DIMACS Technical Report 94-56
December 1994

A Linear-Time Algorithm for Network Decomposition

by

Lenore J. Cowen[1,2]
Department of Mathematical Sciences
Johns Hopkins University
Baltimore, MD 21218

[1]DIMACS postdoctoral visitor, 1994.
[2]Visit supported by an NSF Postdoctoral Fellowship.

DIMACS is a cooperative project of Rutgers University, Princeton University, AT&T Bell Laboratories and Bellcore.

DIMACS is an NSF Science and Technology Center, funded under contract STC–91–19999; and also receives support from the New Jersey Commission on Science and Technology.

ABSTRACT

An algorithm for low-diameter network decomposition on arbitrary graphs is presented which runs in $O(E+n)$ time. Previous algorithms ran in $O(E \log n + n)$ time.
Keywords: algorithms, data structures, greedy, low diameter, clusters.

1 Introduction

For an undirected graph $G = (V, E)$, with $|V| = n$ and $|E| = E$, a (χ, d)-*decomposition* is defined to be a χ-coloring of the nodes of the graph that satisfies the following properties:

1. each color class is partitioned into an arbitrary number of disjoint *clusters;*
2. the distance between any pair of nodes in a cluster is at most d, where *distance* is the length of the shortest path connecting the nodes in G,
3. clusters of the same color are at least distance 2 apart.

A (χ, d)-decomposition is said to be *low-diameter* if χ and d are both $O(\log n)$.

The (χ, d)- decomposition problem, most commonly called *network decomposition* was introduced in [4] as a means of partitioning a network into local regions, though they did not achieve clusters that were low-diameter by the standards of the above definition (They produced χ and d both $O(n^\epsilon)$, for $\epsilon = O(\sqrt{\log\log n}/\sqrt{\log n})$.)

In [5, 6] a simple greedy algorithm was presented that produced a low-diameter network decomposition in $O(E \log n + n)$ time. (In addition, [6] proved that the low-diameter condition was best possible: there exist families of graphs for which to minimize χ and d simultaneously, require $\chi, d = O(\log n)$.)

Further work on network decomposition focused on the construction of low-diameter network decompositions, and applications, in the parallel and distributed models of computation (see [1, 2, 3, 5, 6, 7].)

In this note, it is shown how to modify the simple greedy construction in [6] to save a $\log n$ factor in running time. The new algorithm runs in $O(E + n)$ time, which is optimal. Note that the new construction will produce clusters whose diameter can be larger by a small constant factor than those in [6], so in some applications where network decomposition is used as a data structure, the original algorithm might be preferable in practice, even though it is more costly.

2 The Algorithm

The new algorithm works as follows. Pick a color. Pick an arbitrary vertex (call it a *center* vertex) and in a BFS fashion, greedily grow a ball around it of minimum radius r, so that (a) at least half the vertices in the ball of radius r are in the interior, (i.e. are also in the ball of radius $r - 1$ around the center node), AND (b) at least half the edges which are adjacent to a vertex in the ball of radius r, have an endpoint within the ball of radius $r - 1$. (This is the modification of [6]). The interior of the ball is put into the color class, and the entire ball is removed from the graph. (The *border* (those nodes whose distance from the center vertex is exactly r) will not be colored with the current color). Then pick another arbitrary vertex, and do the same thing, until all nodes in the graph have been processed. Then return all the uncolored vertices (the border vertices) and the edges between them to the graph, and begin again on a new color. Stop when all vertices have been colored.

3 Analysis

First it is shown that the above procedure produces a low-diameter network decomposition. Let G be an arbitrary graph with n vertices and E edges. In what follows, all logarithms are assumed to be to the base 2.

Lemma 1 *The diameter of any cluster will be at most* $6 \log n + 2$.

Proof. Each time condition (a) fails to be met, the number of nodes in the interior doubles. This can happen at most $\log n$ times, before running out of nodes in the graph. Each time condition (b) fails to be met, the number of edges in the interior doubles. Since there are at most n^2 edges, this can happen at most $\log n^2 = 2 \log n$ times. Clearly the worst case is whenever (a) is met, (b) is violated, and visa versa. But this can account for at most $3 \log n$ consecutive radii at which either (a) or (b) is violated. Thus the conditions (a) and (b) will both be met simultaneously for some r, $1 \leq r \leq 3 \log n + 1$. Therefore the radius of any ball will be at most $3 \log n + 1$. □

Lemma 2 *Clusters of the same color are at least distance 2 apart.*

Proof. Follows immediately from the construction. After a cluster is colored, all things of distance 1 from any node in the cluster are placed in the border and removed from the graph, so they will not receive the same color. The borders serve as buffers and prevent two monochromatic clusters from touching. □

Lemma 3 *The number of colors used will be at most* $\log n$.

Proof. Condition (a) insures that more than half the remaining nodes are in the interior of a cluster, rather than in a border. Since in a given iteration (color), all uncolored nodes are placed either in an interior or a border, at least half the remaining uncolored nodes will be colored with each new color. □

It is now easy to see the algorithm runs in linear time as follows. For each color class, the algorithm does a BFS exploration of the remaining graph. It looks at each vertex and edge precisely once. It keeps a count of all vertices and edges it has seen so far in the interior, and counts the vertices and edges it sees in the border, and then compares. If it grows an additional level, it adds the border count to the interior count, and makes a new border count. Since each vertex and edge appears at most once in the interior count, the border count, and the BFS, the algorithm takes $O(E + n)$ time to construct all the clusters of a given color.

However, notice that the conditions (a) and (b) insure that the number of vertices and edges in the graph decrease by a factor of two, in each iteration. Therefore, the running time over all colors is a small constant times

$$(E + n) + (E + n)/2 + (E + n)/4 + \ldots$$

This proves:

Theorem 1 *Given a graph G with n vertices and E edges, the above algorithm produces a low-diameter network decomposition in $O(E + n)$ time.*

References

[1] Y. Afek and M. Riklin. Sparser: A paradigm for running distributed algorithms. *J. of Algorithms*, 1991.

[2] B. Awerbuch, B. Berger, L. Cowen, and D. Peleg. Fast distributed network decomposition. In *Proc. 11th ACM Symp. on Principles of Distributed Computing*, Aug. 1992.

[3] B. Awerbuch, B. Berger, L. Cowen, and D. Peleg. Low-diamter graph decomposition is in NC. In *Proc. 3'rd Scandinavian Workshop on Algorithm Theory*, July 1992.

[4] B. Awerbuch, A. Goldberg, M. Luby, and S. Plotkin. Network decomposition and locality in distributed computation. In *Proc. 30th IEEE Symp. on Foundations of Computer Science*, May 1989.

[5] B. Awerbuch and D. Peleg. Sparse partitions. In *Proc. 31st IEEE Symp. on Foundations of Computer Science*, pages 503–513, 1990.

[6] N. Linial and M. Saks. Decomposing graphs into regions of small diameter. In *Proc. 2nd ACM-SIAM Symp. on Discrete Algorithms*, pages 320–330. ACM/SIAM, Jan. 1991.

[7] A. Panconesi and A. Srinivasan. Improved distributed algorithms for coloring and network decomposition problems. In *Proc. 24th ACM Symp. on Theory of Computing*, pages 581–592, 1992.