Link reversal algorithms

Implementation

Basic idea of Link Reversal algorithms

Question 6 Let D be a node of a directed graph G. Suppose G is acyclic.

If G is D-oriented, then for every node in G there exists a directed path originating at this node and terminating at D (definition). If D was not a sink then it would mean a cycle exist between D and its outgoing neighbour, so by contradiction D is a sink. There cannot be any other sinks in G as it would contradict the D-oriented hypothesis. Therefore D is the sole sink of G.

Suppose D is the sole sink of G. Then starting from every other node in G there is always an outgoing neighbour. If we choose one of these neighbour, and repeat the process, we are always guaranteed to never visit a node we already visited because G is acyclic. As there is only a finite number of nodes in G and the number of unvisited nodes decrease by 1 at each iteration, we are guaranteed to eventually reach D, and have by this process found a directed path to D.

Explain how the above equivalence naturally leads to the link reversal algorithms for constructing a D-orientation.

Using the preceding equivalence we can ensure that a graph that is D-oriented, if and only if it is acyclic and D is the sole sink of the graph. This is the basic idea of link reversal algorithms.

Zero-delayed vs. finite-delayed executions

Question 7 Both u and v are sinks (different from D). Therefore there cannot be an edge connecting these two nodes. Says u achieves a link reversal in LR. Then by the preceding it means that v is still a sink in G_u and may still a link reversal. By symmetry u may also achieve a link reversal in G_v. In both cases, the resulting directed graphs are equal.

Correctness Proof for Full Reversal

Question 8 Let u be a neighbour of D in $\overline{G}$. If D is an out-neighbour of u then u will not perform any link reversal as D will never reverse its links, so u is guaranteed to never become a sink. If D is an in-neighbour of u then u may perform one link reversal before D becomes an in-neighbour. So in either cases u will perform at most one link reversal.

Question 9 Let u, v a pair of neighboring agents in $\overline{G}$, both different from D. Say u performs its k-th link reversal. Then v becomes an out-going neighbour for u. In order for u to perform its $k+1$-th link reversal, it must become a sink. Therefore v must become an in-going neighbour for u, and for that, it must perform a link reversal. So v makes at least one link reversal between $\mathrm{LR_k(u)}$ and $\mathrm{LR_{k+1}(u)}$.

Question 10 Let's suppose by contradiction that one execution of the algorithm is infinite. Then it means that at least one node u performs an infinite number of link reversals. Using *question 9*, it means that for every neighbour v of u in $\overline{G}$, v must perform at least one link reversal between $\mathrm{LR_k(u)}$ and $\mathrm{LR_{k+1}(u)}$, for every natural k. So v must also perform an infinite number of link reversals and by induction it means that every nodes that are not D must perform an infinite number of link reversals. This enters in contradiction with the result of *question 8*, therefore no execution can be infinite.

Question 11 In case of the Full Reversal algorithm, let G_f be the final directed graph. Because it is the final state, it means that D is the sole sink of G_f. Let's prove that G_f is acyclic by induction. G_0 is acyclic. At every link reversal happening in the execution, no cycle are introduced: if a node u perform a link reversal, it means u reverses all of its edges to become outgoing. Therefore none of these edges can then be part of a cycle, since there are no edges into u. So no cycle can be introduced by the link reversal if none existed before. By *question 6*, the two above properties show that G_f is D-oriented.

Using the result from *question 7* repeatedly, it follows by induction that G_f only depends on G_0.

Question 12* We can capture the Partial Reversal algorithm if all links are initially unmarked (labelled 0).