Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6643810, 13 pages
https://doi.org/10.1155/2021/6643810

Research Article

On Constructing t-Spanner in IoT under SINR

Xiujuan Zhang,[1,2] **Yongcai Wang**,[1] **Wenping Chen**,[1] **Yuqing Zhu**,[3] **Deying Li**,[1] **and Guangshun Li**[2]

[1]*School of Information, Renmin University of China, Beijing 100872, China*
[2]*School of Computer Science, Qufu Normal University, Rizhao 276826, China*
[3]*Department of Computer Science, California State University, Los Angeles, CA 90032, USA*

Correspondence should be addressed to Deying Li; deyingli@ruc.edu.cn

Received 25 October 2020; Revised 28 December 2020; Accepted 15 January 2021; Published 2 February 2021

Academic Editor: Zhuojun Duan

Following the recent advances in the Internet of Things (IoT), it is drawing lots of attention to design distributed algorithms for various network optimization problems under the SINR (Signal-to-Interference-and-Noise-Ratio) interference model, such as spanner construction. Since a spanner can maintain a linear number of links while still preserving efficient routes for any pair of nodes in wireless networks, it is important to design distributed algorithms for spanners. Given a constant $t > 1$ as the required stretch factor, the problem of our concern is to design an efficient distributed algorithm to construct a t-spanner of the communication graph under SINR such that the delay for the task completion is minimized, where the delay is the time interval between the time slot that the first node commences its operation to the time slot that all the nodes finish their task of constructing the t-spanner. Our main contributions include four aspects. First, we propose a proximity range and proximity independent set (PISet) to increase the number of nodes transmitting successfully at the same time in order to reduce the delay. Second, we develop a distributed randomized algorithm SINR-Spanner to construct a required t-spanner with high probability. Third, the approximation ratio of SINR-Spanner is proven to be a constant. Finally, extensive simulations are carried out to verify the effectiveness and efficiency of our proposed algorithm.

1. Introduction

The Internet of Things (IoT) has attracted great attention in recent years, owing to its potential military and civilian applications [1, 2]. Such a network generally consists of a large number of autonomous network nodes, in which algorithms are usually distributed since these algorithms have to work without global information and coordinated central control. Hence, there is an imperative need to design efficient distributed algorithms for various network optimization problems in the IoT.

Constructing a t-spanner with a minimum number of edges is one of the fundamental network optimization problems since the spanner property is a critical requirement of topology control in the IoT [3]. The IoT is commonly modeled as a graph $G(V, E)$, in which V is the set of wireless nodes and E represents the set of communication links (edges) connecting the nodes in V. A spanning subgraph H of G is called a t-spanner, for $t \geq 1$ if for all pairs of nodes u, $v \in V$, the length of the shortest path from u to v in H is at most t times of that in G. Here, t is called the *stretch factor*. A spanner can not only decrease the number of links and maintain connectivity but also ensure that the length of a path between any pair of communication nodes is within some constant factor from the shortest possible one. Therefore, constructing a spanner of the communication graph is enormously helpful for topology control, geographic routing, and compact routing in the IoT [3].

Spanners have been extensively studied in computational geometry [4], in which $G(V, E)$ is generally the complete Euclidean graph of the node set V in the Euclidean plane. However, even in the field of computational geometry, computing a minimum stretch factor spanner using not more than a given number of edges is NP-hard [5]. In the IoT, current spanner construction algorithms either do not consider interference, such as [6], or only handle it under the protocol interference model [7].

However, when multiple nodes send messages at the same time, a node may be unable to receive the message from its given sender owing to the interference caused by simultaneous transmissions. The *protocol interference model* and the *SINR (Signal-to-Interference-and-Noise-Ratio) interference model* are the commonly used interference models. SINR can take into account cumulative interference of wireless communications and is more realistic [8], thus has been widely adopted now. However, designing and analyzing algorithms are challenging under the SINR model since each given receiver should compute accumulated interference generated by all other senders at the same time.

1.1. Outline of the Problem. In this paper, we consider a general case of constructing a t-spanner under SINR, namely t-spanner-SINR. That is, given a constant t as the required stretch factor, our objective is to design an efficient distributed algorithm for constructing a t-spanner of the communication graph to minimize the delay of constructing the t-spanner. The delay of constructing a t-spanner is defined as the time interval from the start time-slot that nodes start to work to the last time-slot that all the nodes finish their task of constructing the t-spanner.

A large scale IoT discussed in this work consists of n sensor nodes, with uniform transmission powers, deployed randomly and uniformly in the two-dimensional Euclidean space. Nodes act in synchronous rounds; in every communication round, a node can transmit a message and attempt to receive a message. Each node initially knows only its own unique ID and its own coordinates. Since we adopt the SINR interference model when nodes communicate, the predesigned receivers can successfully decode the messages if and only if SINR constraints are satisfied. Owing to the accumulation and uncertainty of the SINR model, it is challenging to design a t-spanner distributed algorithm based on SINR in wireless networks. How to make more nodes transmit simultaneously and meanwhile make their given neighbor nodes successfully decode the messages is crucial to the performance of the algorithm under SINR.

The authors in [9] proposed a distributed algorithm SINR-Undirected-YG under SINR and proved that the resultant graph is a t-spanner. Zhang et al. [9] claim to be able to construct a spanner at $O(\log n)$ time-slots, but the running time of its algorithm is very large during the simulation. In this paper, we study a general case of constructing a t-spanner under SINR and try to reduce the delay.

1.2. Summary of Contributions. The summary of contributions of this paper is as follows:

(1) We identify the general case of constructing a t-spanner under SINR (t-spanner-SINR problem), and we design an efficient distributed algorithm SINR-Spanner to construct a required t-spanner under SINR with high probability, i.e., with a probability of at least $1 - e^{-(n/4)}$, where n is the total number of nodes in the network. Moreover, the resultant t-spanner has $O(n)$ edges

(2) Our distributed algorithm SINR-Spanner is also a local algorithm, in which the topology can be locally and self-adaptively maintained based on the information from the neighbor nodes without affecting the whole network. We reasonably utilize one kind of proximity graph—Yao graph (YG)—to construct spanner under SINR. YGs divide the surrounding area of each node into k sectors of equal angles and add edges only to the nearest neighbor within each sector [10]. If there are two or more nearest neighbors in a sector, one can choose the first neighbor receiving the message. In our design, each node is capable of independently performing successful local broadcasts to collect its neighborhood information within a certain range, such that it can get the nearest neighbor in each sector and the resultant t-spanner is a special YG

(3) We introduce the definition of proximity range and proximity-independent set (PISet) to increase the number of nodes transmitting successfully at the same time and to reduce the delay in the SINR-Spanner algorithm. The approximation ratio of SINR-Spanner is proven to be a constant

(4) Extensive simulations are carried out to verify the effectiveness and efficiency of our proposed distributed and randomized algorithm

The rest of this paper is organized as follows. Section 2 reports the most related work. Section 3 precisely defines the formulation of the problem and introduces relevant models and notations. The spanner construction algorithm SINR-Spanner is presented in Section 4. Section 5 gives a theoretical analysis of the algorithm. In Section 6, we evaluate the performance of the algorithm via simulation. Section 7 concludes the paper with suggestions for future work.

2. Related Work

In this section, we first investigate the spanner algorithms in computational geometry and in the IoT, then discuss the method of applying randomized and distributed solutions under SINR.

2.1. t-Spanner. The book [4] by Narasimhan and Smid is a comprehensive overview of geometric spanners. For geometric spanners, several structures and methods have been proposed, such as the Greedy method [11], Well-Separated Pair Decomposition method, Delaunay triangulation, θ-graphs, and YGs. Constructing YGs is one of the simplest ways of constructing t-spanners. Yao [10] used YGs to simplify the computation of the Euclidean minimum spanning tree. Althöfer et al. [11] firstly proved that YGs are the t-spanners for the corresponding complete graph. For the corresponding complete graph, YGs are $1/(1 - 2\sin(\pi/k))$-spanners with $k > 6$ [12].

In wireless networks, the spanner property was first discussed by Li et al. in [6]. They modeled the network as a unit disk graph (UDG) and analyzed the energy stretch factor of several common subgraphs of a UDG: $n-1$ for the relative neighborhood graph (RNG), 1 for the Gabriel graph (GG), and $O(1)$ for YG. And these proximity graphs have been widely used in spanner construction as subgraphs of a UDG. There also exist spanner construction mechanisms for quasi-unit disk graphs, disk graphs, and unit ball graphs [13]. In [14], Kothapalli et al. proposed a local-control protocol for establishing a constant density spanner among a set of mobile stations. The LISE (low interference spanner establisher) algorithm was presented to establish a spanner in [7], where the interference definition is based on how many nodes are affected by the communication over a certain link. However, since the above spanner algorithms for wireless networks are all studied without considering interference or handling interference under the protocol interference model, they cannot deal with interference effectively under the SINR interference model. Zhang et al. [9] first consider spanner construction under SINR, and this work improves it.

Constructing spanners under a computational geometry field greatly promotes the study under the wireless network setting. Meanwhile, wireless network requirements, which generally need to efficiently satisfy various topological characteristics, encourage the development of geometric spanner construction. Recent results on sparse geometric spanners focused on satisfying one or multiple topological characteristics such as lightness, small degree [15], fault tolerance, no central agent [16], and multiple characteristics [17].

2.2. SINR Model. In wireless networks, the SINR model received increasing attention [8]. Despite the vast amount of researches in the design and analysis of centralized algorithms under SINR [18], few results are known about distributed solutions in this model, especially for global communication tasks, owing to the accumulation and uncertainty of interference.

There is a growing interest in developing randomized distributed solutions to local broadcast [19], which is defined as successfully transmitting a message to all neighbors in the corresponding reachable proximity of a node. Randomized distributed solutions to local broadcast are often used as a building block for global communication tasks, such as multiple-message broadcast [20], synchronization [21], and multiple channels broadcast [22]. In [20], selected leader nodes adopt local broadcast to collect the messages that arrive at their dominated nonleader nodes and then disseminate the received messages to the whole network. The algorithm in [21] starts from very low probabilities and increases them gradually until nodes can hear a reasonable number of messages and implement all nodes' clock synchronization. In [22], the selected leader in different channel collects locally the messages of its dominated nonleader nodes in the same channel to speedup multiple channel broadcast. Note that most of the existing distributed and randomized works under SINR are still in the theoretical stage except [19] and Fuchs's coloring study, such as [23]. Using local broadcasting as a basic unit, we propose an algorithm of spanner construction in this paper and perform simulations to verify the performance of the algorithm.

3. Model and Problem Formulations

Assume that a set V of n wireless network nodes, modeled as a graph G, is deployed randomly and uniformly in a 2-dimensional geographic plane. Nodes act in synchronous rounds. Each node is conscious of its ID and coordinates and has the same transmission power P ($P > 0$). Let the Euclidean distance for the two endpoints u and v, denoted by d_{uv}, be the length of an edge in G. And let $d_{uv}(G)$ be the length of the shortest path between u and v in G which is defined as the sum of the lengths of its edges.

A commonly assumed model for the propagation effect of wireless nodes is deterministic path loss, i.e., $P_r = P/d_{uv}^{\alpha}$, where u transmits a message to v, P_r is the received power at a receiver v, and α is the path loss exponent (typically, $2 < \alpha \leq 6$). A deterministic path loss model is applied to the following interference model.

3.1. SINR (Signal-to-Interference-and-Noise-Ratio) Interference Model. In the SINR model, a transmission from node u to node v is successful iff the SINR condition holds:

$$\frac{P/d_{uv}^{\alpha}}{N + \sum_{w\in T\setminus\{u\}} (P/d_{wv}^{\alpha})} \geq \beta, \tag{1}$$

where $T \subseteq V$ is the set of transmitting nodes, $\alpha \in (2, 6]$ is the path loss exponent depending on the network environment, $\beta > 1$ is a hardware-defined threshold, and N is the environmental noise.

The transmission range $R_{\max}$ of a node u is the maximum distance at which a node v can receive a clear transmission from u while no other node is transmitting at the same time, i.e., $\sum_{w\in T\setminus\{u\}} (P/d_{wv}^{\alpha}) = 0$. The SINR condition (1) tells us that

$$R_{\max} = (P/N\beta)^{1/\alpha}, \tag{2}$$

for the given power level P. Note that $R_{\max}$ is for only one node u transmitting in the whole network at the time slot.

3.2. Local Broadcasting Range (R_b). We set the local broadcasting range

$$R_b = (1-\varepsilon) R_{\max}, \tag{3}$$

where $0 \leq \varepsilon < 1$ is a fixed model parameter.

We say a node transmits R_b successfully in a time slot if it transmits a message, and this message is received by all its neighbors in a distance smaller or equal to R_b in the time slot. In Section 4, we define the proximity range R_p such that the nodes which are in a distance greater or equal to R_p can transmit R_b successfully at the same time slot.

We denote the region within R_b of node u as a local broadcasting region B_u and the number of nodes in it as Δ_u^b. Furthermore, let $\Delta^b = \max_{u\in V}\{\Delta_u^b\}$.

3.3. Communication Graph. The communication graph $G^{R_b}(V, E^{R_b})$ of a given network consists of all network nodes and edges (u, v) such that $d_{uv} \leq R_b$. Since $R_{\max}$ is for only one transmission in the whole network at the same time, we adopt a slightly smaller range R_b as [22] which suffice for practical communication. In the communication graph $G^{R_b}(V, E^{R_b})$, which is simply denoted by $G^{R_b}(V)$, a node v is a neighbor of node u if $d_{uv} \leq R_b$.

3.4. Yao Graph (YG). The directed Yao graph $\overrightarrow{YG}_k(V)$ with a fixed integer parameter $k > 0$ is defined as follows. Any k equally separated rays starting at the origin node define k sectors. The orientation of the cut is identical for all nodes. Translate the sectors to each node $u \in V$. In each sector with a node u, pick the shortest directed edge $\langle u, v \rangle$, if there is one, to $\overrightarrow{\mathrm{YG}}_k(V)$. Ties are broken arbitrarily.

This implies that $\overrightarrow{\mathrm{YG}}_k(V)$ preserves the shortest outcoming edge in each sector. Accordingly, $\overleftarrow{\mathrm{YG}}_k(V)$ preserves the shortest incoming edge in each sector.

An undirected Yao graph, in which the edge directions are ignored, is denoted by $\mathrm{YG}_k(V)$. $\mathrm{YG}_k^{R_b}(V)$ is the Yao graph in which only the edges whose lengths are no more than R_b are preserved from $\mathrm{YG}_k(V)$. In other words, $\mathrm{YG}_k(V)$ is the spanning subgraph of a complete Euclidean graph $K_n(V)$ on node set V and $\mathrm{YG}_k^{R_b}(V)$ is the spanning subgraph of $G^{R_b}(V)$.

3.5. t-Spanner. Let $t > 1$ be a real number. A spanning subgraph $H(V, E_H)$ of $G(V, E)$ is said to be a t-spanner of G, if for any two nodes u and v in V, the shortest path between u and v in H, whose length is at most t times that of the shortest path in G, i.e.,

$$d_{uv}(H) \leq t \cdot d_{uv}(G). \tag{4}$$

The constant t is called the *stretch factor* of H (w.r.t. G). Note that G can be $K_n(V)$ or a communication graph that is a spanning subgraph of $K_n(V)$.

3.6. Delay of Constructing a t-Spanner. The delay of constructing a t-spanner is defined as the time interval from the start time slot that the first node starts to work to the last time slot that all the nodes finish their task of constructing the t-spanner.

Next, we present the definition of the problem, the t-spanner under the SINR problem, which is our focus in this paper.

3.7. t-Spanner under the SINR Problem (t-Spanner-SINR). Assume that a set V of n wireless network nodes deployed randomly and uniformly in a 2-dimensional geographic plane, in which each node is aware of its ID and coordinates, has the same transmission power P ($P > 0$) and a local broadcasting range R_b; given a constant $t > 1$, the goal is to design a distributed algorithm to find a t-spanner of the corresponding communication graph $G^{R_b}(V, E^{R_b})$ under SINR, such that the delay of constructing a t-spanner is minimized.

4. Algorithm

4.1. Algorithm Outline. The main idea of our algorithm is to construct $\mathrm{YG}_k(V)$ for the t-spanner-SINR problem. The reason is that the $\mathrm{YG}_k(V)$ graph is a $t = 1/(1 - 2\sin(\pi/k))$ -spanner [9]. Consequently, given a constant $t > 1$, we compute a k and construct $\mathrm{YG}_k(V)$ under SINR in our distributed algorithm to get the required t-spanner.

To construct $\mathrm{YG}_k(V)$, each node u needs the node v's information which is the closest to u in the sector v belongs. Accordingly, each node should locally broadcast its ID and coordinates to its neighbors. However, if all the nodes transmit together under SINR, no node will receive any message owing to the interference. To avoid collision, each node could locally broadcast one by one. However, in a distributed algorithm, there is not a centralized coordination for arranging nodes to transmit in sequence. Accordingly, each node can only transmit with the probability $1/n$. Furthermore, in order to reduce the delay, it has an obligation to have as many nodes as possible to transmit simultaneously. Therefore, each node will make a range as the radius of its neighborhood circle region and next take the inverse of the number of nodes in its neighborhood circle region as the sending probability.

First, we will compute the range and give the node set in which all nodes can transmit simultaneously.

4.2. Proximity Range and Proximity Independent Set (PISet). How to make more nodes transmit simultaneously and meanwhile make their neighbor nodes successfully decode the messages is crucial for the performance of the algorithms under SINR. Thus, we define the proximity range R_p such that any nodes u and v can transmit simultaneously if $d_{uv} \geq R_p$. Intuitively, a tiny R_p implies a high degree of channel utilization. We examine how to set a proper R_p to guarantee SINR threshold and meanwhile the highest channel utilization degree. For clarity, we define the proximity range and proximity independent set as follows.

4.2.1. Proximity Range R_p and the Corresponding Proximity Independent Set PISet_p. Proximity range R_p is a length, and a PISet_p is a subset of V that satisfies $d_{uv} \geq R_p$ for $\forall u, v \in \mathrm{PISet}_p (u \neq v)$. A PISet_p is maximal if and only if $d_{uw} < R_p$ for $\forall u \in \mathrm{PISet}_p$ and $\forall w \notin \mathit{PISet}_p$. How to design R_p? The basic idea underlying the design is to ensure the nodes in the same $\mathrm{PIS}et_p$ transmit simultaneously and all their neighbors receive the message successfully.

We refer to the region within R_p of node u as proximity region X_u and the number of nodes in it as Δ_u^p. Besides, let $\Delta^p = \max_{u \in V}\{\Delta_u^p\}$.

Before designing R_p, we first give an example for R_p and PISet_p. R_p and PISet_p are illustrated in Figure 1 where 19 nodes represented by solid circles are in the same PISet_p. Note that this PISet_p is maximal. The distance between the given receiver v with the sender u and the nearest of other senders, which is w in Figure 1, is at least $(R_p - R_b)$. In other words, $d_{wv} \geq (R_p - R_b)$ in Figure 1.

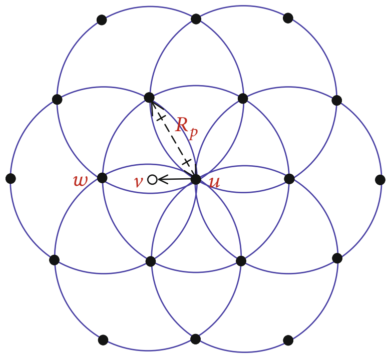

FIGURE 1: Proximity range R_p and a corresponding PISet_p.

By observing the above example, we give the specific relationship between R_p and R_b in the following theorem.

Theorem 1. *Suppose that $R_p = (c_2 \cdot \beta \cdot (c_1/(c_1-1)))^{1/\alpha} \cdot R_b + R_b$ where $c_1 = (1/(1-\varepsilon)^\alpha) > 1$ and $c_2 = 6 + (\pi^2 - 6)(\sqrt{3}/2)^{-\alpha}$, the nodes in the same PISet_p can transmit R_b successfully at the same time slot.*

Proof. Let $I = R_p - R_b$. We begin by estimating the smallest value of I when u transmits to v successfully and u transmits simultaneously with other nodes in the same PISet_p. Further, we prove that the nodes in the same PISet_p can transmit R_b successfully at the same time slot with the above I. We say a node transmits R_b successfully in a time slot if it transmits a message and this message is received by all its neighbors in a distance smaller or equal to R_b in the time slot.

In order for v to be able to receive the message from u, we require $\text{SINR}_{uv} \geq \beta$.

Thus,

$$\frac{P/d_{uv}^{\alpha}}{N + \sum_{w\in \text{PISet}_p\setminus\{u\}} (P/d_{wv}^{\alpha})} \geq \beta. \tag{5}$$

Since the equations (2) and (3), $N = P/c_1\beta R_b^\alpha$ where $c_1 = 1/(1-\varepsilon)^\alpha$ is a fixed parameter.

Now

$$\begin{aligned}\frac{P/d_{uv}^{\alpha}}{N + \sum_{w\in \text{PISet}_p\setminus\{u\}} (P/d_{wv}^{\alpha})} &= \frac{P/d_{uv}^{\alpha}}{(P/c_1\beta R_b^{\alpha}) + \sum_{w\in \text{PISet}_p\setminus\{u\}} (P/d_{wv}^{\alpha})} \\ &= \frac{d_{uv}^{-\alpha}}{(R_b^{-\alpha}/c_1\beta) + \sum_{w\in \text{PISet}_p\setminus\{u\}} d_{wv}^{-\alpha}}.\end{aligned} \tag{6}$$

We derive the lower bound of the above formula. First, $d_{uv}^{-\alpha} \geq R_b^{-\alpha}$ since R_b is the maximum local broadcasting range of a node. Furthermore, if a node w transmit together with u as shown in Figure 1, w produces the largest interference when w and the given receiver v have the closest distance I. If we represent a link as a node as shown in Figure 2(a), for the nodes in the PISet_p, the densest packing of interfering links is the hexagon packing [24] with edge length I as shown in Figure 2(b). There are at most six nodes in the first layer, and the distance is I with respect to the abstracting node uv. Furthermore, the distance between uv and any node in the lth ($l \geq 2$) layer is no less than $(\sqrt{3}/2)lI$ with the lth layer having at most $6l$ nodes.

$$\begin{aligned}\sum_{w\in \text{PISet}_p\setminus\{u\}} d_{wv}^{-\alpha} &\leq 6\cdot I^{-\alpha} + \sum_{l\geq 2} 6l \cdot \left(\frac{\sqrt{3}}{2} lI\right)^{-\alpha} \\ &= 6\cdot I^{-\alpha} + 6\cdot \left(\frac{\sqrt{3}}{2} I\right)^{-\alpha} \cdot \sum_{l\geq 2} l^{-\alpha+1}.\end{aligned} \tag{7}$$

Since $\sum_{l\geq 2} l^{-\alpha+1} = \zeta(\alpha-1) - 1$, where $\zeta(\cdot)$ is the *Riemann zeta function*, considering that $\alpha \geq 3$, then $\zeta(\alpha-1) \leq \zeta(2) = \pi^2/6$. Thus, we have

$$\begin{aligned}\sum_{w\in \text{PISet}_p\setminus\{u\}} d_{wv}^{-\alpha} &\leq 6\cdot I^{-\alpha} + 6\cdot \left(\frac{\sqrt{3}}{2} I\right)^{-\alpha} \cdot \left(\frac{\pi^2}{6} - 1\right) \\ &= \left(6 + (\pi^2 - 6)\left(\frac{\sqrt{3}}{2}\right)^{-\alpha}\right)\cdot I^{-\alpha} = c_2 \cdot I^{-\alpha},\end{aligned} \tag{8}$$

where $c_2 = 6 + (\pi^2 - 6)(\sqrt{3}/2)^{-\alpha}$.

Therefore, to make $(d_{uv}^{-\alpha}/((R_b^{-\alpha}/c_1\beta) + \sum_{w\in \text{PISet}_p\setminus\{u\}} d_{wv}^{-\alpha})) \geq \beta$ valid, it is sufficient to have

$$\frac{R_b^{-\alpha}}{(R_b^{-\alpha}/c_1\beta) + c_2 I^{-\alpha}} \geq \beta. \tag{9}$$

Therefore, $I \geq (c_2 \cdot \beta \cdot (c_1/(c_1-1)))^{1/\alpha} \cdot R_b$.

Hence, $R_p = (c_2 \cdot \beta \cdot (c_1/(c_1-1)))^{1/\alpha} \cdot R_b + R_b$, where $c_1 = 1/(1-\varepsilon)^\alpha > 1$ and $c_2 = 6 + (\pi^2 - 6)(\sqrt{3}/2)^{-\alpha}$.

Conversely, if R_p have the value as shown in the above, the nodes in the same PISet_p, which is maximal as shown in Figure 1 or not maximal, can transmit R_b successfully at the same time slot since the receiver power is no less than $(P\cdot) R_b^{-\alpha}$ in (9) and the cumulative interference is less than $(P\cdot) c_2 I^{-\alpha}$.

Figure 3 depicts visually the relation between proximity range R_p and local broadcasting range R_b with different α, β, and ε, which is helpful to pick these values during subsequent algorithm simulation. These parameters are reasonably set as follows: the path loss exponent $\alpha \in \{3,4,5,6\}$, the threshold $\beta \in [1,10]$, and $\varepsilon \in [0.4, 0.95]$. From Figure 3, one can see that R_p increases when α decreases, β increases, and ε decreases. So we can get that R_p is as about 2.57 times at a minimum as R_b when $\alpha = 6$, $\beta = 1$, and $\varepsilon = 0.95$, while R_p is as about 6.34 times at a maximum as R_b when $\alpha = 3$, $\beta = 10$, and $\varepsilon = 0.4$.

4.3. Algorithm. From the above Theorem 1, the nodes in the same PISet_p can transmit R_b successfully at the same time slot while each node takes the value of R_p as Theorem 1, so the sending probability for each node u is set to the inverse of the number of nodes in its proximity region Δ_u^p. Thus, we

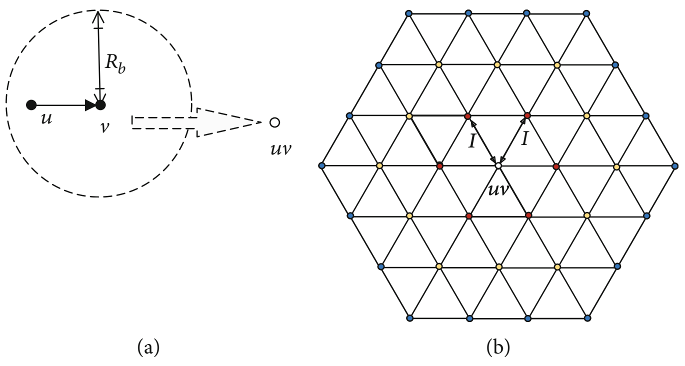

FIGURE 2: Link abstraction and the densest packing of interfering links.

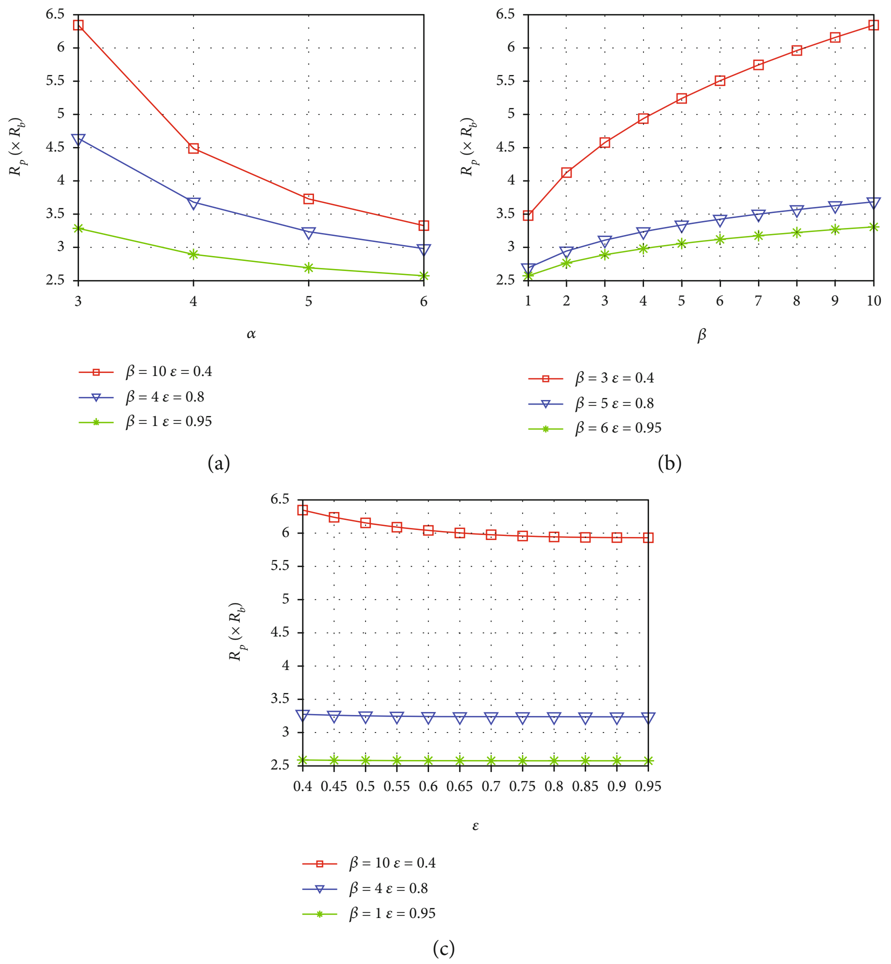

FIGURE 3: R_p vs. α, β, and ε.

guess all its neighbors can receive the message successfully with high probability, and we will give the proof in the next section. The pseudo-code for node u is given in Algorithm 1, which implies that each node runs it independently.

Now, we describe the full operation of our distributed algorithm SINR-Spanner. Each node u carries out the same operations and has its local memory. The algorithm consists of three parts. In part 1 (line 1-line 5), each node performs initialization work. Each node u first computes the required number of sectors depending on the given stretch factor t. Then, each node u takes the inverse of Δ_u^p as the sending probability. Thus how to obtain Δ_u^p *a priori* is critical to our algorithm design. Some existing works, such as [19, 23], assume the availability of Δ_u^p to facilitate the algorithm design and analysis. Such requirements are common under SINR in order to enable initial communication. Therefore, in our algorithm, we also assume that Δ_u^p is available to simplify the presentation. In part 2 (line 6-line 20), each node obtains

```
1: Initialize the stretch factor $t$;
2: Initialize $k = \lceil \pi/\arcsin((t-1)/(2t)) \rceil$;
3: Compute the number of nodes in its proximity region, i.e., $\Delta_u^p$;
4: Initialize $p_u = 1/\Delta_u^p$;
5: Initialize $I_u[i] = -1$ for $i = 1, 2, \cdots, k$;
6: for $j = 1$ to $\Delta^p$ time-slots do
7:   Send a message containing its ID $u$ and coordinates with probability $p_u$, and remains listening with probability $1 - p_u$;
8:   while receiving a message from some node $v$ do
9:     $i$ = the index of the sector to which $v$ belongs;
10:      if $I_u[i] == -1$ or the distance $d_{I_u[i]u} > d_{vu}$ then
11:        $I_u[i] = v$
12       end if
13:          end while
14: end for
15: $\underset{E_u}{\leftarrow} = \varnothing$;
16: for $i = 1$ to $k$ do
17:     if $I_u[i] \neq -1$ then
18:     $\overleftarrow{E_u} = \overleftarrow{E_u} \cup \langle I_u[i], u \rangle$;
19:     end if
20: end for
21: $E_u = \{(u, v) \mid \langle v, u \rangle \in \overleftarrow{E_u}\}$;
22: for $i = 1$ to $\Delta^p$ time-slots do
23:   Broadcast the incoming neighbor set with probability $p_u$;
24:   while receiving a message from some node $v$ do
25:     if $u$ is the incoming neighbor of $v$ and $(u, v) \notin E_u$ then
26:       $E_u = E_u \cup (u, v)$;
27:     end if
28:   end while
29: end for
```

ALGORITHM 1: SINR-Spanner(u).

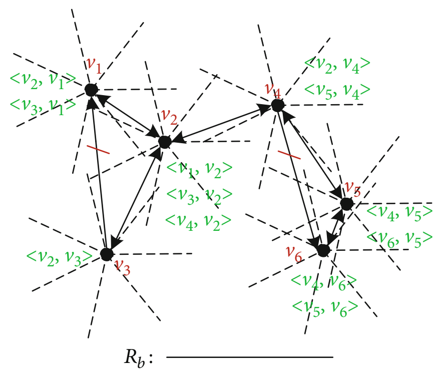

FIGURE 4: $\overleftarrow{YG}_k^{R_b}(V)$ after part 2, where $k = 7$.

the incoming neighbor in each sector by receiving the neighbors' messages, and thus, a directed Yao graph forms. Figure 4 shows an intermediate result after part 2 in which there are 6 nodes; the edge length is at most R_b and $k = 7$. Here, $\overrightarrow{E_{v_1}} = \{\langle v_3, v_1 \rangle \langle v_2, v_1 \rangle\}$, $\overrightarrow{E_{v_2}} = \{\langle v_1, v_2 \rangle \langle v_3, v_2 \rangle \langle v_4, v_2 \rangle\}$, $\overrightarrow{E_{v_3}} = \{\langle v_2, v_3 \rangle\}$, and so on. In part 3 (line 21-line 29), each node sends an acknowledgement message back to its incoming neighbor, and thus, an undirected Yao graph is constructed which is a t-spanner we require. An example of $YG_k^{R_b}(V)$ is presented in Figure 5, in which edge directions are ignored from the graph shown in Figure 4. Here, $E_{v_1} = \{(v_1, v_2), (v_1, v_3)\}$, $E_{v_2} = \{(v_2, v_1), (v_2, v_3), (v_2, v_4)\}$, $E_{v_3} = \{(v_3, v_1), (v_3, v_2)\}$, and so on. E_{v_1} has local data (v_1, v_3), and E_{v_3} has local data (v_3, v_1); thus, the undirected edge (v_1, v_3) is known by two endpoints.

Our distributed and randomized algorithm SINR-Spanner, which is different from the algorithms in [9], can solve the t-spanner-SINR problem. In part 1, the algorithm first initializes the stretch factor t to attain a t-spanner. The sending probability for each node u is set to the inverse of Δ_u^p to reduce the delay; then, u repeats randomized transmission for $8\Delta^p$ time slots, respectively, in part 2 and part 3.

5. Performance Analysis of Algorithm SINR-Spanner

5.1. The Delay of SINR-Spanner. In order to obtain the main result of this section in Theorem 8, we first prove that a graph $YG_k^{R_b}(V)$ is constructed with high probability after the algorithm SINR-Spanner terminates in $16\Delta^p + c$ time slots in Theorem 3; next, analyze why the resultant $YG_k^{R_b}(V)$ is the required t-spanner of the communication graph $G^{R_b}(V)$ in Theorem 3.

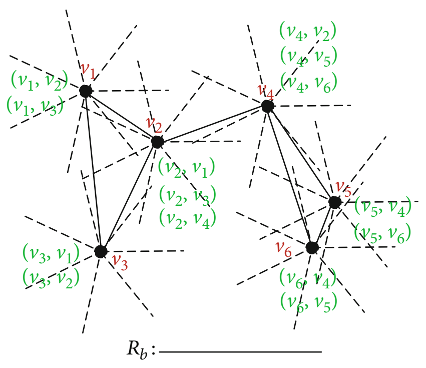

FIGURE 5: $YG_k^{R_b}(V)$ after part 3, where $k = 7$.

Now, we give a form of Chernoff bounds which can be found in [22] and some advanced textbooks, such as [25], for the proof of Theorem 3.

Lemma 2 [22, 25] (Chernoff bounds). *Let $0 < \delta \leq 1$ and X_1, $X_2, \cdots, X_n$ be independent Bernoulli random variables, and let $X := \Sigma_{i=1}^{n} X_i$ and $\mu = E[X]$. Then, for any $\delta > 0$, it holds that*

$$Prob(X < (1 - \delta)\mu) < \left(\frac{e^{-\delta}}{(1 + \delta)^{(1+\delta)}}\right)^{\mu} < e^{-\delta^2\mu/2}. \quad (10)$$

And for $\delta = 1/2$,

$$Prob\left(X < \frac{1}{2}\mu\right) < e^{-\mu/8}. \quad (11)$$

Theorem 3. *A graph $YG_k^{R_b}(V)$ is constructed with high probability after the algorithm SINR-Spanner terminates in $16\Delta^p + c$ time-slots, where Δ^p is the maximum of the node number in one proximity region and c is a constant for the number of time slots for the initialization work.*

Proof. Since R_p is as about 2.57 times to 6.34 times as R_b from Theorem 1 and the resultant graph is connected, $\Delta_u^p > 2$ for any node u. The probability that u transmits as the only transmitting node in its proximity region is

$$C_{\Delta_u^p}^1 \cdot \frac{1}{\Delta_u^p} \cdot \left(1 - \frac{1}{\Delta_u^p}\right)^{\Delta_u^p - 1} = \left(1 - \frac{1}{\Delta_u^p}\right)^{\Delta_u^p - 1} > \left(1 - \frac{1}{\Delta_u^p}\right)^{\Delta_u^p} > \frac{1}{4}. \quad (12)$$

The last inequality holds, since $(1 - (1/\Delta_u^p))^{\Delta_u^p}$ increases when Δ_u^p increases, and it obtains the minimum value 1/4 when Δ_u^p is 2.

Since the nodes can transmit R_b successfully if they belong to the same PISet$_p$ owing to Theorem 1, there are at least n/Δ^p nodes transmitting together in one time slot. Consequently, there are at least $(n/\Delta^p) \cdot (1/4)$ nodes transmitting R_b successfully together in one time slot.

Therefore, after $8\Delta^p$ time slots, there are at least $2n$ nodes transmitting R_b successfully in expectation. By Chernoff bound (Lemma 2), after $8\Delta^p$ time slots, the probability

$$\text{Prob}(X < n) = \text{Prob}\left(X < \left(1 - \frac{1}{2}\right) \cdot 2n\right) < e^{-n/4}. \quad (13)$$

Therefore, the probability for all n nodes transmitting R_b successfully is

$$\text{Prob}(X = n) = 1 - \text{Prob}(X < n) \geq 1 - e^{-n/4}. \quad (14)$$

Thus, after $8\Delta^p$ time slots in part 2 of the algorithm SINR-Spanner, all the node transmit its ID and coordinates R_b successfully with high probability, i.e., each node obtains the "nearest" incoming neighbor in its proper sectors. Then, after a further $8\Delta^p$ time slots in part 3, all the node transmit acknowledgement messages R_b successfully with high probability. As a result, a graph $YG_k^{R_b}(V)$ is constructed with high probability.

In addition, the number of time slots for the initialization work in part 1 is a constant, which is denoted by c. So the algorithm SINR-Spanner terminates in $16\Delta^p + c$ time slots.

Obviously, the number of nodes in the proximity region is upper bound by n. If $\Delta^p = n$, a graph $YG_k^{R_b}(V)$ is constructed with high probability in $16n + c$ time slots.

Next, in order to prove that the resultant $YG_k^{R_b}(V)$ of the algorithm is a t-spanner of the communication graph $G^{Rb}(V)$, we apply some conclusions about $YG_k(V)$ where $YG_k(V)$ is constructed from $K_n(V)$.

Lemma 4 ([11]). *If V is a set of n points in the plane, and the integer $k \geq 2$, then the graph $YG_k(V)$ contains at most $kn = O(n)$ edges.*

Lemma 5 ([12]). *Let $t = 1/(1 - 2\sin(\pi/k))$ for the integer $k > 6$, $YG_k(V)$ is a t-spanner of $K_n(V)$, where n is the number of nodes in V.*

In the next, we show that $YG_k^{R_b}(V)$ is the required t -spanner of the communication graph $G^{R_b}(V)$ if $G^{R_b}(V)$ is connected in the following lemma of our previous work [9]. Zhang et al. [9] first give the condition that $G^{R_b}(V)$ is connected if and only if the longest edge in Euclidean minimum spanning tree of the node set V is at most R_b. Note that if the nearest neighbor of each node in every sector is within the local broadcast region, the graph $YG_k^{R_b}(V)$ is a $YG_k(V)$ and it is also a $1/(1 - 2\sin(\pi/k))$-spanner of $K_n(V)$.

Lemma 6 ([9]). *Let $t = 1/(1 - 2\sin(\pi/k))$. If $G^{R_b}(V)$ is connected and the integer $k > 6$, $YG_k^{R_b}(V)$ is a t-spanner of $G^{R_b}(V)$.*

Now, we show that the resultant $YG_k^{R_b}(V)$ of the algorithm is a t-spanner of the communication graph $G^{R_b}(V)$.

Theorem 7. *Given a constant $t > 1$, setting $k = \lceil \pi/(\arcsin((t - 1)/(2t))) \rceil$, the resultant graph $YG_k^{R_b}(V)$ is the required*

t-spanner with at most kn edges after the algorithm SINR-Spanner terminates.

Proof. As $k > 6$, the sector angle is not larger than $\pi/6$. Hence, $\sin(\pi/k)$ is a monotone decreasing function of k, and its value is less than 1/2. As a result, $t(= 1/(1 - 2\sin(\pi/k)))$ increases as k increases when $k > 6$ and approaches 1 as $k \longrightarrow \infty$, i.e., t is an injective and increasing function of k. Hence, its inverse function $k = \pi/(\arcsin((t-1)/(2t))$ is a decreasing function of t. Therefore, given a constant $t > 1$, we will find a k such that the Yao graph with the number of sectors of no less than k is the required t-spanner.

Since the number of sectors is an integer, and the number of edges becomes larger as the number of sectors increases from Lemma 4, k is assigned to $\lceil \pi/\arcsin((t-1)/(2t)) \rceil$. Therefore, the resultant graph $YG_k^{R_b}(V)$ is the required t-spanner. Furthermore, $YG_k^{R_b}(V)$ have kn edges by Lemma 4.

Based on Theorems 3 and 7, we can state the main conclusion of this section in Theorem 8.

Theorem 8. *The distributed algorithm SINR-Spanner constructs the required t-spanner with high probability, and the delay is $16\Delta^p + c$ time slots, where Δ^p is the maximum of the node number in one proximity region and c is a constant for the number of time slots for the initialization work.*

Lastly, Table 1 illustrates the relation between the stretch factor t and the number of sectors k in theory. In SINR-Spanner, k is the value in the second row given the corresponding t in the first row. Note that $t = 1/(1 - 2\sin(\pi/k))$ is the upper theory bound of the stretch factor for $YG_k^{R_b}(V)$ with respect to $G^{R_b}(V)$ so far, maybe a smaller k is sufficient in practical.

5.2. The Approximation Ratio of SINR-Spanner Algorithm. The goal of t-spanner-SINR problem is to find a suitable t-spanner under SINR and to minimize the delay in the construction. Now, we try to give the approximation ratio of the SINR-Spanner algorithm. The basic idea of our algorithm is that each node should locally broadcast its ID and coordinates to its neighbors R_b successfully under SINR. In order to reduce the delay, there should be as many nodes as possible to transmit simultaneously. As we see, in the SINR-Spanner algorithm, the nodes in different proximity region can transmit simultaneously. Now, we consider whether the nodes in a different local broadcasting region can transmit simultaneously, then discuss the approximation ratio of SINR-Spanner.

Theorem 9. *The approximation ratio of the algorithm SINR-Spanner is bounded by $(e/2) \cdot (R_p^2/R_b^2)$.*

Proof. The probability that any node u transmits as the only transmitting node in its local broad region is

$$C_{\Delta_u^b}^1 \cdot \frac{1}{\Delta_u^b} \cdot \left(1 - \frac{1}{\Delta_u^b}\right)^{\Delta_u^b - 1} = \left(1 - \frac{1}{\Delta_u^b}\right)^{\Delta_u^b - 1}. \quad (15)$$

Let S be the total area of wireless nodes distribution. Assume that a node as the only transmitting node in its local broadcasting region can transmit R_b successfully, the number of nodes transmitting R_b successfully in one time slot is

$$\frac{S}{\pi R_b^2} \cdot \left(1 - \frac{1}{\Delta_u^b}\right)^{\Delta_u^b - 1}. \quad (16)$$

In fact, the assumption cannot be guaranteed by theory and the following simulation, i.e., even though a node is the only transmitting node in its local broadcasting region, the node may not transmit R_b successfully owing to cumulative interference producing by other simultaneously transmitting nodes. So the solution adopted the above assumption is the low bound for t-spanner-SINR problem.

From the analysis of SINR-Spanner algorithm, the delay of constructing a t-spanner under SINR is mainly and inversely proportional to the number of nodes transmitting R_b successfully in one time slot. Hence, the approximation ratio of the algorithm SINR-Spanner is

$$\begin{aligned} &\frac{S}{\pi R_b^2} \cdot \left(1 - \frac{1}{\Delta_u^b}\right)^{\Delta_u^b - 1} \Big/ \left\{\frac{S}{\pi R_p^2} \cdot \left(1 - \frac{1}{\Delta_u^p}\right)^{\Delta_u^p - 1}\right\} \\ &= \frac{R_p^2}{R_b^2}\left\{\left(1 - \frac{1}{\Delta_u^b}\right)^{\Delta_u^b - 1} \Big/ \left(1 - \frac{1}{\Delta_u^p}\right)^{\Delta_u^p - 1}\right\} \\ &\le \frac{R_p^2}{R_b^2} \cdot \frac{1}{2} \Big/ \frac{1}{e} = \frac{e}{2} \cdot \frac{R_p^2}{R_b^2}. \end{aligned} \quad (17)$$

The inequality holds owing to the following: $\Delta_u^p > \Delta_u^b \ge 2$ for connectivity and $(1 - (1/x))^{x-1}$ decreases with x increases when $x \ge 2$. When x is large enough, replacing $x - 1$ by x does not cause much error and $(1 - (1/x))^x \cong 1/e$.

Since R_p is at about 2.57 to 6.34 times as R_b from Theorem 1 and Figure 3, the approximation ratio of SINR-Spanner is a constant.

6. Simulation

In the previous section, we theoretically prove that the algorithm SINR-Spanner performs well in the worst cases and the resultant graph is the required spanner. In this section, we conduct simulations to investigate the average performance of our algorithm.

Our simulations are coded in the Sinalgo simulation framework [26], which is for testing and validating network algorithms and abstracts from the underlying layers. We consider a square area of 1000 by 1000 and deploy n nodes within this network region randomly and uniformly, where $n \in \{3000, 3500, 4000, 4500, 5000\}$. The local broadcasting range R_b varies in $\{10, 12, 14, 16, 18\}$. The ambient noise $N = 5 \times 10^{-8}$mW. Figure 3 has given the ranges of the path loss exponent α, the threshold β, and ε, and the relation between them and the proximity range R_p. R_p mainly affects the sending probability, which affects the delay of our

Table 1: The stretch factor t and the number of sectors k in theory.

t	7.6	4.3	3.2	2.7	2.3	2.1	2	1.9	1.8	1.7	1.6	1.5	1.4	1.3	1.2	1.1
k	7	8	9	10	11	12	13	14	15	16	17	19	22	28	38	70

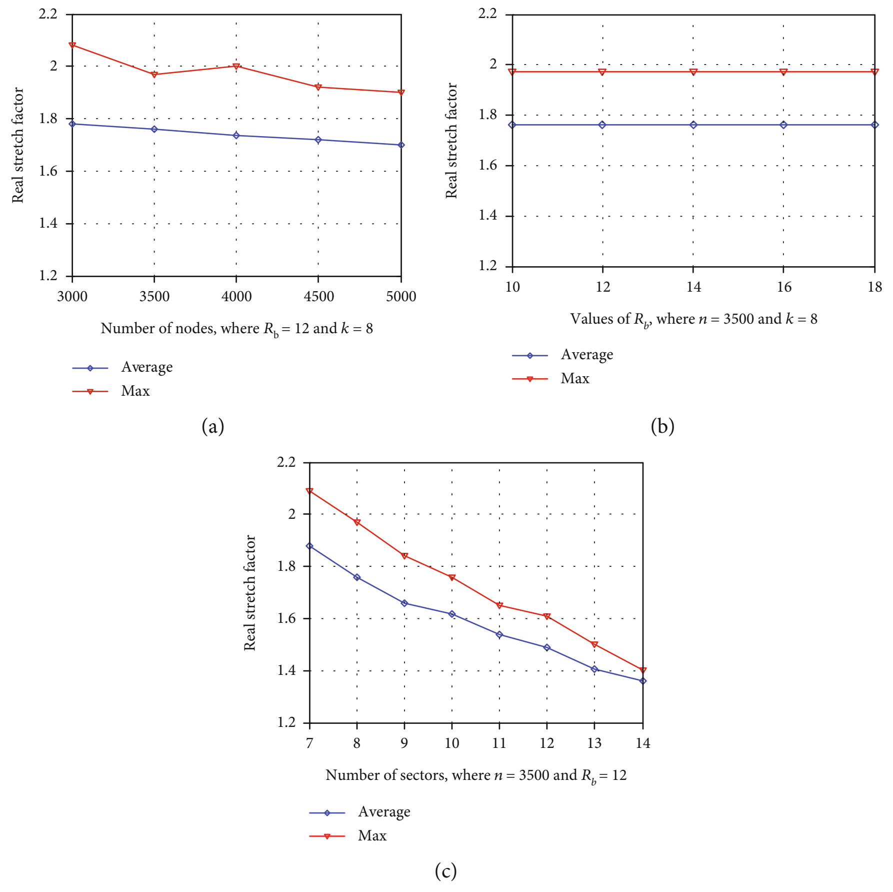

Figure 6: Influences of n, R_b, and k on the number of missing edges.

algorithm. After testing various values, no matter what values α, β, and ε have in the ranges, the variety of the performance and the delay in our algorithm is similar. Therefore, we adopted $\alpha = 5$, $\beta = 4$, and $\varepsilon = 0.8$ in our following reported result. Accordingly, the transmission power $P = N\beta \cdot (R_b/(1-\varepsilon))^\alpha$ owing to the equation (2) and (3). The setting of simulation parameters refers to [9, 19]. With the proof of Theorem 8, we know that each node runs $16\Delta^p + c$ rounds and the algorithm could get the required spanner with high probability. However, the way we adopted was that the algorithm terminates when the resultant graph does not change in 50 continuous timeslots in the simulation, i.e., in 50 continuous time slots no node can find a nearer neighbor within each sector. Over 100 runs of the simulations have been made for each reported average result.

Now, we first explore the stretch factor of the resultant graph (real stretch factor) in Figure 6. In (a), we analyze the influence of the number of nodes. The local broadcasting range was set to $R_b = 12$, and the number of sectors was set to $k = 8$. The maximum and average real stretch factor slightly reduced with the number of nodes increasing. In (b), we investigate the impact of R_b with the number of nodes $n = 3500$ and the number of sectors $k = 8$. The real stretch hardly changes with R_b. In (c), we analyze the influence of the number of sectors k with $n = 3500$ and $R_b = 12$. The maximum and average real stretch factor decrease with k increases. When $k \to \infty$, the stretch factor approaches one. From all figures in Figure 6, it can be seen that both the maximum and average real stretch factor is much smaller than the theory value in Table 1 with the same k. Hence, given the required t, we can choose a smaller k according to practical experience. In summary, the stretch performance of the algorithm SINR-Spanner is better than expected.

Due to randomization of SINR-Spanner, the constructed graphs may not be perfect $YG_k^{R_b}$; for example, the connection link to the nearest neighbor in a sector might be missing. But the algorithm performance is guaranteed with high probability, i.e., the probability for all n nodes transmitting R_b successfully and all edges in $YG_k^{R_b}$ being reserved is bounded by $1 - e^{-n/4}$. To verify this, we evaluate the number of missing edges in the resultant graph from the corresponding perfect

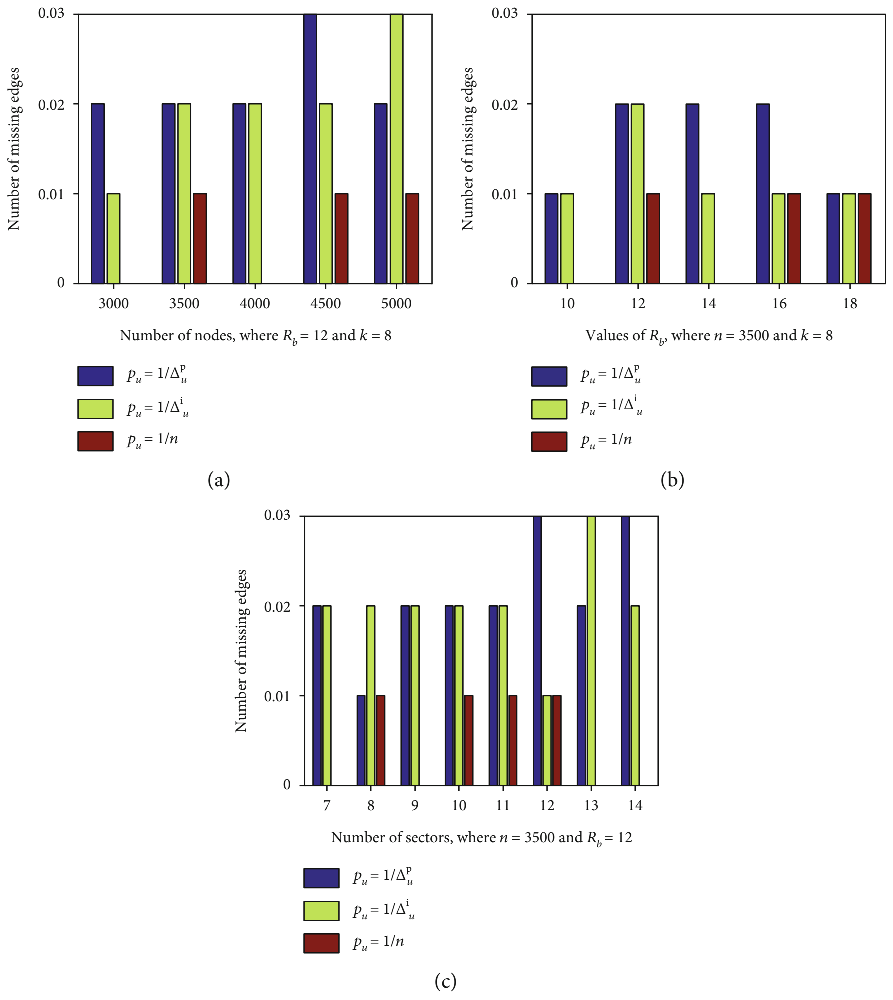

FIGURE 7: Influences of n, R_b, and k on the number of missing edges.

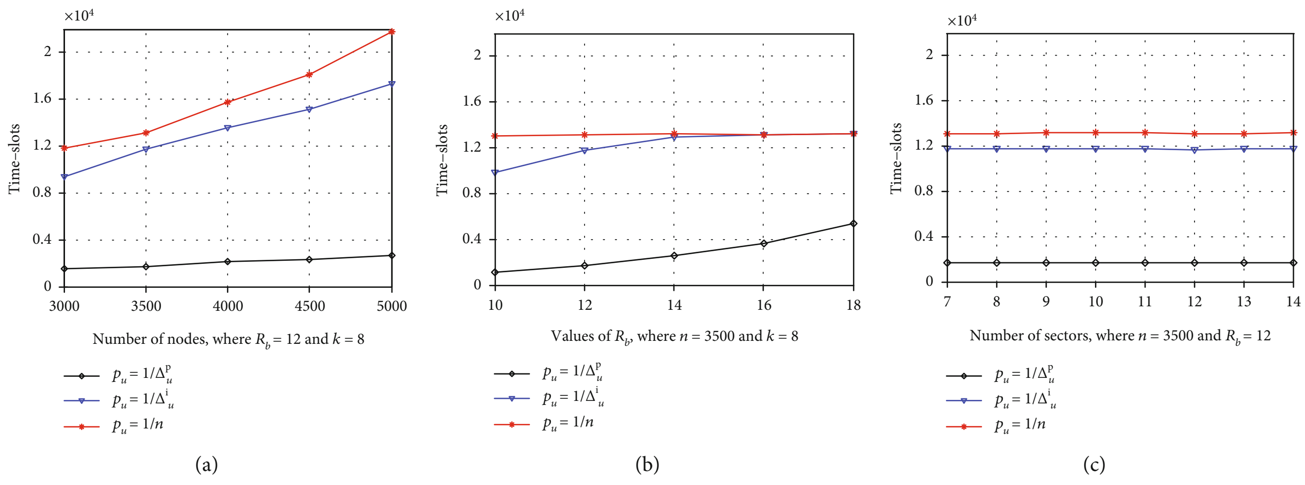

FIGURE 8: Influences of n, R_b, and k on the delay.

$YG_k^{R_b}$ in Figure 7. With a different number of nodes, different values of R_b, and different number of sectors, we compare the number of missing edges in three sending probability cases including $1/n$, $1/\Delta_u^p$ (SINR-Spanner), and $1/\Delta_u^I$ (the algorithm SINR-Undirected-YG in previous work [9]). When the sending probability is $1/\Delta_u^p$, the number of missing edges is a little bit more than the other two cases; the reason is that the nodes transmitting at the same time are more and mutual interference is a little bigger. However, the number of missing edges in all three cases is no more than three in total 100 runs, though each run has no less than 3000 nodes and $O(kn)$ edges. Simulations indicate that the similarity between the resultant graph and the corresponding $\mathrm{YG}_k^{R_b}$ with no-missing edges is close to 100% in all three cases.

We then consider the average delay needed by SINR-Spanner in Figure 8 from the influences of n in (a), R_b in (b), and k in (c), respectively. We still compare the results with three sending probability cases including $1/n$, $1/\Delta_u^p$ (SINR-Spanner), and $1/\Delta_u^I$ (the algorithm SINR-Undirected-YG in previous work [9]). Whether the sending probability is $1/n$, $1/\Delta_u^p$, or $1/\Delta_u^I$ in (a), the delay grows with the number of nodes increasing. In (b), the delay increases with R_b increasing when the sending probability is $1/\Delta_u^p$ and $1/\Delta_u^I$. However, when the sending probability is $1/n$, the delay does not change in (b) since it is only related to the number of nodes. In (c), the delay does not vary with k increasing. In summary, from Figure 8, the delay mainly changes with the change of the sending probability, while the sending probability of $1/\Delta_u^p$ or $1/\Delta_u^I$ mainly varies with n and R_b. Moreover, the average delay needed by the algorithm in [9] is close to the case where each node transmits with the probability $1/n$, and the average delay needed by SINR-Spanner is much smaller than that of previous work [9]. Finally, in theory, $16\Delta^p + c$ is the delay upper bound of the algorithm SINR-Spanner from Theorem 8 and Δ^p is the upper bound by n, while in the simulation, the delay is much smaller than $16n + c$ and the algorithm can achieve reliable performance when the algorithm terminates when the resultant graph does not change in 50 continuous time slots.

7. Summary

In this paper, we present a randomized and distributed algorithm SINR-Spanner to solve the t-spanner-SINR problem using small delay in the IoT, which has the following characteristics: (1) being a distributed algorithm, (2) considering the SINR interference model, (3) applying the YG idea, and (4) theory and simulation guaranteed. In future research, the delay performance of the spanner construction algorithm under SINR may be able to be improved by adopting a smaller proximity region. Other methods for spanner construction except YG are also worthy of investigating.

Data Availability

Our simulations are coded in the Sinalgo simulation framework [26], which is for testing and validating network algorithms and abstracts from the underlying layers. We consider a square area of 1000 by 1000 and deploy n nodes within this network region randomly and uniformly, where $n\{3000, 3500, 4000, 4500, 5000\}$.

Conflicts of Interest

The author(s) declare(s) that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants (12071478, 61972404) and the Natural Science Foundation of Shandong Province under Grants (ZR2019ZD10, F060505)

References

[1] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 144–153, Dallas, TX, USA, 2019.

[2] C. Luo, Y. Hong, D. Li, Y. Wang, W. Chen, and Q. Hu, "Maximizing network lifetime using coverage sets scheduling in wireless sensor networks," *Ad Hoc Networks*, vol. 98, article 102037, 2020.

[3] X. Zhang and J. Yu, "Spanner construction for topology control in wireless networks," *Ruan Jian Xue Bao/Journal of Software*, vol. 26, no. 4, pp. 904–926, 2015.

[4] G. Narasimhan and M. Smid, *Geometric Spanner Networks*, Cambridge University Press, New York, 2007.

[5] R. Klein and M. Kutz, "Computing geometric minimum-dilation graphs is NP-hard," in *Graph Drawing. GD 2006*, M. Kaufmann and D. Wagner, Eds., vol. 4372 of Lecture Notes in Computer Science, pp. 196–207, Springer, Berlin, Heidelberg, 2006.

[6] X. Li, P. Wan, and Y. Wang, "Power efficient and sparse spanner for wireless ad hoc networks," in *Proceedings Tenth International Conference on Computer Communications and Networks (Cat. No.01EX495)*, pp. 564–567, Scottsdale, AZ, USA, 2001.

[7] P. Von Rickenbach, R. Wattenhofer, and A. Zöllinger, "Algorithmic models of interference in wireless ad hoc and sensor networks," *IEEE/ACM Transtion on Networking*, vol. 17, no. 1, pp. 172–185, 2009.

[8] M. M. Halldórsson and T. Tonoyan, "Plain SINR is enough," in *Jul 2019 in Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, pp. 127–136, Toronto, Canada, 2019.

[9] X. Zhang, J. Yu, W. Li, X. Cheng, D. Yu, and F. Zhao, "Localized algorithms for Yao graph-based spanner construction in wireless networks under SINR," *IEEE/ACM Transaction on Networking*, vol. 25, no. 4, pp. 2459–2472, 2017.

[10] A. Yao, "On constructing minimum spanning trees ink-dimensional spaces and related problems," *SIAM Journal on Computing*, vol. 11, no. 4, pp. 721–736, 1982.

[11] I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares, "On sparse spanners of weighted graphs," *Discrete and Computational Geometry*, vol. 9, no. 1, pp. 81–100, 1993.

[12] C. Scheideler, "Overlay networks for wireless ad hoc networks," in *Wireless Communications*, P. Agrawal, P. J. Fleming, L. Zhang, D. M. Andrews, and G. Yin, Eds., vol. 143 of The IMA Volumes in Mathematics and its Applications, pp. 237–258, Springer, New York, NY, 2010.

[13] I. Kanj, "Geometric spanners: recent results and open directions," in *2013 Third International Conference on Communications and Information Technology (ICCIT)*, vol. 82, p. 78, Beirut, Lebanon, 2013.

[14] K. Kothapalli, C. Scheideler, M. Onus, and A. W. Richa, "Constant density spanners for wireless ad-hoc networks," in

Proceedings of the 17th annual ACM symposium on Parallelism in algorithms and architectures - SPAA'05, pp. 116–125, Las Vegas, NV, USA, 2005.

[15] E. Chlamtác and M. Dinitz, "Lowest-degree k-spanner: approximation and hardness," *Theory of Computing*, vol. 12, no. 1, pp. 1–29, 2016.

[16] M. A. Abam and M. S. Qafari, "Geometric spanner games," *Theoretical Computer Science*, vol. 795, pp. 398–407, 2019.

[17] P. Bose, R. Fagerberg, A. van Renssen, and S. Verdonschot, "On plane constrained bounded-degree spanners," *Algorithmica*, vol. 81, no. 4, pp. 1392–1415, 2019.

[18] B. Huang, J. Yu, X. Cheng, H. Chen, and H. Liu, "SINR based shortest link scheduling with oblivious power control in wireless networks," *Journal of Network and Computer Applications*, vol. 77, pp. 64–72, 2017.

[19] O. Goussevskaia, T. Moscibroda, and R. Wattenhofer, "Local broadcasting in the physical interference model," in *Proceedings of the 5th International Workshop on Foundations of Mobile Computing (DialM-POMC'08)*, pp. 35–44, Toronto, Canada, 2008.

[20] D. Yu, Q. Hua, Y. Wang, H. Tan, and F. C. M. Lau, "Distributed multiple-message broadcast in wireless ad-hoc networks under the SINR model," in *Structural Information and Communication Complexity. SIROCCO 2012*, G. Even and M. M. Halldórsson, Eds., vol. 7355 of Lecture Notes in Computer Science, pp. 111–122, Springer, Berlin, Heidelberg, 2012.

[21] T. Jurdzinski, D. R. Kowalski, M. Rozanski, and G. Stachowiak, "On setting-up asynchronous ad hoc wireless networks," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 2191–2199, Kowloon, Hong Kong, 2015.

[22] M. M. Halldórsson, Y. Wang, and D. Yu, "Leveraging multiple channels in ad hoc networks," in *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*, pp. 431–440, Donostia-San Sebastián, Spain, 2015.

[23] F. Fuchs and R. Prutkin, "Simple distributed $\Delta + 1$ coloring in the SINR model," in *Structural Information and Communication Complexity. SIROCCO 2015*, C. Scheideler, Ed., vol. 9439 of Lecture Notes in Computer Science, pp. 149–163, Springer, Cham, 2015.

[24] L. Fu, S. C. Liew, and J. Huang, "Effective carrier sensing in CSMA networks under cumulative interference," in *2010 Proceedings IEEE INFOCOM*, pp. 1–9, San Diego, CA, USA, 2010.

[25] A. Blum, J. Hopcroft, and R. Kannan, *Foundations of Data Science*, Cambridge University Press, New York, 2020.

[26] Distributed Computing Group, ETH Zurich, "Sinalgo - simulator for network algorithms, version 0.75.3. 2008," http://sourceforge.net/projects/.