ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0006

Computing Weighted Graph Spanners in Message-Passing Model

Linhao Liu[1], Shuhan Chen[2], Hongsheng Xie[3], Tao Wu[4], Zishuo Huang[5]

[1]University of Toronto, St.George, ON, M5S1A1, Canada
[2]University of Southern California, Los Angeles, CA 90089, United States
[3]University of San Diego, San Diego, CA 92093, United States
[4]Shanghai Tech University, Shanghai, 201210, China
[5]University of Florida, Gainesville, FL 32611, United States

Abstract

Graph spanners are sparse subgraphs that offers an approximately preserved all pair-wise shortest-path distance in an input graph. A message passing model has a coordinator responsible for generating the spanner output, and several severs storing edge information. This paper focus on the problem of computing a weighted graph spanner when the edges of the input graph are distributed across multiple players arbitrarily and only point-to-point communication between servers and the coordinator is allowed. The goal is to minimize point-to-point communication cost between coordinator and players under two cases: servers have duplicate edges or not. It also shows a trade-off between communication cost and the quality of computed graph spanner.

Keywords

Graph Spanner; Message Passing Model; Graph Compression; Weighted Graph.

1. INTRODUCTION

Graph Spanner A spanner is a subgraph that preserves approximate distance between each pair of nodes, and it is firstly introduced by Peleg and Schaffer [1]. By reducing a graph to its graph spanner, computation complexity and required storage space can be largely simplified, especially for graphs with dense edges, thus it has been applied into various fields, such as internet routing [2-5] and constructing distance oracles [6-7]. Usually there are two notions of approximation provided by a graph spanner. One is additive, for which in spanner H of graph G, exist a positive integer β such that any node pair (u, v) of satisfy $d_H(u, v) \leq d_G(u, v)+\beta$. The second is multiplicative, for which in spanner H of graph G, exist a positive integer α such that any node pair (u, v) satisfy $d_H(u, v) \leq \alpha * d_G(u, v)$. d: V × V → R computes the distance between two nodes in the graph.

Message-passing model A message-passing model consists of a coordinator C and s players, P_1, P_2 ...., P_s. Each player holds part of the input and can communicate with C. In this paper, player P_i holds a subset Ei of edges on a common complete vertex set V. In other words, the input graph $G=(V, \cup_i E_i)$. Usually edges are allowed to overlap in different servers, while we only consider the case where duplicated edges are not allowed. C is responsible to provide point-to-point communication with servers and output the final spanner. It's worth mentioning that every server do not see contents of other servers, not allowed to communicate with other servers directly, and can't know the message content other servers sent to C unless C send it to all servers.

Communication Complexity Most of graph spanner problems focus on reducing space complexity (how small the graph spanner can be compared with the input graph) [8] and

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0006

runtime complexity (how fast the algorithm can output a graph spanner) [9-11], but in this paper we focus on the communication complexity of the algorithm, that is the amount of information required to be sent between servers and C to compute the correct output.

1.1. Previous Work

Boswana-Sen et al. provided a efficient offline approach to compute weighted multiplicative spanner with linear time computation [12]. It avoids any sort of time-consuming local and global distance computation by constructing BFS tree, but generate and traverse an adjacency list to get the result. On the other hand, Manuel Fernandez V et al. has systematically discussed unweighted graph spanners in message-passing model [13].

Table 1. Unweighted multiplicative graph spanner [13]

Spanner	LB	UB
×3-Spanner	$\Omega(s^{1/2}n^{3/2}+sn)$	$\widetilde{O}(s^{1/2}n^{3/2}+sn)$
×(2k-1)-Spanner	$\Omega(s^{1/2-1/2k}n^{1+1/k}+sn)$	$\widetilde{O}(ks^{1-2/k}n^{1+1/k}+snk)$

1.2. Contribution

While Manuel Fernandez V et al. has widely studied unweighted graph spanners in the message-passing model [13], it has not been studied for weighted graphs. Based on Boswana-Sen approach, this work discussed a updated algorithm to achieve lower communication complexity for multiplicative 3-spanneer, and then generalize it to multiplicative (2k - 1)-spanner. The research shows that same level of communication complexity as unweighted graphs can be achieved for weighted graphs.

Table 2. The result

Spanner	Weighted graph UB
×3-Spanner	$\widetilde{O}(s^{1/2}n^{3/2}+sn)$
×(2k-1)-Spanner	$\widetilde{O}(ks^{1-2/k}n^{1+1/k}+snk)$

2. MULTIPLICATIVE SPANNERS

Recall definition of additive spanner.

Definition (Multiplicative spanner): H is an multiplicative α-spanner of graph G, if any node pair (u, v) of satisfy $d_H(u, v) \le \alpha * d_G(u, v)$, where dG: V × V → R computes the distance between two nodes in G and d_H: V × V → R computes the distance between two nodes in H.

2.1. Multiplicative 3-spanners Without Duplication

This work uses a slightly different version of Baswana-Sen vertex-cluster joining algorithm [12].

2.1.1 Algorithm

Phase 1: Forming clusters

Randomly sample a set of vertices R by the probability P = $1/\sqrt{sn}$. R will be treated as the set of cluster centers.

Start with Esp={}. For each vertex v in V

a. If v is not adjacent to any cluster center, add all edges incident to v into E_{sp}

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0006

b. If v has one or more cluster neighbor, mark the cluster C with smallest edge to v, and add the smallest edge to E_{sp} along with all edges incident to v whose weight is smaller than E_C to E_{sp}. Append v into cluster C.

As the last step, discard all edges (u, v) not in E_{sp}, u and v not in R but in the same cluster.

Phase 2: Vertex-cluster joining operation

For each vertex and each cluster, keep the least weight edge between them in E_{sp} and discard the remaining. Therefore, for each vertex-cluster pair there should be only one path that owns the shortest weight.

2.1.2 Proof of correctness

Based on the Lemma 2.2 in previous work[12], for all discard edge (u, v),

(a). u and v in the same cluster: there's a 2-edge path u-x-v with each edge lighter than (u, v), x is the corresponding cluster center.

(b). u and v not in the same cluster: if (u, v) was not added in phase 2, there must exist an edge (u, v') such that v and v' in the same cluster, w(u, v') < w(u, v), and (u, v') was added in phase2. By Lemma 2.1 in [12]:

dH(u, v) = w(u, v') + w(v', x) + w(x, v)

≤ w(u, v) + w(u, v') + w(u, v) (Lemma 2.1)

≤ 3w(u,v)

2.1.3 Compute communication complexity

With P = $1/\sqrt{sn}$, the expected number of sampled vertices is $\sqrt{n/s}$, the coordinator broadcast them with $O(s\sqrt{n/s})=O(\sqrt{sn})$.

For each vertex v, each player detects the smallest edge connects v to a cluster and send it to the coordinator (O(sn)). After the coordinator finds the smallest edge among all edges received from player (O(sn)), it broadcasts the weight of the smallest edge and add all edges players send back to E_{sp}. The expected number of incident edges with less weight is at most $\sqrt{sn}$, thus the expected number of edges sent to the coordinator is $O(n\sqrt{sn})=O(s^{1/2}n^{3/2})$. It's worth mentioning that a vertex without sampled neighbor is expected to have degree smaller than $\sqrt{sn}$, thus will not change the final result. For the last step of phase 1, the coordinator broadcasts each vertex's belonging and needs no response, Thus cost O(sn).

In phase 2, each player sends back an edge to each cluster for each vertex, thus cost O(sn|R|) = $O(n\sqrt{sn})=O(s^{1/2}n^{3/2})$.

In all, this updated algorithm provides a total communication complexity of $O(s^{1/2}n^{3/2}+sn)$ to generate a multiplicative 3-spanner with high probability.

2.1.4 Result comparison

In the initial algorithm given by Baswana et al.[12], the probability P = $1/\sqrt{n}$.

The expected number of clusters is $\sqrt{n}$, the first broadcast takes $O(s\sqrt{n})$, the expected number of edges sent to the coordinator is $O(s\sqrt{n})$. Phase 2 takes $O(sn|R|)=O(sn^{3/2})$, and others keep the same. Thus compare with the initial algorithm, communication complexity gets largely improved when the number of players is relatively small.

However, when adding edges in phase 1, each vertex is expected to add $\sqrt{sn}$ edges, thus the expected number of total edges in the spanner becomes $O(s^{1/2}n^{3/2})$, which is more than $O(n^{3/2})$ in the initial algorithm, so here comes a trade-off relationship between communication complexity and the quality of the graph spanner.

2.2. Multiplicative (2k-1)-Spanners Without Duplication

2.2.1 Algorithm

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0006

This work uses a slightly different version of cluster-cluster joining algorithm based on the work of Manuel Fernandez V et al.[13].

Phase 1: Forming clusters

To begin with, set $d_1 = s^{1-2/k}n^{1/k}$, $d_2 = n^{1/k/s2/k}$, E' = E, ES = ε_0 = {}, C0 ={{v}| v ϵ V}, V' = V.

At the beginning, include all edges incident to vertices of degree at most d_1. Randomly sample clusters from C_0 with probability P= $\log(n)/d_1$. Now for each vertex, if it is adjacent to any sampled cluster, add the least weight edge connects it to one of sampled clusters. And discard the rest from E'.

Phase 2: Expanding clusters

For $\lfloor k/2 \rfloor$ iterations from i = 1, 2, .. $\lfloor k/2 \rfloor$, sample clusters with probability $\log(n)/d_2$, and consider each vertex v:

a. If v is not adjacent to any sampled cluster, add the least weight edge from the set E'(v, c) to ES for each adjacent cluster c, and discard rest E'(v, c) from E'.

b. If v has one or more sampled cluster neighbor, add the least weight edge that connects v with a sampled cluster into ES and ε_i, and discard rest E'(v, c) from E'. In addition, add the least weight edge that connects v and other clusters.

As the last step, remove all edges with both endpoints belongs to the same cluster from E'.

Phase 3: Connecting clusters

If k is odd, add the least weight edge between every pair of clusters in $C_{\lfloor k/2 \rfloor}$.

If k is even, add the least weight edge between every pair of clusters in $C_{\lfloor k/2 \rfloor}$ and $C_{\lfloor k/2 \rfloor - 1}$.

2.2.2 Proof of Correctness

Theorem 1 Given a path (c, s_1, s_2, ... s_n), n ≤ $\lfloor k/2 \rfloor$, if c is the center of a cluster C, and s_1, s_2, ... s_n ϵ C, then $w(c, s_1) < w(s_1, s_2) < w(s_2, s_3) .... < w(s_{n-1}, s_n)$.

Proof

The process is similar with offline version that Boswana et al. gave[12]. If some i in 1... n satisfy $w(s_{i-1}, s_i) > w(s_i, s_{i+1})$, then when adding s_i to C.

(1) s_{i+1} in C, then (s_i, s_{i+1}) will be discarded since both endpoint belongs to C, thus contradiction

(2) s_{i+1} belongs to another sampled cluster C', then s_i will join C', thus contradiction.

(3) s_{i+1} does not belongs to sampled cluster, then (si, s_{i+1}) will be added to E_s, thus s_{i+1} won't join C in i+1th iteration and cause contradiction.

■

If (u, v) is discarded, then

(a) u and v belongs to the same cluster, then u~x~v is an alternative path from u to v, which consists of $2\lfloor k/2 \rfloor$ edges, and the weight of each edge is less than w(u, v). (based on Theorm 1)

(b) u belongs to cluster A and v belongs to B,

(i) if k is odd, then A, B ϵ $C_{\lfloor k/2 \rfloor}$. Suppose (x, y) connects A, B, then w(x, y) < w(u, v), and an alternative path u~a~x~y~b~v exists, a and b are cluster centers. Based on Theorem 1. every edge in u~a~x~y~b~v has weight smaller than w(u, v), and each cluster has radius $\lfloor k/2 \rfloor$. Thus in total $d_H(u, v) \le (4\lfloor k/2 \rfloor+1) * w(u, v) = (2k - 1) * w(u, v)$.

(ii) If k is even, then the proof can be done in similar way.

2.2.3 Compute Communication Complexity

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0006

In phase 1, the algorithm uses d1n = $\widetilde{O}(s^{1-2/k}n^{1+1/k})$ to include all low degree edges, and $\widetilde{O}$(sn) to broadcast sampled cluster information, and assign vertices to sampled clusters. In each iteration of phase 2, it takes $\widetilde{O}$(sn) to broadcast sampled cluster information, $\widetilde{O}$(sn) to receive from vertices with one or more sampled cluster neighbors, and remove intra-cluster edges. If a vertex is not adjacent to any sampled cluster, then with high probability it is adjacent to at most d2 clusters, and thus takes sd2 = $\widetilde{O}(s^{1-2/k}n^{1/k})$ to connect a vertex to all adjacent clusters.

After phase 2, the expected number of clusters when k is odd is

$$\widetilde{O}\left(\frac{n}{d_1 d_2^{\lfloor k/2 \rfloor -1}}\right) = \widetilde{O}\left(\frac{n}{d_2^{\lfloor k/2 \rfloor}}\right) = \widetilde{O}\left(\frac{n}{s d_2^{\lfloor k/2 \rfloor}}\right) = \widetilde{O}\left(\frac{n^{1/2+1/2k}}{s^{1/k}}\right)$$

and communication complexity in phase 2 is

$$\widetilde{O}\left(\mathrm{s}\left(\frac{n^{1/2+1/2k}}{s^{1/k}}\right)^2\right) = \widetilde{O}(\mathrm{s}^{1-2/k}n^{1+1/k})$$

Similarly, the communication complexity when k is even is

$$\widetilde{O}\left(\mathrm{s}\frac{n}{d_1 d_2^{\lfloor k/2 \rfloor -1}}\frac{n}{d_1 d_2^{\lfloor k/2 \rfloor -2}}\right) = \widetilde{O}\left(\frac{n^2}{s d_2^{k-1}}\right) = \widetilde{O}(s^{1-2/k}n^{1+1/k})$$

In all, this work achieves $\widetilde{O}(snk + ks^{1-2/k}n^{1+1/k})$, which is consistent with unweighted case.

2.3. Multiplicative Spanner with Edge Duplication

Manuel Fernandez et al. has proven that generating unweighted multiplicative (2k-1)-spanner has a tight bound $\widetilde{\Theta}(sn^{1+1/k})$ of communication complexity [13]. Since unweighted graph is a special case of weighted graph, which all edges have same weight 1, thus it can be concluded that weighted graph has the same tight bound.

3. CONCLUSION

By considering the worst-case distribution of edges in the message-passing model, we shows that multiplicative weighted graph spanner without edge duplication can achieve same upper bound as unweighted graphs, which maintaining the same amount of approximation or distortion. The result is summarized in table 2.

4. FURTHER WORK

This paper has shown that a better performance of generating multiplicative weighted graph spanner can be achieved by an improved algorithm, but more studies are needed for additive weighted graph spanner, no matter with or without edge duplication. Also, lower bounds are not mentioned in this work.

ACKNOWLEDGEMENTS

We hereby express gratitude to our dear partners in our group and these authors contributed equally: Linhao Liu, Shuhan Chen, Hongsheng Xie, Tao Wu and Zishuo. Huang. Without their effort, this thesis cannot be accomplished. In the process of compilation, they made great contribution on data collecting and analysis. This thesis is rather a common achievement than a private possession.

REFERENCES

[1] David Peleg and Alejandro A Sch¨affer. Graph spanners. Journal of Graph Theory, 13(1):99–116, 1989.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0006

[2] Mikkel Thorup and Uri Zwick. Compact routing schemes. In SPAA, pages 1–10, 2001.

[3] Lenore Cowen. Compact routing with minimum stretch. J. Algorithms, 38(1):170–183, 2001.

[4] Lenore Cowen and Christopher G. Wagner. Compact roundtrip routing in directed networks. J. Algorithms, 50(1):79–95, 2004.

[5] David Peleg and Eli Upfal. A tradeoff between space and efficiency for routing tables (extended abstract). In Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA, pages 43–52, 1988.

[6] Mikkel Thorup and Uri Zwick. Approximate distance oracles. J. ACM, 52(1):1–24, 2005.

[7] Surender Baswana and Sandeep Sen. Approximate distance oracles for unweighted graphs in ˜o(n2) time. In Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004, pages 271–280, 2004.

[8] Amir Abboud and Greg Bodwin. 2016. The 4/3 additive spanner exponent is tight. In Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC'16). 351--361.

[9] David Woodruff. Additive spanners in nearly quadratic time. In ICALP (1), pages 463–474, 2010.

[10] Baswana, S., and Sen, S. 2003. A simple linear time algorithm for computing a (2k−1)-spanner of O(n1+1/k) size in weighted graphs. In Proceedings of the 30th International Colloquium on Automata, Languages and Programming, Eindhoven, The Netherlands. 384--396(2003).

[11] Michael Elkin and Ofer Neiman. Efficient algorithms for constructing very sparse spanners and emulators. In Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19, pages 652–669, 2017.

[12] Surender Baswana and Sandeep Sen. A simple and linear time randomized algorithm for computing sparse spanners in weighted graphs. Random Struct. Algorithms, 30(4):532–563, 2007.

[13] Manuel Fernandez, David P. Woodruff, and Taisuke Yasuda. 2020. Graph Spanners in the Message-Passing Model. In 11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA (LIPIcs), Thomas Vidick (Ed.), Vol. 151. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 77:1–77:18. https://doi.org/10.4230/LIPIcs. ITCS.2020.77