Chapter 6

Light Approximate Shortest Paths Trees

This chapter is based on a paper by Khuller, Raghavachari and Young [4]. Awerbuch, Baratz and Peleg have related results on so-called shallow light trees [3].

There are some other recent papers on related topics. Althofer et al [1] study spanners of weighted graphs. Mansour and Peleg [5], Salman et al [6], and Awerbuch and Azar [2] give approximation algorithms for related problems in network design where the objective function is nonlinear.

6.1 Introduction

Given a graph $G = (V, E)$, nonnegative costs $c(e)$ on the edges $e \in E$, and a specified root node s, can we always find a spanning tree that has both approximately minimum cost and that approximately preserves the cost of shortest s-v paths for all nodes v? Does a shortest paths tree (with root s) or a minimum spanning tree always achieve these goals? Unfortunately, the answer to the second question is no – demonstrating this is left as an exercise for the reader.

For any tree T of G, recall that $c(T)$ denotes the cost of the tree $c(T) = \sum_{vw \in T} c(vw)$. For any node v, let $d(v)$ denote the cost of a shortest s-v path. Let T_s denote a shortest paths tree of G with root s, and let T_M denote a minimum spanning tree of G. Let α and β be numbers ≥ 1. A spanning tree T of G is called an (α, β) light approximate shortest-paths tree (LAST) if

1. for all nodes $v \in V$, the cost of the unique s-v path in T is at most $\alpha \cdot d(v)$, and
2. $c(T) \leq \beta \cdot c(T_M^*)$, where T_M is a minimum spanning tree of G.

6.2 The preorder traversal of a tree

Before developing the LAST algorithm, we need to define a preorder traversal of a tree and to prove a lemma on preorder traversals. Let T be a tree, and take s to be the root node. Consider any node v. A node w adjacent to v in T is called the *parent* of v if w lies on the unique v-s path in

T, otherwise, w is called a *child* of v. Every node except s has one parent (s has no parent), and has zero or more children. For a node $v \in V$, the subtree of T with root v is the subgraph of T induced by all nodes x such that v lies in the unique x-s path of T.

Let the number of nodes in T be n. A *preorder numbering* (or depth-first-search numbering) of T assigns the numbers $1, \ldots, n$ to the nodes of T. More precisely, a preorder numbering is a bijection from $V(T)$ to $\{1, 2, 3, \ldots, n\}$ such that the root s is assigned the number 1, and for each child v_j of s, the subtree T_j with root v_j is assigned a preorder number using the numbers

$$2 + \sum_{i=1}^{j-1} |V(T_i)|, \ \ldots, \ 1 + \sum_{i=1}^{j} |V(T_i)|.$$

The following algorithm computes a preorder numbering of T using the recursive procedure PREORDER.

Algorithm Preorder Numbering of a Tree
input: Tree T and root node s.
output: Preorder numbering num(v), $\forall v \in V(T)$.

$i := 1$; (initialize global variable)
PREORDER(s); (call recursive procedure to do the numbering)
for each node v **do**
  output num(v) **end**;

procedure PREORDER(node v);
num(v) $:= i$;
$i := i + 1$;
for each child w of v **do**
  PREORDER(w); (call recursive procedure)
end; (for)
end; (procedure)

The next lemma is used in the analysis of the LAST algorithm. For any two nodes v and w in a tree T, let $d_T(v, w)$ denote the cost of the unique path in T between v and w.

Lemma 6.1 *Let T be a spanning tree with root s. Let $z_0, z_1, \ldots, z_k$ be any k nodes of T, arranged according to a preorder numbering of T. Then*

$$\sum_{i=1}^{k} d_T(z_{1-1}, z_i) \quad \leq \quad 2\, c(T).$$

Proof: Focus on the walk W formed by "doubling" every edge of T and making a preorder traversal of T; if we draw T on the plane such that each edge is drawn as a thin strip, then W

corresponds to the boundary of the drawing of T. Since each edge occurs exactly two times in W, the cost of W, $c(W)$, equals $2\,c(T) = 2 \sum_{e \in E(T)} c(e)$ (recall that the cost of a walk $W = v_0 v_1 v_2 \ldots v_\ell$ is $\sum_{i=1}^{\ell} c(v_{i-1}, v_i)$).

Note that the first occurrence of node $z_i (0 \leq i < k)$ on W precedes the first occurrence of node z_{i+1} on W, because in the preorder numbering z_0 precedes z_1 precedes ... precedes z_k. Hence, the edges of W can be partitioned into edge-disjoint sequences $W(z_i, z_{i+1})$ $(0 \leq i < k)$ and $W(z_k, z_0)$. The lemma follows since for each of these subwalks $W(z_i, z_{i+1})$, the cost, $c(W(z_i, z_{i+1}))$ is $\geq d_T(z_i, z_{i+1})$. ▮

6.3 An algorithm for finding a light approximate shortest-paths tree

This section presents the LAST algorithm, and proves that for any given numbers α, $1 < \alpha$, and β, $\beta \geq 1 + \frac{2}{\alpha - 1}$, the algorithm correctly finds an (α, β) LAST. Let T_M denote a minimum spanning tree of the given network. Step 3 of the following algorithm constructs a spanning subgraph $H = (V, E')$ of G that contains T_M such that for each node v, the cost of a shortest s-v path in H is at most $\alpha \cdot d(v)$. Theorem 6.2 below proves that the cost of H, $\sum_{vw \in E'} c(vw)$, is at most $\beta \cdot c(T_M)$.

Algorithm (α, β) Light Approximate Shortest-paths Tree
input: Graph $G = (V, E)$, root node s, nonnegative edge costs c,
numbers α and β such that $\alpha > 1$, $\beta \geq 1 + \frac{2}{\alpha-1}$.
output: Spanning tree T^* of G which is an (α, β) LAST.

STEP 1: find a minimum spanning tree T_M of G, c;
find a shortest-paths tree T_S of G, c with start node s;
STEP 2: find a preorder numbering of T_M using s as the start node;
STEP 3: $H := T_M$;
for each node v in the preorder sequence of T_M **do**
find a shortest s-v path P in H;
if $c(P) > \alpha \cdot d(v)$
then add all the edges in a shortest s-v path in G to H;
end; (if)
end; (for)
STEP 4: find a shortest-paths tree T^* of H with start node s;
output T^*;

Theorem 6.2 *Let $\alpha > 1$ and $\beta = 1 + \frac{2}{\alpha-1}$ be two numbers. In the subgraph H,*

(i) for each node v, the cost of a shortest s-v path is at most $\alpha \cdot d(v)$, and

(ii) the total cost $c(H)$ is at most $\beta \cdot c(T_M)$.

Proof: Step 3 of the algorithm ensures that part (i) holds. We prove part (ii). Let $z_0 = s$, and let $z_1, z_2, \ldots, z_k$ be the preorder sequence of vertices that caused the edges of shortest paths to be added to H in Step 3. When z_i $(i = 1, \ldots, k)$ is examined in Step 3, H contains the s-z_i walk formed by taking the last path P_{i-1} (from s to z_{i-1}) added to H followed by the path in T_M from z_{i-1} to z_i, and the cost of this walk is

$$d(z_{i-1}) + d_{T_M}(z_{i-1}, z_i)$$

Since the shortest s-z_i path P_i in H has at most this cost, and since $c(P_i) > \alpha \cdot d(z_i)$ (from Step 3), we have

$$d(z_{i-1}) + d_{T_M}(z_{i-1}, z_i) > \alpha \cdot d(z_i)$$

or

$$\alpha \cdot d(z_i) - d(z_{i-1}) < d_{T_M}(z_{i-1}, z_i).$$

Summing over all $i = 1, \ldots, k$, and noting that $d(z_0) = 0$, we have

$$\begin{array}{lcr}
\alpha \cdot d(z_1) - d(z_0) & < & d_{T_M}(z_0, z_1) \\
\alpha \cdot d(z_2) - d(z_1) & < & d_{T_M}(z_1, z_2) \\
\alpha \cdot d(z_k) - d(z_{k-1}) & < & d_{T_M}(z_{k-1}, z_k) \\
\hline
\alpha \cdot d(z_k) + \sum_{i=1}^{k-1}(\alpha - 1)d(z_i) & < & \sum_{i=1}^{k} d_{T_M}(z_{i-1}, z_i),
\end{array}$$

or $\displaystyle\sum_{i=1}^{k}(\alpha - 1)d(z_i) < \sum_{i=1}^{k} d_{T_M}(z_{i-1}, z_i)$. From the previous lemma, the right-hand side is at most $2\ c(T_M)$, hence, $\displaystyle\sum_{i=1}^{k} d(z_i) < \frac{2\ c(T_M)}{\alpha - 1}$. The proof is complete since $c(H) = c(T_M) + \displaystyle\sum_{i=1}^{k} d(z_i) < \left(1 + \frac{2}{\alpha - 1}\right) c(T_M)$. ∎

6.4 Exercises

1. Construct a family of networks G, c (one for each $|V(G)|$) and a root node $s \in V(G)$ such that given numbers α and β as in the algorithm, and any positive number ϵ, the cost of the spanning tree output by the (α, β) LAST algorithm is at least

$$(1 + \frac{2}{\alpha - 1} - \epsilon)c(T_M),$$

   where $c(T_M)$ is the cost of a minimum spanning tree of G, c.

2. A breadth-first-search (BFS) ordering of a tree with root s is obtained by doing a BFS traversal of the tree, and numbering the nodes in the order in which they are visited. Alternatively, it is the numbering obtained by setting the number of s to 1, then ordering the nodes in a "level" of the tree according to the number of their parents, and then numbering the nodes

in sequence. Prove or disprove:
Suppose that Step 3 of the (α, β) LAST algorithm traverses the minimum spanning tree T_M in BFS order (instead of using preorder), then the subgraph H has cost $c(H)$ at most $(1 + \frac{2}{\alpha-1})c(T_M)$.

Bibliography

[1] I. Althöfer, G. Das, D. Dobkin, D. Joseph and J. Soares, *On sparse spanners of weighted graphs.* Discrete and Computational Geometry, (1992).

[2] B. Awerbuch, and Y. Azar, *Buy-at-bulk network design approximation algorithms.* Proc. 38th Annual IEEE Sympos. on Foundations of Comput. Sci. (FOCS) 1997.

[3] B. Awerbuch, A. Baratz, and D. Peleg, *Cost-sensitive analysis of communication protocols.* Proc. 9th Sympos. on Distributed Computing (PODC), (1990), pp. 177–187.

[4] S. Khuller, B. Raghavachari and N. E. Young, *Balancing minimum spanning and shortest path trees.* Algorithmica **14** (1995), pp. 305–322.

[5] Y. Mansour and D. Peleg, *An approximation algorithm for minimum-cost network design.* The Weizman Institute of Science, Rehovot, 76100 Israel, Tech. Report CS94-22, 10 pages.

[6] F. S. Salman, J. Cheriyan, R. Ravi, and S. Subramanian, *Buy-at-bulk network design: approximating the single-sink edge installation problem.* Proc. 8th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA-1997).