The limited blessing of low dimensionality: when $1 - 1/d$ is the best possible exponent for d-dimensional geometric problems

[Extended Abstract]

Dániel Marx*
Institute of Computer Science and Control
Hungarian Academy of Sciences (MTA SZTAKI)
Budapest, Hungary
dmarx@cs.bme.hu

Anastasios Sidiropoulos
Dept. of Computer Science & Engineering,
and Dept. of Mathematics
The Ohio State University
Columbus, OH, USA
sidiropoulos.1@osu.edu

ABSTRACT

We are studying d-dimensional geometric problems that have algorithms with $1-1/d$ appearing in the exponent of the running time, for example, in the form of $2^{n^{1-1/d}}$ or $n^{k^{1-1/d}}$. This means that these algorithms perform somewhat better in low dimensions, but the running time is almost the same for all large values d of the dimension. Our main result is showing that for some of these problems the dependence on $1 - 1/d$ is best possible under a standard complexity assumption. We show that, assuming the Exponential Time Hypothesis,

- d-dimensional Euclidean TSP on n points cannot be solved in time $2^{O(n^{1-1/d-\epsilon})}$ for any $\epsilon > 0$, and
- the problem of finding a set of k pairwise nonintersecting d-dimensional unit balls/axis parallel unit cubes cannot be solved in time $f(k)n^{o(k^{1-1/d})}$ for any computable function f.

These lower bounds essentially match the known algorithms for these problems. To obtain these results, we first prove lower bounds on the complexity of Constraint Satisfaction Problems (CSPs) whose constraint graphs are d-dimensional grids. We state the complexity results on CSPs in a way to make them convenient starting points for problem-specific reductions to particular d-dimensional geometric problems and to be reusable in the future for further results of similar flavor.

*Research supported by the European Research Council (ERC) grant "PARAMTIGHT: Parameterized complexity and the search for tight complexity results," reference 280152 and OTKA grant NK105645.

SoCG'14, June 8–11, 2014, Kyoto, Japan.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Computations on discrete structures*

General Terms

Theory

1. INTRODUCTION

The curse of dimensionality is a ubiquitous phenomenon observed over and over again for geometric problems: polynomial-time algorithms that work for low dimensions quickly become infeasible in high dimensions, as the running time depends exponentially on the dimension d. Consider, for example, the Euclidean k-center problem, which can be formulated as follows: given a set of P points in d-dimensions, find a set of k unit balls whose union covers P. For $k = 1$, the problem can be solved in linear time [30], but the problem becomes NP-hard even for $k = 2$ [31] and only algorithms with running time of the form $n^{O(d)}$ is known [2]. In recent years, the framework of W[1]-hardness has been used to give evidence that for several problems (including Euclidean 2-center), the exponent of n has to depend on the dimension d; in fact, for many of these problems tight lower bounds have been given that show that no $n^{o(d)}$ algorithm is possible under standard complexity assumptions [13, 14, 7, 23, 22, 12, 6, 8].

For certain other geometric problems, however, the dimension d affects the complexity of the problem in a very different way. Consider, for example, the classical Traveling Salesperson Problem (TSP): given a distance metric on a set of n points, the task is to find a shortest path[1] visiting all n points. This problem can be solved in time $2^n \cdot n^{O(1)}$ using a well-known dynamic programming algorithm of Bellman [5] and of Held and Karp [20] that works for any metric. However, in the special case when the points

[1]This variant of TSP is also known as *path-TSP*. Our lower bound also holds for *tour-TSP*, i.e. the variant where one seeks to find a cycle visiting all vertices. In order to simplify the discussion, we restrict our attention to path-TSP for now; on Section 4 we explain how our proof can be modified to obtain the same lower bound for cycle-TSP.

are in d-dimensional Euclidean space, TSP can be solved in time $2^{d^{O(d)}} \cdot n^{O(dn^{1-1/d})}$ by an algorithm of Smith and Wormald [34], that is, treating d as a fixed constant, the running time is $n^{O(n^{1-1/d})} = 2^{O(n^{1-1/d} \cdot \log n)} = 2^{O(n^{1-1/d+\epsilon})}$ for every $\epsilon > 0$. This means that, as the dimension d grows, the running time quickly converges to the $2^n \cdot n^{O(1)}$ time of the standard dynamic programming algorithm that does not exploit any geometric property of the problem. On the other hand, when the dimension d is small, the algorithm has a moderate gain over dynamic programming: for example, for $d = 2$, we have $2^{O(\sqrt{n}\log n)}$ instead of $2^n \cdot n^{O(1)}$. This behavior is very different compared to the $n^{O(d)}$ running time of algorithms for problems affected by the curse of dimensionality: for those problems, complexity gets constantly worse and worse as d grows, while for TSP the complexity is essentially the same for all large values of d. Therefore, we may call this phenomenon observed for d-dimensional TSP the "limited blessing of low dimensionality": the running time is almost uniformly bad for large values of d, but some amount of improvement can be achieved for low dimensions.

A slightly different example of the same phenomenon appears in the case of packing problems. Consider the following problem: Given a set of n unit balls in d-dimensional space and an integer k, the task is to find k pairwise disjoint balls, or in other words, we have to find an independent set of size k in the intersection graph of the balls. Clearly, this can be done in time $n^{O(k)}$ by brute force for any intersection graph. However, using the geometric nature of the problem, one can reduce the running time to $n^{O(k^{1-1/d})}$ (for $d = 2$, this has been proved by Alber and Fiala [3]; in Appendix 5, we sketch a simple algorithm for any $d \geq 2$ based on a standard sweeping argument and dynamic programming[2]). Again, we are in a similar situation as in the case of d-dimensional TSP: as d grows, the running time quickly converges to the $n^{O(k)}$ running time of brute force, but there is a moderate improvement for low dimensions (for example, $n^{O(\sqrt{k})}$ vs. $n^{O(k)}$ for $d = 2$).

Our results. Can we make the blessing of low dimensionality more pronounced? That is, can we improve $1 - 1/d$ in the exponent of the running time of the algorithms described above to something like $1-1.1/d$ or $1-1/\sqrt{d}$? The main result of the current paper is showing that the exponent $1-1/d$ is best possible for these problems. We prove these results under the complexity assumption called *Exponential Time Hypothesis (ETH)*, introduced by Impagliazzo, Paturi, and Zane [21], stating that n-variable 3SAT cannot be solved in time $2^{o(n)}$. This complexity assumption is the basis of tight lower bounds for many problems, see the survey of Lokshtanov et al. [25].

For d-dimensional TSP, we prove the following result:

THEOREM 1.1. *If for some $d \geq 2$ and $\epsilon > 0$,* TSP *in d-dimensional Euclidean space can be solved in time $2^{O(n^{1-1/d-\epsilon})}$, then ETH fails.*

Note that this lower bound essentially matches the $2^{O(n^{1-1/d+\epsilon})}$ time algorithm of Smith and Wormald [34]. For packing problems, we have the following results:

THEOREM 1.2. *If for some $d \geq 2$ and computable function f, there is a $f(k)n^{o(k^{1-1/d})}$ time algorithm for finding k pairwise nonintersecting d-dimensional balls/axis-parallel cubes, then ETH fails.*

[2]We thank Sariel Har-Peled for suggesting this approach for the algorithm.

That is, the exponent $k^{1-1/d}$ cannot be improved, even if we allow an arbitrary dependence $f(k)$ as a multiplicative constant. That is, in the language of parameterized complexity, we are not only proving that the problem is not fixed-parameter tractable, but we also give a tight lower bound on the dependence of the exponent on the parameter k.

To prove Theorems 1.1 and 1.2, we first develop general tools for approaching d-dimensional geometric problems. We formulate complexity results in the abstract setting of Constraint Satisfaction Problems (CSPs) whose constraint graphs are d-dimensional grids. These results faithfully capture the influence of the number d of dimensions on problem complexity and are stated in a way to facilitate further reductions to d-dimensional geometric problems. Then we can obtain Theorems 1.1 and 1.2 by problem-specific reductions that are mostly local and do not depend very much on the number of dimensions. We believe that our results for d-dimensional CSPs could serve as a useful starting point for proving further results of this flavor for geometric problems. Producing an exhaustive list of such results was not the goal of the current paper; instead, we wanted to demonstrate that $1 - 1/d$ in the exponent can be the best possible dependence on the dimension, build the framework for proving such lower bounds, and provide a sample of results on concrete problems.

Let us remark that the results in Theorems 1.1–1.2 were already known for the special case of $d = 2$. Papadimitriou [32] proved the NP-hardness of Euclidean TSP in $d = 2$ dimensions by a reduction from EXACT-COVER: given an instance of EXACT-COVER with universe size n and m subsets, the reduction creates an equivalent instance of TSP with $O(nm)$ points. It can be shown that an instance of 3-COLORING with n vertices and m edges can be reduced to an instance of EXACT-COVER with universe size $O(n + m)$ and number of sets $O(n + m)$. Therefore, a $2^{o(\sqrt{n})}$ algorithm for TSP in $d = 2$ dimensions would give a $2^{o(n+m)}$ time algorithm for 3-COLORING, contradicting ETH [25].

Marx [26] proved the W[1]-hardness of finding k pairwise nonintersecting unit disks (in $d = 2$ dimensions) by a reduction from k-CLIQUE. The reduction maps an instance of k-CLIQUE to a set of disks where $k' := k^2$ independent disks have to be found. By a result of Chen et al. [10], if k-CLIQUE can be solved in time $f(k)n^{o(k)}$ for some computable function f, then ETH fails. Putting together the result of Chen et al. [10] and the reduction of Marx [26], we get Theorem 1.2 for $d = 2$.

For $d \geq 3$ dimensions, however, the tight lower bounds become much harder to obtain. As we shall see, the hardness proofs rely on constructing embeddings into d-dimensional grids. For $d = 2$, this can be achieved by simple and elementary arguments, but the tight results for $d \geq 3$ require more delicate constructions.

Constraint satisfaction problems. We use the language of CSPs to express the basic lower bounds in a way that is not specific to any geometric problem. A CSP is defined by a set V of variables, a domain D from which the variables can take values, and a set of constraints on the variables. In the current paper, we consider only CSPs where every constraint is binary, that is, involves only two variables and restricts the possible combination of values that can appear on those two variables in a solution (see Section 2 for definitions related to CSPs). It is important to point out that one can consider CSPs where the size of the domain D is a fixed small constant (e.g., 3-Coloring on a graph G can be reduced to a CSP with $|V(G)|$ variables and $|D| = 3$) or CSPs where the domain size is large, much larger than the number of variables (e.g., k-Clique on a graph G can be reduced to a CSP with k variables and $|D| = |V(G)|$). We will need both viewpoints in the current paper.

Intuitively, it is clear how a hardness proof for a d-dimensional geometric problem should proceed. We construct small gadgets able to represent a certain number of states and put copies of these gadgets at certain locations in d-dimensional space. Then each gadget can interact with the at most $2d$ gadgets that are "adjacent" to it in one of the d dimensions. The gadgets should be constructed to ensure that each such interaction enforces some binary constraint on the states of the two gadgets. Therefore, we can effectively express a CSP where the variables are located on the d-dimensional grid and the binary constraints are only on adjacent variables. This means that we need lower bounds on the complexity of such CSPs. In particular, we would like to understand the complexity of CSPs where the graph of constraints is exactly a d-dimensional grid.

There is a large body of literature on how *structural restrictions,* that is, restrictions on the constraint graph influences the complexity of CSP [9, 17, 1, 18, 15, 19, 28, 27, 29, 16]. Specifically, we need a general result of Marx [28] stating that, in a precise technical sense, treewidth of the constraint graph governs the complexity of the problem. Roughly speaking, the result states that, assuming the Exponential Time Hypothesis, there is no $|D|^{o(\mathrm{tw}(G)/\log \mathrm{tw}(G))}$ time algorithm for CSP (where $\mathrm{tw}(G)$ is the treewidth of the constraint graph) and this holds *even if we restrict the constraint graph to any class of graphs.* Therefore, if we restrict CSP to instances where the constraint graph is the d-dimensional grid with $k = m^d$ vertices for some m, then the known fact that such a d-dimensional grid has treewidth $O(m^{d-1}) = O(k^{1-1/d})$ implies that there is no $|D|^{o(k^{1-1/d}/\log k)}$ time algorithm for such CSPs. Therefore, in a sense, the connection to treewidth given by [28] and the treewidth of the d-dimensional grid explain why $1-1/d$ is the right exponent for the d-dimensional geometric problems we are considering.

We still have some work left to prove Theorems 1.1–1.2. First, as a minor issue, we remove the log factor from the lower bound obtained above for CSPs whose constraint graph is a d-dimensional grid and improve it to the tight bound ruling out $|D|^{o(k^{1-1/d})}$ time algorithms. The general result of [28] is a based on constructing certain embeddings exploiting the treewidth of the constraint graph. However, by focusing on a specific class of graphs, we can obtain slightly better embeddings and therefore improve the lower bound. In particular, Alon and Marx [4] gave an embedding of an arbitrary graph into a d-dimensional Hamming graph (generalized hypercube) and it is easy to embed a $(d-1)$-dimensional Hamming graph into a d-dimensional grid. These embeddings together prove the tighter lower bound. Second, we modify the CSPs to make them more suited to reductions to geometric problems. In these CSPs, the domain is $[\delta]^d$ for some integer δ, that is, the solution assigns every variable a d-tuple of integers as a value. Every constraint is of the same form: if variables v_1 and v_2 are adjacent in the i-th dimension (with v_2 being larger by one in the i-th coordinate), then the constraint requires that the i-th component of the value of v_1 is at most the i-th component of the value of v_2. Then problem-specific, but very transparent reductions from these CSPs to packing unit disks or unit cubes prove Theorem 1.2.

To prove Theorem 1.1, we need a slightly different approach. The issue is that the general result of [28] holds only if there is no bound on the domain size. Thus we need a reduction to TSP that works even if the domain size is much larger than the number of variables. However, if we have k variables and domain size $|D|$, then probably the best we can hope for is a reduction to TSP with $n = O(|D|k)$ or so points (and even this is only under the assumption that we can construct gadgets with $O(|D|)$ points to represent each variable and each constraint, which is far from obvious). But then a $2^{O(n^{1-1/d-\epsilon})}$ time algorithm for d-dimensional TSP would give only a $2^{O(|D|k)^{1-1/d-\epsilon}}$ time algorithm for CSP, which would not violate the lower bound ruling out $|D|^{o(k^{1-1/d})}$ time algorithms. Therefore, we prove a variant of the lower bound stating that there is a constant δ such that there is no $2^{O(k^{1-1/d-\epsilon})}$ time algorithm for CSP on d-dimensional grids even under the restriction $|D| \leq \delta$. Again, we prove this lower bound by revisiting the embedding results into d-dimensional grids and Hamming graphs. Then a problem-specific reduction reusing some of the ideas of Papadimitriou [32] for the $d = 2$ case proves Theorem 1.1. Interestingly, our reduction exploits the fact $d \geq 3$: this allows us to express arbitrary binary relations in an easy way without having to worry about crossings.

2. CONSTRAINT SATISFACTION PROBLEMS

Understanding constraint satisfaction problems (CSPs) whose constraint graphs are d-dimensional grids seems to be a very convenient starting point for proving lower bounds on the complexity of d-dimensional geometric problems. In this section, we review the relevant background on CSPs and prove the basic complexity results that will be useful for the lower bounds on specific d-dimensional geometric problems.

Definition 2.1. *An instance of a* constraint satisfaction problem *is a triple (V, D, C), where:*

- *V is a set of variables,*
- *D is a domain of values,*
- *C is a set of constraints, $\{c_1, c_2, \dots, c_q\}$. Each constraint $c_i \in C$ is a pair $\langle s_i, R_i \rangle$, where:*
  - *s_i is a tuple of variables of length m_i, called the* constraint scope, *and*
  - *R_i is an m_i-ary relation over D, called the* constraint relation.

For each constraint $\langle s_i, R_i \rangle$ the tuples of R_i indicate the allowed combinations of simultaneous values for the variables in s_i. The length m_i of the tuple s_i is called the *arity* of the constraint. A *solution* to a constraint satisfaction problem instance is a function f from the set of variables V to the domain of values D such that for each constraint $\langle s_i, R_i \rangle$ with $s_i = (v_{i_1}, v_{i_2}, \dots, v_{i_m})$, the tuple $(f(v_{i_1}), f(v_{i_2}), \dots, f(v_{i_m}))$

is a member of R_i. We say that an instance is *binary* if each constraint relation is binary, i.e., $m_i = 2$ for each constraint (hence the term "binary" refers to the arity of the constraints and *not* to the size of the domain). Note that Definition 2.1 allows for a variable to appear multiple times in the scope of the constraint. Thus a binary instance can contain a constraint of the form $\langle (v, v), R\rangle$, which is essentially a unary constraint. We will deal only with binary CSPs in this paper. We may assume that there is at most one constraint with the same scope, as two constraints $\langle s, R_1\rangle$ and $\langle s, R_2\rangle$ can be merged into a single constraint $\langle s, R_1 \cap R_2\rangle$. Therefore, we may assume that the input size $|I|$ of a binary CSP instance is polynomial in $|V|$ and $|D|$, without going into the details of how the constraints are exactly represented.

The *primal graph* (or *Gaifman graph*) of a CSP instance $I = (V, D, C)$ is a graph with vertex set V such that distinct vertices $u, v \in V$ are adjacent if and only if there is a constraint whose scope contains both u and v. The following classical result shows that treewidth of the primal graph is a relevant parameter to the complexity of CSPs: low treewidth implies efficient algorithms.

THEOREM 2.2 (FREUDER [11]). *Given a binary CSP instance I whose primal graph has treewidth w, a solution can be found in time $|I|^{O(w)}$.*

The *d-dimensional grid* $\mathsf{R}[n, d]$ has vertex set $[n]^d$ and vertices $\mathbf{a} = (a_1, \ldots, a_d)$ and $\mathbf{b} = (b_1, \ldots, b_d)$ are adjacent if and only if $\sum_{i=1}^{d} |a_i - b_i| = 1$, that is, they differ in exactly one coordinate and only by exactly one in that coordinate. In other words, if we denote by $\mathbf{e}_i$ the unit vector whose i-th coordinate is 1 and every other coordinate is 0, then $\mathbf{b}$ is the neighbor of $\mathbf{a}$ only if $\mathbf{b}$ is of the form $\mathbf{a} + \mathbf{e}_i$ or $\mathbf{a} - \mathbf{e}_i$ for some $1 \le i \le d$. Note that the maximum degree of $\mathsf{R}[n, d]$ is $2d$ (for $n \ge 3$). We denote by $\mathcal{R}_d$ the set of graphs $\mathsf{R}[n, d]$ for every $n \ge 1$. For every fixed d, the treewidth of the d-dimensional grid is $\Theta(n^{d-1})$ (this is proved for the related notion of carving width by Kozawa et al. [24], but carving width is known to be between $\mathrm{tw}(G)/3$ and $\Delta\mathrm{tw}(G)$, where Δ is the maximum degree [35]).

PROPOSITION 2.3 (KOZAWA ET AL. [24]). *For any fixed $d \ge 2$, the treewidth of $\mathsf{R}[n, d]$ is $\Theta(n^{d-1}) = \Theta(|V(\mathsf{R}[n, d])|^{1-1/d})$.*

Theorem 2.2 and Proposition 2.3 together imply that, for every fixed d, CSPs restricted to instances where the primal graph is a d-dimensional grid $\mathsf{R}[n, d]$ can be solved in time $|I|^{O(n^{d-1})} = |I|^{O(|V|^{1-1/d})}$.

A result of Marx [28] provides a converse of Theorem 2.2 showing, in a precise technical sense, that it is indeed the treewidth of the primal graph that determines the complexity. This very general result can be used to provide an almost matching lower bound on the complexity of solving CSPs on a d-dimensional grid. For a class $\mathcal{G}$ of graphs, let us denote by $\mathrm{CSP}(\mathcal{G})$ the binary CSP problem restricted to instances whose primal graph is in $\mathcal{G}$.

THEOREM 2.4 (MARX [28]). *If there is a recursively enumerable class $\mathcal{G}$ of graphs with unbounded treewidth, an algorithm $\mathbb{A}$, and a function f such that $\mathbb{A}$ correctly decides every binary $\mathrm{CSP}(\mathcal{G})$ instance in time $f(|V|)|I|^{o(\mathrm{tw}(G)/\log \mathrm{tw}(G))}$, then ETH fails.*

Theorem 2.4 and Proposition 2.3 together imply the following lower bound:

COROLLARY 2.5. *For every fixed $d \ge 2$, there is no $f(|V|)|I|^{o(|V|^{1-1/d}/\log|V|)}$ algorithm for $\mathrm{CSP}(\mathcal{R}_d)$ for any function f, unless ETH fails.*

We extend Corollary 2.5 in two ways. First, by an analysis specific to the class $\mathcal{R}_d$ of graphs at hand and avoiding the general tools used in the proof of Theorem 2.4, we can get rid of the logarithmic factor in the exponent, making the result tight.

THEOREM 2.6. [*] *For every fixed $d \ge 2$, there is no $f(|V|)|I|^{o(|V|^{1-1/d})}$ algorithm for $\mathrm{CSP}(\mathcal{R}_d)$ for any function f, unless ETH fails.*

Second, observe that Theorem 2.4 does not give any lower bound for the restriction of the problem to instances with domain size bounded by a fixed constant δ. In fact, no such strong negative result as Theorem 2.4 can hold for instances with domain size restricted to δ: as the size of the instance is bounded by a function of $|V|$ and δ, it can be solved in time $f(|V|)$ for some function f (assuming δ is a constant). Therefore, we prove a bound of the following form:

THEOREM 2.7. [*] *For every fixed $d \ge 2$, there is a constant δ_d such that there is no $2^{O(|V|^{1-1/d-\epsilon})}$ algorithm for $\mathrm{CSP}(\mathcal{R}_d)$ with domain size at most δ_d for any $\epsilon > 0$, unless ETH fails.*

For the proof of Theorem 2.6 we show how d-dimensional Hamming graphs can be embedded into $(d+1)$-dimensional grids and then invoke a lower bound on d-dimensional Hamming graphs by Alon and Marx [4]. To prove Theorem 2.7, we need to tighten the lower bound on d-dimensional Hamming graphs by revisiting the embedding result of Alon and Marx [4]. Interestingly, this can be done in two ways: either by a construction similar to the one by Alon and Marx [4] together with the randomized rounding of Raghavan and Thompson [33] or by deducing it from the connection between multi-commodity flows and graph Laplacians. Due to lack of space, the proofs of Theorems 2.6 and 2.7 are given in the full version of this paper.

2.1 Geometric CSP problems

In this section, we give lower bounds for CSP problems of certain special forms that are particularly suited for reductions to d-dimensional geometric problems. First, we consider CSPs where every constraint satisfies the following property: we say that a constraint $\langle (u, v), R\rangle$ is a *projection from u to v* if for every $x \in D$, there is at most one $y \in D$ such that $(x, y) \in R$ (a projection from v to u is defined analogously). In other words, the relation R is of the form $\{(x, y) \mid y = p(x)\}$ for some function p; we call this function p the *projection function* associated with the projection constraint. A CSP instance is a *projection CSP* if every binary constraint is a projection; in addition to the binary constraint, the instance may contain any number of unary constraints. Note that an edge uv in the undirected primal graph of a projection CSP does not tell us whether the corresponding constraint is a projection from u to v, or a projection from v to u.

PROPOSITION 2.8. [*] *There is a polynomial time algorithm that, given a CSP instance I on $\mathsf{R}[n, d]$ and domain D, creates an equivalent projection CSP instance I' on $\mathsf{R}[2n, d]$ and domain D^2.*

A *d-dimensional geometric $\leq$-CSP* is a CSP of the following form. The set V of variables is a subset of the vertices of $\mathsf{R}[n,d]$ for some n and the primal graph is an induced subgraph of $\mathsf{R}[n,d]$ (that is, if two variables are adjacent in $\mathsf{R}[n,d]$, then there is a binary constraint on them). The domain is $[\delta]^d$ for some integer $\delta \geq 1$. The instance can contain arbitrary unary constraints, but the binary constraints are of a special form. A *geometric constraint* is a constraint $\langle(\mathbf{a},\mathbf{a}'),R\rangle$ is with $\mathbf{a}'=\mathbf{a}+\mathbf{e}_i$ such that $R=\{((x_1,\ldots,x_d),(y_1,\ldots,y_d)) \mid x_i \leq y_i\}$. In other words, if $\mathbf{a}$ and $\mathbf{a}'$ are adjacent with $\mathbf{a}'$ being larger by one in the i-th coordinate, then the i-th coordinate of the value of $\mathbf{a}'$ should be at least as large as the i-th coordinate of the value of $\mathbf{a}$. Note that a d-dimensional geometric CSP is fully defined by specifying the set of variables and the unary constraints: the binary constraints of the instance are then determined by the definition.

PROPOSITION 2.9. [*] *For every $d \geq 2$, given a projection CSP instance I on $\mathsf{R}[n,d]$ and domain D, one can construct in polynomial time an equivalent d-dimensional geometric $\leq$-CSP instance I' with domain $[2|D|+1]^d$ and $O(n^d)$ variables.*

Theorem 2.6 and Propositions 2.8 and 2.9 imply the following lower bound on geometric CSPs:

THEOREM 2.10. *If for some fixed $d \geq 1$, there is an $f(|V|)n^{o(|V|^{1-1/d})}$ time algorithm for d-dimensional geometric $\leq$-CSP for some function f, then ETH fails.*

3. PACKING PROBLEMS

In this section, we prove the lower bounds on packing d-dimensional unit balls and cubes. These results can be obtained quite easily from the lower bounds for geometric CSP problems proved in Section 2.1. For simplicity of notation, we state the results for open balls and cubes, but of course the same bounds hold for closed balls and cubes.

THEOREM 3.1. *If for some fixed $d \geq 2$, there is an $f(k)n^{o(k^{1-1/d})}$ time algorithm for finding k pairwise nonintersecting d-dimensional open unit balls in a collection of n balls, then ETH fails.*

PROOF. It will be convenient to work with open balls of diameter 1 (that is, radius 1/2) in this proof: then two balls are nonintersecting if and only if the distance of their centers are at least 1. Let I be a d-dimensional $\leq$-CSP instance on variables V and domain $[\delta]^d$. We construct a set B of d-dimensional balls such that B contains a set of $|V|$ pairwise nonintersecting balls if and only if I has a satisfying assignment. Therefore, if we can find k nonintersecting balls in time $f(k)n^{o(k^{1-1/d})}$, then we can solve I in time $f(k)n^{o(|V|^{1-1/d})}$. By Theorem 2.10, this would contradict ETH.

Let $\epsilon = 1/(d\delta^2)$. Let $\mathbf{a}=(a_1,\ldots,a_d)$ be a variable of I and let $\langle(\mathbf{a}),R_{\mathbf{a}}\rangle$ be the unary constraint on $\mathbf{a}$. For every $\mathbf{x}=(x_1,\ldots,x_d)\in R_{\mathbf{a}} \subseteq [\delta]^d$, we introduce into B an open ball of diameter $\frac{1}{2}$ centered at $(a_1+\epsilon x_1,\ldots,a_d+\epsilon x_d)=\mathbf{a}+\epsilon\mathbf{x}$; let $B_{\mathbf{a}}$ be the set of these $|R_{\mathbf{a}}|$ balls. Note that the balls in $B_{\mathbf{a}}$ all intersect each other. Therefore, $B' \subseteq B$ is a set of pairwise nonintersecting balls, then $|B'| \leq |V|$ and $|B'|=|V|$ is possible only if B' contains exactly one ball from each $B_{\mathbf{a}}$. In the following, we prove that there is such a set of $|V|$ pairwise nonintersecting balls if and only if I has a satisfying assignment.

We need the following observation first. Consider two balls centered at $\mathbf{a}+\epsilon\mathbf{x}$ and $\mathbf{a}+\mathbf{e}_i+\epsilon\mathbf{x}'$ for some $\mathbf{x}=(x_1,\ldots,x_d)\in[\delta]^d$ and $\mathbf{x}'=(x'_1,\ldots,x'_d)\in[\delta]^d$. We claim that they are nonintersecting if and only if $x_i \leq x'_i$. Indeed, if $x_i > x'_i$, then the square of the distance of the two centers is

$$\begin{aligned}\sum_{j=1}^{i-1}\epsilon^2(x'_j-x_j)^2+&(1+\epsilon(x'_i-x_i))^2+\sum_{j=i+1}^{d}\epsilon^2(x'_j-x_j)^2\\ &\leq d\epsilon^2\delta^2+(1+\epsilon(x'_i-x_i))^2\\ &\leq \epsilon+(1-\epsilon)^2=1-\epsilon+\epsilon^2<1\end{aligned}$$

(we have used $x'_i,x_i \leq \delta$ in the first inequality and $\epsilon = 1/(d\delta^2)$ in the second inequality). On the other hand, if $x_i \leq x'_i$, then the square of the distance is at least $(1+\epsilon(x'_i-x_i))^2 \geq 1$, hence the two balls do not intersect (recall that the balls are open). This proves our claim. Moreover, it is easy to see that if $\mathbf{a}$ and $\mathbf{a}'$ are not adjacent in $\mathsf{R}[n,d]$, then the balls centered at $\mathbf{a}+\epsilon\mathbf{x}$ and $\mathbf{a}'+\epsilon\mathbf{x}'$ cannot intersect for any $\mathbf{x},\mathbf{x}'\in[\delta]^d$: the square of the distance of the two centers is at least $2(1-\epsilon\delta)^2>1$.

Let f be a satisfying assignment for I. For every variable $\mathbf{a}$, we select the ball $\mathbf{a}+\epsilon f(\mathbf{a})\in B_{\mathbf{a}}$. If $\mathbf{a}$ and $\mathbf{a}'$ are not adjacent, then $\mathbf{a}+\epsilon f(\mathbf{a})$ and $\mathbf{a}'+\epsilon f(\mathbf{a}')$ cannot intersect. If $\mathbf{a}$ and $\mathbf{a}'$ are adjacent, then there is a geometric binary constraint on $\mathbf{a}$ and $\mathbf{a}'$. Therefore, if, say, $\mathbf{a}'=\mathbf{a}+\mathbf{e}_i$, then the binary constraint ensures that the i-th coordinate of $f(\mathbf{a})$ is at most the i-th coordinate of $f(\mathbf{a}')$. By our claim in the previous paragraph, it follows that the balls centered at $\mathbf{a}+\epsilon f(\mathbf{a})$ and $\mathbf{a}'+\epsilon f(\mathbf{a}')$ do not intersect.

Conversely, let $B' \subseteq B$ be a set of $|V|$ pairwise independent balls. This is only possible if for every $\mathbf{a}\in V$, set B' contains a ball from $B_{\mathbf{a}}$, that is, centered at $\mathbf{a}+\epsilon f(\mathbf{a})$ for some $f(\mathbf{a})\in[\delta]^d$. We claim that f is a satisfying assignment of I. First, it satisfies the unary constraints: the fact that $\mathbf{a}+\epsilon f(\mathbf{a})$ is in $B_{\mathbf{a}}$ implies that $f(\mathbf{a})$ satisfies the unary constraint on $\mathbf{a}$. Moreover, let $\mathbf{a}$ and $\mathbf{a}'=\mathbf{a}+\mathbf{e}_i$ be two adjacent variables. Then, as we have observed above, the fact that $\mathbf{a}+\epsilon f(\mathbf{a})$ and $\mathbf{a}'+\epsilon f(\mathbf{a}')$ do not intersect implies that the i-th coordinate of $f(\mathbf{a})$ is at most the i-th coordinate of $f(\mathbf{a}')$. That is, the geometric binary constraint on $\mathbf{a}$ and $\mathbf{a}'$ is satisfied. $\square$

The lower bound for packing d-dimensional axis-parallel unit-side cubes is similar, but there is a slight difference. In the case of unit balls, as we have seen, balls centered at $\mathbf{a}+\epsilon\mathbf{x}$ and $\mathbf{a}+\mathbf{e}_i+\mathbf{e}_j+\epsilon\mathbf{x}'$ cannot intersect (if ϵ is sufficiently small), but this is possible for unit cubes. Therefore, we represent each variable with $2d+1$ cubes: a cube and its two neighbors in each of the d dimensions.

THEOREM 3.2. [*] *If for some fixed $d \geq 2$, there is an $f(k)n^{o(k^{1-1/d})}$ time algorithm for finding k pairwise nonintersecting d-dimensional open axis-parallel unit cubes in a collection of n cubes, then ETH fails.*

PROOF. It will be convenient to work with open cubes of side length 1 in this proof: then two cubes are nonintersecting if and only if the centers differ by at least 1 in at least one of the coordinates. Let I be a d-dimensional $\leq$-CSP instance on variables V and domain $[\delta]^d$. We construct a set B of d-dimensional axis-parallel cubes such that B contains a set of $(2d+1)|V|$ pairwise nonintersecting cubes if and only if I has a satisfying assignment. Therefore, if we can find k nonintersecting cubes in time $f(k)n^{o(k^{1-1/d})}$, then we can solve I in time $f(k)n^{o(|V|^{1-1/d})}$. By Theorem 2.10, this would contradict ETH.

Let $\epsilon = 1/(2\delta)$. Let $\mathbf{a} = (a_1, \dots, a_d)$ be a variable of I and let $\langle(\mathbf{a}), R_{\mathbf{a}}\rangle$ be the unary constraint on $\mathbf{a}$. For every $\mathbf{x} = (x_1, \dots, x_d) \in R_{\mathbf{a}} \subseteq [\delta]^d$, we introduce into B the cube centered at $3\mathbf{a} + \epsilon\mathbf{x}$, and for every $1 \le i \le d$, the two cubes centered at $3\mathbf{a} + \mathbf{e}_i + \epsilon\mathbf{x}$ and $3\mathbf{a} - \mathbf{e}_i + \epsilon\mathbf{x}$. Let us call $B_{\mathbf{a},\mathbf{x}}$ this set of $2d+1$ cubes. Note that the $2d+1$ cubes in $B_{\mathbf{a},\mathbf{x}}$ do not intersect each other (recall that the cubes are open). Moreover, at most $2d+1$ pairwise nonintersecting cubes can be selected from $B_{\mathbf{a}} := \bigcup_{\mathbf{x}\in R_{\mathbf{a}}} B_{\mathbf{a},\mathbf{x}}$: for example, the cubes centered at $\mathbf{a} + \mathbf{e}_i + \epsilon\mathbf{x}$ and $\mathbf{a} + \mathbf{e}_i + \epsilon\mathbf{x}'$ intersect for any $\mathbf{x}, \mathbf{x}' \in [\delta]^d$.

Let f be a satisfying assignment for I. We show that selecting the $2d+1$ cubes $B_{\mathbf{a},f(\mathbf{a})}$ for each variable $\mathbf{a} \in V$ gives a set of $(2d+1)|V|$ pairwise nonintersecting cubes. We claim that if $\mathbf{a}$ and $\mathbf{a}'$ are not adjacent, then the cubes in $B_{\mathbf{a},f(\mathbf{a})}$ do not intersect the cubes in $B_{\mathbf{a}',f(\mathbf{a}')}$. First, if $\mathbf{a}$ and $\mathbf{a}'$ differ by at least 2 in some coordinate, then $3\mathbf{a}$ and $3\mathbf{a}'$ differ by at least 6 in that coordinate and then it is clear that, e.g., the cubes centered at $3\mathbf{a}+\mathbf{e}_i+\epsilon f(\mathbf{a})$ and $3\mathbf{a}'-\mathbf{e}_j+\epsilon f(\mathbf{a}')$ cannot intersect (note that $\epsilon f(\mathbf{a}) \le \epsilon\delta \le 1/2$). On the other hand, if $\mathbf{a}$ and $\mathbf{a}'$ differ in at least two coordinates, then $3\mathbf{a}$ and $3\mathbf{a}$ differ by at least 3 in those two coordinates and $3\mathbf{a} + \epsilon f(\mathbf{a})$ and $3\mathbf{a}' + \epsilon f(\mathbf{a}')$ differ by at least $3 - \epsilon\delta \ge 2$ in those two coordinates. Then adding two unit vectors to the centers cannot decrease both differences to strictly less than 1, that is, $3\mathbf{a} + \mathbf{e}_i + \epsilon f(\mathbf{a})$ and $3\mathbf{a} + \mathbf{e}_j + \epsilon f(\mathbf{a})$ differ by at least 1 in at least one coordinate for any $1 \le i, j \le d$. This means that the cubes in $B_{\mathbf{a},f(\mathbf{a})}$ do not intersect the cubes in $B_{\mathbf{a}',f(\mathbf{a}')}$. Consider now two variables $\mathbf{a}$ and $\mathbf{a}'$ that are adjacent; suppose that $\mathbf{a}' = \mathbf{a}+\mathbf{e}_i$. Then there is a geometric binary constraint on $\mathbf{a}$ and $\mathbf{a}'$, which ensures that the i-th coordinate of $f(\mathbf{a})$ is at most the i-th coordinate of $f(\mathbf{a}')$. It follows that $3\mathbf{a} + \mathbf{e}_i + \epsilon f(\mathbf{a}) \in B_{\mathbf{a},f(\mathbf{a})}$ does not intersect $3\mathbf{a}' - \mathbf{e}_i + f(\mathbf{a}') \in B_{\mathbf{a}',f(\mathbf{a}')}$. Furthermore, the other cubes in $B_{\mathbf{a},f(\mathbf{a})}$ have i-th coordinate less than the i-th coordinate of $3\mathbf{a} + \mathbf{e}_i + \epsilon f(\mathbf{a})$ and the other cubes in $B_{\mathbf{a}',f(\mathbf{a}')}$ have i-th coordinate greater than the i-th coordinate of $3\mathbf{a}'-\mathbf{e}_i+f(\mathbf{a}')$, hence they cannot intersect either.

Conversely, let $B' \subseteq B$ be a set of $(2d+1)|V|$ pairwise independent cubes. This is only possible if for every $\mathbf{a} \in V$, set B' contains $2d+1$ cubes selected from $B_{\mathbf{a}}$; in particular, B' contains a cube centered at $3\mathbf{a} + \epsilon f(\mathbf{a})$ for some $f(\mathbf{a}) \in [\delta]^d$. We claim that f is a satisfying assignment of I. First, it satisfies the unary constraints: the fact that $3\mathbf{a} + \epsilon f(\mathbf{a})$ is in B implies that $f(\mathbf{a})$ satisfies the unary constraint on $\mathbf{a}$. Moreover, let $\mathbf{a}$ and $\mathbf{a}' = \mathbf{a} + \mathbf{e}_i$ be two adjacent variables. Then, as B' contains $2d+1$ cubes from each of $B_{\mathbf{a}}$ and $B_{\mathbf{a}'}$, it has to contain cubes $3\mathbf{a} + \mathbf{e}_i + \epsilon\mathbf{x}_1$ and $3\mathbf{a}' - \mathbf{e}_i + \epsilon\mathbf{x}_2 = 3\mathbf{a} + 2\mathbf{e}_i + \epsilon\mathbf{x}_2$ for some $\mathbf{x}_1, \mathbf{x}_2 \in [\delta]^d$. Now the i-th coordinate of $3\mathbf{a} + \mathbf{e}_i + \epsilon\mathbf{x}_1$ cannot be less than the i-th coordinate of $3\mathbf{a} + \epsilon f(\mathbf{a})$, that is, the i-th coordinate of $\mathbf{x}_1$ is at least the i-th coordinate of $f(\mathbf{a})$. Similarly, the i-th coordinate of $\mathbf{x}_2$ is at least the i-coordinate of $\mathbf{x}_1$, and the i-th coordinate of $f(\mathbf{a}')$ is at least the i-th coordinate of $\mathbf{x}_2$. Putting together, we get that the i-coordinate of $f(\mathbf{a})$ is at least the i-th coordinate of $f(\mathbf{a})$, that is, the geometric binary constraint on $\mathbf{a}$ and $\mathbf{a}'$ is satisfied. $\square$

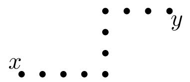

Figure 1: A 1-chain from x to y.

4. THE REDUCTION TO TSP

Let d be a positive integer. An instance of *TSP in d-dimensional Euclidean space* is a pair $\psi = (X, \alpha)$, where X is a finite set of points in $\mathbb{R}^d$, and $\alpha > 0$ is an integer; the goal is to decide whether the length of the shortest TSP tour for X is at most α.

Our reduction is inspired by the NP-hardness proof of TSP in $\mathbb{R}^2$ due to Papadimitriou [32]. We remark that our proof critically assumes $d \ge 3$. However, even though we don't know how to make our argument work in $\mathbb{R}^2$, we can still recover the desired lower bound on the running time for $d = 2$ by the reduction of [32].

We begin by introducing some terminology that will allow us to construct the desired TSP instances. Some of the definitions are from [32]. However, some of them have been extended for our setting. In particular, we use a more elaborate construction that takes advantage of the fact that $d \ge 3$.

1-Chains. Let $x, y \in \mathbb{R}^d$. A *1-chain* from x to y is a sequence of points $\{x_i\}_{i=1}^k$, with $x_1 = x$, $x_k = y$, such that for any $i \in [k-1]$, the points x_i and x_{i+1} differ in exactly one coordinate, and $\|x_i - x_{i+1}\|_1 = 1$ (see Figure 1). We also require that for any $i, j \in [k]$, with $|i - j| \le 20$, we have $\|x_i - x_j\|_1 \ge \||i - j|\|$.

2-Chains. Let $x, y \in \mathbb{R}^d$, and let $\theta \in [0, \pi/100)$. A *θ-ribbon* from x to y is a pair $(P, \mathcal{H})$, where P is a simple polygonal curve with endpoints x and y, consisting of k line segments $s_1, \dots, s_k$, and $\mathcal{H} = \{H_i\}_{i=1}^k$ is a family of 2-dimensional planes in $\mathbb{R}^d$, satisfying the following conditions:

(I) For any $i, j \in [k]$, with $|i - j| > 20$, for any $p \in s_i$, $q \in s_j$, we have $\|p - q\|_2 > 40$.

(II) The segments s_1, and s_k are of length 1. All other segments are of length 2.

(III) For every segment s_i, we have $s_i \subset H_i$.

(IV) For every two consecutive segments s_i, s_{i+1}, at least one of the following conditions holds:

(IV-1) Let p be the common endpoint of s_i, and s_{i+1}, and let ℓ bet the line in H_i passing through p, and being normal to s_i. Then the 2-plane H_{i+1} is obtained by rotating H_i around ℓ by some angle of at most θ.

(IV-2) The segments s_i and s_{i+1} are collinear, and H_{i+1} is obtained by rotating H_i around s_i by some angle of at most θ.

Given a θ-ribbon $R = (P, \mathcal{H})$ from x to y, we define a set of points $Y = Y(R)$, which we refer to as a *2-chain* with *angular defect* θ (or simply a 2-chain when θ is clear from the context) from x to y, corresponding to R. We set $Y := \{x, y\} \cup \bigcup_{i=1}^{k-1}\{p_i, q_i\}$, where for any i, the points $p_i, q_i \in \mathbb{R}^d$ are defined as follows: Let a be the common endpoint of

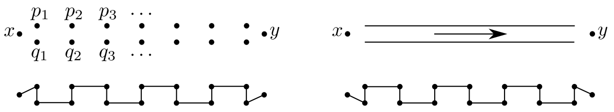

Figure 2: A 2-chain from x to y (top left), its schematic abbreviation (top right), and the two possible optimal paths: mode 1 (bottom left), and mode 2 (bottom right).

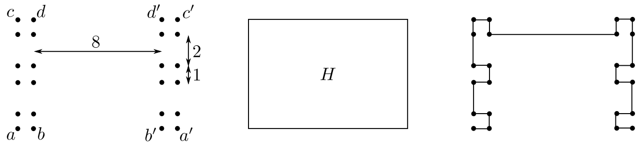

Figure 3: A configuration-H (left), its schematic abbreviation (middle), and an optimal path from a to a' (right).

s_i, and s_{i+1}. Let ℓ be the line in H_{i+1} normal to s_{i+1} that passes through a. Let p_i, q_i be the two points in ℓ that are at distance $1/2$ from a. We assign p_i, q_i so that for any two consecutive segments s_j, s_{j+1}, with $j \in \{1, \ldots, k-1\}$, the angle between the vectors $p_i - q_i$, and $p_{i+1} - q_{i+1}$, is at most θ (this is always possible since $(P, \mathcal{H})$ is a θ-ribbon). Notice that there are precisely two distinct possibilities of assigning the points $p_1, \ldots, p_{k-1}$, and $q_1, \ldots, q_{k-1}$. We refer to the points $\{p_i\}_{i=1}^k$ as the *left side*, and the points $\{q_i\}_{i=1}^k$ as the *right side* of the 2-chain. This concludes the definition of a 2-chain (see Figure 2).

LEMMA 4.1. *There exists a constant $\theta^* > 0$, such that the following holds. Let $x, y \in \mathbb{R}^d$, and let Y be a 2-chain from x to y, and with angular defect θ^*. Let $\{p_i\}_{i=1}^{k-1}$, and $\{q_i\}_{i=1}^{k-1}$ be the left, and right sides of Y respectively. Then, there are precisely two optimal Traveling Salesperson paths from x to y for the set Y: the first one is $xp_1q_1q_2p_2p_3q_3 \ldots y$, and the second one is $xq_1p_1p_2q_2q_3p_3 \ldots y$. In the former case we say that Y is traversed in* mode 1, *and in the latter case in* mode 2.

The configuration-H. We recall the following gadget from [32]. A set of points $Y \subset \mathbb{R}^d$ is called a *configuration-***H** if there exists a 2-plane h containing Y, so that Y on h appears as in Figure 3[3].

LEMMA 4.2 (PAPADIMITRIOU [32]). *Among all Traveling Salesperson paths having as endpoints two of the points in a, a', b, b', c, c', c, d and d', there are 4 optimal paths with length 32, namely those with endpoints (a, a'), (b, b'), (c, c'), (d, d').*

Let $(P, \mathcal{H})$ be a θ-ribbon from x to y, with $\mathcal{H} = \{H_i\}_{i=1}^k$, and let Y be the corresponding 2-chain. Suppose that there exists some *odd* $j \in \{1, \ldots, k-15\}$, such that for any $r, r' \in \{j, \ldots, j+14\}$, we have $H_r = H_{r'}$. Let Z be a configuration-**H** contained in the 2-plane H_j. Suppose further that $Z \cup Y$

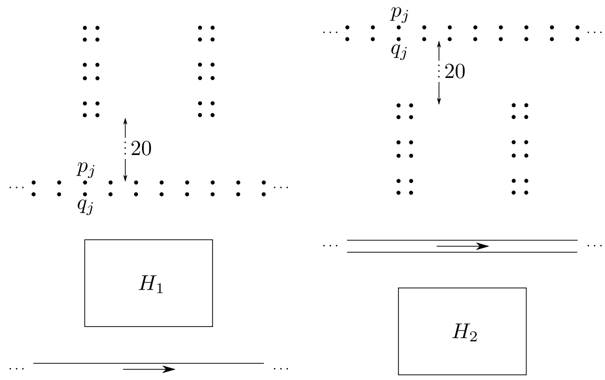

Figure 4: A configuration-H H_1 that is a left neighbor of a 2-chain, and its schematic abbreviation (left). A configuration-H H_2 that is a right neighbor of a 2-chain, and its schematic abbreviation (right).

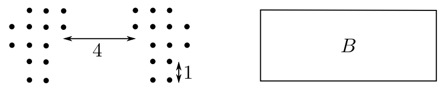

Figure 5: A configuration-B (left), and its schematic abbreviation (right).

appear in H_j as in Figure 4 (top). Then, we say that the configuration-**H** Z is a *left neighbor* of the 2-chain Y at j. Similarly, we define a *right neighbor* of a 2-chain (see bottom of Figure 4).

The configuration-B. Following [32], we say that a set of points $Y \subset \mathbb{R}^d$ is a *configuration B* if there exists a 2-plane h containing Z, so that Z on h appears as in Figure 5.

Let $(P, \mathcal{H})$ be a θ-ribbon from x to y, with $\mathcal{H} = \{H_i\}_{i=1}^k$, and let Y be the corresponding 2-chain. Suppose that there exists some *odd* $j \in [k-15]$, such that for any $r, r' \in \{j, \ldots, j+14\}$, we have $H_r = H_{r'}$. Let Z be a configuration-**B** contained in the 2-plane H_j. Suppose that we replace a subset of Y by a configuration-**B** Z, such that Z is contained in H_j, and is as in figure 6. Then, we say that resulting point-set Y' is obtained by *attaching* Z to Y at j.

The reduction. Let (V, D, C) be an instance of a binary constraint satisfaction problem, with primal graph $G = \mathsf{R}[n, d]$. We may assume w.l.o.g. that every constraint is binary (i.e. there are no unary constraints).

First, we encode the variables. Let $\gamma = \gamma(D)$ be a parameter to be determined later. For any $\mathbf{r} = (r_1, \ldots, r_d) \in \mathbb{Z}^d$, let $U(\mathbf{r}) := \{(x_1, \ldots, x_d) \in \mathbb{R}^d : \text{ for all } i \in [d], r_i\gamma \leq x_i < (r_i + 1)\gamma\}$. Let us identify V with $[n]^d$, in the obvious way. For each $\mathbf{u} \in V$, we introduce a family of d 2-chains $\Gamma(\mathbf{u}, 1), \ldots, \Gamma(\mathbf{u}, |D|)$. We will ensure that $\bigcup_{i=1}^{|D|} \Gamma(\mathbf{u}, i) \subset U(\mathbf{u}) \cup \bigcup_{i=1}^{d} (U(\mathbf{u} - \mathbf{e_i}) \cup U(\mathbf{u} + \mathbf{e_i}))$, where $\mathbf{e}_1, \ldots, \mathbf{e}_d$ is the standard orthonormal basis in $\mathbb{R}^d$. We need to enforce that in any optimal solution, exactly one of the 2-chains $\Gamma(\mathbf{u}, 1)$, $\ldots$, $\Gamma(\mathbf{u}, |D|)$ is traversed in mode 2. Intuitively, this will correspond to assigning the value i to variable $\mathbf{u}$. To that end, we construct the 2-chains $\Gamma(\mathbf{u}, i)$ such that there exists a 2-plane $h(\mathbf{u}, i)$, where subsets of the 2-chains are arranged as in Figure 7. Namely, for any $i \in [|D|]$ we introduce a

[3]The distance that is set to 8 in Figure 3, was set to 6 in the original construction from [32]. This appears to be a minor error in [32].

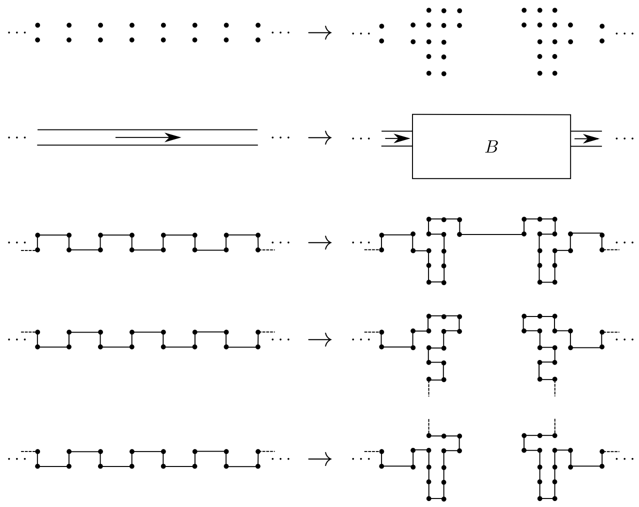

Figure 6: From top to bottom: Attaching a configuration-B to a 2-chain, its schematic abbreviation, the optimal path for mode 2, and the two possible optimal paths for mode 1.

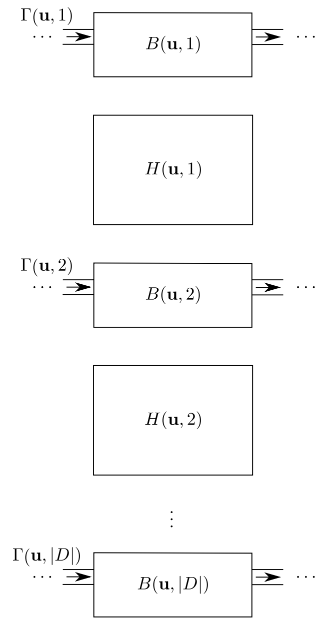

Figure 7: Gadget corresponding to some variable u.

configuration-**B** $B(\mathbf{u}, i)$ that is attached to $\Gamma(\mathbf{u}, i)$ at some j_i. Moreover, for any $i \in [|D|-1]$, we introduce a configuration-**H** $H(\mathbf{u}, i)$, such that $H(\mathbf{u}, i)$ is the right neighbor of $\Gamma(\mathbf{u}, i)$ at j_i, and the left neighbor of $\Gamma(\mathbf{u}, i+1)$ at j_{i+1}.

Next, we encode the constraints. Let $\langle(\mathbf{u}, \mathbf{v}), R\rangle \in C$ be a constraint. For every pair of values $(i, j) \in D^2$, with $(i, j) \notin R$, we need to ensure that in any optimal solution, at most one of the 2-chains $\Gamma(\mathbf{u}, i)$, $\Gamma(\mathbf{v}, j)$ is traversed in mode 1. To that end, we add two new vertices $x(\mathbf{u}, i, \mathbf{v}, j)$, $y(\mathbf{u}, i, \mathbf{v}, j)$, and a new 2-chain $\Gamma(\mathbf{u}, i, \mathbf{v}, j)$ from $x(\mathbf{u}, i, \mathbf{v}, j)$ to $y(\mathbf{u}, i, \mathbf{v}, j)$. We arrange the 2-chains such that there exists a 2-plane $h(\mathbf{u}, \mathbf{v}, i, j)$, with subsets of the 2-chains $\Gamma(\mathbf{u}, i)$, and $\Gamma(\mathbf{v}, j)$ being arranged in $h(\mathbf{u}, \mathbf{v}, i, j)$ as in Figure 8. Namely, we introduce configurations-**B** $B(\mathbf{u}, i, \mathbf{v}, j, 1)$, $B(\mathbf{u}, i, \mathbf{v}, j, 2)$, $B(\mathbf{u}, i, \mathbf{v}, j, 3)$, such that $B(\mathbf{u}, i, \mathbf{v}, j, 1)$ is attached to $\Gamma(\mathbf{u}, i)$ at some ℓ_i, $B(\mathbf{u}, i, \mathbf{v}, j, 2)$ is attached to $\Gamma(\mathbf{u}, i, \mathbf{v}, j)$ at some ℓ_2, and $B(\mathbf{u}, i, \mathbf{v}, j, 3)$ is attached to $\Gamma(\mathbf{v}, j)$ at some ℓ_3. Moreover, we add a configuration-**H** $H(\mathbf{u}, i, \mathbf{v}, j, 1)$ that is a right neighbor of $\Gamma(\mathbf{u}, i)$ at ℓ_1, and a left neighbor of $\Gamma(\mathbf{u}, i, \mathbf{v}, j)$ at ℓ_2, and a configuration-**H** $H(\mathbf{u}, i, \mathbf{v}, j, 2)$ that is a right neighbor of $\Gamma(\mathbf{u}, i, \mathbf{v}, j)$ at ℓ_2, and a left neighbor of $\Gamma(\mathbf{v}, j)$ at ℓ_3.

Finally, we need to ensure that the optimal solution induces a traversal of all the 2-chains, such that each 2-chain is traversed without interruptions. This can be done by introducing 1-chains between the endpoints of 2-chains which we want to appear consecutively in the optimal traversal. Initially, we unmark all 2-chains in the construction. Observe that the graph G is Hamiltonian. Fix a Hamiltonian path P in G. We construct a total ordering of all the 2-chains in the construction as follows. We start from the empty ordering, and we consider all vertices in the order they are visited by P. When considering a vertex u, we extend the ordering by appending all the 2-chains $\Gamma(\mathbf{u}, i)$, for all $i \in [|D|]$. Next, we also append all the 2-chains $\Gamma(\mathbf{u}, i, \mathbf{v}, j)$, for all $i, j \in [|D|]$, and $v \in V(G)$, that we have not added to the ordering yet. This process clearly results in a total ordering of all the 2-chains in the construction. Let k be the total number of 2-chains, for any $i \in [k]$, let p_i, q_i be points such that the i-th chain is from p_i to q_i. For any $i \in [k-1]$, we add a 1-chain from q_i to q_{i+1}. When adding a 1-chain Y between the endpoints of two 2-chains involving only one variable $\mathbf{u}$ (i.e. $\Gamma(\mathbf{u}, i)$, and $\Gamma(\mathbf{u}, i+1)$ for some i), we ensure that $Y \subset U(\mathbf{u})$. Similarly, when we add a 1-chain Y between the endpoints of two 2-chains involving only two variable $\mathbf{u}, \mathbf{v}$ (i.e. either $\Gamma(\mathbf{u}, i)$ and $\Gamma(\mathbf{u}, j, \mathbf{v}, \ell)$ for some i, j, ℓ, or $\Gamma(\mathbf{u}, i)$ and $\Gamma(\mathbf{v}, j)$ for some i, j), we ensure that $Y \subset U(\mathbf{u}) \cup U(\mathbf{v})$.

Finally, we introduce two points p^*, and q^*, and we add a 1-chain from p^* to p_1, and a 1-chain from q_t to q^*. We choose the point p^* to be in $U(-2 \cdot \mathbf{e}_1)$, and the point q^* to be in $U((n+2) \cdot \mathbf{e}_1)$. We can clearly chose the 1-chains from p^* to p_1, and from q_t to q^* so that the following is satisfied.

Lemma 4.3. *Any optimal Traveling Salesperson path from in the constructed instance has endpoints p^* and q^*.*

Lemma 4.4. [$*$] *Let $d \geq 2$. Let $\varphi = (V, D, C)$ be an instance of a constraint satisfaction problem with domain size $|D| = \delta$, with constraint graph $G = \mathsf{R}[n, d]$. Then, there exists a polynomially-time computable instance (X, α) of TSP in d-dimensional Euclidean space, with $|X| \leq n^d \cdot |D|^{O(1)}$, such that the length of the shortest TSP tour for X is at most α, if and only if φ is satisfiable.*

Proof (of Theorem 1.1). It follows by Theorem 2.6 & Lemma 4.4. □

Recall that cycle-TSP is the variant of TSP where one seeks to find a cycle visiting all points. We can prove the same lower bound for cycle-TSP, using a simple modification of the above reduction. We remark that the same modification was used in [32] to show that cycle-TSP in the Euclidean plane is NP-complete.

Theorem 4.5. *If for some $d \geq 2$ and $\epsilon > 0$, cycle-TSP in d-dimensional Euclidean space can be solved in time $2^{O(n^{1-1/d-\epsilon})}$, then ETH fails.*

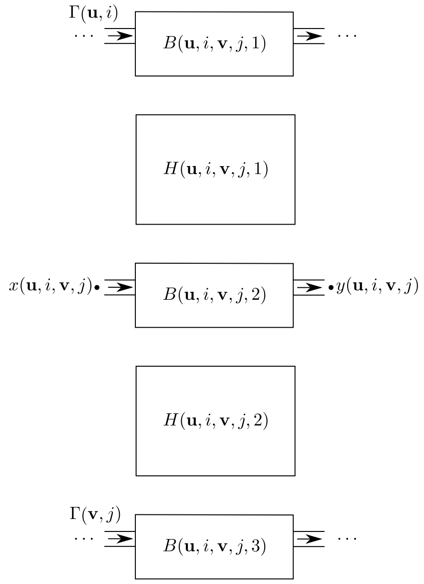

Figure 8: Part of a gadget encoding $(i, j) \notin R$, for some constraint $\langle(\mathbf{u}, \mathbf{v}), R\rangle$.

PROOF. We use the same reduction as for the case of path-TSP above. The only modification needed is to connect p^* with q^* via a 1-chain, that is at distance at least 20, say, from all other gadgets used in the reduction. $\square$

5. AN EXACT ALGORITHM FOR PACKING UNIT BALLS IN $\mathbb{R}^d$

We present an $n^{O(k^{1-1/d})}$ time algorithm for finding a pairwise nonintersecting set k unit balls in d-dimensional space (generalizing the result of Alber and Fiala [3] for $d = 2$). As the technique (combining a sweeping argument, brute force, and dynamic programming) is fairly standard, we keep the discussion brief. Exactly the same argument works for finding k pairwise nonintersecting d-dimensional unit cubes.

THEOREM 5.1. *Let $d \geq 2$ be a fixed constant. There exists an algorithm that, given a set X of unit d-dimensional balls in $\mathbb{R}^d$ and an integer $k \geq 0$, decides in time $n^{O(k^{1-1/d})}$ whether there exist k pairwise nonintersecting balls in X.*

PROOF. Let $s = k^{1/d}$. For any $i \in \{0, \dots, s-1\}$, and for any $j \in [d]$, let $H(i, j) = \{(x_1, \dots, x_d) \in \mathbb{R}^d :$ there exists $r \in \mathbb{Z}$ s.t. $x_j = i + r \cdot s\}$. Note that $H(i, j)$ is the union of parallel $(d-1)$-dimensional hyperplanes, with any two consecutive ones being at distance s. For any $i \in \{0, \dots, s-1\}$, let $H(i) = \bigcup_{j=1}^{d} H(i, j)$. Observe that $\mathbb{R}^d \setminus H(i)$ is the union of open hypercubes.

Let $A \subset \mathbb{R}^d$ be a unit d-dimensional ball. By the union bound, we have

$$\begin{aligned}
&\mathbf{Pr}_{i\in\{0,\dots,s-1\}}\left[A \cap H(i) \neq \emptyset\right] \\
&\qquad \leq \sum_{j=1}^{d} \mathbf{Pr}_{i\in\{0,\dots,s-1\}}\left[A \cap H(i,j) \neq \emptyset\right] \\
&\qquad = O(d/s) = O(1/s),
\end{aligned}$$

where $i \in \{0, \dots, s-1\}$ is chosen uniformly at random.

Suppose that there exists a subset $X^* \subseteq X$ of pairwise nonintersecting balls with $|X^*| = k$. By linearity of expectation, we obtain

$$\begin{aligned}
&\mathbf{E}_{i\in\{0,\dots,s-1\}}[|\{A \in X^* : A \cap H(i) \neq \emptyset\}|] \\
&\qquad \leq \sum_{A \in X^*} \mathbf{Pr}_{i\in\{0,\dots,s-1\}}\left[A \cap H(i) \neq \emptyset\right] \\
&\qquad = O(k/s) = O(k^{1-1/d}).
\end{aligned}$$

By averaging, there exists $i^* \in \{0, \dots, s-1\}$, such that

$$|\{A \in X^* : A \cap H(i^*) \neq \emptyset\}| = O(k^{1-1/d}).$$

The algorithm proceeds as follows. We guess a value $i \in \{0, \dots, s-1\}$, and we guess a subset Y of at most $O(k^{1-1/d})$ balls in X that intersect $H(i)$. Let $X' = X \setminus \{A \in X : A \cap H(i) \neq \emptyset\}$. Let $\mathcal{C}$ be the set of open hypercubes in $\mathbb{R}^d \setminus H(i)$. We partition X' into a collection of subsets $\{X'_C\}_{C\in\mathcal{C}}$, where every X'_C contains all the balls that intersect the open hypercube $C \in \mathcal{C}$. For each $C \in \mathcal{C}$, we define the subset $X''_C \subseteq X'_C$ containing all balls that do not intersect any of the balls in Y, i.e. $X''_C = \{A \in X'_C : A \cap \bigcup_{A' \in Y} A' = \emptyset\}$.

We now proceed to compute a maximum set of pairwise nonintersecting balls in each X''_C. By translating, we may assume that $C = (0, s)^d$. For each $j \in \{0, \dots, s-1\}$, let $C(j) = C \cap \left([j, j+1] \times [0, s]^{d-1}\right)$. Let $X''_{C,j}$ be the set of all balls intersecting $C(j)$ and let $Y''_{C,j} = \bigcup_{j'=0}^{j'=j} X''_{C,j'}$. For any $j \in \{0, \dots, s-1\}$, there can be at most $O(s)$ balls in the solution X^* that intersect $C(j)$. For each $j \in \{0, \dots, s-1\}$, we compute all possible subsets of at most $O(s)$ pairwise nonintersecting balls in $X''_{C,j}$; let $\mathcal{X}_{C,j}$ be the collection of all these sets. We can now compute an maximum set Y_C of pairwise nonintersecting balls in X''_C via dynamic programming, as follows. For every $j \in \{0, \dots, s-1\}$ and subset $Z \in \mathcal{X}_{C,j}$, we compute the maximum size of a set of pairwise nonintersecting balls in $Y''_{C,j}$ whose intersection with $X''_{C,j}$ is precisely Z. The important observation is that the sets $X''_{C,j}$ and $X''_{C,j-2}$ are disjoint. Hence the maximum for a given j and $Z \in \mathcal{X}_{C,j}$ can be computed if we know the maximum for $j-1$ and every $Z \in \mathcal{X}_{C,j-1}$.

After computing the maximum set Y_C for each open hypercube C, we output the set of pairwise nonintersecting balls in X that we find is $Y \cup \bigcup_{C\in\mathcal{C}} Y_C$. The final set of balls is the maximum such set computed for all choices of i, and Y. This concludes the description of the algorithm.

Let us first argue that the algorithm is correct. Indeed, when we choose $i = i^*$, the algorithm will eventually correctly choose the correct set $Y = X^* \cap H(i^*)$. Once we remove the balls that intersect $H(i^*)$, and all the balls that intersect the balls in Y, the remaining subproblems are independent, and are solved optimally. Therefore, the resulting global solution is optimal.

Lastly, let us bound the running time. There are $s = n^{O(1/d)}$ choices for i, and for each such choice, there are at most $n^{O(k^{1-1/d})}$ choices for Y. For every such choice, we solve at most n different subproblems (one for every open hypercube in $\mathcal{C}$). Each subproblem uses dynamic programming with a table with $O(s)$ entries, where each entry stores $n^{O(s)} = n^{O(k^{1-1/d})}$ different partial solutions. It follows that the total running time is $n^{O(k^{1-1/d})}$, as required. $\square$

6. REFERENCES

[1] I. Adler, G. Gottlob, and M. Grohe. Hypertree width and related hypergraph invariants. *European J. Combin.*, 28(8):2167–2181, 2007.

[2] P. K. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *ACM Comput. Surv.*, 30(4):412–458, 1998.

[3] J. Alber and J. Fiala. Geometric separation and exact solutions for the parameterized independent set problem on disk graphs. *J. Alg.*, 52(2):134–151, 2004.

[4] N. Alon and D. Marx. Sparse balanced partitions and the complexity of subgraph problems. *SIAM J. Discrete Math.*, 25(2):631–644, 2011.

[5] R. Bellman. Dynamic programming treatment of the travelling salesman problem. *J. ACM*, 9(1):61–63, Jan. 1962.

[6] S. Cabello, P. Giannopoulos, and C. Knauer. On the parameterized complexity of d-dimensional point set pattern matching. *Inf. Process. Lett.*, 105(2):73–77, 2008.

[7] S. Cabello, P. Giannopoulos, C. Knauer, D. Marx, and G. Rote. Geometric clustering: Fixed-parameter tractability and lower bounds with respect to the dimension. *ACM Trans. on Algorithms*, 7(4):43, 2011.

[8] T. M. Chan. A (slightly) faster algorithm for Klee's measure problem. In *Symposium on Computational Geometry*, pages 94–100, 2008.

[9] H. Chen and M. Grohe. Constraint satisfaction with succinctly specified relations. *J. Comput. Syst. Sci.*, 76(8):847–860, 2010.

[10] J. Chen, X. Huang, I. A. Kanj, and G. Xia. Linear FPT reductions and computational lower bounds. In *Proc. of the 36th Ann. ACM Symp. on Theory of Comp.*, pages 212–221, New York, 2004.

[11] E. C. Freuder. Complexity of k-tree structured constraint satisfaction problems. In *Proc. of AAAI-90*, pages 4–9, Boston, MA, 1990.

[12] P. Giannopoulos, C. Knauer, and G. Rote. The parameterized complexity of some geometric problems in unbounded dimension. In *IWPEC*, pages 198–209, 2009.

[13] P. Giannopoulos, C. Knauer, G. Rote, and D. Werner. Fixed-parameter tractability and lower bounds for stabbing problems. *Comput. Geom.*, 46(7):839–860, 2013.

[14] P. Giannopoulos, C. Knauer, M. Wahlström, and D. Werner. Hardness of discrepancy computation and -net verification in high dimension. *J. Complexity*, 28(2):162–176, 2012.

[15] G. Gottlob, M. Grohe, N. Musliu, M. Samer, and F. Scarcello. Hypertree decompositions: structure, algorithms, and applications. In *Graph-theoretic concepts in computer science*, volume 3787 of *Lecture Notes in Comput. Sci.*, pages 1–15. Springer, Berlin, 2005.

[16] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences*, 64:579–627, 2002.

[17] M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1, 2007.

[18] M. Grohe and D. Marx. Constraint solving via fractional edge covers. In *SODA '06: Proc. of the 17th Ann. ACM-SIAM Symp. on Disc. Alg.*, pages 289–298, New York, NY, USA, 2006. ACM Press.

[19] M. Grohe, T. Schwentick, and L. Segoufin. When is the evaluation of conjunctive queries tractable? In *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 657–666, New York, NY, USA, 2001. ACM Press.

[20] M. Held and R. Karp. The traveling-salesman problem and minimum spanning trees: Part II. *Mathematical Programming*, 1(1):6–25, 1971.

[21] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. System Sci.*, 63(4):512–530, 2001.

[22] C. Knauer. The complexity of geometric problems in high dimension. In *TAMC*, pages 40–49, 2010.

[23] C. Knauer, H. R. Tiwary, and D. Werner. On the computational complexity of ham-sandwich cuts, helly sets, and related problems. In *STACS*, pages 649–660, 2011.

[24] K. Kozawa, Y. Otachi, and K. Yamazaki. The carving-width of generalized hypercubes. *Discrete Mathematics*, 310(21):2867–2876, 2010.

[25] D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the Exponential Time Hypothesis. *Bulletin of the EATCS*, 84:41–71, 2011.

[26] D. Marx. On the optimality of planar and geometric approximation schemes. In *48th Ann. IEEE Symp. on Found. of Comp. Sci.*, pages 338–348, 2007.

[27] D. Marx. Approximating fractional hypertree width. *ACM Trans. Algorithms*, 6(2):1–17, 2010.

[28] D. Marx. Can you beat treewidth? *Theory of Computing*, 6(1):85–112, 2010.

[29] D. Marx. Tractable structures for constraint satisfaction with truth tables. *Theory of Computing Systems*, 48:444–464, 2011.

[30] N. Megiddo. Linear-time algorithms for linear programming in R^3 and related problems. *SIAM J. Comput.*, 12(4):759–776, 1983.

[31] N. Megiddo. On the complexity of some geometric problems in unbounded dimension. *J. Symb. Comput.*, 10(3/4):327–334, 1990.

[32] C. H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theor. Comput. Sci.*, 4(3):237–244, 1977.

[33] P. Raghavan and C. D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[34] W. D. Smith and N. C. Wormald. Geometric separator theorems & applications. In *FOCS*, pages 232–243, 1998.

[35] D. M. Thilikos, M. J. Serna, and H. L. Bodlaender. Constructive linear time algorithms for small cutwidth and carving-width. In *ISAAC*, pages 192–203, 2000.