Theoretical Computer Science 553 (2014) 18–26

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Efficient algorithms for network localization using cores of underlying graphs

Meng Li [a], Yota Otachi [b,*], Takeshi Tokuyama [c]

[a] Department of Computer Science, Rutgers, 110 Frelinghuysen Road, Piscataway, NJ 08854-8019, USA
[b] School of Information Science, Japan Advanced Institute of Science and Technology, Asahidai 1-1, Nomi, Ishikawa 923-1292, Japan
[c] Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan

ARTICLE INFO

Available online 22 February 2014

Keywords:
Network localization
Point set reconstruction
Weighted graph embedding
Graph turnpike problem
Chordal graph
Connected dominating set

ABSTRACT

Network localization is important for networks with no prefixed positions of network nodes such as sensor networks. We are given a subset of the set of $\binom{n}{2}$ pairwise distances among n sensors in some Euclidean space. We want to determine the positions of each sensor from the (partial) distance information. The input can be seen as an edge weighted graph. In this paper, we present some efficient algorithms that solve this problem using the structures of input graphs, which we call their *cores*. For instance, we present a polynomial-time algorithm solving the network localization problem for graphs with connected dominating sets of bounded size. This algorithm allows us to have fixed-parameter tractable algorithms for some restricted instances such as graphs with connected vertex covers of bounded size.

1. Introduction

1.1. Background and formulation

Nowadays sensor networks are used for many important practical applications such as monitoring environmental data (see e.g. [9,26]). Since the nodes in a sensor network do not have physical access to each other, sometimes we should construct it without prefixed positions of the nodes even if it is not a dynamic ad-hoc network; that is, the nodes are not moving. For example, assume that we want to monitor some contaminated environment. It is not possible to put a sensor node manually at a prefixed position since the area is contaminated. Thus we use some flying devices like unmanned helicopters to drop sensor nodes from high altitude. After that we can collect data by crawling the area by the same flying device. Using unmanned aerial vehicles has become a common technique in practical sensor networking [6]. To analyze the contaminated area in detail, it is useful to have spatial data of the nodes. With spatial information, we can know which area is contaminated and which area is not. The problem to determine the positions of each node in network is the *network localization problem* [2]. Equipping each node with a GPS (Global Positioning System) device might be an answer. However, it would be too expensive and impractical if the number of nodes is large. In some settings, information gathering devices may have GPS devices. However, here we consider the following setting that works without any GPS devices:

* Corresponding author.
E-mail addresses: ml910@cs.rutgers.edu (M. Li), otachi@jaist.ac.jp (Y. Otachi), tokuyama@dais.is.tohoku.ac.jp (T. Tokuyama).

http://dx.doi.org/10.1016/j.tcs.2014.02.020
0304-3975/

- each node can communicate with *some* other nodes;
- if two nodes communicate, they can measure the distance between them;
- the central device, e.g. a helicopter, collects the distance information with IDs.

The localization problem of this setting is formalized by using graphs as follows.

Problem: Weighted Graph Embeddability in d-space (WGEd)
Instance: A graph G with nonnegative weights $w_e \geqslant 0$ on each edge $e \in E(G)$.
Question: Is there a mapping $f: V(G) \to \mathbb{R}^d$ such that $w_{uv} = \text{dist}(f(u), f(v))$ for each $uv \in E(G)$, where $\text{dist}(f(u), f(v))$ is the Euclidean distance between $f(u)$ and $f(v)$? (We call such a mapping f a *d-embedding* of G.)

We also consider a variant of the problem described as follows.

Problem: Weighted Graph Embeddability in d-space with Distinctness (WGEdwD)
Instance: A graph G with nonnegative weights $w_e \geqslant 0$ on each edge $e \in E(G)$.
Question: Is there a mapping $f: V(G) \to \mathbb{R}$ such that $f(u) \neq f(v)$ for $u \neq v$, and $w_{uv} = \text{dist}(f(u), f(v))$ for each $uv \in E(G)$?

For convenient purpose, we call Weighted Graph Embeddability in d-space and its variant WGEd and WGEdwD respectively. Unfortunately, WGEd is known to be strongly NP-hard in general and weakly NP-hard for cycles.

Theorem 1.1. *(See Saxe [24], Feder and Motwani [12].) For every positive integer d, WGEd is NP-hard even if every edge has weight one or two. Furthermore, WGE*1 *is weakly NP-complete even for cycles.*

Feder and Motwani [12] studied the problem Graph Turnpike (GT), which is equivalent to the problem WGE1. They also studied the variant of GT called Graph Turnpike with Distinctness (GTwD), which is equivalent to the problem WGE1wD. They showed that this variant is also weakly NP-hard for cycles [12].

Theorem 1.1 implies that a partial distance matrix corresponding to a graph is not always helpful deciding the embeddability. Therefore, it is an interesting problem to ask which graphs (and which d) provide a sufficient condition for designing an efficient algorithm for deciding embeddability. This paper gives an initial work for this direction of research. Considering Theorem 1.1, we have the following natural questions: (1) If there is no long cycle without a chord, does the problem remain hard? (2) Is the complexity of the problem monotone with respect to the dimension d of the embedded space? (3) If there is a dominating set S for which the embedding can be uniquely determined or the number of possible embeddings is small enough, can we design an efficient algorithm for the reconstruction (this corresponds to the problem in surveying engineering)? We answer each of these questions in the following sections.

We assume a computational model used by Saxe [25] in which real numbers are primitive data objects on which exact arithmetic operations (including comparisons and extraction of square roots) can be performed in constant time.

1.2. Our results

Our contribution for the tractability of embedding problems can be divided into two parts: polynomial-time algorithms for cycles and chordal graphs, and fixed-parameter tractable algorithms for graphs with dominating cores. We give a linear-time algorithm to solve WGEd for cycles ($d \geqslant 2$) and an $O(n^2)$-time algorithm for chordal graphs ($d \geqslant 1$). Moreover, we study the graphs with small dominating sets and offer some efficient algorithms to solve WGEd and WGEdwD when $d = 1, 2$. The results are derived from a geometric fact that the intersection of d hyperspheres in d-space includes at most two points if the centers of the hyperspheres are in general position. Suppose we know the positions of d points and all the distances from them to another point p, then we can restrict p at only two possible positions. Therefore, if we have a small set of vertices S which we can guess their positions and the vertices in S have strong enough connection to the other vertices in $V \setminus S$, we can restrict each of vertices in $V \setminus S$ to at most two possible positions. Here a d-dominating set could help us to do the right job. Next, because each vertex in $V \setminus S$ has at most two possible positions, we could assign $0, 1$ to each of them indicating the choice of two positions. To satisfy the edge constraints between the vertices in $V \setminus S$, we can construct a 2-SAT instance and solve it efficiently. We summarize our general frame work as follows:

Theorem 1.2. *Let G and S be a given n-vertex graph and its d-dominating set which is also given. Let the points satisfy the general position condition. If the number of all possible candidates of d-embeddings of $G[S]$ is $g(|S|)$, and all these candidates can be enumerated in* poly(n) *time for each, then we can solve WGEd and WGEdwD in* $O(g(|S|) \cdot (\text{poly}(n) + n^2))$ *time.*

Notice that Theorem 1.2 is under the assumption that the points are already known to be in general position. If we remove this condition, we can still apply Theorem 1.2 when $d = 1, 2$. Unfortunately, we currently have no efficient algorithms for the case when $d > 2$. We will discuss our general frame work when $d = 1, 2$ and apply it on several graphs to obtain some interesting results in Section 4.