SIAM J. DISCRETE MATH.
Vol. 0, No. 0, pp. 000–000

THE IMPACT OF LOCALITY IN THE BROADCAST CONGESTED CLIQUE MODEL*

F. BECKER†, P. MONTEALEGRE‡, I. RAPAPORT§, AND I. TODINCA†

Abstract. The *broadcast congested clique* model (BCLIQUE) is a message-passing model of distributed computation where n nodes communicate with each other in synchronous rounds. First, in this paper we prove that there is a one-round, deterministic algorithm that reconstructs the input graph G if the graph is d-degenerate, and rejects otherwise, using bandwidth $b = \mathcal{O}(d \cdot \log n)$. Then, we introduce a new parameter to the model. We study the situation where the nodes, initially, instead of knowing their immediate neighbors, know their neighborhood *up to a fixed radius r*. In this new framework, denoted BCLIQUE[r], we study the problem of detecting, in G, an induced cycle of length at most k (CYCLE$_{\leq k}$) and the problem of detecting an induced cycle of length at least $k+1$ (CYCLE$_{>k}$). We give upper and lower bounds. We show that if each node is allowed to see up to distance $r = \lfloor k/2 \rfloor + 1$, then a polylogarithmic bandwidth is sufficient for solving CYCLE$_{>k}$ with only two rounds. Nevertheless, if nodes were allowed to see up to distance $r = \lfloor k/3 \rfloor$, then any one-round algorithm that solves CYCLE$_{>k}$ needs the bandwidth b to be at least $\Omega(n/\log n)$. We also show the existence of a one-round, deterministic BCLIQUE algorithm that solves CYCLE$_{\leq k}$ with bandwitdh $b = \mathcal{O}(n^{1/\lfloor k/2 \rfloor} \cdot \log n)$. On the negative side, we prove that, if $\epsilon \leq 1/3$ and $0 < r \leq k/4$, then any ϵ-error, R-round, b-bandwidth algorithm in the BCLIQUE[r] model that solves problem CYCLE$_{\leq k}$ satisfies $R \cdot b = \Omega(n^{1/\lfloor k/2 \rfloor})$.

Key words. broadcast congested clique, induced cycles, graph degeneracy

AMS subject classifications. 68Q25, 68R10, 68Q85

DOI. 10.1137/18M1233534

1. Introduction.

The *broadcast congested clique* BCLIQUE model is a message-passing model of distributed computation where n nodes communicate with each other in synchronous rounds over a *complete network* [1, 2, 4, 6, 7, 8, 16, 19, 21, 24, 30]. The joint input to the n nodes is an undirected graph G on the same set of nodes, with node u receiving the list of its neighbors in G. Nodes have pairwise distinct identities, which are the numbers between 1 and n. All nodes know n, the size of the network.

Each node broadcasts, in each round of the algorithm, a single b-bit message along each of its $n-1$ communication links. The size of the messages is known as the *bandwidth* of the system, and it is a parameter of the model (which could grow with n). Broadcasting is equivalent to writing the messages on a whiteboard, visible to every node. In each round every node produces its message using its input, the contents of the whiteboard, and a sequence of public random bits. An algorithm is correct if it terminates with every node knowing the correct answer with high probability. The

*Received by the editors December 17, 2018; accepted for publication (in revised form) December 12, 2019; published electronically DATE. Some results of this work were presented as a brief announcement in the 2016 *ACM Symposium on Principles of Distributed Computing*. Some other results were presented in the 2018 *Latin American Symposium on Theoretical Informatics*.
https://doi.org/10.1137/18M1233534
Funding: The second author received partial support from Fondecyt 11190482 and from CONICYT + PAI + Convocatoria Nacional Subvención a Instalación en la Academia Convocatoria Año 2017 + PAI77170068. The third author acknowledges partial support from CONICYT via Basal in Applied Mathematics and from Fondecyt 1170021.
†Université d'Orléans, INSA Centre Val de Loire, LIFO EA 4022, Orléans, France (florent.becker@univ-orleans.fr, ioan.todinca@univ-orleans.fr).
‡Facultad de Ingeniería y Ciencias, Universidad Adolfo Ibáñez, Santiago, Chile (p.montealegre@uai.cl).
§DIM-CMM (UMI 2807 CNRS), Universidad de Chile, Santiago, Chile (rapaport@dim.uchile.cl).

round complexity of an algorithm is the maximum number of rounds over all possible input graphs (of size n).

Few fast algorithms are known in the BClique model. In fact, if the bandwidth is $b = \mathcal{O}(\log n)$, then there exist one-round, deterministic algorithms for deciding whether the input graph G has bounded degeneracy [6] or if it contains a fixed forest [16]. If randomization is allowed, then we can decide, in one round, whether the input graph is a cograph [24]. Also, if $b = \mathcal{O}(\text{polylog } n)$, then there is a one-round, randomized algorithm for deciding whether G is connected [1, 2].

One way to increase the computing power of the model is to lift the broadcast restriction and to allow the nodes the possibility of sending *different* messages through different links. This general model, known as *unicast congested clique* [16, 28], gives the possibility to perform a load balancing procedure efficiently. Such enormous intrinsic power has allowed some authors to provide fast algorithms for solving natural problems: an $\mathcal{O}(\log\log\log n)$-round algorithm for finding a 3-ruling set [20], $\mathcal{O}(n^{0.158})$-round algorithms for counting triangles, for counting 4-cycles, and for computing the girth [12], an $\mathcal{O}(1)$-round algorithm for detecting a 4-cycle [12], an $\mathcal{O}(1)$-round algorithm constructing a minimum spanning tree [23], an $\mathcal{O}(\log\log\Delta)$-round algorithm for computing a maximal independent set [18], etc.

Another very natural, much more limited and less dramatic way to increase the computing power of the BClique model *is to expand the local knowledge the nodes initially have about* G. The idea of a constant-radius neighborhood independent of the size of the network is present in the research on local algorithms pioneered by Angluin [3], Linial [27], and Naor and Stockmeyer [29].

We therefore use the KT_r notion, introduced by Awebuch et al. [5], which means *knowledge of topology* up to distance r, excluding edges with both endpoints at distance r. More precisely, we call BClique$[r]$ the extension of the broadcast congested clique model where each node u "sees" (receives as input) the set of *all edges* lying on a path of length at most r, starting in u. Hence, BClique[1] corresponds to the classical broadcast congested clique model, and is simply denoted BClique.

1.1. The problems. In the present paper we study several problems, first, Reconstruction. This problem consists in recovering all the edges of the input graph G in the case that G belongs to a particular class of graphs $\mathcal{G}$. The algorithm must reject when $G \notin \mathcal{G}$. Therefore, provided that $G \in \mathcal{G}$, Reconstruction consists in computing the function $f(G) = E(G)$. The particular class $\mathcal{G}$ that we reconstruct in this paper is the class of d-degenerate graphs. (A graph is d-degenerate if, by iteratively removing vertices of degree *at most* d, we obtain the empty graph.)

The decision problems Cycle$_{\leq k}$ and Cycle$_{>k}$ consist in deciding, respectively, whether the input graph has an *induced* cycle of length at most k and strictly larger than k. (The value k is a fixed parameter). Problems Sub-Cycle$_{\leq k}$ and Sub-Cycle$_{>k}$ are defined in a similar way, but in this case we ask whether the input graph has a cycle as a subgraph (induced or not) of length at most k and strictly larger than k.

The decision problems Even-Sub-Cycle and Odd-Sub-Cycle consist in deciding, respectively, whether the input graph contains a cycle of even length, in the first case, and odd length, in the second. Finally, Connectivity is also a decision problem, where nodes need to decide whether the input graph G is connected.

1.2. Our results. In section 2 we prove that there is a one-round, deterministic algorithm in the BClique model that reconstructs the input graph G if the graph is d-degenerate and rejects otherwise, using bandwidth $b = \mathcal{O}(d \cdot \log n)$. This result represents an improvement with respect to the bandwidth $b = \mathcal{O}(d^2 \cdot \log n)$ presented in [6]. We also prove that our algorithm is tight, even if we allowed randomization and multiple-rounds.

In section 3 we apply our above-mentioned algorithm for solving $\textsc{Cycle}_{\leq k}$ and $\textsc{Sub-Cycle}_{\leq k}$. More precisely, we show the existence of a one-round, deterministic BClique algorithm that solves both problems—which are in fact equivalent—using bandwidth $b = \mathcal{O}(ex(n,k)\log n/n)$, where $ex(n,k)$ is the maximum number of edges of an n-vertex graph not containing, as a subgraph, a cycle of length at most k. From a result of Bondy and Simonovits [10], we conclude that $b = \mathcal{O}(n^{1/\lfloor k/2 \rfloor}\log n)$. On the negative side, we prove that if $\epsilon \leq 1/3$ and $0 < r \leq k/4$, then any ϵ-error, R-round, b-bandwidth algorithm in the BClique$[r]$ model that solves problem $\textsc{Cycle}_{\leq k}$ satisfies $R \cdot b = \Omega(ex(n,k)/n)$. Assuming the Erdős girth conjecture[1] [17] we conclude that $R \cdot b = \Omega(n^{1/\lfloor k/2 \rfloor})$.

In section 4 we develop an algorithm for solving problem $\textsc{Cycle}_{>k}$. For doing this, we give a useful, "local" characterization of graphs which do not have long induced cycles. With such characterization, together with a technique inspired by the linear sketches of [1, 22], we show that if each node is allowed to "see" at distance $\lfloor k/2 \rfloor + 1$, then a polylogarithmic number of bits is sufficient for detecting in two rounds an induced cycle of length strictly larger than k. More precisely, we prove that for every $k \geq 3$, there exists a two-round, randomized algorithm in the BClique$[\lfloor k/2 \rfloor + 1]$ model that solves $\textsc{Cycle}_{>k}$ with high probability using bandwidth $b = \mathcal{O}(\log^4 n)$. On the negative side, we prove that the two characteristics of our algorithm, that is, the two rounds and the local, $\lfloor k/2 \rfloor + 1$ knowledge the nodes have about their neighborhood, are key requirements for achieving a polylogarithmic bandwidth. In fact, we prove that with only one round together with a little bit less local knowledge, any algorithm that solves $\textsc{Cycle}_{>k}$ would need the bandwidth to be $b = \Omega(n/\log n)$. For problem $\textsc{Sub-Cycle}_{>k}$ we apply the degeneracy approach. We show that there exists a one-round, deterministic BClique algorithm that solves problem $\textsc{Sub-Cycle}_{>k}$ and uses bandwidth $b = \mathcal{O}(k \log n)$.

Finally, in section 5, we exhibit one-round, deterministic algorithms for solving problems Even-Sub-Cycle, Odd-Sub-Cycle, and Connectivity. For solving Even-Sub-Cycle we use bandwidth $\mathcal{O}(\log n)$ in the BClique model. For the other two problems, which are related between themselves, the bandwidth is $b = \mathcal{O}(n^{1/r}\log n)$ in the BClique$[r]$ model.

The results of this article are summarized in Tables 1, 2, 3, 4, and 5.

1.3. Related work. The detection of cycles in the input graph G is one of the most frequently addressed problems in the BClique model. The first natural question one can formulate, that is, deciding whether G contains a cycle, has been, until now, the only question amenable to a simple algorithm. In fact, Becker et al. [6] show that a simple set of logarithmic size messages is sufficient to recognize, deterministically and in one round, whether the input graph G is acyclic.

Any other natural question concerning cycles has given strong negative results. Drucker, Kuhn, and Oshman [16] showed that if $\ell \geq 4$, then any algorithm that decides whether the ℓ-node cycle C_ℓ is a subgraph (or an induced subgraph) of the input graph G needs $\Omega(ex(n, C_\ell)/nb)$ rounds where $ex(n, H)$ is the Turán number of H, i.e., the maximal number of edges of an n-node graph which does not contain

[1]This conjecture states that there exist graphs with n vertices and $\Omega(n^{1+1/k})$ edges not containing cycles of length less than or equal to $2k$.

TABLE 1
Results concerning problem CYCLE$_{\leq k}$. *The first row corresponds to the upper-bound (algorithm). The second and third rows correspond to te lower-bounds, which assume the Erdős girth conjecture. Note that* SUB-CYCLE$_{\leq k}$ *is equivalent to* CYCLE$_{\leq k}$.

	BCLIQUE[r]	#Rounds	Bandwidth	Randomized?
Theorem 3.1	$r = 1$	1	$\mathcal{O}(n^{1/\lfloor k/2 \rfloor} \log n)$	Deterministic
Theorem 3.5	$r \leq k/3$	1	$\Omega(n^{1/\lfloor k/2 \rfloor})$	Randomized
Theorem 3.6	$r \leq k/4$	R	$\Omega(n^{1/\lfloor k/2 \rfloor}/R)$	Randomized

TABLE 2
Result concerning problem SUB-CYCLE$_{>k}$.

	BCLIQUE[r]	# Rounds	Bandwidth	Randomized?
Theorem 4.1	$r = 1$	1	$\mathcal{O}(\log n)$	Deterministic

TABLE 3
Results concerning problem CYCLE$_{>k}$. *The first row corresponds to the upper-bound (algorithm). The second row corresponds to the lower bound.*

	BCLIQUE[r]	#Rounds	Bandwidth	Randomized?
Theorem 4.7	$r \geq k/2 + 1$	2	$\mathcal{O}(\log^4 n)$	Randomized w.h.p.
Theorem 4.8	$r \leq k/3$	1	$\Omega(n)$	Randomized ϵ-error

TABLE 4
Result concerning problem EVEN-SUB-CYCLE *(upper bound).*

	BCLIQUE[r]	# Rounds	Bandwidth	Randomized?
Theorem 5.2	$r = 1$	1	$\mathcal{O}(\log n)$	Deterministic

TABLE 5
Results concerning problem ODD-SUB-CYCLE *(upper bounds). Note that deciding the existence of odd cycles is equivalent to deciding bipartiteness.*

	BCLIQUE[r]	# Rounds	Bandwidth	Randomized?
Theorem 5.6	$r \geq 2$	1	$\mathcal{O}(n^{1/r} \log n)$	Deterministic
[1, 2]	$r = 1$	1	$\mathcal{O}(\log^3 n)$	Randomized

a subgraph isomorphic to H. Remark that $ex(n, C_\ell)$ is $\Theta(n^2)$ for odd values ℓ, and $\Theta(n^{1+1/\ell})$ for even values (assuming the Erdős girth conjecture [17]).

Moreover, *even in the very powerful unicast congested clique model*, the algorithms for cycle detection are rather slow. In fact, the best known algorithm for detecting a cycle C_ℓ uses $\mathcal{O}(n^{0.158} \log n)$ rounds for every $\ell \geq 3$ [12], the only exception being the detection of squares C_4, for which a nice $\mathcal{O}(1)$-round algorithm has been devised [12].

1.4. BCLIQUE[r] model. Formally, the BCLIQUE[r] model is defined as follows. There are n nodes which are given distinct identities, which we assume to be numbers between 1 and n. The joint input to the nodes *is a graph G*. More precisely, *each node u receives as input the subgraph of radius r around itself* (i.e., all edges lying on a path of length at most r, starting in u). Nodes execute an algorithm, broadcasting b-bit messages in synchronous rounds. Their goal is to compute some function f that depends on G. When an algorithm stops *every node must know* $f(G)$. Function f defines the problem to be solved. A $0-1$ function corresponds to a decision problem.

There are no restrictions on local computations: each node has unbounded computational power. (In distributed systems the running time of an algorithm is mainly determined by the time needed for the communication.) We are only interested in the number of bits that are communicated by an algorithm, and therefore the *cost* of an algorithm is defined as the bandwidth b times the number of rounds R that the algorithm requires in order to compute $f(G)$. Note that BCLIQUE[1] corresponds to the classical broadcast congested clique model, and it is simply denoted by BCLIQUE.

An algorithm may be deterministic or randomized. We distinguish two subcases of randomized algorithms: the private-coin setting, where each node flips its own coin, and the public-coin setting, where the coin is shared between all nodes. (In this work we consider public-coin algorithms only). An ε-error algorithm $\mathcal{A}$ that computes a function f is a randomized algorithm such that, for every input graph G, $\Pr\{\mathcal{A} \text{ outputs } f(G)\} \geq 1 - \varepsilon$. In the case where $\varepsilon \to 0$ as $n \to \infty$, we say that $\mathcal{A}$ computes f with high probability.

1.5. Some graph terminology. Let $G = (V, E)$ be an undirected graph, and let $u \in V$. We call $N_G(u) = \{v \in V | uv \in E\}$ and $N_G[u] = N_G(u) \cup \{u\}$ the *open and closed neighborhoods* of u, respectively. Similarly, for $U \subseteq V$, $N_G(U) = \cup_{u \in U} N_G(u) - U$ and $N_G[U] = N_G(U) \cup U$ are the *open and closed neighborhoods of* U, respectively. When no ambiguity is possible, we omit the subindices. By extension, we denote $N^r[u]$ the set of vertices at distance at most r from u, and we call it the *closed* r*-neighborhood* of u. Analogously, $N^r(u) = N^r[u] \setminus \{u\}$ is the *open* r*-neighborhood* of u.

Let $\ell > 1$. A set of different vertices $\{x_1, \ldots, x_\ell\}$ of G is called a *path* of length $\ell - 1$ from x_1 to x_ℓ (also called x_1, x_ℓ*-path*) if $x_i x_{i+1}$ are edges of G for every $i \in [\ell - 1] = \{1, \ldots, \ell - 1\}$. If $\ell > 2$ and $x_1 x_\ell$ is also an edge, then $\{x_1, \ldots, x_\ell\}$ is called a *cycle* of length ℓ. In both cases, we call *chord* an edge that connects two nonconsecutive vertices in a path or a cycle.

A graph $H = (V', E')$ is a *subgraph* of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. If for any edge $uv \in E$ with $u, v \in V'$ we also have $uv \in E'$, we say that H is an *induced* subgraph of G or that H is the subgraph of G induced by V'. Given a vertex subset S, the subgraph induced by S is denoted by $G[S]$. We simply write $G - S$ for $G[V \setminus S]$. Also, if F is a subset of edges, we denote by $G - F$ the graph obtained from G by removing the edges of F. If S is a vertex subset of $G = (V, E)$, the *contraction* of S consists in replacing the whole subset S by a unique vertex v_S, such that the neighborhood of v_S in the new graph is $N_G(S)$ while $G - S$ remains unchanged.

A graph G is called *connected* if for every pair of vertices x, y, there exists an x, y-path in G. A vertex set $X \subseteq V(G)$ is *connected* if the induced subgraph $G[X]$ is connected. A *connected component* of G is an inclusion-maximal set of vertices inducing a connected graph. An induced path (resp., cycle) of a graph G is called a *chordless* path (cycle). A graph is called k*-chordal* if it does not contain any induced cycle of length greater than k. The 3-chordal graphs are known as *chordal graphs*. The *girth* of a graph G is the shortest length of a cycle in G.

A graph is d-degenerate if, by iteratively removing vertices of degree *at most* d, we obtain the empty graph. The degeneracy of a graph G is the minimum $d \in \mathbb{N}$ such that G is d-degenerate.

The Turán number $ex(n, k)$ is the maximum number of edges of an n-vertex graph *not containing*, as a subgraph, a cycle of length at most k.

A very helpful result in the study of graphs without short cycles is the one that relates the nonexistence of short cycles in G with the degeneracy of G. More precisely, graphs with no cycles of length at most k (as subgraphs) have a relatively small degeneracy.

PROPOSITION 1.1 (see [16]). *Graphs with no cycles of length at most k are of degeneracy at most $4 \cdot ex(n,k)/n$.*

2. Reconstruction of graphs with bounded degeneracy. We start this section by proving that graphs of degeneracy at most d can be recognized, and even reconstructed, by a one-round algorithm in the BCLIQUE model using bandwidth $b = \mathcal{O}(d \cdot \log n)$. Recall that reconstruction means that, at the end of the algorithm, every node knows all the edges of the input graph.

THEOREM 2.1. *There is a one-round, deterministic algorithm in the* BCLIQUE *model that reconstructs the input graph G if the graph is d-degenerate and rejects otherwise, using bandwidth $b = \mathcal{O}(d \cdot \log n)$.*

For proving Theorem 2.1 we need to prove the existence of a linear function that compresses integer vectors and such that, if the compressed vector x is Boolean and d-sparse, then x can be recovered. (x is d-sparse if its components are nonzero in at most d coordinates.)

$\mathbb{F}_p$ denotes the field of integers *modulo* p, where $p > 0$ is prime. In contrast with the setting of linear sketches [1]—which, in fact, we are going to use in the next sections—the following lemma *does not consider randomization*. Our technique is a form of derandomization of a classical equality testing technique in communication complexity [25].

LEMMA 2.2. *Let $n, d > 0$. There exists a function $f : \mathbb{Z}^n \to \mathbb{F}_p$, for some prime number $p = 2^{\mathcal{O}(d \cdot \log n)}$, such that*

- *f is linear and*
- *f is injective when restricted to d-sparse Boolean inputs.*

Proof. Let $\mathcal{B} = \{b \in \{0,1\}^n : \sum_{i=1}^n b_i \leq d\}$ be the family of d-sparse Boolean vectors of dimension n, and $\mathcal{T} = \{t \in \{-1,0,1\}^n : \exists \text{ distinct } b, b' \in \mathcal{B}, t = b - b'\}$. Let $p = p(n,d)$ be the smallest prime number greater than $(1+n)^{2d} \cdot n$. Recall that this number is at most twice the lower bound by the theorem of Chebyshev [15]. Let now $P(\mathcal{T})$ be the family of polynomials over the field $\mathbb{F}_p$ associating to each $t \in \mathcal{T}$ the polynomial $P(t, X) = \sum_{i=1}^n t_i X^{i-1}$ (values are taken modulo p). Let $\overline{x} = \overline{x}(n,d)$ be the minimum integer in $\mathbb{F}_p$ which is not a root of any polynomial in $P(\mathcal{T})$; $\overline{x}$ exists because there are at most $|\mathcal{T}|$ polynomials in $P(\mathcal{T})$, each polynomial has at most n roots in $\mathbb{F}_p$, and $p > |\mathcal{T}| \cdot n$. We define then, for each $v \in \mathbb{Z}^n$, the function $f(v) = P(v, \overline{x})$. Clearly, f is linear, and, by definition of $\overline{x}$, for any distinct $b, b' \in \mathcal{B}$, we have $f(b) = P(b, \overline{x}) \neq P(b', \overline{x}) = f(b')$. ∎

Proof of Theorem 2.1. The reconstruction algorithm is the following. Each node i broadcasts the message $M_i = (M_i^1, M_i^2) = (d_i, f(a_i))$, where d_i is its degree, a_i is the row of the adjacency matrix corresponding to node i, and f is the function of Lemma 2.2. The number of communicated bits is $\mathcal{O}(d \log n)$. (We recall that, in our model, there are no restrictions on the time complexity of local computations.) Note that the second coordinate M_i^2 of the message sent by node i is an element of $\mathbb{F}_p$, and hence all arithmetic operations performed on M_i^2 are modulo p.

Let $M = \{M_i\}_{i \in V(G)}$ be the *vector of messages*. Nodes use M to prune the graph in at most n phases (computed locally, without any communication). We denote $G_1 = G$ and $M(1) = M$. The input of phase t is the vector of messages $M(t)$ of a graph G_t, starting with $t = 1$. The idea is to look in G_t for a node u_t of degree at most d, together with all its adjacent edges $E(u_t)$ in G_t. Afterward, nodes update $M(t)$ to obtain the new vector of messages $M(t+1)$ consistent with the graph G_{t+1}, which corresponds to the deletion of u_t from G_t. The output of each phase t is $M(t+1)$ together with u_t and $E(u_t)$. If no such u_t exists, the algorithm deduces that the input graph is not d-degenerate and rejects.

The first step of phase t is to look for a node u_t such that $M(t)^1_{u_t} \leq d$. Then, using $M(t)^2_{u_t}$, nodes obtain $E(u_t)$. This can be done by testing, for each $b \in \mathcal{B}$ (the set of Boolean d-sparse vectors), whether $f(b) = M(t)^2_{u_t}$. Since $M(t)^1_{u_t} \leq d$, we have that the adjacency vector of u_t in G_t is d-sparse, and then, according to the injectivity property of f, the *unique* vector of $\mathcal{B}$ such that $f(b) = M(t)^2_{u_t}$ is the adjacency vector of u_t in G_t.

Finally, we can use $E(u_t)$ and $M(t)$ to compute $M(t+1) = \{M(t+1)_i\}_{i \in V(G_{t+1})}$ as follows:

$$M(t+1)_i = \begin{cases} M(t)_i & \text{if } u_t i \notin E(u_t), \\ (M(t)^1_i - 1, M(t)^2_i - f(e_{u_t})) & \text{if } u_t i \in E(u_t), \end{cases}$$

where e_k is the Boolean vector of dimension n with a single one in the kth coordinate.

Note that, if $M(t)$ corresponds to the vector of messages of G_t, then $M(t+1)$ corresponds to the vector of messages of $G_{t+1} = G[V(G_t) - \{u_t\}]$. Let $a(t)_i$, $d(t)_i$ and $a(t+1)_i$, $d(t+1)_i$ be the adjacency vectors and degrees of node i in $G(t)$ and $G(t+1)$, respectively. On one hand, if $u_t i \notin E(u_t)$, then $a(t)_i = a(t+1)_i$ and $d(t)_i = d(t+1)_i$; therefore, $M(t)_i = M(t+1)_i$. On the other hand, if $u_t i \in E(u_t)$, then $M(t+1)_i = d(t+1)_i = d(t)_i - 1 = M(t)_i - 1$ and $a(t+1)_i = a(t)_i - e_{u_t}$. By the linearity of f, we obtain that $M(t+1)_i = f(a(t)_i - e_{u_t}) = M(t)_i - f(e_{u_t})$. ◻

2.1. Lower bounds. Here we show that the previous algorithm is essentially tight, in the following sense. Let $d = d(n)$ be such that $\log(n/d) = \Theta(\log n)$. (This holds, for instance, if $1 \leq d \leq n^\delta$ for any fixed $0 \leq \delta < 1$.) If an algorithm $\mathcal{A}$ reconstructs the class of a d-degenerate graphs in the BClique model, then $\mathcal{A}$ must communicate, at least, the number of bits communicated by the algorithm of Theorem 2.1. Moreover, this holds even for multiround, ϵ-error randomized algorithms.

Theorem 2.3. *Let $d = d(n)$ be such that $\log(n/d) = \Theta(\log n)$ and let $0 < \epsilon < 1/3$. If an ϵ-error, randomized algorithm reconstructs the class of d-degenerate graphs in the* BClique *model using R rounds and bandwidth b, then $Rb = \Omega(d \log n)$.*

Proof. Consider $\mathcal{G}$, the family of n-node bipartite graphs $G = (V_1 \cup V_2, E)$, with $|V_1| = |V_2| = n/2$, and such that every vertex $v \in V_1$ has exactly d neighbors in V_2. (We are assuming, for simplicity and without loss of generality, that n is even.) Clearly, any graph in $\mathcal{G}$ is d-degenerate, and $|\mathcal{G}| = \binom{n/2}{d}^{n/2}$. Hence, $\log|\mathcal{G}| = \Omega(nd \log(n/d)) = \Omega(nd \log n)$.

There must be at least one outcome of the coin tosses for which the correct algorithm reconstructs the input graph in at least $(1-\varepsilon)$ of the cases. Therefore,

$$(n-1)Rb + n = \Omega(\log((1-\varepsilon)|\mathcal{G}|)) = \Omega(\log|\mathcal{G}|) = \Omega(nd \log n).$$

The value $(n-1)Rb + n$ corresponds to the total number of bits received by any node v of the network: $(n-1)Rb$ bits are received from the other nodes and n bits are known by v at the beginning of the algorithm. (This is the indicator function of its neighborhood.) This implies that $R \cdot b = \Omega(\log|\mathcal{G}|/n) = \Omega(d \log n)$. ◻

Remark 2.4. The previous lower bound also holds in the much more powerful unicast congested clique model, where, instead of simply broadcasting, nodes are allowed to send different messages to different nodes.

3. Detection of short cycles. Consider first problem Sub-Cycle$_{\leq k}$. Recall that Proposition 1.1 states that a graph without cycles of length at most k has degeneracy at most $4 \cdot ex(n,k)/n$. Then, using the algorithm of Theorem 2.1, we can give an answer to problem Sub-Cycle$_{\leq k}$.

Theorem 3.1. *There exists a one-round, deterministic* BClique *algorithm that solves problem* Sub-Cycle$_{\leq k}$ *and uses bandwidth* $b = \mathcal{O}(ex(n,k)\log n/n)$.

Proof. By Proposition 1.1, the No-instances of problem Sub-Cycle$_{\leq k}$ must be $(4 \cdot ex(n,k)/n)$-degenerate. Therefore, we simply apply Theorem 2.1 on the input graph *assuming* that its degeneracy is indeed $4 \cdot ex(n,k)/n$. Hence, in one round, and using bandwidth $b = \mathcal{O}(ex(n,k)\log n/n)$, each node either (1) fully reconstructs the graph and decides the property or (2) notices that the degeneracy of the input graph is larger than the degeneracy of any No-instance and concludes that the graph is a Yes-instance. ∎

Remark 3.2. Observe that problems Sub-Cycle$_{\leq k}$ and Cycle$_{\leq k}$ are equivalent. Indeed, if a graph G is a Yes-Instance of Cycle$_{\leq k}$, then G contains an induced cycle of length at most k, which is also a subgraph of G. For the other direction, note that any cycle of length ℓ that is a subgraph of G has an induced cycle of length smaller than or equal to ℓ. Therefore, the upper and lower bounds of these problems coincide.

Corollary 3.3. *There exists a one-round, deterministic problem* Cycle$_{\leq k}$ *that uses bandwidth* $b = \mathcal{O}(ex(n,k)\log n/n)$.

Remark 3.4. Bondy and Simonovits [10] showed that $ex(n,k) = \mathcal{O}(n^{1+2/k})$ if k is even and $ex(n,k) = \mathcal{O}(n^{1+2/(k-1)})$ if k is odd. Therefore, the bandwidth of the one-round, deterministic BClique algorithm that solves both Sub-Cycle$_{\leq k}$ and Cycle$_{\leq k}$ is such that $b = \mathcal{O}(n^{2/k}\log n)$ if k is even and $b = \mathcal{O}(n^{2/(k-1)}\log n)$ if k is odd. On the other hand, the Erdős girth conjecture states that these bounds are tight, implying the results of Table 1. Currently, the best constructions provide a lower bound for $ex(n,k) = \Omega(n^{1+4/(3k-7)})$ if k is even and $ex(n,k) = \Omega(n^{1+4/(3k-9)})$ if k is odd [26].

3.1. Lower bounds. The previous algorithm is rather restrictive. It is deterministic, it works in one-round, and the information each node has about the graph is minimal, consisting of the 1-neighborhood. The question we ask here is the following: is it possible, by lifting previous restrictions, to decrease the *total* number of bits broadcasted by each node? The next results give a negative answer to this question, at least up to logarithmic factors. In other words, the one-round deterministic algorithm based on the degeneracy seems to be the best we can do.

Recall that BClique$[r]$ is the extension of the broadcast congested clique model where each node u receives as input the set of all edges lying on a path of length at most r, starting in u.

Theorem 3.5. *Let* $\epsilon \leq 1/3$, $k \geq 4$, *and* $0 < r \leq k/4$. *Then, any ϵ-error, R-round, b-bandwidth algorithm in the* BClique$[r]$ *model that solves problem* Cycle$_{\leq k}$ *satisfies* $R \cdot b = \Omega(ex(n,k)/n)$.

Proof. The proof of this theorem is very similar to a proof in [16] (where the authors proved the same result for the BClique model, i.e., for $r = 1$). Let $\mathcal{P}$ be an ϵ-error, one-round algorithm in the BClique$[r]$ model that solves, on input graphs of size n, Cycle$_{\leq k}$ in $R(n)$ rounds and using bandwidth $b(n)$. Let $\{G_n\}_n$ be a family of graphs of size n with the maximum number of edges not containing a cycle of length at most k. Notice that, by definition, the number of edges m is such that $m = \mathcal{O}(ex(n,k))$.

We show that we can use $\mathcal{P}$ to solve the communication complexity problem DISJ_m, where two players (Alice and Bob) receive, each, a subset of $\{1,\dots,m\}$ and have to decide if the intersection of their sets is nonempty. The communication complexity of DISJ_m is $\Omega(m)$ [25].

The reduction works as follows: Alice and Bob interpret their inputs as subsets of $E(G_n)$, called E_A and E_B, respectively. Let $\tilde{G}_n$ be the graph composed by two copies of $V(G_n)$, called V_A and V_B, and where each node in V_A is connected by a path of length $2\lfloor k/4\rfloor$ with its corresponding copy in V_B. The edges between nodes in V_A (resp., V_B) are given by E_A (resp., E_B). Notice that $\tilde{G}_n$ has a cycle of length at most k if and only if $E_A \cap E_B \neq \emptyset$.

Let u be a vertex in $V(G_n)$ and let u_A and u_B be the copies of u in V_A and V_B, respectively. We call P_u the path of length $2\lfloor k/4\rfloor$ from u_A to u_B in $\tilde{G}$. For each $u \in V(G)$, Alice (resp., Bob) can simulate $\mathcal{P}$ on each node u_A (resp., $u_B \in V_B$) and in the first (resp., the last) r nodes in P_u, since the messages sent by those nodes only depend on E_A (resp., E_B). We say that Alice (resp., Bob) *owns* those nodes. Observe that if a vertex w belongs to $P_u \setminus (N^r(u_A) \cup N^r(u_B))$, then the message of w can be produced by any node of the graph, since its message does not depend on the edges of $\tilde{G}_n$.

We obtain a communication algorithm for DISJ_m, where at each round Alice and Bob exchange the messages produced by simulating algorithm $\mathcal{P}$ on all the nodes they own. This implies that algorithm $\mathcal{P}$ on graphs of size $2n\lfloor k/4\rfloor$ is such that $2n\lfloor k/4\rfloor \cdot R(2n\lfloor k/4\rfloor) \cdot b(2n\lfloor k/4\rfloor) = \Omega(m(n))$. Therefore, algorithm $\mathcal{P}$ on n-vertex graphs is such that $R \cdot b = \Omega(ex(n,k)/n)$. ∎

In the case where the nodes have more knowledge of the graph, i.e., when $r \leq k/3$, we can obtain lower bounds for one-round algorithms.

THEOREM 3.6. *Let $\epsilon \leq 1/3$, $k \geq 3$, and $0 < r \leq k/3$. Then, any ϵ-error, one-round, b-bandwidth algorithm in the* BCLIQUE$[r]$ *model that solves* CYCLE$_{\leq k}$ *satisfies $b = \Omega(ex(n,k)/n)$.*

Proof. Let $\mathcal{P}$ be an ϵ-error, one-round randomized algorithm in the BCLIQUE$[r]$ model that solves CYCLE$_{\leq k}$ using messages of size $b(n)$. We are going to show that $\mathcal{P}$ can be turned into an ϵ-error protocol solving the two-party communication problem INDEX$_m$, with messages of size $3rn \cdot b((n-2)r+k)$. In INDEX$_m$, Alice receives a Boolean vector $a \in \{0,1\}^m$, Bob receives an index $1 \leq \ell \leq m$, and both of them have to output a_ℓ. It is well-known (see [25]) that the one-round communication complexity of INDEX$_m$ is $\Omega(m)$. This lower-bound also holds to ϵ-error randomized protocols, for $\epsilon \leq 1/3$.

Let G be an n-node graph not containing any cycle of length at most k, such that the number of edges in G is maximum. In other words, if $m = |E(G)|$ is the number of edges of G, then $m = ex(n,k)$. Let us name $e_1,\dots,e_m$ the edges of G. For a vector $a \in \{0,1\}^m$, we call G_a the subgraph of G such that $e_i \in E(G_a)$ if and only if $a_i = 1$. Since G_a is a subgraph of G, all cycles of G_a have length greater than k.

Let us define a family of graphs $G_a(s,t)$, indexed by two parameters $s,t \in \{0,1,\dots,n\}$. Each graph $G_a(s,t)$ has $(n-2)r+k$ vertices, which are numbered from 1 to $(n-2)r+k$. Graph $G_a(s,t)$ is constructed as follows:

- The graph induced by vertices $\{1,\dots,n\}$ is G_a.
- For each $i \in \{1,\dots,n\}$ the vertices $i, i+n, i+2n, \dots, i+(r-1)n$ form a path of length $r-1$, called P_i.

- Vertices $rn+1, \dots, (n-2)r+k$ form path of length $k-2r-1$, called P^*.
- If $s \neq 0$, then vertex $s+(r-1)n$ is adjacent to $rn+1$, and if $t \neq 0$, then vertex $t+(r+1)n$ is adjacent to $(n-2)r+k$.

We claim that $G_a(s,t)$ is a Yes-instance of problem $\text{CYCLE}_{\leq k}$ if and only if s,t are adjacent in G_a. Indeed, if s,t are adjacent, then $P_s \cup P^* \cup P_t$ is an induced cycle of length k. Conversely, by definition of G, if $G_a(s,t)$ contains a cycle a of length at most k, then this cycle must contain the path $P_u \cup P^* \cup P_v$, which already contains k nodes, and then necessarily $\{s,t\}$ is an edge of G_a.

Now observe that for each $i \in \{1,\dots,n\}$ and each $j \in \{1,\dots,r-1\}$, the r-neighborhood of vertex $i+jn \in P_i$ is contained in $N^r(i) \cup P_i$ if $i \notin \{s,t\}$ and is contained in $N^r(i) \cup P^* \cup P_i$ otherwise. Intuitively, the vertices in P_i have at most the same knowledge of G_a as vertex i. Moreover, the vertices in P^* have no knowledge of the edges of G_a.

Given $a \in \{0,1\}^m$ and $s,t \in \{0,1,\dots,n\}$, we call $M_{s,t}(u)$ the message that vertex $u \in V(G_a(s,t))$ produces when protocol $\mathcal{P}$ runs on input $G_a(s,t)$. Previous observations imply the following facts, for each $i \in \{1,\dots,n\}$ and $u \in P_i$:

- $M_{i,t}(u)$ equals $M_{i,0}(u)$ for all $t \in \{0,1,\dots,n\}$ such that $t \neq i$.
- $M_{s,i}(u)$ equals $M_{0,i}(u)$ for all $s \in \{0,1,\dots,n\}$ such that $s \neq i$.
- $M_{s,t}(u)$ equals $M_{0,0}(u)$ for all $s,t \in \{0,1,\dots,n\}$ such that $s \neq i$ and $t \neq i$.

Now let $a\ in\{0,1\}^m$, $\ell \in \{1,\dots,m\}$ be an input of INDEX_m, and call $\mathcal{P}'$ the following one-round protocol: Alice uses $\mathcal{P}$ to produce three messages, for each node $u \in \{1,\dots,rn\}$, that we call $M_1(u) = M_{0,0}(u)$, $M_2(u) = M_{i,0}(u)$, and $M_3(u) = M_{0,i}(u)$, where $i \in \{1,\dots,n\}$ is such that u belongs to P_i. Bob simply communicates ℓ to Alice. Therefore, in the communication round Alice communicates $3rn \cdot b((n-2)r+k)$ bits. After the communication round, Alice outputs a_ℓ. Bob, on the other hand, computes $s,t \in \{1,\dots,n\}$ such that $\{s,t\} = e_\ell$. Then, for each $u \in \{rn+1,\dots,(n-2)r+k\}$ Bob produces $M(u) = M_{s,t}(u)$. Bob can produce these messages without any knowledge of a because the vertices $u \in \{rn+1,\dots,(n-2)r+k\}$ belong to P^*, and then $N^r(u) \subseteq P_s \cup P^* \cup P_t$. Then, Bob gathers the set of messages $\tilde{M} = \{\tilde{M}(u) : u \in \{1,\dots,(n-2)r+k\}\}$, where

- $\tilde{M}(u) = M_1(u)$ if $u \neq P_s \cup P_t$,
- $\tilde{M}(u) = M_2(u)$ if $u \in P_s$,
- $\tilde{M}(u) = M_3(u)$ if $u \in P_t$,
- $\tilde{M}(u) = M(u)$ if $u \in P^*$.

Observe that $\tilde{M}$ is exactly the set of messages that protocol $\mathcal{P}$ produces on input $G_a(s,t)$. Then, Bob runs $\mathcal{P}$ on $\tilde{M}$ obtaining the answer of problem $\text{CYCLE}_{\leq k}$ on input $G_a(s,t)$ with probability ϵ. As we explained above, this answer is affirmative if and only if $a_\ell = 1$. We conclude that $\mathcal{P}'$ solves INDEX_m with messages of size $3rn \cdot b((n-2)r+k)$. Therefore $3rn \cdot b((n-2)r+k) = \Omega(ex(n,k))$. We conclude that $b(n) = \Omega(ex(n,k)/n)$. ∎

Remark 3.7. Bondy and Simonovits [10] showed that $ex(n,k) = \mathcal{O}(n^{1+2/k})$ if k is even, and $ex(n,k) = \mathcal{O}(n^{1+2/(k-1)})$ if k is odd. The Erdős girth conjecture states that this bound is tight, implying the results of Table 1 for $\text{CYCLE}_{\leq k}$. Currently, the best constructions provide a lower bound for $ex(n,k) = \Omega(n^{1+4/(3k-7)})$ if k is even, and $ex(n,k) = \Omega(n^{1+4/(3k-9)})$ if k is odd [26].

From the remarks above, we obtain the following.

COROLLARY 3.8. *Let $\epsilon \leq 1/3$, $k \geq 4$, and $0 < r \leq k/4$. Then, any ϵ-error, R-round, b-bandwidth randomized algorithm in the* BCLIQUE$[r]$ *model that solves*

CYCLE$_{\leq k}$ *requires bandwidth* $b = \Omega(n^{1/(\lfloor k/2 \rfloor)})$. *This bound assumes the Erdős girth conjecture.*

COROLLARY 3.9. *Let* $\epsilon \leq 1/3$, $k \geq 3$, *and* $0 < r \leq k/3$. *Then, any* ϵ*-error, one-round, b-bandwidth randomized algorithm in the* BCLIQUE$[r]$ *model that solves* CYCLE$_{\leq k}$ *requires bandwidth* $b = \Omega(n^{1/(\lfloor k/2 \rfloor)})$. *This bound assumes the Erdős girth conjecture.*

4. Detection of long cycles. Problems SUB-CYCLE$_{>k}$ and CYCLE$_{>k}$, unlike the $\leq$ case of previous section, are obviously very different between themselves. Consider, for example, the n-node complete graph K_n. It has a cycle of length n. But its induced cycles are all triangles. Problem SUB-CYCLE$_{>k}$ can be solved using again the degeneracy approach. In fact, from a result of Birmelé [9], the treewidth (and hence the degeneracy) of graphs with no cycles of length greater than k is at most k.

THEOREM 4.1. *There exists a one-round, deterministic* BCLIQUE *algorithm that solves problem* SUB-CYCLE$_{>k}$ *and uses bandwidth* $b = \mathcal{O}(k \log n)$.

Proof. We simply apply Theorem 2.1 assuming that the input graph is k-degenerate. If it is indeed k-degenerate, then the algorithm reconstructs it (and each node locally verifies whether the input graph has a cycle of length at least $k+1$). Otherwise, it is not k-degenerate, and it must correspond to a Yes-instance. ∎

Recall that graphs without induced cycles of length greater than k are called k-chordal [11]. 3-chordal graphs, i.e., graphs in which every cycle (not necessarily induced) of 4 or more vertices has a chord, are called chordal graphs. It is known that a graph G is chordal if and only if, for each vertex $u \in V$, and each connected component C in $G - N[u]$, the neighborhood $N(C)$ of this component induces a clique in G. This "local" characterization has been exploited by Chandrasekharan and Iyengar [13] for devising a fast parallel algorithm recognizing chordal graphs. We begin this section by extending previous characterization to arbitrary chordalities $k > 3$ in order to take advantage of this in our distributed framework.

Let G be a graph, $u \in V(G)$, and $k > 0$. Let $D_1, \ldots, D_p$ be the p connected components of $G - N^{\lfloor k/2 \rfloor}[u]$ (obtained by removing the vertices at distance at most $\lfloor k/2 \rfloor$ from u). Let H_u^k denote the graph obtained from G by contracting each component D_i into a single node d_i.

LEMMA 4.2. *Let* G *be a graph.* G *is* k*-chordal if and only if, for every* $u \in V(G)$, H_u^k *is* k*-chordal.*

Proof. Suppose first that G is not k-chordal. Let $u \in V(G)$ be a vertex of some chordless cycle of length greater than k. Among the chordless cycles of length greater than k containing u, let C_l be one that has minimum intersection size with $N^{\lfloor k/2 \rfloor}[u]$. We call $u_0, \ldots, u_{l-1}$ the nodes of the cycle C_l, where $u_0 = u$ and u_i is adjacent to $u_{i-1 \bmod l}$ and $u_{i+1 \bmod l}$.

First, observe that if C_l is contained in $N^{\lfloor k/2 \rfloor}[u]$, then necessarily C_l is also contained in H_u^k and then H_u^k is not k-chordal. Therefore, we suppose that at least one vertex of C_l is not contained in $N^{\lfloor k/2 \rfloor}[u]$. The cycle C_l may *leave* the set $N^{\lfloor k/2 \rfloor}[u]$ and then *get in* again, potentially many times. We will show that each time that it does so, the cycle intersects a different connected component of $G - N^{\lfloor k/2 \rfloor}[u]$. Thus, when we contract each such component into a single node, we obtain a chordless cycle of length greater than k. For $t \geq 0$, call $P_1, \ldots, P_t$ the connected components of $G[C_l] - N^{\lfloor k/2 \rfloor}[u]$. Since C_l is chordless, there exists for each $i \in \{1, \ldots, t\}$ a pair $a_i, b_i \in \{0, \ldots, l-1\}$ such that $P_i = \{u_j \in C_l \ : \ a_i < j < b_j\}$ and $|C_l \cap N(P_i)| =$

$\{u_{a_i}, u_{b_i}\}$. Moreover, since $P_i \cap N^{\lfloor k/2 \rfloor}[u] = \emptyset$, u_{a_i} and $u_{b_i}$are at distance exactly $\lfloor k/2 \rfloor$ from u in G.

For $i \in \{1, \dots, t\}$, let D_i be the component of $G - N^{\lfloor k/2 \rfloor}[u]$ that contains P_i. Notice that $N(D_i) \cap C_l = \{a_i, b_i\}$. Indeed, suppose that $|D_i \cap C_l| \geq 3$, and let $J \subset \{1, \dots, l\}$ be such that $j \in J$ if $u_j \in N(D_i)$. Let $m = \min(J)$ and $M = \max(J)$, and call P' a shortest path between u_m and u_M contained in D_i. Clearly, $u_j \notin N(C_i)$ for any $j \in \{0, \dots, m-1, M+1, \dots, l-1\}$. Therefore, the cycle $C' = u_0, \dots, u_{m-1}, u_m, P', u_M, u_{M+1}, \dots, u_{l-1}, u_0$ is chordless. Since u_m and u_M are at distance $\lfloor k/2 \rfloor$ from x in G, C' is of length greater or equal to $k+1$. Then, C' is a chodless cycle of length greater than k that contains u and whose intersection with $N^{\lfloor k/2 \rfloor}[u]$ is strictly smaller than the one of C_l, which contradicts the fact that, by our choice of C_l, the intersection of C_l with $N^{\lfloor k/2 \rfloor}[u]$ is a minimum one. We conclude that each component of $G - N^{\lfloor k/2 \rfloor}[u]$ contains at most one of $\{P_1, \dots, P_t\}$. Therefore, in the graph H_u^k, each path P_i of C_l will be contracted into a different vertex d_i. The new cycle is still a chordless cycle of length greater than k, in the graph H_u^k. So H_u^k is not k-chordal.

For the converse, suppose that there exists $u \in V(G)$ such that H_u^k contains a chordless cycle of length l greater than k; call $C_l = u_0, \dots, u_{l-1}, u_0$ such a cycle. Consider the set $I = \{i \in \{0, \dots, l-1\} \mid u_i \in C_l \backslash N^{\lfloor k/2 \rfloor}[u]\}$ of the indices of nodes in C_l that correspond to contracted components of $G - N^{\lfloor k/2 \rfloor}[u]$. For each $i \in I$, call D_i the connected component of $G - N^{\lfloor k/2 \rfloor}[u]$ corresponding to u_i. Notice that since C_l is chordless, $N(D_i) \cap C_l = \{u_{i-1}, u_{i+1}\}$ (where the subindices are taken modulo l). Let P_i be a chordless u_{i-1}, u_{i+1}-path in $G[D_i \cup \{u_{i-1}, u_{i+1}\}]$.

Call C the cycle of G corresponding to C_l when we replace for each $i \in I$ the subpath u_{i-1}, u_i, u_{i+1} with P_i. Clearly, the length of C is greater than or equal to that of C_l. Also, C is chordless since, for each $i \in I$, the node u_i corresponds to a different component D_i, $|N(P_i) \cap C| = \{u_{i-1}, u_{i+1}\}$, and P_i is a chordless u_{i-1}, u_{i+1}-path. We conclude that G is not k-chordal. ∎

Lemma 4.2 provides us with a strategy for deciding k-chordality, i.e., for deciding whether the input graph G is a No-instance of problem $\text{Cycle}_{>k}$. For doing this every node x must compute the graph H_x^k and then decide whether H_x^k is k-chordal. In order to compute H_x^k, each node x needs first to find the connected components of $G - N^{\lfloor k/2 \rfloor}[x]$.

Let F_x be the set of all edges lying on a path of length at most $\lfloor k/2 \rfloor + 1$ starting in x. We need each node to compute the connected components of $G - F_x$ outside $N^{\lfloor k/2 \rfloor}[x]$.

4.1. Computing the connected components of $G - F_x$. Ahn, Guha, and McGregor provide a probabilistic, one-round algorithm for computing a spanning forest of the input graph G in the BClique model using bandwidth $b = \mathcal{O}(\log^3 n)$ [1]. In their algorithm, each node constructs a message based on its neighborhood and on a sequence of public random coins and broadcasts it to all other nodes. Using all these messages, every node is able to construct a spanning forest of the graph with probability $1 - \epsilon$ for a fixed $\epsilon \leq 1/3$.

We want each node x to compute the connected components of $G - F_x$. Recall that F_x is the set of all edges lying on a path of length at most $\lfloor k/2 \rfloor + 1$ starting in x. We place ourselves in the BClique$[\lfloor k/2 \rfloor + 1]$ model with bandwidth $\mathcal{O}(\log^4 n)$.

We amplify the bandwidth by a $\log(n)$ factor, with respect to the spanning tree algorithm of [1], to ensure that it succeeds with high probability. We do this by running $t = \mathcal{O}(\log n)$ independent copies of the algorithm of [1], each one of them using a different set of random coins. The probability that all copies of the algorithm fail is $\epsilon^t = (1/n)^c$.

Also, every node needs to know all the set of edges F_x, that is, why we choose the BCLIQUE[$\lfloor k/2 \rfloor + 1$] model. Using the spanning forest algorithm of [1], we prove that each node x can construct a spanning forest of $G - F_x$ with high probability.

The key observation is that the messages produced by the vertices are linear functions (with respect to the edges of the graph). Therefore, from the messages of G, each vertex x can compute the messages that the algorithm *would have constructed* on $G - F_x$.

DEFINITION 4.3. *Let $n, k, \delta > 0$. A δ-linear sketch of size k is a function*

$$S : \{0,1\}^{\mathcal{O}(\log n)} \times \{-1,0,1\}^n \to \{0,1\}^k,$$

such that if $S_r = S(r, \cdot)$, then

- *S_r is linear for each $r \in \{0,1\}^{\mathcal{O}(\log n)}$;*
- *if r is chosen uniformly at random, then there is an algorithm that, on input $S_r(x) \in \{0,1\}^k$, returns* ERROR *with probability at most δ and otherwise returns a pair (i, x_i) such that $x_i \neq 0$ and coordinate i is picked uniformly at random among the nonzero coordinates of x. Probabilities are taken over the random choices of r.*

PROPOSITION 4.4 (see [22]). *For each $N, \delta > 0$, there exists a δ-linear sketch of size $\mathcal{O}(\log^2 N \log \delta^{-1})$.*

Let $G = (V, E)$ be a graph of size n, and $x \in V$. We call a^x the *connectivity vector of x in G*, defined as the vector of dimension $\binom{V}{2}$ such that

$$a^x_{\{u,v\}} = \begin{cases} 1 & \text{if } \{u,v\} \in E, x = u \text{ and } u < v, \\ -1 & \text{if } \{u,v\} \in E, x = v \text{ and } u > v, \\ 0 & \text{otherwise.} \end{cases}$$

For $r \in \{0,1\}^{\mathcal{O}(\log n)}$, we say that $S_r(G) = \{S_r(a^x)\}_{x \in V(G)}$ is a δ-*connectivity sketch* of G, where S is a δ-linear sketch. Note that for any $x \in V$, each nonzero coordinate of a^x represents an edge of $N(x)$, and for any $U \subseteq V$ the nonzero coordinates of $\sum_{x \in U} a^x$ are exactly the edges in the cut between U and its complement $V \setminus U$.

The one-round algorithm in the BCLIQUE model devised by Ahn, Guha, and McGregor [1] for computing a spanning forest of G works as follows. Let $t = \lceil \log n \rceil$. Each node computes and sends t independent δ-linear sketches of its connectivity vector, using t random strings $r_1, \ldots, r_t$ picked uniformly at random. Using these messages, any node can compute t independent δ-connectivity sketches of G and therefore it can compute a spanning tree using the following t steps procedure. First, let us denote by $\hat{V}$ the set of *supernodes*, which initially are the n singletons $\{\{u\} | u \in V\}$. At step $0 \leq i < t$, each node samples an incident edge to each set $\hat{v} \in \hat{V}$ using the ith collection of linear sketches $\sum_{x \in \hat{v}} S_{r_i}(a^x)$ and merges the obtained connected components into a single supernode. The procedure finishes before $t = \lceil \log n \rceil$ steps since the number of supernodes at least halves at each step. This idea is behind the proof of the following proposition.

PROPOSITION 4.5 (see Ahn, Guha, and McGregor [1]). *Let $n, \delta > 0$ and $t = \lceil \log n \rceil$. There exists a (centralized) algorithm that receives t independent δ-connectivity sketches of an n-node graph G, produced with $r_1, \ldots, r_t \in \{0,1\}^{\mathcal{O}(\log n)}$ random strings picked uniformly at random, and outputs a spanning forest of G with probability $1 - \delta$.*

LEMMA 4.6. *There is a one-round, randomized algorithm in the* BCLIQUE$[\lfloor k/2\rfloor+1]$ *model that computes, for every node* $x \in V$*, the connected components of* $G - N^{\lfloor k/2\rfloor}[x]$*, using bandwidth* $b = \mathcal{O}(\log^4 n)$ *and with high probability.*

Proof. The algorithm works as follows. First, each node x sends $t = \lceil \log n\rceil$ different $1/n^2$-linear sketches of its connectivity vector a^x, using t random strings $r_1, \ldots, r_t$. Note that each node knows F_x. Observe that the components of $G - N^{\lfloor k/2\rfloor}[x]$ are exactly the components of $G - F_x$ without considering the nodes in $N^{\lfloor k/2\rfloor}[x]$. In the following, we show that after the communication round, each node x can compute a spanning forest of $G - F_x$ with probability at least $1-1/n^2$. Therefore, the whole algorithm succeeds with probability at least $1 - 1/n$.

Let $S_r(G) = (S_r(a^{x_1}), \ldots, S_r(a^{x_n}))$ be one of the $1/n^2$-connectivity sketches of G, produced with the random string r, received in the communication round. Consider, for each $e \in F_x$ and $u \in e$, the vector $b^{u,e}$ of dimension $\binom{n}{2}$, where

$$b^{u,e}_{e'} = \begin{cases} -a^u_e & \text{if } e' = e, \\ 0 & \text{otherwise} \end{cases} \quad \text{for each } e' \in \binom{n}{2}.$$

Let c^u be the connectivity vector of node u in $G - F_x$. Note that, for each $e \in \binom{V}{2}$,

$$c^u_e = a^u_e + \sum_{\{e' \in F_x : u \in e'\}} b^{u,e'}_e = \begin{cases} a^u_e & \text{if } e \in E(G) \setminus F_x, \\ 0 & \text{otherwise.} \end{cases}$$

If we define $S^u_r = S_r(a^u) + \sum_{\{e \in F_x : u \in e\}} S_r(b^{u,e})$, we obtain, by the linearity of S_r, that $S^u_r = S_r(c^u)$. Then, $\{S_r(c^u)\}_{u \in V}$ is a $1/n^2$-connectivity sketch of $G - F_x$ produced with r.

Once the communication round has finished, any node x can obtain t different $1/n^2$-connectivity sketches of $G - F_x$ produced with random strings $r_1, \ldots, r_t$ picked uniformly at random. Therefore, by Proposition 4.5, it can produce a spanning forest of that graph with probability at least $1 - 1/n^2$. ∎

4.2. Deciding *k*-chordality. Now we can show the distributed algorithm for recognizing k-chordal graphs (see Algorithm 4.1).

THEOREM 4.7. *Let* $k \geq 3$*. There exists a two-round, randomized algorithm in the* BCLIQUE$[\lfloor k/2\rfloor + 1]$ *model that recognizes* k*-chordal graphs and thus solves problem* CYCLE$_{>k}$ *with bandwidth* $b = \mathcal{O}(\log^4 n)$ *and high probability.*

Proof. In the first round, each node $x \in G$ computes the connected components of $G - N^{\lfloor k/2\rfloor}[x]$ using the algorithm of Lemma 4.6. After the first round, each node x uses its knowledge of G to locally reconstruct H^k_x by identifying the connected components $D_1, \ldots, D_p$ of $G - N^{\lfloor k/2\rfloor}[x]$ and contracting each D_i into a unique vertex d_i. Note that x sees the edges between D_i and $N^{\lfloor k/2\rfloor}[x]$. Finally, x checks whether H^k_x is k-chordal and communicates the answer in the second round. By Lemma 4.2, the input graph is chordal if and only if each vertex x communicates a YES answer. We emphasize that the second round is needed only because the nodes must all agree on the output.

The algorithm may fail only when some node x fails to compute the components of $G - N^{\lfloor k/2\rfloor}[x]$; this event may occur, from Lemma 4.6, with probability at most $1/n$. ∎

4.3. Lower bounds. Now we are going to prove that the two rounds and the local, $\lfloor k/2\rfloor+1$ knowledge the nodes have about their neighborhood, both features of a previous algorithm, are key requirements for achieving a polylogarithmic bandwidth. In fact, the next result proves that with only one round together with a little bit

Algorithm 4.1 k-chordality.

Round 1

Each node x runs the algorithm of Lemma 4.6 in order to compute the components of graph $G - N^{\lfloor k/2 \rfloor}[x]$

Round 2

Each node x builds H_x^k contracting each component of $G - N^{\lfloor k/2 \rfloor}[x]$

Each node x checks whether H_x^k is k-chordal and broadcasts the answer

less local knowledge, any algorithm that solves $\text{CYCLE}_{>k}$ would need much more bandwidth.

THEOREM 4.8. *Let $\epsilon \leq 1/3$, $k \geq 3$, and $0 < r < k/3$. Then, any ϵ-error, one-round, b-bandwidth algorithm in the* BCLIQUE[r] *model that solves* $\text{CYCLE}_{>k}$ *satisfies* $b = \Omega(n)$.

Proof. Let $\mathcal{P}$ be a protocol solving $\text{CYCLE}_{>k}$ with bandwidth $c(n)$. As we we did for Theorem 3.6, we show that a protocol solving $\text{CYCLE}_{>k}$ with bandwidth $b(n)$ can be used to solve INDEX_m with bandwidth $3rn \cdot b((2n-2)r+k)$, where in this case $m = \Omega(n^2)$.

Let G be a graph on n vertices, partitioned in two sets V_1 and V_2, such that $|V_1| = n_1 = \lfloor n/2 \rfloor$ and $|V_2| = n - n_1$. The vertices of V_1 are numbered from 1 to n_1, and the vertices of V_2 are numbered from $n_1 + 1$ to n. The set of edges is such that V_2 induces a complete graph and V_1, V_2 induce a complete bipartite graph. More formally, the set of edges of G is the set of all pairs $\{v_1, v_2\}$ such that $(v_1, v_2) \in V_2 \times (V_1 \cup V_2)$.

Now let $m = n_1(n_2 - 1)$ and let $e_1, \ldots, e_m$ be the edges of the set $\{\{v_1, v_2\} | v_1 \in \{1, \ldots, n_1\}, v_2 \in \{n_1+1, \ldots, n-1\}\}$. A Boolean vector $a \in \{0,1\}^m$ defines a subgraph G_a of G, containing all edges of G except the edges of the set $\{e_i | a_i = 1\}$. Observe that, in G_a, the set V_1 induces an empty graph, V_2 induces a complete graph, and all vertices of V_1 are connected to vertex $n \in V_2$. Therefore, G_a is a connected and 3-chordal graph for all $a \in \{0,1\}^m$. In particular, G_a is a No-instance of $\text{CYCLE}_{>k}$.

For each $s, t \in \{0, 1, \ldots, 2n\}$ we define $G_a(s,t)$ in the same way as we did it in the proof of Theorem 3.6. More precisely, $G_a(s,t)$ is a graph on $(n-2)r + k$ vertices, numbered from 1 to $(n-2)r + k$. Graph $G_a(s,t)$ is constructed attaching to each vertex v of G_a a path $P_v = i, i+n, \ldots, i+n(r-1)$ of length $r-1$. Vertices numbered $nr+1$ to $(n-2)r+k$ form a path P^*. Finally, if $s \neq 0$, then vertex $nr+1$ is adjacent to s, and if $t \neq 0$, then vertex $(n-2)r+k$ is adjacent to t.

Suppose that $s, t \neq 0$. We claim that $G_a(s,t)$ is a No-instance of problem $\text{CYCLE}_{>k}$ if and only if vertices s and t are adjacent in G_a. Suppose that $G_a(s,t)$ contains a chordless cycle C of length k. Since G_a is 3-chordal, C must contain all vertices in $P_s \cup P^* \cup P_t$ plus a shortest s,t-path P in G_a. Observe that such a path P exists because G_a is connected. Since $P_s \cup P^* \cup P_t$ is a chordless path of length $k-1$, we deduce that P is of length at least 2. We conclude that $G_a(s,t)$ is k-chordal if and only if s and t are adjacent.

The rest of the proof is analogous to the proof of Theorem 3.6. Let $a \in \{0,1\}^m$ and $1 \leq \ell \leq m$ be an input of problem INDEX_m. Alice and Bob run protocol $\mathcal{P}'$ defined in the proof of Theorem 3.6. Let $s \in V_1$ and $t \in V_2$ be such that $\{v_1, v_2\} = e_\ell$. Using the information provided by Alice, Bob can construct the set of messages that the vertices produce running protocol $\mathcal{P}$ on input $G_a(s,t)$. Using this information, Bob can decide with error probability ϵ whether $G_a(s,t)$ contains an induced cycle of length greater than k or not. As we explained above, the answer is affirmative if and only if $a_\ell = 1$, and therefore protocol $\mathcal{P}'$ solves $\textsc{Index}_m$ with error probability $\epsilon \leq 1/3$. We deduce that $3rn \cdot b((2n-2)r+k) = m = \Omega(n^2)$. We conclude that $b(n) = \Omega(n)$. ∎

5. Detection of even/odd cycles. The existence of odd cycles in a graph (as subgraphs, not necessarily induced) is related to other properties of a graph. Of course, bipartite graphs are those not having odd cycles. But as we are going to see here, this is also related to other properties such as connectivity. We consider problems Even-Sub-Cycle and Odd-Sub-Cycle, which consist in deciding, respectively, whether the input graph contains a cycle of odd length, in the first case, and of even length, in the second.

5.1. Even cycles. While graphs without even cycles as subgraphs have constant degeneracy (and are therefore easy to study in our framework), graphs without even induced cycles (called even-hole-free graphs in the literature [11]) contain all chordal graphs. In centralized algorithms, the detection of even-hole-free graphs can be performed in polynomial time. Nevertheless, to our knowledge, the best algorithm that detects even-hole-free graphs requires $\mathcal{O}(n^{11})$ time [14].

The next proposition relates the existence of cycles with the tree-width of a graph.

Proposition 5.1 (see [32]). *For all integers $m > 0$ and $\ell \geq 0$ there exists an integer $k = k(\ell, m) > 0$ such that the tree-width of any graph with at most ℓ vertex-disjoint cycles of length 0 (mod m) is at most k.*

The previous proposition implies that graphs without even cycles (cycles of length $0 \pmod 2$) have constant tree-width. But graphs of tree-width k are of degeneracy at most k, and, more generally, for each fixed graph H, the class of H-minor free graphs is also of bounded degeneracy [31]. Therefore, by using Theorem 2.1, we deduce that graphs without even cycles can be reconstructed in one round in the BClique model using bandwidth $\mathcal{O}(\log n)$.

Theorem 5.2. *There is a one-round, deterministic algorithm in the* BClique *model that solves* Even-Sub-Cycle *using bandwidth $\mathcal{O}(\log n)$.*

5.2. Odd cycles. In contrast with the previous even case, detecting odd induced cycles is equivalent to detecting odd cycles as subgraphs. In fact, every noninduced odd cycle contains a smaller odd cycle. As we are going to show now, there is a strong connection between the existence of odd cycles and the connectivity of the input graph.

Consider the following construction introduced in [1]. For a graph $G = (V, E)$, let $D(G)$ be the graph constructed duplicating all vertices v in V into v_1, v_2. For each edge $\{u, v\} \in E$, we generate two edges in $D(G)$: $\{u_1, v_2\}$ and $\{u_2, v_1\}$.

Proposition 5.3 (see [1]). *The number of connected componentes in $D(G)$ doubles the number of connected components in G if and only if G is bipartite.*

Let $G = (V, E)$ be a graph and $r > 0$. Let σ be some total ordering of E. For each cycle of length at most $2r$, pick the maximum edge of the cycle according to σ. Call $\tilde{E}$ the set of picked edges, and call $\tilde{G} = (V, E - \tilde{E})$.

Proposition 5.4 (see [5]). *G is connected if and only if $\tilde{G}$ is connected; moreover, any spanning forest of $\tilde{G}$ is a spanning forest of G.*

Notice that, from Remark 3.4, we know that $\tilde{G}$ has degeneracy $\mathcal{O}(n^{1/r})$. And, from the previous proposition, it has the same number of connected components as G. We deduce the following theorem.

THEOREM 5.5. *There is a one-round, deterministic algorithm that computes the connected components of the input graph in the* BCLIQUE[r] *model that uses bandwidth* $b = \mathcal{O}(n^{1/r} \log n)$. *The algorithm returns a spanning forest of the input graph.*

Proof. Choose an ordering of the edges of G. For instance, if we denote the edges $e = (u, v)$ with $u < v$, then $(u_1, v_1) < (u_2, v_2)$ if either $u_1 < u_2$ or $u_1 = u_2$ and $v_1 < v_2$. In our algorithm, a node i looks for all cycles of length at most $2r$ in G that contain it. Notice that nodes do this without any communication since they see all neighbors at distance at most r. For each such cycle, i removes the maximum edge according to the edge ordering, obtaining $\tilde{a}_i$, the row of the adjacency matrix of $\tilde{G}$ corresponding to i. Then node i runs the algorithm of Theorem 2.1 on input $\tilde{a}_i$ for a degeneracy $s = \mathcal{O}(n^{1/r})$. Finally, each node uses the messages to reconstruct $\tilde{G}$ and outputs a spanning forest of $\mathcal{G}$ computed locally using any classic algorithm. ∎

Combining Proposition 5.3 with Theorem 5.5 we can solve ODD-SUB-CYCLE.

THEOREM 5.6. *There is a one-round, deterministic algorithm in the* BCLIQUE[r] *model solving* ODD-SUB-CYCLE *with bandwidth* $b = \mathcal{O}(n^{1/r} \log n)$.

Proof. Let G be the input graph. First, each node $v \in V(G)$ computes the row of the adjacency matrix of $D(G)$ corresponding to vertices v_1 and v_2 and simulates the connectivity algorithm on input graph G and in $D(G)$ (playing the role of v_1 and v_2). Upon receiving all messages from other nodes, v can compute a spanning tree of G and $D(G)$ and then compute the number of connected components of G and $D(G)$. Finally, it checks using Proposition 5.3 whether G is bipartite, i.e., whether it contains a cycle of odd length. ∎

6. Conclusion. In this paper we considered the BCLIQUE model. We studied the impact of the local knowledge each node has about its neighborhood on the algorithmic complexity—measured in terms of the bandwidth and/or the number of rounds—of different problems. In some cases the value of r, a parameter of our model, corresponding to the radius that each node knows for free around itself, seems to make a difference. Consider, for instance, the problem of deciding whether the input graph is connected. We showed that we can solve it deterministically, in one round, using bandwidth $\mathcal{O}(n^{1/r} \log n)$. Therefore, for $r \geq 2$, the bandwidth is sublinear. What happens in the classical case, where $r = 1$? Is $\Omega(n)$ a lower bound for the bandwidth in the deterministic, one-round case? A similar phenomenon occurs when one is interested in detecting induced cycles of length at least k. In fact, if $r = \lfloor k/2 \rfloor + 1$, then it is possible to solve this problem in two rounds, with high probability and polylogarithmic bandwidth $b = \mathcal{O}(\log^4 n)$. On the other hand, if $r = \lfloor k/3 \rfloor$, then any one-round algorithm would need the bandwidth to be $b = \Omega(n/\log n)$. For some other problems, such as detecting cycles of length at most k, it seems that the the value of r does not have any impact on the complexity of the problem (provided that r is smaller tan $k/4$).

REFERENCES

[1] K. J. AHN, S. GUHA, AND A. MCGREGOR, *Analyzing graph structure via linear measurements*, in Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, 2012, pp. 459–467.

[2] K. J. AHN, S. GUHA, AND A. MCGREGOR, *Graph sketches: Sparsification, spanners, and subgraphs*, in Proceedings of the 31st Symposium on Principles of Database Systems, 2012, pp. 5–14.

[3] D. Angluin, *Local and global properties in networks of processors*, in Proceedings of the 12th ACM Symposium on Theory of Computing, 1980, pp. 82–93.
[4] H. Arfaoui, P. Fraigniaud, D. Ilcinkas, and F. Mathieu, *Distributedly testing cycle-freeness*, in Proceedings of the 40th International Workshop on Graph-Theoretic Concepts in Computer Science, Lecture Notes in Comput. Sci. 8747, 2014, pp. 15–28.
[5] B. Awerbuch, O. Goldreich, R. Vainish, and D. Peleg, *A trade-off between information and communication in broadcast protocols*, J. ACM, 37 (1990), pp. 238–256.
[6] F. Becker, A. Kosowski, M. Matamala, N. Nisse, I. Rapaport, K. Suchan, and I. Todinca, *Allowing each node to communicate only once in a distributed system: Shared whiteboard models*, Distrib. Comput., 28 (2015), pp. 189–200.
[7] F. Becker, M. Matamala, N. Nisse, I. Rapaport, K. Suchan, and I. Todinca, *Adding a referee to an interconnection network: What can(not) be computed in one round*, in Proceedings of the 25th IEEE International Parallel and Distributed Processing Symposium, 2011, pp. 508–514.
[8] F. Becker, P. Montealegre, I. Rapaport, and I. Todinca, *The simultaneous number-in-hand communication model for networks: Private coins, public coins and determinism*, in Proceedings of the 21st International Colloquium on Structural Information and Communication Complexity, 2014, pp. 83–95.
[9] E. Birmelé, *Tree-width and circumference of graphs*, J. Graph Theory, 43 (2003), pp. 24–25.
[10] J. A. Bondy and M. Simonovits, *Cycles of even length in graphs*, J. Combin. Theory Ser. B, 16 (1974), pp. 97–105.
[11] A. Brandstädt, V. B. Le, and J. P. Spinrad, *Graph Classes: A Survey*, Discrete Math. Appl. 3, SIAM, Philadelphia, 1999.
[12] K. Censor-Hillel, P. Kaski, J. H. Korhonen, C. Lenzen, A. Paz, and J. Suomela, *Algebraic methods in the congested clique*, in Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, 2015, pp. 143–152.
[13] N. Chandrasekharan and S. S. Iyengar, *NC algorithms for recognizing chordal graphs and* k *trees*, IEEE Trans. Comput., 37 (1988), pp. 1178–1183.
[14] H.-C. Chang and H.-I. Lu, *A faster algorithm to recognize even-hole-free graphs*, J. Combin. Theory Ser. B, 113 (2015), pp. 141–161.
[15] P. L. Chebyshev, *Mémoire sur les nombres premiers*, J. Math. Pures Appl., 17 (1852), pp. 366–390.
[16] A. Drucker, F. Kuhn, and R. Oshman, *On the power of the congested clique model*, in Proceedings of the 2014 ACM Symposium on Principles of Distributed Computing, 2014, pp. 367–376.
[17] P. Erdős, *Extremal problems in graph theory*, in Theory of Graphs and its Applications, Proc. Sympos. Smolenice, 1964.
[18] M. Ghaffari, T. Gouleakis, C. Konrad, S. Mitrović, and R. Rubinfeld, *Improved massively parallel computation algorithms for mis, matching, and vertex cover*, in Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC '18, 2018, ACM, New York, pp. 129–138.
[19] S. Guha, A. McGregor, and D. Tench, *Vertex and hyperedge connectivity in dynamic graph streams*, in Proceedings of the 34th ACM Symposium on Principles of Database Systems, 2015, pp. 241–247.
[20] J. W. Hegeman, S. V. Pemmaraju, and V. Sardeshmukh, *Near-constant-time distributed algorithms on a congested clique*, in Proceedings of the 28th International Symposium on Distributed Computing, 2014, pp. 514–530.
[21] S. Holzer and N. Pinsker, *Approximation of distances and shortest paths in the broadcast congest clique*, in Proceedings of the 19th International Conference on Principles of Distributed Systems, LIPIcs Leibniz Int. Proc. Inform. 46, 2016, pp. 1–16.
[22] H. Jowhari, M. Saglam, and G. Tardos, *Tight bounds for lp samplers, finding duplicates in streams, and related problems*, in Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2011, pp. 49–58.
[23] T. Jurdziński and K. Nowicki, *MST in* O(1) *rounds of congested clique*, in Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, PA, 2018, pp. 2620–2632, https://doi.org/10.1137/1.9781611975031.167.
[24] J. Kari, M. Matamala, I. Rapaport, and V. Salo, *Solving the induced subgraph problem in the randomized multiparty simultaneous messages model*, in Proceedings of the 21st International Colloquium on Structural Information and Communication Complexity, 2015, pp. 370–384.
[25] E. Kushilevitz and N. Nisan, *Communication Complexity*, Cambridge University Press, New York, 1997.

[26] F. Lazebnik, V. A. Ustimenko, and A. J. Woldar, *A new series of dense graphs of high girth*, Bull. Amer. Math. Soc., 32 (1995), pp. 73–79.
[27] N. Linial, *Locality in distributed graph algorithms*, SIAM J. Comput., 21 (1992), pp. 193–201.
[28] Z. Lotker, B. Patt-Shamir, E. Pavlov, and D. Peleg, *Minimum-weight spanning tree construction in O (log log n) communication rounds*, SIAM J. Comput., 35 (2005), pp. 120–131.
[29] M. Naor and L. Stockmeyer, *What can be computed locally?*, SIAM J. Comput., 24 (1995), pp. 1259–1277.
[30] B. Patt-Shamir and M. Perry, *Proof-labeling schemes: Broadcast, unicast and in between*, in Proceedings of the 19th International Symposium on Stabilization, Safety, and Security of Distributed Systems, Lecture Notes in Comput. Sci. 10616, 2017, pp. 1–17.
[31] A. Thomason, *The extremal function for complete minors*, J. Combin. Theory Ser. B, 81 (2001), pp. 318–338.
[32] C. Thomassen, *On the presence of disjoint subgraphs of a specified type*, J. Graph Theory, 12 (1988), pp. 101–111, https://doi.org/10.1002/jgt.3190120111.