Project of Algorithms on Node Labeling

Kevin Lei

August 2, 2024

1 Introduction

In this project we discuss the "Node Labeling Problem", in which we attempt to label the nodes of a graph with unique labels from a set of labels. We have the following definitions:

- Let $G = (V, E)$ be an undirected graph.
- Let $d(u, v)$ be the distance between nodes u and v.
- For all nodes $v \in V$, let $N(v, h) \subseteq V$ be the set of nodes that are at most h hops away from v.
- Let $K = \{0, 1, \ldots, k-1\}$ be the set of k integers, where $k \leq |V|$.
- For all $v \in V$, let $c(v) \in K$ be the label of node v, where different nodes may have the same label.
- Let $C(v, h)$ be the set of labels of nodes in $N(v, h)$.
- A labeling of the nodes is *valid* if every label in K is used at least once.
- Let $r(v)$ be the smallest integer such that the node v has all the labels in K in $N(v, r(v))$.
- Let $m(v)$ be the smallest integer such that the node v has at least k nodes in $N(v, m(v))$.

Formally, our relevant sets and values can be defined as follows:

$$\begin{aligned}
N(v, h) &\triangleq \{u \in V \mid d(u, v) \leq h\} \\
C(v, h) &\triangleq \{c(u) \mid u \in N(v, h)\} \\
r(v) &\triangleq \min\{h \mid |C(v, h)| = k\} \\
m(v) &\triangleq \min\{h \mid |N(v, h)| \geq k\}.
\end{aligned}$$

Note that in general, we have $|C(v, h)| \leq |N(v, h)|$, since the labels of nodes in $N(v, h)$ are not necessarily distinct, and $r(v) \geq m(v)$, since there must be at least one node per label but not necessarily one label per node.

The Node-Labeling Decision Problem is defined as follows:

Given:

- An undirected graph $G = (V, E)$
- A set of $k \leq |V|$ labels $K = \{0, 1, \ldots, k-1\}$
- A nonnegative integer R,

does there exist a labeling $c(v)$ for all $v \in V$ such that $|C(v, R)| = k$ for all $v \in V$?

Now consider this as an optimization problem. The Node-Labeling Optimization Problem is defined as follows:

Given:

- An undirected graph $G = (V, E)$
- A set of $k \leq |V|$ labels $K = \{0, 1, \ldots, k-1\}$,

find a valid labeling for all the nodes such that $\max_{v \in V} \frac{r(v)}{m(v)}$ is minimized.

In the case of the optimization problem, if an algorithm that solves it has $\max_{v \in V} \frac{r(v)}{m(v)} \leq \rho$ for all possible instances, then we say that the algorithm has a *proximity ratio* of ρ, and the algorithm is a ρ-proximity algorithm.

First, we will prove that the Node-Labeling Decision Problem is NP-Complete. Then, we will present a polynomial-time algorithm for the Node-Labeling Optimization Problem where the graph is a tree, analyze the proximity ratio of the algorithm, and finally analyze the runtime complexity of the algorithm.

2 NP-Completeness Proof

Theorem 1. *The Node-Labeling Decision Problem is NP-Complete.*

Proof. A problem is NP-Complete if it is in NP and every problem in NP can be reduced to it in polynomial time. We will show the former by presenting a polynomial time algorithm to verify a solution to the Node-Labeling Decision Problem, and the latter by reducing HAM-CYCLE to the Node-Labeling Decision Problem.

First, consider the following algorithm to verify a solution to the Node-Labeling Decision Problem:

Algorithm 1: Verify a Solution to the Node-Labeling Decision Problem

```
Input: An undirected graph $G = (V, E)$, a set of $k \leq |V|$ labels
       $K = \{0, 1, \ldots, k-1\}$, a nonnegative integer $R$, and a labeling
       $c : V \to K$
Output: True if the labeling is valid and $|C(v, R)| = k$ for all $v \in V$,
        False otherwise

for $v \in V$ do
    $l = \emptyset$
    if $\neg$ BFS($v$, 0, $l$) then
        return False
    end
end
return True
```

Algorithm 2: BFS

```
Input: A node $v$, current depth $d$, set of labels seen $l$
Output: True if all $k$ labels are seen within depth $R$, False otherwise

if $d > R$ then
    return False
end
$l = l \cup \{c(v)\}$
if $|l| = k$ then
    return True
end
for each neighbor $u$ of $v$ do
    if BFS($u$, $d+1$, $l$) then
        return True
    end
end
return False
```

This algorithm works by performing a breadth-first search from each node v in the graph, and checking if all k labels are seen within depth R. If a depth of R is reached without seeing all k labels, the algorithm returns False. Otherwise, the algorithm returns True. The algorithm runs in $O(|V| \cdot (|V| + |E|))$ time, which is polynomial in the size of the input. Thus, the Node-Labeling Decision Problem is in NP.

Now we perform a reduction from HAM-CYCLE to the Node-Labeling Decision Problem. Given an instance of HAM-CYCLE with the graph $G = (V, E)$, we construct an instance of the Node-Labeling Decision Problem as follows:

- The graph is the same: $G = (V, E)$.
- $k = |V|$.
- $R = |V| - 1$.

We claim that there exists a Hamiltonian cycle in G if and only if there exists a valid labeling for the corresponding instance of the Node-Labeling Decision Problem.

($\Rightarrow$) Assume there exists a Hamiltonian cycle in G. Label the nodes in the Hamiltonian cycle from 0 to $|V|-1$ in order. Thus, for all $v \in V$, all other nodes are at least $|V|-1$ hops away. This means that $N(v,R)$ contains all nodes in V, so $|C(v,R)| = |V| = k$. Thus, the labeling is valid.

($\Leftarrow$) Assume that there exists a labeling $c(V)$ for all $v \in V$ such that $|C(v,R)| = k$ for all $v \in V$ where $k = |V|$ and $R = |V|-1$. Seeking a contradiction, assume that there is no Hamiltonian cycle in G. For all nodes $v \in V$, since $|C(v,R)| = |V| = k$, v must be able to reach all the other nodes in V within $|V|-1$ hops. Now consider the shortest path starting from v that visits all other nodes in V. From our assumption that there is no Hamiltonian cycle, this path must visit some node more than once. If the shortest path visits some node more than once, then its length is greater than $|V|-1$, contradicting the fact that v can reach all other nodes in V within $|V|-1$ hops. Thus, there must be a Hamiltonian cycle in G.

This reduction can be done in polynomial time, since the graph is the same, and if we really need to, we can count $|V|$ in $O(|V|)$ time. Since the Node-Labeling Decision Problem is in NP and HAM-CYCLE can be reduced to it in polynomial time, the Node-Labeling Decision Problem is NP-Complete. $\square$

3 Approximation Algorithm

Here we discuss an algorithm to solve the Node-Labeling *Optimization* Problem when the input graph is a tree.

4 Pseudocode

Algorithm 3:

```
  Input:
  Output:
```

5 Proximity Ratio Analysis

6 Runtime Complexity Analysis