COMBINATORICA
Akadémiai Kiadó – Springer-Verlag

COMBINATORICA 13 (4) (1993) 441–454

# LOW DIAMETER GRAPH DECOMPOSITIONS*

NATHAN LINIAL** and MICHAEL SAKS†

*Received October 6, 1990*

A *decomposition* of a graph $G=(V,E)$ is a partition of the vertex set into subsets (called *blocks*). The *diameter* of a decomposition is the least $d$ such that any two vertices belonging to the same connected component of a block are at distance $\leq d$. In this paper we prove (nearly best possible) statements of the form: Any $n$-vertex graph has a decomposition into a small number of blocks each having small diameter. Such decompositions provide a tool for efficiently decentralizing distributed computations. In [4] it was shown that every graph has a decomposition into at most $s(n)$ blocks of diameter at most $s(n)$ for $s(n)=n^{O(\sqrt{\log\log n/\log n})}$. Using a technique of Awerbuch [3] and Awerbuch and Peleg [5], we improve this result by showing that every graph has a decomposition of diameter $O(\log n)$ into $O(\log n)$ blocks. In addition, we give a randomized distributed algorithm that produces such a decomposition and runs in time $O(\log^2 n)$. The construction can be parameterized to provide decompositions that trade-off between the number of blocks and the diameter. We show that this trade-off is nearly best possible for two families of graphs: the first consists of skeletons of certain triangulations of a simplex and the second consists of grid graphs with added diagonals. The proofs in both cases rely on basic results in combinatorial topology, Sperner's lemma for the first class and Tucker's lemma for the second.

## 1. Introduction

In this paper, we investigate a problem in algorithmic graph theory that originated in the theory of distributed computing. The systems we are concerned with can be modeled as graphs whose nodes correspond to processors and whose links correspond to communication channels between certain processors. One of the basic difficulties in designing algorithms for such systems is determining the extent to which the actions of the processors must be coordinated, and accomplishing this coordination as efficiently as possible. The most naive approach is to centralize the network operation by appointing one of the processors as a coordinator for the whole network and having all processes act under the direction of the coordinator. Centralization has several advantages; it often simplifies the problem considerably and facilitates the development of distributed algorithms based on known serial algorithms. On the other hand, rigid centralization often degrades system performance because of delays in communication between the coordinator and the other

AMS subject classification codes (1991): 05 C 12, 05 C 15, 05 C 35, 05 C 70, 05 C 85, 68 Q 22, 68 R 10

* A preliminary version of this paper appeared as "Decomposing Graphs into Regions of Small Diameter" in Proc. 2nd ACM–SIAM Symposium on Discrete Algorithms (1991) 321–330.

** This work was supported in part by NSF grant DMS87-03541 and by a grant from the Israel Academy of Science.

† This work was supported in part by NSF grant DMS87-03541 and CCR89-11388.

processors of the system. This problem is particularly significant in networks with large diameter and non-negligible message transmission time.

For these reasons, considerable amounts of research in distributed algorithms has focused on finding ways to decentralize distributed computation. For many problems it is possible to design algorithms in which each node acts with knowledge only of the activity of nearby nodes, and these "local" activities combine together to produce a global solution to the problem. The extent to which this is possible is referred to informally as the *locality* of the problem. Exploiting locality for specific problems leads to algorithms that are among the most novel and interesting in the area, for example, the beautiful symmetry breaking techniques of Cole and Vishkin ([8]) which have been used for graph coloring ([11]). Limitations on locality were addressed in [13].

This leads to the following general problem: find techniques for the design of distributed network algorithms that can be used to exploit locality. One class of techniques that has been proposed involves partitioning the network into regions of small diameter, coordinating action in each region through a local coordinator and combining the partial solutions together. This natural methodology has been considered by a number of authors; its earliest explicit statement known to us is in Awerbuch's [3] analysis of time and communication trade-offs required to achieve network synchronization. In [4] it is further used to improve the time complexity of distributed deterministic maximal independent set (MIS) algorithms, for graph coloring and distributed breadth first search. More recently, the approach has been applied to distributed routing [5] for a distributed all-pairs-shortest-distance algorithm and other similar problems [1].

The above discussion motivates the following graph definitions:

Let $G$ be a graph. A subset $W$ of vertices will be called a *block* of $G$. The *strong diameter* of a block $W$, $SD(W)$ is the maximum diameter of any connected component of the graph $G_W$ induced on $W$. The *weak diameter* $WD(W)$ is the maximum distance in $G$ between two vertices of $W$ that belong to the same connected component of $G_W$. (The difference between strong and weak diameter is that when computing weak diameter we are allowed to shortcut through vertices not in $W$ and thus $WD(W) \leq SD(W)$.) A partition $\Pi$ of the vertex set of a graph $G$ into $\lambda$ disjoint blocks is called a *$\lambda$-decomposition* of $G$. The *strong diameter* $SD(\Pi)$ (*weak diameter* $WD(\Pi)$) of $\Pi$ is the maximum strong diameter (weak diameter) of any of its blocks.

For a given graph we are interested in finding decompositions into a small number of (possibly disconnected) blocks each of small (strong or weak) diameter. This problem was introduced (with somewhat different terminology) in [4] as an attempt to solve a major outstanding problem in the theory of distributed algorithms: is there a deterministic algorithm for finding a maximal independent set in a distributed network that runs in polylog time? There are various partial results known for this tantalizing problem: a randomized distributed algorithm that runs in expected polylog time (Luby [14] and Alon *et al.* [2]), and a deterministic polylog time algorithm for bounded degree graphs (Goldberg *et al.* [11]). It was noted in [4] that given a $\lambda$-decomposition of the graph, an MIS can be constructed in a sequence of $\lambda$ rounds each taking time $O(D\,\text{polylog}(n))$ where $D$ is the strong diameter, through the following iterative procedure: after $i$ iterations there will be

an MIS for the subgraph of vertices in the first $i$ blocks of the decomposition. During round $i+1$, the vertices in block $i+1$ that have no neighbors in the current independent set try to find an MIS among themselves. This can be done separately by each connected component of this subgraph, and the information about the connected component can be collected at one node in time $O(D)$ where $D$ is the strong diameter of the decomposition. In fact, it is easy to see that $D$ can be taken to be the weak diameter of the decomposition as well.

In [4], it was shown that for some $k = n^{O(\sqrt{\log\log n/\log n})}$, every graph has a $k$-decomposition with strong diameter at most $O(k)$. Furthermore, they gave a distributed algorithm running in time $O(k)$ for finding this decomposition. This enabled them to construct the fastest known deterministic distributed algorithm for the maximal independent set problem, which runs in time $n^{O(\sqrt{\log\log n/\log n})}$, (which is $o(n^\epsilon)$ for any positive $\epsilon$, but bigger than any polylog$(n)$).

The obvious questions that come out of their work are:

1. What are the trade-offs between the number of blocks $\lambda$ of a decomposition and its strong (or weak) diameter? In particular, what is the smallest $k(n)$ such that every graph on $n$ vertices has a $k(n)$-decomposition with weak (or strong) diameter $k(n)$?

2. What is the smallest $j(n)$ such that there is a deterministic distributed algorithm that runs on any $n$-vertex graph in polylog$(n)$ time and produces a $j(n)$-decomposition with weak diameter $j(n)$.

Of course, the functions $j(n)$ and $k(n)$ answering these questions satisfy $j(n) \geq k(n)$. What we'd like is that $j(n)$ is $O(\text{polylog}(n))$, in which case we could use the approach of [4] to get an MIS algorithm with the desired time complexity.

As it happens, it is not hard to show that $k(n) = O(\log n)$, using a simple but powerful constructive technique that was introduced by Awerbuch ([3]) and modified by Awerbuch and Peleg ([5]) to solve some graph covering theorems closely related to the decomposition problem. This technique provides a sequential way to build the decomposition one block at a time and enables us to give a deterministic sequential algorithm that, for any $n$-vertex graph $G$ and any $p \in (0,1)$ produces a $\lambda$-decomposition with $\lambda < \frac{\log n}{\log(1/(1-p))}$ and strong (and hence also weak) diameter at most $\frac{2\log n}{\log(1/p)}$. For $\lambda \leq \log n$, this implies that the decomposition has diameter at most $2n^{1/\lambda}\log n$. In particular, the function $k(n)$ referred to in the first question above is $O(\log n)$.

Here and elsewhere, all logarithms are in base 2.

On the other hand, we can show that the trade-off given by this construction is nearly tight; precisely we show that for each $\lambda = O(\log n)$ there is a graph $S$ with at most $n$ vertices such that any $\lambda$-decomposition has weak diameter (and hence strong diameter) at least $\Omega(n^{1/\lambda})$. The graph $S$ is simple: Choose $m$ to be the greatest integer such that $\binom{m+\lambda}{\lambda} \leq n$. The vertex set of $S$ consists of all vectors $\bar{x} = (x_0, \ldots, x_\lambda)$ where $\sum x_i = m$ and all $x_i$ are nonnegative integers. Vertices $\bar{x}, \bar{y}$ are adjacent iff $\bar{x} - \bar{y} \in \{-1, 0, 1\}^{\lambda+1}$. (This graph contains as a spanning subgraph the one-skeleton of the standard triangulation of the $\lambda$-dimensional simplex). The

lower bound on the diameter of any $\lambda$-decomposition is deduced from a well known lemma in combinatorial topology due to Sperner. For $\lambda$ substantially greater than $\log n$, it is easy to see that the upper bound is tight for expander graphs with the appropriate parameters. Another class of graphs where the tradeoff cannot be improved has vertices all integer vectors $\bar{x}=(x_1,\ldots,x_\lambda)$ where $\bar{x},\bar{y}$ are adjacent iff their difference $\bar{x}-\bar{y}$ is in either $\{0,1\}^\lambda$ or $\{0,-1\}^\lambda$. The proof depends this time on Tucker's lemma in combinatorial topology. While this paper was being refereed, we discovered that the relevant property of these graphs was proved previously by Gale [10] (with a different proof) in the analysis of certain combinatorial games.

Most interesting from the point of view of distributed computation, we give a randomized distributed algorithm that runs in expected time $O(n^{1/\lambda}\text{polylog}(n))$ and produces a $\lambda$-decomposition with weak diameter $O(n^{1/\lambda})$ and degree $\leq\lambda$. We note that we do not know how to make a similar guarantee on the strong diameter.

Our results fall short of the original goal; for this we would need to replace the randomized algorithm by a deterministic one. We believe that careful study of the randomized algorithm may lead to a deterministic one, or alternatively, suggest why no deterministic algorithm is possible. Furthermore, as discussed in [4] and [5], low diameter decompositions have a variety of potential applications and a fast randomized distributed algorithm is itself a potentially useful tool for replacing serialization by randomness in distributed computing. For instance, in [3], a graph decomposition is used as a data structure for providing a way to force synchrony in an asynchronous network. The construction given there is serial, and our randomized algorithm can be used to give a fast distributed construction. (Awerbuch [personal communication] has an alternative randomized algorithm with similar properties).

## 2. The existence of low diameter $\lambda$-decompositions

The first result of this section is:

**Theorem 2.1.** *Let $p$ be a real number between 0 and 1, $G$ an $n$-vertex graph and $\lambda=\frac{\log n}{\log(1/(1-p))}$. Then there is a $\lambda$-decomposition of $G$ with strong diameter at most $\frac{2\log n}{\log(1/p)}$.*

By analyzing the above theorem for the two ranges of $\lambda$ depending on whether it is less than or greater than $\log n$ we obtain the following corollaries:

**Corollary 2.1.** *For $\lambda\leq\log n$, every $n$ vertex graph has a $\lambda$-decomposition with strong diameter at most $2n^{1/\lambda}\log n$.*

**Corollary 2.2.** *For $\lambda\geq\log n$, every $n$ vertex graph has a $\lambda$-decomposition with strong diameter at most $\frac{2\log n}{1+\log\lambda-\log\log n}$.*

In particular, when $p=1/2$ we get $\lambda\leq\log n$ and strong diameter at most $2\log n$.

The proof of theorem 2.1 is based on a technique of Awerbuch and is implicit in the work of Awerbuch and Peleg [5].

**Proof.** For two vertices $x,y$ in a graph $H$ let $d(x,y)=d_H(x,y)$ be their distance in $H$. For an integer $r$ let $B_r(x)$ be the ball of radius $r$ around $x$ in $H$, i.e., $\{y\in V(H): d_H(x,y)\le r\}$. Whenever the need arises the graph $H$ will be mentioned explicitly.

Fix $p$ between 0 and 1 and call an integer $r$ *a safe radius* for a vertex $x$ if $p|B_r(x)| < |B_{r-1}(x)|$. Note that if $1,\ldots,r$ are all unsafe for $x$, then necessarily $|B_j(x)|\ge(1/p)^j$ for all $1\le j\le r$, and in particular $n\ge(1/p)^r$. In other words, every vertex $x$ has a safe radius not exceeding $\frac{\log n}{\log(1/p)}$.

We construct a $\lambda$-decomposition one block at a time. The construction of $V_1$ is performed iteratively. Pick any vertex $x_1$ in $G_1 = G$ and let $r_1$ be the smallest safe radius for $x_1$. Add all of the vertices in $B_{r_1-1}(x_1)$ to $V_1$ and define $G_2$ to be the graph obtained by deleting $B_{r_1}(x_1)$ from $G_1$. Select $x_2$ in $G_2$ and let $r_2$ be the smallest safe radius for $x_2$ (in the graph $G_2$). Add the vertices of $B_{r_2-1}(x_2)$ (restricted to $G_2$) to $V_1$ and define $G_3$ to be $G_2\setminus B_{r_2}(x_2)$. Continue this process constructing sequences $\{x_i\}$ of vertices $\{r_i\}$ of radii and $\{G_i\}$ of graphs and enlarging $V_1$ at each stage. The construction of $V_1$ is complete when $G_i$ is empty.

The blocks $V_i$, for $i > 1$ are constructed inductively. Having constructed $V_1,\ldots,V_{i-1}$ define $G^i = G\setminus(V_1\cup V_2\cup\ldots\cup V_{i-1})$ and apply the above process to $G^i$ to obtain $V_i$. Continue until all vertices belong to some $V_i$.

The construction of the first (and each subsequent) block guarantees that its strong diameter is at most twice the largest radius of any of the selected balls. Since these radii are bounded by $\frac{\log n}{\log(1/p)}$ we obtain:

$$SD(\Pi)\le 2\frac{\log n}{\log(1/p)}.$$

Since the ratio of $|B_{r_i-1}(x_i)|$ to $|B_{r_i}|$ is always at least $p$ for each selected ball, the fraction of vertices of $G^i$ that are not assigned to $V_i$ is at most $1-p$, and the number of vertices in $G^{i+1}$ is at most $1-p$ times the number in $G^i$. Thus the number of blocks of the decomposition is bounded as follows:

$$\lambda\le\frac{\log n}{\log(1/(1-p))}.$$

Note that the above proof provides a polynomial time sequential algorithm for constructing the decomposition.

## 3. Tightness of the existence theorem

Our existence theorem cannot be significantly improved for $\lambda>\log n$. A graph of chromatic number $k$ and girth $g$ cannot have a $((k-1)/2)$-decomposition of diameter less than $\left[\frac{g}{2}\right]$, since in such a decomposition each block would be a forest and thus, by two coloring each block we would have a $k-1$ coloring of the graph. Since graphs of chromatic number $k$ and girth $g=\Omega(\log n/\log k)$ exist for $k=O(\log n)$ (see [7]) this shows that Corollary 2.2 is essentially tight.

### 3.1 Simplex graphs

To show that Corollary 2.1 is nearly tight requires a more substantial argument, and this is done in the following theorem:

**Theorem 3.1.** *For any integers $n$ and $\lambda$, there exists a graph $G$ with at most $n$ vertices such that any $\lambda$-decomposition of $G$ has weak (and strong) diameter at least $\frac{1}{e}n^{1/\lambda}-1$.*

We define a family of graphs $S(m,\lambda)$, for positive integers $m$ and $\lambda$. The vertices of $S(m,\lambda)$ are all vectors $\bar{x}=(x_0,\ldots,x_\lambda)$ where $x_i$ are nonnegative integers and $\sum x_i = m$. Vertices $\bar{x}$ and $\bar{y}$ are adjacent iff all coordinates in $\bar{x}-\bar{y}$ are in $\{-1,0,1\}$.

**Lemma 3.1.** *Any $\lambda$-decomposition of $S(m,\lambda)$ has weak diameter at least $m/\lambda$.*

To deduce Theorem 3.1 from this lemma, note first that $S(m,\lambda)$ has $\binom{m+\lambda}{\lambda}$ vertices. Thus given $n$ and $\lambda$ select $m$ to be the integer such that $\binom{m+\lambda-1}{\lambda} \le n < \binom{m+\lambda}{\lambda}$. Using the elementary inequality $\binom{a}{b} \le (\frac{e\cdot a}{b})^b$ (which can be proved by induction on $b$), we have $n \le (\frac{e(m+\lambda)}{\lambda})^\lambda$. Thus $\frac{m}{\lambda} \ge \frac{n^{1/\lambda}}{e} - 1$, as needed.

Thus it remains to prove the lemma, which we will do by some basic geometric-topological considerations. We first recall some basic definitions and facts (see for instance, [16], Chapter 1.) A set of points in Euclidean space is in *convex position* if no one of the points is in the convex hull of the others. The convex hull of a set $P$ of $d+1$ affinely independent points in convex position is called a closed $d$-dimensional *simplex*. The *relative interior* of a simplex consists of all points of the simplex that are not in the convex hull of any proper subset of the extreme points. If $Y$ is a subset of Euclidean space, then a *triangulation* of $Y$ is a collection of closed simplices whose union is $Y$ and whose relative interiors are disjoint. The set of extreme points of the simplices of the triangulations are the *vertices* of the triangulation. The *one-skeleton* of a triangulation is the graph defined on the vertex set of the triangulation with two vertices adjacent if they lie in a common simplex of the triangulation.

Fix $\lambda$ and $m$, and let $X$ be the $\lambda$-dimensional simplex embedded in $\lambda+1$ dimensional real space consisting of all vectors with nonnegative coordinates that sum to $m$. $X$ has $\lambda+1$ extreme points, the *$i$-th vertex of $X$* has $x_i = m$ and all other coordinates 0. The facet $\{x \in X : x_i = 0\}$ opposite to the $i$-th vertex is called the *$i$-th facet of $X$*.

**Lemma 3.2.** *There is a triangulation of $X$ whose one-skeleton is a spanning subgraph of $S(m,\lambda)$.*

The triangulation referred to in the lemma is derived from a standard triangulation of $R^\lambda$. For each vector $\bar{y}$ in $\mathbb{R}^\lambda$ having integer coordinates and each permutation $\sigma$ of $\{1,2,\ldots,\lambda\}$ define a simplex $\delta(\bar{y},\sigma)$ whose vertices are $\bar{y}^0,\bar{y}^1,\ldots,\bar{y}^\lambda$ with $\bar{y}^0=\bar{y}$ and $\bar{y}^i=\bar{y}^{i-1}+\bar{e}_{\sigma(i)}$, where $\bar{e}_j$ is the unit vector in the $j$-th direction. It is well known that the set $\Gamma$ of all such simplices gives a triangulation of $\mathbb{R}^\lambda$ (see [16], pages 30–31). Furthermore, if we let $Z$ denote the simplex $\{(x_1,x_2,\ldots,x_\lambda) : 0 \le x_1 \le x_2 \le \ldots \le x_\lambda \le m\}$ for some positive integer $m$, then any member of $\Gamma$

that contains an interior point of $Z$ is contained in $Z$, and thus the set of simplices of $\Gamma$ that are contained in $Z$ is a triangulation $\Gamma_Z$ of $Z$. Now consider the affine linear transformation from $\mathbb{R}^\lambda$ to $\mathbb{R}^{1+\lambda}$ that takes a vector $(a_1, a_2, \ldots a_\lambda)$ to $(a_1, a_2 - a_1, a_3 - a_2, \ldots, a_\lambda - a_{\lambda-1}, m - a_\lambda)$. This map defines a bijection between $Z$ and the simplex $X$, and the image of $\Gamma_Z$ under the map defines a triangulation $\Upsilon$ of $X$ whose vertices are the vertices of $S(m,\lambda)$. By definition, if two such vertices $\bar{x}$ and $\bar{y}$ are adjacent in the 1-skeleton of $\Upsilon$ then they lie in a common simplex of $\Upsilon$. This occurs if and only if the set $\{\sum_{i=1}^{j}(x_i - y_i) \ : \ 1 \le j \le \lambda\}$ is either $\{0,1\}$ or $\{0,-1\}$, and thus the coordinates of $\bar{x} - \bar{y}$ are from the set $\{-1,0,1\}$, i.e., $\bar{x}$ and $\bar{y}$ are adjacent in $S(m,\lambda)$. Thus the 1-skeleton of $\Upsilon$ is a subgraph of $S(m,\lambda)$ as required.

Next, we need the following fundamental lemma of Sperner:

**Lemma 3.3.** *Let the $\lambda$-dimensional simplex $X$ be triangulated. Let the vertices of the triangulation be assigned labels from $\{0,\ldots,\lambda\}$ so that vertex $i$ is labeled by $i$ and no vertex lying in the $i$-th facet is labeled $i$. Then there is a simplex in the triangulation all vertices of which are labeled differently (a full-colored simplex).*

The derivation of Lemma 3.1 is easy now: Consider any decomposition $\Pi$ of $S(m,\lambda)$ into $\lambda$ blocks.

**Claim:** Some block has a connected component that meets all facets.

Suppose not. Define a labeling of the vertices: each vertex is labeled by the least $i$ such that the connected component of $\Pi$ that contains it misses facet $i$. This labeling satisfies the conditions of Sperner's lemma with respect to the triangulation referred to in Lemma 3.2, and thus there is a full-colored simplex, whose $\lambda+1$ vertices form a clique in $S(m,\lambda)$. Since $\Pi$ has only $\lambda$ blocks, two of the vertices in this clique are in the same block, and since they are joined by an edge, they are assigned the same label, contradicting the fact that the simplex is full-colored.

So there is a connected component of $\Pi$ that meets all facets of $X$. Let $\bar{z}$ be a vertex of the component that lies on the 0-th facet. Since the coordinates sum to $m$, for some $j > 0, z_j$ is at least $m/\lambda$. There is a point $\bar{y}$ of the component that lies on the $j$-th facet and the distance in $S(m,\lambda)$ from $\bar{z}$ to any such point is at least $z_j$.

**Remarks:**

1. We do not know that Lemma 3.1 is tight; it is possible that $m/\lambda$ could be replaced by something as large as $\Omega(m)$, which would improve the lower bound of the theorem by a factor of $\lambda$.
2. Note that the proof implies a slightly stronger result, namely, when the vertices of $S(m,\lambda)$ are decomposed into connected regions of diameter $< \frac{n^{1/\lambda}}{e} - 1$, there will be $\lambda+1$ of these regions any two of which are adjacent (i.e., for regions $R_1, R_2$ there are neighboring vertices $x_1, x_2$ with $x_1 \in R_1, x_2 \in R_2$.) This readily implies that no proper $\lambda$ coloring of the regions is possible, as claimed.

### 3.2 Grid graphs

In what follows we exhibit another class of graphs for which the trade-off in the existence theorem is tight. These are obtained from the grid graph in real space by adding some diagonals. The property of grid graphs given by the following theorem

was first discovered by Gale (see [10]) in a different context; we reprove his result in a somewhat different way. For another recent application, see [12].

Consider the graph $C(m,k)$ ($m,k$ integers) with vertex set $\{1,2,\ldots,m\}^k$ where $(x_1,\ldots,x_k)$ is a neighbor of $(y_1,\ldots,y_k)$ iff $x_i - y_i \in \{0,1\}$ for every $i$ or $x_i - y_i \in \{0,-1\}$ for every $i$. We show:

**Theorem 3.2.** *Let $m \geq 3$ and $k \geq 2$ be integers. Then any $k$-decomposition of $C(m,k)$ has weak diameter (and hence also strong diameter) at least $m$.*

In what follows we view a $k$-decomposition as a coloring of the vertices of $C = C(m,k)$ by $k$ colors. Say that color $c$ *spans* dimension $i$ if there is a path all vertices of which are colored $c$, where one endpoint satisfies $x_i = 1$ and the other $x_i = m$. Theorem 3.2 follows easily from:

**Lemma 3.4.** *Let the vertices of $C = C(m,k)$ be colored with $k$ colors. Then there is an index $i$ such that color $i$ spans dimension $i$.*

The proof of this lemma makes use of a lemma of Tucker in combinatorial topology. (See [17] for an accessible proof of the general statement of the lemma and [9] for a history of the lemma and its applications.) Here one is concerned with triangulations of the $k$-dimensional cube $X = [-1,1]^k$, and not the simplex, as in Sperner's lemma. As usual a point $x \in X$ is said to be a boundary point if it has at least one coordinate which equals either $-1$ or 1. A triangulation $\Upsilon$ of $X$ is said to be *face antipodal*, if whenever $x$ is a vertex of $\Upsilon$ and $x$ is boundary point, then $-x$ is a vertex of $\Upsilon$ as well. In Tucker's lemma we will also be concerned with colorings of the graph (=1-skeleton) of triangulations. Given that $\Upsilon$ is a face-antipodal triangulation of $X$, and given a mapping $\phi$ from the vertices of $\Upsilon$ to the integers, we say that $(\Upsilon,\phi)$ is face-antipodal, if $\phi(-x) = -\phi(x)$ for every vertex $x$ which lies on the boundary of $X$. Here is the statement of Tucker's lemma:

**Lemma 3.5** (Tucker). *Let $X$ be the $k$-dimensional cube $[-1,1]^k$, let $\Upsilon$ be a triangulation of $X$ and let $\phi$ be a mapping from the vertices of $\Upsilon$ to $\{-k,\ldots,-1,1,\ldots,k\}$ such that $(\Upsilon,\phi)$ is face antipodal. Then there is an edge (1-dimensional simplex), say $[y,z]$ of $\Upsilon$ for which $\phi(y)+\phi(z)=0$.*

In keeping with the above terminology, we say that a vertex of $C = C(m,k)$ is a *boundary* vertex if at least one of its coordinates is in $\{1,m\}$, and that the vertex $y$ is *antipodal* to $x$ if $x_i + y_i = m+1$ for every $i$. The following is thus a special case of Tucker's lemma:

**Lemma 3.6.** *Let $f$ be a vertex coloring of $C = C(m,k)$ with colors from $\{-k,\ldots,-1,1,\ldots,k\}$ such that if $x$ is a boundary vertex and $y$ is antipodal to $x$, then $f(y) = -f(x)$. Then there exist two adjacent vertices $u,u'$ with $f(u') = -f(u)$.*

To derive Lemma 3.4 we argue as follows: Define a graph $\bar{C}$ on the vertex set $\{0,\ldots,m+1\}^k$ and the same adjacency relationship as $C$, viz. $x$ is adjacent to $x+e$ for every $e \in \{0,1\}^k$. Lemma 3.6 is to be applied to $\bar{C}$, with the understanding that being a boundary vertex means having a coordinate which equals either 0 or $m+1$, and that vertices $x$ and $y$ are antipodal in $\bar{C}$ if $x_i + y_i = m+1$ for all $i$. Now suppose we are given a vertex coloring of $C$ with colors in $\{1,\ldots,k\}$. A coloring of $\bar{C}$ by $\{-k,\ldots,-1,1,\ldots,k\}$ is induced as follows: For a boundary vertex $x$ of $\bar{C}$ let $j$ be

the first index for which $x_j \in \{0, m+1\}$. If $x_j = 0$, color $x$ by $-j$, and if $x_j = m+1$, color it $j$. Now each vertex $u$ with no 0 or $m+1$ is also a vertex of $C$ and thus has a color $j$; if there is a path consisting of $j$-colored vertices connecting $u$ to the set $\{x : x_j = 0\}$ then recolor $u$ by $-j$, otherwise $u$'s color remains unchanged.

Obviously, there cannot be two adjacent vertices colored $j$ and $-j$, because the rule for sign change calls for replacing the $j$ by $-j$. Lemma 3.6 may be invoked now to conclude that in this coloring of $C$ there are two antipodal vertices whose colors do not sum to zero. Therefore some boundary vertex must have been recolored in the above recoloring process, which means that for some $1 \leq j \leq d$ there is a path of $j$ colored vertices in $C$ connecting a vertex with $x_j = 1$ to one with $x_j = m$, as required. ∎

Keeping the dimension $k$ fixed and letting $m$ tend to infinity, the following geometric theorem is derived:

**Corollary 3.1.** *Let the $k$-dimensional cube be covered by $k$ closed sets. Then there is a connected component of one of these sets which meets two opposite facets of the cube.*

It would be reasonable to assume that this result is known to geometers/topologists, but we could not find it in the literature. If the closed sets are also assumed to be homotopically trivial, this corollary follows from the concept of category for topological spaces (cf. [15]).

## 4. A fast randomized distributed algorithm for constructing low diameter decompositions

### 4.1 The algorithm

We would like to replace the sequential algorithm of Section 2 by a fast distributed one. The sequential algorithm constructs the blocks iteratively, one at a time. The time that the algorithm takes is $O(\lambda t(n))$ where $t(n)$ is the worst case time for one iteration. In the algorithm as described $t(n)$ could be linear in $n$. In this section we show how to use randomization to replace the serial algorithm for the construction of a single block by a parallel distributed one which achieves the same trade-off as the serial algorithm between weak diameter and $\lambda$. The time to construct one block is reduced to $O(D)$ where $D$ is the weak diameter, and thus we achieve total running time of $O(D\lambda)$. In particular, when we balance $D$ and $\lambda$, we produce a $O(\log n)$-decomposition with weak diameter $O(\log n)$ that runs in $O(\log^2 n)$ time.

To construct a single block quickly, we'd like to parallelize the selection of safe radii. However, if we allow many vertices to choose a safe radius simultaneously the balls around the vertices may overlap significantly. In that case, there is no longer an obvious criterion for deciding which vertices are to be placed in the block in such a way that the resulting block is guaranteed both to have small weak diameter and to contain a substantial fraction of the vertices. It is this difficulty that we must resolve.

We make the usual assumption that each vertex $x$ has a unique integer $ID_x$. Of course, if such $ID$'s are not provided then each vertex can select an $ID$ uniformly

at random from, for example, $\{1,2,\ldots,n^2\}$ which guarantees that the $ID$'s are unique with high probability; the algorithm can be modified to work under this assumption.

The algorithm for selecting a block out of a graph $G$, is called *Construct_Block*. First each vertex $y$ selects an integer radius $r_y$ at random (according to a distribution (given below) that is approximately geometric). It then broadcasts $(ID_y, r_y)$ to all vertices that are within distance $r_y$ of it. After collecting all such messages from other vertices, each vertex $y$ selects the vertex $C(y)$ of highest $ID$ from among the vertices whose broadcast it received in the first round (including itself), and joins the current block if $d(y,C(y)) < r_{C(y)}$ (note that it is necessarily the case that $d(y,C(y)) \le r_{C(y)}$).

The distribution by which each vertex $x$ selects its radius $r_x$ is a truncated geometric distribution, which is defined in terms of two parameters, $p$ and $B$:

$$Pr(r_x = j) = p^j(1-p)$$

for $j = 0,\ldots,B-1$, and

$$Pr(r_x = B) = p^B.$$

While the above algorithm is conceptually quite simple, there are some subtleties involved in implementing the above algorithm efficiently in an asynchronous distributed network, i.e., accomplishing the broadcast in such a way that each vertex knows when it has received all the broadcasts that it will get and can thus select $C(y)$. This can be done using standard synchronization techniques, which are discussed in the final subsection of this section.

### 4.2 Proof of correctness

The key properties of this algorithm are summarized by:

**Lemma 4.1.** *Suppose Construct_Block is applied to a graph $G$ with at most $n$ vertices. Let $S$ be the set of vertices comprising the block selected. Then:*

1. *The set of selected vertices has weak diameter at most $2B$.*
2. *For each vertex $y$ of $G$, the probability that it belongs to $S$ is at least $p(1-p^B)^n$.*

If we apply Construct_Block iteratively to decompose the entire graph (using the same values of $p$ and $B$ at each iteration), then the first part of the lemma guarantees that the weak diameter of the resulting decomposition is at most $2B$. The second part of the lemma implies that if $q = p(1-p^B)^n$ then for each vertex $x$, the probability that $x$ is not assigned to one of the first $i$ blocks is at most $(1-q)^i$ and thus the probability that some vertex is unassigned after $i$ iterations is at most $n(1-q)^i$. By selecting $B$ to be $\frac{\log n + \omega(n)}{\log(1/p)}$ where $\omega$ is any function tending to infinity with $n$, it is easily verified that $q = p(1+o(1))$ where the lower order term depends only on the choice of $\omega$. Thus with high probability, the number of iterations (and hence the number of colors) does not exceed $(1+o(1))\frac{\log n}{\log(1/(1-p))}$. The result is a $\lambda$-decomposition with diameter $D$ where the expressions for $\lambda$ and $D$ in terms of $p$ and $n$ are essentially the same as obtained for the sequential algorithm, and therefore the trade-off is the same.

It remains to prove the lemma. The first part of the lemma follows easily from:

**Claim.** *For each connected subset $T$ of $S$, $C(y)$ is the same vertex $x$ for all $y \in T$.*

By the claim, any two vertices in $T$ are at distance at most $B$ from $x$ and thus at most $2B$ from each other.

We prove the claim by contradiction; if it is false then there are two adjacent vertices $y$ and $z$ belonging to $S$ with $C(y) \neq C(z)$. Without loss of generality, the ID of $C(y)$ exceeds that of $C(z)$. By the definition of $S$, $y$ is in $S$ implies $r_{C(y)} > d(C(y),y)$. Since $y$ and $z$ are neighbors, $r_{C(y)} \geq d(C(y),z)$ and thus $z$ received the broadcast sent by $C(y)$ in the first round. This contradicts the fact that $C(z)$ is the vertex of highest ID whose broadcast reached $z$.

We now proceed to the proof of the second part of the lemma. We fix a vertex $y$ and estimate the probability that it is assigned to $S$. We can bound this probability as follows:

$$Pr(y \in S) \geq \sum_{z|d(z,y)<B} Pr(y \in S|C(y) = z)Pr(C(y) = z)$$

For a given vertex $z$, define the following three events:

$D_z : r_z \geq d(z,y)$

$E_z : r_z > d(z,y)$

$F_z$: For every vertex $w$ with ID higher than $z$, $r_{C(w)} < d(w,y)$.

Then for $z$ such that $d(z,y) < B$ we can rewrite $Pr(y \in S|C(y) = z)$ as:

$$Pr(E_z \wedge F_z|D_z \wedge F_z) = Pr(E_z \wedge F_z)/Pr(D_z \wedge F_z) = Pr(E_z)/Pr(D_z) = p,$$

where the first equality follows from the fact that $E_z$ implies $D_z$, the second equality follows from the fact that $F_z$ is independent of both $D_z$ and $E_z$, and the third follows from the definition of the distribution on selected radii which implies that $Pr(E_z) = p^{d(z,y)+1}$ and $Pr(D_z) = p^{d(z,y)}$. Thus,

$$Pr(y \in S) \geq p \sum_{z|d(z,y)<B} Pr(C(y) = z) \geq pPr(d(C(y),y) < B) \geq$$
$$\geq pPr(r_z \neq B, \forall z) \geq p(1 - p^B)^n.$$

### 4.3 Remarks on the model and implementation

The above algorithm is correct under the assumption that all nodes of the algorithm begin at the same time and operate in a synchronous manner. In this section, we discuss some of the details required to implement the algorithm in situations where these assumptions are not valid.

In previous work on the maximal independent set problem and graph decompositions, the assumption has been that all nodes start at the same time. Since the motivation for studying this problem is to better understand what kinds of

distributed computations can be done locally, assuming that all nodes start at the same time is unsatisfying since it represents a global control of the network. On the other hand, without this assumption, it seems that all computations are "global", i.e. require time at least equal to the diameter $d$ of the network, in worst case. For example, if there is only one node that initiates the algorithm, this initiation message must propagate throughout the network and the computation will require at least $d$ (= diameter) time units just for every node to begin.

The way to resolve this difficulty is to consider an alternative measure of running time for a distributed algorithm in which nodes do not start at the same time. Rather than measuring the length of the time interval from the start of the algorithm until its completion, we believe that a better measure is the maximum over all nodes of the amount of time that the node is participating in the algorithm. Under this definition of time, the analysis of the previous section carries over to the case that nodes start at different times.

In addition, we want to consider the case that message transmissions are asynchronous. The assumption here is that there is only an upper bound on the time for any message to traverse a link and the running time is expressed as a multiple of this upper bound. Implementing the algorithm in this case requires that each vertex $y$ send $(ID_y, r_y)$ to every vertex $z$ that is within distance $r_y$ of $y$ and also that each vertex $y$ be able to detect that it has received all such messages that are intended for it (so that it can correctly select $C(y)$). This can be accomplished using a standard synchronization technique as outlined below. The resulting algorithm has the disadvantage that it requires that nodes send very long messages; we also indicate how to modify the algorithm to eliminate this disadvantage.

To synchronize the network we require that each node proceed in a sequence of communication *steps*. During the step $i$, the node sends one message to each of its neighbors and receives one message from each of its neighbors. The node can not send out any step $i+1$ messages until it has sent and received all of its step $i$ messages. Notice that not all nodes will begin step 1 at the same time, and, indeed, some node may not begin step 1 until after it has received some step 1 message from another node. However, this method ensures that two neighboring nodes are never more than one step apart. (This is essentially the synchronization technique referred to by Awerbuch ([3]) as "synchronizer $\alpha$").

The algorithm works by having each node $y$ build a sequence of sets $S[i]_y$, for $0 \le i \le B$, where $S[i]_y$ consists of all pairs $(ID_z, r_z)$ for nodes $z$ for which $d(z,y) = i \le r_z$. These sets are constructed in $B$ steps, with the set $S[i]_y$ being constructed during the $i^{th}$ step of the algorithm. Before step 1, $S[0]_y$ consists of the singleton $(ID_y, r_y)$. Having constructed set $S[i-1]_y$ prior to step $i$, node $y$ defines $T[i]_y$ to be the subset of $S[i-1]_y$ consisting of those pairs $(ID_x, r_x)$ with $r_x \ge i$ and sends the set $T[i]_y$ to each of its neighbors during step $i$. Once it has received $T[i]_z$ from each of its neighbors $z$, it defines $S[i]_y$ to be the union of $T[i]_z$ over all its neighbors $z$, minus the union of $S[j]_y$ over all $j < i$. It is easy to show by induction that the sets $S[i]_y$ are as required. After $B$ steps, the node then selects $C(y)$ to be the node of maximum $ID$ among all of the sets $S[i]_y$.

The drawback to this algorithm is that the sets $T[i]_y$ transmitted in step $i$ can grow very large, requiring very large messages. One way to reduce the message size is to realize that if two pairs $(ID_x, r_x)$ and $(ID_w, r_w)$ are both in $S[i-1]_y$ and

$r_x = r_w$, then $y$ need only put the one with the larger $ID$ (say $ID_x$) in $T[i-1]_y$. Thus at each step, the message sent by a node contains at most $B$ pairs, one for each possible value of $r_x$. It can be shown easily by induction that the maximum element in the union of $S[j]_y$ over $j \leq i$ in the modified algorithm is the same as in the unmodified algorithm.

The messages can be shortened even further by modifying the algorithm as follows. At any time, node $y$ remembers only the pair $(ID_x, r_x)$ for which $ID_x$ is maximum among all pairs that it has received thus far. At step $i$, the node sends this pair to each of its neighbors if $r_x \geq i$, otherwise it sends a null message to its neighbors. The node $C(y)$ is then the node stored after step $B$.

This algorithm does not produce the same result as the algorithms presented above, i.e., $C(y)$ is not necessarily the vertex of maximum $ID$ such that $r_{C(y)} \geq d(C(y), y)$. Instead, $C(y)$ is the vertex of maximum $ID$ such that there is a path of length at most $r_{C(y)}$ from $C(y)$ to $y$ such that for each vertex $w$ on the path, $C(y)$ is the vertex of highest $ID$ in the ball of radius $d(w, C(y))$ around $w$. The proof of Lemma 4.1 can be easily modified to work for this definition of $C(y)$.

Finally, consider the case that $n$ and $B$ $(= \log n)$ are not known in advance. Then the following modification to the algorithm will result in a decomposition whose diameter and number of blocks are both $O(\log n)$ with high probability and whose running time is bounded by the product of the diameter and number of blocks of the resulting decomposition. The idea is to let the quantity $B$ (the radius of the broadcast) vary in the following way: when constructing the $i^{th}$ block of the decomposition $B$ is set equal to $i$, and $p$ is set to $1/2$. Then Lemma 4.1 can be used to show that with high probability, the maximum number of blocks required is $O(\log n)$, and hence so is the maximum diameter of any block.

**Acknowledgements.** We are grateful to Micha Perles for pointing out an error in the original proof of Lemma 3.1.

## References

[1] Y. AFEK and M. RICKLIN: Sparser: A paradigm for running distributed algorithms, 6th International Workshop on Distributed Algorithms, Haifa, Israel November 1992, Springer-Verlag. (*J. of Algorithms*, in press).

[2] N. ALON, L. BABAI, and A. ITAI.: A fast and simple randomized parallel algorithm for the maximal independent set problem, *J. of Algorithms* **7** (1986), 567–583.

[3] B. AWERBUCH: Complexity of network synchronization, *J. ACM* **32** (1985), 804–823.

[4] B. AWERBUCH, A. GOLDBERG, M. LUBY, and S. PLOTKIN: Network decomposition and locality in distributed computation, *Proc. 30th IEEE Symp. on Foundations of Comp. Sci.* (1989) 364–369.

[5] B. AWERBUCH and D. PELEG: Sparse partitions, *FOCS* **31** (1990), 503–513.

[6] I. F. BLAKE and R. C. MULLIN: *An Introduction to Algebraic and Combinatorial Coding Theory*, Academic Press, New York, 1976.

[7] B. BOLLOBÁS: *Extremal graph theory*, Academic Press, New York, 1987.

[8] R. COLE and U. VISHKIN: Deterministic coin tossing and accelerating cascades: micro and macro techniques for designing parallel algorithms, *Proc. 18th ACM Symp. on Theory of Computing* (1986) 206–219.

[9] R. M. FREUND and M. J. TODD: A constructive proof of Tucker's combinatorial lemma, *J. Comb. Theory A* **30** (1981) 321–325.

[10] D. GALE: The game of hex and the Brouwer fixed-point theorem, *Amer. Math. Month.* **86** (1979) 818–827.

[11] A.V. GOLDBERG, S.V. PLOTKIN and G.E. SHANNON: Parallel symmetry-breaking in sparsed graphs, *SIAM J. Disc. Math.* **1** (1989) 434–446.

[12] C. KAKLAMANIS, A.R. KARLIN, F.T. LEIGHTON, V. MILENKOVIC, P. RAGHAVAN, S. RAO, C. THOMBORSON, A. TSANTILAS: Asymptotically tight bounds for computing with faulty arrays of processors, *FOCS* **31** (1990), 285–296.

[13] N. LINIAL: Locality in distributed graph algorithms, *SIAM Journal on Computing*, **21** (1992) 193–201. Preliminary version: N. Linial, Distributive algorithms - global solutions from local data, *FOCS* **28** (1987), 331–335.

[14] M. LUBY: A simple parallel algorithm for the maximal independent set problem. *SIAM J. on Computing*, **15** (1986) 1036–1053.

[15] E. H. SPANIER: *Algebraic Topology*, McGraw-Hill, New York, 1966.

[16] M. TODD: *The computation of fixed points and applications*, Lecture Notes in Economics and Mathematical Systems, 124, Springer-Verlag, 1976.

[17] B. WEISS: A combinatorial proof of the Borsuk-Ulam antipodal point theorem, *Israel J. Math.* **66** (1989) 364–368.

Nathan Linial

*Department of Computer Science*
*The Hebrew University*
*Jerusalem 91904*
*Israel*
nati@cs.huji.ac.il

Michael Saks

*Department of Mathematics*
*Rutgers University*
*New Brunswick, NJ 08903*
and
*Department of Computer Science*
*and Engineering*
*UCSD, La Jolla, CA 92122.*
saks@math.rutgers.edu