

# Embeddings and labeling schemes for A\*

Talya Eden <sup>\*</sup>  
Boston University and MIT

Piotr Indyk <sup>†</sup>  
MIT

Haike Xu <sup>‡</sup>  
Tsinghua University

## Abstract

A\* is a classic and popular method for graphs search and path finding. It assumes the existence of a heuristic function  $h(u, t)$  that estimates the shortest distance from any input node  $u$  to the destination  $t$ . Traditionally, heuristics have been handcrafted by domain experts. However, over the last few years, there has been a growing interest in learning heuristic functions. Such learned heuristics estimate the distance between given nodes based on “features” of those nodes.

In this paper we formalize and initiate the study of such feature-based heuristics. In particular, we consider heuristics induced by norm embeddings and distance labeling schemes, and provide lower bounds for the tradeoffs between the number of dimensions or bits used to represent each graph node, and the running time of the A\* algorithm. We also show that, under natural assumptions, our lower bounds are almost optimal.

## 1 Introduction

A\* is a classic and popular method for graphs search and path finding. It provides a method for computing the shortest path in a given weighted graph  $G = (V, E, W)$  from a source  $s$  to a destination  $t$  that is often significantly faster than classic algorithms for this problem. It assumes the existence of a heuristic function  $h(u, t)$  that estimates the shortest distance from any input node  $u$  to the destination  $t$ . The algorithm uses the greedy approach, at each step selecting a node  $u$  that minimizes  $d(s, u) + h(u, t)$ , where  $d(s, u)$  is the (already computed) distance from  $s$  to  $u$ . Alternatively, A\* can be viewed as a variant of Dijkstra algorithm, with the distance function  $d(u, v)$  replaced by  $d(u, v) + h(v, t) - h(u, t)$ . Since its inception in the 1960s, the algorithm has found many application, e.g., to robotics [Yonetani et al., 2021, Bhardwaj et al., 2017], game solving [Cui and Shi, 2011], computational organic chemistry [Chen et al., 2020]. Over the last two decades, it has been also shown to be highly effective for “standard” shortest path computation tasks in road networks [Goldberg and Harrelson, 2005].

The performance of A\* is governed by the quality of the heuristic function that estimates the distance from a given node to the target. For example, if the heuristic function is perfect, i.e.,  $h(u, t) = d(u, t)$  for all nodes  $u$ , then the number of vertices scanned by A\* is proportional to the number of hops in the shortest path. In practice, the heuristic function is carefully selected based on the properties of the underlying class of problems. E.g., for graphs whose vertices corresponds to points in the plane (e.g., shortest paths in road networks or paths avoiding 2D obstacles), typical choices include Euclidean, Manhattan or Chebyshev distances. In many cases, the task of identifying an appropriate heuristic for a given problem can be quite difficult.

Over the last few years, there has been a growing interest in *learning* heuristic functions based on the properties of input graphs, see e.g., [Yonetani et al., 2021, Bhardwaj et al., 2017, Chen et al., 2020]. Such predictors are trained on a collection of input graphs. After training, the predictor estimates the

---

<sup>\*</sup>CSAIL at MIT, Boston University Department of Computer Science, [talyaa01@gmail.com](mailto:talyaa01@gmail.com). Partially supported by the NSF Grant CCF-1740751, the Eric and Wendy Schmidt Fund, Ben-Gurion University, and the Computer Science Department at Boston University.

<sup>†</sup>CSAIL at MIT, [indyk@mit.edu](mailto:indyk@mit.edu). Partially supported by the NSF TRIPODS program (awards CCF-1740751 and DMS-2022448) and Simons Investigator Award.

<sup>‡</sup>IIIS, Tsinghua University, [xhk18@mails.tsinghua.edu.cn](mailto:xhk18@mails.tsinghua.edu.cn).

distance between given nodes based on the pre-computed “features” of those nodes, plus possibly other auxiliary information (see Section 1.4 for a discussion). The features are vectors in a  $d$ -dimensional space, either handcrafted based on the domain knowledge, or trained using machine learning methods. Thus, such learned heuristics bear similarity to *metric embeddings* [Naor, 2018] or *distance labels* [Gavoille et al., 2004], two notions that have been extensively investigated in theoretical computer science and mathematics. The notions of the quality of embeddings in the above lines of research is, however, quite different from what is required in the  $A^*$  context. Specifically, a typical objective of metric embeddings or distance labelling is to preserve the distances between every pair of vertices up to some (multiplicative) approximation factor. On the other hand, in the context of  $A^*$  search, it is acceptable if the estimated distances deviate significantly from the ground truth, as long as the  $A^*$  search process that uses those estimates does not take too much time. At the same time, approximate estimates of the distances in the metric embedding sense might not be useful if they are not able to disambiguate between many different short paths, leading to high running times. This necessitates studying different notions of embeddings and labeling that are tailored to the  $A^*$  context.

Motivated by these considerations, in this paper we formalize and initiate the study of embeddings and distance labeling schemes that induce efficient  $A^*$  heuristics. In particular, we study tradeoffs between the dimensionality of the embeddings (or the length of the labels in labeling schemes) and the complexity of the  $A^*$  search process. Our focus is on *average-case* performance of  $A^*$ , where the average is taken over all pairs of vertices in the graph. For lower bounds, average-case results provide much stronger limitations than worst-case results. At the same time, those lower bounds naturally complement our algorithms, which rely on the average-case assumptions for technical reasons.

We start by formally defining the metric that we will use to evaluate the quality of a heuristic.

**Definition 1.1.** For a graph  $G = (V, E, W)$ , and a heuristic  $h$ ,  $P(s, t)$  denote the set of vertices on the shortest path between  $s$  and  $t$  with the maximal number of hops, and let  $S_h(s, t)$  be the set of vertices scanned by  $A^*$  given  $s, t$  and  $h$ . Further let  $p(s, t) = |P(s, t)|$ . We say that  $h$  has an additive overhead  $c$  on average if

$$\mathbb{E}_{s, t \in V}[|S_h(s, t)| - p(s, t)] \leq c$$

where  $s$  and  $t$  are chosen independently and uniformly at random from  $V$ .

Informally, the above definition measures the number of “extra” vertices that needs to be scanned, in addition to the size of the shortest path. We note that one could alternatively define the overhead in a “multiplicative” way, by computing the ratio between the number of vertices scanned and the number of hops in a shortest paths. However, several of our results (esp. the lower bounds) rely on instances where we are guaranteed that all shortest paths have constant number of hops. In this case, an additive overhead of  $T$  automatically translates into a multiplicative overhead of  $\Omega(T)$ .

In this paper we focus on heuristics that are *consistent*. The consistency property, introduced by Hart et al. [1968] and stated below, implies that the  $A^*$  search algorithm using the heuristic correctly identifies the shortest path upon termination.

**Definition 1.2** (Consistent heuristic function). A heuristic function  $h(s, t)$  which estimates  $\text{dist}(s, t)$  is consistent if for any destination  $t$  and edge  $(u, v)$ , the modified cost  $w_t(u, v) = w(u, v) + h(v, t) - h(u, t)$  is non-negative and also  $h(t, t) = 0$ .

## 1.1 Heuristic types

We now formally define the types of heuristics that we study in this paper. Specifically, we introduce heuristic induced by norm embeddings and labelling schemes, as well as embeddings induced by “beacons”, which can be viewed as a special type of norm embeddings induced by the  $\ell_\infty$  norm.

**Definition 1.3** (Norm heuristic). A heuristic function  $h : V \times V \rightarrow \mathbb{R}$  is a norm heuristic induced by a function  $\pi : V \rightarrow \mathbb{R}^d$  and the  $l_p$  norm if  $h(s, t) = \|\pi(s) - \pi(t)\|_p$  for any pair of vertices  $s, t \in V$ . The dimension  $d$  is referred to as the dimension of  $h$ .

Note that  $h(s, t)$  as defined above satisfies triangle inequality, i.e.  $h(s, u) + h(u, t) \geq h(s, t)$  for any  $s, t, u \in V$ .

**Definition 1.4** (Labeling heuristic). *A heuristic function  $h : V \times V \rightarrow \{0 \dots C\}$  is a labeling heuristic induced by functions  $f : V \rightarrow \{0 \dots C\}^L$  and  $g : \{0 \dots C\}^L \times \{0 \dots C\}^L \rightarrow \{0 \dots C\}$ , if  $h(s, t) = g(f(s), f(t))$  for any pair of vertices  $s, t \in V$ . Here the parameter  $C$  is assumed to be polynomial in  $|V|$ , and  $L$  is referred to as the label length.*

Note that  $f$  and  $g$  must be fixed functions working for any graph and label pairs. Furthermore, we assume no limitation on the computational power of  $f$  and  $g$ . Finally, note that the function  $g$  does not have to be a norm function, or to satisfy the triangle inequality.

We also define a particular type of heuristic that is induced by a set of “beacons”. This is a natural class of  $A^*$  heuristics that is particularly popular in the context of shortest path problems [Abraham et al., 2010].

**Definition 1.5** (Beacon heuristic). *A heuristic function  $h : V \times V \rightarrow \mathbb{R}$  is a beacon heuristic induced by a set  $B \subset V$  if  $h(s, t) = \|\pi(s) - \pi(t)\|_\infty$ , where  $\pi(s) = (dist(s, b_1), \dots, dist(s, b_{|B|}))$  (so that  $h(s, t) = \max_{b \in B} |dist(s, b) - dist(t, b)|$ ).*

Given the above definitions, we can now formally state our results (also summarized in Table 1).

## 1.2 Lower bounds results

Our first set of results provide lower bounds on the trade-off between the complexity of a heuristic and the complexity of the  $A^*$  algorithm. Specifically, we will show that there exist graphs where all shortest paths have constant complexity (i.e., consist of a constant number of hops), but on which  $A^*$  will scan a large number of vertices unless its heuristic has large complexity. We start from heuristics induced by norms. Recall that the average-case complexity is defined with respect to the set of all possible source-destination pairs.

**Theorem 1.6** (Lower bound for norm heuristics). *There exists an  $n$ -vertex unweighted graph  $G = (V, E)$  of constant diameter such that for any consistent norm heuristic induced by an  $l_p$  norm of dimension  $d$ , where  $d = o(n/\log n)$  for  $p < \infty$  or  $d = o(\log n)$  for  $p = \infty$ , the  $A^*$  algorithm scans  $\tilde{\Omega}(n)$  vertices on average.*

The proof for the above theorem appears in Section 3.1. For the special case of the  $l_\infty$  norm we show a stronger lower bound. However, it requires that the input graph is *weighted* (see Section 3.2).

Type	Settings	Comments	Result	Reference
Norm heuristics	$p < \infty$		$d = o(n/\log n) \Rightarrow \tilde{\Omega}(n)$ avg. overhead	Thm. 1.6
	$p = \infty$		$d = o(\log n) \Rightarrow \tilde{\Omega}(n)$ avg. overhead	
	$p = \infty$ weighted graphs	general graphs	$d = o(n^\alpha) \Rightarrow \tilde{\Omega}(n^{1-\alpha})$ avg. overhead	Thm. 1.7
		grid graphs ( $\alpha < 0.5$ )		Thm. 3.4
		beacon based Assumption 1.9	$d = O(n^\alpha)$ and $O(n^{1-\alpha})$ avg. overhead	Thm. 1.10
Labeling heuristics	weighted graphs	general graphs	$L = o(n^\alpha) \Rightarrow \Omega(n^{1-\alpha})$ avg. overhead	Thm. 1.8
		grid graphs ( $\alpha < 0.5$ )		Thm. 3.9
		Assumption 1.11	$L = O(n^\alpha)$ and $O(n^{1-\alpha})$ avg. overhead	Thm. 1.12

Table 1: Summary of our results. Upper bounds are colored in gray.

**Theorem 1.7** (Lower bound for  $l_\infty$  norm heuristics). *There exists a weighted  $n$ -vertex graph  $G$  such that (i) the shortest path between any pair of vertices in  $G$  consists of a constant number of hops and (ii) for any consistent norm heuristic induced by  $l_\infty$  of dimension  $d = o(n^\alpha)$ , the  $A^*$  algorithm scans  $\Omega(n^{1-\alpha})$  vertices on average.*

We now turn to heuristics induced by general labeling schemes. Our main result here is that a trade-off analogous to that in Theorem 1.7 applies even to general labeling schemes (see Section 3.3).

**Theorem 1.8** (Lower bound for labeling heuristics). *For any consistent labeling heuristic function  $h$  with length  $L = o(n^\alpha)$ , there exists an  $n$ -vertex weighted graph  $G$  such that (i) the shortest path between any pair of vertices in  $G$  consists of a constant number of hops and (ii) the  $A^*$  algorithm scans  $\Omega(n^{1-\alpha})$  vertices on average.*

For the last two theorems, we also show their variants for a graph  $G$  that is a grid graph (with weighted edges). See Theorems 3.4 and 3.9 in Appendix A for further details. This makes our instances realizable in the robotics planning problems, which is one of the main applications where heuristics have been applied Bhardwaj et al. [2017] (see Section 1.4 for more details).

### 1.3 Upper bound results

We complement the lower bounds from the previous sections with upper bounds, i.e., constructions of embedding or labeling heuristics. Our constructions, however, require certain assumptions regarding breaking ties in the  $A^*$  algorithm. Specifically, at any step of the algorithm, there could be multiple vertices with the same value of the estimate  $d(s, u) + h(u, t)$ , and breaking such ties appropriately can have significant impact on the efficiency of  $A^*$  [Asai and Fukunaga, 2016, 2017, Rathfux et al., 2019]. In our theoretical analysis we deal with tie-breaking in two ways. Our first approach is to make an assumption about the  $A^*$  algorithm. Specifically, Assumption 1.9 states that the  $A^*$  algorithm can break ties in a way that minimizes its complexity. Our second approach is to make an assumption about the input graph. Specifically, Assumption 1.11 states that the weighted graph  $G$  is such that there is a unique shortest path between any pair of vertices, and that there is a separation between the lengths of the shortest and the second shortest path. We note that the lower bounds corresponding to our upper bounds are consistent with these assumptions, as per discussions in the corresponding sections.

We start by stating the assumption about the tie-breaking behavior of the algorithm.

**Assumption 1.9** (Breaking ties). *We assume that, when there is a tie on the estimated distance lower bound (as specified in Definition 2.3) for multiple different vertices, the  $A^*$  algorithm chooses the next vertex in a way that minimizes the total running time.*

Note that in the context of Fact 2.4 stated in Preliminaries, the assumption states that  $A^*$  only scans those vertices that *must* be scanned.

Under this assumption, we show that Theorem 1.7 is almost optimal. In fact, the upper bound is based on a beacon heuristic (see Section 4.1).

**Theorem 1.10** (Beacon-based upper bound). *Under Assumption 1.9, for any weighted graph  $G = (V, E, W)$  with  $|V| = n$  vertices, there exists a beacon heuristic induced by a set  $B$  with  $|B| = n^\alpha$  with an additive overhead  $n^{1-\alpha}$ , i.e., such that  $A^*$  scans at most  $n^{1-\alpha} + \mathbb{E}_{s,t}[|P(s, t)|]$  vertices on average.*

We now consider the assumption about lack of ties in the input graph.

**Assumption 1.11** (Unique shortest path). *We assume that for any pair  $(s, t)$  of vertices in  $G$ , the difference between the length of their shortest path and second shortest path between  $s$  and  $t$  is greater than a constant  $c > 0$  (e.g.  $c = 3$ ).*

Under this assumption, we show that Theorem 1.8 is almost optimal (see Section 4.2).

**Theorem 1.12** (Labeling-based upper bound). *Under Assumption 1.11, for any weighted graph  $G = (V, E, W)$  with vertex set size  $|V| = n$ , there exists a consistent labeling heuristic of length  $L = O(n^\alpha)$  with an additive overhead  $n^{1-\alpha}$ , i.e., such that  $A^*$  scans at most  $n^{1-\alpha} + \mathbb{E}_{s,t}[|P(s, t)|]$  vertices on average.*

## 1.4 Learned A\* overview

As discussed previously, the main motivation for this study are recent works on learned A\* heuristic functions, where the learned features on which the  $h$  heuristic is based can be viewed as embedding or labeling schemes. In this section we briefly describe several approaches used in recent papers on A\* with learned heuristic functions [Yonetani et al., 2021, Bhardwaj et al., 2017, Chen et al., 2020]. As these are applied machine learning papers, our description here abstracts away many technical details, focusing on high-level ideas.

- [Bhardwaj et al., 2017] is focused on learning A\* heuristics for robot path planning. The experiments are done on graphs induced by two-dimensional environments modeled as 2D grids with obstacles. The heuristic function is represented by a feed-forward neural network with two fully connected hidden layers containing respectively 100 and 50 units with ReLu activation. The input to the network is a 17 dimensional feature vector. The features contain the descriptor of the node  $u$  (containing various features of the spatial point) as well as the descriptor of the current “state” of the neural A\* algorithm when reaching  $u$ .
- [Yonetani et al., 2021] is also focused on robot path planning in 2D grids with obstacles. It uses a neural network architecture called U-net to transform the input instance into a “guidance map”, which for each graph vertex provides an adjustment to a “standard” heuristic function based on Chebyshev distance between the given node and the destination node  $t$ . The input to U-net consists of the input graph, the source node  $s$  and destination node  $t$ . The presence of the last component technically makes it possible to map each node into a one-dimensional scalar equal to the true distance to target node  $t$ . However, given that training of the heuristic function is expensive, it is natural to aim for a heuristic that is “target independent”, i.e., trained for a particular environment, independently of the choice of the source and destination. In this way one can reuse the same heuristic trained for a particular grid for many source/destination pairs.
- [Chen et al., 2020] addresses problems in a different domain, focusing on path searching in the molecule space. The goal of the planner is to identify a series of re-actions that can lead to the synthesis of a target product, a classic problem in organic chemistry. The feature vector used as an input to the learned heuristic is based on the “Morgan fingerprint of radius 2”, which is 2048 bit long. The vector is then fed into a single-layer fully connected neural network of hidden dimension 128, which provides an estimate of the distance.

In the context of the aforementioned works, our results provide insights into the complexity of *representing* efficient heuristics for various classes of graphs. We leave other important issues, such as the complexity of *learning* those heuristics, to future work.

## 2 Preliminaries

**Notation.** We use  $dist(u, v)$  to denote the length of the shortest paths between nodes  $u$  and  $v$  in a graph. For  $(u, v) \in E$  we use  $w(u, v)$  to denote the edge length.

**Heuristic properties.** We first recall several standard definitions of heuristic functions which will be useful in our proofs.

**Definition 2.1** (Admissible heuristic function). *For a heuristic function  $h(s, t)$  which estimates  $dist(s, t)$ , it is said to be admissible if for any source and destination  $(s, t)$ , we have  $h(s, t) \leq dist(s, t)$ .*

It is well-known that all consistent heuristic functions (Definition 1.2) are admissible.

**Definition 2.2** (Sub-additive heuristic function). *For a heuristic function  $h(s, t)$  which estimates  $dist(s, t)$ , it is said to be sub-additive if for any three vertices  $(u, v, w)$ , we have  $h(u, v) + h(v, w) \geq h(u, w)$ .*

Note that consistent heuristics may not be sub-additive, but all norm heuristics (Definition 1.3) and beacon heuristics (Definition 1.5) are sub-additive.

**Definition 2.3** (Estimated distance lower bound). *Given a query pair  $dist(s, t)$ , for any vertex  $u$ , we define  $g(u)$  to be the estimated distance lower bound calculated by  $A^*$  for shortest path from  $s$  to  $t$  passing  $u$ :  $g(u) = dist(s, u) + h(u, t)$ . In particular,  $g(t) = dist(s, t)$ .*

**A\* algorithm.** Now, we describe the operation of  $A^*$  with heuristic function  $h$ , which will be the main object of our analysis in the paper. (See Algorithm 1)

---

**Algorithm 1:**  $A^*$  search

---

**Input:**  $G(V, E, W)$ , start point:  $s$ , end point:  $t$   
1 Initialize  $S = \{s\}$ ,  $d(s) = 0$ ,  $d(S \setminus s) = \infty$   
2 **while**  $t \notin S$  **do**  
3      $u \leftarrow \underset{u \in V \setminus S}{\operatorname{argmin}} \{d(u) + h(u, t)\}$   
4     **for**  $v$  s.t.  $(u, v) \in E$  **do**  
5          $d(v) = \min(d(v), d(u) + w(u, v))$   
6      $S = S \cup \{u\}$

---

**Fact 2.4** (Scan condition, see Hart et al. [1968]). *Suppose we apply  $A^*$  to calculate  $dist(s, t)$  with a consistent heuristic function  $h(u, t)$ . Then a vertex  $u$  must be scanned if and only if  $dist(s, u) + h(u, t) < dist(s, t)$  and may be scanned if and only if  $dist(s, u) + h(u, t) \leq dist(s, t)$ .*

## 3 Lower bounds

### 3.1 An average-case lower bound for $l_p$ norms

In this section, we prove Theorem 1.6.

**Theorem 1.6** (Lower bound for norm heuristics). *There exists an  $n$ -vertex unweighted graph  $G = (V, E)$  of constant diameter such that for any consistent norm heuristic induced by an  $l_p$  norm of dimension  $d$ , where  $d = o(n/\log n)$  for  $p < \infty$  or  $d = o(\log n)$  for  $p = \infty$ , the  $A^*$  algorithm scans  $\tilde{\Omega}(n)$  vertices on average.*

Before exactly defining the lower bound instance (depicted in Figure 1) and proving the theorem, we give a short overview of the ideas behind the construction and proof.

**Proof overview.** The central part of our constructed hard instance contains a star with  $n$  petals. We then add to it  $O(\log n)$  additional sets of vertices, each of size  $O(n)$ , in a way that ensures that for every pair of star petals  $(u, v)$ , there exists  $\Omega(n^2)$  vertex pairs in the additional sets whose shortest path goes through  $(u, v)$ . Given this construction, we rely on a standard “packing argument” stating that a low dimensional space cannot contain a large equilateral vertex set. Therefore, it holds that if the dimension  $d$  of the embedding is too small, then by the admissibility property of the embedding, for at least one pair of the star petals, its embedded distance is distorted. We then utilize this pair of distorted petals to show that every query  $(s, t)$  whose shortest path goes through the distorted pair has a large query overhead. Specifically, we rely on the design of non-uniqueness of the shortest paths to further amplify the penalty due to the distorted pair to  $\Omega(n)$  per each query as above, so that in total we get that there exists  $\Omega(n^2)$  pairs for which the query overhead is  $\Omega(n)$ .



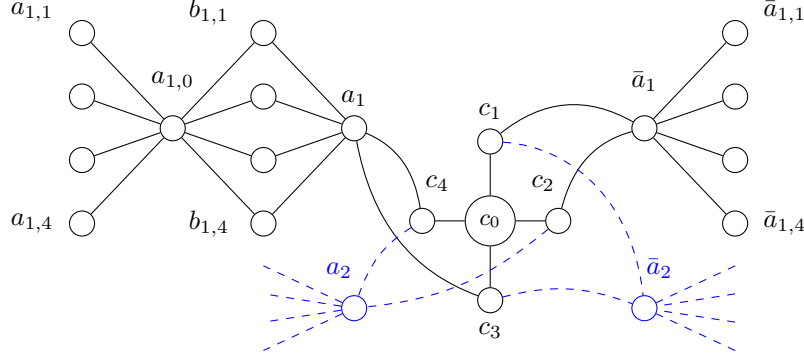


Figure 1: Average-case complexity lower bound instance for  $l_p$  norms with  $k = 2$  and  $n = 4$ .

**The lower bound instance.** We continue to formally define our lower bound instance for heuristics induced by  $l_p$  norms. Note that the lower bound instance is both sparse and unweighted. The lower bound graph contains a star with  $n$  leaves  $\{c_1, \dots, c_n\}$  connected to a center  $c_0$ . Without loss of generality, we assume  $n = 2^k$  is a power of 2. Then we have  $k$  pairs of vertices  $\{(a_i, \bar{a}_i)\}_{i=1}^k$ . For each pair  $a_i$  and  $\bar{a}_i$ , and for each  $c_j$ , we connect  $c_j$  to  $a_i$  if the  $i$ -th bit in the binary representation of  $j - 1$  is 1 and to  $\bar{a}_i$  otherwise. Then, for each  $a_i$ , we create auxiliary vertex sets  $\{a_{i,j}\}_{j=0}^n$  and  $\{b_{i,j}\}_{j=1}^n$ . We connect  $a_{i,0}$  to all  $\{a_{i,j}\}_{j=1}^n$ , and, for each  $j$ , we connect  $b_{i,j}$  to  $a_{i,0}$  and  $a_i$ . Similarly, for each  $\bar{a}_i$ , we connect it to all  $\{\bar{a}_{i,j}\}_{j=1}^n$ . Our instance has size  $|V| = \Theta(n \log n)$  and  $|E| = O(|V|)$ . See Figure 1 for an illustration with  $k = 2$  and  $n = 4$ .

**Lemma 3.1.** Suppose  $h$  is a norm heuristic function  $h(s, t) = \|\pi(s) - \pi(t)\|$ . For the instance described above with  $|V| = \Theta(n \log n)$ , as long as one pair of distinct vertices in  $\{c_i\}_{i=1}^n$ , say  $c_i$  and  $c_j$ , satisfies  $h(c_i, c_j) = \|\pi(c_i) - \pi(c_j)\| < 2$ , the  $A^*$  algorithm scans at least  $\Omega\left(\frac{n}{\log^2 n}\right)$  vertices on average.

*Proof.* Suppose  $\|\pi(c_u) - \pi(c_v)\| < 2$  and  $u \neq v$ , then there exists a digit  $p$  s.t. their binary representations differ, e.g.  $u$ 's digit is 0 and  $v$ 's is 1. Then, for the pair  $(a_p, \bar{a}_p)$  whose real distance is  $\text{dist}(a_p, \bar{a}_p) = 4$ , by sub-additivity, their embedded distance satisfies

$$\|\pi(a_p) - \pi(\bar{a}_p)\| \leq \|\pi(a_p) - \pi(c_u)\| + \|\pi(c_u) - \pi(c_v)\| + \|\pi(c_v) - \pi(\bar{a}_p)\| < 4 = \text{dist}(a_p, \bar{a}_p).$$

Now, we can check that for any query pair  $(s, t)$  with  $s \in \{a_{p,j_1}\}_{j_1=1}^n$  and  $t \in \{\bar{a}_{p,j_2}\}_{j_2=1}^n$ ,  $A^*$  must scan the whole set  $\{b_{p,j_1}\}_{j_1=1}^n$ . To see this, first note that all vertices  $u \in \{b_{p,j_1}\}_{j_1=1}^n$  lie on the shortest path between  $s$  and  $t$ , and satisfy  $h(u, t) < \text{dist}(u, t)$  (due to the sub-additivity and since  $\|\pi(a_p) - \pi(\bar{a}_p)\| < \text{dist}(a_p, \bar{a}_p)$ ). Therefore, every  $u \in \{b_{p,j_1}\}_{j_1=1}^n$  will be scanned by  $A^*$  according to Fact 2.4. To calculate the average query complexity, observe that for a random query pair  $(s, t)$ , with probability at least  $\Omega(\frac{1}{\log^2 n})$  we have that  $s \in \{a_{p,j_1}\}_{j_1=1}^n$  and  $t \in \{\bar{a}_{p,j_1}\}_{j_1=1}^n$  simultaneously hold. Once this event occurs,  $A^*$  will scan at least  $\Omega(n)$  vertices, so the average query complexity is lower bounded by  $\Omega(\frac{n}{\log^2 n})$ .  $\square$

**Definition 3.2.** In a metric space  $l_p^d$ , we say a vertex set  $X$  is equilateral if any two different vertices  $x, y \in X$  satisfy  $\|x - y\|_p = 1$ . We define  $e(l_p^d)$  to be the size of the largest equilateral vertex set in  $l_p^d$  and its inverse function  $e^{-1}(p, n)$  equals to the minimum  $d$  such that  $e(l_p^d) \geq n$ .

**Lemma 3.3.** For a metric space  $l_p^d$ , we have the following upper bounds on  $e(l_p^d)$

$$e(l_p^d) \leq \begin{cases} O(d \log d) & p = 1 & [\text{Alon and Pudlák, 2003}] \\ d + 1 & p = 2 & [\text{Guy, 1983}] \\ O(d \log d) & 2 < p < \infty & [\text{Swanepoel and Monthly, 2004}] \\ 2^d & p = \infty & [\text{Guy, 1983}] \end{cases}$$

and corresponding lower bounds for  $e^{-1}(p, n)$

$$e^{-1}(p, n) \geq \begin{cases} \Omega(\frac{n}{\log n}) & p = 1 \\ n - 1 & p = 2 \\ \Omega(\frac{n}{\log n}) & 2 < p < \infty \\ \log_2 n & p = \infty \end{cases}$$

Here our asymptotic bound is in terms of  $d$  and hides the dependence on  $p$ .

*Proof of Theorem 1.6.* We only need to verify that the assumption made in Lemma 3.1 is true. By the definition of an equilateral set (in Definition 3.2) and a consistent heuristic function (in Definition 1.2), we know that for metric space  $l_p^d$  if  $d < e^{-1}(p, n)$ , then for a set of  $n$  2-equidistant vertices  $\{c_i\}_{i=1}^n$  in the original graph, at least two of them will have embedded distance smaller than 2.  $\square$

### 3.2 A stronger lower bound for $l_\infty$ norm space

In this section, we prove a stronger lower bound for embeddings into  $l_\infty^d$  both for general graphs and for grid graphs. The first lower bound instance is a general graph with constant diameter (as in the previous section), which means that the ratio of the number of vertices scanned by  $A^*$  to the actual path length is large. Based on this instance we prove Theorem 1.7, restated below.

**Theorem 1.7** (Lower bound for  $l_\infty$  norm heuristics). *There exists a weighted  $n$ -vertex graph  $G$  such that (i) the shortest path between any pair of vertices in  $G$  consists of a constant number of hops and (ii) for any consistent norm heuristic induced by  $l_\infty$  of dimension  $d = o(n^\alpha)$ , the  $A^*$  algorithm scans  $\Omega(n^{1-\alpha})$  vertices on average.*

The second lower bound instance has super-constant diameter, but it is a grid graph, which is a more natural structure in the context of robot planning applications. This instance is described in Appendix A, and is used to prove the grid variant of Theorem 1.7.

**Theorem 3.4.** *There exists a weighted grid graph  $G$  so that any consistent norm heuristic  $h(s, t)$  induced by  $\pi : V \rightarrow \mathbb{R}^d$  with dimension  $d < o(n^\alpha)$  for  $\alpha < 0.5$  and the  $l_\infty$  norm will result in  $A^*$  average-case query complexity of at least  $\Omega(n^{1-\alpha})$ .*

Importantly, Theorem 1.7, as well as its grid analog, Theorem 3.4, hold even under Assumption 1.9, i.e., when  $A^*$  can break ties arbitrarily. This is because Lemma 3.7 below holds for *any* tie-breaking rule, i.e., it lower bounds the number of vertices that *must* be scanned. Therefore, the upper bound Theorem 1.10 is a matching upper bound. For further details on the upper bound see Section 4.1.

We start with the intuition behind the lower bound construction and proof.

**Proof idea.** The proof idea here follows the one in the previous section, where we first prove that there exist some pairs of vertices with large distortion, and then amplify the query cost penalty due to these pairs. The previous equilateral set argument works well for  $l_p$  norm with  $p < \infty$  but works poorly for the  $l_\infty$  norm. Therefore, we use a special property of  $l_\infty$  to get a tighter lower bound, specifically, that the  $l_\infty$  norm solely depends on one coordinate (with the maximum absolute value). We can observe that if the distance that our embedding “wants to memorize” is random, then it cannot remember too much information in each one dimensional space. Our construction consists of a clique of size  $O(m)$ , and the edge weights are chosen uniformly in  $[10, 11]$ . This ensures that the shortest path between any pair of vertices is the edge directly connecting them, and that the weights are randomized and independent of each other. Therefore, each dimension can only “memorize”  $O(m)$  shortest path distances, while there are  $O(m^2)$  pairs of vertices. Similarly to the previous case, the second step of amplifying the penalty is achieved by adding auxiliary sets to the main lower bound construction, so that each distorted pair appears in the shortest path for many queries  $(s, t)$ .



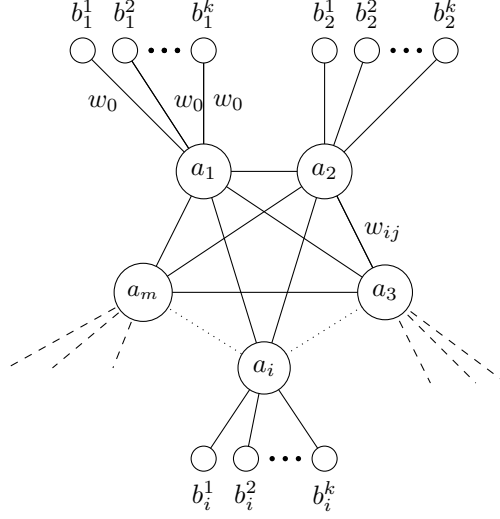


Figure 2: Average-case complexity lower bound instance for  $l_\infty$  norms.

**The lower bound instance.** Our lower bound instance (in Figure 2) consists of a clique  $\{a_i\}_{i=1}^m$  where each  $a_i$  is attached to  $k$  leaves  $\{b_{i,j}\}_{j=1}^k$ . The edge weight between  $(a_i, a_j)$  is  $w_{ij} = 10 + u_{ij}$ , where  $u_{ij}$  are i.i.d. random variables chosen from the uniform distribution over the interval  $[0, 1]$ . The edge weight between each attached leaf to its corresponding vertex in the clique is equal to  $w_0$ , specified subsequently.

**Definition 3.5** (Approximated-tie). *For a set of values  $\{w_i\}_{i=1}^n$  and an parameter  $\epsilon$ , we say there exists an approximated-tie if there exists a coefficient vector  $\{c_i\}_{i=1}^n$  where each  $c_i \in \{-4 \dots 4\}$  with at least one non-zero component and  $|\sum_{i=1}^n c_i w_i| \leq \epsilon$ .*

**Proposition 3.6.** *For weights  $\{w_{i,j}\}_{i,j \in [m]}$  generated as above, there exists an  $\epsilon > 0$  such that no approximated-tie exists in  $\{w_{i,j}\}$  with probability at least 0.99.*

Let  $\epsilon$  be as in the above proposition. We set the edge weights between the clique to the leaves to be  $w_0 = \frac{\epsilon}{16n}$ . From now on, we assume that there is no  $\epsilon$  approximated-tie in the graph. We can observe that in this graph, the shortest path between any two vertices in  $\{a_i\}$  is exactly the edge connecting them.

**Lemma 3.7.** *For a pair of vertices  $(a_i, a_j)$ , if the embedding distance between them has error larger than  $\frac{\epsilon}{2n}$ , then any query  $(b_i^p, b_j^q)$  for  $p, q \in [k]$  will always scan  $\Omega(k)$  vertices (under any tie-breaking rule.)*

*Proof.* In the setting of Theorem 1.7, we denote  $h(s, t) = \|\pi(s) - \pi(t)\|_\infty$ . If  $\|\pi(a_i) - \pi(a_j)\| \leq w_{i,j} - \frac{\epsilon}{2n}$ , we can check that for query  $\text{dist}(b_i^p, b_j^q)$  all vertices in  $\{b_i^{p'}\}_{p' \in [k]}$  will be scanned by  $A^*$ : Let  $b_i^{p'}$  be an arbitrary leaf attached to  $a_i$ .

$$\begin{aligned}
& \text{dist}(b_i^p, b_i^{p'}) + \|\pi(b_i^{p'}) - \pi(b_j^q)\| \\
& \leq \text{dist}(b_i^p, b_i^{p'}) + \|\pi(b_i^{p'}) - \pi(a_i)\| + \|\pi(a_i) - \pi(a_j)\| + \|\pi(a_j) - \pi(b_j^q)\| \\
& \leq 2w_0 + w_0 + w_{i,j} - \frac{\epsilon}{2n} + w_0 \\
& \leq w_{i,j} - \frac{\epsilon}{4n} \\
& < \text{dist}(b_i^p, b_j^q)
\end{aligned}$$

Therefore, for every  $p' \in [k]$ , by the scan condition,  $b_i^{p'}$  will be scanned. □

**Lemma 3.8.** *For the heuristic considered by Theorem 1.7, as long as  $d \leq o(m)$ , at least half of pairs  $(s, t)$  where  $s, t \in \{a_i\}, i = 1 \dots m$ , satisfy  $h(s, t) < \text{dist}(s, t) - \frac{\epsilon}{2n}$ .*

*Proof.* By the definition of  $l_\infty$  norm, if  $\|\pi(u) - \pi(v)\|_\infty \geq \text{dist}(u, v) - \frac{\epsilon}{2n}$ , then there must exist a coordinate  $i \in [d]$  where  $|\pi(u)_i - \pi(v)_i| \geq \text{dist}(u, v) - \frac{\epsilon}{2n}$ . We call such coordinate a “crucial coordinate” for the pair of vertices  $(u, v)$ .

We claim that for each coordinate  $i \in [d]$ , at most  $m - 1$  pairs of vertices use coordinate  $i$  as their crucial coordinate. To show this, we first construct an auxiliary graph with vertices  $\{a_i\}$  where we assign an edge to  $(u, v)$  if  $(u, v)$  uses the current coordinate as their “crucial coordinate”. Because the auxiliary graph has  $m$  vertices, if there are more than  $m$  edges, at least one simple cycle exists with length  $l \leq m$ . We can go along the cycle to get an approximated-tie consisting of a  $\pm 1$  weighted sum of  $w_{i,j}$ . Because  $l \cdot \frac{\epsilon}{2n} \leq \epsilon$ , such a cycle would violate Proposition 3.6.

Therefore if there are at most  $o(m)$  coordinates, there can be at most  $o(m^2)$  “crucial coordinates”, which means that at least  $\frac{m^2}{4}$  pairs are embedded with error more than  $\frac{\epsilon}{2n}$ .  $\square$

*Proof of Theorem 1.7.* We set  $m = n^\alpha$  and  $k = n^{1-\alpha}$  then  $|V| = \Theta(n)$ . Now we calculate the overall average complexity lower bound. Each pair  $(a_i, a_j)$  with  $h(a_i, a_j) < \text{dist}(a_i, a_j) - \frac{\epsilon}{2n}$  contributes  $k^2$  pairs of queries, each of which causes  $A^*$  to have query complexity  $\Omega(k)$ . Therefore the average query complexity is lower bounded by the following:

$$\Omega\left(\frac{\frac{m^2}{4} \cdot k^2 \cdot k}{n^2}\right) = \Omega\left(\frac{n^{2\alpha} \cdot n^{2(1-\alpha)} \cdot n^{1-\alpha}}{n^2}\right) = \Omega(n^{1-\alpha}).$$

$\square$

### 3.3 An average-case lower bound for labeling heuristics

In this section, we state two lower bounds for labeling heuristics. The first holds for general graphs, and the second for grid graphs. Note that labeling heuristics do not have the sub-additivity property.

First, we make some assumptions on the graphs we consider. As usual, we consider weighted undirected graphs  $G(V, E, W)$  where  $|V| = n$ . We further assume that the edge weight between any two vertices  $u$  and  $v$ ,  $w(u, v)$ , is an integer and that the sum of weights satisfies  $\sum_{u,v} w(u, v) \leq C$ , where  $C = \text{poly}(n)$ . Therefore, each quantity produced by our algorithm can be represented by  $B = O(\log n)$  bits.

We first recall the lower bound that holds for general, non grid, graphs.

**Theorem 1.8** (Lower bound for labeling heuristics). *For any consistent labeling heuristic function  $h$  with length  $L = o(n^\alpha)$ , there exists an  $n$ -vertex weighted graph  $G$  such that (i) the shortest path between any pair of vertices in  $G$  consists of a constant number of hops and (ii) the  $A^*$  algorithm scans  $\Omega(n^{1-\alpha})$  vertices on average.*

We also provide a grid graph which preserves the complexity lower bound and is common in robotics planning problem.

**Theorem 3.9.** *For any consistent labeling heuristic function  $h$  with length  $L = o(n^\alpha)$ , there exists an  $n$ -vertex weighted grid graph  $G$  such that the  $A^*$  algorithm scans  $\Omega(n^{1-\alpha})$  vertices on average.*

Importantly, both the graph used in Theorem 1.8 and the one used to prove Theorem 3.9 satisfy Assumption 1.11 regarding unique shortest paths, after an appropriate weight scaling. Therefore, Theorem 1.12 shown later in Section 4.2 is a matching upper bound result.

In this section we only prove the general theorem, and we defer the proof of Theorem 3.9 to Appendix A.2, as it follows similar ideas. We start from a short overview.

**Proof idea.** This proof inherits the idea of reducing a good embedding to a “memorizing random number” task. We utilize the hardness of the “indexing” communication complexity problem to prove that it is impossible to remember too many random numbers using a limited amount of storage.

**The lower bound instance.** The lower bound instance is depicted in Figure 3. It consists of a clique  $\{a_i\}_{i=1}^m$  and additionally, each  $a_i$  has a corresponding leaf set  $\{b_i^p\}_{p=1}^k$ . To avoid boundary cases, we assume both  $m, k \geq 10$ . Indeed, this graph has the same structure as that in Figure 2, however, we choose a different set of parameters. The edge weight between each attached leaf to its corresponding vertex in the clique is 1 and the edge weight between  $(a_i, a_j)$  is  $w_{ij} = 6 \cdot (2^b + \delta_{ij}) - 2$ , where  $\delta_{ij} \in [0, 2^b - 1]$  to be specified later. The setting of  $w_{ij}$  guarantees that the shortest distance between any two vertices  $a_i, a_j$  is due to the edge  $(a_i, a_j)$ . Finally, we define an  $m$ -tuple to be a set consisting of  $m$  vertices  $(v_1, \dots, v_m) = (b_1^{i_1}, \dots, b_m^{i_m})$  for indices  $i_j \in [k]$ . Therefore, there are  $k^m$  different  $m$ -tuples. See in Figure 3 an example to one possible  $m$ -tuple colored in blue.

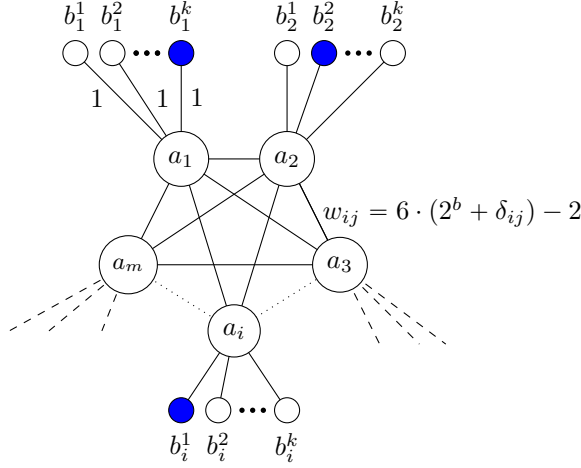


Figure 3: Average-case complexity lower bound instance for labeling.

For the ease of proof, we define the binary version of labeling heuristics which only allows for 0/1 bits and the length is multiplied by  $B$ .

**Definition 3.10** (Binary labeling heuristics). *Labeling heuristic with length  $L$  consists of two fixed deterministic functions  $f : V \rightarrow \{0, 1\}^{BL}$  and  $g : \{0, 1\}^{BL} \times \{0, 1\}^{BL} \rightarrow \{0, 1\}^B$ . The distance of two vertices  $u, v$  in the embedded space is defined to be  $g(f(u), f(v))$ .*

Interpreting strings in  $\{0, 1\}^B$  as the binary version of integers within the range  $[0, 2^B - 1]$ , the consistent definition still applies to the new definition.

We will reduce our embedding problem to the following well known indexing problem.

**Definition 3.11** (Indexing problem). *Alice gets a vector,  $x \in [0, 2^b - 1]^n$  chosen uniformly at random. Bob gets an index  $i \in [1, n]$  chosen uniformly at random. The goal is for Bob to report  $x_i$  after receiving a single message from Alice.*

We set the bit length for each number in the input of indexing problem to be  $b = \frac{B}{8}$ . Another mild condition is that  $B$  should be set such that  $2^b < n$  and  $n^6 < 2^B$ , to ensure that every quantity produced in the reduction process can still be represented in  $B$  bits.

**Theorem 3.12** (Folklore, see e.g., [Rao and Yehudayoff \[2020\]](#)). *Any one-way protocol for the indexing problem defined above requires  $\Omega(nb)$  bits of communication in order to succeed with probability  $> \frac{1}{2}$ .*

**Definition 3.13** (A bad pair). *We say that a pair of vertices  $v_i, v_j$  is a bad pair if they have large distortion, defined as  $g(f(v_i), f(v_j)) < \text{dist}(v_i, v_j) - 3$ .*

We continue to argue about the contribution of bad pairs to the average query complexity of  $A^*$ .

**Lemma 3.14.** *If there are  $\Omega(m^2)$  bad pairs in every  $m$ -tuple, the contribution of bad pairs to the query complexity of  $A^*$  (over all possible pair queries) is  $\Omega(m^2 k^3)$ , so the average complexity for  $A^*$  is at least  $\Omega(k)$ .*

*Proof.* We first argue that if a pair of vertices  $(b_i^p, b_j^q)$  is a bad pair (as defined in Definition 3.13), then for any shortest path query of the form  $(b_i^{p'}, b_j^q)$  for any  $p' \in [k]$  will scan  $b_i^p$ . By Fact 2.4, for a shortest path query  $(s, t)$ , a vertex  $u$  is always scanned by  $A^*$  if  $\text{dist}(s, u) + g(f(u), f(t)) < \text{dist}(s, t)$ . By Definition 3.13, if  $(b_i^p, b_j^q)$  is a bad pair, then  $g(f(b_i^p), f(b_j^q)) < \text{dist}(b_i^p, b_j^q) - 3 = w_{ij} - 1$ . Therefore, for any query from  $b_i^{p'}$  to  $b_j^q$  where  $b_i^{p'}$  is any other element in the set  $\{b_i^p\}_{p=1}^k$

$$\text{dist}(b_i^{p'}, b_i^p) + g(f(b_i^p), f(b_j^q)) < 2 + w_{i,j} - 1 < \text{dist}(b_i^{p'}, b_j^q).$$

implying that  $b_i^p$  will be scanned. Therefore, every bad pair contributes  $\Omega(k)$  cost to the summation of all-pair shortest path queries cost.

It remains to lower bound the number of such bad pairs at  $\Omega(k^2 m^2)$ . By the assumption that there are  $\Omega(m^2)$  bad pairs for each  $m$ -tuple, counting with repetitions, there are at least  $\Omega(m^2 \cdot k^m)$  bad pairs. Since every bad pair appears in at most  $k^{m-2}$   $m$ -tuples, it follows that there are at least  $\Omega(k^2 \cdot m^2)$  distinct bad pairs. Hence, the bad pairs contribute  $\Omega(m^2 k^3)$  to the overall query complexity of  $A^*$ . Because  $n = \Theta(m \times k)$ , the average-case complexity is at least  $\Omega(k)$ .  $\square$

**Lemma 3.15.** *Consider the graph described above and the  $A^*$  algorithm using a consistent  $L = o(m)$  labeling heuristic. If for any choice of  $\{\delta_{ij}\}$  there exists an  $m$ -tuple  $(v_1, \dots, v_m)$  (where recall that  $v_i = b_i^{p_i}$  for some  $p_i \in [k]$ ), for which at most  $o(m^2)$  number of the induced all-pair embedding distances  $\{g(f(v_i), f(v_j))\}_{i,j \in [m]}$  have large distortion,  $g(f(v_i), f(v_j)) < \text{dist}(v_i, v_j) - 3$ , then there is a one-way protocol for the indexing problem using  $o(nb)$  bits and succeeds with probability at least  $1 - o(1)$ .*

*Proof.* Suppose that for every choice of  $\{\delta_{ij}\}$ , there exists at least one  $m$ -tuple  $(v_1, \dots, v_m)$  with its induced all-pair embedding distance satisfying that for at least  $1 - o(1)$  fraction of the pairs  $v_i, v_j$ ,  $\text{dist}(v_i, v_j) - 3 \leq g(f(v_i), f(v_j)) \leq \text{dist}(v_i, v_j)$ . Then we construct the following protocol for the indexing problem. We let  $n = \binom{m}{2}$ , and we think of the index that bob receives as a tuple  $(i, j)$ , where  $i, j \in [m]$  and  $i < j$ .

Alice: given an input  $x \in [0, 2^b - 1]^{\binom{m}{2}}$ , set  $\{\delta_{ij}\} = x$  respectively. Enumerate over all the  $m$ -tuples to find the one whose embedded distance has distortion smaller than 3 for at least  $1 - o(1)$  fraction of all pairs. Note that a distortion smaller than 3 means that  $\text{dist}(a_i, a_j) - 1 \leq g(f(v_i), f(v_j)) \leq \text{dist}(a_i, a_j) + 2$ . Recall that  $\text{dist}(a_i, a_j) = w_{ij} = 6 \cdot (2^b + \delta_{ij}) - 2$ . Hence,  $\left\lceil \frac{g(f(v_i), f(v_j))}{6} \right\rceil - 2^b = \delta_{ij}$ . Alice then sends to Bob the embedding  $\{f(v_i)\}_{i=1}^m$ , which has bit length  $o(m^2 B)$ , or equivalently  $o(m^2 b)$  in terms of the indexing bit length  $b$ .

Bob: Receive  $\{f(v_i)\}$  from Alice, use them to construct  $\delta'_{ij} = \left\lceil \frac{g(f(v_i), f(v_j))}{6} \right\rceil - 2^b$  where at least  $1 - o(1)$  fraction of them satisfy  $\delta'_{ij} = \delta_{ij} = x$ . Given Bob's query  $(i, j)$ , Bob answers the corresponding term  $\delta'_{ij}$ .

Therefore, the protocol above can solve the indexing problem with probability greater than  $1 - o(1)$  and communication complexity  $o(m^2 b)$  for any input.  $\square$

We are now ready to prove Theorem 1.8.

*Proof of Theorem 1.8.* We set  $m = n^\alpha$  and  $k = n^{1-\alpha}$ , implying that the size of the lower bound graph is  $\Theta(m \cdot k) = \Theta(n)$  as required. We prove this theorem by contradiction. If there exists a consistent labeling heuristic  $h$  with length  $L = o(n^\alpha)$  such that for any choice of  $\{\delta_{ij}\}$ ,  $A^*$  with heuristic  $h$  scans  $o(n^{1-\alpha})$  vertices. Then by Lemma 3.14, for each choice of  $\{\delta_{ij}\}$  we can find a  $m$ -tuple, where there is at most  $o(1)$  fraction of bad-pairs. Finally, by Lemma 3.15, we can get an algorithm for the indexing problem, contradicting Theorem 3.12.  $\square$

## 4 Upper bounds

### 4.1 Random beacons

In this subsection, we analyze the performance of beacon heuristic under Assumption 1.9 when  $A^*$  has the ability to arbitrarily break ties. Assumption 1.9 means that for the scanning condition stated in Fact 2.4, when answering query  $\text{dist}(s, t)$ ,  $A^*$  will only scan vertex  $u$  that *must* be scanned, i.e., if either  $u \in P(s, t)$  or  $\text{dist}(s, u) + h(u, t) < \text{dist}(s, t)$ .

We can verify that the random beacon embedding is consistent according to Definition 1.2.

**Theorem 1.10** (Beacon-based upper bound). *Under Assumption 1.9, for any weighted graph  $G = (V, E, W)$  with  $|V| = n$  vertices, there exists a beacon heuristic induced by a set  $B$  with  $|B| = n^\alpha$  with an additive overhead  $n^{1-\alpha}$ , i.e., such that  $A^*$  scans at most  $n^{1-\alpha} + \mathbb{E}_{s,t}[|P(s, t)|]$  vertices on average.*

*Proof.* First, we uniformly at random select the set  $B$  from the vertex set  $V$ . By Assumption 1.9,  $A^*$  answering query  $\text{dist}(s, t)$  will visit a vertex  $u \notin P(s, t)$  if  $\text{dist}(s, u) + h(u, t) < \text{dist}(s, t)$ , or equivalently  $\|\pi(u) - \pi(t)\| < \text{dist}(s, t) - \text{dist}(s, u)$ . By the definition of  $\|\pi(u) - \pi(t)\| = \max_{v \in B} |\text{dist}(u, v) - \text{dist}(t, v)|$  in the random beacon embedding,  $u$  will satisfy the inequality above only if  $s$  can give a better distance estimation for  $\text{dist}(u, t)$  than all beacons in  $B$ . This happens with probability at most  $\frac{1}{|B|+1}$  over a random selection of  $B \cup \{s\}$  for a fixed pair  $u, t$ . In expectation, for any pair  $(u, t)$ , there are at most  $\frac{n}{|B|}$  such vertices  $s$ . Therefore, taking a summation over all pairs of  $(u, t)$  yields that the expected number of extra vertices scanned is at most  $\frac{n^2 \cdot n/|B|}{n^2} = \frac{n}{|B|}$ . Finally, we choose  $|B| = n^\alpha$  to get the desired result.  $\square$

### 4.2 A labeling heuristic

In this section we present a labeling heuristic with parameters as in Theorem 1.12, assuming Assumption 1.11. First, we observe that this assumption holds for “generic” graphs, i.e., holds with high probability assuming that the edge weights are generated randomly. We defer the proof to Appendix B.

**Proposition 4.1.** *For any weighted graph  $G = (V, E, W)$  with  $|V| = n$  with edge weight independently generated from a discrete distribution whose probability mass at each point is upper bounded by  $\frac{1}{n^5}$ , Assumption 1.11 holds with probability at least  $1 - \frac{1}{n}$ .*

Since it is easy to re-scale the edge weights, we set the constant in Assumption 1.11 to be 3 for the ease of later proofs. For convenience, we restate the main theorem shown in this section.

**Theorem 1.12** (Labeling-based upper bound). *Under Assumption 1.11, for any weighted graph  $G = (V, E, W)$  with vertex set size  $|V| = n$ , there exists a consistent labeling heuristic of length  $L = O(n^\alpha)$  with an additive overhead  $n^{1-\alpha}$ , i.e., such that  $A^*$  scans at most  $n^{1-\alpha} + \mathbb{E}_{s,t}[|P(s, t)|]$  vertices on average.*

**Proof idea.** Since a “naive” beacon embedding produces ties, our idea for the proof is to slightly increase the embedded distance so that potential ties are avoided by adding more information to the embedding. This preserves the consistency of the heuristic, except in the case where the original embedded distance is already tight. The latter case only happens when one of the three vertices  $(s, u$  or  $t)$  lies on the shortest path between the other two. Under the unique shortest path assumption (Assumption 1.11), this case is easy to detect if we encode the Euler tour of the shortest path tree rooted at each beacon in the “extra” bits of the labels. At the same time, adding this information increases the label length by only a negligible factor.

In order to prove the theorem, we first extend the definition of the beacon heuristic from the previous section.

**Definition 4.2** (Beacon heuristic with tie breaking). *For a beacon set  $B$ , we define  $\pi$  to be a concatenation of two  $d$  dimensional embeddings  $\pi^0$  and  $\pi^1$  where  $\pi_i^0(s) = \text{dist}(s, b_i)$  is the same as beacon-based embedding*





$w(u, v) + \text{dist}(v, t)$  which is greater than  $\text{dist}(u, t)$  and therefore greater than  $h(u, t)$  by admissibility shown above.  $\square$

*Proof of Theorem 1.12.* The analysis extends on the proof of Theorem 1.10 by using random beacon-based embedding which is  $\pi^0$  in Definition 4.2.

Considering a pair of query  $\text{dist}(s, t)$ , for those vertices  $u$  satisfying  $\text{dist}(s, u) + \|\pi^0(u) - \pi^0(t)\|_\infty < \text{dist}(s, t)$ , they are inevitably scanned, and by invoking Theorem 1.10 with  $O(n^\alpha)$  dimension, the average number of such vertices  $u$  is upper bounded by  $O(n^{1-\alpha})$ . Also, for those vertices  $u$  satisfying  $\text{dist}(s, u) + \|\pi(u) - \pi(t)\|_\infty > \text{dist}(s, t)$ , they will not bother our  $A^*$  because  $h(s, v) \geq \|\pi^0(s) - \pi^0(t)\|_\infty$  is a strictly tighter distance estimation than  $\pi^0$ . Now we only care about those  $u$  satisfying  $\text{dist}(s, u) + \|\pi(u) - \pi(t)\|_\infty = \text{dist}(s, t)$  and there are two cases: If  $\|\pi(u) - \pi(t)\|_\infty = \text{dist}(u, t)$ , we have  $\text{dist}(s, u) + \text{dist}(u, t) = \text{dist}(s, t)$ , so  $u$  lies on  $P(s, t)$  and should be scanned by  $A^*$ . Otherwise  $\|\pi(u) - \pi(t)\|_\infty < \text{dist}(u, t)$  and  $h(u, t) = \|\pi(u) - \pi(t)\|_\infty + 2$ , breaking the tie, so  $A^*$  will not scan such a vertex  $u$ . Therefore, all previous ties impose no extra scanning complexity for  $A^*$ .  $\square$

## References

- Ittai Abraham, Amos Fiat, Andrew V Goldberg, and Renato F Werneck. Highway dimension, shortest paths, and provably efficient algorithms. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 782–793. SIAM, 2010.
- Noga Alon and Pavel Pudlák. Equilateral sets in  $l_p^n$ . *Geometric & Functional Analysis GAFA*, 13(3):467–482, 2003.
- Masataro Asai and Alex Fukunaga. Tiebreaking strategies for  $a^*$  search: How to explore the final frontier. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.
- Masataro Asai and Alex Fukunaga. Tie-breaking strategies for cost-optimal best first search. *Journal of Artificial Intelligence Research*, 58:67–121, 2017.
- Mohak Bhardwaj, Sanjiban Choudhury, and Sebastian Scherer. Learning heuristic search via imitation. In *Conference on Robot Learning*, pages 271–280. PMLR, 2017.
- Binghong Chen, Chengtao Li, Hanjun Dai, and Le Song. Retro\*: learning retrosynthetic planning with neural guided  $a^*$  search. In *International Conference on Machine Learning*, pages 1608–1616. PMLR, 2020.
- Xiao Cui and Hao Shi.  $A^*$ -based pathfinding in modern computer games. *International Journal of Computer Science and Network Security*, 11(1):125–130, 2011.
- Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. *Journal of Algorithms*, 53(1):85–112, 2004.
- Andrew V Goldberg and Chris Harrelson. Computing the shortest path:  $A^*$  search meets graph theory. In *SODA*, volume 5, pages 156–165, 2005.
- Richard K. Guy. An olla-podrida of open problems, often oddly posed. *The American Mathematical Monthly*, 90(3):196–200, 1983. ISSN 00029890, 19300972. URL <http://www.jstor.org/stable/2975549>.
- Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. doi: 10.1109/TSSC.1968.300136.
- Assaf Naor. Metric dimension reduction: a snapshot of the ribe program. In *Proceedings of the International Congress of Mathematicians: Rio de Janeiro 2018*, pages 759–837. World Scientific, 2018.

Anup Rao and Amir Yehudayoff. *Communication Complexity: and Applications*. Cambridge University Press, 2020.

Thomas Rathfux, Hermann Kaindl, Ralph Hoch, and Franz Lukasch. Efficiently finding optimal solutions to easy problems in design space exploration: A\* tie-breaking. In *Proceedings of the 14th International Conference on Software Technologies*, ICSOFT 2019, page 595–604, Setubal, PRT, 2019. SCITEPRESS - Science and Technology Publications, Lda. ISBN 9789897583797. doi: 10.5220/0008119405950604. URL <https://doi.org/10.5220/0008119405950604>.

Konrad J. Swanepoel and Amer Math Monthly. A problem of kusner on equilateral sets. *Archiv der Mathematik*, 2004.

Ryo Yonetani, Tatsunori Taniai, Mohammadamin Barekatain, Mai Nishimura, and Asako Kanezaki. Path planning using neural A\* search. In *International Conference on Machine Learning*, 2021.

## A Lower bounds for grid graphs

The main technical difficulty for the grid version lower bounds is that we are no longer allowed to construct a clique as in the proofs of the general theorems. Hence, the independence property of each shortest path distance is lost. Fortunately, with a careful choice of parameters, we can restrict the shortest path to follow a certain trajectory, and then we can use similar ideas to the ones for the general case lower bounds.

### A.1 Stronger lower bound for $l_\infty$ norm on a grid

In this section we modify our lower bound instance so that is based on a 2D grid. This makes it more similar to the instances in the path planning applications discussed in the introduction. Instead of drawing the instance in the form of vertices connected by weighted edges, it will be more natural to associate vertices with grid nodes (cells), and put weights on vertices. The edges from each vertex/cell lead to up to four of its neighbors in the grid, and are unweighted.

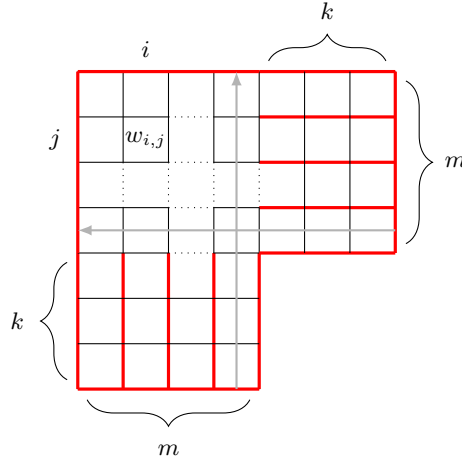


Figure 5: Grid lower bound instance.

Our instance (call it  $G$ ) is depicted in Figure 5. It consists of a concatenation of 3 rectangles. The upper-left rectangle  $R$  is of dimension  $m \times m$  and the two rectangles attached to its right ( $R_r$ ) and lower ( $R_l$ ) sides are of dimensions  $m \times k$  and  $k \times m$ , where  $m$  and  $k$  are defined later. For the simplicity of notation, we label the lower right node of  $R$  with coordinates  $(0, 0)$  and all other nodes are labeled with coordinates  $(x, y)$ , where the x-axis is horizontal and pointing to the *left*, while the y-axis is vertical and pointing to the *top*. We denote the node with coordinates  $(x, y)$  by  $v_{x,y}$ , and denote its weight by  $w_{x,y}$ ; the values of the weights will be defined later. Finally, the edges corresponding to the red lines in the picture (interpreted as obstacles) are removed. We use  $n = |V|$  to denote the size of the graph. This instance has size  $|V| = n = m^2 + 2mk$ . We set the weights  $w_{i,j}$  as follows:

$$w_{x,y} = \begin{cases} (n - x) \cdot n^3 + u_{x,y} & x, y \geq 0 \\ \epsilon_w & x \text{ or } y < 0 \end{cases}$$

where  $u_{x,y}$  are i.i.d. random variables chosen from the uniform distribution over the interval  $[0, 1]$  and  $\epsilon_w$  is a small number to be specified later.

Recall we aim to prove Theorem 3.4.

**Theorem 3.4.** *There exists a weighted grid graph  $G$  so that any consistent norm heuristic  $h(s, t)$  induced by  $\pi : V \rightarrow \mathbb{R}^d$  with dimension  $d < o(n^\alpha)$  for  $\alpha < 0.5$  and the  $l_\infty$  norm will result in  $A^*$  average-case query complexity of at least  $\Omega(n^{1-\alpha})$ .*

**Proposition A.1.** For weights  $\{w_{x,y}\}$  generated from the description above, we can find a constant  $\epsilon$  where no approximated-tie exists in  $\{w_{x,y}\}$  with probability at least 0.99.

Now, we set the parameter  $\epsilon_w = \frac{\epsilon}{8mk}$  where  $\epsilon$  is defined by Proposition A.1.

**Lemma A.2.** Consider an auxiliary graph with vertices  $V' = \{a_i\}_{i=0}^{m-1} \cup \{b_i\}_{i=0}^{m-1}$  and edges  $(a_i, b_j)$  with weights equal to the distance  $\text{dist}(v_{i,0}, v_{0,j})$  in the original graph. Then, for any simple cycle  $C$  in this auxiliary graph, any  $\pm 1$  weighted sum of the weights of the nodes in the cycle has absolute value larger than  $\epsilon$ .

*Proof.* The proof is similar to the proof of Lemma 3.8. First, from the choice of the weights  $w_{x,y}$ , we observe that the shortest path from  $v_{i,0}$  to  $v_{0,j}$  must first go straight up to  $v_{i,j}$  and then go straight right to  $v_{0,j}$ . Let  $\sum c_{i,j} \cdot w_{i,j}$  be the  $\pm 1$  weighted sum of the weights in the cycle. To apply Definition 3.5 and Proposition A.1, we need to show that  $|c_{i,j}| \leq 4$  and  $\sum |c_{i,j}| > 0$ . For the first condition, if  $w_{i,j}$  appears on the shortest path between  $v_{i',0}$  to  $v_{0,j'}$  and  $(v_{i',0}, v_{0,j'})$  is an edge in  $C$ , then either  $i = i'$  or  $j = j'$ . There are at most 4 such edges in  $C$ , so  $|c_{i,j}| \leq 4$ . For the second condition, we only need to show that there exists at least one  $|c_{i,j}| > 0$ . We choose  $v_{i,0}$  with the maximal  $i$  appearing in  $C$  and then  $v_{0,j}$  which is adjacent to the  $v_{i,0}$  on some occurrence of  $v_{i,0}$  and has the maximal index  $j$ . For this edge  $(a_i, b_j)$  in the cycle,  $v_{i,j}$  is passed through only once and has  $c_{i,j} = \pm 1$ .  $\square$

**Lemma A.3.** For any pair of vertices  $(v_{i,0}, v_{0,j})$  ( $i, j \in [0, m-1]$ ), if their embedded distance has error larger than  $\frac{\epsilon}{2m}$ , then for any query  $(v_{i,-p}, v_{-q,j})$ , ( $p, q \in [1, k]$ ) the algorithm will scan  $\Omega(k)$  vertices.

*Proof.* In the setting of Theorem 3.4, we denote  $h(s, t) = \|\pi(s) - \pi(t)\|_\infty$ . If  $\|\pi(v_{i,0}) - \pi(v_{0,j})\| \leq \text{dist}(v_{i,0}, v_{0,j}) - \frac{\epsilon}{2m}$ , we can check that for query  $\text{dist}(v_{i,-p}, v_{-q,j})$  with  $(p, q \in [1, k])$ , all vertices that lie below  $v_{i,0}$  or to the right of  $v_{0,j}$  will be scanned by  $A^*$ . We verify this for one such vertex  $v_{i,-p'} (p' > 0)$  here:

$$\begin{aligned} & \text{dist}(v_{i,-p}, v_{i,-p'}) + \|\pi(v_{i,-p'}) - \pi(v_{-q,j})\| \\ & \leq \text{dist}(v_{i,-p}, v_{i,-p'}) + \|\pi(v_{i,-p'}) - \pi(v_{i,0})\| + \|\pi(v_{i,0}) - \pi(v_{0,j})\| + \|\pi(v_{0,j}) - \pi(v_{-q,j})\| \\ & \leq k\epsilon_w + k\epsilon_w + \text{dist}(v_{i,0}, v_{0,j}) - \frac{\epsilon}{2m} + k\epsilon_w \\ & = \text{dist}(v_{i,0}, v_{0,j}) - \frac{\epsilon}{8m} \\ & < \text{dist}(v_{i,0}, v_{0,j}) < \text{dist}(v_{i,-p}, v_{-q,j}) \end{aligned}$$

The last inequality follows because we have obstacles (depicted in red) between different rows and columns in  $R_r$  and  $R_l$ , so the shortest path from  $v_{i,-p}$  to any  $v_{j,0}$  must pass  $v_{i,0}$ . A similar statement holds for  $v_{-q,j}$ .  $\square$

**Lemma A.4.** For the consistent  $l_\infty$  norm heuristic considered by Theorem 3.4, as long as  $d \leq o(m)$ , at least half of the pairs from  $(s, t) \in \{v_{i,0}\}_{i=0}^{m-1} \times \{v_{0,j}\}_{j=0}^{m-1}$  satisfies  $h(s, t) < \text{dist}(s, t) - \frac{\epsilon}{2m}$ .

*Proof.* By the definition of  $l_\infty$  norm, if  $\|\pi(u) - \pi(v)\|_\infty \geq \text{dist}(u, v) - \frac{\epsilon}{2m}$ , then there must exist a coordinate  $i \in [d]$  where  $|\pi(u)_i - \pi(v)_i| \geq \text{dist}(u, v) - \frac{\epsilon}{2m}$ . We call such coordinate a “crucial coordinate” for the pair of vertices  $(u, v)$ .

We claim that for one coordinate  $i$ , at most  $2m - 1$  pairs of vertices uses coordinate  $i$  as their crucial coordinate. The reason is that if we construct an auxiliary graph where we assign an edge to  $(u, v)$  if  $(u, v)$  uses the current coordinate as their “crucial coordinate”, the auxiliary graphs have  $2m$  vertices. If there are more than  $2m$  edges, at least one simple cycle exists with length  $l \leq 2m$ . Because  $l \cdot \frac{\epsilon}{2m} \leq \epsilon$ , such a cycle will violate Lemma A.2.

Therefore if there are at most  $o(m)$  coordinates, there can be at most  $o(m^2)$  “crucial coordinates”, which means that at least  $\frac{m^2}{2}$  pairs are embedded with error more than  $\frac{\epsilon}{2m}$ .  $\square$

*Proof of Theorem 3.4.* We set  $m = n^\alpha$  and  $k = n^{1-\alpha}$  then  $|V| = \Theta(n)$  when  $\alpha \leq 0.5$ .

Now we calculate the overall average complexity lower bound. Each pair  $(v_{i,0}, v_{0,j})$  with  $h(v_{i,0}, v_{0,j}) < \text{dist}(v_{i,0}, v_{0,j}) - \frac{\epsilon}{2m}$  will produce  $k^2$  pairs of queries with query complexity  $\Omega(k)$ , so the average query complexity is lower bounded by the following:

$$\Omega\left(\frac{\frac{m^2}{2} \cdot k^2 \cdot k}{n^2}\right) = \Omega\left(\frac{n^{2\alpha} \cdot n^{2(1-\alpha)} \cdot n^{1-\alpha}}{n^2}\right) = \Omega(n^{1-\alpha})$$

□

## A.2 A lower bound for labeling heuristics on a grid

In this section we prove Theorem 3.9, which we restate here for the sake of convenience.

**Theorem 3.9.** *For any consistent labeling heuristic function  $h$  with length  $L = o(n^\alpha)$ , there exists an  $n$ -vertex weighted grid graph  $G$  such that the  $A^*$  algorithm scans  $\Omega(n^{1-\alpha})$  vertices on average.*

**The lower bound instance.** The hard instance we use in this section has the same structure as that in Figure 5, but with a different setting of parameters. We set the weights as follows:

$$w_{x,y} = \begin{cases} (n-x) \cdot n^4 + \delta_{x,y} \cdot n - 2k & x, y \geq 0 \\ 1 & x \text{ or } y < 0 \end{cases} \quad (\text{A.1})$$

where  $\delta_{x,y} \in [-n^2, n^2]$  to be specified later. Additionally, we define  $\Delta(i, j)$  to be the sum of  $\delta_{i,j}$  along the shortest path from  $v_{i,0}$  to  $v_{0,j}$ .

**Lemma A.5.** *For any choice of  $\delta_{x,y}$  in Equation A.1, the shortest path of any pair  $(v_{i,0}, v_{0,j})$  is first going up from  $v_{i,0}$  to  $v_{i,j}$ , and then going right to  $v_{j,0}$  (recall Figure 5).*

*Proof.* Observe that for any possible  $x$  and  $\delta_{x,y}$  values, the first term in the weight  $w_{x,y}$  is larger than the summation of all grids' second terms (since  $x < m$  and  $\delta_{x,y} \in [-n^2, n^2]$ ). Hence, any shortest path must minimize the number of visited cells. Furthermore, the first term  $(n-x) \cdot n^4$  only depends on the  $x$ -axis and is smaller when  $x$  is large, implying that the shortest path should first go up and then go right. □

**Lemma A.6.** *For any vector  $x \in [0, 2^b - 1]^{m^2}$ , there exists a choice of  $\{\delta_{i,j}\}$  satisfying  $|\delta_{i,j}| \leq n^2$  for any  $i, j$  and  $\Delta(i, j) = x_{i \cdot m + j}$ .*

*Proof.* We first construct those  $\{\delta_{i,j}\}$  with  $i = 0$  or  $j = 0$ . We set  $\delta_{0,0} = x_0$ ,  $\delta_{i,0} = x_{i \cdot m} - x_{(i-1) \cdot m}$ , and  $\delta_{0,j} = x_j - x_{j-1}$ . Then, we iterate  $i, j$  in the order of  $v_{1,1}, \dots, v_{1,m-1}, v_{2,1}, \dots, v_{m-1,m-1}$  and set  $\delta_{i,j} = \delta_{i-1,j-1} + (x_{i \cdot m + j} + x_{(i-1) \cdot m + j - 1}) - (x_{i \cdot m + j - 1} + x_{(i-1) \cdot m + j})$ . It holds that  $|\delta_{i,j}| \leq 2m \cdot \max_{i,j} |x_{i,j}| \leq n^2$ , and it can be verified that for every  $i, j$ ,  $\Delta(i, j) = x_{i \cdot m + j}$ . □

**Definition A.7** ( $2m$ -tuple). *We define a  $2m$ -tuple to be a set consisting of  $2m$  vertices  $(a_0, \dots, a_{m-1}, b_0, \dots, b_{m-1})$  where  $a_i \in \{v_{i,-p}\}_{p=1}^k$  and similarly  $b_i \in \{v_{-q,i}\}_{q=1}^k$ . Therefore, there are  $k^{2m}$  different  $2m$ -tuple for our graph.*

**Definition A.8** (A bad pair). *We refer to a pair of vertices  $(a_i, b_j)$  with distortion larger than  $2k$  as a bad pair.*

**Proposition A.9.** *If a pair of vertices  $(v_{i,-p}, v_{-q,j})$  where  $p, q > 0$  is a bad pair as defined above, then for any shortest pair query  $(v_{i,-p'}, v_{-q,j})$  ( $p' > 0$ )  $A^*$  will scan  $v_{i,-p}$ . Therefore, one bad pair contribute  $\Omega(k)$  cost to the summation of all-pair shortest path queries cost.*

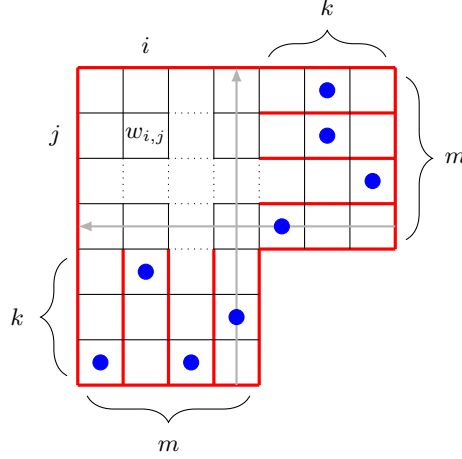


Figure 6: The lower bound instance, with an example of one possible  $2m$ -tuple in blue.

*Proof.* By Fact 2.4, for a pair of shortest path query  $(s, t)$ , a vertex  $u$  must be scanned by  $A^*$  if  $\text{dist}(s, u) + g(f(u), f(t)) < \text{dist}(s, t)$ . If  $(v_{i,-p}, v_{-q,j})$  is a “bad” pair, it means that  $g(f(v_{i,-p}), f(v_{-q,j})) < \text{dist}(v_{i,-p}, v_{-q,j}) - 2k$ . Then for any query from  $v_{i,-p'}$  to  $v_{-q,j}$ , we have

$$\text{dist}(v_{i,-p'}, v_{i,-p}) + g(f(v_{i,-p}), f(v_{-q,j})) < k + \text{dist}(v_{i,p}, v_{-q,j}) - 2k < \text{dist}(v_{i,-p'}, v_{-q,j})$$

Therefore  $v_{i,-p}$  will be scanned by at least  $k$  queries.  $\square$

Next, we lower bound the number of such “bad” pairs at  $\Omega(n^2)$ .

**Lemma A.10.** *If there are  $\Omega(m^2)$  bad pairs in every  $2m$ -tuple, then the total number of “bad pairs” is lower bounded by  $\Omega(k^2 m^2)$ .*

The proof of this lemma is similar to Lemma 3.14, so we omit it here.

**Lemma A.11.** *Consider the graph described above and the  $A^*$  algorithm using a consistent  $L = o(m)$  labeling heuristic. If for any choice of  $\{\delta_{i,j}\}$  there exists a  $2m$ -tuple  $(a_0, \dots, a_{m-1}, b_0, \dots, b_{m-1})$ , at most  $o(m^2)$  pairs of the induced all-pair embedding distances  $\{g(f(a_i), f(b_j))\}_{i,j \in [m]}$  have large distortion,  $g(f(a_i), f(b_j)) < \text{dist}(a_i, b_j) - 2k$ , then there is an one-way protocol for indexing problem using  $o(nb)$  bits and succeeds with probability at least  $1 - o(1)$ .*

*Proof.* Suppose for every choice of  $\{\delta_{i,j}\}$ , there exists at least one  $2m$ -tuple with their induced all-pair embedding distances satisfying that for at least  $1 - o(1)$  fraction of the pairs,  $\text{dist}(a_i, b_j) - 2k \leq g(f(a_i), f(b_j)) \leq \text{dist}(a_i, b_j)$ . Then we construct the following protocol for the indexing problem. We let  $n = m^2$  and we think of the index that Bob receives as a tuple  $(i, j)$  where  $i, j \in [m]$ .

Alice: given an input  $x \in [0, 2^b - 1]^{m^2}$ , invoke Lemma A.6 to produce  $\{\delta_{i,j}\}$  s.t.  $\forall (i, j), \Delta_{i,j} = x_{i \cdot m + j}$ . Enumerate over all the  $2m$ -tuples to find the one such that for at least  $1 - o(1)$  fraction of its pairs, the embedded distance distortion is smaller than  $2k$ . Note that smaller than  $2k$  distortion means that  $\text{dist}(v_{i,0}, v_{0,j}) - 2k \leq g(f(a_i), f(b_j)) \leq \text{dist}(v_{i,0}, v_{0,j}) + 2k$  so that  $\left\lceil \frac{g(f(a_i), f(b_j))}{n} \right\rceil - (n - i) \cdot n^3 = \Delta_{i,j}$ . Alice sends to Bob  $\{f(a_i)\}$  and  $\{f(b_j)\}$ , which has bit length  $o(m^2 b)$ .

Bob: Receive  $\{f(a_i)\}$  and  $\{f(b_j)\}$  from Alice, and use them to construct  $\Delta'_{i,j} = \left\lceil \frac{g(f(a_i), f(b_j))}{n} \right\rceil - (n - i) \cdot n^3$ , where at least  $1 - o(1)$  fraction of them satisfy  $\Delta'_{i,j} = x_{i,j}$ . Given Bob’s query  $(i, j)$ , Bob answers the corresponding term  $\Delta'_{i,j}$ .

Therefore, the protocol above can solve the indexing problem with probability greater than  $1 - o(1)$  and communication complexity  $o(nb)$  for any input.  $\square$



*Proof of Theorem 3.9.* We set  $m = n^\alpha$  and  $k = n^{1-\alpha}$  with  $(\alpha \leq 0.5)$  implying that the size of the graph is still  $\Theta(n)$ . We prove this theorem by contradiction. If there exists a consistent labeling heuristic  $h$  with length  $L = o(n^\alpha)$  such that for any choice of  $\{\delta_{ij}\}$ ,  $A^*$  with heuristic  $h$  scans  $o(n^{1-\alpha})$  vertices. Then by Proposition A.9 and Lemma A.10, for each choice of  $\{\delta_{ij}\}$  we can find a  $2m$ -tuple, there are at most  $o(m^2)$  bad-pairs. Finally, by Lemma A.11, we can get an algorithm for indexing problem contradicting Theorem 3.12.  $\square$

## B Proof of Proposition 4.1

*Proof.* We define a boolean variable  $tie(s, t)$  which equals 1 if and only if there exists two equal length shortest path from  $s$  to  $t$ ,  $N(s)$  to denote the set of neighbors of vertex  $s$ ,  $w(u, v)$  to be the edge weight for edge  $(u, v)$ .

$$\begin{aligned} & Pr[\cup_{s,t \in V} \mathbb{I}[tie(s, t)]] \\ & \leq \sum_{s,t} \sum_{u,v \in N(s) \wedge u \neq v} Pr[w(s, u) + dist(u, t) = w(s, v) + dist(v, t)] \\ & \leq n^4 Pr[w(s, u) + dist(u, t) = w(s, v) + dist(v, t)] \end{aligned}$$

In the following, we prove that for any fixed  $t, u, v$  where  $u, v \in N(s)$  and  $u \neq v$ , the expression above  $Pr[w(s, u) + dist(u, t) = w(s, v) + dist(v, t)]$  is bounded by  $n^{-5}$ . We use  $d_{s,u}(\cdot, \cdot)$  to be the shortest distance using all edges in  $E \setminus (s, u)$ , and  $d(\cdot, \cdot)$  to be the shortest distance using all edges in  $E$  (an abbreviation for  $dist(\cdot, \cdot)$ ). Then, the probability of the existence of non-unique shortest path can be upper bounded as the following:

$$\begin{aligned} & Pr[w(s, u) + d(u, t) = w(s, v) + d(v, t)] \\ & = Pr[w(s, u) = w(s, v) + d(v, t) - d(u, t)] \\ & = Pr[d(v, t) = d_{s,u}(v, t) \wedge d(u, t) = d_{s,u}(u, t) \wedge w(s, u) = w(s, v) + d(v, t) - d(u, t)] \\ & + \underbrace{Pr[(d(v, t) \neq d_{s,u}(v, t) \vee d(u, t) \neq d_{s,u}(u, t)) \wedge w(s, u) = w(s, v) + d(v, t) - d(u, t)]}_{\text{this case is impossible}} \end{aligned} \quad (B.1)$$

$$\begin{aligned} & \leq Pr[w(s, u) = w(s, v) + d_{s,u}(v, t) - d_{s,u}(u, t)] \\ & \leq \max_C Pr[w(s, u) = C] \\ & \leq n^{-5} \end{aligned} \quad (B.2)$$

On line B.1, we expand the probability basing on whether the distance function changes after adding edge  $(s, u)$ . We can observe that as long as the addition of edge  $(s, u)$  influences the shortest path  $P(v, t)$  or  $P(u, t)$ , the later equality cannot hold. In line B.2,  $w(s, u)$  is independent of all the other quantities. Therefore, we prove that the probability of violation of Assumption 1.11 is upper bounded by  $\frac{1}{n}$ .

Because our edge weight are drawn from a discrete distribution, unique shortest path is enough to show the existence of a constant margin as required in Assumption 1.11.  $\square$