller. (The larger minimum was carried by a spurious signal).

(c) The larger minimum equals the smaller plus 1. (There are several candidates.) Then the bridge computes the exclusive-or of the station address with its own address and the address(es) of the bridge(s) the larger minimum came from. It is competent toward the smaller minimum if and only if its own XOR is the smallest. (This singles out a bridge, and that in a possibly diversified way.)

(Herein, (a) is the Aesopic case, (b) is the Balaam case, (c) is the Buridan case.)

Competence is **valid** in case it has been unchanged for a 2T (or 3T) period (or for 2 or 3 serial numbers).

If there is a change, that is, competence turns out different from the previous setting or no signal has arrived for 2T (or 3T), then competence is invalidated (reconfiguration is to take place). Before a possible revalidation, at least U times hop count secs must elapse. Data packets possibly waiting for transmission may either be immediately discarded or held up for a short period, hoping for eventual revalidation.

R o u t i n g

Whenever a data packet is addressed to a station, it is forwarded by the competent bridges. If none, the packet gets lost.

Broadcasts, narrowcasts, etc. (if needed) are easily implemented. (A nice idea is to forward a broadcast along reversed competences.)

V a r i a n t s

Hello signals may be doubled with a span of, say, 0.1U secs in between. (So a false belief as to station disappearance emerges only in case of four consecutive signals lost.)

Extra signals may be issued in case of a change.

With station signals grouped according to local net, station disappearance can be earlier detected. Likewise for bridge disappearance, if bridges are included as stations in the signals.

Station addresses need not be included in **all** signals. (They are the bulk, so rarifying them eases network load.)

For a more sophisticated routing in inhomogeneous networks (taking into account significantly different speeds or loads, e.g., "long" bridges [called "remote", for unknown reasons]), "costs" can replace the hop counts in the routing decisions. Forwarding and competence rules must then be reformulated in the obvious manner: the point is that the incoming values on either side have to be compared with the incremented-by-cost values incoming on the opposed side, which is harder to explain but not harder to do. An increased U (and perhaps T) might be required.

Dynamically changing costs are implementable if desired; too frequent changes, however, are prone to deteriorate performance (notwithstanding widespread delusions to the contrary).

The convention wishing for "unconditional forwarding when no direction is known" (as suggested by the "spanning tree" school) could be adapted and adopted to replace the "losing" rule; but this looks unnecessary here and, thus, undesirable.

Closing Remarks

Such routing might be dubbed a "spanning thicket". More accurately, it produces a set of spanning trees, one for each local net to reach (the stations in) it.

It is obvious that the routing (if any) is unique and there is no duplication and no order inversion of packets between the same pair of stations. (Whereas, of course, the route in the opposite direction is not necessarily the same route reversed.) The routing marks out a shortest path between any pair; there are no spare bridges. The same states always result in the same routings. A change affects only the routes involved.

Clearly there is no explosion since (with a good U) each signal is forwarded but once (at most) by each bridge; 10-sec signals and 10000 stations make one signal per millisec to handle on the average; the number of packets to be sent can be much less if signals are lumped.

Data packet forwarding takes time comparable with "spanning tree". Bridge memory requirements, too, are similar.