LOWER BOUNDS FOR LOCAL MONOTONICITY RECONSTRUCTION FROM TRANSITIVE-CLOSURE SPANNERS*

ARNAB BHATTACHARYYA†, ELENA GRIGORESCU‡, MADHAV JHA§, KYOMIN JUNG¶, SOFYA RASKHODNIKOVA‡, AND DAVID P. WOODRUFF‖

Abstract. Given a directed graph $G = (V, E)$ and an integer $k \geq 1$, a *k-transitive-closure-spanner (k-TC-spanner)* of G is a directed graph $H = (V, E_H)$ that has (1) the same transitive-closure as G and (2) diameter at most k. Transitive-closure spanners are a common abstraction for applications in access control, property testing and data structures.

We show a connection between 2-TC-spanners and *local monotonicity filters*. A local monotonicity filter, introduced by Saks and Seshadhri (SIAM Journal on Computing, 2010), is a randomized algorithm that, given access to an oracle for an almost monotone function $f : \{1, 2, \ldots, m\}^d \to \mathbb{R}$, can quickly evaluate a related function $g : \{1, 2, \ldots, m\}^d \to \mathbb{R}$ which is guaranteed to be monotone. Furthermore, the filter can be implemented in a distributed manner. We show that an efficient local monotonicity filter implies a sparse 2-TC-spanner of the directed hypergrid, providing a new technique for proving lower bounds for local monotonicity filters. Our connection is, in fact, more general: an efficient local monotonicity filter for functions on any partially ordered set (poset) implies a sparse 2-TC-spanner of the directed acyclic graph corresponding to the poset.

We present nearly tight upper and lower bounds on the size of the sparsest 2-TC-spanners of the directed hypercube and hypergrid. These bounds imply stronger lower bounds for local monotonicity filters that nearly match the upper bounds of Saks and Seshadhri.

Key words. Property Testing, Property Reconstruction, Monotone Functions, Spanners, Hypercube, Hypergrid

AMS subject classification. 05C20, 68Q17

1. Introduction.

We show a connection between transitive-closure spanners and local filters. Let us start by defining these objects and explaining the context in which they originally arose.

1.1. Transitive-closure spanners.

Graph spanners were introduced by Awerbuch [4] and Peleg and Schäffer [21] in the context of distributed computing, and since then have found numerous applications, such as efficient routing [14, 15, 23, 25, 32], simulating synchronized protocols in unsynchronized networks [22], parallel and distributed algorithms for approximating shortest paths [12, 13, 17], and algorithms for distance oracles [5, 33]. Several variants of graph spanners have been defined. In this work, we focus on *transitive-closure spanners* that were formally introduced by Bhattacharyya *et al.* [7] as a common abstraction for applications in access control, property testing and data structures.

For a directed graph $G = (V, E)$ and two vertices $u, v \in V$, let $d_G(u, v)$ denote the smallest number of edges on a path in G from u to v.

DEFINITION 1.1 (TC-spanner). *Given a directed graph $G = (V, E)$ and an integer $k \geq 1$, a **k-transitive-closure-spanner** (k-**TC-spanner**) of G is a directed graph $H = (V, E_H)$ with the following properties.*

1. *E_H is a subset of the edges in the transitive closure of G.*
2. *For all vertices $u, v \in V$, if $d_G(u, v) < \infty$, then $d_H(u, v) \leq k$.*

---

*The preliminary version of the paper appeared in RANDOM 2010 [6].

†Princeton University, USA. Email: arnabb@princeton.edu. Research conducted when this author was at MIT CSAIL. Research supported in part by a DOE Computational Science Graduate Fellowship and NSF Awards 0514771, 0728645, and 0732334.

‡Georgia Institute of Technology, USA. Email: elena_g@csail.mit.edu. Research conducted when this author was at MIT CSAIL. Research supported in part by NSF grant CCR-0829672 and NSF award 1019343 to the Computing Research Association for the CI Fellows Project.

§Pennsylvania State University, USA. Email: {mxj201,sofya}@cse.psu.edu. Supported by NSF CAREER award 0845701.

¶KAIST, South Korea. Email:kyomin@kaist.edu. Supported by the Engineering Research Center of Excellence Program of Korea Ministry of Education, Science and Technology(MEST)/National Research Foundation of Korea(NRF)(Grant 2010-0001713).

‖IBM Almaden Research Center, USA. Email: dpwoodru@us.ibm.com.

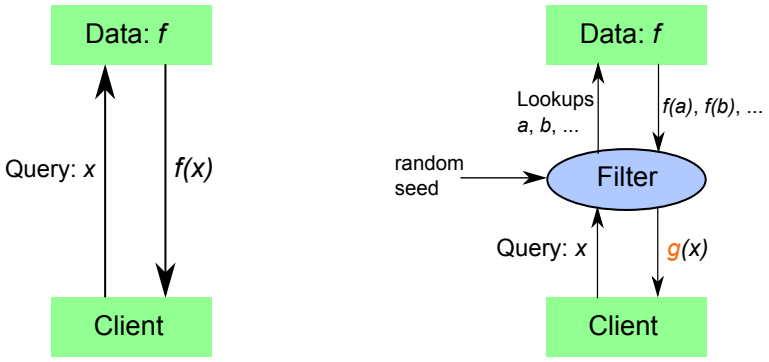

FIG. 1.1. *A property-preserving filter. Given a query x, the filter looks up a few values of the data function f and outputs $g(x)$, where the reconstructed function g satisfies a desired property.*

Thus, a k-TC-spanner is a graph with small diameter[1] that preserves the connectivity of the original graph. In the applications above, the goal is to find the sparsest k-TC-spanner for a given k and G. The number of edges in the sparsest k-TC-spanner of G is denoted by $S_k(G)$. We review previous work on bounding $S_k(G)$ for different families of graphs in Section 1.4. For a detailed description of recent results on TC-spanners, we refer the reader to the survey by Raskhodnikova [24].

1.2. Local property reconstruction. Property-preserving data reconstruction was introduced by Ailon *et al.* [1]. In this model, a reconstruction algorithm, called a *filter*, sits between a *client* and a *dataset*. A dataset is viewed as a function $f : \mathcal{D} \to \mathcal{R}$. The client accesses the dataset using *queries* of the form $x \in \mathcal{D}$ to the filter. Given a query x, the filter generates a small number of *lookups* $a \in \mathcal{D}$ to the dataset, from which it receives the values $f(a)$ and then computes a value $g(x)$, where g must satisfy some fixed structural property $\mathcal{P}$, such as being a monotone function. (See Figure 1.1 for an illustration.) Extending this notion, Saks and Seshadhri [26] defined local reconstruction. A filter is *local* if it allows for a local (or distributed) implementation: namely, if the output function g does not depend on the order of the queries.

DEFINITION 1.2 (Local filter). *A* local filter *for reconstructing property $\mathcal{P}$ is a randomized algorithm A that has oracle access to a function $f : \mathcal{D} \to \mathcal{R}$ and to an auxiliary random string ρ (the "random seed"), and takes as input a query $x \in \mathcal{D}$. For fixed f and ρ, algorithm A runs deterministically on input x and produces an output $A_{f,\rho}(x) \in \mathcal{R}$. (Note that a local filter has no internal state to store previously made queries.) The function $g(x) = A_{f,\rho}(x)$ output by the filter must obey the following conditions.*

- *For each f and ρ, the function g must satisfy $\mathcal{P}$.*
- *If f satisfies $\mathcal{P}$, then g must be identical to f with probability at least $1-\delta$, for some* error probability *$\delta \leq 1/3$. The probability is taken over ρ.*

When answering a query $x \in \mathcal{D}$, a filter may look up values of f at domain points of its choice using its oracle. The *lookup complexity* of a local filter A is the maximum number of lookups performed by A for any f and ρ and for any input query x. A local filter is *nonadaptive* if its lookups on input query x do not depend on answers given by the oracle.

Saks and Seshadhri actually considered more restrictive filters which required that g be sufficiently close to f.

DEFINITION 1.3 (Distance-respecting local filter). *For a function $B : \mathbb{Z}^+ \to \mathbb{Z}^+$, a* distance-respecting local filter with error blowup *$B(n)$ is a local filter in the sense of Definition 1.2 for which the following holds. With high probability (over the choice of ρ), $Dist(g, f) \leq B(n) \cdot Dist(f, \mathcal{P})$, where $Dist(g, f)$ is the number of points in the domain on which f and g differ and $Dist(f, \mathcal{P})$ is $\min_{g \in \mathcal{P}} Dist(g, f)$.*

[1] In this work and other papers on transitive-closure spanners, the diameter of a graph is defined to be the maximum distance from u to v over all nodes u and v, such that there is a path from u to v in the graph.

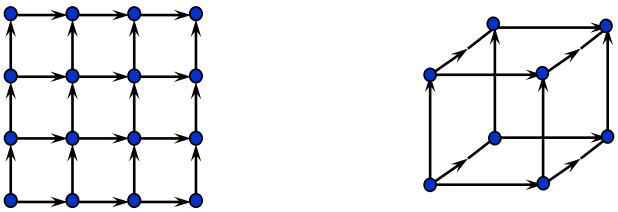

FIG. 1.2. *Examples of hypergrids: $\mathcal{H}_4^2$ (left) and $\mathcal{H}_3$ (right). All edges are directed towards vertices with larger coordinates.*

Local monotonicity filters. The most studied property in the local reconstruction model is monotonicity of functions [26, 1]. To define it, consider an n-element poset V_n and let $G_n = (V_n, E)$ be the relation graph, *i.e.,* the Hasse diagram, for V_n. A function $f : V_n \to \mathbb{R}$ is called *monotone* if $f(x) \leq f(y)$ for all $(x, y) \in E$. We focus on posets whose relation graph is a *directed hypergrid*. The directed hypergrid, denoted $\mathcal{H}_m^d$ and illustrated in Figure 1.2, has vertex set[2] $[m]^d$ and edge set $\{(x, y) : \exists$ unique $i \in [d]$ such that $y_i - x_i = 1$ and for $j \neq i, y_j = x_j\}$. For the special case $m = 2$, the hypergrid $\mathcal{H}_2^d$ is called the *hypercube* and is also denoted by $\mathcal{H}^d$. A monotonicity filter needs to ensure that the output function g is monotone. For instance, if G_n is a directed line, $\mathcal{H}_n^1$, the filter needs to ensure that the output sequence specified by g is sorted.

To motivate monotonicity filters for hypergrids, consider the scenario of rolling admissions: An admissions office assigns d scores to each application, such as the applicant's GPA, SAT results, essay quality, etc. Based on these scores, some complicated (third-party) algorithm outputs the probability that a given applicant should be accepted. The admissions office wants to make sure "on the fly" that strictly better applicants are given higher probability, that is, probabilities are *monotone* in scores. A hypergrid monotonicity filter may be used here. If it is local, it can be implemented in a distributed manner with an additional guarantee that every copy of the filter will correct to the same monotone function of the scores. This can be done by supplying the same random seed to each copy of the filter.

Saks and Seshadhri [26] give a distance-respecting local monotonicity filter for the directed hypergrid $\mathcal{H}_m^d$ that makes $(\log m)^{O(d)}$ lookups per query for $m \geq 3$ (where, as always in this article, the logarithm has base 2). No nontrivial monotonicity filter for the hypercube $\mathcal{H}^d$, meaning a filter performing $o(2^d)$ lookups per query, is known. One of the monotonicity filters of Ailon *et al.* [1] is a local filter for the directed line $\mathcal{H}_m^1$ with $O(\log m)$ lookups per query (but a worse error blowup than in [26]). As observed in [26], this upper bound is tight. Saks and Seshadhri [26] also present a lower bound of $2^{\beta d}$ on the number of lookups per query for a *distance-respecting* local monotonicity filter on $\mathcal{H}^d$ with *error blowup* $2^{\beta d}$, where β is a sufficiently small constant. Notably, all known local monotonicity filters are *nonadaptive*.

1.3. Our contributions. The contributions of this work fall into two categories.

1. We show that an efficient local monotonicity filter implies a sparse 2-TC-spanner of the directed hypergrid, providing a new technique for proving lower bounds for local monotonicity filters.

2. We present nearly tight upper and lower bounds on the size of the sparsest 2-TC-spanners of the directed hypercube and hypergrid. These bounds imply stronger lower bounds for local monotonicity filters for these graphs that, for nonadaptive filters and for filters that lookup function values only on points comparable to x on every query x, nearly match the upper bounds by Saks and Seshadhri [26]. (Two nodes x, y are called *comparable* if x is reachable from y or y is reachable from x; otherwise, they are *incomparable*.)

[2]For a positive integer m, we denote $\{1, 2, \ldots, m\}$ by $[m]$.

1.3.1. Our lower bounds for local monotonicity reconstruction. We show how to construct sparse 2-TC-spanners from local monotonicity filters with low lookup complexity. These constructions, together with our lower bounds on the size of 2-TC-spanners of the hypergrid and the hypercube, stated in Section 1.3.2, imply lower bounds on lookup complexity of local monotonicity filters for these graphs with arbitrary error blowup. Table 1.1 summarizes our results from this section.

We state the properties of our transformations from nonadaptive and adaptive filters separately.

THEOREM 1.4 (From nonadaptive filters to 2-TC-spanners). *Let $G_n = (V_n, E_n)$ be a poset on n nodes. Suppose there is a* nonadaptive *local monotonicity filter A for G_n that looks up at most $\ell(n)$ values on any query and has* error *probability at most δ. Then there is a 2-TC-spanner of G_n with $O(n\ell(n) \cdot \lceil \frac{\log n}{\log(1/\delta)} \rceil)$ edges.*

Next theorem applies even to *adaptive* local monotonicity filters. It takes into account how many lookups on query x are to nodes that are *incomparable* to x. In particular, if there are no such lookups, then the constructed 2-TC-spanner is of the same size as in Theorem 1.4.

THEOREM 1.5 (From adaptive filters to 2-TC-spanners). *Let $G_n = (V_n, E_n)$ be a poset on n nodes. Suppose there is an (*adaptive*) local monotonicity filter A for G_n that, for every query $x \in V_n$, looks up at most $\ell_1(n)$ vertices comparable to x and at most $\ell_2(n)$ vertices incomparable to x, and has* error *probability at most δ. Then there is a 2-TC-spanner of G_n with $O(n\ell_1(n) \cdot 2^{\ell_2(n)} \lceil \frac{\log n}{\log(1/\delta)} \rceil)$ edges.*

In Theorems 1.4 and 1.5, when δ is sufficiently small, the bounds on the 2-TC-spanner size become $O(n\ell(n))$ and $O(n\ell_1(n) \cdot 2^{\ell_2(n)})$, respectively. As mentioned earlier, all known monotonicity filters are nonadaptive. It is an open question whether it is possible to give a transformation from adaptive local monotonicity filters to 2-TC-spanners without incurring an exponential dependence on the number of lookups made to points incomparable to the query point. We do not know whether this dependence is an artifact of the proof or an indication that lookups to incomparable points might be helpful for adaptive local monotonicity filters.

In Theorems 1.8 and 1.9, stated in Section 1.3.2, we present nearly tight bounds on the size of the sparsest 2-TC-spanners of the hypercube and the hypergrid. Theorems 1.4 and 1.5, together with the lower bounds in Theorems 1.8 and 1.9, imply the following lower bounds on the lookup complexity of local monotonicity filters for these graphs, with arbitrary error blowup.

COROLLARY 1.6. *Consider a nonadaptive local monotonicity filter with constant error probability δ. If the filter is for functions $f : \mathcal{H}_m^d \to \mathbb{R}$ (where $1 < d < \frac{2 \log m}{\log \log m}$), it must perform*

$$\Omega\left(\frac{\log^{d-1} m}{d^d (2 \log \log m)^{d-1}}\right)$$

lookups per query. If the filter is for functions $f : \mathcal{H}^d \to \mathbb{R}$, it must perform $\Omega(\frac{2^{\alpha d}}{d})$ lookups per query, where $\alpha \geq 0.1620$.

COROLLARY 1.7. *Consider an (adaptive) local monotonicity filter with constant error probability δ, that for every query $x \in V_n$, looks up at most ℓ_2 vertices incomparable to x. If the filter is for functions $f : \mathcal{H}_m^d \to \mathbb{R}$ (where $1 < d < \frac{2 \log m}{\log \log m}$), it must perform*

$$\Omega\left(\frac{\log^{d-1} m}{2^{\ell_2} d^d (2 \log \log m)^{d-1}}\right)$$

lookups to vertices comparable to x per query x. If the filter is for functions $f : \mathcal{H}^d \to \mathbb{R}$, it must perform $\Omega(\frac{2^{\alpha d - \ell_2}}{d})$ comparable lookups, where $\alpha \geq 0.1620$.

Prior to this work, no lower bounds for monotonicity filters on $\mathcal{H}_m^d$ with dependence on both m and d were known. Unlike the bound of Saks and Seshadhri [26], our lower bounds hold for any error blowup and for filters which are not necessarily distance-respecting.

	Type	Lower Bound	Blowup $B(m,d)$	Reference
Low-Dimensional Hypergrid $\mathcal{H}_m^d$ $\left(1 < d < \frac{2\log m}{\log\log m}\right)$	nonadaptive	$\Omega(\frac{\log^{d-1} m}{d^d(2\log\log m)^{d-1}})$	arbitrary	Corollary 1.6
	adaptive	$\Omega(\frac{\log^{d-1} m}{2^{\ell_2} d^d(2\log\log m)^{d-1}})$		Corollary 1.7
Hypercube $\mathcal{H}^d$	nonadaptive	$\Omega(\frac{2^{\alpha d}}{d})$	arbitrary	Corollary 1.6
	adaptive	$\Omega(\frac{2^{\alpha d-\ell_2}}{d})$		Corollary 1.7
	adaptive	$\Omega(2^{\beta d})$	$< 2^{\beta d}$	[26]

TABLE 1.1

Summary of our lower bounds for lookup complexity of local monotonicity filters. The parameter $\alpha \approx 0.1620$ and the parameter β is a sufficiently small constant (see [26]). The only known local filter for monotonicity on hypergrids (from [26]) is nonadaptive and makes $(\log m)^{O(d)}$ lookups for $m \geq 3$. In adaptive lower bounds, ℓ_2 denotes the maximum number of incomparable elements looked up on a single query (see Theorem 1.5).

Our bounds are tight for nonadaptive filters. Specifically, for the hypergrid $\mathcal{H}_m^d$ of constant dimension d and $m \geq 3$, the number of lookups is $(\log m)^{\Theta(d)}$, and for the hypercube $\mathcal{H}^d$, it is $2^{\Theta(d)}$ for any error blowup.

Testers vs. filters. Bhattacharyya *et al.* [7] obtained monotonicity testers from 2-TC-spanners. Unlike in the application to monotonicity testing, here we use *lower bounds* on the size of 2-TC-spanners to prove *lower bounds* on complexity of local monotonicity filters. Lower bounds on the size of 2-TC-spanners do not imply corresponding lower bounds on monotonicity testers. *E.g.,* the best monotonicity tester on $\mathcal{H}^d$ runs in $O(d^2)$ time [18, 16], while, as shown in Theorem 1.9, every 2-TC-spanner of $\mathcal{H}^d$ must have size exponential in d.

1.3.2. Our bounds on the size of 2-TC-spanners of the hypercube and the hypergrid. Table 1.2 summarizes our results from this section. Our main theorem, proved in Section 4, gives a set of explicit bounds on $S_2(\mathcal{H}_m^d)$.

THEOREM 1.8 (Hypergrid). *Let $S_2(\mathcal{H}_m^d)$ denote the number of edges in the sparsest 2-TC-spanner of $\mathcal{H}_m^d$. Then*

$$\Omega\left(\frac{m^d \log^d m}{(2d\log\log m)^{d-1}}\right) \;=\; S_2(\mathcal{H}_m^d) \;\leq\; m^d \log^d m.$$

The upper bound holds for all $m \geq 3$ while the lower bound is interesting (better than naive bounds, see Table 1.2) for $d < \frac{2\log m}{\log\log m}$.

The upper bound in Theorem 1.8 follows from a general construction of k-TC-spanners of graph products for arbitrary $k \geq 2$, presented in Section 4.1. The lower bound is the most technically difficult part of our work. It is proved by a reduction of the 2-TC-spanner construction for $[m]^d$ to that for the $2 \times [m]^{d-1}$ grid and then directly analyzing the number of edges required for a 2-TC-spanner of $2 \times [m]^{d-1}$. We show a tradeoff between the number of edges in the 2-TC-spanner of the $2 \times [m]^{d-1}$ grid that stay within the hyperplanes $\{1\} \times [m]^{d-1}$ and $\{2\} \times [m]^{d-1}$ versus the number of edges that cross from one hyperplane to the other. The proof proceeds in multiple stages. Assuming an upper bound on the number of edges staying within the hyperplanes, each stage is shown to contribute a substantial number of new edges crossing between the hyperplanes. The proof of this tradeoff lemma is already nontrivial for $d = 2$ and is presented first in Section 4.2.1. The proof for $d > 2$ is presented in Section 4.2.2.

While Theorem 1.8 is most useful when m is large and d is small, in Section 6, we present bounds on $S_2(\mathcal{H}_m^d)$ which are optimal up to a factor of d^{2m} and thus supersede the bounds from Theorem 1.8 when m is small. These bounds are stated in Theorem 6.2. In particular, for constant m, our upper and lower bounds differ only by a factor polynomial in the dimension d. The general form of these bounds is a somewhat complicated combinatorial expression, but they can be estimated numerically. Specifically, $S_2(\mathcal{H}_m^d) = 2^{c_m d}\,\mathrm{poly}(d)$,

	This Work		Naive bounds	
	Lower Bound	Upper Bound	Lower Bound	Upper Bound
Hypergrid $\mathcal{H}_m^d$	$\Omega\left(\frac{m^d \log^d m}{(2d \log\log m)^{d-1}}\right)$ if $d < \frac{\log m}{2 \log\log m}$	$m^d \log^d m$	$dm^{d-1}(m-1)$	$\left(\frac{m^2+m}{2}\right)^d - m^d$
	$\Omega(2^{c_m d})$	$O(d^{2m} 2^{c_m d})$		
Hypercube $\mathcal{H}^d$	$\Omega(2^{cd})$	$O(d^3 2^{cd})$	$2^{d-1}d$	$3^d - 2^d$

TABLE 1.2

Comparison of our results on the size of 2-TC-spanners of the hypergrid and the hypercube with naive bounds. Constant $c \approx 1.1620$ while c_m depends on m. No nontrivial bounds were known prior to this work.

where $c_2 \approx 1.1620$ and $c_3 \approx 1.85$, both significantly smaller than the exponents corresponding to the transitive closure sizes for appropriate m.

First, we consider the special case of $m = 2$ (i.e., the hypercube) in Section 5 and then generalize the arguments to larger m in Section 6. Specifically, we obtain the following theorem for the hypercube.

THEOREM 1.9 (Hypercube). *Let $S_2(\mathcal{H}^d)$ be the number of edges in the sparsest 2-TC-spanner of $\mathcal{H}^d$. Then $\Omega(2^{cd}) = S_2(\mathcal{H}^d) = O(d^3 2^{cd})$, where $c \approx 1.1620$.*

We prove the theorem by giving nearly matching upper and lower bounds on $S_2(\mathcal{H}^d)$ in terms of an expression with binomial coefficients, and later numerically estimating the value of the expression. We prove the upper bound in Theorem 1.9 by presenting a randomized construction of a 2-TC-spanner of the directed hypercube. Curiously, even though the upper and lower bounds above differ by a factor of $O(d^3)$, we can show that our construction yields a 2-TC-spanner of $\mathcal{H}^d$ of size within $O(d^2)$ of the optimal.

As a comparison point for our bounds, note that the obvious bounds on $S_2(\mathcal{H}^d)$ are the number of edges in the d-dimensional hypercube, $2^{d-1}d$, and the number of edges in the transitive closure of $\mathcal{H}^d$, which is $3^d - 2^d$. (An edge in the transitive closure of $\mathcal{H}^d$ has 3 possibilities for each coordinate: both endpoints are 0, both endpoints are 1, or the first endpoint is 0 and the second is 1. This includes self-loops, so we subtract the number of vertices in $\mathcal{H}^d$ to get the desired quantity.) Thus, $2^{d-1}d \le S_2(\mathcal{H}^d) \le 3^d - 2^d$. Similarly, the straightforward bounds on the number of edges in a 2-TC-spanner of $\mathcal{H}_m^d$ in terms of the number of edges in the directed hypergrid and in its transitive closure are $dm^{d-1}(m-1)$ and $\left(\frac{m^2+m}{2}\right)^d - m^d$, respectively.

1.4. Previous work on bounding S_k for other families of graphs. Thorup [29] considered a special case of TC-spanners of graphs G that have at most twice as many edges as G, and conjectured that for all directed graphs G on n nodes there are such k-TC-spanners with k polylogarithmic in n. He proved this for planar graphs [30], but later Hesse [19] gave a counterexample to Thorup's conjecture for general graphs. For all small $\epsilon > 0$, he constructed a family of graphs with $n^{1+\epsilon}$ edges for which all n^ϵ-TC-spanners require $\Omega(n^{2-\epsilon})$ edges.

TC-spanners were also studied, implicitly in [34, 10, 9, 2, 11, 16, 3, 20, 28] and explicitly in [8, 31], for simple families of graphs, such as directed trees. For the directed line, Chandra, Fortune and Lipton [10, 9] implicitly (in the context of work on circuit complexity) expressed $S_k(\mathcal{H}_n^1)$ in terms of the inverse Ackermann function. (See Section 2.1 for a definition.) The construction of TC-spanners of the directed line in [10, 9] is implicit; it appears explicitly, for example, in the survey on TC-spanners by Raskhodnikova [24]. Narasimhan and Smid [20] and Solomon [28] consider a graph object, related to TC-spanners, called *T-monotone 1-spanners*, where T is an undirected tree. When T is an undirected path (that is, $\mathcal{H}_n^1$ without edge orientations), a T-monotone 1-spanner of diameter k directly corresponds to a k-TC-spanner of the directed line $\mathcal{H}_n^1$.

LEMMA 1.10 ([10, 9, 2, 20, 24, 28]). *Let $S_k(\mathcal{H}_n^1)$ be the number of edges in the sparsest k-TC-spanner of the directed line $\mathcal{H}_n^1$. Then $S_2(\mathcal{H}_n^1) = \Theta(n \log n)$, $S_3(\mathcal{H}_n^1) =$*

$\Theta(n \log\log n)$, $S_4(\mathcal{H}^1_n) = \Theta(n \log^* n)$ *and, more generally,* $S_k(\mathcal{H}^1_n) = \Theta(n\lambda_k(n))$ *for all* $k \geq 2$*, where* $\lambda_k(n)$ *is the inverse Ackermann function.*

The same bound holds for directed trees [2, 11, 31, 28]. An $O(n \log n \cdot \lambda_k(n))$ bound on S_k for H-minor-free graph families (e.g., bounded genus and bounded tree-width graphs) was given in [7].

2. Preliminaries. For $x \in \{0,1\}^d$, we use $|x|$ to denote the weight of x, that is, the number of non-zero coordinates in x. A level i in a hypercube contains all vertices of weight i. The partial order $\preceq$ on the hypergrid $\mathcal{H}^d_m$ is defined as follows: $x \preceq y$ for two vertices $x, y \in [m]^d$ iff $x_i \leq y_i$ for all $i \in [d]$. Similarly, $x \prec y$ if x and y are distinct vertices in $[m]^d$ satisfying $x \preceq y$. More generally, we identify each poset with its relation graph, and denote its partial order on the vertices by $\prec$. Vertices x and y are *comparable* if either y is *above* x (that is, $x \preceq y$) or y is *below* x (that is, $y \preceq x$). We denote a path from v_1 to v_ℓ, consisting of edges $(v_1, v_2), (v_2, v_3), \ldots, (v_{\ell-1}, v_\ell)$ by $(v_1, \ldots, v_\ell)$.

2.1. The inverse Ackermann hierarchy. Our definition of inverse Ackermann functions is derived from the discussion in [27]. For a given function $f : \mathbb{R}^{\geq 0} \to \mathbb{R}^{\geq 0}$, such that $f(x) < x$ for all $x > 2$, define the function $f^*(x) : \mathbb{R}^{\geq 0} \to \mathbb{R}^{\geq 0}$ to be the following:

$f^*(x) = \min\{k \in \mathbb{Z}^{\geq 0} : f^{(k)}(x) \leq 2\}$, where $f^{(k)}$ denotes f composed with itself k times.

We note that the solution to the following recursion:

$$T(n) \leq \begin{cases} 0 & \text{if } n \leq 2 \\ a \cdot n + \frac{n}{f(n)} \cdot T(f(n)) & \text{if } n > 2 \end{cases}$$

is $T(n) = a \cdot n \cdot f^*(n)$. This follows from the fact that $f^*(f(n)) = f^*(n) - 1$ for $n > 2$.

We define the inverse Ackermann hierarchy to be a sequence of functions $\lambda_k(\cdot)$ for $k \geq 0$. As the base cases, we have $\lambda_0(n) = n/2$ and $\lambda_1(n) = \sqrt{n}$. For $j \geq 2$, we define $\lambda_j(n) = \lambda^*_{j-2}(n)$. Thus, $\lambda_2(n) = \Theta(\log n)$, $\lambda_3(n) = \Theta(\log\log n)$ and $\lambda_4(n) = \Theta(\log^* n)$. Note that the $\lambda_k(\cdot)$ functions defined here coincide (upto constant additive differences) with the $\lambda(k, \cdot)$ functions in [2] although they were formulated a bit differently there.

Finally, we define the inverse Ackermann function $\alpha(\cdot)$ to be $\alpha(n) = \min\{k \in \mathbb{Z}^{\geq 0} : \lambda_{2k}(n) \leq 3\}$.

3. From local monotonicity filters to 2-TC-spanners. In this section, we prove Theorems 1.4 and 1.5 that summarize the properties of our transformations from local monotonicity filters to 2-TC-spanners. The main idea in both transformations is to construct a graph H in which each vertex v is incident to every vertex looked up by the filter on query x and random string ρ for a small subset of random strings. To prove that H is a 2-TC-spanner, we show that for every pair of comparable vertices $x \prec y$, the associated sets of lookup vertices must have a nonempty intersection. If this does not hold, we can find a random string ρ and construct a function h, such that the output function g of the filter on input h and random seed ρ is not monotone: namely, $g(x) = 0$ and $g(y) = 1$.

3.1. From nonadaptive local monotonicity filters to 2-TC-spanners.

THEOREM 1.4 (restated). *Let* $G_n = (V_n, E)$ *be a poset on* n *nodes. Suppose there is a* nonadaptive *local monotonicity filter* A *for* G_n *that looks up at most* $\ell(n)$ *values on any query and has* error *probability at most* δ*. Then there is a 2-TC-spanner of* G_n *with* $O(n\ell(n) \cdot \lceil \frac{\log n}{\log(1/\delta)} \rceil)$ *edges.*

Proof. Let A be a local filter given by the statement of the theorem. Let $\mathcal{F}$ be the set of pairs (x, y) with x, y in V_n such that $x \prec y$. Then $\mathcal{F}$ is of size at most $\binom{n}{2}$. Given $(x, y) \in \mathcal{F}$, let $\mathsf{cube}(x, y)$ be the set $\{z \in V_n : x \preceq z \preceq y\}$. Define function $f^{(x,y)}(v)$ to be 1 on all $v \succeq x$, and 0 everywhere else. Also, define function $f^{(\overline{x,y})}(v)$, which is identical to $f^{(x,y)}(v)$ for all $v \notin \mathsf{cube}(x, y)$ and 0 for $v \in \mathsf{cube}(x, y)$. Functions $f^{(x,y)}$ and $f^{(\overline{x,y})}$ are illustrated in Figure 3.1. For all $(x, y) \in \mathcal{F}$, both functions are monotone.

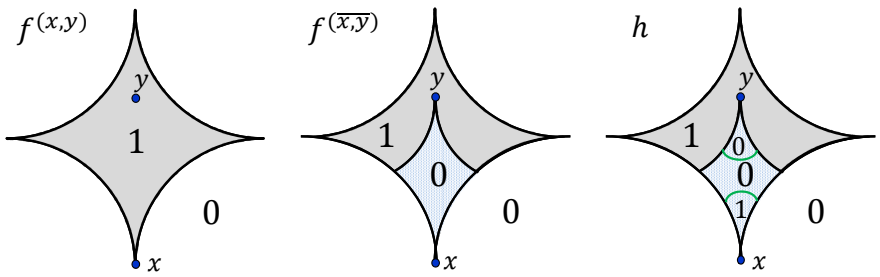

FIG. 3.1. *Functions defined in the proof of Theorem 1.4:* $f^{(x,y)}$, $f^{(\overline{x,y})}$ *and* h. *Observe that* $f^{(x,y)}(x) = f^{(x,y)}(y) = 1$; $f^{(\overline{x,y})}(x) = f^{(\overline{x,y})}(y) = 0$; $h(x) = 1$ *and* $h(y) = 0$.

Let A_ρ be the deterministic algorithm which runs A with the random seed fixed to ρ. We say a string ρ is *good* for $(x, y) \in \mathcal{F}$ if the filter A_ρ on input $f^{(x,y)}$ returns $g = f^{(x,y)}$ *and* on input $f^{(\overline{x,y})}$ returns $g = f^{(\overline{x,y})}$.

Now we show that there exists a set S of size $s = \lceil \frac{2 \log n}{\log(1/2\delta)} \rceil$, consisting of strings A uses as random seeds, such that for every $(x, y) \in \mathcal{F}$, some string $\rho \in S$ is good for (x, y). We choose S by picking strings used as random seeds uniformly and independently at random. Since A has error probability at most δ, we know that for every monotone f, with probability at least $1 - \delta$ (with respect to the choice of ρ), the function $A_{f,\rho}$ is identical to f. Then for a fixed pair $(x, y) \in \mathcal{F}$ and a uniformly random string ρ,

$$\begin{aligned}\Pr[\rho \text{ is not } \textit{good} \text{ for } (x,y)] \leq \quad & \Pr[A_\rho \text{ on input } f^{(x,y)} \text{ fails to output } f^{(x,y)}] \\ & + \Pr[A_\rho \text{ on input } f^{(\overline{x,y})} \text{ fails to output } f^{(\overline{x,y})}] \leq 2\delta.\end{aligned}$$

Since strings in S are chosen independently, $\Pr[\text{no } \rho \in S \text{ is good for } (x, y)] \leq (2\delta)^s$. For $s = \lceil \frac{2 \log n}{\log(1/2\delta)} \rceil$, this expression is equal to $\frac{1}{n^2} < \frac{1}{|\mathcal{F}|}$. By a union bound over $\mathcal{F}$,

$$\Pr[\text{for some } (x, y) \in \mathcal{F}, \text{ no } \rho \in S \text{ is good for } (x, y)] < 1.$$

Thus, there exists a set S with the required properties.

We construct our 2-TC-spanner $H = (V_n, E_H)$ of G_n using the set S described above. Let $\mathcal{N}_\rho(x)$ be the set consisting of x and all vertices looked up by A_ρ on query x. (Note that, given x and ρ, the lookups made by the algorithm are the same for all input functions f, since A is nonadaptive.) For each string $\rho \in S$ and each vertex $x \in V_n$, connect x to all comparable vertices in $\mathcal{N}_\rho(x)$ (other than itself) and orient these edges according to the partial order of G_n: that is, from smaller to larger elements. (Recall that we identify poset elements with the corresponding vertices of the relation graph.)

We prove H is a 2-TC-spanner of G_n as follows. Suppose not, *i.e.,* there exists $(x, y) \in \mathcal{F}$ with no path of length at most 2 in H from x to y. Then we will show that for some input function $h(v)$ and some random seed ρ, the output function $A_{h,\rho}(v)$ is not monotone, reaching a contradiction. Consider $\rho \in S$ which is *good* for (x, y). Define function h by setting $h(v) = f^{(x,y)}(v)$ for all $v \notin \mathsf{cube}(x, y)$. Then $h(v) = f^{(\overline{x,y})}(v)$ for all $v \notin \mathsf{cube}(x, y)$, by the definition of $f^{(\overline{x,y})}$. For a vertex $v \in \mathsf{cube}(x, y)$, set $h(v)$ to 1 if $v \in \mathcal{N}_\rho(x)$ and to 0 if $v \in \mathcal{N}_\rho(y)$. All unassigned values are set to 0. By the assumption above, $\mathcal{N}_\rho(x) \cap \mathcal{N}_\rho(y)$ does not contain any vertices in $\mathsf{cube}(x, y)$. Therefore, h is well-defined. See the third item in Figure 3.1 for an illustration of h. Since ρ is *good* for (x, y) and h is identical to $f^{(x,y)}$ for all lookups performed by A_ρ on query x, the output $A_\rho(x) = h(x) = 1$. Similarly, h is identical to $f^{(\overline{x,y})}$ for all lookups performed by A_ρ on query y and, consequently, $A_\rho(y) =$

$h(y) = 0$. But $x \prec y$, so $A_{h,\rho}(v)$ is not monotone, which contradicts the fact that A is a local monotonicity filter.

The number of edges in H is at most

$$\sum_{x \in V_n, \rho \in S} |\mathcal{N}_\rho(x)| \leq n \cdot \ell(n) \cdot s = n\ell(n) \cdot \left\lceil \frac{2 \log n}{\log(1/2\delta)} \right\rceil . \qquad \square$$

3.2. From adaptive local monotonicity filters to 2-TC-spanners. The complication in the transformation from an adaptive filter is that the set of vertices looked up by the filter depends on the oracle that the filter is invoked on.

THEOREM 1.5 (restated). *Let $G_n = (V_n, E)$ be a poset on n nodes. Suppose there is an (*adaptive*) local monotonicity filter A for G_n that, for every query $x \in V_n$, looks up at most $\ell_1(n)$ vertices comparable to x and at most $\ell_2(n)$ vertices incomparable to x, and has* error *probability at most δ. Then there is a 2-TC-spanner of G_n with $O(n\ell_1(n) \cdot 2^{\ell_2(n)\lceil \frac{\log n}{\log(1/\delta)} \rceil})$ edges.*

Proof. Define $\mathcal{F}$, $f^{(x,y)}$, $f^{(\overline{x,y})}$, A_ρ and S as in the proof of Theorem 1.4. As before, for each $x \in V_n$, we define sets $\mathcal{N}_\rho(x)$, and construct the 2-TC-spanner H by connecting each x to comparable elements in $\mathcal{N}_\rho(x)$ for all $\rho \in S$ and orienting the edges according to the partial order of G_n: from smaller to larger elements. However, now $\mathcal{N}_\rho(x)$ is a union of several sets $\mathcal{N}_\rho^{b,w}(x)$, indexed by $b \in \{0,1\}$ and $w \in \{0,1\}^{\ell_2(n)}$. For each $x \in V_n$, $b \in \{0,1\}$ and $w \in \{0,1\}^{\ell_2(n)}$, let $\mathcal{N}_\rho^{b,w}(x) \subseteq V_n$ be the set consisting of x and all vertices looked up by A_ρ on query x, assuming that the oracle answers all lookups as follows. When a lookup y is comparable to x, it answers 0 if $y \prec x$, b if $y = x$, and 1 if $x \prec y$. Otherwise, if y is the i'th lookup made to an incomparable vertex for some $i \in [\ell_2]$, it answers $w[i]$. As we mentioned, $N_\rho(x)$ is the union of the sets $N_\rho^{b,w}(x)$ over all $b \in \{0,1\}$ and $w \in \{0,1\}^{\ell_2(n)}$. This completes the description of $\mathcal{N}_\rho(x)$ and the construction of H.

We demonstrate that H is a 2-TC-spanner as in the proof of Theorem 1.4. In particular, the pair of vertices $x \prec y$, the string ρ and the function h are defined as before, and the contradiction is reached by demonstrating that $A_{h,\rho}(x) = h(x) = 1$ and $A_{h,\rho}(y) = h(y) = 0$. The only difference is in the argument that h is identical to $f^{(x,y)}$ for all lookups performed by A_ρ on query x, and to $f^{(\overline{x,y})}$ for all lookups on query y. The caveat is that an adaptive local filter might choose lookups based on the answers to previous lookups.

First, consider the behavior of A_ρ on query x. Since $h(v)$ may be different from $f^{(x,y)}(v)$ only for vertices v in $\mathsf{cube}(x,y)$, but not in $\mathcal{N}_\rho(x)$, the filter A_ρ can detect a difference between the two functions only if it looks up such a vertex. Suppose, for the sake of contradiction, that A_ρ looks up a vertex like that on query x, and let v be the the first such lookup. The filter receives the following answers to the lookups preceding v from the oracle for function h: 1 if the lookup is x or above x, and 0 if the lookup is below x or incomparable to x. Let w be a string of ℓ_2 zeros. Then $v \in \mathcal{N}_\rho^{1,w}(x) \subseteq \mathcal{N}_\rho(x)$, a contradiction.

Second, consider the behavior of A_ρ on query y. Suppose, for the sake of contradiction, that A_ρ on query y looks up a vertex $\mathsf{cube}(x,y)$, but not in $\mathcal{N}_\rho(y)$, and let v be the first such lookup. The filter receives the following answers to the lookups preceding v from the oracle for function h: 1 if the lookup is above y, 0 if the lookup is y or below y, and either 1 or 0 if it is incomparable to y (depending on whether it is above x or incomparable to x). Consider a binary string recording the answers on lookups to vertices incomparable to y, prior to the lookup v. Append zeros at the end of that string to obtain a string of length $\ell_2(n)$. Call the resulting string w. Then $v \in \mathcal{N}_\rho^{0,w}(y) \subseteq \mathcal{N}_\rho(x)$, a contradiction.

We proved that $A_{h,\rho}(x) = h(x) = 1$ and $A_{h,\rho}(y) = h(y) = 0$. Therefore, $A_{h,\rho}(v)$ is not monotone, which contradicts the fact that A is a local monotonicity filter. We conclude that H is a 2-TC-spanner of G_n.

We proceed to bound the number of edges E_H in H. For each $\rho \in S$, $x \in V_n$, $b \in \{0,1\}$, and $w \in \{0,1\}^{\ell_2(n)}$, the number of vertices in $N_{b,w}^\rho(x)$ comparable to x is at most $\ell_1(n)$. Therefore,

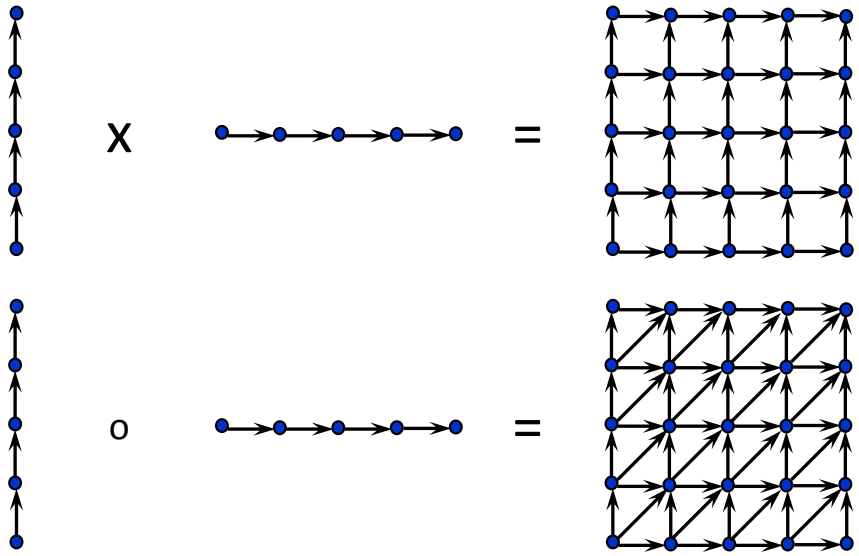

FIG. 4.1. *Graph products: Cartesian (top) and strong (bottom).*

$$|E_H| \leq n \cdot \ell_1(n) \cdot 2 \cdot 2^{\ell_2(n)} \cdot |S| \leq O\left(n \cdot \ell_1(n) \cdot 2^{\ell_2(n)} \left\lceil \frac{\log n}{\log(1/\delta)} \right\rceil\right). \qquad \square$$

4. 2-TC-spanners of low-dimensional hypergrids. In this section, we describe the proof of Theorem 1.8 which gives explicit bounds on the size of the sparsest 2-TC-spanner of $\mathcal{H}_m^d$. The upper bound in Theorem 1.8 is proved in Section 4.1 and the lower bound, in Section 4.2.

4.1. An upper bound for low-dimensional hypergrids. The upper bound in Theorem 1.8 is a straightforward consequence of a more general statement about TC-spanners of product graphs presented in Section 4.1.1. The upper bound in Theorem 1.8 is derived in Section 4.1.2.

4.1.1. A k-TC-spanner construction for product graphs. This section explains how to construct a TC-spanner of the Cartesian product of graphs G_1 and G_2 from TC-spanners of G_1 and G_2. Since the directed hypergrid is the Cartesian product of directed lines, and an optimal TC-spanner construction is known for the directed line, our construction yields a sparse TC-spanner of the grid (Corollary 4.3).

We start by defining two graph products: Cartesian and strong. See Figure 4.1.

DEFINITION 4.1 (Graph products). *Given graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, a product of G_1 and G_2 is a new graph G with the vertex set $V_1 \times V_2$. For the* Cartesian graph product*, denoted by $G_1 \times G_2$, graph G contains an edge from (u_1, u_2) to (v_1, v_2) if and only if $u_1 = v_1$ and $(u_2, v_2) \in E_2$, or $(u_1, v_1) \in E_1$ and $u_2 = v_2$. For the* strong graph product*, denoted by $G_1 \circ G_2$, graph G contains an edge from (u_1, u_2) to (v_1, v_2) if and only if the rules for the Cartesian product are satisfied, or $(u_1, v_1) \in E_1$ and $(u_2, v_2) \in E_2$.*

For example, $\mathcal{H}_m^2 = \mathcal{H}_m^1 \times \mathcal{H}_m^1$ and $\text{TC}(\mathcal{H}_m^2) = \text{TC}(\mathcal{H}_m^1) \circ TC(\mathcal{H}_m^1)$, where $\text{TC}(G)$ denotes the transitive closure of G. Now, observe the following:

LEMMA 4.2. *Let G_1 and G_2 be directed graphs with k-TC-spanners H_1 and H_2, respectively. Then $H_1 \circ H_2$ is a k-TC-spanner of $G = G_1 \times G_2$.*

Proof. Suppose (u, v) and (u', v') are comparable vertices in $G_1 \times G_2$. Then by the definition of the Cartesian product, $u \preceq u'$ in G_1 and $v \preceq v'$ in G_2. Let $(u_1, u_2, \ldots, u_\ell)$ be the shortest path in H_1 from $u = u_1$ to $u' = u_\ell$, and $(v_1, v_2, \ldots, v_t)$ be the shortest path in H_2 from $v = v_1$ to $v' = v_t$. Then $\ell \leq k$ and $t \leq k$, by the definition of a k-TC-spanner. Assume

without loss of generality that $\ell \leq t$. Then $((u_1, v_1), (u_2, v_2), \ldots, (u_\ell, v_\ell) \ldots, (u_\ell, v_t))$ is a path in $H_1 \circ H_2$ of length at most k from (u, v) to (u', v'). Therefore, $H_1 \circ H_2$ is a k-TC-spanner of $G = G_1 \times G_2$. □

4.1.2. A k-TC-spanner construction for the directed hypergrids. Lemma 4.2 together with the previous results on the size of k-TC-spanners of the line $\mathcal{H}^1_m$, summarized in Lemma 1.10, implies an upper bound on the size of a k-TC-spanner of the directed hypergrid $\mathcal{H}^d_m$.

COROLLARY 4.3. *Let $S_k(\mathcal{H}^d_m)$ denote the number of edges in the sparsest k-TC-spanner of the directed d-dimensional hypergrid $\mathcal{H}^d_m$.*

(i) *Then $S_k(\mathcal{H}^d_m) = O(m^d \lambda_k(m)^d c^d)$ for an appropriate constant c.*

(ii) *More precisely, $S_2(\mathcal{H}^d_m) \leq m^d \log^d m$ for $m \geq 3$.*

Proof. Let $H = ([m], E)$ be a k-TC-spanner of the line $\mathcal{H}^1_m$ with $O(m\lambda_k(m))$ edges, given by Lemma 1.10. By Lemma 4.2, the graph $H \circ \cdots \circ H$, where the strong graph product is applied d times, is a k-TC-spanner of the directed hypergrid $\mathcal{H}^d_m$. By the definition of the strong graph product, the number of edges in this k-TC-spanner is $(|E| + m)^d - m^d = O(m^d \lambda_k(m)^d c^d)$ for an appropriate constant c, as claimed in part (1) of the corollary.

The more precise statement for $k = 2$ in part (ii) follows from Claim 4.1 below which gives a more careful analysis of the size of the sparsest 2-TC-spanner of the line. Specifically, it shows that $S_2(\mathcal{H}^1_m) \leq m \log m - m$ for $m \geq 3$. □

CLAIM 4.1. *For all $m \geq 3$, the directed line $\mathcal{H}^1_m$ has a 2-TC-spanner with at most $m \log m - m$ edges.*

Proof. Construct a graph H on the vertex set $[m]$ recursively. First, define the middle node $v_{mid} = \lceil \frac{m}{2} \rceil$. Add edges (v, v_{mid}) for all nodes $v < v_{mid}$ and edges (v_{mid}, v) for all nodes $v > v_{mid}$. Then recurse on the two line segments resulting from removing v_{mid} from the current line. Proceed until each line segment contains exactly one node.

H is a 2-TC-spanner of the line $\mathcal{H}^1_m$, since every pair of nodes $u, v \in [m]$ is connected by a path of length at most 2 via a middle node. This happens in the stage of the recursion where u and v are separated into different line segments, or one of these two nodes is removed.

There are $t = \lfloor \log m \rfloor$ stages of the recursion, and in each stage $i \in [t]$ each node that is not removed by the end of the this stage connects to the middle node in its current line segment. Since 2^{i-1} nodes are removed in the ith stage, exactly $m - (2^i - 1)$ edges are added in that stage. Thus, the total number of edges in H is $m \cdot t - (2^{t+1} - t - 2) \leq m \log m - m$. The last inequality holds for $m \geq 3$. □

4.2. A lower bound for low-dimensional hypergrids. In this section, we show the lower bound on $S_2(\mathcal{H}^d_m)$ stated in Theorem 1.8. We first treat the special case of this lower bound for $d = 2$, since it captures many ideas needed to prove the general bound and is significantly easier to understand. The extension to arbitrary dimension is presented in the subsequent section.

4.2.1. A lower bound for $d = 2$. In this section, we prove a lower bound on the size of a 2-TC-spanner of the 2-dimensional directed grid, stated in Theorem 4.4. This is a special case of the lower bound in Theorem 1.8.

THEOREM 4.4. *The number of edges in a 2-TC-spanner of the 2-dimensional grid $\mathcal{H}^2_m$ is*

$$\Omega\left(\frac{m^2 \log^2 m}{\log \log m}\right).$$

One way to prove the $\Omega(m \log m)$ lower bound on the size of a 2-TC-spanner of the directed line $\mathcal{H}^1_m$, stated in Lemma 1.10, is to observe that at least $\lfloor \frac{m}{2} \rfloor$ edges are cut when the line is halved: namely, at least one per vertex pair $(v, m - v + 1)$ for all $v \in \left[\lfloor \frac{m}{2} \rfloor\right]$. This is depicted in Figure 4.2. Continuing to halve the line recursively, we obtain the desired bound.

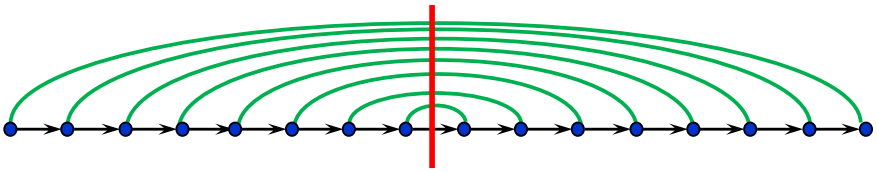

FIG. 4.2. *An illustration to the proof of the lower bound on the size of a 2-TC-spanner for the line.*

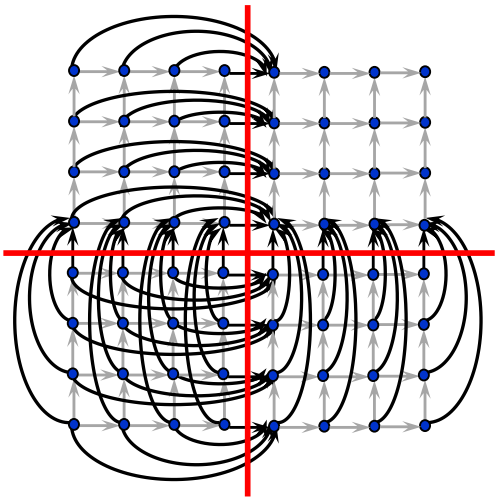

FIG. 4.3. *A 2-TC-spanner of the grid with $O(m^2)$ edges connecting vertices in different quarters. In addition to the depicted edges, it contains a transitive closure of each quarter and an edge from each vertex in the lower left quarter to the smallest vertex of the upper right quarter.*

A natural extension of this approach to proving a lower bound for the grid is to recursively halve the grid along both dimensions, hoping that each such operation on an $m \times m$ grid cuts $\Omega(m^2 \log m)$ edges. This would imply that the size $S(m)$ of a 2-TC-spanner of the $m \times m$ grid satisfies the recurrence $S(m) = 4S(m/2) + \Omega(m^2 \log m)$; that is, $S(m) = \Omega(m^2 \log^2 m)$, matching the upper bound in Theorem 1.8.

An immediate problem with this approach, as shown in Figure 4.3, is that in some 2-TC-spanners of the grid, only $O(m^2)$ edges connect vertices in different quarters. One example of such a 2-TC-spanner is the graph containing the transitive closure of each quarter and only at most $3m^2$ edges crossing from one quarter to another: namely, for each node u and each quarter q with vertices above u, this graph contains an edge (u, v_q), where v_q is the smallest node above u in q.

The TC-spanner in the example above is not optimal because it has too many edges inside the quarters. The first step in our proof of Theorem 4.4 is understanding the tradeoff between the number of edges *crossing* the cut and the number of edges *internal* to the subgrids, resulting from halving the grid along some dimension. The simplest manifestation of this tradeoff occurs when a $2 \times m$ grid is halved into two lines. (In the case of one line, there is no trade off: the $\Omega(m)$ bound on the number of crossing edges holds even if each half-line contains all edges of its transitive closure.) Lemma 4.5 formulates the tradeoff for the two-line case, while taking into account only edges needed to connect comparable vertices on different lines by paths of length at most 2.

LEMMA 4.5 (Two-Lines Lemma). *Let U be a graph with the vertex set $[2] \times [m]$ that contains a path of length at most 2 from $u = (u_1, u_2)$ to $v = (v_1, v_2)$ for every $u \in \{1\} \times [m]$ and $v \in \{2\} \times [m]$, where $u \preceq v$. An edge (u, v) in U is called* internal *if $u_1 = v_1$, and* crossing *otherwise. If U contains at most $\frac{m \log^2 m}{32}$ internal edges, it must contain at least $\frac{m \log m}{16 \log \log m}$ crossing edges.* Note that if the number of internal edges is unrestricted, a 2-TC-

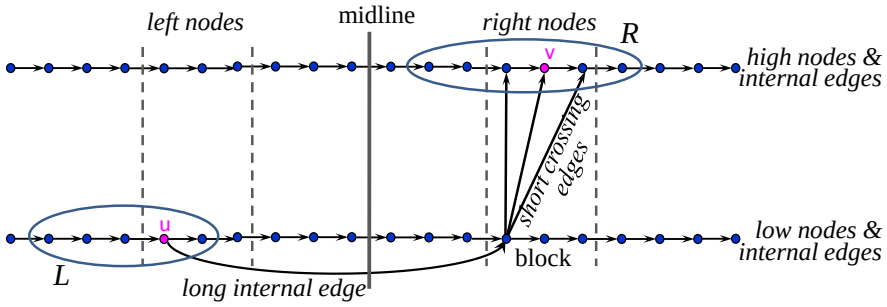

FIG. 4.4. *An illustration of the first stage in the proof of Lemma 4.5.*

spanner of $\mathcal{H}^2_m$ may have only m crossing edges.

Proof. The proof proceeds in $\frac{\log m}{2 \log \log m}$ stages dealing with pairwise disjoint sets of crossing edges. In each stage, we show that U contains at least $\frac{m}{8}$ crossing edges in the prescribed set.

In the first stage, divide U into $\log^2 m$ blocks, each of length $\frac{m}{\log^2 m}$: namely, a node (v_1, v_2) is in block i if

$$v_2 \in \left[\frac{(i-1) \cdot m}{\log^2 m} + 1, \frac{i \cdot m}{\log^2 m} \right].$$

Call an edge *long* if it starts and ends in different blocks, and *short* otherwise. Assume, for contradiction, that U contains fewer than $\frac{m}{8}$ long crossing edges.

Call a node (v_1, v_2) *low* if $v_1 = 1$ (*high* if $v_1 = 2$), and *left* if $v_2 \in \left[\frac{m}{2}\right]$ (*right* otherwise). Also, call an edge (u, v) *low-internal* if $u_1 = v_1 = 1$ and *high-internal* if $u_1 = v_1 = 2$. Let L be the set of low left nodes that are not incident to long crossing edges. Similarly, let R be the set of high right nodes that are not incident to long crossing edges. Since there are fewer than $\frac{m}{8}$ long crossing edges, $|L| > \frac{m}{4}$ and $|R| > \frac{m}{4}$.

A node $u \in L$ can connect to a node $v \in R$ via a path of length at most 2 only by using a long internal edge. Observe that each long low-internal edge can be used by at most $\frac{m}{\log^2 m}$ such pairs (u, v): one low node u and high nodes v from one block. This is illustrated in Figure 4.4. Analogously, every long high-internal edge can be used by at most $\frac{m}{\log^2 m}$ such pairs. Since $|L| \cdot |R| > \frac{m^2}{16}$ pairs in $L \times R$ connect via paths of length at most 2, the graph U contains more than $\frac{m^2}{16} / \frac{m}{\log^2 m} = \frac{m \log^2 m}{16}$ long internal edges, which is a contradiction.

In each subsequent stage, call blocks used in the previous stage *megablocks*, and denote their length by B. Subdivide each megablock into $\log^2 m$ blocks of equal size. Call an edge *long* if it starts and ends in different blocks, but stays within one megablock. Assume, for contradiction, that U contains fewer than $\frac{m}{8}$ long crossing edges.

Call a node (v_1, v_2) *left* if it is in the left half of its megablock, that is, if $v_2 \leq \frac{\ell + r}{2}$ whenever (v_1, v_2) is in a megablock $[2] \times \{\ell, \ldots, r\}$. (Call it *right* otherwise). Consider megablocks containing fewer than $\frac{B}{4}$ long crossing edges each. By an averaging argument, at least $\frac{m}{2B}$ megablocks are of this type. (Recall that there are $\frac{m}{B}$ megablocks overall). Within each such megablock more than $\frac{B}{4}$ low left nodes and more than $\frac{B}{4}$ high right nodes have no incident long crossing edges. By the argument from the first stage, each such megablock contributes more than $\frac{B^2}{16b}$ long internal edges, where $b = \frac{B}{\log^2 m}$ is the size of the blocks. Hence, there must be more than $\frac{B^2}{16b} \cdot \frac{m}{2B} = \frac{m \log^2 m}{32}$ long internal edges, which is a contradiction to the fact that U contains at most $\frac{m \log^2 m}{32}$ internal edges.

We proceed to the next stage until each block is of length 1. Therefore, the number of stages, t, satisfies $\frac{m}{\log^{2t} m} = 1$. That is, $t = \frac{\log m}{2 \log \log m}$, and each stage contributes $\frac{m}{8}$ new crossing edges, as desired. □

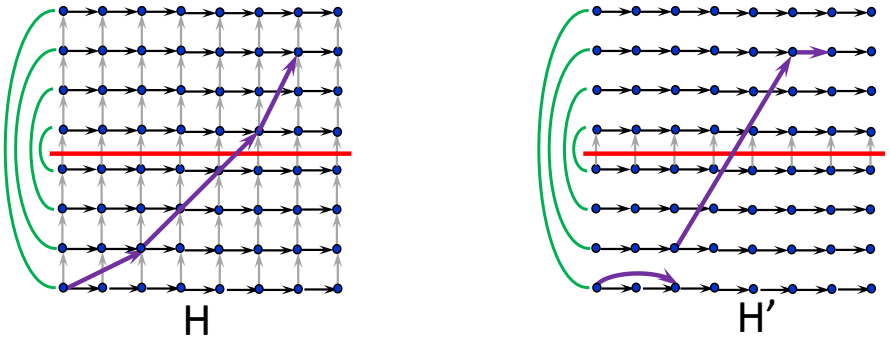

FIG. 4.5. *An illustration to the proof of Lemma 4.6: an example transformation from H to H'. Green arches connect matched lines.*

Next we generalize Lemma 4.5 to understand the tradeoff between the number of internal edges and crossing edges resulting from halving a 2-TC-spanner of a $2\ell \times m$ grid with the usual partial order.

LEMMA 4.6. *Let H be a 2-TC-spanner of the directed $[2\ell] \times [m]$ grid. An edge (u, v) in H is called* internal *if $u_1, v_1 \in [\ell]$ or $u_1, v_1 \in \{\ell + 1, \dots, 2\ell\}$, and* crossing *otherwise. If H contains at most $\frac{\ell m \log^2 m}{64}$ internal edges, it must contain at least $\frac{\ell m \log m}{32 \log \log m}$ crossing edges.*

Proof. For each $i \in [\ell]$, we match the lines $\{i\} \times [m]$ and $\{2\ell - i + 1\} \times [m]$. Observe that a path of length at most 2 between the matched lines cannot use any edges with both endpoints in $\{i + 1, \dots, 2\ell - i\} \times [m]$. We modify H to ensure that there are no edges with only one endpoint in $\{i + 1, \dots, 2\ell - i\} \times [m]$ for all $i \in [\ell]$, and then apply Lemma 4.5 to the matched pairs of lines.

Call the $[\ell] \times [m]$ subgrid and all vertices and edges it contains *low*, and the remaining $\{\ell + 1, \dots, 2\ell\} \times [m]$ subgrid and its vertices and edges *high*. Transform H into H' as follows: change each low internal edge (u, v) to $(u, (u_1, v_2))$, change each high internal edge (u, v) to $((v_1, u_2), v)$, and finally change each crossing edge $((i_1, j_1), (2\ell - i_2 + 1, j_2))$ to $((i, j_1), (2\ell - i + 1, j_2))$, where $i = \min(i_1, i_2)$. Intuitively, we are projecting the edges in H to be fully contained in one of the matched pairs of lines, while preserving whether the edge is internal or crossing. Crossing edges are projected onto the outer matched pair of lines chosen from the two pairs that contain the endpoints of a given edge. See Figure 4.5 for an illustration of the transformation.

The new graph H' has several important properties:

- H' contains at most as many internal (respectively, crossing) edges as H.
- H' contains a path of length at most 2 from u to v for every comparable pair (u, v) where u is low, v is high, and u and v belong to the same pair of matched lines. Indeed, since H is a 2-TC-spanner, it contains either the edge (u, v) or a path (u, w, v). In the first case, H' also contains (u, v). In the second case, if (u, w) is a crossing edge then H' contains the path $(u, (v_1, w_2), v)$, and if (u, w) is an internal edge then H' contains the path $(u, (u_1, w_2), v)$.
- For each edge in H', both endpoints belong either to the same line or to two matched lines. This implies that a path between two vertices that belong to the same pair of matched lines can use only vertices from these two lines as intermediate points, enabling us to apply the Two-Lines Lemma (Lemma 4.5) to each matched pair of lines independently.

Finally, we apply Lemma 4.5. If H contains at most $\frac{\ell m \log^2 m}{64}$ internal edges, then so does H', and so at least half $\left(\text{i.e., } \frac{\ell}{2}\right)$ of the matched line pairs each contain at most $\frac{m \log^2 m}{32}$ internal edges. By Lemma 4.5, each of these pairs contributes at least $\frac{m \log m}{16 \log \log m}$ crossing edges. Thus, H' must contain at least $\frac{\ell m \log m}{32 \log \log m}$ crossing edges. Since H contains at least as many crossing edges as H', the lemma follows. □

Now we prove Theorem 4.4 by recursively halving $\mathcal{H}^2_m$ along the horizontal dimension. We show that either at one of the recursive steps at least half of the resulting subgrids have many internal edges or at each recursive step at least half the subgrids have few internal edges. In the second case, we apply Lemma 4.6 to such subgrids, concluding that they contribute large, pairwise disjoint sets of crossing edges.

Proof of Theorem 4.4 Assume m is a power of 2 for simplicity.

For each step $i \in [\frac{1}{2}\log m]$, partition $\mathcal{H}^2_m$ into the following 2^{i-1} equal-sized subgrids: $\{1,\ldots,l_i\}\times[m], \{l_i+1,\ldots,2l_i\}\times[m],\ldots,\{m-l_i+1,\ldots,m\}\times[m]$ where $l_i = \frac{m}{2^{i-1}}$. For each of these subgrids, define internal and crossing edges as in Lemma 4.6.

First, consider the case when for some step i, at least half of the 2^{i-1} subgrids have more than $\frac{l_i m \log^2 m}{64}$ internal edges. Since at a fixed step i, the subgrids are pairwise disjoint, there are $2^{i-1}\cdot\Omega(l_i m\log^2 m) = \Omega(m^2\log^2 m)$ edges in H, proving the theorem.

If the case above does not hold then for every $i \in [\frac{1}{2}\log m]$, at least half of the 2^{i-1} subgrids have at most $\frac{l_i m\log^2 m}{64}$ internal edges. Then by Lemma 4.6, the number of crossing edges in each such subgrid is at least $\frac{l_i m\log m}{32\log\log m}$. Observe that the sets of crossing edges in different steps are pairwise disjoint. Counting over all steps i and for all appropriate subgrids from those steps, the number of edges in H is bounded by

$$\Omega\left(m^2\log m\frac{\log m}{\log\log m}\right) = \Omega\left(m^2\frac{\log^2 m}{\log\log m}\right). \qquad \square$$

4.2.2. A lower bound for general d. In this section, we extend the above proof to establish lower bounds on $S_2(\mathcal{H}^d_m)$ for arbitrary $d \geq 2$. The following theorem implies the lower bound expression in Theorem 1.8.

THEOREM 4.7. *The number of edges in a 2-TC-spanner of the d-dimensional hypergrid $\mathcal{H}^d_m$ (where $d < \frac{2\log m}{\log\log m}$) is at least*

$$\frac{m^d}{64}\frac{\log^d m}{(2d\log\log m)^{d-1}}.$$

The main ingredient in the proof is the Two-Hyperplanes Lemma, an analogue of the Two-Lines Lemma (Lemma 4.5) for d dimensions. The difficulty in extending the proof of the Two-Lines Lemma to work for two hyperplanes is in generalizing the definitions of blocks and megablocks, so that, on one hand, each stage in the proof contributes a substantial number of crossing edges and, on the other hand, the crossing edges contributed in separate stages are pairwise disjoint.

LEMMA 4.8 (Two-Hyperplanes Lemma). *Let U be a graph with the vertex set $[2]\times[m]^{d-1}$ that contains a path of length at most 2 from u to v for every $u \in \{1\}\times[m]^{d-1}$ and $v \in \{2\}\times[m]^{d-1}$, where $u \preceq v$. As in Lemma 4.5, an edge (u,v) in U is called* internal *if $u_1 = v_1$, and* crossing *otherwise. Then if U contains less than $\frac{m^{d-1}\log^d m}{(d-1)2^{2d+3}}$ internal edges, it must contain at least*

$$\frac{m^{d-1}}{8}\left(\frac{\log m}{2d\log\log m}\right)^{d-1}$$

crossing edges.

Proof. As for Lemma 4.5, the proof proceeds in several stages. The stages are indexed by $(d-1)$-tuples $\boldsymbol{i}$ in $\{0,1,\ldots,\frac{\log m}{d\log\log m}-1\}^{d-1}$. Consequently, the number of stages is

$$\left(\frac{\log m}{d\log\log m}\right)^{d-1}.$$

We will show that each stage contributes at least $\frac{m^{d-1}}{2^{d+2}}$ separate edges to the set of crossing edges, thus proving our lemma.

As in the proof of Lemma 4.5, at each stage vertices are partitioned into megablocks and blocks. In stage $\boldsymbol{i} = (i_1, \ldots, i_{d-1})$, we partition U into $(\log m)^{d(i_1+\cdots+i_{d-1})}$ equal-sized *megablocks* indexed by $\boldsymbol{b} = (b_1, \ldots, b_{d-1})$, where $b_j \in [\log^{d \cdot i_j} m]$ for all $j \in [d-1]$.

A vertex v is in a megablock $\boldsymbol{b}$ if

$$v_{j+1} \in \left[(b_j - 1)\frac{m}{\log^{di_j} m} + 1, b_j \frac{m}{\log^{di_j} m} \right]$$

for each $j \in [d-1]$. So, initially when $\boldsymbol{i} = \vec{0}$, there is only one megablock, and each time $\boldsymbol{i}$ increases by 1 in one coordinate, the volume of the megablocks shrinks by a factor of $\log^d m$. (The volume of a megablock is equal to the number of vertices it contains.)

Each megablock $\boldsymbol{b}$ is further partitioned into $(\log m)^{d(d-1)}$ equal-sized *blocks* indexed by $\boldsymbol{c} \in [\log^d m]^{d-1}$.

A vertex v in a megablock $\boldsymbol{b}$ lies in block $\boldsymbol{c}$ if for each $j \in [d-1]$,

$$(v - \boldsymbol{b}_{min})_{j+1} \in \left[(c_j - 1)\frac{\ell_j}{\log^d m} + 1, c_j \frac{\ell_j}{\log^d m} \right],$$

where $\boldsymbol{b}_{min}$ denotes the smallest vertex in megablock $\boldsymbol{b}$ and ℓ_j denotes the length of $\boldsymbol{b}$ in the the j'th dimension. Note that vertices $(1, v_2, \ldots, v_d)$ and $(2, v_2, \ldots, v_d)$ belong to the same (mega)block. At the last stage, each block contains only two vertices (differing by the first coordinate).

Next, we specify the set of crossing edges contributed at each stage. A crossing edge (u, v) in U is said to be *long* in stage $\boldsymbol{i}$ if:

(i) u and v lie in the same megablock, and

(ii) If u lies in block $(c_1, \ldots, c_{d-1})$ and v lies in block $(c'_1, \ldots, c'_{d-1})$, then $c_j < c'_j$ for all $j \in [d-1]$.

We claim that if $\boldsymbol{i} \neq \boldsymbol{i'}$, the sets of long crossing edges in stages $\boldsymbol{i}$ and $\boldsymbol{i'}$ are disjoint. To see this, let j be an index such that $i_j \neq i'_j$; suppose without loss of generality that $i_j < i'_j$. Then the length of the megablocks in the j'th dimension for stage $\boldsymbol{i'}$ is at most the length of the blocks in the j'th dimension for stage $\boldsymbol{i}$. Hence, condition (ii) above implies that long crossing edges in stage $\boldsymbol{i}$ must have endpoints in different megablocks of stage $\boldsymbol{i'}$, and so violate condition (i) for being a long crossing edge in stage $\boldsymbol{i'}$.

It remains to show that every stage contributes at least $\frac{m^{d-1}}{2^{d+2}}$ long crossing edges. For the sake of contradiction, suppose that the number of long crossing edges at some stage $\boldsymbol{i}$ is less than $\frac{m^{d-1}}{2^{d+2}}$. Let

$$B = \frac{m^{d-1}}{(\log m)^{d(i_1+\cdots+i_{d-1})}}$$

be the volume of the megablocks restricted to one of the two hyperplanes. By an averaging argument, at least $\frac{m^{d-1}}{2B}$ megablocks contain less than $\frac{B}{2^{d+1}}$ long crossing edges (otherwise, there would be at least $\frac{m^{d-1}}{2^{d+2}}$ long crossing edges). But we show next that if a megablock contains less than $\frac{B}{2^{d+1}}$ long crossing edges, then there are at least $\frac{B \log^d m}{(d-1)2^{2d+2}}$ internal edges with both endpoints inside the megablock. This would imply that the total number of internal edges is at least

$$\frac{m^{d-1}}{2B} \cdot \frac{B \log^d m}{(d-1)2^{2d+2}} = \frac{m^{d-1} \log^d m}{(d-1)2^{2d+3}},$$

a contradiction.

Suppose then that a megablock contains less than $\frac{B}{2^{d+1}}$ long crossing edges. Let Low be the set of vertices in the megablock with each coordinate at most the average value of that

coordinate in the megablock, and $High$ the set of vertices with each coordinate greater than the average value of that coordinate. Then

$$|Low| \geq \frac{B}{2^d} \text{ and } |High| \geq \frac{B}{2^d},$$

and each vertex in Low is comparable to each vertex in $High$. By the bound on the number of long crossing edges, there must exist a set L of at least $\frac{B}{2^{d+1}}$ vertices in Low not incident to any long crossing edge, and a set R of at least $\frac{B}{2^{d+1}}$ vertices in $High$ not incident to any long crossing edges. L lies in the lower hyperplane, R in the upper hyperplane, and each vertex in L is comparable to each vertex in R. Call a crossing edge *short* if it satisfies condition (i), but violates condition (ii) above. A path in U of length at most 2 from a vertex in L to a vertex in R must consist of one internal edge and one short crossing edge. The number of short crossing edges incident to a given vertex v is at most $(d-1)\frac{B}{\log^d m}$, by counting, for each of the $d-1$ block indices, the number of vertices in the megablock that share the value of that block index with v. So, each internal edge helps connect at most $(d-1)\frac{B}{\log^d m}$ pairs of vertices. Since $\frac{B^2}{2^{2d+2}}$ pairs of vertices need to be connected by a path, there must exist at least

$$\frac{B^2}{2^{2d+2}} \cdot \frac{\log^d m}{(d-1)B} = \frac{B \log^d m}{(d-1)2^{2d+2}}$$

internal edges. □

The analogue of Lemma 4.6 in d dimensions (Lemma 4.9) and the rest of the proof of Theorem 4.7 are straightforward generalizations of the 2-dimensional case.

LEMMA 4.9. *Let H be a 2-TC-spanner of the directed $[2\ell] \times [m]^{d-1}$ grid. An edge (u, v) in H is called* internal *if $u_1, v_1 \in [\ell]$ or $u_1, v_1 \in \{\ell+1, \ldots, 2\ell\}$, and* crossing *otherwise. If H contains less than $\frac{\ell m^{d-1} \log^d m}{(d-1)2^{2d+4}}$ internal edges, it must contain at least*

$$\frac{\ell}{16}\left(\frac{m \log m}{2d \log\log m}\right)^{d-1}$$

crossing edges.

Proof. We can generalize the proof of Lemma 4.6 in a straightforward way. For each $i \in [\ell]$, instead of matching the lines, we match the hyperplanes $\{i\} \times [m]^{d-1}$ and $\{2\ell - i + 1\} \times [m]^{d-1}$. The rest of the proof follows an identical argument with the one in the proof of Lemma 4.6. Still, we repeat the argument here for the sake of completion. For this proof, we define some additional notation. Given vertex $v = (v_1, \ldots, v_d) \in [m]^d$, let $\mathbf{v_2}$ denote $(v_2, v_3, \ldots, v_d)$. Further, for $\mathbf{v_2} = (v_2, v_3, \ldots, v_d)$ let $(v_1, \mathbf{v_2})$ denote the vertex $v = (v_1, \ldots, v_d)$.

Now, observe that a path of length at most 2 between the matched hyperplanes cannot use any edges with both endpoints in $\{i+1, \ldots, 2\ell - i\} \times [m]^{d-1}$. We modify H to ensure that there are no edges with only one endpoint in $\{i+1, \ldots, 2\ell - i\} \times [m]^{d-1}$ for all $i \in [\ell]$, and then apply Lemma 4.8 to the matched pairs of hyperplanes.

Call the $[\ell] \times [m]^{d-1}$ subgrid and all vertices and edges it contains *low*, and the remaining $\{\ell+1, \ldots, 2\ell\} \times [m]^{d-1}$ subgrid and its vertices and edges *high*. Transform H into H' as follows: change each low internal edge (u, v) to $(u, (u_1, \mathbf{v_2}))$, change each high internal edge (u, v) to $((v_1, \mathbf{u_2}), v)$, and finally change each crossing edge $((u_1, \mathbf{u_2}), (2\ell - v_1 + 1, \mathbf{v_2}))$ to $((t_1, \mathbf{u_2}), (2\ell - t_1 + 1, \mathbf{v_2}))$, where $t_1 = \min(u_1, v_1)$. Intuitively, we are projecting the edges in H to be fully contained in one of the matched pairs of hyperplanes, while preserving whether the edge is internal or crossing. Crossing edges are projected onto the outer matched pair of hyperplanes chosen from the two pairs that contain the endpoints of a given edge. See Figure 4.5 for an illustration of the transformation for the case when $d = 2$.

The new graph H' has several important properties:

- H' contains at most as many internal (respectively, crossing) edges as H.

- H' contains a path of length at most 2 from u to v for every comparable pair (u, v) where u is low, v is high, and u and v belong to the same pair of matched hyperplanes. Indeed, since H is a 2-TC-spanner, it contains either the edge (u, v) or a path (u, w, v). In the first case, H' also contains (u, v). In the second case, if (u, w) is a crossing edge then H' contains the path $(u, (v_1, \mathbf{w_2}), v)$, and if (u, w) is an internal edge then H' contains the path $(u, (u_1, \mathbf{w_2}), v)$.
- For each edge in H', both endpoints belong either to the same hyperplane or to two matched hyperplanes. This implies that a path between two vertices that belong to the same pair of matched hyperplanes can use only vertices from these two hyperplanes as intermediate points, enabling us to apply the Two-Hyperplanes Lemma (Lemma 4.8) to each matched pair of hyperplanes independently.

Finally, we apply Lemma 4.8. If H contains at most $\frac{m^{d-1}\log^d m}{(d-1)2^{2d+4}}$ internal edges, then so does H', and so at least half $\left(\text{i.e., } \frac{\ell}{2}\right)$ of the matched hyperplane pairs each contain at most $\frac{m^{d-1}\log^d m}{(d-1)2^{2d+3}}$ internal edges. By Lemma 4.8, each of these pairs contributes at least

$$\frac{m^{d-1}}{8}\left(\frac{\log m}{2d\log\log m}\right)^{d-1}$$

crossing edges. Thus, H' must contain at least

$$\frac{\ell}{16}\left(\frac{m\log m}{2d\log\log m}\right)^{d-1}$$

crossing edges. Since H contains at least as many crossing edges as H', the lemma follows. □

Proof of Theorem 4.7. For simplicity, assume m is a power of 2.

For each step $i \in [\frac{1}{2}\log m]$, partition $\mathcal{H}_m^d$ into the following 2^{i-1} equal-sized subgrids: $\{1, \ldots, l_i\} \times [m]^{d-1}$, $\{l_i + 1, \ldots, 2l_i\} \times [m]^{d-1}$, $\ldots$, $\{m - l_i + 1, \ldots, m\} \times [m]^d$ where $l_i = \frac{m}{2^{i-1}}$. For each of these subgrids, define internal and crossing edges as in Lemma 4.9. Now, suppose that there exists a step i such that at least half of the 2^{i-1} subgrids have at least $\frac{l_i m^{d-1}\log^d m}{(d-1)2^{2d+3}}$ internal edges. Since at a fixed step i, the subgrids are pairwise disjoint, there are at least $2^{i-2}\frac{l_i m^{d-1}\log^d m}{(d-1)2^{2d+3}} = \frac{m^d \log^d m}{(d-1)2^{2d+4}}$ edges in H, which is enough to prove the theorem. On the other hand, suppose that for every $i \in [\frac{1}{2}\log m]$, at least half of the 2^{i-1} subgrids have less than $\frac{l_i m^{d-1}\log^d m}{(d-1)2^{2d+3}}$ internal edges. Then by Lemma 4.9, the number of crossing edges in each such subgrid is at least

$$\frac{l_i m^{d-1}}{16}\left(\frac{\log m}{2d\log\log m}\right)^{d-1}.$$

Counting over all steps i and for all appropriate subgrids from those steps, the number of edges in H is at least

$$\frac{\log m}{2}\cdot 2^{i-2}\cdot\frac{l_i m^{d-1}}{16}\left(\frac{\log m}{2d\log\log m}\right)^{d-1} = \frac{m^d}{64}\frac{\log^d m}{(2d\log\log m)^{d-1}}.$$ □

5. 2-TC-spanners of the hypercube. In this section, we prove Theorem 1.9, that is, we analyze the size of the sparsest 2-TC-spanner of the d-dimensional hypercube $\mathcal{H}^d$. Lemma 5.1 presents the upper bound on $S_2(\mathcal{H}^d)$. Lemma 5.2 presents the lower bound. The upper and lower bounds differ only by a factor of $O(d^3)$, and are dominated by the same combinatorial expression. A numerical approximation to this expression is given in Lemma 5.3. Remark 5.1 at the end of the section explains why our randomized construction in Lemma 5.1 yields a 2-TC-spanner of $\mathcal{H}^d$ of size within $O(d^2)$ of the optimal.

5.1. Upper bound.

LEMMA 5.1. *There is a 2-TC-spanner of $\mathcal{H}^d$ with*

$$O\left(d^3 \max_{i,j:i<j} \min_{k:i\le k\le j} \frac{\binom{d}{k}}{\binom{j-i}{k-i}} \max\left\{\binom{k}{i}, \binom{d-k}{d-j}\right\}\right) \text{ edges.}$$

Proof. Consider the following probabilistic construction that connects all comparable vertices in levels i and j of $\mathcal{H}^d$ by paths of length at most 2.

Given levels $i, j \in \{0, 1, ..., d\}$, $i < j$,

1. Initialize the set $E_{i,j}$ to $\emptyset$.
2. Let $k_{i,j} = \operatorname{argmin}_{k:i\le k\le j} \left(\frac{\binom{d}{k}}{\binom{j-i}{k-i}} \max\left\{\binom{k}{i}, \binom{d-k}{d-j}\right\}\right)$.
3. Let $S_{i,j}$ be a set of $3d\frac{\binom{d}{k}}{\binom{j-i}{k-i}}$ vertices chosen uniformly at random from the $\binom{d}{k}$ vertices in level $k = k_{i,j}$.
4. For each vertex $v \in S_{i,j}$, set $E_{i,j}$ to $E_{i,j} \cup \{(x, v) : |x| = i \wedge x \prec v\} \cup \{(v, y) : |y| = j \wedge v \prec y\}$. That is, connect v to all comparable vertices in levels i and j.
5. Output $E_{i,j}$.

CLAIM 5.1. *For all $0 \le i < j \le d$, with probability at least $\frac{1}{2}$, the edge set $E_{i,j}$ contains a path of length at most 2 between every pair of vertices (x, y), such that $x \prec y$, $|x| = i$, and $|y| = j$.*

Proof. Consider a pair of vertices (x, y) with $x \prec y$, such that $|x| = i$ and $|y| = j$. The number of vertices in level k that are above x and below y is exactly $\binom{j-i}{k-i}$. Therefore, the probability that $S_{i,j}$ does not contain such a vertex is

$$\left(1 - \binom{j-i}{k-i} / \binom{d}{k}\right)^{3d\frac{\binom{d}{k}}{\binom{j-i}{k-i}}} \le e^{-3d}.$$

The number of comparable pairs (x, y) is $\binom{d}{i}\binom{d-i}{d-j}$. By a union bound, the probability that there exists an (x, y), such that no vertex $v \in S_{i,j}$ satisfies $x \prec v \prec y$ is at most $\binom{d}{i}\binom{d-i}{d-j}e^{-3d} \le 2^{2d}e^{-3d} < \frac{1}{2}$. □

Claim 5.1 implies that for every i and j, there exists a set $S_{i,j}$, such that all comparable pairs from the levels i and j are connected by a path of length at most 2 via a vertex in $S_{i,j}$. Let $E^*_{i,j}$ be the set of edges returned by the algorithm when this $S_{i,j}$ is chosen. We set $E = \bigcup_{0\le i<j\le d} E^*_{i,j}$. Then the graph $(\{0, 1\}^d, E)$ is a 2-TC-spanner of $\mathcal{H}^d$.

Now, we show that the size of E is as claimed in the lemma statement. The main observation is that in Step 4, for every $v \in S_{i,j}$, the set

$$\{(x, v) : |x| = i \wedge x \prec v\} \cup \{(v, y) : |y| = j \wedge v \prec y\}$$

has size $\binom{k_{i,j}}{i} + \binom{d-k_{i,j}}{d-j} \le 2\max\left\{\binom{k_{i,j}}{i}, \binom{d-k_{i,j}}{d-j}\right\}$. Therefore, for all $0 \le i < j \le d$,

$$\begin{aligned}|E^*_{i,j}| &\le 3d \cdot 2\frac{\binom{d}{k_{i,j}}}{\binom{j-i}{k_{i,j}-i}} \max\left\{\binom{k_{i,j}}{i}, \binom{d-k_{i,j}}{d-j}\right\} \\ &= 6d \min_{k:i\le k\le j} \frac{\binom{d}{k}}{\binom{j-i}{k-i}} \max\left\{\binom{k}{i}, \binom{d-k}{d-j}\right\}.\end{aligned}$$

Since $|E| = \sum_{0\le i<j\le d} |E^*_{i,j}|$, where the sum has $O(d^2)$ terms, the claimed bound follows. □

5.2. Lower bound.

LEMMA 5.2. *Any 2-TC-spanner of the directed hypercube $\mathcal{H}^d$ has*

$$\Omega\left(\max_{i,j:i<j}\min_{k:i\le k\le j}\frac{\binom{d}{k}}{\binom{j-i}{k-i}}\max\left\{\binom{k}{i},\binom{d-k}{d-j}\right\}\right)$$

edges.

Proof. Let H be a 2-TC-spanner of $\mathcal{H}^d$. We will count the edges in H that occur on paths connecting two particular levels of $\mathcal{H}^d$. Let $P_{i,j}$ be the pairs $\{(v_1,v_2):|v_1|=i,|v_2|=j,v_1\prec v_2\}$. We will lower bound $e^*_{i,j}$, the number of edges in the paths of length at most 2 in H that connect the pairs $P_{i,j}$. Let $e_{k,\ell}$ denote the number of edges in H that connect vertices in level k to vertices in level ℓ. Then $e^*_{i,j}=e_{i,j}+\sum_{k=i+1}^{j-1}(e_{i,k}+e_{k,j})$.

We say that a vertex v *covers* a pair of vertices (v_1,v_2) if H contains the edges (v_1,v) and (v,v_2) or, for the special case $v=v_1$, if H contains (v_1,v_2). Let $V^{(k)}_{i,j}$ be the set of vertices of weight k that cover pairs in $P_{i,j}$. Let α_k be the fraction of pairs in $P_{i,j}$ that are covered by a vertex in $V^{(k)}_{i,j}$. Since each pair in $P_{i,j}$ must be covered by a vertex in levels i to $j-1$,

$$\sum_{k=i}^{j-1}\alpha_k\ge 1.$$

For every vertex $v\in V^{(k)}_{i,j}$, let in_v be the number of incoming edges from vertices of weight i incident to v and let out_v be the number of outgoing edges to vertices of weight j incident to v. For each $k\in\{i+1,...,j-1\}$, since each vertex $v\in V^{(k)}_{i,j}$ covers $in_v\cdot out_v$ pairs,

$$\sum_{v\in V^{(k)}_{i,j}} in_v\cdot out_v\ge\alpha_k|P_{i,j}|=\alpha_k\binom{d}{i}\binom{d-i}{d-j}. \tag{5.1}$$

We upper bound $\sum_{v\in V^{(k)}_{i,j}} in_v\cdot out_v$ as a function of $e_{i,k}+e_{k,j}$, and then use (5.1) to lower bound $e_{i,k}+e_{k,j}$.

For all $k\in\{i+1,...,j-1\}$, variables in_v and out_v satisfy the following constraints:

$$\sum_{v\in V^{(k)}_{i,j}} in_v\le e_{i,k}+e_{k,j},\quad \sum_{v\in V^{(k)}_{i,j}} out_v\le e_{i,k}+e_{k,j}\ .$$

$$in_v\le\binom{k}{i}\forall v\in V^{(k)}_{i,j},\quad out_v\le\binom{d-k}{d-j}\forall v\in V^{(k)}_{i,j}\ .$$

The last two constraints hold because in_v and out_v count the number of edges to a vertex of weight k from vertices of weight i and from a vertex of weight k to vertices of weight j, respectively. Using these bounds we obtain

$$\sum_{v\in V^{(k)}_{i,j}} in_v\cdot out_v\le\sum_{v\in V^{(k)}_{i,j}}\binom{k}{i}\cdot out_v=\binom{k}{i}\cdot\sum_{v\in V^{(k)}_{i,j}} out_v\le\binom{k}{i}\cdot(e_{i,k}+e_{k,j}).$$

Similarly, $\sum_{v\in V^{(k)}_{i,j}} in_v\cdot out_v\le\binom{d-k}{d-j}\cdot(e_{i,k}+e_{k,j})$. Therefore, for all $k\in\{i+1,...,j-1\}$:

$$\sum_{v\in V^{(k)}_{i,j}} in_v\cdot out_v\le(e_{i,k}+e_{k,j})\min\left\{\binom{k}{i},\binom{d-k}{d-j}\right\}.$$

Let $s_{i,k,j} = \frac{\binom{d}{i}\binom{d-i}{d-j}}{\min\left\{\binom{k}{i},\binom{d-k}{d-j}\right\}}$. Then (5.1) implies that $e_{i,k} + e_{k,j} \geq \alpha_k s_{i,k,j}$ for all $k \in \{i+1, ..., j-1\}$. Therefore,

$$e^*_{i,j} = e_{i,j} + \sum_{k=i+1}^{j-1} (e_{i,k} + e_{k,j}) \geq \alpha_i \binom{d}{i}\binom{d-i}{d-j} + \sum_{k=i+1}^{j-1} \alpha_k s_{i,k,j} \geq \sum_{k=i}^{j-1} \alpha_k s_{i,k,j} \geq \min_{k:i\leq k\leq j} s_{i,k,j}$$

Since this holds for arbitrary i and j, the number of edges in the 2-TC-spanner H is at least

$$\max_{i,j:i<j} \min_{k:i\leq k\leq j} s_{i,k,j}.$$

Finally, a simple algebraic manipulation finishes the proof.

CLAIM 5.2. $s_{i,k,j} = \frac{\binom{d}{k}}{\binom{j-i}{k-i}} \max\left\{\binom{k}{i}, \binom{d-k}{d-j}\right\}$.

Proof. Take the ratio of the two sides:

$$\frac{s_{i,k,j}}{\frac{\binom{d}{k}}{\binom{j-i}{k-i}} \max\left\{\binom{k}{i}, \binom{d-k}{d-j}\right\}} = \frac{\binom{d}{i}\binom{d-i}{d-j}\binom{j-i}{k-i}}{\binom{d}{k}\binom{k}{i}\binom{d-k}{d-j}} = \frac{\binom{d}{i}\binom{d-i}{j-i}\binom{j-i}{k-i}}{\binom{d}{k}\binom{k}{i}\binom{d-k}{j-k}} = 1.$$

The first equality follows from the fact that $\max(x, y) \cdot \min(x, y) = x \cdot y$. The last equality can be proved either by expanding the binomial coefficients into factorials, or by realizing that both $\binom{d}{i}\binom{d-i}{j-i}\binom{j-i}{k-i}$ and $\binom{d}{k}\binom{k}{i}\binom{d-k}{j-k}$ count the number of ways i red balls, $j-k$ blue balls, and $k-i$ green balls can be placed into d slots, each of which can hold at most one ball. This completes the proof of the claim. □

This completes the proof of the lemma. □

The following lemma gives a handle on the expression capturing the size of a 2-TC-spanner.

LEMMA 5.3. *Let* $s = \max_{i,j:i<j} \min_{k:i\leq k\leq j} \frac{\binom{d}{k}}{\binom{j-i}{k-i}} \max\left\{\binom{k}{i}, \binom{d-k}{d-j}\right\}$. *Then* $s = 2^{cd}$, *where* $c \approx 1.1620$.

Proof. We use the fact that $\binom{n}{cn} = 2^{(H_b(c)-o_n(1))n}$, where "$o_n(1)$" is a function of n that tends to zero as n tends to infinity, and $H_b(p) = -p\log p - (1-p)\log(1-p)$ is the binary entropy function. Substituting $i = \alpha d, j = \beta d$ and $k = \gamma d$ in the resulting expression for s, and taking the logarithm of both sides, we get

$$\log_2 s = \max_{0\leq\alpha<\beta\leq 1} \min_{\alpha\leq\gamma\leq\beta} \Big[H_b(\gamma) - H_b\Big(\frac{\gamma-\alpha}{\beta-\alpha}\Big)(\beta-\alpha) + \max\Big(H_b\Big(\frac{\alpha}{\gamma}\Big)\gamma, H_b\Big(\frac{1-\beta}{1-\gamma}\Big)(1-\gamma)\Big)\Big]d.$$

In other words, $\log_2 s = cd$ where c is a constant. We can check numerically that $c \approx 1.1620$. □

REMARK 5.1. *We note that if the first maximum in the expression for* s *in Lemma 5.3 is replaced with the sum then Lemma 5.1 holds for* $O(d \cdot s)$ *instead of* $O(d^3 \cdot s)$ *while Lemma 5.2 holds for* $\Omega(\frac{s}{d})$ *instead of* $\Omega(s)$. *The proofs of these modified statements are similar. (We do not have an analogue of Lemma 5.3 for the modified expression for* s.*) Observe that the modified bounds differ by a factor of* $O(d^2)$ *instead of* $O(d^3)$. *This demonstrates that our randomized construction yields a 2-TC-spanner of* $\mathcal{H}^d$ *of size within* $O(d^2)$ *of the optimal.* ♢

6. 2-TC-spanners of high-dimensional hypergrids. In this section, we generalize the arguments for the hypercube from Section 5 to the directed hypergrid, $\mathcal{H}^d_m$, to find the size of the sparsest 2-TC-spanner of $\mathcal{H}^d_m$ to within $\mathrm{poly}(d)$ factors. This result supersedes the results of Section 4 when, for instance, m is constant and d is growing. The expression we

obtain can be evaluated numerically for small m using standard approximations of binomial coefficients. For example, this was done in Lemma 5.3 for the case $m = 2$.

Before stating Theorem 6.2, we introduce some notation.

DEFINITION 6.1. *For the hypergrid $\mathcal{H}_m^d$, define a* level *to be a set of vertices, indexed by vector $\mathbf{i} \in [d]^m$ with $i_1 + \cdots + i_m = d$, that consists of vertices $x = (x_1, \ldots, x_d) \in [m]^d$ containing i_1 positions of value 1, i_2 positions of value 2, ..., and i_m positions of value m.*

Notice that the number of vertices in level $\mathbf{i} = (i_1, i_2, \ldots, i_m)$ is the multinomial coefficient

$$\binom{d}{\mathbf{i}} = \binom{d}{i_1, ..., i_d} = \binom{d}{i_1}\binom{d-i_1}{i_2}\binom{d-i_1-i_2}{i_3}\cdots\binom{d-\sum_{l=1}^{m-1} i_l}{i_m}.$$

Indeed, there are $\binom{d}{i_1}$ choices for the coordinates of value 1. For each such choice there are $\binom{d-i_1}{i_2}$ choices for the coordinates of value 2, and repeating this argument one obtains the above expression.

For levels $\mathbf{i}, \mathbf{j} \in [d]^m$, say $\mathbf{j}$ *majorizes* $\mathbf{i}$, denoted $\mathbf{j} \succ \mathbf{i}$, if level $\mathbf{j}$ contains a vertex which is above some vertex in level $\mathbf{i}$, *i.e.,* , if

$$\sum_{\ell=t}^{m} j_\ell \geq \sum_{\ell=t}^{m} i_\ell \text{ for all } t \in \{m, m-1, ..., 1\}.$$

For $\mathbf{j} \succ \mathbf{i}$, the number of vertices y in level $\mathbf{i}$ comparable to a fixed vertex x in level $\mathbf{j}$ is $\mathcal{M}(\mathbf{i}, \mathbf{j})$:

$$\binom{j_m}{i_m}\binom{j_m + j_{m-1} - i_m}{i_{m-1}}\binom{j_m + j_{m-1} + j_{m-2} - i_m - i_{m-1}}{i_{m-2}}\cdots\binom{\sum_{l=1}^{m} j_l - \sum_{l=2}^{m} i_l}{i_1}.$$

Indeed, there are $\binom{j_m}{i_m}$ choices for the coordinates of value m in y. For each such choice, there are $\binom{j_m + j_{m-1} - i_m}{i_{m-1}}$ choices for the coordinates of value $m-1$ in y, and one can repeat this argument to obtain the claimed expression.

For $\mathbf{j} \succ \mathbf{i}$, the number of vertices y in level $\mathbf{j}$ comparable to a fixed vertex x in level $\mathbf{i}$ is

$$\mathcal{N}(\mathbf{i}, \mathbf{j}) = \frac{\mathcal{M}(\mathbf{i}, \mathbf{j})\binom{d}{\mathbf{j}}}{\binom{d}{\mathbf{i}}}.$$

Indeed, there are $\mathcal{M}(\mathbf{i}, \mathbf{j})\binom{d}{\mathbf{j}}$ comparable pairs of vertices in levels $\mathbf{i}$ and $\mathbf{j}$, and level $\mathbf{i}$ contains $\binom{d}{\mathbf{i}}$ vertices. Since, by symmetry, each vertex in level $\mathbf{i}$ is comparable to the same number of vertices in level $\mathbf{j}$, we get the desired expression.

THEOREM 6.2. *Let*

$$\mathcal{B}(m, d) = \max_{\mathbf{i},\mathbf{j}:\mathbf{j}\succ\mathbf{i}} \min_{\mathbf{k}:\mathbf{i}\prec\mathbf{k}\prec\mathbf{j}} \frac{\mathcal{M}(\mathbf{i}, \mathbf{j})\binom{d}{\mathbf{j}}}{\mathcal{M}(\mathbf{i}, \mathbf{k})\mathcal{N}(\mathbf{k}, \mathbf{j})} \max\left\{\mathcal{M}(\mathbf{i}, \mathbf{k}), \mathcal{N}(\mathbf{k}, \mathbf{j})\right\}.$$

Then the number of edges in the sparsest 2-TC-spanner of the directed hypergrid $\mathcal{H}_m^d$ is $O\left(d^{2m}\mathcal{B}(m, d)\right)$ and $\Omega\left(\mathcal{B}(m, d)\right)$.

The bounds stated in Theorem 6.2 are presented separately as Lemma 6.3 (the upper bound) and Lemma 6.4 (the lower bound) below.

6.1. Upper bound.

LEMMA 6.3. *There is a 2-TC-spanner of $\mathcal{H}_m^d$ with $O\left(d^{2m}\mathcal{B}(m, d)\right)$ edges, where $\mathcal{B}(m, d)$ is defined as in Theorem 6.2.*

Proof. Let $v \in \mathbf{i}$ denote that vertex v belongs to level $\mathbf{i}$. Consider the following probabilistic construction that connects comparable vertices in levels $\mathbf{i}$ and $\mathbf{j}$ of $\mathcal{H}_m^d$ by paths of length at most 2.

Given levels $\mathbf{i}, \mathbf{j} \in [d]^m$, $\mathbf{j} \succ \mathbf{i}$,

1. Initialize the set $E_{\mathbf{i},\mathbf{j}}$ to $\emptyset$.
2. Let $\mathbf{k}_{\mathbf{i},\mathbf{j}} = \underset{\mathbf{k}:\mathbf{i}\prec\mathbf{k}\prec\mathbf{j}}{\operatorname{argmin}} \left(\frac{\mathcal{M}(\mathbf{i},\mathbf{j})\binom{d}{\mathbf{j}}}{\mathcal{M}(\mathbf{i},\mathbf{k})\mathcal{N}(\mathbf{k},\mathbf{j})} \max\{\mathcal{M}(\mathbf{i},\mathbf{k}), \mathcal{N}(\mathbf{k},\mathbf{j})\} \right)$.
3. Let $S_{\mathbf{i},\mathbf{j}}$ be a set of $d^m \frac{\mathcal{M}(\mathbf{i},\mathbf{j})\binom{d}{\mathbf{j}}}{\mathcal{M}(\mathbf{i},\mathbf{k})\mathcal{N}(\mathbf{k},\mathbf{j})}$ vertices chosen uniformly at random from the $\binom{d}{\mathbf{k}}$ vertices in level $\mathbf{k} = \mathbf{k}_{\mathbf{i},\mathbf{j}}$.
4. For each vertex $v \in S_{\mathbf{i},\mathbf{j}}$, set $E_{\mathbf{i},\mathbf{j}}$ to $E_{\mathbf{i},\mathbf{j}} \cup \{(x,v) : x \in \mathbf{i} \wedge x \prec v\} \cup \{(v,y) : y \in \mathbf{j} \wedge v \prec y\}$. That is, connect v to all comparable vertices in levels $\mathbf{i}$ and $\mathbf{j}$.
5. Output $E_{\mathbf{i},\mathbf{j}}$.

CLAIM 6.1. *For all $\mathbf{i} \prec \mathbf{j}$, with probability at least $\frac{1}{2}$, the edge set $E_{\mathbf{i},\mathbf{j}}$ contains a path of length at most 2 between every pair of vertices (x,y), such that $x \prec y$, $x \in \mathbf{i}$, and $y \in \mathbf{j}$.*

Proof. Fix a pair of vertices (x,y) with $x \prec y$, such that $x \in \mathbf{i}$ and $y \in \mathbf{j}$. We will first show that $Pr_{v\in\mathbf{k}}[x \prec v \prec y] \geq p$, where $p = \frac{\mathcal{M}(\mathbf{i},\mathbf{k})\mathcal{N}(\mathbf{k},\mathbf{j})}{\mathcal{M}(\mathbf{i},\mathbf{j})\binom{d}{\mathbf{j}}}$.

Toward that end, notice that there are $\mathcal{M}(\mathbf{i},\mathbf{j})\binom{d}{\mathbf{j}}$ pairs of comparable vertices (u,w) with $u \in \mathbf{i}$ and $w \in \mathbf{j}$. Each vertex in $S_{\mathbf{i},\mathbf{j}}$ connects exactly $\mathcal{M}(\mathbf{i},\mathbf{k})\mathcal{N}(\mathbf{k},\mathbf{j})$ pairs of nodes from levels $\mathbf{i}$ and $\mathbf{j}$. It is enough to show that for any such pair (u,w), the number of vertices in level $\mathbf{k}$ that are comparable to both u and v is independent of u, w, i.e., that number only depends on the levels $\mathbf{i}, \mathbf{k}, \mathbf{j}$, and thus is the same for all such pairs. To see that, for a vertex $u \in \mathbf{z}$, denote by $T_l(u)$ the set of positions of value l in u. Notice that $|T_l(u)| = z_l$. For $x \prec v \prec y$, it holds that $T_m(x) \subseteq T_m(v) \subseteq T_m(y)$. Hence, there are $\binom{j_m - i_m}{k_m - i_m}$ choices for the m-values in the vector v. Similarly, we must have $T_{m-1}(x) \subseteq T_{m-1}(v) \subseteq T_m(y) \cup T_{m-1}(y)$. Hence, there are $\binom{j_m + j_{m-1} - k_m - i_{m-1}}{k_{m-1} - i_{m-1}}$ choices for the values $m-1$ in v. Repeating this process, we obtain that the number of possible v's does not depend on the particular choice of x and y.

Thus, the probability that $S_{\mathbf{i},\mathbf{j}}$ does not contain such a vertex v with $x \prec v \prec y$ is $(1-p)^{d^m/p} \leq e^{-d^m}$.

The number of comparable pairs (x,y) is at most m^{2d}, and by a union bound, the probability that there exists (x,y), such that there is no $v \in S_{\mathbf{i},\mathbf{j}}$ with $x \prec v \prec y$, is at most $m^{2d}e^{-d^m} < 1/2$. □

By Claim 6.1, for every $\mathbf{i}$ and $\mathbf{j}$, there exists a set $S_{\mathbf{i},\mathbf{j}}$, such that comparable pairs from the levels $\mathbf{i}$ and $\mathbf{j}$ are connected by a path of length at most 2 via a vertex in $S_{\mathbf{i},\mathbf{j}}$. Let $E^*_{\mathbf{i},\mathbf{j}}$ be the set of edges returned by the algorithm when this $S_{\mathbf{i},\mathbf{j}}$ is chosen. We set $E = \bigcup_{\mathbf{j}\succ\mathbf{i}} E^*_{\mathbf{i},\mathbf{j}}$. Then the graph $([m]^d, E)$ is a 2-TC-spanner of $\mathcal{H}^d_m$.

Now, we show that the size of E is as claimed in the lemma statement. The main observation is that in Step 4, for every $v \in S_{\mathbf{i},\mathbf{j}}$, the set

$$\{(x,v) : x \in \mathbf{i} \wedge x \prec v\} \cup \{(v,y) : y \in \mathbf{j} \wedge v \prec y\}$$

has size $\mathcal{M}(\mathbf{i},\mathbf{k}) + \mathcal{N}(\mathbf{k},\mathbf{j})$.

The claimed bound follows since $|E| = \sum_{\mathbf{j}\succ\mathbf{i}} |E^*_{\mathbf{i},\mathbf{j}}|$, where the sum has d^m terms. □

6.2. Lower bound.

LEMMA 6.4. *Every 2-TC-spanner of $\mathcal{H}^d_m$ has $\Omega(\mathcal{B}(m,d))$ edges, where $\mathcal{B}(m,d)$ is defined as in Theorem 6.2.*

Proof. Let H be a 2-TC-spanner of $\mathcal{H}^d_m$. We count the edges in H that occur on paths connecting two particular levels of $\mathcal{H}^d_m$. Let $P_{\mathbf{i},\mathbf{j}} = \{(v_1,v_2) : v_1 \in \mathbf{i}, v_2 \in \mathbf{j}, v_1 \prec v_2\}$. We will lower bound $e^*_{\mathbf{i},\mathbf{j}}$, the number of edges in the paths of length at most 2 in H that connect the pairs $P_{\mathbf{i},\mathbf{j}}$. Notice that $|P_{\mathbf{i},\mathbf{j}}| = \binom{d}{\mathbf{j}}\mathcal{M}(\mathbf{i},\mathbf{j})$.

Let $e_{\mathbf{k},\boldsymbol{\ell}}$ denote the number of edges in H that connect vertices in level $\mathbf{k}$ to vertices in level $\boldsymbol{\ell}$. Then

$$e^*_{\mathbf{i},\mathbf{j}} = e_{\mathbf{i},\mathbf{j}} + \sum_{\mathbf{i}\prec\mathbf{k}\prec\mathbf{j}} (e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}}). \tag{6.1}$$

We say that a vertex v *covers* a pair of vertices (v_1, v_2) if H contains the edges (v_1, v) and (v, v_2) or, for the special case $v = v_1$, if H contains (v_1, v_2). Let $V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}$ be the set of vertices in level $\mathbf{k}$ that cover pairs in $P_{\mathbf{i},\mathbf{j}}$. Let $\alpha_{\mathbf{k}}$ be the fraction of pairs in $P_{\mathbf{i},\mathbf{j}}$ that are covered by the vertices in $V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}$. Since each pair in $P_{\mathbf{i},\mathbf{j}}$ must be covered by a vertex in levels $\mathbf{k}$ with $\mathbf{i} \prec \mathbf{k} \prec \mathbf{j}$, we have

$$\sum_{\mathbf{i} \prec \mathbf{k} \prec \mathbf{j}} \alpha_{\mathbf{k}} \geq 1.$$

For any vertex $v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}$, let in_v be the number of incoming edges from vertices of level $\mathbf{i}$ incident to v and let out_v be the number of outgoing edges to vertices of level $\mathbf{j}$ incident to v. For each level $\mathbf{k}$ with $\mathbf{i} \prec \mathbf{k} \prec \mathbf{j}$, since each vertex $v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}$ covers $in_v \cdot out_v$ pairs,

$$\sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} in_v \cdot out_v \geq \alpha_{\mathbf{k}} |P_{\mathbf{i},\mathbf{j}}| \geq \alpha_{\mathbf{k}} \mathcal{M}(\mathbf{i}, \mathbf{j}) \binom{d}{\mathbf{j}}. \tag{6.2}$$

We upper bound $\sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} in_v \cdot out_v$ as a function of $e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}}$, and then use (6.2) to lower bound $e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}}$. For all $\mathbf{k}$ with $\mathbf{i} \prec \mathbf{k} \prec \mathbf{j}$, variables in_v and out_v satisfy the following constraints:

$$\sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} in_v \leq e_{\mathbf{i},\mathbf{k}} \leq e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}}, \quad \sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} out_v \leq e_{\mathbf{k},\mathbf{j}} \leq e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}},$$

$$in_v \leq \mathcal{M}(\mathbf{i}, \mathbf{k}) \; \forall v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}, \quad out_v \leq \mathcal{N}(\mathbf{k}, \mathbf{j}) \; \forall v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}.$$

The last two constraints hold because in_v and out_v count the number of edges to a vertex of level $\mathbf{k}$ from vertices of level $\mathbf{i}$, and from a vertex of level $\mathbf{k}$ to vertices of level $\mathbf{j}$, respectively. These bounds imply that

$$\sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} in_v \cdot out_v \leq \sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} \mathcal{M}(\mathbf{i}, \mathbf{k}) \cdot out_v = \mathcal{M}(\mathbf{i}, \mathbf{k}) \cdot \sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} out_v \leq \mathcal{M}(\mathbf{i}, \mathbf{k}) \cdot (e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}}).$$

Similarly, $\sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} in_v \cdot out_v \leq \mathcal{N}(\mathbf{k}, \mathbf{j}) \cdot (e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}})$. Therefore,

$$\sum_{v \in V^{(\mathbf{k})}_{\mathbf{i},\mathbf{j}}} in_v \cdot out_v \leq (e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}}) \min \left\{ \mathcal{M}(\mathbf{i}, \mathbf{k}), \mathcal{N}(\mathbf{k}, \mathbf{j}) \right\}.$$

Then (6.2) implies that

$$e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}} \geq \alpha_{\mathbf{k}} \mathcal{M}(\mathbf{i}, \mathbf{j}) \binom{d}{\mathbf{j}} \frac{1}{\min \left\{ \mathcal{M}(\mathbf{i}, \mathbf{k}), \mathcal{N}(\mathbf{k}, \mathbf{j}) \right\}} \text{ for all } \mathbf{i} \prec \mathbf{k} \prec \mathbf{j}.$$

Applying (6.1) and the fact that $\sum_{\mathbf{i} \prec \mathbf{k} \prec \mathbf{j}} \alpha_{\mathbf{k}} \geq 1$, we get

$$\begin{aligned} e^*_{\mathbf{i},\mathbf{j}} = e_{\mathbf{i},\mathbf{j}} + \sum_{\mathbf{i} \prec \mathbf{k} \prec \mathbf{j}} (e_{\mathbf{i},\mathbf{k}} + e_{\mathbf{k},\mathbf{j}}) &\geq \sum_{\mathbf{k}} \alpha_{\mathbf{k}} \frac{1}{\min \left\{ \mathcal{M}(\mathbf{i}, \mathbf{k}), \mathcal{N}(\mathbf{k}, \mathbf{j}) \right\}} \mathcal{M}(\mathbf{i}, \mathbf{j}) \binom{d}{\mathbf{j}} \\ &\geq \min_{\mathbf{k}} \frac{1}{\min \left\{ \mathcal{M}(\mathbf{i}, \mathbf{k}), \mathcal{N}(\mathbf{k}, \mathbf{j}) \right\}} \mathcal{M}(\mathbf{i}, \mathbf{j}) \binom{d}{\mathbf{j}} \\ &= \min_{\mathbf{k}} \frac{1}{\mathcal{M}(\mathbf{i}, \mathbf{k}) \mathcal{N}(\mathbf{k}, \mathbf{j})} \mathcal{M}(\mathbf{i}, \mathbf{j}) \binom{d}{\mathbf{j}} \max \left\{ \mathcal{M}(\mathbf{i}, \mathbf{k}), \mathcal{N}(\mathbf{k}, \mathbf{j}) \right\}. \end{aligned}$$

Since this holds for arbitrary $\mathbf{i}$ and $\mathbf{j}$, the size of the 2-TC-spanner H is at least $\mathcal{B}(m, d)$. □

REFERENCES

[1] N. AILON, B. CHAZELLE, S. COMANDUR, AND D. LIU, *Property-preserving data reconstruction*, Algorithmica, 51 (2008), pp. 160–182.

[2] NOGA ALON AND BARUCH SCHIEBER, *Optimal preprocessing for answering on-line product queries*, Tech. Report 71/87, Tel-Aviv University, 1987.

[3] MIKHAIL J. ATALLAH, MARINA BLANTON, NELLY FAZIO, AND KEITH B. FRIKKEN, *Dynamic and efficient key management for access hierarchies*, ACM Trans. Inf. Syst. Secur., 12 (2009).

[4] BARUCH AWERBUCH, *Communication-time trade-offs in network synchronization*, in Proceedings of the fourth annual ACM symposium on Principles of distributed computing, Michael Malcolm and Ray Strong, eds., PODC '85, New York, NY, USA, 1985, ACM, pp. 272–276.

[5] SURENDER BASWANA AND SANDEEP SEN, *Approximate distance oracles for unweighted graphs in expected $\tilde{O}(n^2)$ time*, ACM Trans. Algorithms, 2 (2006), pp. 557–577.

[6] ARNAB BHATTACHARYYA, ELENA GRIGORESCU, MADHAV JHA, KYOMIN JUNG, SOFYA RASKHODNIKOVA, AND DAVID P. WOODRUFF, *Lower bounds for local monotonicity reconstruction from transitive-closure spanners*, in Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010, Maria J. Serna, Ronen Shaltiel, and Klaus Jansen a Jos D. P. Rolim, eds., vol. 6302 of Lecture Notes in Comput. Sci., Springer, 2010, pp. 448–461.

[7] ARNAB BHATTACHARYYA, ELENA GRIGORESCU, KYOMIN JUNG, SOFYA RASKHODNIKOVA, AND DAVID P. WOODRUFF, *Transitive-closure spanners*, in Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, Claire Mathieu, ed., SODA '09, Philadelphia, PA, USA, 2009, SIAM, pp. 932–941.

[8] HANS L. BODLAENDER, GERARD TEL, AND NICOLA SANTORO, *Trade-offs in non-reversing diameter*, Nordic J. Comput., 1 (1994), pp. 111–134.

[9] ASHOK K. CHANDRA, STEVEN FORTUNE, AND RICHARD J. LIPTON, *Lower bounds for constant depth circuits for prefix problems*, in ICALP, 1983, pp. 109–117.

[10] ———, *Unbounded fan-in circuits and associative functions*, J. Comput. Syst. Sci., 30 (1985), pp. 222–234.

[11] BERNARD CHAZELLE, *Computing on a free tree via complexity-preserving mappings*, Algorithmica, 2 (1987), pp. 337–361.

[12] EDITH COHEN, *Fast algorithms for constructing t-spanners and paths with stretch t*, SIAM J. Comput., 28 (1998), pp. 210–236.

[13] ———, *Polylog-time and near-linear work approximation scheme for undirected shortest paths*, J. ACM, 47 (2000), pp. 132–166.

[14] LENORE COWEN, *Compact routing with minimum stretch*, J. Algorithms, 38 (2001), pp. 170–183.

[15] LENORE COWEN AND CHRISTOPHER G. WAGNER, *Compact roundtrip routing in directed networks*, J. Algorithms, 50 (2004), pp. 79–95.

[16] YEVGENIY DODIS, ODED GOLDREICH, ERIC LEHMAN, SOFYA RASKHODNIKOVA, DANA RON, AND ALEX SAMORODNITSKY, *Improved testing algorithms for monotonicity*, in Randomization, Approximation, and Combinatorial Optimization. Algorithms and Techniques, 2nd International Workshop, APPROX 1999, 3rd International Workshop, RANDOM 1999, Dorit S. Hochbaum, Klaus Jansen, Jos D. P. Rolim, and Alistair Sinclair, eds., vol. 1671 of Lecture Notes in Comput. Sci., Springer, 1999, pp. 97–108.

[17] MICHAEL ELKIN, *Computing almost shortest paths*, in Proceedings of the twentieth annual ACM symposium on Principles of distributed computing, Ajay Kshemkalyani and Nir Shavit, eds., PODC '01, New York, NY, USA, 2001, ACM, pp. 53–62.

[18] ODED GOLDREICH, SHAFI GOLDWASSER, ERIC LEHMAN, DANA RON, AND ALEX SAMORODNITSKY, *Testing monotonicity*, Combinatorica, 20 (2000), pp. 301–337.

[19] WILLIAM HESSE, *Directed graphs requiring large numbers of shortcuts*, in Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms, Martin Farach-Colton, ed., SODA '03, Philadelphia, PA, USA, 2003, SIAM, pp. 665–669.

[20] GIRI NARASIMHAN AND MICHIEL H. M. SMID, *Geometric spanner networks*, Cambridge University Press, 2007.

[21] DAVID PELEG AND ALEJANDRO A. SCHÄFFER, *Graph spanners*, J. Graph Theory, 13 (1989), pp. 99–116.

[22] DAVID PELEG AND JEFFREY D. ULLMAN, *An optimal synchronizer for the hypercube*, SIAM J. Comput., 18 (1989), pp. 740–747.

[23] DAVID PELEG AND ELI UPFAL, *A trade-off between space and efficiency for routing tables*, J. ACM, 36 (1989), pp. 510–530.

[24] SOFYA RASKHODNIKOVA, *Transitive-closure spanners: a survey*, in Property Testing, Oded Goldreich, ed., vol. LNCS 6390, Springer, Heidelberg, 2010, pp. 167–196.

[25] LIAM RODITTY, MIKKEL THORUP, AND URI ZWICK, *Roundtrip spanners and roundtrip routing in directed graphs*, in Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms, David Eppstein, ed., SODA '02, Philadelphia, PA, USA, 2002, SIAM, pp. 844–851.

[26] MICHAEL SAKS AND C. SESHADHRI, *Local monotonicity reconstruction*, SIAM J. Comput., 39 (2010), pp. 2897–2926.

[27] RAIMUND SEIDEL, *Understanding the inverse Ackermann function*. Available at `http://cgi.di.uoa.gr/~ewcg06/invited/Seidel.pdf`, 2006.

[28] SHAY SOLOMON, *An optimal-time construction of sparse euclidean spanners with tiny diameter*, in Proceed-

ings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, Dana Randall, ed., SODA '11, Philadelphia, PA, USA, 2011, SIAM, pp. 820–839.

[29] MIKKEL THORUP, *On shortcutting digraphs*, in Graph-Theoretic Concepts in Computer Science, 18th International Workshop, WG, Wiesbaden-Naurod, Germany, 1992, pp. 205–211.

[30] ———, *Shortcutting planar digraphs*, Combin. Probab. Comput., 4 (1995), pp. 287–315.

[31] ———, *Parallel shortcutting of rooted trees*, J. Algorithms, 23 (1997), pp. 139–159.

[32] MIKKEL THORUP AND URI ZWICK, *Compact routing schemes*, in Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures, Arnold Rosenberg, ed., SPAA '01, New York, NY, USA, 2001, ACM, pp. 1–10.

[33] MIKKEL THORUP AND URI ZWICK, *Approximate distance oracles*, J. ACM, 52 (2005), pp. 1–24.

[34] ANDREW C. YAO, *Space-time tradeoff for answering range queries (extended abstract)*, in Proceedings of the fourteenth annual ACM symposium on Theory of computing, Harry R. Lewis, Barbara B. Simons, Walter A. Burkhard, and Lawrence H. Landweber, eds., STOC '82, New York, NY, USA, 1982, ACM, pp. 128–136.