Spectral Vertex Sparsifiers and Pair-Wise Spanners over Distributed Graphs - Supplement

Chun Jiang Zhu [1] **Qinqing Liu** [2] **Jinbo Bi** [2]

1. Local Spectral Sparsification

In this section, we prove Theorem 4 [1] and the accompanying algorithm (Algorithm 4) for the local spectral sparsification method.

Algorithm 4 *LocalSS*

Input: Local graph G_i at each site S_i, terminals T, and a parameter ϵ
Output: An approximate Schur Complement of G
1: **for** site S_i **do**
2: Constructs $H_i = (1+\epsilon)$-$SS(G_i)$ by any standard spectral sparsification algorithm, and then transmits it to the coordinator;
3: Upon receiving all H_i, the coordinator takes their union $H = \cup_{i=1}^{s} H_i$ and then returns $SC(H, T)$;

Theorem 4. *For a subset of terminals $T \subseteq V$ in a graph $G(V, E)$, using communication cost $\tilde{O}(ns)$ the local spectral sparsification method (Algorithm 4) outputs a $(1+\epsilon)$-$SC(G, T)$.*

Proof. By the decomposability of spectral sparsifiers (Theorem 3), it is easy to see H is a spectral sparsifier of G, $H = (1+\epsilon)$-$SS(G)$. Since a spectral sparsifier well approximates the spectrum of the input graph, the constructed structure $SC(H, T) = (1+\epsilon)$-$SC(G, T)$. Because each site transmits a spectral sparsfier of size $O(n)$ to the coordinator, the communication cost is $\tilde{O}(ns)$ □

[1]Department of Computer Science, University of North Carolina at Greensboro, Greensboro, NC, USA [2]Department of Computer Science and Engineering, University of Connecticut, Storrs, CT, USA. Correspondence to: Chun Jiang Zhu <chunjiang.zhu@uncg.edu>, Jinbo Bi <jinbo.bi@uconn.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

[1]In this Appendix, the numbering of theorems and algorithms is consistent with that in the paper, and all new theorems and algorithms are numbered after those in the paper.

2. Local Schur Complement

2.1. Proof of Theorem 7

Theorem 7 (Decomposability of Spectral Sparsifiers for Multi-Graphs). *Let $E_1, \cdots, E_s$ be any partition of the set of edges E in a weighted undirected multi-graph $G(V, E)$. It holds that $\cup_{i=1}^{s}(1+\epsilon)SS(G_i(V, E_i)) = (1+\epsilon)SS(G)$.*

Proof. Throughout the proof, let $F_i = (1+\epsilon)SS(G_i(V, E_i))$, for $1 \leq i \leq s$. According to the definition of spectral sparsifiers, we have for every $x \in R^n$, it holds that

$$(1-\epsilon)x^T L_{G_i} x \leq x^T L_{F_i} x \leq (1+\epsilon)x^T L_{G_i} x. \quad (1)$$

Consider two parallel edges $e_1(u, v)$ of weight $W(e_1)$ and $e_2(u, v)$ of weight $W(e_2)$ in the multi-graph G. In the adjacency matrix A_G of G, the entry for edge (u, v) is the summation of weights of edges e_1 and e_2, $W(e_1) + W(e_2)$. Then it is easy to see that summing all L_{G_i} for $1 \leq i \leq s$ results in L_G.

Summing all inequalities (1) for $1 \leq i \leq s$, we get that

$$(1-\epsilon)x^T L_G x \leq x^T L_F x \leq (1+\epsilon)x^T L_G x,$$

where F is the union of all F_i for $1 \leq i \leq s$. Note that the union of two parallel edges $e_1'(u, v)$ of weight $W(e_1')$ and $e_2'(u, v)$ of weight $W(e_2')$ is an edge (u, v) of weight $W(e_1') + W(e_2')$. By definition of spectral sparsifiers, we have that $F = \cup_{i=1}^{s}(1+\epsilon)SS(G_i(V, E_i)) = (1+\epsilon)SS(G)$. □

2.2. Proof of Theorem 8

We first re-state Algorithm 1 in the main paper and then prove Theorem 8.

Theorem 8. *For a subset of terminals $T \subseteq V$ and boundary vertices $B \subseteq V$ in a graph $G(V, E)$, Algorithm 1 constructs a $(1+\epsilon)$-$SC(G, T)$ using communication cost $\tilde{O}(|T \cup B|s)$, which is close to the communication lower bound $\Omega(|T|s)$.*

Proof. By Theorem 6 in the main paper, we have $H' = \cup_{i=1}^{s} H_i'$ is a Schur complement of G w.r.t. $T \cup B$, $H' =$

Algorithm 1 *LocalSC*

Input: Local graph G_i at each site S_i, terminals T, boundary vertices B, and a parameter ϵ
Output: $(1+\epsilon)$-$SC(G,T)$
1: **for** site S_i **do**
2: Constructs $H_i' = SC(G_i, T \cup B)$;
3: Constructs $H_i'' = (1+\epsilon)$-$SS(H_i')$ by any standard spectral sparsification algorithm, and then transmits it to the coordinator;
4: Upon receiving all H_i'', the coordinator takes their union $H'' = \cup_{i=1}^{s} H_i''$ and then returns $SC(H'', T)$;

$SC(G, T \cup B)$. By the decomposability of spectral sparsifiers (Theorem 3), we have $H'' = \cup_{i=1}^{s} H_i''$ constructed at the coordinator is a spectral sparsifier of H', $H'' = (1+\epsilon)$-$SS(H')$. Since a spectral sparsifier H'' well approximates the spectrum of H', we have $H'' = (1+\epsilon)$-$SC(G, T \cup B)$, and thus the constructed structure $SC(H'', T) = (1+\epsilon)$-$SC(G,T)$. Because each site transmits a spectral sparsfier of size $O(T \cup B)$ to the coordinator, the communication cost is $\tilde{O}(|T \cup B| s)$. When all the vertices in a graph are included in T (*i.e.*, $|T| = n$), the problem degenerates to distributed spectral sparsification problem, which has a communication lower bound $\Omega(ns) = \Omega(|T|s)$ (Chen et al., 2016). □

3. Distributed Schur Complement in the Blackboard Model

The message passing model represents distributed systems with point-to-point communication, while the blackboard model represents distributed systems with a broadcast channel, which can be used to broadcast a message to all sites. We extend the local spectral sparsification to the blackboard model. In the local spectral sparsification method (Algorithm 5), all sites cooperate to construct a $(1+\epsilon)$-spectral sparsifier H of G in the blackboard by any distributed spectral sparsfication algorithm, *e.g.*, (Chen et al., 2016; Zhu et al., 2019). Then the coordinator constructs $SC(H, T)$ and returns it. However, this method incurs a communication cost $\tilde{O}(n+s)$, which could be very large especially when the number of terminals $|T|$ is much smaller than n.

One may attempt to extend the local Schur complement method to the blackboard model. One can still construct SC of the local graph G_i w.r.t. $T \cup B$, $H_i' = SC(G_i, T \cup B)$, at each site S_i as in Step 2 of Algorithm 1. However, we cannot employ existing spectral sparsification algorithms (Chen et al., 2016; Zhu et al., 2019) to construct $(1+\epsilon)$-$SS(\cup_{i=1}^{s} H_i')$ using communication cost of $\tilde{O}(|T \cup B| + s)$. This is because the algorithms do not work for multi-graphs. It is unclear how the sampling-based algorithms perform sampling on multiple edges incident to the same two vertices at different sites. Therefore, it is an open problem to construct a $(1+\epsilon)$-$SC(G,T)$ in the blackboard model using communication cost $o(|T \cup B|^2 s, n+s)$, where the latter is the communication cost of Algorithm 5. Similar to the message passing model, when all vertices in G are included in the terminals, the problem degenerates to distributed spectral sparsification problem and incurs a communication lower bound $\Omega(n+s) = \Omega(|T|+s)$ (Chen et al., 2016). This further shows the importance of the generalization of sparsification to multi-graphs in the message passing model as in Theorem 7.

Algorithm 5 *LocalSS-BL*

Input: Local graph G_i at each site S_i, terminals T, and a parameter ϵ
Output: An approximate Schur Complement of G
1: All sites cooperate to construct $H = (1+\epsilon)$-$SS(G)$ in the blackboard by any distributed spectral sparsification algorithm, *e.g.*, (Chen et al., 2016; Zhu et al., 2019);
2: The coordinator returns $SC(H, T)$;

Theorem 9. *For a subset of terminals $T \subseteq V$ in a graph $G(V, E)$, in the blackboard model Algorithm 5 constructs a $(1+\epsilon)$-$SC(G,T)$ using communication cost $\tilde{O}(n+s)$.*

Proof. By the correctness of the distributed spectral sparsification in the blackboard model, we have H is a spectral sparsifier of G, $H = (1+\epsilon)$-$SS(G)$. Since a spectral sparsifier well approximates the spectrum of the input graph, the constructed structure $SC(H,T) = (1+\epsilon)$-$SC(G,T)$. The communication cost is the same as that of distributed spectral sparsfication, $\tilde{O}(n+s)$ (Chen et al., 2016; Zhu et al., 2019). □

4. Distributed Pair-Wise Spanners

In this section, first we define the notations we will use and then provide the missing details of our results for source-wise, pair-wise, and subset-wise spanners in Sections 4.1, 4.2, and 4.3, respectively.

Notations. In a graph $G(V, E)$, the set of neighbors of a vertex $u \in V$ is $N(u) = \{v \,|\, (u,v) \in E\}$. The neighbors of a set of vertices $U \subseteq V$ are $N(U) = \cup_{v \in U} N(v)$. Let the edge set between vertex u and a vertex set U, $E(u, U) = \{(u,v) \,|\, (u,v) \in E \text{ and } v \in U\}$ and $e(u,U)$ be an arbitrary edge in $E(u,U)$. For a subset of edges $F \subseteq E$, let $E(u, U, F) = E(u,U) \cap F$ and $e(u,U,F)$ be an arbitrary edge in $E(u,U,F)$. We denote the cluster of vertex u as $C(u)$ and the clusters corresponding to vertex set U as $C(U) = \cup_{u \in U} C(u)$. For a set of vertex pairs $P \subseteq V \times V$, let its hitting set $\mathcal{H}(P) = \{H \subseteq V \,|\, \forall (u,v) \in P, u \in H \text{ or } v \in H\}$. The distance between two vertices $u, v \in V$ in a subgraph G' of G is denoted by $d(u, v, G')$ and the shortest path realizing it is $\Pi(u, v, G')$ (ties are broken arbitrarily).

The function $Sample(U, p)$ takes a set U and a parameter p as input and outputs a set formed by independently sampling each element of U with probability p. The function $Send(X, Dest)$ sends data X to the destination $Dest$ and $Dest$ can be the coordinator (CO) or all sites (AS). Similarily, $Send(X, Y, Dest)$ sends data X and Y to $Dest$. All steps in the pseudo-code are executed in the coordinator CO, unless stated explicitly in all sites AS.

Algorithm 6 *BFSTree*

Input: A vertex r and a graph G
Output: An edge set T
1: $L = \{r\}, T = (\{r\}, \emptyset)$; AS: $L^* = \emptyset$;
2: **while** $L \neq \emptyset$ **do**
3: $Send(L, AS)$;
4: AS: $L^* = L^* \cup L$, $Send(N(L) - L^*, e(N(L) - L^*, L), CO)$;
5: $N = \cup_{AS}(N(L) - L^*)$;
6: **for** each vertex $v \in N$ **do**
7: $T = T \cup (v, e(v, L))$;
8: $L = N$;
9: **return** T;

(Fernandez et al., 2020) provided an algorithm for constructing a BFS tree from a given vertex in the message passing model. We formally present the algorithm in Algorithm 6 and will use the following theorem in our proofs. One may notice that 0-S-sourcewise and 0-P-pairwise spanners H of G can be constructed by growing a BFS tree from each vertex in S and $\mathcal{H}(P)$ in G respectively and then adding all edges in the BFS trees to H. Then it is easy to derive the results for constructing 0-sourcewise and 0-pairwise spanners in Theorems 11 and 12, respectively.

Theorem 10 (A BFS Tree (Fernandez et al., 2020)). *The deterministic communication complexity of computing a BFS tree from a given vertex in the message passing model with or without duplication is $\tilde{O}(sn)$.*

Theorem 11 (0-Sourcewise Spanner). *For a graph $G(V, E)$ and source vertices $S \subseteq V$, the algorithm which executes a BFS tree from each vertex in S and returns the union of the BFS trees, outputs a 0-S-sourcewise spanner of size $O(|S|n)$ using communication cost $\tilde{O}(|S|ns)$.*

Theorem 12 (0-Pairwise Spanner). *For a graph $G(V, E)$ and vertex pairs $P \subseteq V \times V$, the algorithm which executes a BFS tree from each vertex in $\mathcal{H}(P)$ and returns the union of the BFS trees, outputs a 0-P-pairwise spanner of size $O(|\mathcal{H}(P)|n)$ using communication cost $\tilde{O}(|\mathcal{H}(P)|ns)$.*

4.1. Source-Wise Spanners (Proof of Theorem 1)

We first re-state the two sub-routines, $Clustering$ and $PathBuying$-S, for sourcewise spanners and then derive the communication cost of $PathBuying$-S.

Algorithm 2 *Clustering*

Input: A vertex set U and a graph G
Output: A set C of clusters and an edge set H
1: $C = \{C(u) = \emptyset \mid u \in U\}$; $H = \emptyset$;
2: $Send(U, AS)$
3: AS: $Send(N(U), e(N(U), U), CO)$
4: $N = \cup_{AS}(N(U))$;
5: **for** each vertex $v \in N$ **do**
6: $(u, v) = e(v, U)$;
7: $H = H \cup (u, v)$;
8: $C(u) = C(u) \cup v$;
9: $X = V - N - U$; $Send(X, AS)$;
10: AS: $Send(E(X, V), CO)$;
11: $H = H \cup (\cup_{AS} E(X, V))$;
12: **return** (C, H);

Algorithm 3 *PathBuying-S*

Input: A vertex set S, a graph G, an edge set H, a parameter f and a set C of clusters
Output: An edge set F
1: $F = \emptyset$;
2: **for** each vertex $r \in S$ **do**
3: $L = \{r\}, T = (\{r\}, \emptyset)$; AS: $L^* = \emptyset$;
4: **while** $L \neq \emptyset$ **do**
5: $Send(L, AS)$;
6: AS: $L^* = L^* \cup L$, $Send(N(L) - L^*, e(N(L) - L^*, L), CO)$;
7: $N = \cup_{AS}(N(L) - L^*)$;
8: **for** each vertex $v \in N$ **do**
9: **if** $e(v, L, H) \neq \emptyset$ **then**
10: $T = T \cup (v, e(v, L, H))$;
11: **else**
12: $T = T \cup (v, e(v, L))$;
13: **if** $|\{e \mid e \in \Pi(r, v, T)$ and $e \notin H\}| > f$ **then**
14: $N = N - \{v\}$; **Continue**;
15: **if** $C(v)$ has not been reached before **then**
16: $F = F \cup \Pi(r, v, T)$;
17: **if** all clusters in C have been reached **then**
18: Go to Line 2;
19: $L = N$;
20: **return** F;

Theorem 13. *The communication complexity of $PathBuying$-S (Algorithm 3) is $\tilde{O}(|S|ns)$.*

Proof. The asymptotical communication complexity of Algorithm 3 is the same as that of executing $|S|$ BFSTree routines (Algorithm 6), each for one vertex in S. According to Theorem 10, the communication complexity is $\tilde{O}(|S|ns)$. □

Now we are ready to present the algorithm (Algorithm 7) for constructing $(+2)$-sourcewise spanners and prove the corresponding theorem, Theorem 1.

Theorem 1 ($(+2)$-Sourcewise Spanner). *For a graph $G(V, E)$, source vertices $S \subseteq V$ and a constant c, using communication cost $\tilde{O}(|S|ns)$ Algorithm 7 constructs a $(+2)$-S-sourcewise spanner of size $O(n^{5/4}|S|^{1/4}\log^{3/4} n)$*

Algorithm 7 *(+2)-Sourcewise-(Pairwise-)Spanner*

Input: A graph $G(V, E)$, a source vertex set S (or vertex pairs P) and a parameter c
Output: An edge set H
1: $H = \emptyset$; $h = (n|S|)^{1/4}\log^{3/4} n$; {respectively, $h = |P|^{1/3}\log^{2/3} n$;}
2: **if** $|S| \le h$ {$|\mathcal{H}(P)| \le h$} **then**
3: $H = \cup_{r\in S} BFSTree(r, G)$; {respectively, $H = \cup_{r\in\mathcal{H}(P)} BFSTree(r, G)$;}
4: **return** H;
5: $U = Sample(V, c\log n/h)$; $(C, H) = Clustering(U, G)$;
6: $R = Sample(U, h^2/(cn\log n))$; $T = \cup_{r\in R} BFSTree(r, G)$;
7: $H = H \cup T$;
8: $F = PathBuying\text{-}S(S, G, H, 3c^2 n\log^2 n/h^2, C)$; {$F = PathBuying\text{-}2P(P, G, H, 3c^2 n\log^2 n/h^2, C)$;}
9: $H = H \cup F$;
10: **return** H;

with probability $1 - O(n^{2-c})$.

Proof. When the size of S, $|S| \le h$, the proof becomes trivial based on Theorem 11. In the following, we only consider the complementary case.

Stretch. We first prove that with probability $1 - O(n^{-c})$ the constructed structure H is a $(+2)$-S-sourcewise spanner. That is, for every $u, v \in S\times V$, $d(u, v, H) \le d(u, v, G)+2$.

We say the shortest path $\Pi(u, v)$ from $u \in S$ to $v \in V$ is *expensive* if after Line 6, Π has more than $3c^2 n\log^2 n/h^2$ missing edges in H. Otherwise, Π is *cheap*. For an expensive path $\Pi(u, v)$, the number of distinct clusters to which Π's vertices belong must be at least $c^2 n\log^2 n/h^2$. Otherwise, there will be four vertices in one cluster, implying that there is a shorter path from u to v than Π, contradicting its optimality. Then with high probability Π passes through a cluster C_i whose center u_i is sampled as a root to grow a BFS tree and edges of the tree are included in H. The probability that Π does not pass through such a cluster is also the probability that none of the clusters Π's vertices belong to is sampled, *i.e.*, at most $(1 - h^2/(cn\log n))^{c^2 n\log^2 n/h^2} = O(n^{-c})$. Let w be a vertex in $\Pi \cap C_i$. We have $d(u, u_i, H) = d(u, u_i, G) \le d(u, w, G) + 1$. The first equation is because we add edges in the BFS tree rooted at u_i to H and the second inequality is due to the triangle inequality. Similarly, we also have $d(u_i, v, H) \le d(w, v, G) + 1$. By the triangle inequality, we then have

$$\begin{aligned} d(u, v, H) &\le d(u, u_i, H) + d(u_i, v, H) \\ &\le d(u, w, G) + d(w, v, G) + 2 \\ &= d(u, v, G) + 2. \end{aligned}$$

For a cheap path $\Pi(u, v)$, let w be the first clustered vertex in Π when traversed from v to u and let C_i and u_i be the cluster and the cluster center, respectively. If there exists no such a clustered vertex w, then all vertices in Π are in H already, and $d(u, v, H) = d(u, v, G)$. Because edges of all unclustered vertices are in H, $d(w, v, H) = d(w, v, G)$. By construction, $PathBuying\text{-}S$ adds the shortest path Π' between every pair of a source vertex and a cluster to H, if Π' contains at most $3c^2 n\log^2 n/h^2$ missing edges in H. Then there is a vertex w' in C_i such that the shortest path $\Pi(u, w')$ from u to w' in G is added to H by $PathBuying\text{-}S$, and $d(u, w', H) = d(u, w', G) \le d(u, w, G)$. Finally, by the triangle inequality we have

$$\begin{aligned} d(u, v, H) &\le d(u, w', H) + d(w', w, H) + d(w, v, H) \\ &\le d(u, w, G) + 2 + d(w, v, G) \\ &= d(u, v, G) + 2. \end{aligned}$$

Size. We now prove that the size of H (# edges in H) is $O(n^{5/4}|S|^{1/4}\log^{3/4} n)$ with probability $1 - O(n^{2-c})$. In the Clustering function (Algorithm 2), each clustered vertex adds one edge to H, and thus the number of edges added by clustered vertices is $O(n)$. The probability that a vertex of degree larger than h has no sampled neighbors and left unclustered is at most $(1 - c\log n/h)^h = O(n^{-c})$. By a union bound, all vertices of degree larger than h are clustered with probability $1 - O(n^{2-c})$. Then the number of edges added by unclustered vertices is $O(nh)$ with probability $1 - O(n^{2-c})$.

In Line 6, the expected number of sampled vertices as BFS tree roots R is $n \cdot c\log n/h \cdot h^2/(cn\log n) = h$. By a Chernoff bound, the probability of $|R| \ge 4h$ is $O(exp(-h))$. Then the number of edges added by BFS trees is $O(nh)$ with probability at least $1 - exp(-h) \ge 1 - O(1/n^c)$. In Line 8, recall that $PathBuying\text{-}S$ adds, for every pair of a source vertex $u \in S$ and a cluster $C_i \in C$, the shortest path between them to H, if it contains at most $3c^2 n\log^2 n/h^2$ missing edges in H. The expected number of clusters in Line 5 is $|C| = nc\log n/h$. By a Chernoff bound, the probability of $|C| \ge 4nc\log n/h$ is $O(exp(-nc\log n/h)) \le O(n^{-c})$, where the last inequality is due to $h \le n$. Then the number of edges added by $PathBuying\text{-}S$ is $|F| = 3c^2 n\log^2 n/h^2 \cdot |S| \cdot |C| = O(n^2|S|\log^3 n/h^3)$. Summing over all steps and substituting $h = (n|S|)^{1/4}\log^{3/4} n$, the total number of edges in H is $O(n^{5/4}|S|^{1/4}\log^{3/4} n)$ with probability $1 - O(n^{2-c})$.

Communication Complexity. We first prove that by taking the sampled cluster centers U as an input, $Clustering$ in Line 5 has communication cost $\tilde{O}(hns)$. In $Clustering$ (Algorithm 2), Line 2 has communication cost $\tilde{O}(ns)$ because $O(|U|) = O(n)$ vertices are transmitted from the coordinator to each of the s sites. Line 3 has the same communication cost because $O(n)$ vertices, each with at most one edge, are transmitted from each of the s sites to the coordinator. Finally, Line 10 w.h.p. incurs communication

cost $\tilde{O}(hns)$ because, as we proved earlier in the size bound, w.h.p. the number of edges of unclustered vertices is $O(nh)$ and the fact that these edges may be presented in each of the s sites. Therefore, in total the function $Clustering$ incurs a communication cost $\tilde{O}(hns)$. For Line 6, the executions of $|R|$ $BFSTree$ incur $\tilde{O}(|R|ns) = \tilde{O}(hns)$ communication. By Theorem 13, $PathBuying$-S in Line 8 incurs $\tilde{O}(|S|ns)$ communication. The other steps will incur no communication because they are executed only in the coordinator. Therefore, the total communication cost is $\tilde{O}(hns + hns + |S|ns) = \tilde{O}(|S|ns)$, where the equality follows from that $h < |S|$. This completes the proof. □

4.2. Pair-Wise Spanners (Proof of Theorem 2)

For ease of discussions, we separate Theorem 2 into two theorems, Theorem 15 for $(+2)$-pairwise spanners and Theorem 17 for $(+4)$-pairwise spanners, and then prove each of them independently.

We first derive the communication cost of the path-buying procedure, $PathBuying$-$2P$, for $(+2)$-pairwise spanners.

Algorithm 8 *PathBuying-2P*

Input: Vertex pairs P, a graph G, an edge set H, a parameter f and a set C of clusters
Output: An edge set F

1: $F = \emptyset$;
2: **for** each vertex $r \in \mathcal{H}(P)$ **do**
3: $L = \{r\}$, $T = (\{r\}, \emptyset)$, $S_r = \{u \,|\, (r,u) \in P\}$; AS: $L^* = \emptyset$;
4: **while** $L \neq \emptyset$ **do**
5: $Send(L, AS)$;
6: AS: $L^* = L^* \cup L$, $Send(N(L) - L^*, e(N(L) - L^*, L), CO)$;
7: $N = \cup_{AS}(N(L) - L^*)$;
8: **for** each vertex $v \in N$ **do**
9: **if** $e(v, L, H) \neq \emptyset$ **then**
10: $T = T \cup (v, e(v, L, H))$;
11: **else**
12: $T = T \cup (v, e(v, L))$;
13: **if** $|\{e \,|\, e \in \Pi(r, v, T)$ and $e \notin H\}| > f$ **then**
14: $N = N - \{v\}$; **Continue**;
15: **if** $v \in S_r$ **then**
16: $F = F \cup \Pi(r, v, T)$;
17: **if** all vertices in S_r have been reached **then**
18: Go to Line 2;
19: $L = N$;
20: **return** F;

Theorem 14. *The communication complexity of $PathBuying$-$2P$ (Algorithm 8) is $\tilde{O}(|\mathcal{H}(P)|ns)$.*

Proof. The asymptotical communication complexity of Algorithm 8 is the same as that of executing $|\mathcal{H}(P)|$ BFSTree routines (Algorithm 6), each for one vertex in $\mathcal{H}(P)$. According to Theorem 10, the communication complexity is $\tilde{O}(|\mathcal{H}(P)|ns)$. □

Now we are ready to prove the main theorem for $(+2)$-pairwise spanners.

Theorem 15 **($(+2)$-Pairwise Spanner).** *For a graph $G(V, E)$, vertex pairs $P \subseteq V \times V$ and a constant c, using communication cost $\tilde{O}(|\mathcal{H}(P)|ns)$ Algorithm 7 constructs a $(+2)$-P-pairwise spanner of size $O(n|P|^{1/3}\log^{2/3} n)$ with probability $1 - O(n^{2-c})$.*

Proof. When $\mathcal{H}(P) \leq h$, the proof is trivial based on Theorem 12 and thus we assume $\mathcal{H}(P) > h$. As shown in Algorithm 7, the path-buying procedure $PathBuying$-$2P$ is different from $PathBuying$-S in source-wise spanners while the clustering procedure and the BFS trees constructions remain the same.

Stretch. We first prove that with probability $1 - O(n^{-c})$ the constructed structure H is a $(+2)$-P-pairwise spanner, *i.e.*, for every $(u, v) \in P$, $d(u, v, H) \leq d(u, v, G) + 2$. Similar to the proof of Theorem 1, we also separate the shortest paths $\Pi(u, v)$ for $(u, v) \in P$ into expensive and cheap paths. $\Pi(u, v)$ is *expensive* if after Line 6, Π has more than $3c^2 n \log^2 n/h^2$ missing edges in H. Otherwise, Π is *cheap*. The stretch guarantees for expensive paths are provided by the clustering procedure and the BFS trees constructions, as can be found in the proof in Theorem 1. For cheap paths $\Pi(u, v)$, $PathBuying$-$2P$ (Algorithm 8) adds their missing edges in H. Therefore, $d(u, v, H) = d(u, v, G)$.

Size. As proved in Theorem 1, the number of edges added by $Clustering$ is $O(nh)$ with probability $1 - O(n^{2-c})$ and the number of edges added by BFS trees is $O(nh)$ with probability $1 - O(1/n^c)$. The number of edges added by $PathBuying$-$2P$ is $|F| = 3c^2 n \log^2 n/h^2 \cdot |P| = O(n|P| \log^2 n/h^2)$. Summing over all steps and substituting $h = |P|^{1/3} \log^{2/3} n$, the total number of edges in H is $O(n|P|^{1/3} \log^{2/3} n)$ with probability $1 - O(n^{2-c})$.

Communication Complexity. As proved in Theorem 1, the function $Clustering$ incurs communication cost $\tilde{O}(hns)$, and the BFS trees incur $\tilde{O}(hns)$ communication. By Theorem 14, $PathBuying$-$2P$ in Line 8 incurs $\tilde{O}(|\mathcal{H}(P)|ns)$ communication. The other steps will incur no communication because they are executed only in the coordinator. Therefore, the total communication cost is $\tilde{O}(hns + hns + |\mathcal{H}(P)|ns) = \tilde{O}(|\mathcal{H}(P)|ns)$, where the equality follows from that $h < |\mathcal{H}(P)|$. This completes the proof. □

For $(+4)$-$Pairwise$-$Spanner$, the clustering procedure and the BFS trees constructions remain the same as those in source-wise spanners, as shown in Algorithm 11. However, there is an additional prefix-suffix-buying procedure $PrefixSuffixBuying$ before the path-buying procedure $Pathbuying$-$4P$. For every pair (u, v) in P, it adds the first l and the last l missing edges in $\Pi(u, v, G)$ to an edge set F.

Algorithm 9 *PrefixSuffixBuying*

Input: Vertex pairs P, a graph G, an edge set H, a parameter l and a set C of clusters
Output: An edge set F

$F = \emptyset$;
for each vertex $r \in \mathcal{H}(P)$ **do**
  $L = \{r\}$, $T = (\{r\}, \emptyset)$, $S_r = \{u | (r,u) \in P\}$; AS: $L^* = \emptyset$;
  while $L \neq \emptyset$ **do**
    $Send(L, AS)$;
    AS: $L^* = L^* \cup L$, $Send(N(L) - L^*, e(N(L) - L^*, L), CO)$;
    $N = \cup_{AS}(N(L) - L^*)$;
    for each vertex $v \in N$ **do**
      if $e(v, L, H) \neq \emptyset$ **then**
        $T = T \cup (v, e(v, L, H))$;
      else
        $T = T \cup (v, e(v, L))$;
      if $v \in S_r$ **then**
        Add the first l missing edges and the last l missing edges in the path from r to v in T to F;
        if all vertices in S_r have been reached **then**
          Go to Line 2;
    $L = N$;
return F;

Algorithm 10 *PathBuying-4P*

Input: A vertex set X, a graph G, an edge set H, a parameter f and a set C of clusters
Output: An edge set F

$F = \emptyset$;
for each vertex $r \in X$ **do**
  $L = \{r\}, T = (\{r\}, \emptyset)$; AS: $L^* = \emptyset$;
  while $L \neq \emptyset$ **do**
    $Send(L, AS)$;
    AS: $L^* = L^* \cup L$, $Send(N(L) - L^*, e(N(L) - L^*, L), CO)$;
    $N = \cup_{AS}(N(L) - L^*)$;
    for each vertex $v \in N$ **do**
      if $e(v, L, H) \neq \emptyset$ **then**
        $T = T \cup (v, e(v, L, H))$;
      else
        $T = T \cup (v, e(v, L))$;
      if $|\{e \mid e \in \Pi(r, v, T) \text{ and } e \notin H\}| > f$ **then**
        $N = N - \{v\}$; **Continue**;
      if $C(v) \in C(X)$ and $C(v)$ has not been reached before **then**
        $F = F \cup \Pi(r, v, T)$;
        if all clusters in $C(X)$ have been reached **then**
          Go to Line 2;
    $L = N$;
return F;

This can be achieved by growing a BFS-like tree from each vertex u in P's hitting set, $\mathcal{H}(P) = \{X \subseteq V \mid \forall (u,v) \in P, u \in X \text{ or } v \in X\}$ and when a vertex v which forms a pair with u in P is reached, maintaining F accordingly. After $PrefixSuffixBuying$, $Pathbuying$-$4P$ is only required to be performed for shortest paths between a subset of cluster centers X, which is obtained by independently sampling each cluster center in U with probability $3c \log n/l$. In the following, we prove that this can reduce the number of edges to $\tilde{O}(n|P|^{2/7})$ at the expense of a larger stretch $(+4)$ while keeping the communication cost $\tilde{O}(\mathcal{H}(P)ns)$.

Theorem 16. *The communication complexity of $PrefixSuffixBuying$ (Algorithm 9) and $PathBuying$-$4P$ (Algorithm 10) are $\tilde{O}(|\mathcal{H}(P)|ns)$ and $\tilde{O}(|X|ns)$, respectively.*

Proof. The asymptotical communication complexity of Algorithm 9 is the same as that of executing $|\mathcal{H}(P)|$ BFSTree routines (Algorithm 6), each for one vertex in $\mathcal{H}(P)$. The asymptotical communication complexity of Algorithm 10 is the same as that of executing $|X|$ BFSTree routines, each for one vertex in X. Then by Theorem 10, their communication complexity are $\tilde{O}(|\mathcal{H}(P)|ns)$ and $\tilde{O}(|X|ns)$, respectively. □

Theorem 17 ($(+4)$**-Pairwise Spanner).** *For a graph $G(V, E)$, vertex pairs $P \subseteq V \times V$ and a constant c, using communication cost $\tilde{O}(|\mathcal{H}(P)|ns)$ Algorithm 11 constructs a $(+4)$-P-pairwise spanner of size $O(n|P|^{2/7} \log^{6/7} n)$ with probability $1 - O(n^{2-c})$.*

Proof. When $|P| < c \log^4 n$, a spanner constructed by $(+2)$-$Pairwise$-$Spanner$ satisfies all required performance measures. Otherwise, when $\mathcal{H}(P) \leq h$, the proof is trivial based on Theorem 12 and thus we assume $\mathcal{H}(P) > h$.

Stretch. We first prove that with probability $1 - O(n^{2-c})$ the constructed structure H is a $(+4)$-P-pairwise spanner, *i.e.*, for every $(u,v) \in P$, $d(u,v,H) \leq d(u,v,G) + 4$. For the shortest path $\Pi(u,v,G)$ between every pair $(u,v) \in P$, if before $PrefixSuffixBuying$, the number of its missing edges in H is at most $2l$, then $PrefixSuffixBuying$ adds all its missing edges to an edge set F_1. Since F_1 will be added to H, we have $d(u,v,H) = d(u,v,G)$. Otherwise, Π's prefix with l missing edges passes through at least $l/3$ distinct clusters. As otherwise, there will be four vertices in one cluster, implying that there is a shorter path from u to v than Π, contradicting its optimality. Then with high probability Π's prefix with l missing edges passes through a cluster whose center is from X. The probability that Π's prefix with l missing edges does not pass through such a cluster is also the probability that none of the $l/3$ cluster centers is sampled, *i.e.*, at most $(1 - 3c \log n/l)^{l/3} = O(n^{-c})$. Similarly, Π's suffix with l missing edges passes through a cluster with its center in X with probability $1 - O(n^{-c})$. By a union bound, all prefixes and suffixes of shortest paths in G pass through a cluster with its center in X with probability $1 - O(n^{2-c})$.

Let C_i and C_j be the clusters which the prefix and the suffix of Π pass through respectively, and let u_i and u_j be their cluster center respectively. Let u' and v' be vertices in $\Pi \cap C_i$

Algorithm 11 *(+4)-Pairwise-Spanner*

Input: A graph $G(V,E)$, vertex pairs P and a parameter c
Output: An edge set H

$H=\emptyset$; $h=|P|^{2/7}\log^{6/7} n$; $l=n\log^3 n/h^{5/2}$;
if $|P|<c\log^4 n$ **then**
  return $(+2)$-$Pairwise$-$Spanner(G,P,c)$;
if $|\mathcal{H}(P)|\le h$ **then**
  $H=\cup_{r\in\mathcal{H}(P)}BFSTree(r,G)$;
  return H;
$U=Sample(V,c\log n/h)$; $(C,H)=Clustering(U,G)$;
$R=Sample(U,h^2/(cn\log n))$; $T=\cup_{r\in R}BFSTree(r,G)$;
$H=H\cup T$;
$F_1=PrefixSuffixBuying(P,G,H,l,C)$;
$H=H\cup F_1$;
$X=Sample(U,3c\log n/l)$;
$F_2=PathBuying$-$4P(X,G,H,3c^2n\log^2 n/h^2,C)$;
$H=H\cup F_2$;
return H;

and $\Pi\cap C_j$, respectively. If after $Clustering$ and the BFS trees constructions, the shortest path $\Pi(u_i,v',G)$ is an expensive path [2] , *i.e.*, it has more than $3c^2n\log^2 n/h^2$ missing edges in H, then we have $d(u_i,v',H)\le d(u_i,v',G)+2$ as proved in Theorem 1. By the triangle inequality, we have

$$\begin{aligned}d(u,v,H)&\le d(u,u',H)+d(u',u_i,H)+d(u_i,v',H)+\\&\quad d(v',v,H)\\&\le d(u,u',G)+1+d(u_i,v',G)+2+d(v',v,G)\\&\le d(u,u',G)+1+d(u',v',G)+1+2+\\&\quad d(v',v,G)\\&=d(u,v,G)+4.\end{aligned}$$

If $\Pi(u_i,v',G)$ is a cheap path, *i.e.*, it has no more than $3c^2n\log^2 n/h^2$ missing edges in H, then $PathBuying$-$4P$ adds the shortest path between u_i and a vertex w in C_j to an edge set F_2 by construction. Since F_2 will be added to H, we have $d(u_i,w,H)=d(u_i,w,G)$. By the triangle inequality and by construction, we have

$$\begin{aligned}d(u,v,H)&\le d(u,u',H)+d(u',u_i,H)+d(u_i,w,H)+\\&\quad d(w,v',H)+d(v',v,H)\\&\le d(u,u',G)+1+d(u_i,w,G)+2+d(v',v,G)\\&\le d(u,u',G)+1+d(u_i,v',G)+2+d(v',v,G)\\&\le d(u,u',G)+1+d(u_i,u',G)+d(u',v',G)+\\&\quad 2+d(v',v,G)\\&\le d(u,u',G)+1+1+d(u',v',G)+2+\\&\quad d(v',v,G)\\&=d(u,v,G)+4.\end{aligned}$$

[2] We extend the notion of expensive and cheap paths to all pairs of vertices in G, instead of only the pairs in P.

Size. As proved in Theorem 1, the number of edges added by $Clustering$ is $O(nh)=O(n|P|^{2/7}\log^{6/7} n)$ with probability $1-O(n^{2-c})$, and the number of edges added by BFS trees is $O(nh)=O(n|P|^{2/7}\log^{6/7} n)$ with probability $1-O(1/n^c)$. By construction, $PrefixSuffixBuying$ adds for each pair in P at most $2l$ edges, and thus the number of edges it adds is $|F_1|=2l\cdot|P|=O(n\log^3 n/h^{5/2}|P|)=O(n|P|^{2/7}\log^{6/7} n)$. Since X is formulated by sampling each vertex in U independently with probability $3c\log n/l$, its expected size is $n\cdot c\log n/h\cdot 3c\log n/l=3nc^2\log^2 n/hl=3c^2|P|^{3/7}\log^{2/7} n$. By a Chernoff bound, the probability of $|X|\ge 12c^2|P|^{3/7}\log^{2/7} n$ is $O(exp(-3c^2|P|^{3/7}\log^{2/7} n))=O(1/n^c)$, where the last equality follows from the assumption that $|P|\ge c\log^4 n$. Therefore, the number of edges added by $PathBuying$-$4P$ is $|F_2|=3c^2n\log^2 n/h^2\cdot|X|^2=O(n|P|^{2/7}\log^{6/7} n)$. Summing over all steps, the total number of edges in H is $O(n|P|^{2/7}\log^{6/7} n)$ with probability $1-O(n^{2-c})$.

Communication Complexity. As proved in Theorem 1, the function $Clustering$ incurs communication cost $\tilde{O}(hns)$, and the BFS trees incur $\tilde{O}(hns)$ communication. According to Theorem 16, $PrefixSuffixBuying$ and $PathBuying$-$4P$ incur $\tilde{O}(|\mathcal{H}(P)|ns)$ and $\tilde{O}(|X|ns)=\tilde{O}(|P|^{3/7}ns)$ communication, respectively. The other steps will incur no communication because they are executed only in the coordinator. Therefore, the total communication cost is $\tilde{O}(hns+hns+|\mathcal{H}(P)|ns+|P|^{3/7}ns)=\tilde{O}(|\mathcal{H}(P)|ns)$, where the equality follows from that $h<|\mathcal{H}(P)|$ and $|P|\ge c\log^4 n$. This completes the proof. □

4.3. Subset-Wise Spanners (Proof of Corollary 1)

With the algorithms for constructing pair-wise spanners with stretch $(+2)$ and $(+4)$, we immediately get algorithms for constructing T-subsetwise spanners with the corresponding stretch. Specifically, a $(+2)$-T-subsetwise and $(+4)$-subsetwise-spanner of G can be constructed by setting $P=T\times T$ and then applying a $(+2)$-P-pairwise and $(+4)$-P-pairwise spanner in G, respectively. In the following, we separate Corollary 1 into Corollaries 2 and 3 and formally prove them.

Corollary 2 ($(+2)$**-Subsetwise Spanner).** *For a graph $G(V,E)$, terminal vertices T and a constant c, using communication cost $\tilde{O}(|T|ns)$ Algorithm 12 constructs a $(+2)$-T-subsetwise spanner of size $O(n|T|^{2/3}\log^{2/3} n)$ with probability $1-O(n^{2-c})$.*

Proof. Because $P=T\times T$, we have $|P|=|T|^2$ and $\mathcal{H}(P)=T$. Then according to Theorem 15, Algorithm 12 returns a T-subsetwise spanner of stretch $(+2)$ and size $O(n|P|^{1/3}\log^{2/3} n)=O(n|T|^{2/3}\log^{2/3} n)$. The communication cost is $\tilde{O}(|\mathcal{H}(P)|ns)=\tilde{O}(|T|ns)$. □

Corollary 3 (**$(+4)$-Subsetwise Spanner).** *For a graph $G(V, E)$, terminal vertices T and a constant c, using communication cost $\tilde{O}(|T|ns)$ Algorithm 12 constructs a $(+4)$-T-subsetwise spanner of size $O(n|T|^{4/7}\log^{6/7} n)$ with probability $1 - O(n^{2-c})$.*

Proof. Because $P = T \times T$, we have $|P| = |T|^2$ and $\mathcal{H}(P) = T$. Then according to Theorem 17, Algorithm 12 returns a T-subsetwise spanner of stretch $(+4)$ and size $O(n|P|^{2/7}\log^{6/7} n) = O(n|T|^{4/7}\log^{6/7} n)$. The communication cost is $\tilde{O}(|\mathcal{H}(P)|ns) = \tilde{O}(|T|ns)$. □

Algorithm 12 *(+2)-((+4)-)Subsetwise-Spanner*

Input: A graph G, terminal vertices T and a parameter c
Output: An edge set H
1: $H = (+2)\text{-}Pairwise\text{-}Spanner(G, T \times T, c)$ {respectively, $H = (+4)\text{-}Pairwise\text{-}Spanner(G, T \times T, c)$;}
2: **return** H;

5. More Experimental Results

Datasets and Methods. We use two synthetic datasets, *Circles* and *Gaussians*, and four real-world datasets *Sculpture*, *Sculpture-1M*, *Sculpture-11M*, and *Beach*. The *Circles* dataset consists of $2K$ vertices and about $17K$ edges. The vertices are sampled from two circles of the same origin and different radii 1 and 1.05, respectively. We connect two vertices if they are mutually k-nearest neighbors of each other for $k = 20$, and use the standard RBF similarity $W(u, v) = exp\{-||u - v||_2^2/2\sigma^2\}$ for $\sigma = 10$. The *Gaussians* dataset has $2K$ vertices and about $12K$ edges with $k = 16$ and $\sigma = 10$ and each vertex from each of four clusters is sampled from an isotropic Gaussians of variance 0.01. For the *Sculpture* dataset, we use a 55×120 version of a photo of *The Greek Slave* [3] resulting in about $7K$ vertices and $100K$ edges with $k = 14$ and $\sigma = 5$. We also use two larger datasets *Sculpture-1M* and *Sculpture-11M* from 73×160 and 110×480 versions of the same photo, respectively. *Sculpture-1M* consists of $11,680$ vertices and $1,063,878$ edges with $k = 220$ and $\sigma = 5$. *Sculpture-11M* has $52,800$ vertices and $11,151,920$ edges with $k = 500$ and $\sigma = 5$. For the *Beach* dataset we use a 524×88 version of a beach photo [4] and it has $46,112$ vertices and $73,159,983$ edges with $k = 4000$ and $\sigma = 5$. We consider each pixel to be a vertex by mapping each pixel to a point in R^5, *i.e.*, (x, y, r, g, b), where the last three coordinates are the RGB values.

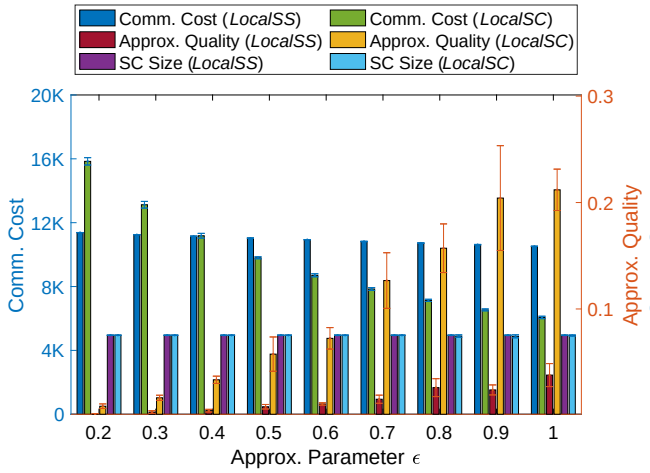
Each graph edge is assigned to a site S_i for $i \in [1, s]$ by a rule respecting the coordinators of its end vertex with a smaller identifier. We first separate the 2-D space containing all the vertices into s non-overlapping partitions, based on angles in *Circles* or grids in the other datasets, and obtain the vertices located within each partition. Then all edges (u, v) of a vertex u in a partition, where $u < v$, are assigned to the corresponding site.

We independently sample each vertex in the graph with probably r and then use the sampled vertices as the terminals. For the spectral sparsification, we employ the implementation of Spielman and Srivastava (Spielman & Srivastava, 2011) from `github.com/danspielman/Laplacians.jl`. The computation of approximation quality for checking how well one graph approximates another is also from the same code. For simplicity, the communication cost is the total number of edges communicated, which approximates the total number of bits by a logarithmic factor. The size of the constructed Schur complement (or called *SC size*) is also recorded as it will affect the computational complexity of the subsequent tasks. All performance measures are averaged over five runs and reported together with their standard deviation.

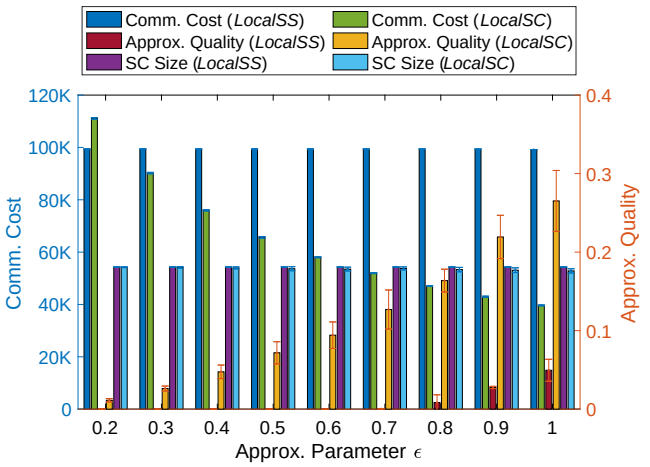
Additional Results. The experimental results on the *Gaussians* and *Sculpture* datasets under the baseline setting are shown in Figures 1a and 1b, respectively. We observe that the communication cost of *LocalSS* is close to the number of edges in the graph even we increase the parameter ϵ, because the spectral sparsification does not reduce the number of edges by a large margin. However, *LocalSC* has a significantly smaller communication cost, especially when ϵ is relatively larger, *e.g.*, 0.6. We emphasize that the approximation qualities of the Schur complements constructed by both methods are always much smaller than the parameter ϵ for all its tested values. For the *Sculpture-1M* dataset, Table 1 shows the small communication cost and good approximation quality of *LocalSC* compared to the centralized method where all m edges in the original graph are communicated.

On the *Circles* dataset, we also investigate the effects when the size of terminals, *i.e.*, the sample rate r, increases but other parameters remain the same. Figure 1c shows that the communication cost of *LocalSC* increases slowly while the communication cost of *LocalSS* keeps the same. Both are consistent with our theoretical analysis. However, *LocalSS* constructs a much denser Schur complement when $|T|$ is relatively large. Different from that *LocalSC* constructs the Schur complement for a graph with the vertex set $T \cup B$, *LocalSS* constructs the Schur complement for a graph with the vertex set V, which requires more star-mesh transforms and then adds more edges to the Schur complement.

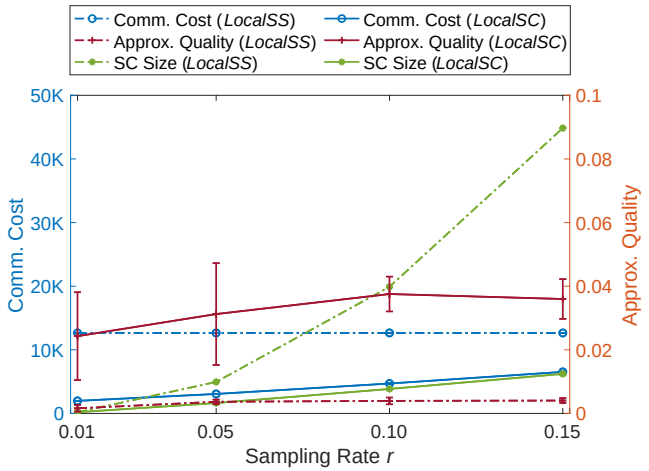
[3] artgallery.yale.edu/collections/objects/14794
[4] www.kdnuggets.com/2019/08/introduction-image-segmentation-k-means-clustering.html



(a) The baseline results for *Gaussians*



(b) The baseline results for *Sculpture*



(c) Effects of varied sample rate r

Figure 1. Additional experimental results.

ϵ	Comm. Cost (cost/m*100%)	Approx. Quality
0.2	106 (0.2)	0.02 (0.004)
0.3	85 (0.2)	0.04 (0.006)
0.4	71 (0.2)	0.06 (0.005)
0.5	61 (0.2)	0.09 (0.02)
0.6	52 (0.2)	0.13 (0.02)
0.7	43 (0.1)	0.18 (0.03)
0.8	36 (0.07)	0.23 (0.03)
0.9	29 (0.07)	0.32 (0.07)

Table 1. Performance of *LocalSC* on *Sculpture-1M* dataset in the baseline setting. Numbers in the parentheses are standard deviations.

6. More Studies on Graph Sparisifiers and Their Distributed Constructions

Graph sparsification or reduction is the task of approximating an arbitrary graph, and is often useful in the design of efficient approximation algorithms. Several notions of graph sparsification has been proposed. (Benczur & Karger, 1996) presented *cut sparsifiers* where the graph cut for every subset of vertices is approximately the same as that in the input graph. (Spielman & Teng, 2011) generalized cut sparsiiers to *spectral sparsifiers*, which approximate the graph spectrum or called the quadratic form of the graph Laplacian in the input graph. (Dinitz et al., 2015) studied *resistance sparsifiers* which approximate effective resistances between all pairs of vertices in the original graph. *Graph spanners* are a subgraph that approximates shortest distances between all vertex pairs in the input graph and were firstly proposed in (Peleg & Schaffer, 1989). All these sparsifiers are constructed on the same vertex set as the input graphs.

Schur Complements. Different from the above sparsifiers, given a subset of vertices called terminals in a graph, Schur complement (Zhang, 2005) preserves effective resistances and energy of electrical flows between the terminals (Durfee et al., 2019). Its applications in semi-supervised learning, Markov chains and finite-element analysis, electrical networks, and computer vision can be found in (Wagner et al., 2018; Kyng et al., 2016; Koutis et al., 2011; Dorfler

& Bullo, 2013; Zhang, 2005). Unfortunately, there is no known study of constructing Schur complements in any distributed computational model. Schur complements are also known as Kron reduction, and have been compared to other graph reduction techniques, which construct a small graph, possibly with a reduced number of vertices, but preserve the spectral property of the original graph (Loukas & Vandergheynst, 2018; Loukas, 2019; Bravo Hermsdorff & Gunderson, 2019). However, it is unclear how these techniques can be adapted in the distributed environment. (Saad & Sosonkina, 1999) studied how to use Schur complements to solve a distributed sparse linear system, but they did not work on distributed constructions of Schur complements.

Source-Wise and Pair-Wise Spanners. (Coppersmith & Elkin, 2006) for the first time studied exact source-wise spanners (called preservers). They showed a source-wise preserver of size $O(\min\{n^{1/2}|S|^2, n|S|\})$ (a pair-wise preserver of size $O(\min\{n|P|^{1/2}, n^{1/2}|P|\})$, respectively). (Bodwin & Williams, 2016) proved a new upper bound of $O(n^{2/3}|P|^{2/3} + n|P|^{1/3})$ for undirected and unweighted graphs using a new type of tiebreaking scheme. Further studies focused on approximate pair-wise spanners of different stretch including $(+2), (+4)$ and $(+6)$ and size $O(n|P|^{1/3})$, $\tilde{O}(n|P|^{2/7})$ and $O(n|P|^{1/4})$ respectively, and source-wise spanners with stretch *e.g.*, $(+2), (+4)$ and $(+6)$ and size $\tilde{O}(n^{5/4}|S|^{1/4})$, $\tilde{O}(n^{11/9}|S|^{2/9})$ and $O(n^{6/5}|S|^{1/5})$, respectively (Cygan et al., 2013; Parter, 2014; Kavitha, 2017; Abboud & Bodwin, 2016). Later, (Abboud & Bodwin, 2016) considered lower bounds for pairwise spanners and established their connection with the lower bounds for pairwise preservers. (Censor-Hillel et al., 2018) studied distributed constructions of pair-wise spanners in the CONGEST model, but they are not for the message passing model.

References

Abboud, A. and Bodwin, G. Error amplification for pairwise spanner lower bounds. In *Proceedings of SODA Conference*, pp. 841–856, 2016.

Benczur, A. and Karger, D. Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. In *Proceedings of STOC Conference*, pp. 47–55, 1996.

Bodwin, G. and Williams, V. V. Better distance preservers and additive spanners. In *Proceedings of SODA Conference*, pp. 855–872, 2016.

Bravo Hermsdorff, G. and Gunderson, L. M. A unifying framework for spectrum-preserving graph sparsification and coarsening. In *Proceedings of NeurIPS Conference*, pp. 7736–7747, 2019.

Censor-Hillel, K., Kavitha, T., Paz, A., and Yehudayoff, A. Distributed construction of purely additive spanners. *Distributed Computing*, 31(3):223–240, 2018.

Chen, J., Sun, H., Woodruff, D., and Zhang, Q. Communication-optimal distributed clustering. In *Proceedings of NIPS Conference*, pp. 3720–3728, 2016.

Coppersmith, D. and Elkin, M. Sparse sourcewise and pairwise preservers. *SIAM Journal on Discrete Mathematics*, 20(2):463–501, 2006.

Cygan, M., Grandoni, F., and Kavitha, T. On pairwise spanners. In *Proceedings of STACS Conference*, pp. 209–220, 2013.

Dinitz, M., Krauthgamer, R., and Wagner, T. Towards resistance sparsifiers. In *Proceedings of APPROX/RANDOM Conference*, pp. 738–755, 2015.

Dorfler, F. and Bullo, F. Kron reduction of graphs with applications to electrical networks. *IEEE Transactions on Circuits and System*, 60(1):150–163, 2013.

Durfee, D., Gao, Y., Goranci, G., and Peng, R. Fully dynamic spectral vertex sparsifiers and applications. In *Proceedings of STOC Conference*, pp. 914–925, 2019.

Fernandez, M. V., Woodruff, D. P., and Yasuda, T. Graph spanners in the message-passing model. In *Proceedings of ITCS Conference*, 2020.

Kavitha, T. New pairwise spanners. *Theory of Computing Systems*, 61:1011–1036, 2017.

Koutis, I., Miller, G. L., and Tolliver, D. Combinatorial preconditioners and multilevel solvers for problems in computer vision and image processing. *Computer Vision and Image Understanding*, 115(12):1638–1646, 2011.

Kyng, R., Lee, Y., Peng, R., Sachdeva, S., and Spielman, D. Sparsified cholesky and multigrid solvers for connection laplacians. In *Proceedings of STOC Conference*, pp. 842–850, 2016.

Loukas, A. Graph reduction with spectral and cut guarantees. *Journal of Machine Learning Research*, 20:1–42, 2019.

Loukas, A. and Vandergheynst, P. Spectrally approximating large graphs with smaller graphs. In *Proceedings of ICML Conference*, pp. 3243–3252, 2018.

Parter, M. Bypassing Erdos' girth conjecture: Hybrid spanners and sourcewise spanners. In *Proceedings of ICALP Conference*, pp. 608–619, 2014.

Peleg, D. and Schaffer, A. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

Saad, Y. and Sosonkina, M. Distributed Schur complement techniques for general sparse linear systems. *SIAM Journal of Scientific Computing*, 21(4):1337–1356, 1999.

Spielman, D. and Srivastava, N. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6): 1913–1926, 2011.

Spielman, D. and Teng, S.-H. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011.

Wagner, T., Guha, S., Kasiviswanathan, S. P., and Mishra, N. Semi-supervised learning on data streams via temporal label propagation. In *Proceedings of ICML Conference*, pp. 5082–5091, 2018.

Zhang, F. The Schur complement and its applications. *Numerical Methods and Algorithms*, 2005.

Zhu, C., Zhu, T., Lam, K.-Y., Han, S., and Bi, J. Communication-optimal distributed dynamic graph clustering. In *Proceedings of AAAI Conference*, pp. 5957–5964, 2019.