Ferianakademie 2010 Course 2: "Distance Problems: Theory and Praxis"

Distance Labelings

Stepahn M. Günther

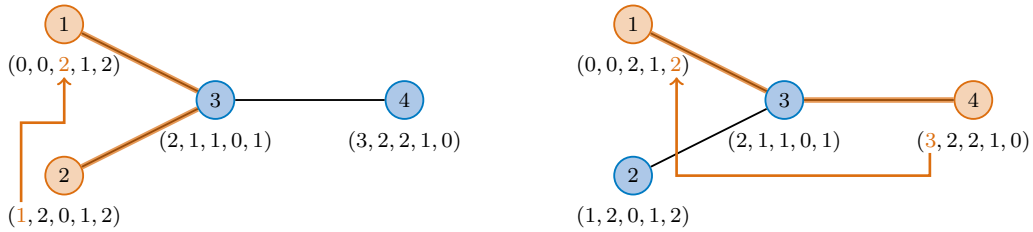
September 23, 2010

Abstract

Distance labels allow to infer the shortest distance between any two vertices of a graph given their labels. It is straight-forward to construct such labels by listing the single source shortest path distances for each vertex. For a graph with n vertices this results in a memory requirement of $\mathcal{O}(n \log n)$ bits per vertex and constant lookup time. The question arises whether there are labeling schemes which allow for smaller labels at cost of lookup time. Such a scheme might be useful to reduce the size of routing tables in sensor networks which typically consist of nodes with very limited storage and processing capabilities. Instead of storing the next-hop for all possible destinations it is sufficient to store the labels of neighboring nodes. Once a packet arrives its destination label can be used to determine the distance between all neighboring nodes and the packet's destination. The next-hop is then given by the neighbor with minimum distance to the packet's destination.

This report discusses a labeling scheme introduced by Gavoille et al. in [GPPR02] for general, connected, and undirected graphs with unit edge weights. It allows for $\mathcal{O}(n)$ bit labels per vertex and a lookup time of $\mathcal{O}(\log \log n)$. Schemes which allow for sub linear sized labels are available for certain classes of graphs and are discussed in [GPPR02].

Contents



(a) $f(L(1), L(2)) = d(1, 2) = 2$ (b) $f(L(1), L(4)) = d(1, 4) = 2$

Figure 1: The rank of vertices $u \in V = \{1, 2, 3, 4\}$ is given by their ordering according to V. In (a) the rank of vertex 2 is read from label $L(2)$ and used as index in the list of distances stored in $L(1)$ which gives the distance $d(1, 2)$. A second example is shown in (b).

1 Introduction

We denote an undirected graph by $G = (V, E)$ where V is the set of vertices and $E \subseteq V \times V$ the set of edges. A graph belongs to a certain class $\mathcal{G}$ of graphs. For the scope of this report we only discuss the class of general, connected, and undirected graphs with unit edge weights. Further we denote the shortest distance between vertices $u, v \in V$ by $d(u, v)$. A distance labeling scheme is defined as tuple $\langle L, f \rangle$ where L denotes the set of labels $L(u)$ for all $u \in V$ and f represents a valid distance decoder, i.e.

$$f(L(u), L(v)) = d(u, v), \ \forall u, v \in V. \tag{1}$$

A trivial distance labeling scheme is given as follows. Choose an arbitrary but fixed ordering of vertices in V. Assign to each vertex $u \in V$ a label consisting of two fields

(a) the rank order(u) according to the ordering of V and

(b) the list of distances $\{d(u, v) \mid \forall v \in V\}$.

Thus, each label needs $\log n + n \log n \in \mathcal{O}(n \log n)$ bits of memory. Given two labels $L(u)$ and $L(v)$ one can easily compute the shortest distance between u and v by using field (a) of label $L(v)$ as index in field (b) of label $L(u)$. An example is given in Figure 1.

2 Preliminaries

The distance labeling scheme presented in Section 3 relies on the hierarchical decomposition of a graph into *dominating sets* which are introduced in this section. Furthermore, an algorithm to determine dominating sets of limited size is outlined.

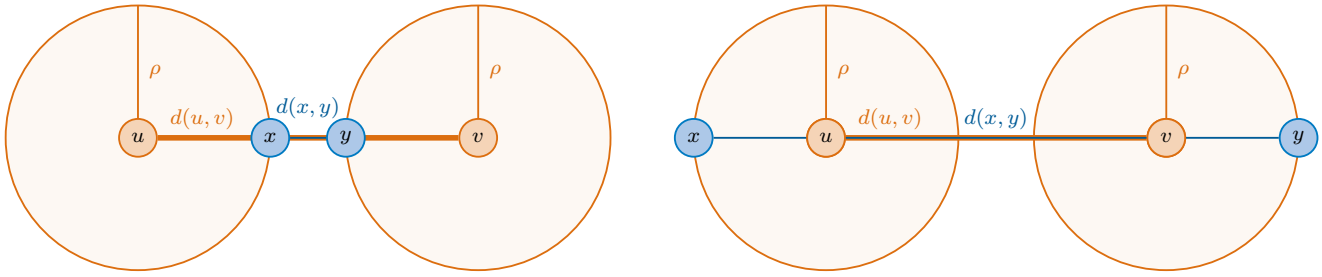
2.1 Dominating sets and dominating collections

Given a graph $G = (V, E)$, a subset $S \subseteq V$ of nodes is called *ρ-dominating set* if

$$\forall v \in V \ \exists u \in S : \ d(u, v) \leq \rho. \tag{2}$$

We call node $u \in S$ *dominator of* v if

$$v = \arg\min_{u \in S} d(u, v). \tag{3}$$



(a) Lower bound for $d(x, y)$ given $d(u, v)$ (b) Upper bound for $d(x, y)$ given $d(u, v)$

Figure 2: Distance between dominators

Let $d_{\max}$ denote the maximum distance between two vertices in V. Then, for $\rho \geq d_{\max}$ any single vertex $u \in V$ forms a ρ-dominating set. Conversely, from $\rho = 0$ follows that $S = V$ since each vertex dominates only itself. For a vertex $u \in V$ we denote the dominator of u with respect to the dominating set S by $\mathrm{dom}_S(u)$. If the distance between the dominators of two vertices is known, the distance between those two nodes is bounded by the following lemma.

Lemma 2.1. *Given a graph $G = (V, E)$ and a ρ-dominating set S, then for every two vertices $x, y \in V$ the following two facts hold:*

1. $d(dom_S(x), dom_S(y)) - 2\rho \leq d(x, y) \leq d(dom_S(x), dom_S(y)) + 2\rho$
2. *The distance $d(x, y)$ can be derived if*
   - *the dominating radius ρ,*
   - *the value $d(x, y) \mod (4\rho + 1)$, and*
   - *the distance between dominators $d(dom_S(x), dom_S(y))$*

   are known.

Proof. Fact (1) can easily be seen when considering Figure 2. The dominators for $x, y \in V$ are given by $\mathrm{dom}_S(x) = u$ and $\mathrm{dom}_S(y) = v$. Since $d(x, u) \leq \rho$ and $d(y, v) \leq \rho$, the distance between x and y is bounded below by $d(u, v) - 2\rho$ and bounded above by $d(u, v) + 2\rho$. Fact (2) holds since (1) defines a range of $2\rho + 1$ consecutive values and exactly one of these values equals to $d(x, y) \mod (4\rho + 1)$. □

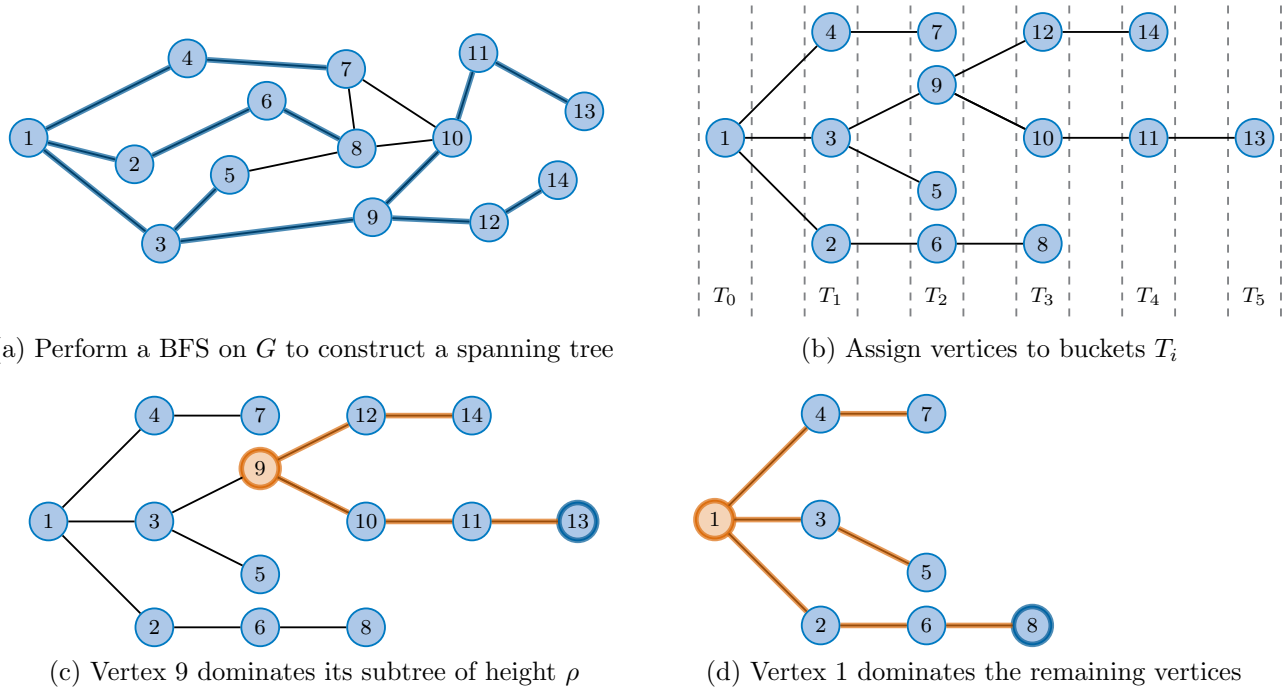
Given a graph G and a decreasing sequence of integers $\{\rho_i\}$ for $0 \leq i \leq k$ with $\rho_k = 0$, we call the set of ρ_i-dominating sets

$$\mathcal{S} = \{(S_i, \rho_i) \mid 0 \leq k \leq i\} \tag{4}$$

a *dominating collection* for G. The dominating sets S_i typically become progressively larger with increasing index since ρ_i decreases. The set S_k consists of all vertices $v \in V$ since ρ_k is declared zero. Such a collection can be seen as a hierarchical decomposition of G where each vertex $x \in S_i$, $i > 0$ is dominated by a node $x' \in S_{i-1}$.

2.2 Determining dominating sets

Although any ρ-dominating set also forms a ρ'-dominating set for $\rho' > \rho$ we are interested in small dominating sets. Ideally, we would choose minimum dominating sets. Unfortunately, the problem of deciding whether a given dominating set is a minimum dominating set can be reduced to the set covering problem which is $\mathcal{NP}$-hard. The computational effort to determine minimum



(a) Perform a BFS on G to construct a spanning tree

(b) Assign vertices to buckets T_i

(c) Vertex 9 dominates its subtree of height ρ

(d) Vertex 1 dominates the remaining vertices

Figure 3: Construction of a ρ-dominating set for $\rho = 3$.

dominating sets is therefore not affordable. For this reason we demand dominating sets which are "small enough". The following lemma [PU89] gives an upper bound for the size of a ρ-dominating set.

Lemma 2.2. *For every n-vertex graph G and integer $\rho \geq 0$ there exists a dominating set S with*

$$|S| \leq \max\left\{\left\lfloor \frac{n}{\rho+1} \right\rfloor, 1\right\}. \tag{5}$$

Proof. Given a connected and undirected general graph $G = (V, E)$ with unit edge weights a ρ-dominating set S which obeys Lemma 2.2 can be found as follows:

1. Perform a BFS on G to construct a spanning tree T.
2. Assign vertices $v \in T$ to buckets T_i according to their depth in T.
3. Starting at an arbitrary leaf $l \in T_i$ with maximal depth i repeat the following steps:
   - Go up ρ nodes yielding an inner vertex $w \in T$.
   - Add w to the ρ-dominating set S.
   - Remove the subtree rooted at w and update the affected buckets.
   - If the height of the remaining spanning tree is less than or equal to ρ, add its root to S and terminate.

Starting at a leaf l and finding its ρ-th predecessor w ensures that w is a dominator for at least $\rho + 1$ vertices – namely itself and ρ vertices below it. If there are branches along the path from w to l, then l dominates strictly more than $\rho + 1$ vertices. By choosing a leaf l in maximum depth it is ensured that the whole subtree rooted at w has height ρ and is thus dominated by w. If the height of the remaining spanning tree is less than or equal to ρ, then the tree's root dominates the remaining vertices and is added to S. □

$\text{order}_{S_0}(u)$	$\{d(u,v) \mid \forall v \in S_0\}$

(a) Label $L^0(u)$ of vertex $u \in S_0$

$L^{i-1}(\text{dom}_{S_{i-1}}(u))$	$\text{order}_{S_i}(u)$	$\{d(u,v) \mod (4\rho_{i-1}+1) \mid \forall v \in S_i\}$

(b) Label $L^i(u)$ of vertex $u \in S_i$

Figure 4: Labels for nodes in S_0 and S_i, $0 < i \leq k$.

An example for the algorithm outlined above is given in Figure 3. Regarding the time complexity of this algorithm we have to perform a BFS once which takes $\mathcal{O}(|V| + |E|) \subseteq \mathcal{O}(n^2)$ since $|E| \leq n(n-1)/2$. Afterwards, each ρ-dominating set can be constructed by traversing the spanning tree starting at the leafs which takes $\mathcal{O}(n)$ steps per set.

3 Upper bound for general graphs

Based on the hierarchical decomposition of G, a labeling scheme is discussed in this section which allows for labels linear in size to the number of vertices in G. The upper bounds $\mathcal{O}(n)$ for the label size and $\mathcal{O}(\log \log n)$ for the decoding time are proofed by construction.

3.1 Creating labels

First, we define a dominating collection $\mathcal{S} = \{(S_i, \rho_i) \mid 0 \leq i \leq k\}$ with

$$k = \lceil \log \log n \rceil \quad \text{and} \quad \rho_i = 2^{k-i} - 1. \tag{6}$$

Given this definition of ρ_i and Lemma 2.2 it follows directly that $|S_i| \leq \frac{n}{2^{k-i}}$. Note, that S_0 is the smallest dominating set and that the sets S_i become progressively larger. It is easy to verify that $\rho_k = 0$ and thus $S_k = V$ for a given graph $G = (V, E)$. The ordering of vertices within a dominating set S_i can be is arbitrary but fixed. Once the dominating collection $\mathcal{S}$ has been determined labels for vertices are created as follows (see Figure 4):

1. For all nodes $u \in S_0$ create labels which consist of the following two fields:
   (a) The rank $\text{order}_{S_0}(u)$ according to the ordering of S_0
   (b) The list of distances $\{d(u,v) \mid \forall v \in S_0\}$
2. For $0 < i \leq k$ create labels for all vertices $u \in S_i$ which consist of the following three fields:
   (a) The label $L^{i-1}(\text{dom}_{S_{i-1}}(u))$ of the dominator of u in the set S_{i-1}
   (b) The rank $\text{order}_{S_i}(u)$ according to the ordering of S_i
   (c) The list of distances $\{d(u,v) \mod (4\rho_{i-1}+1) \mid \forall v \in S_i\}$

Step (1) stores the real distances between all vertices $u, v \in S_0$. For all other dominating sets S_i, $0 < i \leq k$ the dominator $u' = \text{dom}_{S_{i-1}(u)}$ of $u \in S_i$ is determined. The label $L^{i-1}(u')$ becomes part of the label $L^i(u)$. Additionally, the rank of u according to the order of S_i is added. The trick is now not to store the real distances $d(u,v)$ between u and other vertices $v \in S_i$ but the value $d(u,v) \mod (4\rho_{i-1}+1)$. Thus, the memory required to store a distance value is limited to $\log(4\rho_{i-1}+1)$ instead of $\log n$ bits. Since $S_k = V$, it is made sure that each vertex obtains a valid label. Note, that one would end up with the trivial labeling scheme outlined in Section 1 if $S_0 = V$ (and $k = 0$) was chosen.

3.2 Decoding labels

Given two vertices $x, y \in V$, the distance $d(x, y)$ can be inferred recursively using the following distance decoder:

1. Obtain the labels $L^{i-1}(x')$ and $L^{i-1}(y')$ for $x' = \mathrm{dom}_{S_{i-1}}(x)$ and $y' = \mathrm{dom}_{S_{i-1}}(y)$ using field (a) of the labels $L^i(x)$ and $L^i(y)$, respectively
2. Determine the distance $d(x', y')$ between the dominators (recursive step)
3. Determine the rank order(y) from field (b) of $L^i(y)$
4. Use order(y) as index in field (c) of $L^i(x)$ to determine the value $d(x, y) \mod (4\rho_{i-1} + 1)$
5. Use Lemma 2.1 to infer $d(x, y)$ relying on the fact that S_i is a ρ_i-dominating set

The recursion stops for $i = 0$ since the distance between vertices in S_0 can be determined directly.

3.3 Size of labels

The label of any node $v \in V$ is built over k steps. In each step $0 \leq i \leq k$ the respective rank order$_{S_i}$ according to the ordering of S_i is added. Consequently, a total of $\mathcal{O}(k \log n)$ bits is needed for the ranks. In step $i = 0$ the real distances between all vertices in S_0 and thus $|S_0| \log n$ bits per label are stored. For the remaining steps $0 < i \leq k$ the distances are computed modulo $(4\rho_{i-1} + 1)$. For this reason, additionally $|S_i| \log(4\rho_i + 1)$ bits are needed per label and step. Putting all together we end up with an upper bound for the size of labels which is given by

$$L_{\max} \leq |S_0| \log n + \sum_{i=1}^{k} |S_i| \log(4\rho_{i-1} + 1) + \mathcal{O}(k \log n). \tag{7}$$

For the individual terms we obtain

- $|S_0| \log n \leq \dfrac{n}{2^k} \log n = \dfrac{n}{2^{\log \log n}} \log n = \dfrac{n}{\log n} \log n = n$,
- $\displaystyle\sum_{i=1}^{k} |S_i| \log(4\rho_{i-1} + 1) \leq 8n$ (see Appendix A), and
- $\mathcal{O}(k \log n) \in \mathcal{O}(\log \log n \log n)$.

Thus, the label size $L_{\max}$ is bounded by $9n + \mathcal{O}\log \log n \in \mathcal{O}(n)$.

3.4 Time needed for decoding

The individual steps for decoding outlined in Section 3.2 are basically lookup operations in arrays (obtaining labels, the rank of nodes, distance values, and computing distances according to Lemma 2.1) and thus require constant time. The number of recursive steps is bounded above by $k = \lceil \log \log n \rceil$. Consequently, the time needed for decoding is in $\mathcal{O}(\log \log n)$.

4 Conclusion

We have seen a distance labeling scheme for general graphs with unit edge weights which allows for label sizes linear in the number of vertices. The upper bound was explicitly derived. A lower bound is shown in [GPPR02]. Furthermore, for specific classes of graphs even better schemes are shown in [GPPR02]. In particular, the bounds

- $\Theta(\log^2 n)$ for trees,
- $\mathcal{O}(\sqrt{n}\log n)$ and $\Omega(n^{1/3})$ for planar graphs, and
- $\Omega(\sqrt{n})$ for bounded degree graphs.

are derived.

A Label sizes

In Section 3.3 it is claimed that

$$\sum_{i=1}^{k} |S_i| \log(4\rho_{i-1}+1) \leq 8n.$$

Recall, that $k = \lceil \log\log n \rceil$, $\rho_i = 2^{k-i}-1$ and thus $|S_i| \leq \frac{n}{2^{k-i}}$. Then we obtain

$$\sum_{i=1}^{k} |S_i| \log(4\rho_{i-1}+1) = \sum_{i=1}^{k} \frac{n}{2^{k-i}} \log(4\cdot(2^{k-i+1}-1)+1) = n\cdot\sum_{i=1}^{k} \frac{\log(2^{k-i+3}-3)}{2^{k-i}}$$

$$\leq n\cdot\sum_{i=1}^{k} \frac{k-i+3}{2^{k-i}} = n\cdot\left(\frac{3}{2^0}+\frac{4}{2^1}+\frac{5}{2^2}+\ldots+\frac{k+2}{2^{k-1}}\right) = n\cdot\sum_{i=0}^{k-1}\frac{i+3}{2^i} = n\cdot\left(\sum_{i=0}^{k-1}\frac{i}{2^i}+3\sum_{i=0}^{k-1}\frac{1}{2^i}\right)$$

$$= n\cdot\left(\sum_{i=0}^{k-1}\left[\frac{\partial}{\partial x}x^{i+1}\right]_{x=\frac{1}{2}} - \sum_{i=0}^{k-1}\frac{1}{2^i}+3\sum_{i=0}^{k-1}\frac{1}{2^i}\right) = n\cdot\left(\left[\frac{\partial}{\partial x}\sum_{i=0}^{k-1}x^{i+1}\right]_{x=\frac{1}{2}}+2\sum_{i=0}^{k-1}\frac{1}{2^i}\right)$$

$$\leq n\cdot\left(\left[\frac{\partial}{\partial x}\sum_{i=0}^{\infty}x^{i+1}\right]_{x=\frac{1}{2}}+2\sum_{i=0}^{\infty}\frac{1}{2^i}\right) = n\cdot\left(\left[\frac{\partial}{\partial x}\frac{1}{1-x}\right]_{x=\frac{1}{2}}+\frac{2}{1-\frac{1}{2}}\right)$$

$$= n\cdot\left(\left[\frac{1}{(1-x)^2}\right]_{x=\frac{1}{2}}+4\right) = 8n.$$

References

[GPPR02] Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. Journal of Algorithms, 53:85–112, 2002.

[PU89] David Peleg and Eli Upfal. A tradeoff between space and efficiency for routing tables. Journal of the ACM, 53(3):510–530, 1989.