In-Class Activity #6

Vancouver Summer Program 2017

Algorithms and the Internet

---

1. In Internet routing, there are delays on lines but also, more significantly, delays at routers. This motivates the following problem. Suppose we have a computer network consisting of R routers $r_1, \ldots, r_R$. Suppose that router r_i has estimated delay $c(r_i) > 0$, and, if two routers r_i, r_j are connected by a line, then the line has delay $\ell(r_i, r_j) > 0$. Now define the cost of a path between two routers to be the sum of its line delays plus the delays of all the routers on the path (including the endpoints). Give an efficient algorithm for the following problem.

   Input: (1) The network information as a bit string S of length R^2: Each R-bit substring (starting from the first position in S) represents routers $r_1, \ldots, r_R$, and a bit in the same location in two different R-bit substrings is 1 iff there is a link between the routers corresponding to that location. (2) Positive router delay estimates $c(r_i)$ and positive line delay estimates $\ell(r_i, r_j)$. (3) A starting router s.
   Output: An array $\texttt{cost}[\cdot]$ such that for every router r, $\texttt{cost}[r]$ is the least delay of any path from s to r, under the definition above.

2. (Tolerant colourings.) Consider a graph where every vertex has degree at most 25. We would like to colour the graph using two colours, *red* and *black*, such that no vertex has more than 12 adjacent vertices of the same colour. To achieve this, your friend Mercator suggests the following scheme: For each vertex in the graph, toss a coin and if the coin toss results in a head then colour that vertex red otherwise colour it black. Label each vertex that has more than 12 adjacent vertices of the same colour with a *FixMe* label. Pick any vertex that has a FixMe label and call this vertex v. (If no such vertex exists then stop.) If v is coloured red then recolour it black; if v is coloured black then recolour it red. Update all the FixMe labels. Repeat the recolouring process until there is no vertex with a FixMe label.

   Show not only that Mercator's algorithm will stop but that it will do so in no more than $O(m)$ colouring/recolouring and label update *iterations* where m is the number of edges in the graph.

   (It is possible to show that the entire algorithm can run in $O(m + n)$ time where n is the number of vertices but you do not have to include such a proof in your solution.)