Node-Disjoint Multipath Spanners and their Relationship with Fault-Tolerant Spanners

Cyril Gavoille[1⋆], Quentin Godfroy[1⋆], and Laurent Viennot[2⋆⋆]

[1] University of Bordeaux, LaBRI
[2] INRIA, University Paris 7, LIAFA

Abstract. Motivated by multipath routing, we introduce a multi-connected variant of spanners. For that purpose we introduce the p-multipath cost between two nodes u and v as the minimum weight of a collection of p internally vertex-disjoint paths between u and v. Given a weighted graph G, a subgraph H is a p-multipath s-spanner if for all u, v, the p-multipath cost between u and v in H is at most s times the p-multipath cost in G. The s factor is called the stretch.
Building upon recent results on fault-tolerant spanners, we show how to build p-multipath spanners of constant stretch and of $\tilde{O}(n^{1+1/k})$ edges[3], for fixed parameters p and k, n being the number of nodes of the graph. Such spanners can be constructed by a distributed algorithm running in $O(k)$ rounds.
Additionally, we give an improved construction for the case $p = k = 2$. Our spanner H has $O(n^{3/2})$ edges and the p-multipath cost in H between any two node is at most twice the corresponding one in G plus $O(W)$, W being the maximum edge weight.

1 Introduction

It is well-known [2] that, for each integer $k \geqslant 1$, every n-vertex weighted graph G has a subgraph H, called *spanner*, with $O(n^{1+1/k})$ edges and such that for all pairs u, v of vertices of G, $d_H(u, v) \leqslant (2k-1) \cdot d_G(u, v)$. Here $d_G(u, v)$ denotes the distance between u and v in G, i.e., the length of a minimum cost path joining u to v. In other words, there is a trade-off between the size of H and its *stretch*, defined here by the factor $2k - 1$. Such trade-off has been extensively used in several contexts. For instance, this can be the first step for the design of a Distance Oracle, a compact data structure supporting approximate distance query while using sub-quadratic space [25,4,5]. It is also a key ingredient for several distributed algorithms to quickly compute a sparse skeleton of a connected graph, namely a connected spanning subgraph with only $O(n)$ edges. This can

⋆ Supported by the European project "EULER", the ANR-project "ALADDIN", the équipe-projet INRIA "CÉPAGE". The first author is Member of the "Insitut Universitaire de France".

⋆⋆ Supported by the European project "EULER", the ANR-project "ALADDIN", and the équipe-projet INRIA "GANG".

[3] Tilde-O notation is similar to Big-O up to poly-logarithmic factors in n.

be done by choosing $k = O(\log n)$. The target distributed algorithm can then be run on the remaining skeleton [3]. The skeleton construction can be done in $O(k)$ rounds, whereas computing a spanning tree requires diameter rounds in general. We refer the reader to [21] for an overview on graph spanner constructions.

However, it is also proved in [25] that if G is directed, then it may have no sub-digraph H having $o(n^2)$ edges and constant stretch, the stretch being defined analogously by the maximum ratio between the *one-way* distance from u to v in H and the one-way distance from u to v in G. Nevertheless, a size/stretch trade-off exists for the *round-trip* distance, defined as the sum of a minimum cost of a dipath from u to v, and a minimum cost dipath from v to u (see [7,23]). Similar trade-offs exist if we consider the *p-edge-disjoint multipath* distance (in undirected graphs) for each $p \geqslant 1$, that is the minimum sum of p edge-disjoint paths joining u and v, see [10].

1.1 Trade-offs for non-increasing graph metric

More generally, we are interested in size/stretch trade-offs for graphs (or digraphs) for some non-increasing graph metric. A *non-increasing* graph metric δ associates with each pair of vertices u, v some non-negative cost that can only decrease when adding edges. In other words, $\delta_G(u,v) \leqslant \delta_H(u,v)$ for all vertices u, v and spanning subgraphs H of G. Moreover, if $\delta_H(u,v) \leqslant \alpha \cdot \delta_G(u,v) + \beta$, then we say that H is an (α,β)-spanner and that its *stretch* (w.r.t. the graph metric δ) is at most (α,β). We simply say that H is an α-spanner if $\beta = 0$. The *size* of a spanner is the number of its edges.

In the previous discussion we saw that every graph or digraph has a spanner H of size $o(n^2)$ and with bounded stretch for graph metrics δ such as round-trip, p-edge-disjoint multipath, and the usual graph distance. However, it does not hold for one-way distance. A fundamental task is to determine which graph metrics δ support such size/stretch trade-off. We observe that the three former graph metrics cited above have the triangle inequality property, whereas the one-way metric does not.

This paper deals with the construction of spanners for the vertex-disjoint multipath metric. A p-multipath between u and v is a subgraph composed of the union of p pairwise internally vertex-disjoint paths joining u and v. The *cost* of a p-multipath between u and v is the sum of the weight of the edges it contains. Given an undirected positively weighted graph G, define $\delta^p_G(u,v)$ as the minimum cost of a p-multipath between u and v if it exists, and ∞ otherwise. A p-multipath s-spanner is a spanner H of G with stretch at most s w.r.t. the graph metric δ^p. In other words, for all vertices u, v of G, $\delta^p_H(u,v) \leqslant s \cdot \delta^p_G(u,v)$, or $\delta^p_H(u,v) \leqslant \alpha \cdot \delta^p_G(u,v) + \beta$ if $s = (\alpha,\beta)$. It generalizes classical spanners as $d_G(u,v) = \delta^p_G(u,v)$ for $p = 1$.

1.2 Motivations

Our interest in the node-disjoint multipath graph metric stems from the need for multipath routing in networks. Using multiple paths between a pair of nodes is an obvious way to aggregate bandwidth. Additionally, a classical approach to quickly overcome link failures consists in pre-computing fail-over paths which are disjoint from primary paths [14,19,18]. Multipath routing can be used for traffic load balancing and for minimizing delays. It has been extensively studied in ad hoc networks for load balancing, fault-tolerance, higher aggregate bandwidth, diversity coding, minimizing energy consumption (see [17] for a quick overview). Considering only a subset of links is a practical concern in link state routing in ad hoc networks [13]. This raises the problem of computing spanners for the multipath graph metric, a first step towards constructing compact multipath routing schemes.

1.3 Our contributions

Our main contribution is to show that sparse p-multipath spanners of constant stretch do exist for each $p \geqslant 1$. Moreover, they can be constructed *locally* in a constant number of rounds. More precisely, we show that:

1. Every weighted graph with n vertices has a p-multipath $kp \cdot O(1 + p/k)^{2k-1}$-spanner of size $\tilde{O}(p^2 \cdot n^{1+1/k})$, where k and p are integral parameters $\geqslant 1$. Moreover, such a multipath spanner can be constructed distributively in $O(k)$ rounds.
2. For $p = k = 2$, we improve this construction whose stretch is 18. Our algorithm provides a 2-multipath $(2, O(W))$-spanner of size $O(n^{3/2})$ where W is the largest edge weight of the input graph.

Distributed algorithms are given in the classical $\mathcal{LOCAL}$ model of computations (cf. [20]), a.k.a. the free model [15]. In this model nodes operate in synchronous discrete rounds (nodes are also assumed to wake up simultaneously). At each round, a node can send and/or receive messages of unbounded capacity to/from its neighbors and can perform any amount of local computations. Hence, each round costs one time unit. Also, nodes have unique identifiers that can be used for breaking symmetry. As long as we are concerned with running time (number of rounds) and not with the cost of communication, synchronous and asynchronous message passing models are equivalent.

1.4 Overview

Multipath spanners have some flavors of fault-tolerant spanners, notion introduced in [6] for general graphs. A subgraph H is an *r-fault tolerant s-spanner* of G if for any set F of at most $r \geqslant 0$ faulty vertices, and for any pair u, v of vertices outside F, $d_{H \setminus F}(u, v) \leqslant s \cdot d_{G \setminus F}(u, v)$.

At first glance, r-fault tolerant spanners seem related to $(r + 1)$-multipath spanners. (Note that both notions coincide to usual spanners if $r = 0$.) This is motivated by the fact that, if for an edge uv of G that is not in H, and if, for each set F of r vertices, u and v are connected in $H \setminus F$, then by Menger's Theorem H must contain some p-multipath between u and v. If the connectivity condition fulfills, there is no guarantee however on the cost of the p-multipath in H compared to the optimal one in G. Actually, as presented on Fig. 1, there are 1-fault tolerant s-spanners that are 2-multipath but with arbitrarily large stretch.

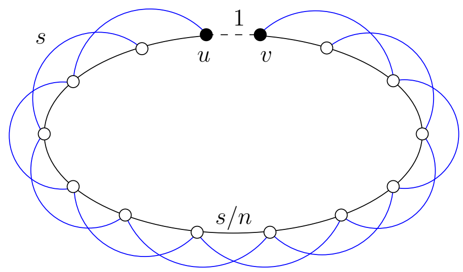

Fig. 1. A weighted graph G composed of a cycle of $n+1$ vertices plus $n-1$ extra edges, and a spanner $H = G \setminus \{uv\}$. Edge uv has weight 1, non-cycle edges have weight s, and cycle edges weight s/n so that $d_H(u,v) = s$. Removing any vertex $z \notin \{u,v\}$ implies $d_{G\setminus\{z\}}(u,v) = 1$ and $d_{H\setminus\{z\}}(u,v) = 2s(1-1/n)$. For other pairs of vertices x, y, $d_{H\setminus\{z\}}(x,y)/d_{G\setminus\{z\}}(x,y) < 2s$. Thus, H is a 1-fault tolerant $2s$-spanner. However $\delta^2_H(u,v)/\delta^2_G(u,v) \geqslant sn/s$. Thus, H is a 2-multipath spanner with stretch at least n.

Nevertheless, a relationship can be established between p-mutlipath spanners and *some* r-fault tolerant spanners. In fact, we prove in Section 2.4 that every r-fault tolerant s-spanner that is b-*hop* is a $(r+1)$-multipath spanner with stretch bounded by a function of b, r and s. Informally, a b-hop spanner H must replace every edge uv of G not in H by a path simultaneously of low cost and composed of at most b edges. We observe that many classical spanner constructions (including the greedy one) do not provide bounded-hop spanners, although such spanners exist as proved in Section 2.1. Some variant presented in [6] of the Thorup-Zwick constructions [25] are also bounded-hop (Section 2.2). Combining these specific spanners with the generic construction of fault tolerant spanners of [9], we show in Section 2.3 how to obtained a $\mathcal{LOCAL}$ distributed algorithm for computing a p-mutlipath spanner of bounded stretch. A maybe surprising fact is that the number of rounds is independent of p and n. We stress that the distributed algorithm that we obtain has significantly better running time than the original one presented in [9] that was $\Omega(p^3 \log n)$.

For instance for $p = 2$, our construction can produce a 2-multipath 18-spanner with $O(n^{3/2} \log^{3/2} n)$ edges. For this particular case we improve the general construction in Section 3 with a completely different approach providing a low multiplicative stretch, namely 2, at the cost of an additive term depending of the largest edge weight.

We note that the graph metric δ^p does not respect the triangle inequality for $p > 1$. For $p = 2$, a cycle from u to w and a cycle from w to v does not imply the existence of a cycle from u to v. The lack of this property introduces many complications for our second result. Basically, there are $\Omega(n^2)$ pairs u, v of vertices, each one possibly defining a minimum cycle $C_{u,v}$ of cost $\delta^2_G(u,v)$. If we want to create a spanner H with $o(n^2)$ edges, we cannot keep $C_{u,v}$ for all pairs u, v. Selecting some vertex w as pivot for going from u to v is usually a solution of save edges (in particular at least one between u and v). One pivot can indeed serve for many other pairs. However, without the triangle inequality, $C_{u,w}$ and $C_{w,v}$ do not give any cost guarantee on $\delta^2_H(u,v)$.

2 Main Construction

In this section, we prove the following result:

Theorem 1. *Let G be a weighted graph with n vertices, and p, k be integral parameters $\geqslant 1$. Then, G has a p-multipath $kp \cdot O(1+p/k)^{2k-1}$-spanner of size $O(kp^{2-1/k} n^{1+1/k} \log^{2-1/k} n)$ that can be constructed w.h.p. by a randomized distributed algorithm in $O(k)$ rounds.*

Theorem 1 is proved by combining several constructions presented now.

2.1 Spanners with few hops

An s-spanner H of a weighted graph G is *b-hop* if for every edge uv of G, there is a path in H between u and v composed of at most b edges and of cost at most $s \cdot \omega(uv)$ (where $\omega(uv)$ denotes the cost of edge uv). An *s-hop spanner* is simply an s-hop s-spanner.

If G is unweighted (or the edge-cost weights are uniform), the concepts of s-hop spanner and s-spanner coincide. However, not all s-spanners are s-hop. In particular, the $(2k-1)$-spanners produced by the greedy[4] algorithm [2] are not.

For instance, consider a weighted cycle of $n+1$ vertices and any stretch s such that $1 < s < n$. All edges of the cycle have unit weight, but one, say the edge uv, which has weight $\omega(uv) = n/s$. Note that $d_G(u,v) = \omega(uv) > 1$. The greedy algorithm adds the n unit cost edges but the edge uv to H because $d_H(u,v) = n \leqslant s \cdot \omega(uv)$ (recall that uv is added only if $d_H(u,v) > s \cdot d_G(u,v)$). Therefore, H is an s-spanner but it is only an n-hop spanner.

However, we have:

Proposition 1. *For each integer $k \geqslant 1$, every weighted graph with n vertices has a $(2k-1)$-hop spanner with less than $n^{1+1/k}$ edges.*

Proof. Consider a weighted graph G with edge-cost function ω. We construct the willing spanner H of G thanks to the following algorithm which can be seen as the dual of the classical greedy algorithm, till a variant of Kruskal's algorithm:

[4] For each edge uv in non-decreasing order of their weights, add it to the spanner if $d_H(u,v) > s \cdot d_G(u,v)$.

(1) Initialize H with $V(H) := V(G)$ and $E(H) := \varnothing$;
(2) Visit all the edges of G in non-decreasing order of their weights, and add the edge uv to H only if every path between u and v in H has more than $2k-1$ edges.

Consider an edge uv of G. If uv is not in H then there must exist a path P in H from u to v such that P has at most $2k-1$ edges. We have $d_H(u,v) \leqslant \omega(P)$. Let e be an edge of P with maximum weight. We can bound $\omega(P) \leqslant (2k-1)\cdot\omega(e)$. Since e has been considered before the edge uv, $\omega(e) \leqslant \omega(uv)$. It follows that $\omega(P) \leqslant (2k-1)\cdot\omega(uv)$, and thus $d_H(u,v) \leqslant (2k-1)\cdot\omega(uv)$. Obviously, if uv belongs to H, $d_H(u,v) = \omega(uv) \leqslant (2k-1)\cdot\omega(uv)$ as well. Therefore, H is $(2k-1)$-hop.

The fact that H is sparse comes from the fact that there is no cycle of length $\leqslant 2k$ in H: whenever an edge is added to H, any path linking its endpoints has more than $2k-1$ edges, i.e., at least $2k$.

We observe that H is simple even if G is not. It has been proved in [1] that every simple n-vertex m-edge graph where every cycle is of length at least $2k+1$ (i.e., of girth at least $2k+1$), must verify the Moore bound:

$$n \;\geqslant\; 1 + d\sum_{i=0}^{k-1}(d-1)^i \;>\; (d-1)^k$$

where $d = 2m/n$ is the average degree of the graph. This implies that $m < \frac{1}{2}(n^{1+1/k}+n) < n^{1+1/k}$.

Therefore, H is a $(2k-1)$-hop spanner with at most $n^{1+1/k}$ edges. □

2.2 Distributed bounded hop spanners

There are distributed constructions that provide s-hop spanners, at the cost of a small (poly-logarithmic in n) increase of the size of the spanner compared to Proposition 1.

If we restrict our attention to deterministic algorithms, [8] provides for unweighted graphs a $(2k-1)$-hop spanner of size $O(kn^{1+1/k})$. It runs in $3k-2$ rounds without any prior knowledge on the graph, and optimally in k rounds if n is available at each vertex.

Proposition 2. *There is a distributed randomized algorithm that, for every weighted graph G with n vertices, computes w.h.p. a $(2k-1)$-hop spanner of $O(kn^{1+1/k}\log^{1-1/k} n)$ edges in $O(k)$ rounds.*

Proof. The algorithm is a distributed version of the spanner algorithm used in [6], which is based on the sampling technique of [25]. We make the observation that this algorithm can run in $O(k)$ rounds. Let us briefly recall the construction of [6, p. 3415].

To each vertex w of G is associated a tree rooted at w spanning the *cluster* of w, a particular subset of vertices denoted by $C(w)$. The construction of $C(w)$ is a refinement over the one given in [25]. The main difference is that the clusters' depth is no more than k edges. The spanner is composed of the union of all such cluster spanning trees. The total number of edges is $O(kn^{1+1/k}\log^{1-1/k} n)$. It is proved in [6] that for every edge uv of G, there is a cluster $C(w)$ containing u and v. The path of the tree from w to one of the end-point has at most $k-1$ edges and cost $\leqslant (k-1)\cdot\omega(uv)$, and the path from w to the other end-point has at most k edges and cost $\leqslant k\cdot\omega(uv)$. This is therefore a $(2k-1)$-hop spanner.

The random sampling of [25] can be done without any round of communications, each vertex randomly select a level independently of the other vertices. Once the sampling is performed, the clusters and the trees can be constructed in $O(k)$ rounds as their the depth is at most k. □

2.3 Fault tolerant spanners

The algorithm of [9] for constructing fault tolerant spanners is randomized and generic. It takes as inputs a weighted graph G with n vertices, a parameter $r \geqslant 0$, and any algorithm $\mathbf{A}$ computing an s-spanner of $m(\nu)$ edges for any ν-vertex subgraph of G. With high probability, it constructs for G an r-fault tolerant s-spanner of size $O(r^3 \cdot m(2n/r)\cdot\log n)$. It works as follows:
Set $H := \varnothing$, and repeat independently $O(r^3\log n)$ times:

(1) Compute a set S of vertices built by selecting each vertex with probability $1-1/(r+1)$;
(2) $H := H \cup \mathbf{A}(G\setminus S)$.

Then, they show that for every fault set $F \subset V(G)$ of size at most r, and every edge uv, there exists with high probability a set S as computed in Step (1) for which $u, v \notin S$ and $F \subseteq S$. As a consequence, routine $\mathbf{A}(G\setminus S)$ provides a path between u and v in $G\setminus S$ (and thus also in $G\setminus F$) of cost $\leqslant s\cdot\omega(uv)$. If uv lies on a shortest path of $G\setminus F$, then this cost is $\leqslant s\cdot d_{G\setminus F}(u,v)$. From their construction, we have:

Proposition 3. *If $\mathbf{A}$ is a distributed algorithm constructing an s-hop spanner in t rounds, then algorithm [9] provides a randomized distributed algorithm that in t rounds constructs w.h.p. an s-hop r-fault tolerant spanner of size $O(r^3 \cdot m(2n/r)\cdot\log n)$.*

Proof. The resulting spanner H is s-hop since either the edge uv of G is also in H, or a path between u and v approximating $\omega(uv)$ exists in some s-hop spanner given by algorithm $\mathbf{A}$. This path has no more than s edges and cost $\leqslant s\cdot\omega(uv)$.

Observe that the algorithm [9] consists of running in parallel $q = O(r^3\log n)$ times independent runs of algorithm $\mathbf{A}$ on different subgraphs of G, each one using t rounds. Round i of all these q runs can be done into a single round of communication, so that the total number of rounds is bounded by t, not by q.

More precisely, each vertex first selects a q-bit vector, each bit set with probability $1-1/(r+1)$, its jth bit indicating whether it participates to the jth run of **A**. Then, q instances of algorithm **A** are run in parallel simultaneously by all the vertices, and whenever the algorithms perform their ith communication round, a single message concatenating the q messages is sent. Upon reception, a vertex expands the q messages and run the jth instance of algorithm **A** only if the jth bit of its vector is set.

The number of rounds is no more than t. □

2.4 From fault tolerant to multipath spanner

Theorem 2. *Let H be a s-hop $(p-1)$-fault tolerant spanner of a weighted graph G. Then, H is also a p-multipath $\varphi(s,p)$-spanner of G where $\varphi(s,p) = sp \cdot O(1+p/s)^s$ and $\varphi(3,p) = 9p$.*

To prove Theorem 2, we need the following intermediate result, assuming that H and G satisfy the statement of Theorem 2.

Lemma 1. *Let uv be an edge of G of weight $\omega(uv)$ that is not in H. Then, H contains a p-multipath connecting u to v of cost at most $\varphi(s,p) \cdot \omega(uv)$ where $\varphi(s,p) = sp \cdot O(1+p/s)^s$ and $\varphi(3,p) = 9p$.*

Proof. From Menger's Theorem, the number of pairwise vertex-disjoint paths between two non-adjacent vertices x and y equals the minimum number of vertices whose removal disconnects x and y.

By definition of H, $H \setminus F$ contains a path P_F of at most s edges between u and v for each set F of at most $p-1$ vertices (excluding u and v). This is because u and v are always connected in $G \setminus F$, precisely by a single edge path of cost $\omega(uv)$. Consider P_H the subgraph of H composed of the union of all such P_F paths (so from u to v in $H \setminus F$ – see Fig. 2 for an example with $p = 2$ and $s = 5$).

Vertices u and v are non-adjacent in P_H. Thus by Menger's Theorem, P_H has to contain a p-multipath between u and v. Ideally, we would like to show that this multipath has low cost. Unfortunately, Menger's Theorem cannot help us in this task.

Let $\kappa_s(u,v)$ be the minimum number of vertices in P_H whose deletion destroys all paths of at most s edges between u and v, and let $\mu_s(u,v)$ denote the maximum number of internally vertex-disjoint paths of at most s edges between u and v. Obviously, $\kappa_s(u,v) \geqslant \mu_s(u,v)$, and equality holds by Menger's Theorem if $s = n-1$. Equality does not hold in general as presented in Fig. 2. However, equality holds if s is the minimum number of edges of a path between u and v, and for $s = 2,3,4$ (cf. [16]).

Since not every path of at most s edges between u and v is destroyed after removing $p-1$ vertices in P_H, we have that $\kappa_s(u,v) \geqslant p$. Let us bound the total number of edges in a p-multipath Q of minimum size between u and v in P_H. Let r be the least number such that $\mu_r(u,v) \geqslant p$ subject to $\kappa_s(u,v) \geqslant p$. The total number of edges in Q is therefore no more than pr.

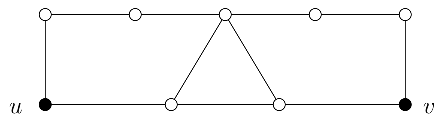

Fig. 2. A subgraph P_H constructed by adding paths between u and v with at most $s = 5$ edges and with $p = 2$. Removing any vertex leaves a path of at most 5 edges, so $\kappa_5(u, v) > 1$. However, there aren't two vertex-disjoint paths from u to v of at most 5 edges, so $\kappa_5(u, v) > \mu_5(u, v)$. Observe that $\mu_6(u, v) = \kappa_5(u, v) = 2$.

By construction of P_H, each edge of P_H comes from a path in $H \setminus F$ of cost $\omega(P_F) \leqslant s \cdot d_{G \setminus F}(u, v) \leqslant s \cdot \omega(uv)$. In particular, each edge of Q has weight at most $s \cdot \omega(uv)$. Therefore, the cost of Q is $\omega(Q) \leqslant prs \cdot \omega(uv)$.

It has been proved in [22] that r can be upper bounded by a function $r(s, p) < \binom{p+s-2}{s-2} + \binom{p+s-3}{s-2} = O(1 + p/s)^s$ for integers s, p, and $r(3, p) = 3$ since as seen earlier $\kappa_3(u, v) = \mu_3(u, v)$. It follows that H contains a p-multipath Q between u and v of cost $\omega(Q) \leqslant sp \cdot O(1 + p/s)^s \cdot \omega(uv)$ as claimed. □

Proof of Theorem 2. Let x, y be any two vertices of a graph G with edge-cost function ω. We want to show $\delta^p_H(x, y) \leqslant \varphi(s, p) \cdot \delta^p_G(x, y)$. If $\delta^p_G(x, y) = \infty$, then we are done. So, assume that $\delta^p_G(x, y) = \omega(P_G)$ for some minimum cost p-multipath P_G between x and y in G. Note that $\omega(P_G) = \sum_{uv \in E(P_G)} \omega(uv)$.

We construct a subgraph P_H between x and y in H by adding: (1) all the edges of P_G that are in H; and (2) for each edge uv of P_G that is not in H, the p-multipath Q_{uv} connecting u and v in H as defined by Lemma 1.

The cost of P_H is therefore:

$$\omega(P_H) \;=\; \sum_{uv \in E(P_H)} \omega(uv) \;=\; \left(\sum_{uv \in E(P_G) \cap E(H)} \omega(uv) \right) + \left(\sum_{uv \in E(P_G) \setminus E(H)} \omega(Q_{uv}) \right) .$$

By Lemma 1, $\omega(Q_{uv}) \leqslant \varphi(s, p) \cdot \omega(uv)$. It follows that:

$$\omega(P_H) \leqslant \varphi(s, p) \cdot \sum_{uv \in E(P_G)} \omega(uv) \;=\; \varphi(s, p) \cdot \omega(P_G) \;=\; \varphi(s, p) \cdot \delta^p_G(x, y)$$

as $\varphi(s, p) \geqslant 1$ and by definition of P_G.

Clearly, all edges of P_H are in H. Let us show now that P_H contains a p-multipath between x and y. We first assume x and y are non-adjacent in P_H. By Menger's Theorem applied between x and y in P_H, if the removal of every set of at most $p - 1$ vertices in P_H does not disconnect x and y, then P_H has to contain a p-multipath between x and y.

Let S be any set of less than $p - 1$ faults in G. Since P_G is a p-multipath, P_G contains at least one path between x and y avoiding S. Let's call this path Q. For each edge uv of Q not in H, Q_{uv} is a p-multipath, so it contains one path avoiding S. Note that Q_{uv} may intersect Q_{wz} for different edges uv and wz of

Q. If it is the case then there is a path in $Q_{uv} \cup Q_{wz}$ from u to z (avoiding v and w), assuming that u, v, w, z are encountered in this order when traversing Q. Overall there must be a path connecting x to y and avoiding S in the subgraph $(Q \cap H) \cup \bigcup_{uv \in Q \setminus H} Q_{uv}$. By Menger's Theorem, P_H contains a p-multipath between x and y.

If x and y are adjacent in P_H, then we can subdivide the edge xy into the edges xz and zy by adding a new vertex z. Denote by P'_H this new subgraph. Clearly, if P'_H contains a p-multipath between x and y, then P_H too: a path using vertex z in P'_H necessarily uses the edges xz and zy. Now, P'_H contains a p-multipath by Menger's Theorem applied on P'_H between x and y that are non-adjacent.

We have therefore constructed a p-multipath between x and y in H of cost at most $\omega(P_H) \leqslant \varphi(s,p) \cdot \delta^p_G(x,y)$. It follows that $\delta^p_H(x,y) \leqslant \varphi(s,p) \cdot \delta^p_G(x,y)$ as claimed. □

Theorem 1 is proved by applying Theorem 2 to the construction of Proposition 3, which is based on the distributed construction of s-hop spanners given by Proposition 2. Observe that the number of edges of the spanner is bounded by $O(kp^3 \cdot m(2n/p) \cdot \log n) = O(kp^{2-1/k} n^{1+1/k} \log^{2-1/k} n)$.

3 Bi-path Spanners

In this section we concentrate our attention on the case $p = 2$, i.e., 2-multipath spanners or *bi-path* spanners for short. Observe that for $p = k = 2$ the stretch is $\varphi(3,2) = 18$ using our first construction (cf. Theorems 1 and 2). We provide in this section the following improvement on the stretch and on the number of edges.

Theorem 3. *Every weighted graph with n vertices and maximum edge-weight W has a 2-multipath $(2, O(W))$-spanner of size $O(n^{3/2})$ that can be constructed in $O(n^4)$ time.*

While the construction shown earlier was essentially working on edges, the approach taken here is more global. Moreover, this construction essentially yields an additive stretch whereas the previous one is only multiplicative. Note that a 2-multipath between two nodes u and v corresponds to an elementary cycle. We will thus focus on cycles in this section.

An algorithm is presented in Section 3.1. Its running time, the size of the spanner is analyzed in Section 3.2, and the stretch in Section 3.3. Due to space limitation some proofs of these sections appear in the long version [11].

3.1 Construction

Classical spanner algorithms combines the use of trees, balls, and clusters. These standard structures are not suitable to the graph metric δ^2 since, for instance, two nodes belonging to a ball centered in a single vertex can be in two different bi-components[5] and therefore be at an infinite cost from each other. We will adapt theses standard notions to structures centered on edges rather than vertices.

Consider a weighted graph G and with an edge uv that is not a cut-edge[6]. Let us denote by $G[uv]$ the bi-component of G containing uv, and by $\delta^2_H(uv,w)$ the minimum cost of a cycle in subgraph H passing through the edge uv and vertex w, if it exists and ∞ otherwise.

We define a 2-*path spanning tree of root* uv as a minimal subgraph T of G such that every vertex w of $G[uv]$ belongs to a cycle of T containing uv. Such definition is motivated by the following important property (see Property 1 in Section 3.3): for all vertices a, b in $G[uv] \setminus \{u, v\}$, $\delta^2_G(a,b) \leqslant \delta^2_T(uv,a) + \delta^2_T(uv,b)$. This can be seen as a triangle inequality like property.

If $\delta^2_T(uv,w) = \delta^2_G(uv,w)$ for every vertex w of $G[uv]$, T is called a *shortest* 2-*path spanning tree*. An important point, proved in Lemma 2 in Section 3.2, is that such T always exists and contains $O(\nu)$ edges, ν being the number of vertices of $G[uv]$.

In the following we denote by $B^2_G(uv,r) = \{w : \delta^2_G(uv,w) \leqslant r\}$ and $B_G(u,r) = \{w : d_G(u,w) \leqslant r\}$ the 2-*ball* (resp. 1-*ball*) of G centered at edge uv (at vertex u) and of radius r. We denote by $N_G(u)$ the set of neighbors of u in G. We denote by $\mathrm{BFS}(u,r)$ any shortest path spanning tree of root u and of depth r (not counting the edge weights). Finally, we denote by ${\mathrm{SPST}^2}_G(uv)$ any shortest 2-path spanning tree of root uv in $G[uv]$.

The spanner H is constructed with Algorithm 1 from any weighted graph G having n vertices and maximum edge weight W. Essentially, the main loop of the algorithm selects an edge uv from the current graph lying at the center of a dense bi-component, adds the spanner H shortest 2-path spanning tree rooted at uv, and then destroys the neigborhood of uv.

$F := G$, $H := (\varnothing, \varnothing)$;
while $\exists uv \in E(G)$, $|B^2_G(uv,4W) \cap (N_G(u) \cup N_G(v))| > \sqrt{n}$ **do**
  $H := H \cup {\mathrm{SPST}^2}_F(uv) \cup \mathrm{BFS}_G(u,2) \cup \mathrm{BFS}_G(v,2)$;
  $G := G \setminus (B^2_G(uv,4W) \cap (N_G(u) \cup N_G(v)))$
$H := H \cup G$

Algorithm 1: Construction of H.

3.2 Size analysis

The proof of the spanner's size is done in two steps, thanks to the two next lemmas.

First, Lemma 2 shows that the while loop does not add too much edges: a shortest 2-path spanning tree with linear size always exists. It is built upon the algorithm of Suurballe-Tarjan [24] for finding shortest pairs of edge-disjoint paths in weighted digraphs.

[5] A short for 2-vertex-connected components.

[6] A cut-edge is an edge that does not belong to a cycle.

Lemma 2. *For every weighted graph G and for every non cut-edge uv of G, there is a shortest 2-path spanning tree of root uv having $O(\nu)$ edges where ν is the number of vertices of $G[uv]$. It can be computed in time $O(n^2)$ where n is the number of vertices of G.*

Secondly, Lemma 3 shows that the graph G remaining after the while loop has only $O(n^{3/2})$ edges. For that, G is transformed as an unweighted graph (edge weights are set to one) and we apply Lemma 3 with $k = 2$. The result we present is actually more general and interesting in its own right. Indeed, it gives an alternative proof of the well-known fact that graphs with no cycles of length $\leqslant 2k$ have $O(n^{1+1/k})$ edges since $B^2_G(uv, 2k) = \varnothing$ in that case.

Lemma 3. *Let G be an unweighted graph with n vertices, and $k \geqslant 1$ be an integer. If for every edge uv of G, $|B^2_G(uv, 2k) \cap N_G(u)| \leqslant n^{1/k}$, then G has at most $2 \cdot n^{1+1/k}$ edges.*

Combining these two lemmas we have:

Lemma 4. *Algorithm 1 creates a spanner of size $O(n^{3/2})$ in time $O(n^4)$.*

Proof. Each step of the while loop adds $O(n)$ edges from Lemma 2, and as it removes at least $\sqrt{n}$ vertices from the graph this can continue at most $\sqrt{n}$ times. In total the while loop adds $O(n^{3/2})$ edges to H.

After the while loop, the graph G is left with every $B^2_G(uv, 4W) \cap (N_G(u) \cup N_G(v))$ smaller than $\sqrt{n}$. If we change all edges weights to 1, it is obvious that every $B^2_G(uv, 4) \cap (N_G(u) \cup N_G(v))$ is also smaller than $\sqrt{n}$. Then as $B^2_G(uv, 4) \cap N_G(u)$ is always smaller than $B^2_G(uv, 4) \cap (N_G(u) \cup N_G(v))$ we can apply Lemma 3 for $k = 2$, and therefore bound the number of edges added in the last step of Algorithm 1.

The total number of edges of H is $O(n^{3/2})$.

The costly steps of the algorithm are the search of suitable edges uv and the cost of construction of SPST^2.

The search of suitable edges is bounded by the number of edges as an edge e which is not suitable can be discarded for the next search: removing edges from the graph cannot improve $B^2(e, 4W)$. Then for each edge a BFS of depth 4 must be computed, whose cost is bounded by the number of edges. So in the end the search costs at most $O(n^4)$.

The cost of building a SPST^2 is bounded by the running time of [24], which at worst costs $O(n^2)$ (the reduction is essentially in $O(m + n)$). Since the loop is executed at most $\sqrt{n}$ times, the total cost is $O(n^{7/2})$.

So the total running time is $O(n^4)$. □

3.3 Stretch analysis

The proof for the stretch is done as follows: we consider a, b two vertices such that $\delta^2_F(a, b) = \ell$ is finite (if it is infinite there is nothing to prove). We need to prove that the spanner construction is such that at the end, $\delta_H^2(a,b) \leqslant 2\ell + O(W)$. To this effect, we define $P_F = P_F^1 \cup P_F^2$ as a cycle composed of two disjoint paths (P_F^1 and P_F^2) going from a to b such that its weight sums to $\delta_F^2(a,b)$.

Proving the stretch amounts to show that there exists a cycle $P_H = P_H^1 \cup P_H^2$ joining a and b in the final H, with cost at most $2\ell + O(W)$. Observe that if the cycle P_F has all its edges in H then one candidate for P_H is P_F and we are done. If not, then there is at least one 2-ball whose deletion provokes actual deletion of edges from P_F (that is edges of P_F missing in the final H).

In the following, let uv be the root edge of the first 2-ball whose removal deletes edges from P_F (that is they are not added in H neither during the while loop nor the last step of the algorithm). Let G_i be the graph G just before the removal of $B_G^2(uv, 4W) \cap (N_G(u) \cup N_G(v))$, and G_{i+1} the one just after.

The rest of the discussion is done in G_i otherwise noted.

The proof is done as follows: we first show in Lemma 5 that any endpoint of a deleted edge (of P_F) belongs to an elementary cycle comprising the edge uv and of cost at most $6W$. We then show in Lemma 6 that we can construct cycles using a and/or b passing through the edge uv, effectively bounding $\delta_H^2(uv,a)$ and $\delta_H^2(uv,b)$ due to the addition of the shortest 2-path spanning tree rooted at uv. Finally we show in Lemma 7 that the union of a cycle passing through uv and a and another one passing through uv and b contains an elementary cycle joining a to b, its cost being at most the sums of the costs of the two original cycles.

Lemma 5. *Let $e = wt$ be an edge of $(G_i \setminus G_{i+1}) \setminus H$. Then in G_i both w and t are connected to uv by a cycle of cost at most $6W$.*

We now show that we can use this lemma to exhibit cycles going from a to uv and from b to uv.

From the vertices belonging to both $B_{G_i}^2(uv, 6W)$ and P_F we choose the ones which are the closest from a and b (we know that at least two of them exist because one edge was removed from P_F during step i of the loop). There are at maximum four of them (a_1, a_2, b_1, b_2), one for each sub-path P_F^i and each extremity $\{a, b\}$. Note that each extremity is connected to the root edge by an elementary cycle of cost at most $6W$. Two cases are possible (the placement of the vertices is shown on Fig. 3):

- **Case 1:** There are only two extremities (then they belong to the same sub-path) and their cycles which connect them to uv do not intersect the second subpath (w.l.o.g we can suppose it is a_1 and b_1).
- **Case 2:** There are more than two extremities: either some edges of the second path were removed or one of the cycles going from one of the extremities a_1 or b_1 to uv intersects the second path.

We show next that we can bound $\delta_H^2(uv,a)$ and $\delta_H^2(uv,b)$ with the help of the cycles connecting the endpoints and the path P_G. This is done with the two next lemmas.

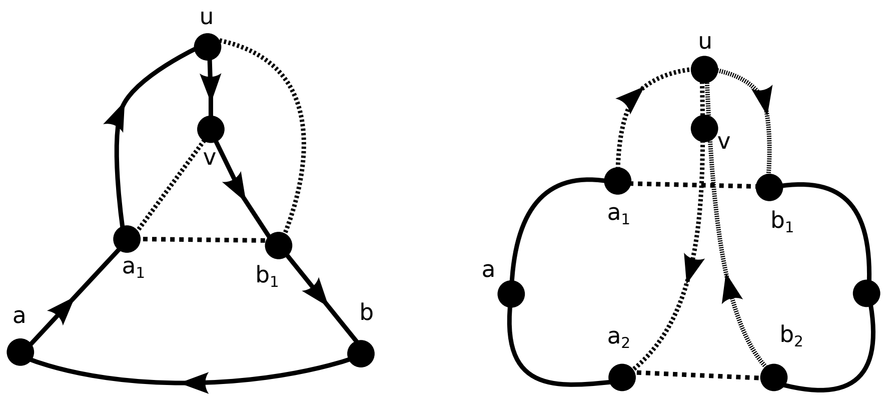

Fig. 3. Proof of Lemma 7: the two cases for the simple paths.

Lemma 6. *For any two vertices joined to the same edge uv by elementary cycles there is a simple path of cost at most the sum of the cycles' costs and passing through the edge uv.*

Lemma 7. *Let a, b be two vertices such that an elementary cycle of cost $\delta^2(a, b)$ has common vertices with some $B^2(uv, 6W)$. Then $\delta^2(a, uv)$ and $\delta^2(b, uv)$ are bounded by $\delta^2(a, b) + 12W$*

Property 1. Let uv be a non cut-edge of G and T be any 2-path spanning tree rooted at uv. Then, for all vertices a, b in $G[uv] \setminus \{u, v\}$, $\delta^2_G(a, b) \leqslant \delta^2_T(uv, a) + \delta^2_T(uv, b) - \omega(uv)$.

Proof. There is in T a cycle joining a to uv of cost $\delta^2_T(uv, a)$, and another one joining b to uv of cost $\delta^2_T(uv, b)$. Consider the subgraph P containing only the edges from these two cycles. The cost of P is $\omega(P) \leqslant \delta^2_T(uv, a) + \delta^2_T(uv, b) - \omega(uv)$ as edge uv is counted twice. It remains to show that P contains an elementary cycle between a and b. Note that since $a \notin \{u, v\}$, a has in P two vertex-disjoint paths leaving a and excluding edge uv: one is going to u, and one to v. Similarly for vertex b.

W.l.o.g. we can assume that a and b are not adjacent in P. Otherwise we can subdivide edge ab to obtain a new subgraph P'. Clearly, if P' contains an elementary cycle between a and b, then P too. Consider that one vertex z, outside a and b, is removed in P. From the remark above, in $P \setminus \{z\}$, there must exists a path leaving a and joining some vertex $w_a \in \{u, v\} \setminus \{z\}$ and one path leaving b and joining some vertex $w_b \in \{u, v\} \setminus \{z\}$. If $w_a = w_b$, then a and b are connected in $P \setminus \{z\}$. If $w_a \neq w_b$, then edge uv belongs to $P \setminus \{z\}$ since in this case $z \notin \{u, v\}$, and thus a path connected a to b in $P \setminus \{z\}$. By Menger's Theorem, P contains a 2-multipath between a and b. □

Lemma 8. *H is a 2-multipath $(2, 24W)$-spanner.*

Proof. If there is in F a path of cost $\delta^2(a,b)$ such that every edge of it is in H, then there is nothing to prove. If there is some removed edge, then we can identify the loop order i which removed the first edge, and we can associate the graph G_i just before the deletion performed in the second step of the loop (so P_F still completely exist in G_i). By virtue of Lemma 5 we can identify some root-edge uv and we know that there are some vertices of P_F linked to uv by an elementary cycle of length at most $6W$. Lemma 7 can then be applied, and so in G_i, $\delta^2_{G_i}(a,uv)$ and $\delta^2_{G_i}(b,uv)$ are both bounded by $\delta^2_{G_i}(a,b)+12W$. As the loop's first step is to build a shortest 2-path spanning tree rooted in uv we know that in H

$$\delta^2_H(a,uv) \leqslant \delta^2_{G_i}(a,uv) \leqslant \delta^2_{G_i}(a,b) + 12W$$

and the same for b. Property 1 can then be used in the 2-path spanning tree, to bound $\delta^2_H(a,b)$:

$$\delta^2_H(a,b) \leqslant \delta^2_H(a,uv) + \delta^2_H(b,uv) \leqslant 2 \cdot \delta^2_{G_i}(a,b) + 24W$$

Finally, as in G_i P_F still exists completely, we have that $\delta^2_{G_i}(a,b) = \delta^2_F(a,b)$, so

$$\delta^2_H(a,b) \leqslant 2 \cdot \delta^2_F(a,b) + 24W$$

□

4 Conclusion

We have introduced a natural generalization of spanner, the vertex-disjoint path spanners. We proved that there exists for multipath spanners a size-stretch trade-off similar to classical spanners. We also have presented a $O(k)$ round distributed algorithm to construct p-multipath $kp \cdot O(1+p/k)^{2k-1}$-spanners of size $\tilde{O}(p^2 n^{1+1/k})$, showing that the problem is *local*: it does not require communication between distant vertices.

Our construction is based on fault tolerant spanner. An interesting question is to know if better construction (in term of stretch) exists as suggested by our alternative construction for $p = 2$.

The most challenging question is to explicitly construct the p vertex-disjoint paths in the p-multipath spanner. This is probably as hard as constructing efficient routing algorithm from sparse spanner. We stress that there is a significant difference between proving the existence of short routes in a graph (or subgraph), and constructing and explicitly describing such short routes. For instance it is known (see [12]) that sparse spanners may exist whereas routing in the spanner can be difficult (in term of space memory and stretch of the routes).

References

1. N. Alon, S. Hoory, and N. Linial, *The Moore bound for irregular graphs*, Graphs and Combinatorics, 18 (2002), pp. 53–57.
2. I. Althöfer, G. Das, D. P. Dobkin, D. A. Joseph, and J. Soares, *On sparse spanners of weighted graphs*, Discr. & Comp. Geometry, 9 (1993), pp. 81–100.

3. L. Barenboim and M. Elkin, *Deterministic distributed vertex coloring in polylogarithmic time*, in 29^{th} ACM Symp. PODC, July 2010, pp. 410–419.
4. S. Baswana, A. Gaur, S. Sen, and J. Upadhyay, *Distance oracles for unweighted graphs: Breaking the quadratic barrier with constant additive error*, in 35^{th} Int'l Coll. on Automata, Lang. and Progr. (ICALP), LNCS 5125, 2008, p. 609.
5. S. Baswana and T. Kavitha, *Faster algorithms for approximate distance oracles and all-pairs small stretch paths*, in 47^{th} Annual IEEE Symp. on Foundations of Computer Science (FOCS), IEEE Comp. Soc. Press, Oct. 2006, pp. 591–602.
6. S. Chechik, M. Langberg, D. Peleg, and L. Roditty, *Fault tolerant spanners for general graphs*, SIAM Journal on Computing, 39 (2010), pp. 3403–3423.
7. L. J. Cowen and C. Wagner, *Compact roundtrip routing in directed networks*, in 19^{th} ACM Symp. PODC, 2000, pp. 51–59.
8. B. Derbel, C. Gavoille, D. Peleg, and L. Viennot, *On the locality of distributed sparse spanner construction*, in 27^{th} ACM Symp. PODC, 2008, pp. 273.
9. M. Dinitz and R. Krauthgamer, *Fault-tolerant spanners: Better and simpler*, Tech. Rep. 1101.5753v1 [cs.DS], arXiv, Jan. 2011.
10. C. Gavoille, Q. Godfroy, and L. Viennot, *Multipath spanners*, in 17^{th} Int'l Coll. SIROCCO, vol. 6058 of LNCS, Springer, June 2010, pp. 211–223.
11. C. Gavoille, Q. Godfroy, and L. Viennot, *Node-Disjoint Multipath Spanners and their Relationship with Fault-Tolerant Spanners*, HAL-00622915, Sep. 2011.
12. C. Gavoille and C. Sommer, *Sparse spanners vs. compact routing*, in 23^{rd} ACM Symp. SPAA, June 2011, pp. 225–234.
13. P. Jacquet and L. Viennot, *Remote spanners: what to know beyond neighbors*, in 23^{rd} IEEE International Parallel & Distributed Processing Symp. (IPDPS), IEEE Computer Society Press, May 2009.
14. N. Kushman, S. Kandula, D. Katabi, and B. M. Maggs, *R-bgp: Staying connected in a connected world*, in 4th Symp. on NSDI, 2007.
15. N. Linial, *Locality in distributed graphs algorithms*, SIAM Journal on Computing, 21 (1992), pp. 193–201.
16. L. Lovász, V. Neumann-Lara, and M. D. Plummer, *Mengerian theorems for paths of bounded length*, Periodica Mathematica Hungarica, 9 (1978), pp. 269–276.
17. S. Mueller, R. P. Tsang, and D. Ghosal, *Multipath routing in mobile ad hoc networks: Issues and challenges*, in Performance Tools and Applications to Networked Systems, Rev. Tutorial Lectures [from MASCOTS '03], pp. 209–234.
18. A. Nasipuri, R. Castañeda, and S. R. Das, *Performance of multipath routing for on-demand protocols in mobile ad hoc networks*, Mobile Networks and Applications, 6 (2001), pp. 339–349.
19. P. Pan, G. Swallow, and A. Atlas, *Fast Reroute Extensions to RSVP-TE for LSP Tunnels*. RFC 4090 (Proposed Standard), 2005.
20. D. Peleg, *Distributed Computing: A Locality-Sensitive Approach*, SIAM Monographs on Discrete Mathematics and Applications, 2000.
21. S. Pettie, *Low distortion spanners*, in 34^{th} Int'l Coll. on Automata, Languages and Programming (ICALP), vol. 4596 of LNCS, Springer, July 2007, pp. 78–89.
22. L. Pyber and Z. Tuza, *Menger-type theorems with restrictions on path lengths*, Discrete Mathematics, 120 (1993), pp. 161–174.
23. L. Roditty, M. Thorup, and U. Zwick, *Roundtrip spanners and roundtrip routing in directed graphs*, ACM Transactions on Algorithms, 3 (2008), Article 29.
24. J. W. Suurballe and R. E. Tarjan, *A quick method for finding shortest pairs of disjoint paths*, Networks, 14 (1984), pp. 325–336.
25. M. Thorup and U. Zwick, *Approximate distance oracles*, Journal of the ACM, 52 (2005), pp. 1–24.