1 Introduction

Today, Dr. Soma started the class by talking about the Vertex Label algorithm, which is one way to solve the link reversal algorithm. With this algorithm, we tried to solve the problem by labeling the vertex in a graph with the labeling function that we would describe in detail in the following section.

2 Vertex Label Algorithm

Definition 1 Let $L$ be totally ordered, countably infinite set of labels, that is partitioned into a collection $\{L_v|v \in V\}$ of distinct subsets. Let $\mathcal{L}$ be the set of all functions $h$ from $V$ to $L$ such that $h(v) \in L_v$ for all $v \in V$. Each function $h$ in $\mathcal{L}$ is called a vertex labeling of graph $G$, $h$ induces an orientation of $G$ denoted $h.G$, which is $\bar{G} = (v, \bar{E})$ where $(u, v) \in \bar{E}$ if and only if $(u, v) \in E$ and $h(u) > h(v)$.

2.1 General Idea

For each vertex $v \in V$, we define a function $g_v$ from vertex labeling of $G$ to labels for vertex $v$ that produces a new label for $v$. Note that functions depend only on label of $v$ and its neighbors, and it is increasing.

1. If $h(u) = h'(u)$ for all $u \in v \cup N_G(v)$ then $g_v(h) = g_v(h')$

2. For each infinite sequence $< h_1, h_2, ... >$ of vertex labeling of $G$ such that $v$ is a sink in $h_i.G$ and $h_{i+1}(v) = g_v(h_i)$, for all $i \geq 1$, and the sequence $< h_1(v), h_2(v), ... >$ increases without bound.

Lemma 1 IVL always terminates.

Proof: We assume for contradiction, that there is an execution of this algorithm that does not terminate. Let $W$ be the set of vertices that have been chosen infinitely often in $S$. Since $V$ is finite, $W \neq \emptyset$, also $V - W \neq \emptyset$, since $D \in V - W$. Since $G$ is connected, $\exists (u, v)$ such that $u \in W, v \in V - W$. Let $t$ be the iteration where $v$ takes the last step. After $t$, $v$'s label does not change. However, since label of $u$ increases every time $u$ takes a step after $t$, and $u$ takes infinitely many steps, $u$'s label will eventually exceed $v$'s label and the edge will be directed from $u$ to $v$, so $u$ will no longer take step. This is the contradiction. ■
Algorithm 1 Increasing Vertex Label (IVL)

Require: Undirected graph $G = (V, E)$ with distinguished vertex $D \in V$ and vertex labeling $h$

while $h.G$ has a sink other than $D$ do
    Choose a subset $S$ of sinks in $h.G$, such that $D \not\in S$
    for all $v \in S$ do
        $h(v) := g_v(h)$
    end for
end while

Lemma 2 IVL maintains acyclicity.

Proof: We follow directly from fact that labels are totally ordered and orientation of $G$ is induced by vertex labels. Thus, it maintains acyclicity. ■

Theorem 3 IVL ensures destination-orientation.

Gafni and Bertsekas proved for any $h.G$ that “No matter what subsets are chosen at each iteration, the final destination-orientation graph will be the same and each vertex has same number of reversal”.

3 Pair Algorithm (Implements FR using IVL)

We assume $v = \{0, 1, ..., n\}$ with 0 being destination. Each label for $v \in V$ is $(c, v)$. A sink changes label be setting $c$ to be one more than max of all neighbors. $\forall v \in V, L_v = \mathbb{N} \times \{v\}$ countably infinite label ordered based on lexicographic ordering. $g_v(h) = (1 + \max(C), v)$ where $C = \{c|h(u) = (c, u) \text{ for some } u \in N_g(v)\}$.

Lemma 4 Pair algorithm is a special case of IVL.

Proof: Clearly, $g_v$ depends only in label of $N(v)$ and increases without bound. ■

4 Triple Algorithm (Implements PR using IVL)

Each label for vertex $v \in V$ is an ordered triple of integers $(a, b, v)$. The label of a sink vertex $v$ is changed to ensure 2 things, which are:

1. The new label is larger than those of neighbors with smallest first component among all neighbors.
2. The new label is smaller than those of all other neighbors.

```latex
\texttt{for all } v \in V \texttt{ do }
\begin{align*}
& L_v \leftarrow \mathbb{N} \times \mathbb{Z} \times \{v\} \\
& \text{Compute } a \text{ and } b \\
& g_V(h) \leftarrow (a, b, v)
\end{align*}
\texttt{end for}
```

We can compute the value of $a$ and $b$ by:

```latex
\texttt{for all } v \in V \texttt{ do }
\begin{align*}
& h(v) \leftarrow (a_0, b_0, v) \\
& \text{Define } a_1 = \min\{a | h(u) = (a, b, u) \text{ for some } b \text{ and some } u \in N_G(v) + 1\} \\
& \text{if } \{b | h(u) = (a, b, u) \text{ for some } u \in N_G(v) = \emptyset \text{ then} \\
& \quad \text{define } b_1 = b_0 \\
& \quad \text{else} \\
& \quad \quad b_1 = \min\{b | h(u) = (a, b, u) \text{ for some } u \in N_G(v)\} - 1 \\
& \text{end if}
\end{align*}
\texttt{end for}
```

**Lemma 5** Triple algorithm is a special case of IVL where:

1. $g_v$ depends only on neighbors
2. Increasing without bound

**Corollary 6** Pair algorithm and Triple algorithm terminate with a destination-oriented graph.

## 5 Class Discussion

In the class, Vladimir had mentioned about the increasing function in the algorithm that there may be a chance that the execution of algorithm can be error because if we have a countably infinite set of labels such that all labels for vertex $v$ are still smaller than all labels of vertex $w$, then, we are able to see that it is not enough for each set $L_v$ to be unbounded but they need to be unbounded with respect to the entire set $L$. Consequently, Dr. Soma had added some correcting point to assure correctness by adding the constraint that for all labels $a$ and all vertices $v$, there is a label $a_v$ of vertex $v$ that is larger than $a$. That is:

\[ \forall a \in L, \forall v \in V, \exists a_v \in L_v, a < a_v \]
6 Conclusion

For summary, we finished the class talking about Vertex Labeling algorithm to solve the reversal problem with one sink vertex. Next class, we are going to have a discussion about Link Label Algorithm. The rough idea of link label algorithm is to label the link with binary value instead of labeling the vertex.