1 Introduction

In the last lecture, we defined the Binary Link Labeling algorithm (BLL). Today, we show that the Full Reversal (FR) and Partial Reversal (PR) algorithms of Gafni and Bertsekas are special cases of BLL.

Throughout these notes, $G$ is an edge-labeled graph. The labels on the edges are binary, either 0 or 1. One vertex of $G$ is called the destination and written as $D$. The objective of both FR and PR is to produce $\overrightarrow{G}$, an orientation of $G$ that is $D$-oriented, meaning that there is a path from every vertex of $G$ to $D$ along the directed edges of $G$.

2 BLL as Full Reversal or Partial Reversal

Suppose we start BLL with any orientation of $G$ as its input. Different link labeling rules permit BLL to execute different link reversal algorithms. With appropriate edge-labeling we obtain either Full Reversal or Partial Reversal, which we have already discussed in previous lectures. BLL is capable of executing other algorithms as well, but we won’t go into them.

Lemma 1 If all links of $\overrightarrow{G}$ are initially labeled 1, then BLL equals FR.

Proof: By simple induction at every step, the labels are never changed. Since all labels are 1, the LR2 rule of BLL applies always: all links are reversed, but no labels are changed. So all labels remain at 1. Therefore, BLL behaves the same as FR, since all incident edges are reversed at each step.

Lemma 2 If all links of $\overrightarrow{G}$ are initially labeled 0, then BLL equals PR.

Proof: The first step by a vertex $v$ (when it becomes a sink) is that it reverses all of $v$’s links and labels them 1. Then label 1 on link $(v, u)$ means that the link was reversed since the last time $u$ was a sink. Otherwise, the label of $(v, u)$ would already have been 1, and $u$ would have flipped the label to 0, meaning the label would not be 1 now, a contradiction.

At the next reversal step of $u$, either $u$ reverses edge $(v, u)$ and leaves the label 1 (this occurs iff all labels are 1), or does not reverse the edge $(v, u)$, in which case it changes that edge’s label to 0. Thus, label 0 means the link was not reversed at the last step of $u$.

Since the LR1 and LR2 cases of BLL are identical to the cases of PR for the same conditions, the result follows.
There is a tradeoff between the link labeling algorithm and the vertex labeling algorithm—or, more specifically, between their implementations BLL and IVL. BLL uses bounded labels, while the labels in IVL can grow without bound, but BLL has greater preprocessing cost, because the input to the algorithm must be an acyclic graph with edge labels that satisfy a specific algebraic property. IVL can accept any graph as input, and still ensure that the output graph will be acyclic.

3 Work and time complexity of link reversal algorithms

We now discuss some results by Busch and Tirthapura about the work and time complexity of link reversal algorithms, from the paper Analysis of Link Reversal Routing Algorithms, SIAM J Comp, 2005.

3.1 Definitions

Definition 1 Let the work complexity of vertex $v$ be the number of reversals done by $v$, and denote this $\sigma_v$.

As an aside, Busch and Tirthapura show that, for FR, the work complexity of a given vertex $v$ is invariant in the sense that it remains the same in any execution of the algorithm.

Definition 2 Global work complexity, denoted $\sigma$, is the sum of work complexity of all vertices.

Definition 3 A vertex in the input graph is called bad if it has no (directed) path to the destination $D$. Otherwise, the vertex is called good.

We can prove by induction on the distance from $D$ that good vertices do not take steps.

3.2 Work complexity of the Pair Algorithm

Let $h$ be any Pair Algorithm vertex labeling for the input graph $G$. Given the resulting orientation $h \circ G$, partition $V$ into layers recursively, as follows.

- $L_0 := $ a set of good vertices.
- For $k > 0$, vertex $v$ is in layer $L_k$ if $k$ is the smallest integer such that either
  1. there is an incoming link from $v$ to a vertex in layer $L_{k-1}$; or
  2. there is an outgoing link from $v$ to a vertex in layer $L_k$. 

Busch-Tirthapura 2005 shows that each vertex that is initially in layer $L_j$ takes exactly $j$ steps before halting. Thus, $\sigma_v = j$ when $v \in L_j$.

If there are $n_b$ bad vertices in $h \circ G$, then, since the maximum layer index is $n_b$, the global work complexity will be $O(n_b^2)$. This bound is tight. For every $n_b > 0$, there is a graph with $n_b$ bad vertices in which the global work complexity of the Pair Algorithm is $\Omega(n_b^2)$. An example of a worst-case graph is a simple chain, where all the links are initially pointing away from the destination. This is illustrated in Figure 1.

**Definition 4** For each node $u$, the reversal distance $rd(u)$ is the minimum over all the chains $\rho$ from $u$ to $D$ of the number of edges in chain $\rho$ that are wrong-way.

**Definition 5**

$$rd = \sum_u rd(u)$$

**Lemma 3** In any state of execution of the Pair Algorithm, $rd \leq n_b^2$, where $n_b$ is the number of bad nodes as before. For good nodes $u$, $rd(u) = 0$; for bad nodes $v$, $rd(v) \leq n_b$.

**Proof:** We argue that each step reduces this metric $rd$. At each step, at least one node $v$ fires. Then the node $v$ has its $rd$ reduced by 1. Further, the firing of $v$ does not affect the $rd$ of any other node. (Either $v$ was not in the path to $D$, or the path gains by losing a wrong-way edge. Either way, there’s no change to any other node’s $rd$.) So after at most $n_b^2$ steps, the algorithm terminates, with a destination-oriented graph.

**Claim 4** A good node never fires.

**Proof:** We prove this by induction on the length $k$ of the shortest path from $v$ to $D$.

For $k = 0$, $v = D$, and $D$ never fires because it is the destination. Now assume the claim is true for length $k$. Consider $v$ with shortest path $k + 1$ to $D$. Let $(v, v')$ be the first edge in the path. By the inductive hypothesis, $v'$ never fires, so the edge $(v, v')$ never reverses, and $v$ can never become a sink, so $v$ will never fire either.

Any two executions of FR starting with the same link directions are “equivalent” in the following sense: each node performs the same number of reversals, and the final link directions of the output orientation are the same.
Figure 1: An example of worst-case work complexity of the Full Reversal algorithm.