Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.02

Solutions to Chapter 18

Updated April 22, 2012

---

1. To avoid consuming an excessive amount of network bandwidth, a node should re-broadcast any given link-state advertisement (LSA) only once on each link. The sequence number helps implement this rule. When a node receives an LSA from any origin node with a sequence number larger than the previous one from that origin, then the node re-broadcasts that LSA as part of the flooding protocol. Otherwise, that LSA is not re-broadcast.

2. The hop limit field is used as part of the forwarding process at a switch. If a packet has already been forwarded H times, where H is the original value of the field set at the source, then the packet is dropped. The reason is that packets that end up getting stuck in routing loops may otherwise remain in the network forever, just wasting bandwidth.

3. Suppose the minimum-cost path between nodes $n_0(=S)$ and $n_h(=D)$ in a network has h hops, $n_0n_1n_2\ldots n_h$. In the worst case, each node n_i could advertise the best path for destination D to node n_{i-1} immediately after n_{i-1} finishes its integration step, because the nodes implement their timers independently and operate asynchronously. This, in turn, means that it takes h hops before knowledge of the *best* path reaches node S. Note that S will hear information about *some* path to D in h' hops, where h' is the length of the path with the minimum number of hops between S and D. But that path may not be the minimum-cost path, in general.

4. Suppose the node hearing HELLO packets decides that a link has failed if it fails to receive k consecutive HELLO packets from its neighbor. The probability that k consecutive HELLO packets get lost and the link is still alive is equal to p^k where p is the packet loss probability on the link. Hence, we want $p^k \leq 0.001$, and in our problem, $p = 0.1$. So $k \geq 3$. Because a link may fail immediately after a HELLO is sent, it can take as long as $k+1$ HELLO intervals to determine a failed link. In this problem, that works out to 4×10 seconds $= 40$ seconds.

5. See PSet.

6. (a) The key insight to observe is that each introduces a delay of at least $T/2$ because it takes that long between the integration and advertisement steps. Given this fact, the answer would be
$$(\frac{N}{2} - 1) \cdot (\frac{T}{2} + \delta).$$
However, there is a small "fence-post error" in this argument. As stated in the problem, the nodes labeled 1 and N update their routing tables at time t to include node $N+1$. In the best case, these two nodes could both immediately send out advertisements, and nodes 2 and $N-1$ could run their integration steps immediately after receiving these advertisements. Because of that, the delay of $T/2$ only starts applying to the other nodes in the network. Hence, the answer is
$$(\frac{N}{2} - 2) \cdot \frac{T}{2} + (\frac{N}{2} - 1) \cdot \delta.$$

(b) The key insight is that it takes in the worst case $T+T/2$ seconds per hop because each node may get the information about the new node **just after** it completes the previous integration step. So it has to wait T for the next integration, and then another $T/2$ to advertise.

Hence, the correct answer is

$$(\frac{N}{2} - 1) \cdot (\frac{3T}{2} + \delta).$$

7. (a) All except SecondMinCost will work correctly.

   To see why SecondMinCost will not work: consider the triangle topology with 3 nodes A, B, D, and equal cost on all the links. The second route at A to D is via B, and the second best route at B to D is via A, resulting in a routing loop.

   (b) To implement MinCostSquared, your integrate routine should add the square of the link cost (instead of just the link cost) to any route costs advertised over that link.

8. (a) Node S: 10 seconds, Node A: 110 seconds, Node B: 110 seconds; Node C: 210 seconds.

   Here's why. At time $t = 10$, D advertises to S, A, and C. They integrate this advertisement into their routing tables, so that cost$(S, D) = 2$, cost$(A, D) = 2$, cost$(C, D) = 7$. Note that only S's route is correct.

   In the worst case, we wait 100 seconds for the next round of advertisements. So at time $t = 110$, S, A, and C all advertise about D, and everyone integrates. Now cost$(A, D) = 4$ (via S), cost$(B, D) = 4$ (via S), and cost$(C, D) = 7$ still. A and B's routes are correct; C's is not. Finally, after 100 more seconds, another round of advertisements is sent. In particular, C hears about B's route to D, and updates cost$(C, D) = 5$ (via B).

   (b) The link $S \rightarrow D$ carries the most traffic. Every node's best route to D is via S.

   (c) 7. Here's why. S needs to advertise a high enough cost such that everyone's path to D via S will no longer be the best path. In particular, since B's cost to D *without* going through S is the highest (8), S must advertise a cost so that linkcost(B, S) + advertisedcost$(S, D) > 8$. Hence, S advertises a cost of 7.

9. See PSet.