Local Partition Hierarchies for General Graphs

Yitzhak Birk[*], Idit Keidar[*], Liran Liss[*] and Assaf Schuster[†]

December 1, 2005

Abstract

We present an algorithm for constructing a hierarchy of partitions with certain locality properties for general unweighted connected graphs. Every level in the hierarchy is a collection of disjoint sets of neighboring nodes, which we refer to as weak clusters. (Weak clusters are not necessarily connected.) The construction has two salient properties. First, it forms a refinement hierarchy, i.e., each (weak) cluster of a certain level is fully subsumed in some cluster of the next level. Second, every level is associated with a radius r and a slack function $\alpha(r) < r$, which characterize the sizes of its clusters. More specifically, r dictates an upper bound on a cluster's radius and $\alpha(r)$ is a lower bound on the minimal radius of some neighborhood that a cluster must cover. This construction servers as a building block for an efficient distributed aggregation algorithm. It may also suite other locality-sensitive algorithms based on hierarchal clustering, in which minimal cluster sizes are a concern.

1 Introduction

Graph constructions are an important building block for efficient locality-sensitive algorithms, which are essential for contemporary large-scale distributed systems. Such constructions are often called Locality-Preserving (LP) representations, as their structure "faithfully captures the topology of the network itself" [9]. Examples for LP representations and their role in distributed computing include using graph coloring for resource allocation [8], hierarchal covers for routing [1], and geographic partitions for resource allocation [6].

In this paper, we provide an LP representation for solving distributed aggregation problems. Specifically, we present a sequential algorithm, HPART, for constructing a hierarchal partition with special locality properties, which is required by the efficient I-LEAG aggregation algorithm [2]. Our construction is applicable to any connected graph G. Each level of the hierarchy provides three data structures: clusters of nodes that partition the graph, cluster representatives called pivots, and routing trees that span clusters.

At the lowest level, every node is its own cluster. Apart from the highest level, which has a single cluster that comprises the entire graph, every cluster is completely subsumed in some cluster of the next level. In level i, cluster sizes are determined according to a radius $r = \theta^i$, where θ is an algorithmic parameter. Every cluster is subsumed by a neighborhood of its pivot p with radius r, and subsumes a neighborhood (of p) with radius $\alpha(r)$, where $\alpha(r) < r$ is a slack function. Therefore, while clusters cannot grow too much in diameter, they are also guaranteed to cover a minimum area of the graph. We do not require clusters be connected, since the connectivity of G allows their nodes to communicate. Finally, every cluster is provided with a directed tree that connects its pivot with those of its subsumed clusters in the previous level. Based on these trees, a pivot can communicate with all nodes covered by its cluster.

[*]Electrical Engineering Department, The Technion – Israel Institute of Technology.
Email: {birk,idish}@ee,liranl@tx.technion.ac.il.

[†]Computer Science Department, The Technion – Israel Institute of Technology.
Email: {assaf}@cs.technion.ac.il.

In I-LEAG, clusters define the environments in which aggregation calculations are held, pivots coordinate the computation, and all communication is based on the routing trees. The hierarchal nature of the partition enables I-LEAG to compute the correct aggregation result in typical scenarios at low levels of the hierarchy without needing to recompute at higher levels [2], thus achieving local computation.

Related Work Hierarchal covers that enforce similar radius constraints on their clusters have been proposed [9] and utilized mainly for routing schemes [1]. However, these constructions are not necessarily partitions. Partition hierarchies have also been suggested, e.g., [3]. Here, the cluster cardinality was constrained rather than the radius and hence clusters do not generally span local environments.

There is a large body of work on constructing graph representations in a distributed local manner, e.g., coloring [7, 8], minimum vertex cover [4], and network decomposition [5]. We concentrate on the feasibility of the construction and provide a simple sequential algorithm, which is intended to be used as a preprocessing stage in higher-level locality-sensitive distributed algorithms.

2 Problem Definition

Let $G = G(V, E)$ be an unweighted connected graph, and let $\Lambda_\theta(G) = \lceil \log_\theta(\mathit{Diameter}(G)) \rceil$. We use the following graph-theoretic notation:

Definition 2.1 (Cluster) *A* cluster *is a subset $S \subseteq V$ of vertices whose induced subgraph $G(S)$ is connected. A* weak cluster *is a subset $S \subseteq V$ that is connected in G but not necessarily in $G(S)$. (If G is connected, any set of nodes is a weak cluster.)*

Definition 2.2 (Distance) *For every two nodes $v_1, v_2 \in V$, the distance between v_1 and v_2 in G, $dist(v_1, v_2)$, is the length of the shortest path connecting them. Given a node v and a cluster S, the distance between v and S is defined as $dist(v, S) = min_{w \in S}(dist(w, v))$.*

Definition 2.3 (Neighborhood) *The $r-$neighborhood ($r \in \mathbb{R}^+$) of a node v, $\Gamma_r(v)$, is the set of nodes $\{v' \mid dist(v, v') \leq r\}$.*

Definition 2.4 (Weak Radius) *Let S be a cluster. The* weak radius *of S with respect to a node $v \in S$ is defined as $WRad(v, S) = max_{w \in S}(dist(v, w))$. The weak radius of S is defined as: $WRad(S) = min_{v \in S}(WRad(v, S))$.*

The following two definitions (taken from [2]) require our construction to be a (θ, α)-local partition hierarchy:

Definition 2.5 (Partition Hierarchy) *An m-level* partition hierarchy *is a triplet $\langle \{\mathcal{S}_i\}, \{\mathcal{P}_i\}, \{\mathcal{T}_i\} \rangle, 0 \leq i \leq m$, where:*

- *$\{\mathcal{S}_i\}$ is a set of partitions, in which for every cluster $S' \in \mathcal{S}_{i-1}$ there exists a cluster $S \in \mathcal{S}_i$ such that $S' \subseteq S$. The topmost level, $\mathcal{S}_m$, contains a single cluster equal to V.*
- *$\{\mathcal{P}_i\}$ is a set of pivots. $\mathcal{P}_i$ includes a single pivot for every cluster $S \in \mathcal{S}_i$. $S_i(p) \in \mathcal{S}_i$ denotes the cluster such that $p \in S$.*
- *$\{\mathcal{T}_i\}$ is a set of forests. For every $p \in \mathcal{P}_i$, $\mathcal{T}_i$ contains a directed tree $T_i(p)$ whose root is p and whose leaves are either $\{p' \in \mathcal{P}_{i-1} \mid S_{i-1}(p') \subseteq S_i(p)\}$ or $S_0(p)$ if $i = 0$.*

The forest set $\{\mathcal{T}_i\}$ induces a *logical tree* that has a single root and spans the entire graph by concatenating the paths formed by tree edges between levels. (The underlying graph of this logical tree can contain cycles, and its edges can be traversed more than once.) For every $p \in \mathcal{P}_i$, we refer to $\widetilde{T}_i(p)$ as the *logical subtree* rooted at p. We denote by $\mathit{Height}(T_i(p))$ and $\mathit{Height}(\widetilde{T}_i(p))$ the heights of $T_i(p)$ and $\widetilde{T}_i(p)$, respectively. We refer to a hierarchy whose clusters can be weak as a *weak* partition hierarchy. Otherwise, it is strong.

A K-bounded *slack function* is a monotone non-decreasing function $\alpha{:}\mathbb{N} \rightarrow \mathbb{R}^+$ such that $\alpha(d) \in [\frac{d}{K}, d]$, for some $K \geq 1$.

Definition 2.6 **((θ, α)-local Partition Hierarchy)** *Let $\theta \geq 2$ and let α be a slack function. A Λ_θ-level partition hierarchy $\langle\{\mathcal{S}_i\}, \{\mathcal{P}_i\}, \{\mathcal{T}_i\}\rangle$ is called (θ, α)-local if for every $p \in \mathcal{P}_i$, it holds that $\Gamma_{\alpha(\theta^i)}(p) \subseteq S_i(p) \subseteq \Gamma_{\theta^i}(p)$, and $\mathit{Height}_i(\widetilde{T}_i(p)) \leq \theta^i$.*

3 Partition Hierarchy Construction

Our hierarchal partitioning algorithm (HPART) is depicted in Algorithm 1. HPART accepts a graph G and a constant $\theta \geq 5$ and returns a (θ, α)-local weak partition hierarchy of G, where $\alpha(r) = r/\theta$ is a θ-bounded slack function. At level 0 (line 1), every node is a cluster. Subsequently, HPART operates in phases, constructing a partition level in each phase. The construction itself is done in two loops:

- Lines 4 - 14: Build a group of connected clusters that fulfill the requirements for the current level. These clusters, however, do not necessarily cover the whole graph.
- Lines 15 - 21: Expand the clusters of (1) until all the whole graph is covered. While the resulting clusters may not be connected, the partition requirements are maintained.

During the first loop, $\overline{V}$ holds uncovered nodes, and $\overline{\mathcal{P}} \subseteq \overline{V}$ holds only those uncovered nodes that are pivots of the previous level. In every iteration, a new cluster S is added to the current level i based on an uncovered θ^{i-1}-neighborhood of some node p (line 5). S comprises the nodes of all clusters of level $i-1$ that intersect with this neighborhood (lines 6-7), and its spanning tree T is formed by connecting the pivots of these clusters to p using shortest paths while avoiding cycles (lines 8-11). p acts as the cluster's pivot. Finally, S, T and p are added to the current level (line 12), and $\overline{\mathcal{P}}$ and $\overline{V}$ are updated.

In the second loop, each iteration selects one uncovered cluster (of the previous level) to be covered by one of the newly created clusters of the current level i. Specifically, HPART selects a yet uncovered cluster with pivot $p' \in \mathcal{P}_{i-1}$, and adds its nodes to a cluster $S \in \mathcal{S}_i$ whose pivot $p \in \mathcal{P}_i$ is closet to p' (lines 16-17). After updating the corresponding tree T (line 18), HPART adjusts level i's cluster and tree sets (line 19), and updates $\overline{\mathcal{P}}$. ($\overline{V}$ is not used in this loop.) It is easy to see that HPART has polynomial execution time.

Correcntess The proof is based on the following three lemmas, which establish several properties that hold after each of the algorithm's two main loops.

Lemma 3.1 *For every level i, $\mathcal{S}_i$ is a partition.*

Proof. By induction on i. S_0 is trivially a partition. We assume that the lemma holds for level $i-1$ and prove for level i. During the first loop, whenever a new cluster S is formed (line 7), every pivot $p' \in \mathcal{P}_{i-1}$ whose cluster $S_{i-1}(p')$ is added to S is removed from $\overline{\mathcal{P}}$. We refer to such pivots as covered by S. As a result, each new cluster covers disjoint sets of pivots. Since $\mathcal{S}_{i-1}$ is a partition, every covered pivot uniquely identifies a cluster that does not overlap with any other cluster of level $i-1$. Hence, upon

Algorithm 1 (HPART)

Input: A connected graph $G(V, E)$ and an integer $\theta \geq 5$
Output: A (θ, α)-local partition hierarchy $\langle\{\mathcal{S}_i\}, \{\mathcal{P}_i\}, \{\mathcal{T}_i\}\rangle, i \in [0, \Lambda_\theta(G)]$, for $\alpha(r) = r/\theta$
Variables: Partition set $\{\mathcal{S}_i\}$, pivots set $\{\mathcal{P}_i\}$ and forest set $\{\mathcal{T}_i\}$, all initially $\emptyset$

```
1: $\mathcal{S}_0 \leftarrow \{\{v\} \mid v \in V\}$, $\mathcal{P}_0 \leftarrow V$, $\mathcal{T}_0 \leftarrow \emptyset$
2: for level $i = 1$ to $\Lambda_\theta(G)$) do
3:    $\overline{\mathcal{P}} \leftarrow \mathcal{P}_{i-1}$, $\overline{V} \leftarrow V$
4:    while $\exists v \in \overline{V}$ s.t. $\Gamma_{\theta^{i-1}}(v) \subseteq \overline{V}$ do    /* build initial clusters */
5:       let $p \in \{v \in \overline{V} \mid \Gamma_{\theta^{i-1}}(v) \subseteq \overline{V}\}$
6:       $\mathcal{P} \leftarrow \{p' \in \overline{\mathcal{P}} \mid \mathcal{S}_{i-1}(p') \cap \Gamma_{\theta^{i-1}}(p) \neq \emptyset\}$
7:       $S \leftarrow \bigcup_{p' \in \mathcal{P}} S_{i-1}(p')$
8:       $T \leftarrow \emptyset$
9:       for all $p' \in \mathcal{P}$ do
10:         $T \leftarrow T \cup \{e \in L \mid L \subseteq E$ is some shortest path from $p'$ to $p$ s.t. $T \cup L$ has no cycles$\}$
11:      end for
12:      $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{S\}$, $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \{p\}$, $\mathcal{T}_i \leftarrow \mathcal{T}_i \cup \{T\}$
13:      $\overline{\mathcal{P}} \leftarrow \overline{\mathcal{P}} - \{p\}$, $\overline{V} \leftarrow \overline{V} - S$
14:   end while
15:   while $\overline{\mathcal{P}} \neq \emptyset$ do    /* expand clusters to include all clusters of $\mathcal{S}_{i-1}$ */
16:      let $p' \in \overline{\mathcal{P}}$ and $p \in \mathcal{P}_i$ s.t. $dist(p, p') = min_{p \in \mathcal{P}_i}\{dist(p, p')\}$
17:      $S \leftarrow S_i(p)$, $S' \leftarrow S \cup S_{i-1}(p')$
18:      $T \leftarrow T_i(p)$, $T' \leftarrow T \cup \{e \in L \mid L$ is some shortest path from $p'$ to $p$ s.t. $T \cup L$ has no cycles$\}$
19:      $\mathcal{S}_i \leftarrow (\mathcal{S}_i - \{S\}) \cup \{S'\}$, $\mathcal{T}_i \leftarrow (\mathcal{T}_i - \{T\}) \cup \{T'\}$
20:      $\overline{\mathcal{P}} \leftarrow \overline{\mathcal{P}} - \{p\}$
21:   end while
22: end for
23: return $\langle\{\mathcal{S}_i\}, \{\mathcal{P}_i\}, \{\mathcal{T}_i\}\rangle, i \in [0, \Lambda_\theta(G)]$
```

completing the first loop, all clusters in $\mathcal{S}_i$ are disjoint. As the second loop only expands these cluster by covering any remaining (uncovered, disjoint) clusters from $\mathcal{S}_{i-1}$, we conclude that $\mathcal{S}_i$ is a partition. ■

Lemma 3.2 *For every level $i > 0$ and every $p \in \mathcal{P}_i$, $T_i(p)$ is a tree that connects every $p' \in \mathcal{P}_{i-1}$ such that $p' \in S_i(p)$ to p by a shortest path.*

Proof. Let $i > 0$ and $p \in \mathcal{P}_i$ be a pivot. The algorithm attempts to join $T_i(p)$ every $p' \in \mathcal{P}_{i-1}$ such that $p' \in S_i(p)$, either in line 10 (when $S_i(p)$ is created) or in line 18 (when $S_i(p)$ is expanded to cover $S_{i-1}(p')$). Because $T_i(p)$ is initialized to $\emptyset$ and cycles are avoided at all times, $T_i(p)$ is always a tree. Furthermore, $T_i(p)$ is comprised of shortest paths only. Therefore, we only need to show that whenever lines 10 or 18 are executed, either $T_i(p)$ already includes a path from p' to p, or there exists a shortest path from p' to p that does not create a cycle in $T_i(p)$.

Consider a pivot $p' \in \mathcal{P}_{i-1}$ that is about to be added to $T_i(p)$. If $T_i(p)$ already includes a path from p' to p, we are done. Otherwise, denote by L' some shortest path from p' to p. (L' exists because the graph is connected.) If L' does not introduce cycles to $T_i(p)$, we are done. Otherwise, L' must cross at least one node that is already spanned by $T_i(p)$. Let v be the first such node (in the direction from p' to p). Denote the existing path in $T_i(p)$ that connects v' to p by L''. Since every part of a shortest path is also a shortest path, the distance from v to p along L'' must equal that along L'. Therefore, the path L formed by concatenating the path from p' to v (along L') and L'' is also a shortest path, which does not introduce any cycles to $T_i(p)$. ■

Lemma 3.3 *For every level i, $\forall p \in \mathcal{P}_i$: (1) $\Gamma_{\theta^{i-1}}(p) \subseteq S_i(p)$; (2) $Height(T_i(p)) \leq 4\theta^{i-1}$; and (3), $S_i(p) \subseteq \Gamma_{5\theta^{i-1}}(p)$.*

Proof. By induction on i. The lemma holds trivially for S_0. We assume that the lemma holds for level $i-1$ and prove for level i. It follows immediately from the construction of a new cluster in line 7 and the fact that $\mathcal{S}_{i-1}$ is a partition that $\forall p \in \mathcal{P}_i$: $\Gamma_{\theta^{i-1}}(p) \subseteq S_i(p)$ upon completing the first loop. Since the second loop does not introduce new clusters nor reduce existing ones, (1) holds.

To prove (2) and (3), we initially claim that upon completing the first loop, $\forall p \in \mathcal{P}_i$: $WRad(p, S_i(p)) \leq 3\theta^{i-1}$. To see this, let p be the node chosen in line 5, $\mathcal{P}$ be the set of level $i-1$ pivots defined in line 6, and S be the cluster constructed in line 7. For every $p' \in \mathcal{P}$: $dist(p, S_{i-1}(p')) \leq \theta^{i-1}$ by construction. In addition, according to the induction hypothesis and the fact that $\theta \geq 5$: $S_{i-1}(p') \subseteq \Gamma_{5\theta^{i-2}}(p') \subseteq \Gamma_{\theta^{i-1}}(p')$, so $WRad(S_{i-1}(p')) \leq \theta^{i-1}$. Thus, it holds that:

$$WRad(p,S) = max_{v\in S}(dist(p,v)) = max_{p'\in\mathcal{P}}\Big(max_{v\in S_{i-1}(p')}(dist(p,v))\Big) \leq$$

$$max_{p'\in\mathcal{P}}\Big(dist(p, S_{i-1}(p')) + 2\,WRad(S_{i-1}(p'))\Big) \leq 3\theta^{i-1}.$$

For every cluster (created in the first loop) that is not expanded in the second loop, both (2) and (3) follow immediately from the claim and Lemma 3.2. For the remaining clusters, we show that the lemma holds after every iteration of the second loop (which expands some existing cluster $S \in \mathcal{S}_i$). Let $p' \in \mathcal{P}_{i-1}$ and $p \in \mathcal{P}_i$ be the pivots chosen in line 16 in some iteration. Observe that upon completing the first loop, for every $v \in \overline{V}$: $min_{S\in\mathcal{S}_i}(dist(v,S)) \leq \theta^{i-1}$. (If $\exists v \in \overline{V}$ such that $\forall S \in \mathcal{S}_i$: $dist(v,S) > \theta^{i-1}$, it holds that $\Gamma_{\theta^{i-1}}(v) \subseteq \overline{V}$ contradicting the fact that the first loop had terminated.) Specifically, this observation holds for p'. Therefore, $\exists p'' \in \mathcal{P}_i$ such that $dist(p', \widetilde{S}_i(p'')) \leq \theta^{i-1}$, where $\widetilde{S}_i(p'')$ denotes p'''s cluster just before beginning the second loop. Consequently,

$$dist(p,p') \leq dist(p'',p') \leq dist(p', \widetilde{S}_i(p'')) + WRad(p'', S_i(p'')) \leq \theta^{i-1} + 3\theta^{i-1} = 4\theta^{i-1}.$$

Since p' is connected to p in $T_i(p)$ by a shortest path (Lemma 3.2), this bounds $Height(T_i(p))$, proving (2). Noting that $WRad(S_{i-1}(p')) \leq \theta^{i-1}$ (follows from the induction hypothesis as shown above), we also have that: $\forall v \in S_{i-1}(p')$: $dist(p,v) \leq dist(p,p') + WRad(S_{i-1}(p')) \leq 5\theta^{i-1}$, which implies (3). ■

Theorem 3.4 *For every $\theta \geq 5$, algorithm HPART constructs a (θ, α)-local weak partition hierarchy for $\alpha(r) = r/\theta$.*

Proof. We first note that in every level i, both loops terminate in finite time because of their dependence on the (finite) pivot set $\overline{\mathcal{P}}$, for which at least one element is removed in every iteration. (In the first loop, each removal of a pivot from $\overline{\mathcal{P}}$ implies removal of nodes from $\overline{V}$, which determines the stopping condition.) Lemma 3.1 ensures that for every level i, $\mathcal{S}_i$ is a partition. The fact that the partitions form a refinement hierarchy, i.e., every cluster of $\mathcal{S}_{i-1}$ is subsumed in some cluster of $\mathcal{S}_i$, is immediate from the construction.

By assigning $\alpha(r) = r/\theta$, it follows from Lemma 3.3 that:

$$\forall p \in \mathcal{P}_i\text{: } \Gamma_{\alpha(\theta^i)}(p) = \Gamma_{\theta^{i-1}}(p) \subseteq S_i(p) \subseteq \Gamma_{5\theta^{i-1}}(p) \subseteq \Gamma_{\theta^i}(p).$$

Finally, we bound the heights of logical subtrees. Let $Height(i) \triangleq max_{p\in\mathcal{P}_i}(Height(\widetilde{T}_i(p)))$. According to Lemma 3.3(2), we have the following recursion for every level i:

$$Height(i) \leq 4\theta^{i-1} + Height(i-1) = 4\sum_{n=0}^{i-1}\theta^i + Height(0) = 4\Big(\frac{\theta^i - 1}{\theta - 1}\Big).$$

For $\theta \geq 5$, we obtain that $Height(i) \leq \theta^i$, concluding that the hierarchy is (θ, α)-local. ■

4 Discussion and Future Work

We presented a simple algorithm for constructing local partition hierarchies for any graph, which can serve as a building block in an efficient aggregation algorithm. This construction can also be applicable for other locality-sensitive algorithms that operate on hierarchal clusters, particulary in cases where the radius of the neighborhoods covered by clusters should reside in a controlled interval.

Our construction guarantees weak partition hierarchies, and provides a multiplicative slack function, namely r/θ. Two immediate questions that arise are whether strong (θ, α)-local partition hierarchies exist in the general case, and to what degree can the corresponding slack functions be tightened. (A tight slack function results in more uniform clusters.) Due to the conflict between connectivity and the requirement that every distinct cluster must subsume a neighborhood of some minimal size, we conjecture that no (θ, α)-local partition exists for general graphs. However, attractive (θ, α)-local partition hierarchies exist for some graph families. For example, in [2], we provide a hierarchy with $\theta = 2$ and an additive slack function (for $r \geq 2$) of $\alpha(r) = max\{r-1, r/2\}$.

Finally, although HPART has been presented as a centralized algorithm, it is not difficult to devise a corresponding distributed implementation for it. Moreover, HPART's output (the partition hierarchy itself) can be represented distributively in a memory efficient manner (each node only needs to hold its parent and children in the trees it belongs to; typically, a node will belong to a single tree in every level).

References

[1] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg. Compact distributed data structures for adaptive network routing. *Proc. 21st ACM Symp. on Theory of Computing*, pages 230–240, May 1989.

[2] Yitzhak Birk, Idit Keidar, Liran Liss, Assaf Schustery, and Ran Wolff. Veracity radius - capturing the locality of distributed computations. *Submitted for publication.*

[3] Shlomi Dolev, Evangelos Kranakis, Danny Krizanc, and David Peleg. Bubbles: Adaptive routing scheme for high-speed dynamic networks. *SIAM Journal on Computing*, 29:804–833, 1999.

[4] T. Moscibroda F. Kuhn and R. Wattenhofer. What cannot be computed locally! In *Proceedings of the 23rd Symposium on Principles of Distributed Computing (PODC)*, July 2004.

[5] F. Kuhn, T. Moscibroda, and R. Wattenhofer. On the locality of bounded growth. In *24th Annual Symposium on Principles of Distributed Computing (PODC)*, July 2005.

[6] J. Li, J. Jannotti, D.S.J. De Couto, and R. Morris D.R. Karger. A scalable location service for geographic ad hoc routing. In *6th ACM Intl. Conf. on Mobile Computing and Networking*, August 2000.

[7] N. Linial. Locality in distributed graph algorithms. *SIAM J. Computing*, 21:193–201, 1992.

[8] M. Naor and L. Stockmeyer. What can be computed locally? *25th ACM Symposium on Theory of Computing*, pages 184–193, 1993.

[9] D. Peleg. *Distributed Computing: A Locality-Sensitive Approach.* SIAM Monographs on Discrete Mathematics and Applications, 2000.