Communication costs in a geometric communication network

Sima Hajiaghaei Shanjani, & Valerie King

2023

Theoretical Computer Science 977 (2023) 114161

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Communication costs in a geometric communication network ☆,☆☆,☆☆☆

Sima Hajiaghaei Shanjani *, Valerie King

Department of Computer science, University of Victoria, Victoria, BC, Canada

A R T I C L E   I N F O

Article history:
Received 16 August 2021
Received in revised form 29 June 2023
Accepted 25 August 2023
Available online 1 September 2023

Keywords:
Distributed algorithm
CONGEST
Communication complexity
Farthest pair
Convex hull
Closest pair

A B S T R A C T

We represent a communication network as a graph in which each node has only local information about the graph, except for an upper bound on the number of nodes, and nodes communicate by passing messages along its edges. Here, we consider a *geometric communication network* where the nodes also occupy points in space and the distance between points is the Euclidean distance. Our goal is to understand the communication cost needed to solve several fundamental geometry problems, including Farthest Pair, Convex Hull, Closest Pair, and approximations of these problems, in the asynchronous CONGEST KT1 model, where each node knows its ID and those of its neighbors. This extends the 2011 result of Rajsbaum and Urrutia for finding a convex hull of a planar geometric communication network to networks of arbitrary topology.

We define a new model where each node has a position on the plane and nodes can communicate to each other if and only if there is an edge between them. We motivate the model and study a number of geometric problems in this model. We prove lower bounds on the communication complexity of the problems in this new model and present approximation algorithms for them. We prove lower bounds on the number of expected bits required for any randomized algorithm to solve the problems.

1. Introduction

The communications network is a formal way to model communication in distributed systems with an arbitrary topology. A communication network is a graph in which each node has only local information about the graph and the nodes communicate by passing messages along its edges. More formally, the communication network is a connected graph $G = (V, E)$, where $|V| = n$, $|E| = m$, and each node in V represents a processor. Each edge in E represents a two-way communication link. Two nodes can communicate directly with each other only if there is an edge between them. Here, we consider the *geometric communication network* where each node of the communication network occupies a point on the plane and the distance between points is the Euclidean distance. The goal is to study the communication complexity of fundamental computational geometry problems in this setting.

Our paper extends the work of Rajsbaum and Urrutia [34], the only paper known to the authors which addresses the communication complexity of a similar geometric problem in a geometric communication network. That paper considers

☆ This article belongs to Section A: Algorithms, automata, complexity and games, Edited by Paul Spirakis.

☆☆ This is an extended and revised version of a preliminary conference report that was presented in ICDCN 2021 [18].

☆☆☆ This work was supported by NSERC Discovery Grant RGPIN 2016-04234.

* Corresponding author.
E-mail addresses: sima@uvic.ca (S. Hajiaghaei Shanjani), val@uvic.ca (V. King).

https://doi.org/10.1016/j.tcs.2023.114161

the problems of finding a convex hull and external face in an asynchronous planar network. Here we consider networks with arbitrary topology.

Variants of communication networks have been studied which are distinguished by the types of messages, the existence of a global clock (synchronous vs. asynchronous), and the amount of local knowledge known to the nodes. See [6,32]. Unless otherwise specified, we assume the CONGEST KT1 asynchronous model [30], which we extend to a geometric model. That is, each node may send (possibly different) messages of size $O(\log n)$ bits to a subset of its neighbors at the same time. Each node knows its position on the plane and its unique ID and the IDs of its neighbors in the graph ("KT1") (but not their positions). A motivation for the latter assumption is that between computations, the nodes may move, but their communication links are fixed. We assume there is no global clock. Each ID and each position is assumed to be specifiable with $O(\log n)$ bits, so that all points lie on an $[n^c] \times [n^c]$ grid for c a constant. Note that the points lie on a grid, but the edges between the points are of arbitrary topology and the graph may not be planar.

Our lower bounds on communication are independent of time and therefore require the assumption of asynchronicity (see Section 3.1).

We consider the following problems defined on a geometric communication network. For Farthest Pair and Closest Pair, we require that every node learns the solution, which is the positions of the pair of points. In Convex Hull, we require that each node learns whether it is on the convex hull or not, and if so, the position of both its neighbors in the convex hull. In all of these problems, the distance between points means the Euclidean distance between them and not the hop distance in the graph; the solution to a problem is independent of the topology of the communication network.

- **Farthest Pair:** Determine a pair of points furthest from each other, as measured by Euclidean distance.
- **Convex Hull:** Compute the convex hull of the geometric communication network, which is the smallest convex shape that contains all the points.
- **Closest Pair:** Determine a pair of closest points, as measured by Euclidean distance.

We give algorithms for exact and approximate solutions of these problems and prove lower bounds for these. We show that the exact Farthest Pair, Convex Hull, and Closest Pair problems require $\Omega(n^2)$ bits of communication. In Farthest Pair and Closest Pair this lower bound holds even if the network is planar. Our randomized algorithms approximate solutions to these problems using $\tilde{O}(n^{3/2})$ bits of communication, and they succeed with high probability.

The algorithms for approximating Farthest Pair and Convex Hull first find a small set of representative points which is an ϵ-kernel of the points, and use this to solve the problem. This method has been used in approximation algorithms for these problems and others. We give approximation algorithms for ϵ-kernel in Section 2.2.

- ϵ**-kernel**: Determine an ϵ*-kernel* which is a subset of points whose convex hull approximates the width of the original set in any direction.

In addition, we formulate the following simple problem as a building block for our lower bounds, for any function $f(x, y)$:

- **Path Computation of** $f(x, y)$
  Given a communication network which is the path $P = (Alice = v_0, v_1, v_2, ..., v_m = Bob)$, assume initially, Alice knows x and Bob knows y. After exchanging messages, all nodes know $f(x, y)$.

1.1. Model details and definitions

In our model, the geometric communication network is a connected graph $G = (V, E)$, where $|V| = n$. Each node in V is located at a distinct point specified by (p_x, p_y) where p_x, p_y are integers between 1 and n^c, and c is a constant. Each node has a unique ID of $O(\log n)$ bits which is given by a name that may be independent of its position. Each node knows its own ID and position. In addition, we assume each node knows an upper bound of n^c, for c a constant, on the number of nodes in the network. This constraint is needed to construct a spanning tree (see Subsection 2.1).

We use standard terminology from [32] to describe message size and local knowledge: In the CONGEST model, each node may send a message of size $b = O(\log n)$ to every neighbor in the same time step. In the CONGEST KT1 model, each node starts with the knowledge of its neighbors' IDs. In the CONGEST KT0 model, each node has a dedicated port to each of its neighbors, but is unaware of its neighbors' IDs. We write CONGEST-1 when we assume the message size is one bit.

In an *asynchronous* network, there is no global clock. Delays between sending and receiving any message may be adversarily set, though all messages sent eventually arrive. Aside from an initial wake-up, actions are event-driven. We assume all nodes wake up at the start of the algorithm. Time in the asynchronous model is defined by the length of the longest chain of events (receipt of a message) in which each event is waiting for the previous event to occur.

Nodes in the geometric network have private randomness, that is, coins which they may flip to decide on their actions. The outcomes of a node's coinflips are not known to other nodes. The input to the algorithm is set before any random bits are known. The delay in transmitting a message may depend on the contents of all messages sent so far, including that message, but are independent of the outcomes of coin flips which have not yet been executed.

Table 1
Results of our work: upper bound and lower bound of the number of messages of size $O(\log n)$ bits for a geometric communication network with n nodes whose positions lie on an $[n^c] \times [n^c]$ grid. $\mathcal{ST}$ is the cost of computing a spanning tree.

	Upper Bound	Lower Bound
Path Computation of Disjointness of size n over a path of $O(n)$ nodes	$O(n^2/\log n)$	$\Omega(n^2/\log n)$
ϵ-kernel	$O(n/\sqrt{\epsilon}) + \mathcal{ST}$	$\Omega(min\{n^2, 1/\epsilon\})$
Farthest Pair	$O(n^2)$	$\Omega(n^2/\log n)$
$(1-\epsilon)$-approx. Farthest Pair	$O(n/\sqrt{\epsilon}) + \mathcal{ST}$	$\Omega(min\{n^2, 1/\epsilon\})$
Convex Hull	$O(n^2)$	$\Omega(n^2/\log n)$
ϵ-approx. Convex Hull	$O(n/\sqrt{\epsilon}) + \mathcal{ST}$	-
Closest Pair	$O(n^2)$	$\Omega(n^2/\log n)$
$\frac{n^c}{\sqrt{(n-1)/2}}$-approx. Closest Pair*	$O(n \log n) + \mathcal{ST}$	-
$\frac{n^{c-1/2}}{2}$-approx. Closest Pair **	-	$\Omega(n^2/\log n)$

* for a constant $c > 1/2$. ** for a constant $c > 1 + 1/(2\lg n)$.

The communication complexity of an algorithm (also referred to as a "protocol") is measured by the expected number of bits or messages used over the worst case input and delays. The complexity of a problem is the minimum complexity over all algorithms which correctly solve the problem.

An algorithm for a minimization problem has the *approximation ratio* (factor) ρ, for some $\rho > 1$, if for every input I, the algorithm's solution is feasible and the value of the solution is at most $\rho \cdot OPT(I)$, where $OPT(I)$ is the value of the optimal solution of the problem for input I. An algorithm for a maximization problem has the approximation ratio of ρ, for some $\rho < 1$, if the value of the algorithm's solution for every input I is at least $\rho \cdot OPT(I)$. An approximation algorithm is called a ρ-approximation if it has the approximation factor ρ. Approximate convex hull may be defined in different ways; we use $\epsilon - hull$. An ϵ-hull of a set of points P is a subset of the points such that every point in P is within a distance of ϵ from the convex hull of the subset, see Section 2.4 [8].

1.2. Techniques

There is a simple algorithm to solve any problem on the network where each node needs to learn only $O(\log n)$ bits of information. Once a spanning tree is computed, it suffices to send the location of all points to a central location which then computes the answer and sends each node the information required. Hence any problem where each node needs to learn an $O(\log n)$ bit solution can be solved with $O(n^2)$ messages of size $O(\log n)$ in the weakest model, asynchronous CONGEST KT0, as it is easy to see that a spanning tree can be computed in $O(m)$ communication and time $O(n)$. We prove these bounds on communication are optimal for the problems examined here within a $\log n$ factor.

Our algorithms which run in $o(m)$ communication require the initial randomized computation of a spanning tree in $o(m)$ communication, using the algorithms of Mashreghi and King, [30,31], see Section 2.1. The other parts of the algorithms are deterministic and can be done in the KT0 model.

After a spanning tree is found, it is used to communicate and summarize information held by the nodes to the root. For example, for the ϵ-kernel computation, a root of a subtree of the spanning tree passes on to its parent an ϵ-kernel of the points in the subtree. The algorithms for approximating Farthest Pair and Convex Hull find an ϵ-kernel of the points which approximates the solutions to these problems. The approximate Closest Pair algorithm uses a binary search over the possible locations to locate two points close together.

Our lower bounds hold for the randomized asynchronous KT1 model (and therefore in the KT0 model). The Set Disjointness problem is a well-studied problem in the literature on two-party communication. We reduce it to our problems to prove nearly matching lower bounds for these approximation results, and as well as the lower bounds for the exact problems. Essentially, we show that each of our problems embeds a problem we call Path Communication. A protocol for this in the asynchronous CONGEST KT1 model can be used to solve Set Disjointness in the randomized two-party communication model.

1.3. Results

Table 1 shows the specific results regarding communication which is the main contribution of this work. There, $\mathcal{ST}$ denotes the communication bounds for computing a spanning tree. Running times for each problem are given with the algorithms for each problem.

The lower bounds on the Approximate Closet Pair are related to the size of the grid. For example when $c = 1.1$, any randomized asynchronous distributed algorithm for approximating Closest Pair in a graph with $n > 32$ nodes on an $[n^{1.1}] \times [n^{1.1}]$ grid in the CONGEST KT1 model within a $\frac{n^{0.6}}{2}$ factor of the optimum requires an expected $\Omega(n^2/\log n)$ messages of size $O(\log n)$. However, there exists an asynchronous algorithm in CONGEST KT1 which computes a $2n^{0.6}$-approximation for this problem with high probability using $O(n \log n) + \mathcal{ST}$ messages of size $O(\log n)$.

1.4. Related work

For a long time, it was believed that $\Omega(m)$ bits of communication were required to build a spanning tree in a distributed message-passing model. This was first proved in the CONGEST KT1 model where each message could depend on only a constant number of node IDs ("atomic" model) or the node ID space is exceedingly large, beyond polynomial in size, in 1990 [6]. The same lower bound was also shown in the general CONGEST KT0 model [26,27].

The first algorithmic breakthrough came in the synchronous model in 2015 where an algorithm to find a minimum spanning tree (and spanning tree) with $\tilde{O}(n)$ messages in the CONGEST KT1 model was shown [24]. This was followed by another synchronous algorithm with $\tilde{O}(n^{3/2})$ messages with the optimal time steps $\Theta(D + \sqrt{n})$ in 2018 [14,15]. Gmyr and Pandurangan explore other tradeoffs between time and messages for this problem in the synchronous model [15]. In the asynchronous model, the first communication algorithm which required $o(m)$ communication in dense networks, were given by Mashreghi and King, [30,31]. This algorithm, which uses $\tilde{O}(n^{3/2})$ communication is an essential building block for our algorithms. All these techniques use messages, each of which can depend on a large number of $ID's$ and thus are able to avoid the lower bounds of [6].

A natural question to ask is what other interesting graph problems can be solved in $o(m)$ communication and polynomial time in a message-passing network. Recently, Robinson proved time-communication tradeoffs in the synchronous CONGEST KT1 model for the problem of computing a graph spanner [36]. They show a lower bound on the communication complexity of computing a $(2t-1)$-spanner that consists of at most $O(n^{1+\frac{1}{t}+\epsilon})$ edges, where $\epsilon = \theta(\frac{1}{t^2})$. Any $O(poly(n))$-time algorithm requires at least $\tilde{\Omega}(\frac{1}{t^2} n^{1+\frac{1}{2t}})$ bits of communication in the CONGEST KT1 model. Prior to this work only a trivial lower bound of $\tilde{\Omega}(n)$ bits was known. Comparing this result with the result of [24] shows that computing a minimum spanning tree can be done with less communication than finding a spanner.

A reduction of two-party Set Disjointness has been used to prove lower bounds on time in the CONGEST KT1 model, for example, on verification problems on graphs, by Das Sarma et al. [37,38], and on graph diameter problem by Amir Abboud et al. [1] and Ofer Grossman et al. [17]. In the synchronous network, any two-party decision problem can be solved with a constant amount of communication and sufficient time, for the whole process; the first party communicates its entire input of n bits by sending a single bit at a round whose number is equal to the (up to 2^n) value of its input. A synchronous model for two-party communication with a global clock was introduced by Impagliazzo and Williams where time-communication tradeoffs are given [20].

Our work extends the work of Rajsbaum and Urrutia [34]. In their model edges are straight line segments connecting pairs of vertices such that no two of them intersect. Their model is a fixed-position model, where the ID of each node is equal to the name of the point on the plane. The model used in [34] is asynchronous KT0, and each message may contain information about only a constant number of node IDs (atomic model). They obtain an $O(n \log^2 n)$ message complexity algorithm to find the convex hull, and an $O(n \log n)$ message complexity algorithm to identify the external face of a geometric planar network with n nodes. They present a matching lower bound for the external face problem. They also prove that the message complexity of leader election in a geometric ring is $\Omega(n \log n)$.

There are many distributed models for dynamic networks. For example, in mobile *ad-hoc* Networks and *Wireless Networks*, the communication graph is often a Unit Disk Graph, in which two nodes are connected if their distance is at most one unit. A *Hybrid* ad hoc network models the communication in two layers: one for local communication, similar to ad-hoc networks, and one for sending a direct message to any node but with more cost. For example, a set of mobile nodes may each have a cell phone and a wireless device. Nodes can send messages via wireless communication if a node is within a range, otherwise they can call the other node. Jung et al. in [22] model a cell phone network as a hybrid directed graph, in which each node can move and has a unique phone number (ID) and there is a Unit Disk Graph to model local communication using their wifi interface. A node u can send a message to node v if u is within distance one from v, or if u learns the phone number (ID) of v. Many distributed and geometric problems have been studied in these models, such as routing, broadcasting, minimum spanning tree, and spanners [9,29,28,13,39], where the main concern is time complexity.

An ϵ*-kernel* is a type of coreset. For a given problem and an input point set, a coreset is a small subset of the points that the solution of the problem to the coreset approximates the solution of the problem to the whole point set [3]. Coresets are specifically useful in distributed settings as coresets can be built in parallel. Feldman and Sugaya designed an effective coreset algorithm by merging coresets in *Sensor Networks* for a version of the k-line center problem in the parallel setting, where they ignore the communication cost [12]. Balcan et al. proposed distributed algorithms for k-means and k-median, where each node computes a portion of the coreset similar to our algorithm [7]. There, the data is distributed across nodes whose communication is restricted to the edges of an arbitrary graph. The input of each node is a set of points, and the goal is to find k-means and k-median of the union of the input points of all the nodes. That paper proposes algorithms for finding small coresets in order to reduce communication cost.

Much effort has been done to find a fast algorithm that computes a small ϵ-kernel in the sequential and streaming models, see e.g. [5,40,3,2,10]. The algorithm proposed by Arya and Chan uses $O((1/\epsilon)^{d/2+O(1)})$ space for computing ϵ-kernel of a point set in d dimensions [5]. There is also extensive work for approximating Convex Hull in the streaming model [8,19]. The first one-pass streaming algorithm for computing ϵ-hull of a point set on the plane was proposed by Hershberger and Suri, and this algorithm uses $O(1/\sqrt{\epsilon})$ space [19].

A *composable coreset* has the property that for a collection of sets, the approximate solution to the union of the sets in the collection can be obtained given the union of the composable coresets. Composable coresets can be used to obtain efficient solutions in the streaming and distributed models [2,21]:

- Streaming: If we have a composable coreset of size k, then we divide the data stream of n elements to $\sqrt{n/k}$ blocks each of size $\sqrt{nk}$. The algorithm finds the coreset of each block and stores the coresets separately until the end of the algorithm and solves the problem for the union of the coresets, or the algorithm might do a compression, in which the coresets get combined during the algorithm to reduce the amount of space, but this might increase the approximation factor.
- Distributed: Each processor finds a coreset of its own part of the data and sends it to a server, and the server solves the problem for the union of the coresets. Alternatively, the algorithm can have a compression step in which processors can send their data or coreset to a local server, and the local server combines the coresets.

Similar ideas have been used to show lower bounds in these two models, as it is common in both of these models to use reductions from the communication complexity problems. However, to the best of our knowledge no general relation has been shown between these two models and we cannot necessarily interpret a lower bound in one model in another one. In this research we show lower bounds for problems whose lower bounds are shown in the streaming model. Feigenbaum et al. show that the diameter and closest pair problems require $\Omega(n)$ bits space in the streaming model even if we are allowed to have multiple passes through the input, where n is the number of the input data and each input can be described in $O(\log n)$ bits [11]. They also show that a version of the convex hull problem in which the input point set is guaranteed to admit a convex hull with at most k sides, *k-promised convex hull*, requires $\Omega(n)$ space in the one-pass streaming model. They show these results by reductions from two well-known communication complexity problems, Set Disjointness and Index.

1.5. Organization of the paper

In Section 2, we give approximation algorithms for ϵ-kernel, Farthest Pair, Convex Hull, and Closest Pair. In Section 3, first we discuss the Path Computation problem and prove a lower bound for this problem, and then we present lower bounds for Farthest Pair, Convex Hull, and Closest Pair.

2. Algorithms

First we discuss known Spanning Tree results in Section 2.1 and solve ϵ-kernel in Section 2.2, then use them as a building block for approximating Farthest Pair, Convex Hull, and Closet Pair.

2.1. Spanning trees

Here, we review what is known concerning the spanning tree problem in a communications network, as we use these results extensively.

In a distributed network, the spanning tree problem begins with each node having local knowledge of the graph. Using an exchange of messages, a spanning tree is determined in that each node knows which of its incident edges are in it.

A Monte Carlo algorithm is a randomized algorithm whose output may be incorrect with a certain probability, and we say the algorithm succeeds with high probability if the probability of success is more than $1 - \frac{1}{k^c}$ for any given constant $c > 0$, where k is the size of the input. For the following, let n be the number of nodes and m be the number of edges in the network. The Monte Carlo algorithms described below require that the nodes start with the knowledge of an upper bound on the number of nodes, $n^{c'} \geq n$ where c' is a constant.

- **Theorem 2.1** *([30]). In CONGEST KT1 where all nodes awake at the start, a spanning tree can be built by a randomized asynchronous Monte Carlo with communication $\tilde{O}(\min\{n^{3/2}, m\})$ and high probability of success.*

- In CONGEST KT1, a spanning tree can be built by a randomized synchronous Monte Carlo with communication and time $\tilde{O}(n)$ and high probability of success [24].
- In CONGEST KT0, the expected number of messages required for any randomized algorithm to compute a spanning tree in the synchronous (and asynchronous) model with probability at least 53/56 is $\Omega(m)$ [26,27].

With the exception of the subroutine to build the spanning tree, our algorithms will work in the weakest model described in this paper, the asynchronous CONGEST KT0 model. We state the communication cost of each of our algorithms as a function of the communication needed to build the spanning tree on n nodes plus the communication needed for the other part of the algorithm. We use the term $diam(\mathcal{ST})$ to refer to the hop-diameter of the spanning tree.

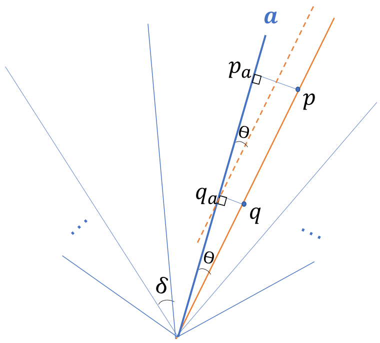

Fig. 1. ϵ- kernel proof.

2.2. ϵ-Kernel

We present the ϵ-kernel problem, which we use as a subroutine to compute approximations for Farthest Pair and Convex Hull.

Let $\mathcal{S}$ denote the unit circle centered at the origin in $\mathcal{R}^2$. We may view a point $u \in \mathcal{S}$ as being a vector from the origin to u and call it a *direction*. Given a set of points $P \in \mathcal{R}^2$, the *directional width of* P *in direction* u is denoted by $w(u, P) = max_{p \in P} < u, p > -min_{p \in P} < u, p >$, where $< ., . >$ is the standard inner product. An ϵ-kernel of P is a subset of P which approximates P with respect to directional width. More formally, a subset $Q \subset P$ is called an ϵ-*kernel* of P if for each $u \in \mathcal{S}$, $(1-\epsilon)w(u, P) \leq w(u, Q)$ [2,3].

Algorithm 1 computes an ϵ-kernel of the set of points in $\mathcal{R}^2$ by using the idea of rounding directions, which was introduced in [4].

Let $loc(p)$ denote the position of node p. Let $0 < \epsilon < 1$ and $\delta = \sqrt{2\epsilon}$. In Algorithm 1, A is the set of lines passing through $(0, 0)$ which, with the x-axis, make angles δi for $i = 1, ..., \lceil \pi/\delta \rceil$, and $|A| = \Theta(1/\sqrt{\epsilon})$. Given a set of points P and a line a through the origin, the *extreme points for the line* a, denoted by $extreme(P, a)$, are the pair of points r and s in P whose projections onto the a are farthest apart on the line: r is a point such that $< r, a >= max_{p \in P} < a, p >$ and s is a point such that $< s, a >= min_{p \in P} < a, p >$, where $< ., . >$ is the standard inner product.

A spanning tree $\mathcal{ST}$ is first computed by Theorem 2.1. Starting with the leaves of $\mathcal{ST}$, each node compares its own location with the location of the extreme points received from its children (if it is not a leaf) to find the best two candidates for the extreme points for each direction, and sends them to its parent, until eventually the root receives all the extreme points. The set S_L of these extreme points is an ϵ-kernel for this set of points.

Algorithm 1 Asynchronous ϵ-Kernel.

procedure ϵ-KERNEL(P is a set of n nodes)
  Find a spanning tree $\mathcal{ST}$ with leader L.
  L broadcasts $< start >$ when $\mathcal{ST}$ is complete.
  When a leaf p receives $< start >$, it sends $S_p = \{loc(p)\}$ to its parent.
  When a non-leaf node $p \neq L$ receives S_q from all of its children q. Let $P' = \bigcup_q S_q \cup \{loc(p)\}$. Then p computes the set $S_p = \bigcup_{a \in A} extreme(P', a)$ and passes S_p to its parent.
  When the leader L receives S_q from all of its children q, L computes the set S_L of extreme points for all lines in A among the point set $\bigcup_q S_q \cup \{loc(L)\}$.
end procedure

Theorem 2.2. *S_L is an ϵ-kernel for the point set P.*

Proof. Given any pair of points p, q, let $d(p, q)$ denote the distance between points p and q, and let p_a denote the projection of point p onto line a. Let $\mathcal{S}$ denote the unit sphere centered at the origin in $\mathcal{R}^2$. We show that for any direction $a^* \in \mathcal{S}$, there is a direction $a \in A$ such that for any $p, q \in P$,

$$(1-\epsilon)d(p_{a^*}, q_{a^*}) \leq d(p_a, q_a)$$

It is sufficient to show $(1-\epsilon)d(p, q) \leq d(p_a, q_a)$, as $d(p_{a^*}, q_{a^*}) \leq d(p, q)$.

Let $\overrightarrow{pq}$ denote the line passing through p and q, and let $\theta(p, q, a)$ be the acute angle formed by $\overrightarrow{pq}$ and a. Then it is not hard to observe (see Fig. 1):

$$d(p_a, q_a) \leq d(p, q) = d(p_a, q_a)/cos(\theta(p, q, a))$$

As there is a line forming an angle with the x-coordinate at every δ angle interval, there must be some line $a \in A$ forming an angle $\leq \delta$ with $\overrightarrow{pq}$.

Finally, the Taylor series for $\cos(\delta)$ implies $cos(\delta) \geq 1 - \delta^2/2$. In particular, for $\delta = \sqrt{2\epsilon}$, then $cos(\delta) \geq 1 - \epsilon$. $\square$

Theorem 2.3. *There exists an asynchronous algorithm in CONGEST KT1 which computes an ϵ-kernel with high probability in a graph of n nodes with $O(\frac{n}{\sqrt{\epsilon}})$ messages in time $O(\frac{diam(\mathcal{ST})}{\sqrt{\epsilon}})$ plus the costs of computing $\mathcal{ST}$.*

Proof. In Algorithm 1 each node passes up no more than $2|A| = O(1/\sqrt{\epsilon})$ locations using $O(diam(\mathcal{ST})/\sqrt{\epsilon})$ time for a total of $O(n/\sqrt{\epsilon})$ words of size $O(\lg n)$. In addition, there is the initial communication cost of computing the spanning tree. By Theorem 2.2, at the end of this algorithm S_L is an ϵ-kernel for the point set P. $\square$

2.3. Approximate farthest pair

Having an ϵ-kernel provides approximation algorithms for many related problems, such as minimum enclosing ball, minimum enclosing box, diameter, and width. Here we propose an ϵ-kernel based approximation algorithm for Farthest Pair that gives us a tradeoff between the approximation ratio and the amount of communication. Then in Section 3.2 we prove lower bounds for this problem.

Algorithm 2 Asynchronous $(1-\epsilon)$-Approximate Farthest Pair.

```
procedure ϵ-FARTHEST PAIR(P is a set of n nodes)
    Find a spanning tree ST with leader L and an ϵ-kernel S.
    L computes the two farthest points r and s in S and broadcasts the position of r and s through ST.
end procedure
```

Theorem 2.4. *There exists an asynchronous algorithm in CONGEST KT1 which computes a $(1-\epsilon)$-Approximate Farthest Pair in a graph of n nodes with high probability using $O(\frac{n}{\sqrt{\epsilon}})$ messages in time $O(\frac{diam(\mathcal{ST})}{\sqrt{\epsilon}})$ plus the costs of computing $\mathcal{ST}$.*

Proof. In Algorithm 2, once the ϵ-kernel S is received by the leader, the leader finds the two points furthest from each other in S and broadcasts them. The Farthest Pair of S is a $(1-\epsilon)$-approximation for the Farthest Pair of P by the definition of ϵ-kernel. More formally, let p_1 and p_2 be the furthest points in P based on Euclidean distance. Assume p_1 and p_2 are extreme points of P for some direction $a^* \in \mathcal{S}^{d-1}$. As S is an ϵ-kernel of P, then for each $u \in \mathcal{S}^{d-1}$ including a^*, $(1-\epsilon)w(u,P) \leq w(u,S)$, where $w(u,P)$ is the directional width of P in direction u: $w(u,P) = max_{p\in P} < u, p > -min_{p\in P} < u, p >$.

The number of messages in this algorithm is the number of messages we need to compute ϵ-kernel plus $O(n)$ messages to broadcast the two farthest points. $\square$

2.4. Approximate convex hull

There are different notions of *approximate convex hull*. An ϵ-hull is one of the commonly used notions and it is closely related to ϵ-kernel.

Let P be a set of n points in $\mathcal{R}^d$. Let $C(P)$ denote the convex hull of P. An *ϵ-hull* S is a subset of P such that all the points in P are either in $C(S)$ or within distance ϵ from $C(S)$.

If S is an ϵ'-kernel of P then the convex hull of S is an $(\epsilon' \cdot \Delta(P))$-hull where $\Delta(P)$ is the distance between the farthest pair of P [3]. If we obtain an ϵ-approximation to $\Delta(P)$, call it D', then the ϵ'-kernel of P for $\epsilon' = \epsilon(1-\epsilon)/D'$ is an ϵ-hull of P.

Algorithm 3 Asynchronous Approximate Convex Hull.

```
procedure ϵ-CONVEX HULL(P is a set of n nodes)
    Find spanning tree ST with leader L and ϵ-kernel S.
    L broadcasts the vertices of convex hull of S through ST.
end procedure
```

Theorem 2.5. *There exists an asynchronous algorithm in CONGEST KT1 which computes an $f(\epsilon)-$ hull with high probability in a graph of n nodes with $O(\frac{n}{\sqrt{\epsilon}})$ messages in time $O(\frac{diam(\mathcal{ST})}{\sqrt{\epsilon}})$ plus the costs of computing $\mathcal{ST}$, where $f(\epsilon) = \epsilon \cdot \Delta(P)$.*

Proof. In Algorithm 3, once the ϵ-kernel S is received by the leader, the leader finds the convex hull of S and broadcasts its vertices, so that every node knows whether it is on the convex hull and knows its neighbors in both the clockwise and the counter-clockwise directions. The ϵ-kernel is a multiplicative error version of ϵ-hull [3]. By definition, ϵ-kernel S approximates the convex hull of P within a $1-\epsilon$ factor in any direction.

The convex hull of S is an $\epsilon \cdot \Delta(P)$-hull, as all the points in P are either in the convex hull of S or within $\epsilon \cdot \Delta(P)$ from the convex hull of S, where $\Delta(P)$ is the distance between the farthest pair of P. This is because if $d(x, S)$ is the distance of any point $x \in P \setminus S$ from convex hull S, then $d(x, S) \leq w(u, p) - w(u, S) \leq \epsilon w(u, P)$ for any direction $u \in \mathcal{S}^{d-1}$, where $w(u, P)$ is the directional width of P in direction u. Thus $d(x, S) \leq \epsilon \cdot \Delta(P)$, and the ϵ-kernel is an $\epsilon \cdot \Delta(P)$-hull. $\square$

2.5. Approximate closest pair

Here we present an approximation algorithm that gives us a tradeoff between the approximation ratio and the amount of communication. Then in Section 3.4, we show that any distributed algorithm for solving Closest Pair in a graph of n nodes on an $[n^c] \times [n^c]$ grid requires $\Omega(n^2)$ expected bits of communication to approximate within a $\frac{n^{c-1/2}}{4}$ factor of the optimum, for a constant $c > 1 + 1/(2\lg n)$.

Algorithm 4 computes an $\frac{n^c}{\sqrt{\frac{n-1}{2}}}$- Approximation Closest Pair with high probability in a graph of n nodes on an $[n^c] \times [n^c]$ grid, for a constant $c > 1/2$, in the asynchronous CONGEST KT1 model.

The main idea of these algorithms is to divide the $[n^c] \times [n^c]$ grid to a $\sqrt{n}$ by $\sqrt{n-1}$ grid, for a total of $\sqrt{n^2-n}$ grid cells: $C = c_1, c_2, \ldots, c_{\sqrt{n^2-n}}$. There are n nodes and fewer than n cells, so there is a cell with more than one node. The algorithm finds such a cell and broadcasts the nodes in it.

A spanning tree is first constructed by Theorem 2.1. Then, we use binary search on $c_1, c_2, \ldots, c_{\sqrt{n^2-n}}$ to find a cell with more than two points in $\lg(\sqrt{n^2-n})$ phases, where each phase involves a communication down the tree and back up. Let $loc(p)$ denote the position of node p and $index(p)$ denote the index of the cell in C that contains $loc(p)$. Let $mid(s, e) = \lfloor \frac{s+e}{2} \rfloor$ for any s and e.

The leader L broadcasts $< start, s_i, e_i >$ in phase i for an interval $I_i = [s_i, e_i]$ of the indices, where s_i is the index of the starting cell and e_i is the index of the ending cell to be searched. In the first phase $s_1 = 1$ and $e_1 = \sqrt{n^2-n}$. The goal is to find the number of nodes which are in cells whose indices are in the first half of the interval I_i. Starting with the leaves of $\mathcal{ST}$, each node p computes x_p, the number of nodes in the subtree rooted at p which are in cells whose indices are in $[s_i, mid(s_i, e_i)]$, and sends $< s_i, e_i, x_p >$ to its parent. When the leader L receives messages for a specific I_i from all of its children, it computes x_L and picks I_{i+1} to be the half of the interval that has more nodes than the number of cells, and broadcasts I_{i+1}. When the leader receives information about the last two cells, it picks the cell ce that contains at least two points and broadcasts $< cell, index(ce) >$ to find two points in ce. When a node p receives $< cell, index(ce) >$ and $p \in ce$, it sends $< decision, loc(p) >$ to inform the leader about its location.

Algorithm 4 Asynchronous $\frac{n^c}{\sqrt{\frac{n-1}{2}}}$-Approximation Closest Pair.

1: **procedure** Closest Pair(P is a set of n nodes)
2: Find a spanning tree $\mathcal{ST}$ with leader L.
3: L broadcasts $< start, 1, \sqrt{n^2-n} >$ when $\mathcal{ST}$ is complete.
4: When a leaf p receives $< start, s, e >$, if $index(p) \in [s, mid(s, e)]$ then p sends $< s, e, 1 >$ to its parent; otherwise sends $< s, e, 0 >$.
5: For specific s and e, when a non-leaf node p, including the leader, receives $< s, e, x_q >$ from all of its children q, p computes $x_p = t_p + \sum_q x_q$, where $t_p = 1$ if $index(p) \in [s, mid(s, e)]$ and $x_p = 0$ otherwise. If $p \neq L$, then p sends $< s, e, x_p >$ to its parent.
6: For specific s and e, when L receives $< e, s, x_q >$ from all of its children q and computes x_L,
If $e - s = 1$, L broadcasts $< cell, index(ce) >$ to its children, where ce is one of c_e or c_s that contains at least two points.
Else If $x_L > mid(s, e) - s + 1$, L broadcasts $< start, s, mid(s, e) >$.
Else L broadcasts $< start, mid(s, e) + 1, e >$.
7: When a node $p \neq L$ receives $< cell, index(ce) >$, if $index(p) = index(ce)$ and p has not sent more than one $< decision >$ message to its parent before, then p sends $< decision, loc(p) >$ to its parent.
8: When a non-leaf node p receives $< decision, loc(q) >$, from one of its children q if p has not send more than one $< decision >$ message to its parent before, then p sends $< decision, loc(q) >$ to its parent.
9: When L receives $< decision, loc(q) >$, from one of its children q if L has already received another $< decision >$ message or $index(L) = index(ce)$, then L broadcasts the location of these two points in ce through ST.
10: **end procedure**

Theorem 2.6. *There exists an asynchronous algorithm in CONGEST KT1 which computes an* $\frac{n^c}{\sqrt{\frac{n-1}{2}}}$*- Approximation Closest Pair with high probability in a graph of n nodes on an $[n^c] \times [n^c]$ grid, for a constant $c > 1/2$, with $O(n \lg n)$ messages and time $O(diam(\mathcal{ST}) \log n)$ plus the costs for building the $\mathcal{ST}$.*

Proof. We observe that in Algorithm 4 after computing a spanning tree, each node sends $O(\lg \sqrt{n^2-n})$ messages of size $O(\log n)$. Thus the total number of messages is $O(n \lg \sqrt{n^2-n})$ or $O(n \log n)$ plus the cost of ST.

The approximation ratio of the algorithm is $\frac{n^c}{\sqrt{\frac{n-1}{2}}}$. This is because there is always a cell with more than one point, and the cells are $\frac{n^c}{\sqrt{n}}$ by $\frac{n^c}{\sqrt{n-1}}$. Thus, there is a pair of points with a distance of at most the diameter of a $\frac{n^c}{\sqrt{n}}$ by $\frac{n^c}{\sqrt{n-1}}$ cell, which is at most $\frac{\sqrt{2}n^c}{\sqrt{n-1}}$, but the distance of the actual closest pair could be as small as 1, as the points are on the grid and this is the smallest possible distance between any two points. □

Corollary 2.7. *There exists an asynchronous algorithm in CONGEST KT1 which computes an $\frac{n^c}{\sqrt{\frac{k}{2}}}$-Approximate Closest Pair with high probability in a graph of n nodes on an $[n^c] \times [n^c]$ grid, for a constant $c > 1/2$, in $O(n \log k)$ messages in time $O(diam(\mathcal{ST}) \log k)$ plus the cost of computing $\mathcal{ST}$, for any $k \le n - 1$.*

Proof. If in Algorithm 4 we divide the grid to $\sqrt{k}$ by $\sqrt{k}$ cells evenly, each size of $\frac{n^c}{\sqrt{k}}$, the diameter of each cell is $\frac{n^c}{\sqrt{\frac{k}{2}}}$, and so is the approximation ratio of the algorithm.

Each node p sends $O(\log k)$ messages of size $O(\log n)$, plus the cost of computing a spanning tree. □

3. Lower bounds

In Section 3.1, we study Set Disjointness problem which is a well-studied problem on two-party communication. Then we formulate Path Computation and show a lower bound for this problem.

In Sections 3.2, 3.3 and 3.4, we reduce Path Computation to Farthest Pair, Convex Hull, and Closest Pair to prove nearly matching lower bounds for our approximation results, and as well as the lower bounds for the exact problems.

3.1. Path computation

In this section, first we review two-party communication lower bounds, then formulate the path computation problem and prove a lower bound for this problem.

Two-party communication

In the basic two-party communication model (see [25]), there are two players, Alice and Bob, Alice knows x, Bob knows y and they wish to evaluate a function $f(x, y)$. The players communicate with each other via a two-way channel.

The algorithm is described by a *protocol tree* where each internal node is labeled by a player who sends a message, each leaf is labeled with the outcome for $f(x, y)$, and the tree branches depending on the value of the string. The cost of the protocol is the length of the longest path from the root to the leaf.

A protocol with private randomness also contains nodes where the player flips a coin. With public randomness, the random bits are revealed to both players. We may assume this happens at the start of the protocol; Alice and Bob choose a common random string independent of (x, y) which selects a deterministic protocol from a distribution of such protocols. If we allow randomness (either private or public) then for any given input (x, y), there is a probability of reaching a particular leaf, and the expected cost of the protocol on an input (x, y) is the expected number of bits needed to be communicated over all choices of the random strings to reach a leaf. We may then consider the worst case expected cost of the randomized algorithm to be the worst case cost over all inputs. The communication complexity of a problem is the maximum over all inputs of the expected cost. When we allow randomization, protocols might have errors. Let P be a randomized protocol. P computes a function f with ϵ-error if for every (x, y), $Pr[P(x, y) = f(x, y)] \ge 1 - \epsilon$ over the random choices of the random strings [25].

The basic two-party model assumes messages arrive as soon as they are sent but there is no notion of a global clock, so that information is not gained if Alice waits or sends an empty message. In a 2010 paper [20], Impagliazzo and Williams introduced the *synchronous bit two-party communication model* in which there is a global clock. The levels of the protocol tree represent time steps, and at each time step, a party may send a 0,1, or $*$, where $*$ means no message is sent. Cost is measured by the number of 0's and 1's sent. There are clearly communication-time tradeoffs. For example, Alice can send an n bit string by sending one bit in exponential time equal to the value of the string. Or, in general with time polynomial in n, Alice can send an n bit string using a cost of $n/\log n$ in this model. See [20,36]. Hence it does not make sense to talk about a lower bound on communication costs in the synchronous model without an upper bound on the time. For the remainder of this section, we prove lower bounds on communication cost in models with no global clock.

The following well-studied problem will be used as a basis for proving lower bounds in this research:

- **Set Disjointness:** Alice has set A and Bob has set B, A and $B \subset \{1, 2, ..., n\}$. Give a two-party communication protocol to answer if $A \cap B = \emptyset$.

No algorithm can do better asymptotically than the one where Alice sends the n bit characteristic vector of A to Bob, and Bob returns the one bit solution to Alice.

Theorem 3.1. *[23,35,16] Alice has set A and Bob has set B, A and $B \subset \{1, 2, ..., n\}$. Any randomized two-party communication protocol to answer if $A \cap B = \emptyset$ requires $\Omega(n)$ bits of communication to achieve a constant error probability c, for any $c < 1/2$, even if there is public randomness.*

Path computation problem

In this section, we formulate and analyze the Path Computation problem.

For any two-party function f, we can define a problem in a communication network consisting of a path.

- **Path Computation of f**
  We are given a function $f(x, y)$ and a communication network which consists of a path $P = (v_0, v_1, v_2, ..., v_m)$, where $v_0 =$ Alice, $v_m =$ Bob. Alice knows x, Bob knows y, and all the nodes know m and n, where n is the size of the input, i.e. $n = |x| = |y|$. Assume Alice and Bob are awake at the start. Nodes pass messages until every node in P learns $f(x, y)$.

One of the naive solutions for Path Computation is that Alice sends x to Bob by passing exactly x itself through the whole path, and Bob passes y or $f(x, y)$ to Alice similarly. Thus, if x and y are n-bit strings and the path has $m + 1$ nodes including Alice and Bob, this approach uses at least mn bits of communication to send only x to Bob.

Here we explain one way that we save some communication in Path Computation compared to the naive strategy.

The idea is to assign an ordered set of $\sqrt{m}$ distinct bit-strings to each node in the path, and every node knows the assigned strings for themselves and their neighbors. If node i wants to send an n-bit message x to the node $i + 1$, it checks if the last $\log n$ bit of the message is included in the assigned strings for $i + 1$. If yes, it replaces the suffix with the index of the string, and sends it. Otherwise, it sends the message x with no change. The receiver can distinguish whether the message has been shortened by checking the size of the message. If the size is less than n, then the message has been shortened and needs to be *decompressed*.

Now we explain this idea in more detail. Assume x and y are n-bit strings, and Alice wants to send x to Bob. In the following algorithm we assume that the nodes are labeled 0 to m and $n \le m \le n^2$. Given $\{v_0, v_1, ..., v_m\}$, the goal is to send n bits of information from v_0 to v_m. Every node runs this protocol independently upon waking up. All the nodes know n, m, and the function $F : N \to S_i$, where $N = [0, m]$ and S_i is an ordered set of $\sqrt{m}$ distinct bit-string each of length $\log n$ bits, such that the multiset $\bigcup_{1\le i\le m} S_i$ has $\sqrt{m}$ copies of all the members of $\{0, 1\}^{\log n}$. In other words, every string of length $\log n$ is repeated $\sqrt{m}$ times in the multiset union of the function's image. In this algorithm the ith node in the path that receives a message, first decompresses the message by using the *decompress* function, defined below. Then the node compresses the result by computing the *compress* function, defined below, and sends the result to the next node.

$$\text{decompress}_i(x) = \begin{cases} x & \text{if } |x| = n \\ \text{prefix}(x, n - \log n).S_i[\text{suffix}(x, \log n/2)] & \text{otherwise} \end{cases}$$

When the node i sends message x to node $i + 1$,

$$\text{compress}_i(x) = \begin{cases} \text{prefix}(x, n - \log n).\text{index}(S_{i+1}, \text{suffix}(x, \log n)) & \text{if suffix}(x, \log n) \in S_{i+1} \\ x & \text{otherwise} \end{cases}$$

where suffix(x, k) is the last k bits of x, prefix(x, k) is the first k bits of x, $S_i[k]$ is the kth array in S_i, index(S, x) is the index of the string x in S, and "." denotes concatenation.

Algorithm 5 Path Computation.

```
procedure PATH(i)
    v_0 computes compress_0(x) and sends it to v_1.
    When v_m receives the message x, computes decompress_m(x).
    When v_i receives the message x from v_{i-1}, computes compress_i(decompress_i(x)) and sends it to v_{i+1}.
end procedure
```

Analysis of Algorithm 5: For any given x, suffix$(x, \log n)$ appears $\sqrt{m}$ times in multiset union of $\bigcup_{1\le i\le m} S_i$. Thus, the compress function is called $\sqrt{m}$ times during the algorithm. A compressed message has $\log n/2$ bits less than other messages, and this means the algorithm saves $\log n/2$ bits $\sqrt{m}$ times. Thus, the algorithm uses $mn - \sqrt{m}\log n/2$ bits of communication.

Theorem 3.2. *There exists a protocol with variable size messages in the KT1 model which computes Path Computation of $f(x, y) = x$ for a path of $m + 1$ nodes with $mn - \sqrt{m}\log n/2$ bits of communication, where $|x|, |y| = n$ and $n \le m \le n^2$.*

As we can see the complexity of Path Computation is not obvious. In the next section we prove a lower bound for Path Computation.

Lower bound for path computation

We use two-party communication lower bounds to prove a lower bound on Path Computation.

Theorem 3.3. *Let $C_2(f)$ be the randomized complexity of f in the basic two-party communication model with private randomness. Then the randomized complexity of Path Computation of f in the CONGEST-b KT1 model where each message has b bits is at least $\Omega(\frac{m}{b}(C_2(f) - \lg m))$, where m is the length of the path.*

To prove a lower bound, it suffices to show there is some adversary strategy which requires this amount of communication. We assume the adversary acts in a specific way, and this assumption can only make it easier for the algorithm. First, we prove a bound for $b = 1$. We call the model with this restricted adversary the *r-asynchronous* CONGEST-1 KT1. The restricted model takes away adversary control over the order in which messages are received, while there is still no global clock, as in the basic two-party communication model. Hence the adversary only controls the inputs. Messages are received in the order they are sent; where there is a choice as to which of two messages is received from two different neighbors, the message from the lower numbered node is received first; and if these rules leave a choice as to which of two messages to be sent first in opposite directions over the same link, the message from the lower numbered node is sent first. The nodes in the path have ID's $(0, 1, ..., m)$. In the proofs below, the term "path protocol" refers to an algorithm to solve Path Computation of f, in the r-asynchronous CONGEST-b KT1 model. Now we show:

Lemma 3.4. *For every path protocol in the asynchronous model which sends an expected k bits across any edge $\{i, i+1\}$ there is a two-party communications protocol with private randomness and expected cost no greater than k.*

Proof. We show how to construct decision trees for a randomized two-party protocol (with private randomness) from the asynchronous protocol with a suitably defined schedule for delivery of messages. Let path $A = [0, 1, ..., i]$ and path $B = [i+1, ..., n]$. The schedule consists of iterations with two parts. In part (a) messages from i to $i+1$ and in path B are kept waiting; in part (b), messages from $i+1$ to i and in path A are kept waiting.

(a) For a given r_A and x, Alice simulates the nodes and their coinflips on the path from 0 to i, assuming any message delivery schedule which completes all deliveries between nodes on path A, and uses r_A to determine the random coinflips in the distributed protocol as needed, until all messages which would be sent from i to $i+1$ in the asynchronous model (without receiving more messages from $i+1$) are sent by A in the order in which they would be sent by i. These are received by $i+1$.

(b) The messages from part (a) are received by $i+1$. For a given r_B and y, Bob similarly simulates the actions of the nodes in path B, and determines and sends all the messages $i+1$ would send to i in the order these would have been sent (without receiving more messages from i). The messages sent from $i+1$ are received by i.

We observe the followings:

- For a given (r_A, x), (r_B, y) each iteration extends a path in a two-party protocol's decision tree whose nodes are labeled first by (r_A, x) and then by (r_B, y).
- The expected number of bits exchanged between i and $i+1$ is the expected number of bits exchanged by i and $i+1$ in the asynchronous distributed network with the particular schedule of message delivery as described.

This completes the proof. □

Lemma 3.5. *For every path protocol in the r-asynchronous CONGEST-1 KT1 model, for every edge, there is at least one input which requires an expected $C_2(f)$ bits to be communicated over that edge.*

Proof. Assume by contradiction that $e = \{i, i+1\}$ is the edge closest to 0 which carries an expected $l < C_2(f)$ bits of communication for all input. By Lemma 3.4, there is a two-party protocol $\mathcal{P}$ which uses no more than an expected l bits of communication for all input, contradicting the lower bound of $C_2(f)$. □

Lemma 3.6. *Suppose there is a path protocol $\mathcal{P}$ such that in the r-asynchronous CONGEST-1 KT1 model for f on input I the expected communication cost is no greater than $C_{\mathcal{P}}(I)$ for a path of length m. Then there is a randomized path protocol $\mathcal{P}'$ such that the expected number of bits communicated across any edge is no greater than $C_{\mathcal{P}}(I)/m + \lceil \lg m \rceil$.*

Proof. We will define a protocol $\mathcal{P}'$ which simulates $\mathcal{P}$. We assume the first and last nodes wake up.

1. Node 0 randomly picks a number $a \in \{0, 1, ..., m-1\}$ and communicates a down the path to node m using $\lceil \lg m \rceil$ bits per edge.
2. Node 0 simulates the nodes in $[0, a]$ and forwards whatever message M that a would have sent to $a+1$ down the path to m.

3. After node m receives the value a, it simulates the nodes in $[a+1, m]$. When $a+1$ would send a message M' to a in the simulation, node m sends this message down the path to node 0.
4. All nodes $a' \in \{1, ..., m-1\}$ upon receiving a message from $a'-1$ pass it to node $a'+1$ and upon receiving a message from $a'+1$ passes it to $a'-1$.

Let $C_{\mathcal{P}}(I)$ denote the expected communication cost for input I under protocol $\mathcal{P}$. We can write $C_{\mathcal{P}}(I) = \sum_{i=0}^{m-1} c_{\mathcal{P}}(i, i+1, I) + c_{\mathcal{P}}(i+1, i, I)$, where $c_{\mathcal{P}}(d, e, I)$ denotes the expected number of bits sent from node d to node e under protocol $\mathcal{P}$ on input I.

We observe that when $\mathcal{P}'$ passes b bits from node 0 to node m or from node m to node 0, the bits arrive at m and the communication cost over all the edges is mb. Then, by this observation, for any input I, for a given choice of a, and for any edge in the path, under $\mathcal{P}'$, the communication per edge (in both directions) is no greater than $\lceil \lg m \rceil$ plus the communication cost $c_{\mathcal{P}}(a, a+1, I) + c_{\mathcal{P}}(a+1, a, I)$. Taking the expectation over all choices of a chosen uniformly at random, we have that the expected number of bits sent over any one edge $\{i, i+1\}$ in both directions by $\mathcal{P}'$ on input I, $c_{\mathcal{P}'}(i, i+1, I) + c_{\mathcal{P}'}(i, i-1, I)$ is:

$$\begin{aligned}
&\leq \lceil \lg m \rceil + \sum_{i=0}^{m-1} Pr(a=i)[c_{\mathcal{P}}(i, i+1, I) + c_{\mathcal{P}}(i+1, i, I)] \\
&= \lceil \lg m \rceil + 1/m[\sum_{i=0}^{m-1} c_{\mathcal{P}}(i, i+1, I) + c_{\mathcal{P}}(i+1, i, I)] \\
&= \lceil \lg m \rceil + (1/m) C_{\mathcal{P}}(I)
\end{aligned}$$

This completes the proof of the lemma. □

The proof of Theorem 3.3 follows: By Lemma 3.5, we know that for any protocol $\mathcal{P}'$ and any edge $\{i, i+1\}$, there is some I such that the expected number of bits passed in both directions over that edge $c_{\mathcal{P}'}(i, i+1, I) + c_{\mathcal{P}'}(i+1, i, I) \geq C_2(f)$. Hence by Lemma 3.6, $(1/m)C_{\mathcal{P}}(I) + \lceil \lg m \rceil \geq C_2(f)$ which implies that $C_{\mathcal{P}}(I) \geq m(C_2(f) - \lceil \lg m \rceil)$. Let $\mathcal{P}$ be the optimal algorithm for the path computation of f and I be its worst case input, then the communication complexity for the r-asynchronous CONGEST KT1 model $C_{m,r}(f) \geq C_{\mathcal{P}}(f, I) \geq m(C_2(f) - \lceil \lg m \rceil)$.

We observe that the lower bound for the asynchronous CONGEST-1 KT1 model $C_m(f) \geq C_{m,r}(f)$ since the restricted model can be simulated by an adversary strategy in the non-restricted model. If we consider the CONGEST-b KT1 model, then as b bits are communicated with each message, we can conclude that the complexity of computing f in this model is at least $C_m(f)/b$ or $\frac{m}{b}(C_2(f) - \lceil \lg m \rceil)$. □

Corollary 3.7. *The communication cost of computing Set Disjointness of size n over a path of length $O(n)$ is $\Omega(n^2/\log n)$ messages of size $O(\log n)$.*

3.2. Lower bound for farthest pair

We reduce the problem of Path Set Disjointness to Farthest Pair. Similar ideas have been used to show lower bounds in a streaming model [11]. Given a path of $m = n$ edges with endpoints Alice and Bob, where Alice and Bob each know subsets A and B resp., of $\{1, ..., n\}$, the communication network must compute the Set Disjointness problem on (a, b), where $a = (a_1, ..., a_n)$ and $b = (b_1, ..., b_n)$ are the characteristic vectors of A and B.

Consider the following graph $G_{A,B}(V, E)$ where $|V| = 3n$. Let $V = W \cup U \cup H$, $W = \{w_1, w_2, ..., w_n\}$, $U = \{u_1, u_2, ..., u_n\}$, $H = \{h_1, h_2, ..., h_n\}$, and the position of these nodes are described later.

We draw evenly spaced lines $L_1, ..., L_n$ through the center of the $[n^c] \times [n^c]$ grid, for a constant $c \geq 2 + 2/\lg n$. The angle between the two consecutive lines is $\delta = \pi/n$. Each line L is composed of a *low* ray $l(L)$ which goes from the center to below the center and the *upper* ray $u(L)$ which goes from the center to above the center. The lines are drawn so that the upper rays are at equally spaced angles from each other and between 0 and π with the x axis. The points of the Farthest Pair problem are determined as follows (Fig. 2), where $r = n$ and $R = 2n^2$.

- For each $1 \leq i \leq n$, if $a_i = 0$, w_i is on $u(L_i)$ at distance r from the center and otherwise at distance R.
- For each $1 \leq i \leq n$, if $b_i = 0$, u_i is on $l(L_i)$ at distance r from the center and otherwise at distance R.
- The h_i's, where $1 \leq i \leq n$, are located on distinct grid points inside the circle centered at the origin with radius r.

In the above description of the location of the points, in the case that any coordinate is not an integer coordinate, we round the point as follows: For each $1 \leq i \leq n$, if the angle of $u(L_i)$ with the x axis is less than or equal $\pi/2$, then we round w_i to the top right corner of the cell that contains w_i, and if the angle is more than $\pi/2$, we round w_i to the top left corner of the cell that contains w_i. For each $1 \leq i \leq n$, if the angle of $l(L_i)$ with the x axis is less than or equal $3\pi/2$, then we round u_i to the bottom left corner of the cell that contains u_i, and if the angle is more than $3\pi/2$ we round u_i to the bottom right corner of the cell that contains u_i.

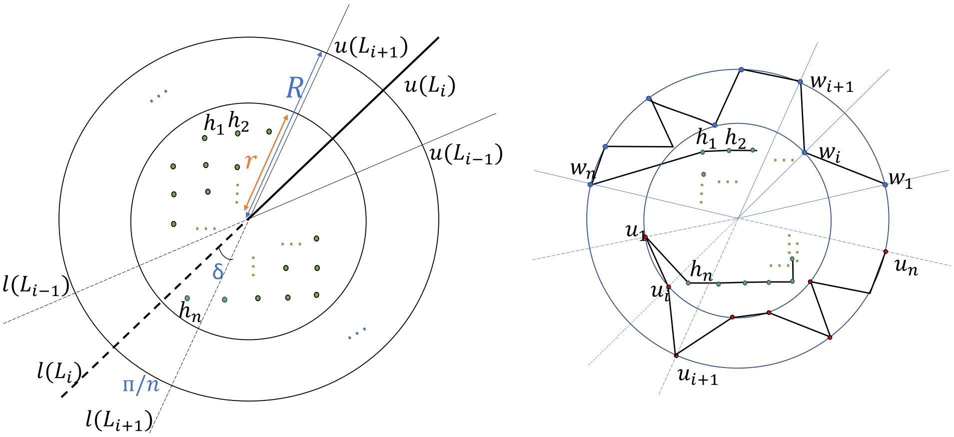

Fig. 2. Reduction from Set Disjointness to Farthest Pair. In this example, $a_1 = 1, a_i = 0, a_{i+1} = 1, a_n = 1, b_1 = 0, b_i = 0, b_{i+1} = 1$, and $b_n = 1$.

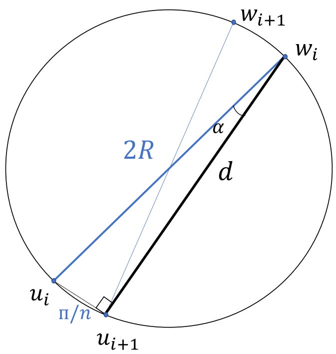

Fig. 3. The distance between w_i and u_{i+1} is the second largest distance of a pair of points after the farthest pair.

Note that a circle centered at the origin with radius r contains at least πr^2 lattice points [33]. So choosing $r = n$ guarantees that there are at least n grid points inside the circle to locate points of H. As $R = 2n^2$, no two points of W and U are in the same grid cell and they would not round to the same grid point.

We next describe the edges.

- For each $1 \le i \le n$, E contains $\{w_i, w_{i+1}\}$ and $\{u_i, u_{i+1}\}$.
- For each $1 \le i \le n$, E contains $\{h_i, h_{i+1}\}$.
- E contains $\{w_n, h_1\}$ and $\{h_n, u_1\}$.

Claim 3.8. *The distance of Farthest Pair of $G_{A,B}$ is $2R$ if and only if $A \cap B \neq \emptyset$.*

Proof. Let C_R, C_r be the circles centered at the center of the $[n^c] \times [n^c]$ grid at radius $R = 2n^2$ and $r = n$, respectively, for a constant $c \ge 2 + 2/\lg n$.

For any pair of points in $U \cup W$, the distance is strictly less than $2R$ if and only if one of the following is true: (1) the two points are on two different lines; (2) the two points are on the same line, one at distance r and the other at distance R.

If the points are rounded, this is still true for (1) as the length of the longest chord after the diameter of the circle is at most $d + 2\sqrt{2}$, where $d = 2R \cdot \cos\alpha$, and $\alpha = \frac{\delta}{2} = \frac{\pi}{2n}$, see Fig. 3. This is because the maximum rounding error occurs when both end points of the chord are rounded, and each of the roundings can add at most $\sqrt{2}$ to the length of the chord. By the Taylor series for $\cos\alpha$, $\cos\alpha \le 1 - \frac{\alpha^2}{2!} + \frac{\alpha^4}{4!} \le 1 - \frac{11\alpha^2}{4!}$. Since $\alpha = \pi/2n$ and $R = 2n^2$, then $2R \cdot \cos\alpha + 2\sqrt{2} < 2R - 11\frac{\pi^2}{4n^2 \cdot 24} \cdot 4n^2 + 2\sqrt{2} < 2R$.

If the points are rounded, this is still true for (2) as the maximum distance of the two points on the same line, one at distance r and the other at distance R, after rounding is $R + r + 2\sqrt{2}$. Since $r = n$ and $R = 2n^2$, then $R + r + 2\sqrt{2} < 2R$, for any $n \ge 2$.

If there is an L_i such that both w_i and u_i are distance R from the center, then their distance is at least $2R$. This occurs only when for some i, $a_i = b_i = 1$, so $A \cap B \neq \emptyset$. $\square$

Theorem 3.9. *Any randomized asynchronous distributed algorithm in the CONGEST KT1 model for solving Farthest Pair in a graph of n nodes requires $\Omega(n^2)$ expected bits of communication.*

Proof. Suppose Alice and Bob are two endpoints of a path with $N+1$ edges numbered $0, 1, \ldots, N$, where $N = n/3$. Alice and Bob each know subsets A and B resp., of $\{1, \ldots, N\}$, and they want to solve the Path Computation for the Set Disjointness problem on (a, b) over a path of size N, where $a = (a_1, \ldots, a_N)$ and $b = (b_1, \ldots, b_N)$ are the characteristic vectors of A and B. Suppose both Alice and Bob know H, the path of size N. Alice can construct the subgraph of $G_{A,B}$ induced by $H \cup W$ and given B, Bob can construct the subgraph induced by $H \cup U$. They can then each carry out a simulation of the protocol which determines the distance of the Farthest Pair of $G_{A,B}$ and output 1 or 0 depending on whether the distance of Farthest Pair is at least $2R$ or not. Thus the communication cost is no less than the cost of computing Set Disjointness over a path of length N where $n = 3N$ is the total number of nodes in $G_{A,B}$. By Corollary 3.7, $\Omega(n^2/\lg n)$ messages of size $O(\log n)$ are required. $\square$

Theorem 3.10. *Any randomized asynchronous distributed algorithm in the CONGEST KT1 model for approximating Farthest Pair in a graph of n nodes within a $1-\epsilon$ factor of the optimum requires $\Omega(min\{n^2, 1/\epsilon\})$ expected bits of communications.*

Proof. Consider the reduction from Set Disjointness to Farthest Pair described before with vertices $V = W \cup U \cup H$ as shown in Fig. 2. Let C_R, C_r be the circles centered at the center of the $[n^c] \times [n^c]$ grid at distance $R = 2n^2$ and $r = n$, respectively, for a constant $c \geq 2 + 2/\lg n$. The maximum distance between any pair of points is the diameter of the largest circle, which is at least $2R$. The next greatest distance in this structure is $d + 2\sqrt{2} = 2R\cos\alpha + 2\sqrt{2}$, as this is the longest chord after diameter on the largest circle, and it is greater than the diameter of the smaller circle, which is $r = n$, see Fig. 3. The reason that the next greater distance is $d + 2\sqrt{2}$ and not exactly d is that we are rounding the points to the grid points. As no more than one point is in one cell and the points are rounded to the outside of the circle, the rounding can only increase the distance of the points from the center of the circle. This means the ratio of $\frac{d}{2R}$ after the rounding is not more than $\frac{d+2\sqrt{2}}{2R} = \frac{2R\cdot\cos\alpha+2\sqrt{2}}{2R} = \cos\alpha + \frac{2\sqrt{2}}{2R}$. As we assumed $R = 2n^2$ and $\alpha = \frac{\pi}{2n}$, the ratio of $\frac{d+2\sqrt{2}}{2R}$ after the rounding is not more than $\cos\frac{\pi}{2n} + \frac{\sqrt{2}}{2n^2}$.

Therefore, if there is a $(1-\epsilon)$-approximation algorithm for the Farthest Pair problem, where $1-\epsilon \geq \cos\frac{\pi}{2n} + \frac{\sqrt{2}}{2n^2}$, then the algorithm solves the problem of Set Disjointness of size $n/3$. Thus, if $1-\epsilon \geq \cos\frac{\pi}{2n} + \frac{\sqrt{2}}{2n^2}$, then by Theorem 3.9 any randomized asynchronous distributed algorithm for approximating Farthest Pair of n points within a $1-\epsilon$ factor of the optimum requires $\Omega(n^2)$ bits of communications.

If $1-\epsilon < \cos\frac{\pi}{2n} + \frac{\sqrt{2}}{2n^2}$, we build the same structure but with $\frac{3}{2\sqrt{\epsilon}}$ points, where $R = 2n^2$, and reduce a Set Disjointness problem with size $n' = \frac{1}{2\sqrt{\epsilon}}$ to this problem. In this structure the angle between any two consecutive lines is $\delta = \pi/n' = 2\pi\sqrt{\epsilon}$, and $\alpha' = \delta/2 = \pi\sqrt{\epsilon}$. Now we show that in this structure, the ratio of the second longest chord to the diameter is less than $1-\epsilon$.

The ratio of the second longest chord to the diameter in this structure is at most $\frac{2R\cdot\cos\alpha'+2\sqrt{2}}{2R}$. By the Taylor series for $\cos\alpha'$ we know that $\cos\alpha' \leq 1 - \frac{\alpha'^2}{2!} + \frac{\alpha'^4}{4!} \leq 1 - \frac{11\alpha'^2}{4!}$. Since $\alpha' = \pi\sqrt{\epsilon}$ and $R = 2n^2$, then

$$\frac{2R\cdot\cos\alpha'+2\sqrt{2}}{2R} \leq \cos\alpha' + \frac{\sqrt{2}}{R} < 1 - \frac{11\pi^2\epsilon}{24} + \frac{\sqrt{2}}{2n^2} \leq 1 - 4\epsilon + \frac{\sqrt{2}}{2n^2}$$

We assumed that $1-\epsilon < \cos\frac{\pi}{2n} + \frac{\sqrt{2}}{2n^2}$, and by the Taylor series for $\cos\frac{\pi}{2n}$ we know that $1 - \cos\frac{\pi}{2n} > \frac{11\pi^2}{4!n^2}$. Thus,

$$\epsilon > 1 - \cos\frac{\pi}{2n} - \frac{\sqrt{2}}{2n^2} > \frac{11\pi^2}{4!n^2} - \frac{\sqrt{2}}{2n^2} = \frac{11\pi^2 - 24\sqrt{2}}{4!n^2} + \frac{\sqrt{2}}{2n^2} > \frac{\sqrt{2}}{2n^2}$$

Therefore, $\frac{2R\cdot\cos\alpha'+2\sqrt{2}}{2R} \leq 1 - 4\epsilon + \frac{\sqrt{2}}{2n^2} < 1 - 4\epsilon + \epsilon \leq 1 - 3\epsilon < 1 - \epsilon$.

This means that in this structure the ratio of the second longest chord to the diameter is less than $1-\epsilon$, and a $(1-\epsilon)$-approximation algorithm for the Farthest Pair solves the problem of Set Disjointness of size $n' = \frac{1}{2\sqrt{\epsilon}}$. By Theorem 3.9 any randomized asynchronous distributed algorithm for approximating Farthest Pair of $n' = \frac{1}{2\sqrt{\epsilon}}$ points within $1-\epsilon$ factor of the optimum requires $\Omega(1/\epsilon)$ bits of communications. $\square$

This lower bound shows a tradeoff between the approximation ratio and the amount of communication that any randomized asynchronous distributed algorithm needs to compute Approximate Farthest Pair. When ϵ is very small, $\epsilon \leq \frac{1}{n^2}$, the approximation is very close to the exact solution, and $\Omega(n^2)$ expected bits of communication are needed; otherwise $\Omega(1/\epsilon)$ expected bits of communication are needed.

Corollary 3.11. *Any randomized asynchronous distributed algorithm for computing ϵ-kernel in the CONGEST KT1 model, for any $\epsilon > 0$, requires $\Omega(min\{n^2, 1/\epsilon\})$ expected bits of communications.*

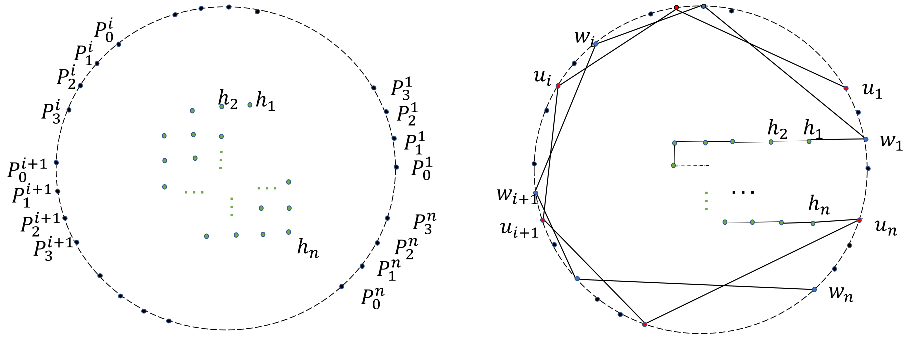

Fig. 4. Reduction from Set Disjointness to Convex Hull. In this example, $a_1 = 1, a_i = 0, a_{i+1} = 1, a_n = 0, b_1 = 1, b_i = 0, b_{i+1} = 0$, and $b_n = 1$.

3.3. Lower bound for convex hull

We reduce the problem of Path Computation for Set Disjointness to Convex Hull. We show that if Convex Hull can be solved in $o(n^2)$ communication in an asynchronous network with n nodes, then this implies an algorithm for Set Disjointness on a path of length $n/3$ with fewer than $\Omega(n^2)$ expected bits of communication, giving a contradiction.

Given a path of $m = n$ edges with endpoints Alice and Bob, where Alice and Bob each know subsets A and B resp., of $\{1, \dots, n\}$, the communication network must compute the Set Disjointness problem on (a, b), where $a = (a_1, \dots, a_n)$ and $b = (b_1, \dots, b_n)$ are the characteristic vectors of A and B. Consider the following graph $G_{A,B}(V, E)$ where $|V| = 3n$. Let $V = W \cup U \cup H$, $W = \{w_1, w_2, \dots, w_n\}$, $U = \{u_1, u_2, \dots, u_n\}$, $H = \{h_1, h_2, \dots, h_n\}$.

We consider $4n$ positions equally distanced on a circle centered at the center of the $[n^c] \times [n^c]$ grid, for a constant $c > 1 + 1/(2 \lg n)$, with radius $R = 2n$. There are n groups of points consecutively ordered by their superscripts around the circle. Each group contains four points, consecutively ordered by their subscripts. These are given by $P = \{P^i_j | 1 \le i \le n, 0 \le j \le 3$, as shown in Fig. 4. In the case that any coordinate is not an integer coordinate, we round the point to one of the grid points of the cell that contains this point as mentioned before in Section 3.2. The points of vertices V for the Convex Hull problem are determined as follows (Fig. 4):

- For each $1 \le i \le n$, if $a_i = 0$, w_i is on the location P^i_0; otherwise on the location P^i_1;
- For each $1 \le i \le n$, if $b_i = 0$, u_i is on the location P^i_2; otherwise on the location P^i_3;
- The h_i's, where $1 \le i \le n$, are located on distinct grid points inside the circle centered at the center of the grid with radius $r = n$.

Note that a circle centered at the origin with radius r contains at least πr^2 lattice points [33]. So choosing $r = n$ guarantees that there are more than n grid points inside the circle to locate the points of H. We choose $R = 2n$ to make sure no two points of W and U are in the same grid cell and would not round to the same grid point.

We next describe the edges.

- For each $1 \le i \le n$, E contains $\{w_i, w_{i+1}\}$ and $\{u_i, u_{i+1}\}$.
- For each $1 \le i \le n$, E contains $\{h_i, h_{i+1}\}$.
- E contains $\{w_1, h_1\}$ and $\{h_n, u_n\}$.

Claim 3.12. *$A \cap B \neq \emptyset$ if and only if there is some i for which w_i is on P^i_1 and the location of its neighbor on the convex hull of $G_{A,B}$ is on position P^i_3 in the counter-clockwise direction. Equivalently, $A \cap B \neq \emptyset$ if and only if there is some i for which u_i is on P^i_3 and the location of its neighbor on the convex hull of $G_{A,B}$ is on position P^i_1 in the clockwise direction.*

Proof. In the solution of Convex Hull for these points, all the u_i's and w_i's are on the convex hull and no h_i is on the convex hull. For any $1 \le i \le n$, u_i is the neighbor of w_i on the convex hull of $G_{A,B}$ in the counter-clockwise direction. So, in the solution of Convex Hull, if there is some i for which w_i is on P^i_1 and the location of its neighbor on the convex hull of $G_{A,B}$ is on position P^i_3 in the counter-clockwise direction, this means u_i is on P^i_3. So, both $a_i = 1$ and $b_i = 1$. Equivalently, $A \cap B \neq \emptyset$ if and only if there is some i for which u_i is on P^i_3 and the location of its neighbor on the convex hull of $G_{A,B}$ is on position P^i_1 in the clockwise direction. □

Theorem 3.13. *Any randomized asynchronous distributed algorithm in the CONGEST KT1 model for solving the Convex Hull in a graph of n nodes requires $\Omega(n^2)$ expected bits of communication.*

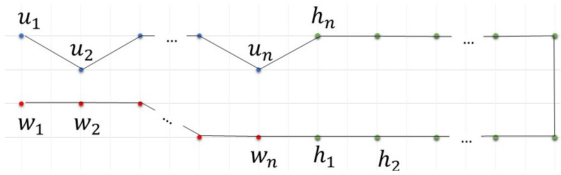

Fig. 5. Reduction from Set Disjointness to Closest Pair. In this example $a_1 = 1$, $a_2 = 1$, $a_n = 0$, $b_1 = 0$, $b_2 = 1$, and $b_n = 1$.

Proof. Suppose Alice and Bob are two endpoints of a path with $N+1$ edges numbered $0, 1, ..., N$, where $N = n/3$. Alice and Bob each know subsets A and B resp., of $\{1, ..., N\}$, and they want to solve the Path Computation of the Set Disjointness problem on (a, b) over a path of size N, where $a = (a_1, ..., a_N)$ and $b = (b_1, ..., b_N)$ are the characteristic vectors of A and B. Both Alice and Bob know the candidate positions P and the path H. Alice can construct the subgraph of $G_{A,B}$ induced by $H \cup W$ and given B, Bob can construct the subgraph induced by $H \cup U$. They can then each carry out a simulation of the protocol which computes the convex hull of vertices of G and output 1 or 0 depending on the answer of Convex Hull of $G_{A,B}$ by using Claim 3.12. Since Alice knows the states of all w_i and each w_i knows the location of its neighbors on the convex hull, Alice outputs 1 if and only if there is some i for which w_i is on P_1^i and the location of its neighbor on the convex hull of $G_{A,B}$ is on position P_3^i in the counter-clockwise direction. Similarly Bob outputs 1 if and only if there is some i for which u_i is on P_3^i and the location of its neighbor on the convex hull of $G_{A,B}$ is on position P_1^i in the clockwise direction. Thus the communication cost is no less than the cost of computing Set Disjointness of size N over a path of length N where $n = 3N$ is the total number of nodes in G. By Corollary 3.7, $\Omega(n^2/\lg n)$ messages of size $O(\log n)$ are required. □

3.4. Lower bound for closest pair

We show a reduction from Path Computation for Set Disjointness to Closest Pair. Similar ideas were used in the streaming model [11].

Given an instance of the Path Computation for Set Disjointness problem, in which there are two players Alice and Bob the two endpoints of a path of $m = n$ edges with endpoints Alice and Bob, Alice has subset A with characteristic vector $a = (a_1, ..., a_n)$ and Bob has subset B with characteristic vector $b = (b_1, ..., b_n)$, we consider the following graph $G_{A,B}(V, E)$ where $|V| = 3n$. Let $V = W \cup U \cup H$, $W = \{w_1, w_2, ..., w_n\}$, $U = \{u_1, u_2, ..., u_n\}$, $H = \{h_1, h_2, ..., h_n\}$. The locations of the vertices of $G_{A,B}$ are as follows (See Fig. 5):

- For each $1 \le i \le n$, if $a_i = 0$, w_i is on coordinate $(2i, 0)$; otherwise on coordinate $(2i, 1)$.
- For each $1 \le i \le n$, if $b_i = 0$, u_i is on coordinate $(2i, 3)$; otherwise on coordinate $(2i, 2)$.
- For each $1 \le i \le n/2$, h_i is located on coordinate $(2(i+n), 0)$, and for each $n/2 + 1 \le i \le n$, h_i is on coordinate $(2(2n - i + 1), 3)$.

We next describe the edges.

- For each $1 \le i \le n$, E contains $\{w_i, w_{i+1}\}$ and $\{u_i, u_{i+1}\}$.
- For each $1 \le i \le n$, E contains $\{h_i, h_{i+1}\}$.
- E contains $\{w_n, h_1\}$ and $\{h_n, u_1\}$.

Claim 3.14. *The distance of the two closest vertices in the problem of Closest Pair on $G_{A,B}$ is 1 if and only if $A \cap B \neq \emptyset$.*

Proof. It is clear that the distance between any two points in W or any two points in U is at least 2. The distance between any pair of points in $H \cup W$ and any pair of points in $H \cup U$ is also at least 2. The distance between the points of W and the points of U is 1 if and only if there is an i such that w_i is on $(2i, 1)$ and u_i is on $(2i, 2)$. This occurs only when $a_i = 1$ and $b_i = 1$, so $A \cap B \neq \emptyset$. □

Theorem 3.15. *Any randomized asynchronous distributed algorithm for solving Closest Pair in a graph with n nodes on an $[n^c] \times [n^c]$ grid in the CONGEST KT1 model, for a constant $c > 1 + 1/(2\lg n)$, requires an expected $\Omega(n^2)$ bits of communication.*

Proof. Suppose Alice and Bob are two endpoints of a path with $N+1$ edges numbered $0, 1, ..., N$, where $N = n/3$. Alice and Bob each know subsets A and B resp., of $\{1, ..., n\}$, and they want to solve the Path Computation of the Set Disjointness problem on (a, b) over a path of size N, where $a = (a_1, ..., a_N)$ and $b = (b_1, ..., b_N)$ are the characteristic vectors of A and B. Both Alice and Bob know H, the path of size N.

Alice can construct the subgraph of $G_{A,B}$ induced by $H \cup W$ and given B, Bob can construct the subgraph induced by $H \cup U$. They can then each carry out a simulation of the protocol which determines the closest pair of $G_{A,B}$ and output 0 iff the closest pair is distance 1 away from each other. Thus the communication cost is no less than the cost of computing

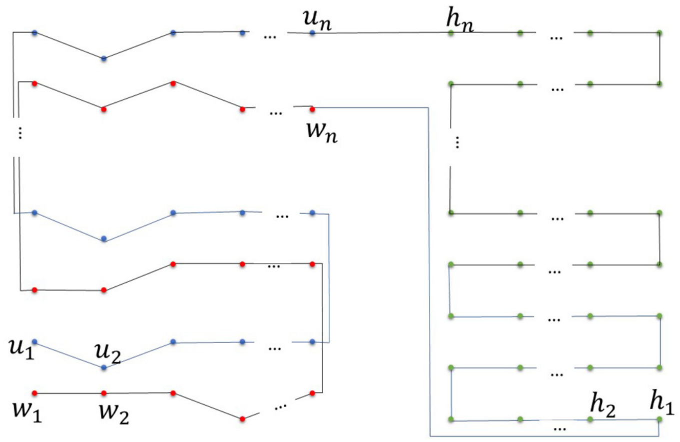

Fig. 6. Reduction from Set Disjointness to Closest Pair. In this example, $a_1 = 1, a_2 = 1, a_n = 0, b_1 = 0, b_2 = 1$, and $b_n = 0$.

Set Disjointness over a path of length N, where $n = 3N$ is the total number of nodes in G. By Corollary 3.7, $\Omega(n^2/\lg n)$ messages of size $O(\log n)$ are required. □

Theorem 3.16. *Any randomized asynchronous distributed algorithm for approximating Closest Pair in a graph with n nodes on an $[n^c] \times [n^c]$ grid in the CONGEST KT1 model, for a constant $c > 1 + 1/(2\lg n)$, within a $\frac{n^{c-1}}{2}$ factor of the optimum requires an expected $\Omega(n^2)$ bits of communications.*

Proof. Consider the reduction shown in the previous section from Set Disjointness to Closest Pair with a slightly different location for the points such that the minimum possible distance for a pair of points is 1 or $\frac{n^{c-1}}{2}$ of each other in the similar structure shown in Fig. 5: Let $k = \frac{n^{c-1}}{2}$,

- For each $1 \le i \le n$, if $a_i = 0$, w_i is on coordinate $(ki, 0)$; otherwise on coordinate (ki, k).
- For each $1 \le i \le n$, if $b_i = 0$, u_i is on coordinate $(ki, 2k+1)$; otherwise on coordinate $(ki, k+1)$.
- h_i is located on coordinate $(k(i+n), 0)$ for $1 \le i \le n/2$ and on coordinate $(k(i+n), 2k+1)$ for $n/2+1 \le i \le n$.

If there is an approximation algorithm for Closest Pair with the approximation ratio of at most $\frac{n^{c-1}}{2}$ and uses $o(n^2)$ bits of communication, this means that the algorithm can distinguish between distance $\frac{n^{c-1}}{2}$ and distance 1 and answer Path Computation for Set Disjointness of size n with using $o(n^2)$ bits of communication, which is a contradiction. □

The following theorem shows a better lower bound for Closest Pair with the idea of spreading the points even more on the plane. The result of Theorem 3.17 subsumes the result of the two previous theorems (Theorem 3.15 and Theorem 3.16), but these two theorems help to elucidate the structure of the graph in Theorem 3.17.

Theorem 3.17. *Any randomized asynchronous distributed algorithm for approximating Closest Pair in a graph with n nodes on an $[n^c] \times [n^c]$ grid in the CONGEST KT1 model, for a constant $c > 1 + 1/(2\lg n)$, within a $\frac{n^{c-1/2}}{2}$ factor of the optimum requires an expected $\Omega(n^2)$ bits of communications.*

Proof. When reducing Path Computation for Set Disjointness to Closest Pair, instead of locating all the n points on four horizontal lines, $y = 0, k, k+1, 2k+1$, where $k = \frac{n^{c-1}}{2}$, similar to Fig. 5, here we locate the points differently: on $\frac{3}{2}\sqrt{n}$ groups where each group contains $\frac{2}{3}\sqrt{n}$ points as shown in Fig. 6 and described below.

Let $k' = \frac{n^{c-1/2}}{2}$,

- For each $0 \le i \le \frac{3}{2}\sqrt{n}$ and $0 \le j \le \frac{2}{3}\sqrt{n}$, $ik' + j \le n$, if $a_{jk'+i} = 0$, v_i is on coordinate $(k'i, 3jk' + j)$; otherwise on coordinate $(k'i, (3j+1)k' + j)$.
- For each $0 \le i \le \frac{3}{2}\sqrt{n}$ and $0 \le j \le \frac{2}{3}\sqrt{n}$, $ik' + j \le n$, if $b_{jk'+i} = 0$, u_i is on coordinate $(k'i, (3j+1)k' + j + 1)$; otherwise on coordinate $(k'i, (3j+2)k' + j + 1)$.
- For each $0 \le i \le \sqrt{n}/2$ and $0 \le j \le 2\sqrt{n}$, $ik' + j \le n$, $h_{jk'+i}$ is on coordinate $(k'i + \frac{3}{4}n^c, jk')$.

In this way the minimum possible distance between a pair of points is 1 or k'. If there is an approximation algorithm for Closest Pair with an approximation ratio no more than $\frac{n^{c-1/2}}{2}$ and uses $o(n^2)$ bits of communication, this means that the algorithm can distinguish between distance $\frac{n^{c-1/2}}{2} - \epsilon$ and distance 1 and answer Path Computation for Set Disjointness of size n with using $o(n^2)$ bits of communication, which is a contradiction. □

4. Conclusion and open problems

We prove lower bounds on the number of expected bits required for any randomized algorithm in a geometric communication network of arbitrary topology with n nodes to solve problems in the asynchronous CONGEST KT1 model. Specifically, we show that Farthest Pair, Convex Hull, and Closest Pair problems require $\Omega(n^2)$ bits of communication. In Farthest Pair and Closest Pair our result holds even if the network is planar.

We present approximation algorithms for geometric communication networks, when the communication is in the asynchronous CONGEST KT1 model. Our algorithms use $o(m)$ words of communication to approximate each of these problems, where m is the number of edges in the network and the network is sufficiently dense.

We also define a new version of the two-party communication problem, Path Computation, where two parties are communicating through a path. We prove a lower bound on the communication complexity of this problem.

Open problems raised by this research are:

(1) A bottleneck in our upper bounds is the computation of the spanning tree which in the asynchronous model appears to require significantly more communication ($\tilde{O}(n^{3/2})$) than what is needed for the synchronous model ($\tilde{O}(n)$). Is this gap necessary?

(2) Euclidean minimum spanning tree (EMST) on a geometric communication network in the fixed-position KT1 model is the minimum spanning tree of the graph, where the weight of an edge between each pair of nodes is the Euclidean distance between the location of two nodes. EMST can be found by the algorithm in [24], since in the fixed-position model each node knows the weights of its incident edges. But if a node's ID is not related to its position, merely knowing the neighbors' IDs may be insufficient to solve this problem in $o(m)$ communication. How can we find EMST efficiently in a geometric communication network when each node's ID is not related to its position?

(3) We showed that Farthest Pair, Convex Hull, Closest Pair problems require $\Omega(n^2)$ bits of communication. What geometric problems in the asynchronous KT1 model in a geometric communication network can be solved in $o(m)$ communication?

(4) How can we extend this work to understanding the time/communication tradeoffs in a synchronous geometric communication network?

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Amir Abboud, Keren Censor-Hillel, Seri Khoury, Near-linear lower bounds for distributed distance computations, even in sparse networks, CoRR, arXiv:1605.05109 [abs], 2016.
[2] Pankaj K. Agarwal, Sariel Har-Peled, Kasturi R. Varadarajan, Approximating extent measures of points, J. ACM (2004) 606–635.
[3] Pankaj K. Agarwal, Sariel Har-Peled, Kasturi R. Varadarajan, Geometric approximation via coresets, in: Combinatorial and Computational Geometry, MSRI, 2005, pp. 1–30.
[4] Pankaj K. Agarwal, Jirí Matousek, Subhash Suri, Farthest neighbors, maximum spanning trees and related problems in higher dimensions, Comput. Geom. (1991) 189–201.
[5] Sunil Arya, Timothy M. Chan, Better ϵ-dependencies for offline approximate nearest neighbor search, Euclidean minimum spanning trees, and ϵ-kernels, in: SOCG 2014, 2014, p. 416.
[6] Baruch Awerbuch, Oded Goldreich, David Peleg, Ronen Vainish, A trade-off between information and communication in broadcast protocols, J. ACM (1990) 238–256.
[7] Maria Fiorina Balcan, Steven Ehrlich, Yingyu Liang, Distributed k-means and k-median clustering on general topologies, in: NIPS'13, Curran Associates Inc., 2013, pp. 1995–2003.
[8] Avrim Blum, Vladimir Braverman, Ananya Kumar, Harry Lang, Lin F. Yang, Approximate convex hull of data streams, in: ICALP 2018, 2018, 21.
[9] Prosenjit Bose, Pat Morin, Ivan Stojmenovic, Jorge Urrutia, Routing with guaranteed delivery in ad hoc wireless networks, in: Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M 1999), ACM, 1999, pp. 48–55.
[10] Timothy M. Chan, Faster core-set constructions and data-stream algorithms in fixed dimensions, Comput. Geom. (2006) 20–35.
[11] Joan Feigenbaum, Sampath Kannan, Jian Zhang, Annotation and computational geometry in the streaming model, Technical Report, Yale University, 2003.
[12] Dan Feldman, Andrew Sugaya, Daniela Rus, An effective coreset compression algorithm for large scale sensor networks, in: IPSN'12, Association for Computing Machinery, 2012, pp. 257–268.
[13] Jie Gao, Leonidas J. Guibas, John Hershberger, Li Zhang, An Zhu, Geometric spanner for routing in mobile networks, in: Proceedings of the 2nd ACM Interational Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001), ACM, 2001, pp. 45–55.
[14] Mohsen Ghaffari, Fabian Kuhn, Distributed MST and broadcast with fewer messages, and faster gossiping, in: DISC 2018, 2018, 30.
[15] Robert Gmyr, Gopal Pandurangan, Time-message trade-offs in distributed algorithms, in: DISC 2018, 2018, pp. 1–18.
[16] Göös Mika, Thomas Watson, Communication complexity of set-disjointness for all probabilities, in: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques(APPROX/RANDOM 2014), in: LIPIcs, vol. 28, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014, pp. 721–736.

[17] Ofer Grossman, Seri Khoury, Ami Paz, Improved hardness of approximation of diameter in the CONGEST model, in: Hagit Attiya (Ed.), 34th International Symposium on Distributed Computing, DISC 2020, October 12-16, 2020, Virtual Conference, in: LIPIcs, vol. 179, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 19.

[18] Sima Hajiaghaei Shanjani, Valerie King, Communication costs in a geometric communication network, in: Proceeding of International Conference on Distributed Computing and Networking (ICDCN'21), ACM, 2021, pp. 36–45.

[19] John Hershberger, Subhash Suri, Adaptive sampling for geometric problems over data streams, in: PODS 2004, 2004, pp. 252–262.

[20] Russell Impagliazzo, Ryan Williams, Communication complexity with synchronized clocks, in: CCC 2010, 2010, pp. 259–269.

[21] Piotr Indyk, Sepideh Mahabadi, Mohammad Mahdian, Vahab S. Mirrokni, Composable core-sets for diversity and coverage maximization, in: PODS 2014, 2014, pp. 100–108.

[22] Daniel Jung, Christina Kolb, Christian Scheideler, Jannik Sundermeier, Competitive routing in hybrid communication networks, in: Algorithms for Sensor Systems - 14th International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS 2018) Revised Selected Papers, in: Lecture Notes in Computer Science, vol. 11410, Springer, 2018, pp. 15–31.

[23] Kalyanasundaram Bala, Georg Schintger, The probabilistic communication complexity of set intersection, SIAM J. Discrete Math. (1992).

[24] Valerie King, Shay Kutten, Mikkel Thorup, Construction and impromptu repair of an MST in a distributed network with o(m) communication, in: PODC 2015, 2015, pp. 71–80.

[25] Eyal Kushilevitz, Noam Nisan, Communication Complexity, Cambridge University Press, 1997.

[26] Shay Kutten, Gopal Pandurangan, David Peleg, Peter Robinson, Amitabh Trehan, On the complexity of universal leader election, in: PODC 2013, 2013, pp. 100–109.

[27] Shay Kutten, Gopal Pandurangan, David Peleg, Peter Robinson, Amitabh Trehan, On the complexity of universal leader election, J. ACM (2015) 7.

[28] Xiang-Yang Li, Approximate MST for UDG locally, in: Proceedings of the 9th Annual International Conference on Computing and Combinatorics (COCOON 2003), in: Lecture Notes in Computer Science, vol. 2697, Springer, 2003, pp. 364–373.

[29] Xiang-Yang Li, Applications of Computational Geometry in Wireless Networks, Springer US, 2004, pp. 197–264.

[30] Ali Mashreghi, Valerie King, Broadcast and minimum spanning tree with o(m) messages in the asynchronous CONGEST model, in: DISC 2018, 2018, 37.

[31] Ali Mashreghi, Valerie King, Brief announcement: faster asynchronous MST and low diameter tree construction with sublinear communication, in: DISC 2019, 2019, 49.

[32] David Peleg, Distributed Computing: A Locality-Sensitive Approach. Discrete Mathematics and Applications, Society for Industrial and Applied Mathematics, 2000.

[33] Shaunna Plunkett-Levin, Problems related to lattice points in the plane, PhD thesis, Cardiff University, 2011.

[34] Sergio Rajsbaum, Jorge Urrutia, Some problems in distributed computational geometry, Theor. Comput. Sci. (2011) 5760–5770.

[35] A.A. Razborov, On the distributional complexity of disjointness, Theor. Comput. Sci. (1992) 385–390.

[36] Peter Robinson, Being fast means being chatty: the local information cost of graph spanners, in: Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, SIAM, 2021, pp. 2105–2120.

[37] Atish Das Sarma, Stephan Holzer, Liah Kor, Amos Korman, Danupon Nanongkai, Gopal Pandurangan, David Peleg, Roger Wattenhofer, Distributed verification and hardness of distributed approximation, in: STOC 2011, 2011, pp. 363–372.

[38] Atish Das Sarma, Stephan Holzer, Liah Kor, Amos Korman, Danupon Nanongkai, Gopal Pandurangan, David Peleg, Roger Wattenhofer, Distributed verification and hardness of distributed approximation, SIAM J. Comput. (2012) 1235–1265.

[39] Yu Wang, Xiang-Yang Li, Ophir Frieder, Distributed spanners with bounded degree for wireless ad hoc networks, Int. J. Found. Comput. Sci. 14 (2) (2003) 183–200.

[40] Hamid Zarrabi-Zadeh, An almost space-optimal streaming algorithm for coresets in fixed dimensions, Algorithmica 60 (1) (2011) 46–59.