Tel Aviv University
Raymond and Beverly Sackler Faculty of Exact Sciences
The Blavatnik School of Computer Science

Topics in Distributed Computing and Simultaneous Multi-Party Communication Complexity

by

Orr Fischer

under the supervision of
Dr. Rotem Oshman
and
Prof. Uri Zwick

Thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science

2016

Abstract

Topics in Distributed Computing and Simultaneous Multi-Party Communication Complexity

Orr Fischer
Master of Science
School of Computer Science
Tel Aviv University

Communication complexity plays a powerful role in proving lower bounds in distributed computing, and is one of primary tools to study the protocols with limited bandwidth (CONGEST model). a common technique is to reduces problems to the well-developed theory of 2-player communication complexity, but some of these reductions sometime fall short of capturing an n-player problem.Therefore a natural avenue of study is multi-party communication complexity. In this work we study multi party communication complexity, in hope to improve our understanding of these types of problems. In particular, we studied simultaneous multi-party communication complexity.

For two player simultaneous complexity, it is known that giving the players a public (shared) random string is much more useful than private randomness: public-coin protocols can be unboundedly better than deterministic protocols, while private-coin protocols can only give a quadratic improvement on deterministic protocols.

We extend the two-player gap to multiple players, and show that the private-coin communication complexity of a k-player function f with deterministic cost $D(f)$, has communication cost $\Omega(\sqrt{D(f)})$ for any $k \geq 2$. Perhaps surprisingly, this bound is tight: although one might expect the gap between private-coin and deterministic protocols to grow with the number of players, we show that the All-Equality function, where each player receives n bits of input and the players must determine if their inputs are all the same, can be solved by a private-coin protocol with $\tilde{O}(\sqrt{nk} + k)$ bits. Since All-Equality has deterministic complexity $\Theta(nk)$, this shows that sometimes the gap scales only as the square root of the number of players, and consequently the number of bits each player needs to send actually decreases as the number of players increases.

In Distributed, I include my part in a joint work on the Sinkless Orientation problem in the LOCAL

model, a work which obtained new bounds on the Distributed Lovasz Local Lemma, as well as some additional work done in turning the LOCAL algorithm to the CONGEST model.

Acknowledgements

I would like to thank my advisors, Rotem Oshman and Uri Zwick, for their patience, their dedicated guidance and insights. I would also like to thank my colleges in Finland - Jukka Suomela, Tuomo Lempiinen, Joel Rybicki, and Juho Hirvonen , and my colleges in Switzerland - Roger Wattenhofer, Sebastian Brandt, Barbara Keller and Jara Uitto, for introducing me to the world of distributed, for hosting me at their universities and for sharing with me their interesting research. And finally, I would like to thank to my loving parents and sister, for which this thesis is dedicated to.

Contents

Chapter 1

Simultaneous Multi-Party Communication Complexity

1.1 Introduction

In his seminal '79 paper introducing the notion of two-party communication complexity [Yao79], Yao also briefly considered communication between more than two players, and pointed out "one situation that deserves special attention": two players receive private inputs, and send randomized messages to a third player, who then produces the output. Yao asked what is the communication complexity of the Equality function (called "the identification function" in [Yao79]) in this model: in the Equality function EQ_n, the two players receive vectors $\{0,1\}^n$, and the goal is to determine whether $x = y$.

Yao showed in [Yao79] that EQ_n requires $\Omega(n)$ bits for determistic communication protocols, even if the players can communicate back-and-forth. Using a *shared* random string, the complexity reduces to $O(1)$, and using *private* randomness, but more than a single round, the complexity is $\Theta(\log n)$. In modern nomenclature, the model described above is called *the 2-player simultaneous model*, and the third player (who announces the output) is called the *referee*. Yao's question is then: what is the communication complexity of EQ_n using private randomness in the simultaneous model of communication complexity?

Some seventeen years later, Yao's question was answered: Newman and Sezegy showed in [NS96] that EQ_n requires $\Theta(\sqrt{n})$ bits to compute in the model above, if the players are allowed only private randomness. (Using shared randomness the complexity reduces to $O(1)$, even for simultaneous protocols.) Moreover, Babai and Kimmel showed in [BK97] that for *any* function f, if the deterministic simultaneous complexity of f is $D(f)$, then the private-coin simultaneous communication complexity of f is $\Omega(\sqrt{D(f)})$, so in this sense private randomness is of only limited use for simultaneous protocols.

In this work we study multi-player simultaneous communication complexity*, and ask: how useful are

*We consider the *number-in-hand* model, where each player receives a private input, rather than the perhaps more familiar *number-on-forehead* model, where each player can see the input of all the other players but not its own.

private random coins for more than two players? Intuitively, one might expect that as the number of players grows, the utility of private randomness should decrease. We first extend the $\Omega(\sqrt{D(f)})$ lower bound of [BK97] to the multi-player setting, and show that for any k-player function f, the private-coin simultaneous communication complexity of f is $\Omega(\sqrt{D(f)})$. We then show, perhaps contrary to expectation, that the extended lower bound is still tight in some cases.

To see why this may be surprising, consider the function $\textsc{AllEq}_{k,n}$, which generalizes $\textsc{Eq}_n$ to k players: each player i receives a vector $x_i \in \{0,1\}^n$, and the goal is to determine whether all players received the same input. It is easy to see that the deterministic communication complexity of $\textsc{AllEq}_{k,n}$ is $\Omega(nk)$ (not just for simultanoues protocols), and each player must send n bits to the referee in the worst case. From the lower bound above, we obtain a lower bound of $\Omega(\sqrt{nk})$ for the private-coin simultaneous complexity of $\textsc{AllEq}_{k,n}$. It is easy to see that $\Omega(k)$ is also a lower bound, as each player must send at least one bit, so together we have a lower bound of $\Omega(\sqrt{nk}+k)$. If this lower bound is tight, then the average player only needs to send $O(\sqrt{n/k}+1)$ bits to the referee in the worst-case, so in some sense we even *gain* from having more players, and indeed, if $k=\Omega(n)$, then the per-player cost of $\textsc{AllEq}_{k,n}$ with private coins is constant, just as it would be with shared coins.

Nevertheless, our lower bound *is* nearly tight, and we are able to give a simultaneous private-coin protocol for $\textsc{AllEq}_{k,n}$ where each players sends only $O(\sqrt{n/k}+\log(k))$ bits to the referee, for a total of $O(\sqrt{nk}+k\log\min\{k,n\})$ bits. This matches the lower bound of $\Omega(\sqrt{nk})$ when $k=O(n/\log^2 n)$. We also show that $\textsc{AllEq}_{k,n}$ requires $\Omega(k\log n)$ bits, so in fact our upper bound for $\textsc{AllEq}$ is tight.

We then turn our attention to a harder class of k-player problems: those obtained by taking a 2-player function f and asking "do there exist two players on whose inputs f returns 1?". An example for this class is the function $\textsc{ExistsEq}_{k,n}$, which asks whether there exist two players that received the same input. We show that $\textsc{ExistsEq}_{k,n}$ requires $\tilde{\Theta}(k\sqrt{n})$ bits for private-coin simultaneous protocols, and moreover, any function in the class above has private-coin simultaneous complexity $\tilde{O}(kR(f))$, where $R(f)$ is the private-coin simultaneous complexity of f (with constant error).

1.1.1 Related Work

As we mention above, two-player simultaneous communication complexity was first considered by Yao in [Yao79], and has received considerable attention since. The Equality problem was studied in [BK97, NS96, BW], and another optimal simultaneous protocol is given in [Amb96], using error-correcting codes. In [KNR95], a connection is established between simultaneous and one-round communication complexity and the VC-dimension. [CSWY01, JK09] consider the question of simultaneously solving multiple copies of Equality and other functions, and in particular, [CSWY01] shows that solving m copies of Equality requires $\Omega(m\sqrt{n})$ bits for private-coin simultaneous 2-player protocols.

Multi-player communication complexity has also been extensively studied, but typically in the *number-on-forehead* model, where each player can see the inputs of all the other players but not its own. This model was introduced in [CFL83]; sufficiently strong lower bounds on protocols in this model, even under

restricted (but not simultaneous) communication patterns, would lead to new circuit lower bounds. Simultaneous communication complexity for number-on-forehead is considered in [BGKL04].

In contrast, in this work we consider the *number-in-hand* model, where each player knows only its own input. This model is related to distributed computing and streaming (see, e.g., [WW15], which gives a lower bound for a promise version of Set Disjointness in our model).

An interesting "middle case" between the number-in-hand and number-on-forehead models is considered in [AGM12, BMN$^+$11, BMRT14]: there the input to the players is an undirected graph, and each player represents a node in the graph and receives the edges adjacent to this node as its input. This means that each edge is known to *two* players. This gives the players surprising power; for example, in [AGM12] it is shown that graph connectivity can be decided in a total of $O(n \log^3 n)$ bits using public randomness. The power of private randomness in this model remains a fascinating open question and is part of the motivation for our work.

The functions ALLEQ and EXISTSEQ considered in this work were also studed in, e.g., [CRR14], but not in the context of simultaneous communication; the goal there is to quantify the communication cost of the network topology on communication complexity, in a setting where not all players can talk directly with each other.

1.2 Preliminaries

Notation. For a vector x of length n, we let x_{-i} denote the vector of length $n-1$ obtained by dropping the i-th coordinate of x (where $i \in [n]$).

Simultaneous protocols. Fix input domains $\mathcal{X}_1, \ldots, \mathcal{X}_k$ of sizes $m_1, \ldots, m_k$ (respectively). A *private-coin k-player simultaneous communication protocol* Π on $\mathcal{X}_1 \times \ldots \times \mathcal{X}_k$ is a tuple of functions $(\pi_1, \ldots, \pi_k, O)$, where each π_i maps the inputs $\mathcal{X}_i$ of player i to a distribution on a finite set of messages $\mathcal{M}_i \subseteq \{0,1\}^*$, and O is the referee's output function, mapping each tuple of messages in $\mathcal{M}_1 \times \ldots \times \mathcal{M}_k$ to a distribution on outputs $\{0,1\}$.

We say that Π *computes a function* $f : \mathcal{X}_1 \times \ldots \times \mathcal{X}_k \to \{0,1\}$ *with error* ϵ if for each $(x_1, \ldots, x_k) \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_k$ we have:

$$\Pr_{\{m_i \sim \pi_i(x_i)\}_{i \in [k]}} [O(m_1, \ldots, m_k) \neq f(x_1, \ldots, x_k)] \leq \epsilon.$$

A *deterministic* protocol is defined as above, except that instead of distributions on messages, the protocol maps each player's input to a deterministic message, and the referee's output is also a deterministic function of the messages it receives from the players.

Communication complexity. The *communication complexity* of a protocol Π (randomized or deterministic), denoted by $\mathrm{CC}(\Pi)$, is defined as the maximum total number of bits sent by the players to the referee in

any execution of the protocol on any input.[†]

For a function f, the *deterministic communication complexity of f* is defined as

$$D(f) = \min_{\Pi} \mathrm{CC}(\Pi),$$

where the minimum is taken over all deterministic protocols that compute f with no errors. The *private-coin ϵ-error communication complexity of f* is defined as

$$R_\epsilon(f) = \min_{\Pi:\Pi \text{ computes } f \text{ with error } \epsilon} \mathrm{CC}(\Pi).$$

Individual communication complexity of a player. We let $\mathrm{CC}_i(\Pi)$ denote the maximum number of bits sent by player i to the referee in any execution. For general communication protocols, it could be that the players never simultaneously reach their worst-case message sizes — that is, we could have $\mathrm{CC}(\Pi) < \sum_{i=1}^k \mathrm{CC}_i(\Pi)$. However, with *simultaneous* protocols this cannot happen:

Observation 1. *For any private-coin (or deterministic) simultaneous protocol Π we have $\mathrm{CC}(\Pi) = \sum_{i=1}^k \mathrm{CC}_i(\Pi)$.*

Proof. For each $i \in [k]$, let x_i be some input on which player i sends $\mathrm{CC}_i(\Pi)$ bits with non-zero probability. Then on joint input $(x_1, \ldots, x_k)$, there is a non-zero probability that each player i sends $\mathrm{CC}_i(\Pi)$ bits, for a total of $\sum_{i=1}^k \mathrm{CC}_i(\Pi)$ bits. Therefore $\mathrm{CC}(\Pi) \geq \sum_{i=1}^k \mathrm{CC}_i(\Pi)$. The inequality in the other direction is immediate, as there cannot be an execution of the protocol in which more than $\sum_{i=1}^k \mathrm{CC}_i(\Pi)$ bits are sent. □

In the sequel we assume for simplicity that all players always send the same number of bits, that is, each player has a fixed message size. By the observation above, this does not change the communication complexity.

Maximal message complexity of a protocol. The *maximal message complexity* of a protocol Π is the maximum individual communication complexity over all players. The deterministic maximum message complexity is $D^\infty = \min_{\Pi} \max_i \mathrm{CC}_i(\Pi)$, and the *private-coin ϵ-error maximal message complexity of f* is defined as $R_\epsilon^\infty = \min_{\Pi \text{ computes } f \text{ with error } \epsilon} \max_i \mathrm{CC}_i(\Pi)$

Problem statements. The two main problems we consider in this work are:

- $\textsc{AllEq}_{k,n}(x_1, \ldots, x_k) = 1$ iff $x_1 = \ldots = x_k$, where $x_1, \ldots, x_k \in \{0,1\}^n$;
- $\textsc{ExistsEq}_{k,n}(x_1, \ldots, x_k) = 1$ iff for some $i \neq j$ we have $x_i = x_j$, where $x_1, \ldots, x_k \in \{0,1\}^n$.

We often omit the subscript when the number of players and the input size are clear from the context.

[†] Another reasonable definition for randomized protocols is to take the maximum over all inputs of the *expected* total number of bits sent. For two players this is asymptotically equivalent to the definition above [KN97]. For $k > 2$ players, the expectation may be smaller than the maximum by a factor of $\log(k)$.

1.3 Lower Bound

In this section we extend the lower bound from [BK97] to multiple players, and show that for any k-player function f and constant error probability $\epsilon \in (0, 1/2)$ we have $R_\epsilon(f) = \Omega(\sqrt{D(f)})$.

When proving two-party communication complexity lower bounds, it is helpful to view the function being computed as a matrix, where the rows are indexed by Alice's input, the columns are indexed by Bob's input, and each cell contains the value of the function on the corresponding pair of inputs. The natural extension to k players is a "k-dimensional matrix" (or tensor) where the i-th dimension is indexed by the inputs to the i-th player, and the cells again contain the values of the function on that input combination. For conciseness we refer to this representation as a "matrix" even for $k > 2$ players.

In [BK97] it is observed that the deterministic simultaneous communication complexity of a function is exactly the sum of the logarithms of the number of unique rows and the number of unique columns in the matrix (rounded up to an integer). We generalize the notion of "rows and columns" to multiple players as follows.

Definition 2 (Slice)**.** *Fix a k-dimensional matrix $M \in \{0,1\}^{m_1 \times \ldots \times m_k}$. For a player i and an input (i.e., index) $x_i \in [m_i]$, we define* the (i, x_i)-th slice *of M to be the projection of M onto a $(k-1)$-dimensional matrix $M|_{(i,x_i)} \in \{0,1\}^{m_1 \times \ldots \times m_{i-1} \times m_{i+1} \times \ldots \times m_k}$ obtained by fixing player i's input to x_i. That is, for each $x \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_k$ we have $M|_{(i,x_i)}(x_{-i}) = M(x)$.*

Note that for $k = 2$ and a 2-dimensional matrix M, the $(1, x)$-th slice of M is simply the row indexed by x, and the $(2, y)$-th slice is the column indexed by y.

We assume that the matrices we deal with have *no redundant slices*: there does not exist a pair $(i, x_i), (i, x_i')$ (where $x_i \neq x_i'$) such that $M|_{(i,x_i)} = M|_{(i,x_i')}$. If there are redundant slices, we simply remove them; they correspond to inputs to player i on which the function value is the same, for any combination of inputs to the other players. Such inputs are "different in name only" and we can eliminate the redundancy without changing the communication complexity of the function being computed.

Let $\dim_i(M)$ denote the length of M in the i-th direction: this is the number of possible inputs to player i, after redundant slices are removed (i.e., the number of unique slices for player i in M). We rely upon the following observation, which generalizes the corresponding observation for two players from [BK97]:

Observation 3. *Let $f : \mathcal{X}_1 \times \ldots \times \mathcal{X}_k \to \{0,1\}$ be a k-player function, and let M_f be the matrix representing f. Then in any deterministic protocol for f, each player i sends at least $\log \dim_i(M_f)$ bits in the worst case, and $D(f) = \sum_{i=1}^{k} \lceil \log \dim_i(M_f) \rceil$.*

Proof. Suppose for the sake of contradiction that there is a deterministic protocol Π for f where some player i that always sends fewer than $\lceil \log \dim_i(M_f) \rceil$ bits in Π. For this player there exist two slices (i.e., inputs to player i) $M|_{(i,x_i)}$ and $M|_{(i,x_i')}$, with $x_i \neq x_i'$, on which the player sends the same message. Because we assumed that there are no redundant slices, there exists an input x_{-i} to the other players such that

$M|_{(i,x_i)}(x_{-i}) \neq M|_{(i,x'_i)}(x_{-i})$. But all players send the same messages to the referee on inputs (x_i, x_{-i}) and (x'_i, x_{-i}), which means that on one of the two inputs the output of the referee is incorrect.

This shows that each player i must send at least $\lceil \log \dim_i(M_f) \rceil$ bits in the worst-case. This number of bits from each player is also sufficient to compute f, as the players can simply send the referee their input (after removing redundant slices, the number of remaining inputs is the number of unique slices). Therefore by Observation 1, $D(f) = \sum_{i=1}^{k} \lceil \log \dim_i(M_f) \rceil$. □

In [BK97], Babai and Kimmel prove the following for two players:

Lemma 4 ([BK97])**.** *For any 2-player private-coin protocol* Π *with constant error* $\epsilon < 1/2$*,*

$$\mathrm{CC}_1(\Pi) \cdot \mathrm{CC}_2(\Pi) \geq \Omega(\log \dim_1(M_f) + \log \dim_2(M_f)).$$

Using this property of 2-player protocols, we can show:

Lemma 5. *Let* Π *be a* k*-player private-coin protocol for* $f : \mathcal{X}_1 \times \ldots \times \mathcal{X}_k \to \{0,1\}$ *with constant error* $\epsilon \in (0, 1/2)$*. Then for each* $i \in [k]$*:*

$$\mathrm{CC}_i(\Pi) \cdot \left(\sum_{j \neq i} \mathrm{CC}_j(\Pi) \right) = \Omega(\log \dim_i(M_f)).$$

Proof. Fix a player $i \in [k]$. The k-player protocol Π induces a 2-player protocol Π', where Alice plays the role of player i, and Bob plays the role of all the other players. We have $\mathrm{CC}_1(\Pi') = \mathrm{CC}_i(\Pi)$ and $\mathrm{CC}_2(\Pi') = \sum_{j \neq i} \mathrm{CC}_j(\Pi)$ (recall that we assume the message size of each player is fixed).

The 2-player function computed by Π' is still f, but now we view it as a 2-player function, represented by a 2-dimensional matrix M'_f with rows indexed by $\mathcal{X}_i$ and columns indexed by $\mathcal{X}_1 \times \ldots \times \mathcal{X}_{i-1} \times \mathcal{X}_{i+1} \times \ldots \times \mathcal{X}_k$. Note that $\dim_1(M'_f) \geq \dim_i(M_f)$: if $M_f|_{(i,x_i)}$ and $M_f|_{(i,x'_i)}$ are slices of M_f that are not equal, then the corresponding rows of M'_f, indexed by x_i and x'_i, differ as well. Thus, by Lemma 4,

$$\mathrm{CC}_i(\Pi) \cdot \left(\sum_{j \neq i} \mathrm{CC}_j(\Pi) \right) = \mathrm{CC}_1(\Pi') \cdot \mathrm{CC}_2(\Pi') = \Omega(\log \dim_1(M'_f)) = \Omega(\log \dim_i(M_f)).$$

□

We can now show:

Theorem 6. *For any* k*-player function* f *and constant error* $\epsilon < 1/2$ *we have* $R_\epsilon(f) = \Omega(\sqrt{D(f)})$*.*

Proof. Let Π be an ϵ-error private-coin simultaneous protocol for f. By the lemma, for each $i \in [k]$ we have

$$\mathrm{CC}_i(\Pi) \cdot \left(\sum_{j=1}^{n} \mathrm{CC}_j(\Pi) \right) \geq \mathrm{CC}_i(\Pi) \cdot \left(\sum_{j \neq i} \mathrm{CC}_j(\Pi)) \right) = \Omega \left(\log \dim_i(M_f) \right).$$

Summing across all players, we obtain

$$\left(\sum_{i=1}^{n} \mathrm{CC}_i(\Pi) \right) \cdot \left(\sum_{j=1}^{n} \mathrm{CC}_j(\Pi) \right) = \Omega \left(\sum_{i=1}^{n} \log \dim_i(M_f) \right),$$

that is, by Observations 1 and 3,

$$\mathrm{CC}(\Pi)^2 = \Omega \left(D(f) \right).$$

The theorem follows. □

From the theorem above we see that the *average* player must send $\Omega(\sqrt{D(f)/k})$ bits. But what is the relationship between the *maximum* number of bits sent by any player in a private-coin protocol and a deterministic protocol for f? This question is mainly of interest for non-symmetric functions, since for symmetric functions all players must send the same number of bits in the worst-case.

Theorem 7. *For any k-player function f and constant error ϵ we have $R_\epsilon^\infty(f) = \Omega(\sqrt{D^\infty(f)/k})$.*

Proof. Recall that by Observation 3, $D^\infty(f) = \max_i \log \dim_i(M_f)$. Let i be a player maximizing $\log \dim_i(M_f)$. As we showed in the proof of Theorem 6, for this player we have in any private-coin simultaneous protocol Π:

$$\mathrm{CC}_i(\Pi) \cdot \left(\sum_{j=1}^{n} \mathrm{CC}_j(\Pi) \right) = \Omega \left(\log \dim_i(M_f) \right) = \Omega(D^\infty(f)).$$

Now let ℓ be the player with the maximum communication complexity in Π, that is, $\mathrm{CC}_j(\Pi) \leq \mathrm{CC}_\ell(\Pi)$ for each $j \in [k]$. We then have

$$\mathrm{CC}_i(\Pi) \cdot \left(\sum_{j=1}^{n} \mathrm{CC}_j(\Pi) \right) \leq \mathrm{CC}_\ell(\Pi) \cdot (k-1)\mathrm{CC}_\ell(\Pi) < k\mathrm{CC}_\ell^2(\Pi).$$

Combining the two, we obtain $\mathrm{CC}_\ell(\Pi) = \Omega\left(\sqrt{D^\infty(f)/k}\right)$, which proves the theorem. □

Lower Bound of $\Omega(k \log n)$ for $\textsc{AllEq}_{k,n}$

We next show that in the specific case of ALLEQ, each player needs to send at least $\Omega(\log n)$ bits, yielding a lower bound of $\Omega(k \log n)$. This improves on the lower bound of Theorem 6 when $k = \Omega(n/\mathrm{polylog}(n))$,

and will show that the protocol in the next section is optimal.

Theorem 8. *For any constant $\epsilon < 1/2$ we have $R_\epsilon(\text{ALLEQ}_{k,n}) = \Omega(k \log n)$.*

Proof. Fix a player $i \in [k]$. To show that player i must send $\Omega(\log n)$ bits, we reduce from EQ_n, but this time our reduction constructs a *one-way protocol*, where Alice, taking the role of player i, sends a message to Bob, representing all the other players and the referee; and Bob then outputs the answer. It is known that EQ_n requires $\Omega(\log n)$ bits of communication for private-coind protocols — this is true even with unrestricted back-and-forth communication between the two players [KN97]. The lower bound follows.

Let Π be a simultaneous private-coin protocol for $\text{ALLEQ}_{k,n}$. We construct a one-way protocol for EQ_n as follows: on input (x, y), Alice sends Bob the message that player i would send on input x in Π. Bob computes the messages each player $j \neq i$ would send on input y, and then computes the output of the referee; this is the output of the one-way protocol. Clearly, $\text{ALLEQ}_{k,n}(x, y, \ldots, y) = \text{EQ}_n(x, y)$, so the one-way protocol succeeds whenever Π succeeds.

□

The lower bounds above use a series of 2-player reductions; they do not seem to exploit the full "hardness" of having k players with their own individual private randomness. This makes it more surprising that the lower bounds are tight, as we show in the next section.

1.4 Tight Upper bound for ALLEQ

In this section, we show that the lower bound proven in section 1.3 is tight for $\text{ALLEQ}_{k,n}$. This is done by showing a protocol with maximal message of size $O(\sqrt{\frac{n}{k}} + \log(\min(n, k)))$ bits per player, and $O(\sqrt{nk} + k \log(\min(n, k)))$ bits of communication overall.

Theorem 9. *There exists a private-coin one sided error randomized simultaneous protocol for $\text{ALLEQ}_{k,n}$ with maximal message of size $O(\sqrt{\frac{n}{k}} + \log(\min(n, k))) = O(\sqrt{\frac{D^\infty(\text{ALLEQ}_{k,n})}{k}} + \log(\min(n, k)))$ bits per player.*

Corollary 10. *There exists a private-coin one sided error randomized simultaneous protocol for $\text{ALLEQ}_{n,k}$ of cost $O(\sqrt{nk} + k \log(\min(n, k))) = O(\sqrt{D(\text{ALLEQ}_{k,n})} + k \log(\min(n, k)))$.*

We note that the *deterministic* communication complexity of $\text{ALLEQ}_{n,k}$ is $\Theta(nk)$, and hence also $D^\infty(\text{ALLEQ}_{k,n}) = \Theta(n)$. This follows immediately from Observation 3.

Our randomized private-coin protocol is as follows.

Error-correcting codes. In the first step of the protocol, each player encodes its input using a predetermined error correcting code, and uses the encoded string as the new input. We review the definition of an error correcting code. In the definition below, n and k are the standard notation for error correcting codes,

which we keep for the sake of consistency with the literature in coding; they are unrelated to the parameters n, k of the communication complexity problem and will be used in this context in the following definition only.

Definition 11 ([MS81])**.** *$M \subseteq \{0,1\}^n$ is called an $[n,k,d]$-*code *if it contains 2^k elements (that is, $|M| = 2^k$) and $d_H(x,y) \geq d$ for every two distinct x, y, where d_H is the Hamming distance. For a $[n,k,d]$ code, let $\delta = \frac{d}{n}$ denote the* relative distance *of the code.*

An $[n,k,d]$-code maps each of 2^k inputs to a code word of n bits, such that any two distinct inputs map to code words that have large relative distance. We use a simple error-correcging code (see [MS81]), which was also used in [Amb96]:

Lemma 12 ([MS81], Theorem 17.30[‡])**.** *For each $m \geq 1$ there is a $[3m, m, \frac{m}{2}]$-code.*

The relative distance of the code in lemma 12 is $\delta = \frac{(1/2)m}{3m} = \frac{1}{6}$.

When the players use the code from Lemma 12 to encode their inputs, each player's input grows by a constant factor (3), while the relative Hamming distance of any two differing inputs becomes at least δ. Let $N = 3n$ denote the length of the encoded inputs, and let $\bar{x}_i$ denote the encoding of player i's input x_i.

Partitioning into blocks. After computing the encoding of their inputs, each player splits its encoded input into blocks of $L = \lceil \frac{N}{k} \rceil$ bits each, except possibly the last block, which may be shorter. For simplicity we assume here that all blocks have the same length, that is, L divides n. Let $b = N/L$ be the resulting number of blocks; we note that $b \leq \min(3n, k)$. Let $B_i(\ell) \in \{0,1\}^L$ denote the ℓ-th block of player i.

Because any two differing inputs have encodings that are far in Hamming distance, we can show that two players with different inputs will also disagree on many *blocks*:

Lemma 13. *For any two players i, j such that $x_i \neq x_j$, we have $|\{\ell \in [b] st B_i(\ell) \neq B_j(\ell)\}| \geq \delta b$.*

Proof. Assume for the sake of contradiction that $|\{\ell \in [b] | B_i(\ell) \neq B_j(\ell)\}| < \delta b$.

Let $\Delta = \{s \in [N] st \bar{x}_i(s) \neq \bar{x}_j(s)\}$ be the set of coordinates on which players i, j disagree. By the properties of the error correcting code, $|\Delta| \geq \delta N$.

Now partition Δ into disjoint sets $\Delta_1, \ldots, \Delta_b$, where each Δ_ℓ contains the locations inside block ℓ on which the encoded inputs disagree. Each Δ_ℓ contains between 0 and N/b coordinates, as the size of each block is $L = N/b$. By our assumption, there are fewer than δb blocks that contain any differences, so the number of non-empty sets Δ_ℓ is smaller than δb. It follows that $|\Delta| < \delta b \cdot (N/b) = \delta N$, which contradicts the relative distance of the code. $\square$

[‡]The theorem in [MS81] gives a general construction for any distance up to $1/2$; here we use distance $1/6$.

Comparing blocks. Our goal now is to try to find two players that disagree on some block. We know that if there are two players with different inputs, then they will disagree on many different blocks, so choosing a random block will expose the difference with good probability. In order to compare the blocks, we use an optimal 2-player private-coin simultaneous protocol for EQ:

Theorem 14 ([BK97] Theorem 1.5)**.** *There exists a private-coin one-sided error simultaneous protocol for the two player function EQ_m of cost $\Theta(\sqrt{m})$. If the inputs are equal, the protocol always outputs "Equal". If the inputs are not equal, then the protocol outputs "Equal" with probability $< 1/3$.*

Remark 1. *We refer here to the symmetric variant of the equality protocol in Remark 3.3 of [BK97], in which both Alice and Bob use the same algorithm to compute their outputs.*

We proceed as follows. Each player i chooses a block $\ell \in [b]$ at random. The player applies Alice's algorithm from [BK97]'s 2-player equality protocol on the chosen block $B_i(\ell)$, and sends the output to the referee, along with the index ℓ of the block. In this process each player sends $O(\sqrt{\frac{n}{k}} + \log(\min(n,k)))$ bits, because the length of a block is $L = O(n/k)$, and $b \leq \min(3n, k)$.

The referee receives the player's outputs $o_1, ..., o_k$, and for each pair that chose the same block index, it simulates [BK97]'s 2-player equality referee. If for all such pairs the output is 1 then the referee also outputs 1, otherwise it outputs 0. Let us denote by $Ref(o_1, \ldots, o_k)$ the referee's output function.

Analysis of the error probability. Note that if all inputs are equal, then our protocol always outputs 1: the EQ_L protocol from [BK97] has one-sided error, so in this case it will output 1 for any pair of blocks compared. On the other hand, if there exist two different inputs, we will only detect this if (a) the two corresponding players choose a block on which their encoded inputs differ, and (b) the EQ_L protocol from [BK97] succeeds and outputs 0. We show that this does happen with good probability:

Lemma 15. *If $\text{ALLEQ}(x_1, ..., x_k) = 0$, then the protocol outputs 0 with probability at least $\frac{2}{3}\delta(1 - e^{-\frac{1}{2}})$.*

Proof. Since there are at least two distinct input strings, there exists an input string received by at most half the players. Let i be a player with such a string, and let $j'_1, ..., j'_{\frac{k}{2}}$ be $\frac{k}{2}$ players that disagree with player i's input.

Let A_t be the event that player j'_t chose the same block index as player i. Then

$$\Pr\left[Ref(o_1, ..., o_k) = 0\right] \geq \Pr\left[Ref(o_1, ..., o_k) = 0 \,\middle|\, \bigcup_{t=1}^{k/2} A_t\right] \cdot \Pr\left[\bigcup_{t=1}^{k/2} A_t\right].$$

We bound each of these two factors individually.

Since all A_t's are independent, and for a fixed t we have $Pr[A_t] = \frac{1}{b}$ then

$$\Pr\left[\bigcup_{t=1}^{k/2} A_t\right] = 1 - \left(1 - \frac{1}{b}\right)^{k/2} \geq 1 - \left(1 - \frac{1}{k}\right)^{k/2} \geq 1 - \left(e^{-1/k}\right)^{k/2} = 1 - e^{-1/2}.$$

Next, let us condition on the fact that some specific A_r occurred. Given that at least one of the A_t's occurred, let A_r be such an event, that is, player r chose the same block as player i.

Clearly, conditioning on A_r does not change the probability of each block being selected, because the blocks are chosen uniformly and independently: that is, for each $i, r \in [k]$ and $\ell \in [b]$,

$$\Pr\left[\text{player } i \text{ chose block } \ell \mid A_r\right] = \frac{1}{b}.$$

Therefore, by Lemma 13, given the event A_r, players i and r disagree on the block they sent with probability at least $(\delta b)/b = \delta$. Whenever i and r send a block they disagree on, the protocol from [BK97] outputs 0 with probability $2/3$. So overall,

$$\Pr\left[Ref(o_1, ..., o_k) = 0 \,\middle|\, \bigcup_{t=1}^{\frac{k}{2}} A_t\right] \geq \frac{2}{3}\delta.$$

Combining the two yields $Pr[Ref(o_1, ..., o_k) = 0] \geq \frac{2}{3}\delta(1 - e^{-\frac{1}{2}})$. □

Proof of Theorem 9. By lemma 15, if $\textsc{AllEq}(x_1, ..., x_k) = 0$ the algorithm errs with constant probability. If $\textsc{AllEq}(x_1, ..., x_k) = 1$ then since $\forall_{i,p,p'} B_p(i) = B'_p(i)$, and the fact that [BK97]'s protocol is a one-sided error protocol, the global protocol will always output 1, which is the correct value. Since this is a one-sided error protocol with constant error probability, this protocol can be amplified by repeating the protocol in parallel a constant number of times, so that the error probability becomes an arbitrarily small constant, and the communication is only increased by a constant factor. □

1.5 On ExistsEq

The upper bound of Section 1.4 reduces the AllEq problem to a collection of 2-player Eq problems, which can then be solved efficiently using known protocols (e.g., from [BK97]). This works because asking whether *all* inputs are equal, and finding any pair of inputs that are not equal is sufficient to conclude that the answer is "no". What is the situation for the ExistsEq problem, where we ask whether there *exists* a pair of inputs that are equal? Intuitively the approach above should not help, and indeed, the complexity of ExistsEq is higher:

Theorem 16. *If $k \leq 2^{n-1}$, then $R_\epsilon(\textsc{ExistsEq}_{k,n}) = \Omega(k\sqrt{n})$ for any constant $\epsilon < 1/2$.*

Proof. We show that each player i must send $\Omega(\sqrt{n})$ bits in the worst case, and the bound then follows by Observation 1. The proof is by reduction from 2-player $\textsc{Eq}_{n-1}$ (we assume that $n \geq 2$).

Let Π be a private-coin simultaneous protocol for $\textsc{ExistsEq}_{k,n}$ with error $\epsilon < 1/2$. Consider player 1 (the proof for the other players is similar). Assign to each player $i \in \{3, \ldots, k\}$ a unique label $b_i \in \{0,1\}^{n-1}$ (this is possible because $k \leq 2^{n-1}$.

We construct a 2-player simultaneous protocol Π' for EQ_{n-1} with error $\epsilon < 1/2$ as follows: on inputs $(x, y) \in \{0,1\}^{n-1}$, Alice plays the role of player 1 in Π, feeding it the input $1x$ (that is, the n-bit vector obtained by prepending '1' to x); Bob plays the role of player 2 with input $1y$; and the referee in Π' plays the role of all the other players and the referee in Π, feeding each player i the input $0b_i$, where b_i is the unique label assigned to player i.

After receiving messages from Alice and Bob, and sampling the messages players $3, \ldots, k$ would send in Π when given the inputs described above, the referee computes the output of Π and that is also the output of Π'.

Because we prefixed the true inputs x, y with 1, and players $3, \ldots, k$ received inputs beginning with 0, we have $\text{EXISTSEQ}(1x, 1y, 0b_3, \ldots, 0b_k) = \text{EQ}(x, y)$. Therefore Π' succeeds whenever Π does, and in particular it has error at most ϵ. By the lower bound of [BK97], then, player 1 must send $\Omega(\sqrt{n})$ bits in the worst case. □

We note that if $k \geq 2^n$ then EXISTSEQ is trivial, as there must always exist two players with the same input. (The depencence of k on n in Theorem 16 can be improved to $k \leq (1 - o(1)) \cdot 2^n$ by assigning inputs to player $3, \ldots, k$ more cleverly.)

The lower bound above is tight up to $O(\log k)$, and indeed we can state something more general: for any 2-player function $f : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, let $\exists_k f$ be the k-player function that outputs 1 on input $(x_1, \ldots, x_k)$ iff for some $i, j \in [k]$ we have $f(x_i, x_j) = 1$.

Lemma 17. *For any 2-player function f and $k \geq 2$ we have $R_{k^2\epsilon}(\exists_k f) = O(k \cdot R_\epsilon(f))$.*

Proof. Let $\Pi = (\Pi_A, \Pi_B, O)$ be a 2-player private-coin simultaneous protocol for f with communication complexity $R_\epsilon(f)$.

We construct a protocol Π' for $\exists_k f$ as follows: on input $(x_1, \ldots, x_k)$, each player i samples two messages, $M_A^i \sim \Pi_A(x_i)$ and $M_B^i \sim \Pi_B(x_i)$, and sends them to the referee. The referee samples outputs $Z_{i,j} \sim O(M_A^i, M_B^j)$ for each $(i, j) \in [k]^2$ $(i \neq j)$ and outputs "1" iff for some pair $Z_{i,j} = 1$.

If $\exists_k f(x_1, \ldots, x_k) = 1$, then there exist i, j such that $f(x_i, x_j) = 1$, and for this pair of players we have $\Pr[Z_{i,j} = 0] \leq \epsilon$. Therefore the referee outputs "1" except with probability ϵ.

On the other hand, if $\exists_k f(x_1, \ldots, x_k) = 0$, then for every pair i, j of players we have $f(x_i, x_j) = 0$, so $\Pr[Z_{i,j} = 1] \leq \epsilon$. By union bound, in this case the probability that the referee outputs "1" is bounded by $\binom{k}{2}\epsilon < k^2\epsilon$. □

To handle the increased error of the protocol for $\exists_k f$, we can use a protocol for f that has error $O(1/k^2)$; this is achieved by taking a constant-error protocol for f, executing it $O(\log k)$ independent times, and taking the majority output [KN97]. We obtain the following:

Theorem 18. *For any 2-player function f, $k \geq 2$, and constant $\epsilon < 1/2$ we have $R_\epsilon(\exists_k f) = O(k \log k \cdot R_\epsilon(f))$.*

Corollary 19. *For* EXISTSEQ *we have* $R_\epsilon(\text{EXISTSEQ}) = O(k \log k \sqrt{n})$*, matching our lower bound up to a logarithmic factor.*

1.6 Conclusion

In this work we extended the classical results of Babai and Kimmel [BK97] to the multi-player setting, and gave a tight bound for the gap between private-coin and deterministic communication complexity in the simultaneous setting. We showed that contrary to our initial expectations, the gap does not grow larger with the number of players, and indeed the per-player gap can shrink as the number of players grows. We also addressed a class of functions defined by taking a two-party function and asking whether there are two players whose inputs cause it to output 1.

Our work leaves open the interesting question of simultaneous lower bounds for the model considered in [AGM12, BMN$^+$11, BMRT14], where each player represents a node in a graph and is given the edges adjacent to that node. Our techniques do not apply to this scenario because of the sharing of edges between players. Indeed, the connectivity problem for this model (see [McG]) cannot be addressed by reductions from two-player communication complexity, because it is easy for two players: we can simply have Alice and Bob compute spanning forests for their part of the input and send them to each other, at a total cost of $O(n \log n)$. Thus, further multi-party techniques need to be developed to address the hardness of connectivity.

Chapter 2

Sinkless Orientation

In this chapter we study the problem of Sinkless Orientation. The work was done as part of a joint work with Sebastian Brandt, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela and Jara Uitto [BFH$^+$16]. The work gives a new type of lower bound in the LOCAL model, and gave a $\Omega(\log\log n)$ lower bound on the distributed Lova'sz Local Lemma, improving the previous lower bound of $\Omega(\log^* n)$ by [CPS14]. A followup work by [CKP16] showed using similar techniques the first exponential separation between randomized and deterministic round complexity in the LOCAL model, solving an important open problem in distributed computing. This section details the reduction to the local lemma, and a deterministic $O(\log n)$ solution for this problem, and shows how to solve variants of the problem such as sinkless sourceless orientation. We further show that the algorithm can be adapted to the CONGEST model, in which each edge has a restricted bandwidth per round of communication. This result is new and unpublished prior to this thesis. We note that the $O(\log n)$ deterministic algorithm in the LOCAL model in section 2.3 is nearly identical to an algorithm in [GS17] and was developed independently and in parallel of it.

2.1 Preliminaries

The Distributed LOCAL Setting In this model we have n processors, who are connected in a communication network graph G. The processors only know their neighbors and are oblivious to the rest of the network. The processors may communicate with each other in synchronous rounds, and in each round each processors may send a message (of unlimited size) to each of its neighbors, and it may also output a value of its choice, terminating its role in the network. It is further assumed that each processor has unlimited computational power.

Denote the graph output as a vector of size n, such that in the i'th coordinate contains the output of vertex i. A local distributed problem P is defined as a set of valid graph outputs per communication graph G. The goal of the processors is to output so that the output of the graph would be valid. The round complexity

of a deterministic protocol is defined as the number of rounds it takes for all the processors running the protocol to terminate. For randomized algorithms this is defined as the worst case round over the coins of the processors. The round complexity of P defied as the minimal round complexity of a protocol solving P with probability at least $\frac{2}{3}$.

Notations Denote δ as the minimal degree in G, and denote $dist(v, u)$ as the distance between u and v in the communication graph G. For a set of vertices A, let $dist(A, u) = \min_{v \in A} dist(v, u)$.

2.2 Sinkless Orientation

The sinkless orientation problem is defined as follows:

Problem 20. *Given a graph $G = (V, E)$ with minimal degree $\delta \geq 3$, output an orientation for each edge such that the minimal out degree of each vertex is at least* 1.

In order for this problem to be well defined, there should exist a sinkless orientation for any such graph. We show this is the case in the following claim, which also gives us a better understanding of the sinkless orientation's nature.

Claim 21. *Given a graph $G = (V, E)$ with minimal degree $\delta \geq 2$, there exists a sinkless orientation.*

Proof. Fix a connected component G_1 in G. Since the degree of each vertex is at least 2, there exists a cycle in G_1. Fix such a cycle C, and consider an orientation of its edges such that the cycle is oriented in one of its two directions. For each vertex not in the cycle we do a "reverse BFS" from the cycle to the rest of the graph, orienting the edges inwards. Formally let $Level_i(C) = \{v | dist(C, v) = i\}$ be a partition to BFS layers from the cycle. For each vertex in $Level_i$ there exists a neighbor in $Level_{i-1}$. Orienting each vertex towards its parent, each vertex in level ≥ 1 has an outgoing edge, while vertices on $Level_0$ are on the cycle and therefore have an outgoing edge. By orienting the rest of the edges arbitrarily, an sinkless orientation is obtained. □

Claim 22. *Fix an oriented graph edges as $\vec{E}$ and the associated directed graph $\vec{G}$. $\vec{G}$ is a sinkless orientation if and only if every $v \in \vec{G}$ has a path to a cycle in $\vec{G}$.*

Proof. If v has a directed path to a cycle, trivially v has an outwards edge. Assume $\vec{G}$ has no sinks, and let v_1 be a vertex in G. Consider a path $v_1 v_2 v_3 ... v_{n+1}$, which is obtained by following an arbitrary outgoing edge from v_i in the i'th step. Such an edge always exists since all vertices have an outwards edge. Since the path is of length $n + 1$, the path contains a cycle. □

The original paper [BFH$^+$16] shows a $\Omega(\log \log n)$ lower bound on the the Symmetric Distributed Algorithmic Local Lemma problem [BFH$^+$16], using a $\Omega(\log \log n)$ lower bound on randomized sinkless orientation on developed in the paper. For this purpose, a reduction from LLL to 3-regular 3-edge-colored is shown in section 2.4.

Problem 23. *Given a d-regular graph G, $d \geq 3$, output a sinkless orientation on G.*

A reduction for the case $d \geq 4$ is immediate. The interesting case is $d = 3$, which is solved by a combinatorial reduction to $d = 4$. This shows that an LLL instance is hard even for dependency $d = 3$.

2.3 An $O(\log n)$ Deterministic Algorithm Problem in the LOCAL model

2.3.1 Problem Definition

This section describes an algorithm to solve the sinkless orientation problem under the LOCAL model in $O(\log n)$ rounds. This was developed independently and in parallel to a nearly identical algorithm in [GS17].

The key observation for the algorithm is that for G such that $\delta \geq 3$ any vertex in G is of distance at most $\log n$ from a cycle of length at most $\log n$. The main idea of the algorithm is for each vertex to find such a cycle close to it, and to attempt to direct itself to the cycle, and direct the cycle in one of its two orientations. Of course, as all vertices attempt this simultaneously there could be contradicting manners in which an edge can be ordered to be directed. For this, we introduce order priorities. In the algorithm, when a vertex v sends an order to direct a (distant) edge, it assigns the order a priority $k = v.id$. At the end of the algorithm, an edge will determine its final orientation by the orientation order with the maximal priority it received during the course of the algorithm.

2.3.2 Algorithm Description

Algorithm 1 SinklessLocal($v, Neigbors, Id$)

- **For** $i = 1, ..., \log n$:
  - Gathers neighbors of distance i.
- Let c be the lexicographically smallest observed cycle in the $\log n$ neighborhood, and let p be the lexicographically smallest shortest path from v to c.
- With order priority $= v.id$, Send direction orders to Direct p in the direction of c, and to direct the cycle c in the following direction: Let a be the vertex with the highest Id in the c, and let b be the neighbor in c with the larger Id., and direct it in the direction so that $a \to b$.
- After $3 \log n$ rounds, orient each unoriented edge towards the higher Id vertex, and terminate.

Lemma 24. *If $\delta \geq 3$, Then each vertex there exists a cycle of size $\log n$ is at distance at most from a cycle of length at most $\log n$.*

Proof. Assume by contradiction that there exists $v \in V$ for which that does not hold, meaning that in the $\log n$ neighborhood of v is a tree. Consider its $\log n$ neighborhood, and denote $Level_i(v) = \{u | dist(v, u) = i\}$ the BFS layers of the vertex v. Since there are no cycles, the $\log n$ neighborhood of v is a tree, and since the degree of each vertex is at least 3, $|level_{i+1}| \geq 2 \cdot |level_i|$. The $\log n$ neighborhood has $1 + 2^{\log n} = n + 1 > n$ vertices, contradiction, since there are only n vertices in the graph. $\square$

Corollary 25. *$SinklessLocal$ terminates after $O(\log n)$ rounds. At the end of the algorithm each vertex has an outgoing edge.*

Proof. The protocol explicitly terminates after $3 \log n$ rounds. Note that both the neighbor gathering phase and the direction order phase complete successfully as from the previous lemma, the neighborhood gathering phase terminates after $\log n$ rounds, since there exists a cycle of size $\log n$ in its $\log n$ neighborhood. This also means that the directing phase terminates after $\log n$ rounds. Let $v \in V$, and let k be the maximum priority v encounters during the algorithm. From the definition of orientation priorities, there exists some vertex u with $id = k$ that sent the orientation order to e. Since u sent orientation orders to a cycle and the path to it, u also sent an orientation order to an edge e leaving v with priority also k. and since k is the maximum priority seen by v and the fact that id's are unique, then this order has the highest priority on e, and thus at the end of the algorithm e is oriented outwards from v. □

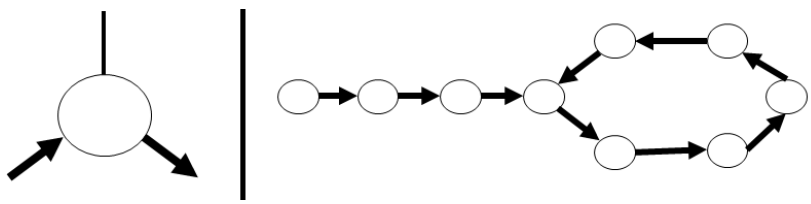

Figure 2.1: As an orientation order consists of a directed path connected to a cycle (where each vertex has out-degree > 1), each order encountered by a vertex has an order to orient one of its an edges outwards from it, so such an edge associated with the highest id encountered is oriented outwards at the end of the algorithm.

Theorem 26. *$SinklessLocal$ is a procedure that outputs a sinkless orientation and terminates after $O(\log n)$ rounds.*

2.4 Randomized Sinkless algorithm using Algorithmic local Lemma

This section aims to reduce the sinkless orientation problem for d regular graph to the Distributed Lovaz Algorithmic local Lemma (LLL) with variable dependency d (To be defined later). The first result gives a reduction from LLL to any d using an assumption of d-edge coloring, which is sufficient for the original lower bound purposes, and the second result shows how to lift the need of d-edge coloring using maximal matching, and extending the results to many related problems, such as sinkless sourceless orientation.

2.4.1 Distributed Algorithms for the Lovasz Local Lemma

The setting of the Symmetric Algorithmic Lovasz Local Lemma by Moser and Tardos [MT10] is the following:

Problem 27 (Moser and Tardos)**.** *Let P be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be an set of events, $p \in [0,1]$ such that for any event $A \in \mathcal{A}$, $Pr(A) \leq p$, and each event in $\mathcal{A}$ share variables with at most d other events. Given that $ep(d+1) < 1$, Find an assignment for all variables such that no event in $\mathcal{A}$ occurs.*

the dependency graph $G_{\mathcal{A}}$, is defined as a graph where each $A \in \mathcal{A}$ is a vertex, and two vertices have an edge if they share common variables (are Dependant). In the distributed model we assume that our communication graph is the dependency graph, and our goal is for each vertex to output a value of its variables such that the problem no $A \in \mathcal{A}$ occurs.

In [CPS14], two distributed LLL algorithms were developed for the LOCAL model, with slightly different dependence assumptions. Given $ep(d+1) < 1$, there is an algorithm for solving a LLL instance in running in $O(log^2(d) \cdot \log n)$ rounds, and given $epd^2 < 1$ it can be done in $O(\log n)$ rounds.

Definition 28. *Given a d-regular graph $G = (V, E)$, define a random variable for each edge to be its direction, and for each $v \in V$ define a bad event $A_v =$"v is a sink" ($out\text{-}deg(v) = 0$). Let G_{Sink} be the dependency graph of these bad events and probability space.*

Lemma 29. *The following observations hold for G_{Sink}:*

1. *A_v shares variables with A_u if and only if $(u, v) \in E$.*
2. *The dependency graph G_{Sink} is isomorphic to G, and $\varphi(v) = A_v$ is an isomorphism function.*
3. *$Pr(A_v) = \frac{1}{2^d}$*

Proof. For (1), since the independent variables are the edge directions, if u, v are not neighbors, then "u is a sink" and "v is a sink" do not share a common variable. In the other direction if u, v are neighbors and u is a sink, then the edge between u and v must be directed (v, u), meaning v isn't a sink, thus A_v and A_u are dependent. Note that (2) follows immediately from the first claim and the definition of the dependency graph.

Claim (3) holds since G is a d-regular graph, so the probability of a bad event A_v is $\frac{1}{2^d}$, as the edge directions are chosen i.i.d uniformly. □

2.4.2 $O(\log n)$ Algorithm on d-regular graphs with $d > 3$

This subsection gives the description of the algorithm for $d > 3$. This follows directly by applying the LLL algorithm.

Lemma 30. *Given a 4-regular graph, a sinkless orientation can be found in $O(\log n)$ rounds using the distributed algorithmic local lemma.*

Proof. For an i.i.d random choice of edge directions, it holds that for any A_v, $P(A_v) = \frac{1}{16}$, meaning $ep(d+1) = \frac{e \cdot 5}{16} < 1$, therefore by the distributed algorithmic local lemma it can be solved in $O(\log n)$ rounds. □

Corollary 31. *Given a d-regular graph, $d > 3$, a sinkless orientation can be found in $O(\log n)$ rounds using the distributed algorithmic local lemma.*

Proof. For $4 \leq d \leq 8$ the algorithm in [CPS14] terminates after $O(log^2(d) \cdot \log n) = O(\log n)$ rounds w.h.p, and gives direction assignments to each edge. For $d > 8$ the stronger $epd^2 < 1$ holds so using the second LLL algorithm in[CPS14], an LLL can be obtained in $O(\log n)$ rounds. □

2.4.3 $O(\log n)$ Algorithm on 3-regular graphs

Unfortunately, for the 3-regular case, the Local Lemma is not directly applicable since $ep(d+1) > 1$. We show a reduction to the 4-regular case.

The key observation is that two 3-degree adjacent vertices act together like a single 4-degree vertex: if we contract the two adjacent vertices into one by their common edge, the new vertex has 4 edges, and if the graph was directed the new vertex isn't a sink if and only if in the original graph we can redirect the inner edge such that they are both not a sink.

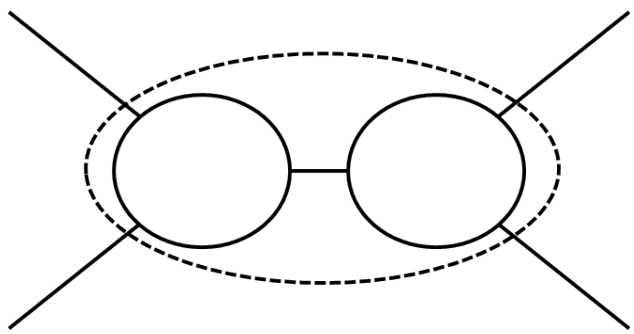

Figure 2.2: Contracted 3-degree vertices act like a 4-degree vertex

Reduction for 3-edge-colored graph

For the purpose of showing lower bounds on the local lemma in the LOCAL model, it is sufficient to show it for 3-edge-colored 3-regular graphs. Using the observation above, we note that each of the three edge colors form a perfect matching.

Algorithm 2 Graph reduction algorithm($DependancyGraph$)

- Take M to be the perfect matching which is the set of edges of color 1.
- Contract vertices sharing edges $(u, v) \in M$. (contraction may create double edges)
- Denote G' as the multi-graph created by this process

We note that each vertex v in G' has degree exactly $d(v) = 4$, and that Any distributed algorithm on G' can be simulated by G with constant overhead, by having a contracted node in G' be simulated by the two nodes in G that form it.

Lemma 32. *Given a sinkless orientation on G', a sinkless orientation can be found on the network G in constant time.*

Proof. Consider a partial orientation on G induced by the uncontracted edges in G (which are the edges in G'), and giving them the same orientation. If G' is sinklessly oriented, then each matched pair has an edge leaving one of its vertices. Direct each of the contracted edges so that both matched vertices touching each edge are satisfied. This is possible since if a contracted vertex w_{uv} isn't a sink in G', then it has an outgoing edge, meaning that in G either u or v have an outgoing edge, and therefore isn't a sink. Assume w.l.o.g that v isn't a sink, then directing the edge $\{v, u\}$ towards u makes it so both v and u are not sinks □

All that remains is to obtain a sinkless orientation for multi-graphs of degree 4. This can be obtained using the distributed LLL similar to the 4-regular case. By choosing the edge direction at random, the probability that a vertex is a sink is exactly $\frac{1}{2^4}$, on the other hand, each vertex has of at most 4 distinct neighbors, therefore it has degree at most 4 in the dependency graph. since $ep(d+1) = e\frac{5}{16} < 1$, we obtain a sinkless orientation via the use of the distributed local lemma, which concludes the reduction.

Algorithm for general 3-regular graphs

In the previous case, we used a perfect matching to reduce to the 4-regular case. a perfect matching is sufficient, but don't necessarily exist, or are hard to obtain in general 3-regular graphs. In this subsection we show that a maximal matching is sufficient for the reduction. This is not needed for the lower bound on the Lovasz Local Lemma, but can be used as a tool to solve variants of the sinkless orientation, such as sinkless sourceless orientation. To find a maximal matching M, and contract matched vertices into a new 4-degree vertex (figure 2.2). After that, the graph is almost a 4-regular graph, up to the unmatched isolated vertices. These will be forced by the reduction to act as an edge between two neighboring 4-degree vertices (figure 2.3).

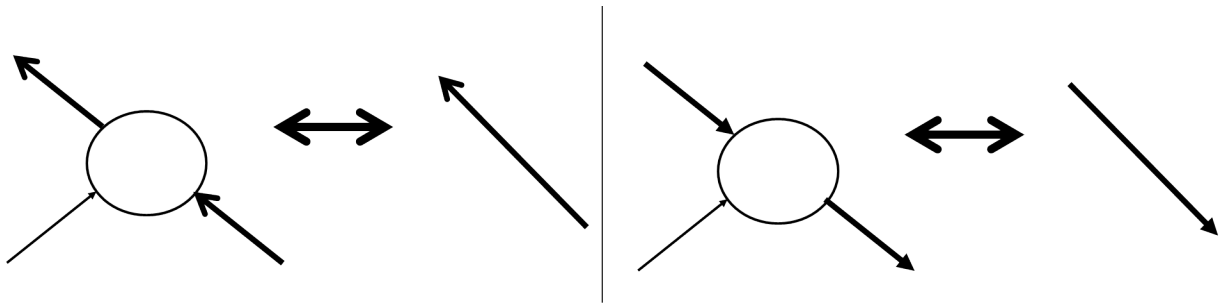

Figure 2.3: After fixing one of their edges inward, we can make isolated unmatched vertices acts as an edge between two matched neighboring vertices

Algorithm 3 Graph reduction algorithm($DependancyGraph$,M)

Denote the multi-graph created by this process G':

- Contract vertices sharing edges $(u, v) \in M$. (contraction may create double edges)
- For each isolated unmatched vertex u, replace two arbitrary edges $(u, v_1), (u, v_2)$ with an edge (v_1, v_2), and remove the vertex and third edge

Lemma 33. *Given an orientation on G' such that each $u \in \{v|d(v) = 4\}$ is not a sink, a sinkless orientation can be found on the network G in constant time.*

Proof. For each edge in G' created by two edges of an isolated vertex u in G, direct them both as directed by G' as shown in figure 2.3. Direct the third edge of an isolated vertex into the isolated vertex. The remaining non-matching edge in G exist in G' and direct them the way it is directed in G'. For isolated vertices, we note that no matter how we orient an edge in G', it has an outgoing edge. Since it is a sinkless orientation in G', then each matched pair has an edge leaving one of its vertices. Direct the edges in the matching M so that both matched vertices are satisfied. This is possible since if a contracted vertex w_{uv} isn't a sink in G', then in G either u or v isn't a sink. □

Lemma 34. *Directing G' to an orientation such that all 4-degree vertices are not a sink can be reduced to solving an LLL instance of the same size.*

Proof. This is similar to the 4-regular case. Choose a random orientation for all edges in G'. In this case only 4-degree vertices sinks are considered bad events, and we note that the dependency graph has dependency 4, and each two vertices in the dependency graph are at most of distance 2 in the communication graph. Bad events as before have $Pr(A_v) \leq \frac{1}{16}$ as each of their edges are chosen uniformly and independently. Since $ep(d+1) = e \cdot \frac{1}{16} \cdot 5 < 1$, it follows that the LLL algorithm converges after $O(log^2(d) \cdot \log n) = O(\log n)$ rounds, meaning we get an orientation such that any $u \in \{v|d(v) = 4\}$ isn't a sink. From Lemma 2.4.3 we can obtain a sinkless orientation on G in constant time. □

Algorithm 4 ThreeRegularReduction($DependancyGraph$)

- Obtain a MIS M of G.
- Calculate G' using M
- Run an LLL algorithm on the graph G'
- Translate the directions into G in the following manner:
  - If $e \in G'$ is an edge in both G and G' direct it in the same manner as G'.
  - If $e \in G'$ was created by two edges of an isolated vertex, direct them the same way as the edge in G' as shown in figure 2.3.
  - If $e = (u, v)$ ia an edge in M, then direct it so that both u and v aren't sinks

Theorem 35. *A 3-regular graph sinkless orientation instance can be reduced to a Distributed Lovasz Local Lemma instance in $O(log^* n)$ rounds.*

Proof. Obtaining an MIS takes $O(\log^* n)$ rounds on bounded degree graphs, reduction to G' takes $O(1)$ rounds, And by using the LLL algorithm on G' the instance can be solved. □

Corollary 36. *Sinkless orientation on 3-regular graphs can be solved in $O(\log n)$ rounds.*

2.4.4 Conclusion

We've studied the sinkless orientation problem, shown a deterministic algorithm, and shown a reduction to the Lovasz Local Lemma in regular graphs, overcoming a difficulty in the case $d = 3$, where the LLL criteria is too weak for it. We note that using the distributed algorithmic Lovasz Local Lemma we can solve similar problems to the sinkless orientation, such as sinkless sourceless orientations, even if at first they don't seem to meet the criteria of the LLL using the tools in the final subsection. This is done iteratively by constructing a new multi-graph by gaining a maximal matching, amplifying the success probability of each bad event, until the LLL criteria is met.

2.5 An $O(\log n)$ Deterministic Algorithm For Sinkless Orientation in the CONGEST model

A natural question regarding Sinkless orientation is if it can be done efficiently while restricting the edge bandwidth in each round to be $O(\log n)$ (i.e the CONGEST model). For a randomized protocol, the LLL algorithm of [CPS14] gives us an immediate positive answer. This is due to the fact that their algorithm consists of finding a maximal independent set of bad event vertices in the dependency graph and re-sampling them, but the bad events defined in section 2.4 clearly form an independent set, as if v is a sink, then all of its neighbors in G must not be sinks, as they have an outgoing edge to v, and so by lemma 29 part (2) the claim follows. For a deterministic algorithm, naively implementing the algorithm in section 2.3 would not work, as each vertex learns its $\log n$ neighborhood, and therefore has a very high message length. in the following section we give an efficient algorithm for sinkless orientation in a restricted bandwidth model to the deterministic case as well.

In this algorithm, having vertices attempt to orient a path connected to a cycle plays a key role as well, but we can't have each vertex learning such a path and cycle, as it might be too expensive to do in parallel.

Definition 37. *v is a dominating vertex if it has the maximum id in its $2\log n$ neighborhood*

Each dominating vertex can safely perform a BFS on its $\log n$ neighborhood without interference from other dominating vertices. It is not clear how such a vertex learns the path and cycle in a restricted bandwidth setting. For this we define cycle witnesses

Definition 38. *Denote a vertex u a witness cycle from a vertex v with regards to a BFS protocol originating from v if it has received at least two incoming messages during the BFS.*

We note that these messages must be received in two different edges.The key idea of the algorithm is as follows: each dominating vertex v orients a path and a cycle. It does so by performing a BFS on its $\log n$ neighborhood, finding a cycle witness u. By definition a cycle witness has two edges in which it has received a BFS message. By considering two tokens - a red token and a blue token, each starting at u. u sends one token through each such edge, and then the token backwards traversal through the BFS up to the root v. We note that the edges on which a token was traveresed form a path connected to a cycle (See figure). Oriented correctly, this yields a directed path starting from v, connected to an oriented cycle that contains u. The process above satisfies the dominating vertices, and the part of their $\log n$ neighborhood participating in the path and cycle defined above (denoted active nodes in the algorithm). The rest of the vertices are not satisfied by this process, but as will be shown later, is sufficient if each non-active (passive) vertex picks the edge from its edges that is closest to the largest id in its $2 \log n$ neighborhood ($passiveParent$) and orient it outwards.

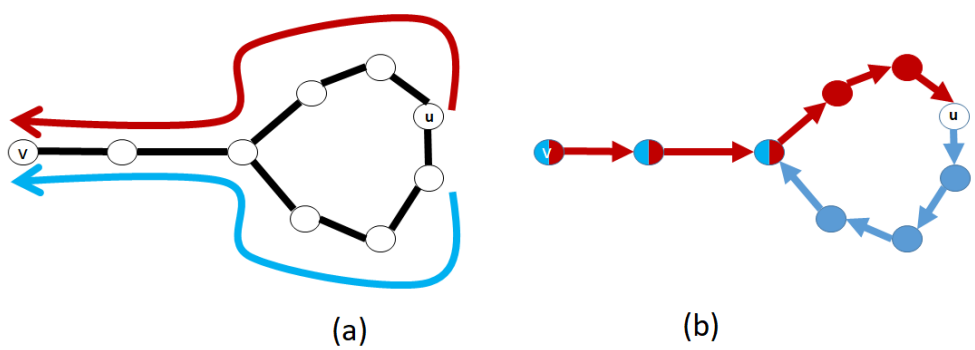

Figure 2.4:
(a) The path of the tokens from the chosen witness w to its dominating vertex v
(b)The orientation each edge receives in this process.If it seen only blue - towards parent,if it seen red - towards child

Algorithm 5 SinklessCongest(G)

- Using a priority BFS for $2\log n$ rounds from all vertices, each vertex checks if it is a dominating vertex, and if not it saves two pointers - $passiveRoot, passiveParent$ as the maximum id encountered and the edge in which it entered the earliest (breaking ties by ids of parents).
- Each dominating vertex starts a BFS for $\log(n)$ rounds. Note that as two dominating vertices have a distance of at least $2\log n$, the two BFS originating from them are disjoint
- All cycle witnesses of a vertex v from the BFS send their id through their parent vertex in the BFS up to their root by priority BFS for $\log(n)$ steps.
- If a dominating vertex v receives a cycle witness, it chooses the cycle witness with the largest id and sends it to its $\log n$ neighborhood
- The chosen witness of v chooses two incoming edges of the BFS, sends a red token through one, and a blue token through the second. These tokens traverse back through the parent edges until reaching back to v.
- Through each edge that the red token traversed, direct the parent edge towards the child. If the blue token passes to the parent edge, and the red token did not, direct it towards the parent. If a vertex encounters any token in this phase, it marks itself as active, otherwise it marks itself passive
- Each passive vertex directs its $passiveParent$ edge outwards
- Direct any undirected edge into the larger id

Lemma 39. *Each dominating vertex receives a cycle witness during the algorithm.*

Proof. As two dominating vertices have distance at least $2\log n$, the BFS is uninterrupted by other dominating vertices.By Lemma 24, each vertex has a cycle of length $\log n$ in its $\log n$ neighborhood. Therefore in a BFS of length $\log n$, at least one vertex from that cycle receives two messages. □

Lemma 40. *Edges oriented by active nodes of a dominating vertex v before the end of the algorithm form a directed path connected to a directed cycle, where each edge is between two active vertices.*

Proof. Let u be the chosen witness of v. All such edges are oriented by the red and blue token traversal. As the red and blue tokens originate together from u, and they move at the first step into two different vertices, and both reach v, there must be a vertex w in which they both "re-meet" (in the sense that they both go through w). Furthermore, as they both traverse on parents of the BFS, once they "re-meet", the path traversed from w to v is traversed from both. The union of the two (disjoint) paths traversed up to w is u by both tokens form a cycle, and the path from w to v forms a path. The way the algorithm orients them forms a directed path, connected to a directed cycle. Clearly all nodes with edges oriented this way in the algorithm have a token traversing through them, therefore all edges are between two active vertices. □

Lemma 41. *Edges orientations of the algorithm are well defined.*

Proof. Consider the edges oriented before the last step. By Lemma 40, all edges oriented by active nodes of a dominating vertex v form a path connected to a cycle and are well defined. A passive vertex only orients the edge $passiveParent$. If its $passiveParent$ is between two passive nodes, the edge orientation is well defined unless the two vertices have the same edge pointer of $passiveParent$. They have to be

different, as if the id of $passiveRoot$ is the same at both vertices, then the earliest entry time of the message to their $passiveParent$ edge differs by one, by definition of BFS, indicating that $passiveParent$ is not the same edge. In the case that $passiveRoot$ is different in these two vertices, which clearly indicates that $passiveParent$ is different. If the edge is between a passive and active node it is well defined as by Lemma 40 all edges oriented by active nodes are between two active nodes. Clearly all remaining unoriented edges in the last step are oriented in a consistent manner. □

Corollary 42. *The oriented edges of G form a sinkless orientation.*

Proof. As each passive vertex has an outgoing edge to its $passiveParent$, and each active vertex has an outgoing edge as it participates in a directed path that connects into a directed cycle, all vertices have an outgoing edge. □

Theorem 43. *SinklessCongest terminates after $O(\log n)$ rounds and the orientation obtained is a sinkless orientation.*

Proof. The round complexity is clearly $O(\log n)$, as the time complexity of each step is bounded by at most $2 \log n$ rounds, and by Corollary 42 the orientation obtained is a sinkless orientation. □

Bibliography

[AGM12] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Graph sketches: Sparsification, spanners, and subgraphs. In *Proceedings of the 31st Symposium on Principles of Database Systems*, PODS '12, pages 5–14, 2012.

[Amb96] A. Ambainis. Communication complexity in a 3-computer model. *Algorithmica*, 16(3):298–301, 1996.

[BFH+16] Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela, and Jara Uitto. A lower bound for the distributed lovász local lemma. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 479–488, 2016.

[BGKL04] László Babai, Anna Gál, Peter G. Kimmel, and Satyanarayana V. Lokam. Communication complexity of simultaneous messages. *SIAM J. Comput.*, 33(1):137–166, 2004.

[BK97] L. Babai and P. G. Kimmel. Randomized simultaneous messages: Solution of a problem of yao in communication complexity. In *Proceedings of the 12th Annual IEEE Conference on Computational Complexity*, CCC '97, pages 239–. IEEE Computer Society, 1997.

[BMN+11] Florent Becker, Martín Matamala, Nicolas Nisse, Ivan Rapaport, Karol Suchan, and Ioan Todinca. Adding a referee to an interconnection network: What can(not) be computed in one round. In *25th IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2011*, pages 508–514, 2011.

[BMRT14] Florent Becker, Pedro Montealegre, Ivan Rapaport, and Ioan Todinca. *Structural Information and Communication Complexity: 21st International Colloquium, SIROCCO 2014, Takayama, Japan, July 23-25, 2014. Proceedings*, chapter The Simultaneous Number-in-Hand Communication Model for Networks: Private Coins, Public Coins and Determinism, pages 83–95. 2014.

[BW] J. Bourgain and A. Wigderson. Personal communication (see [BK97]).

[CFL83] Ashok K. Chandra, Merrick L. Furst, and Richard J. Lipton. Multi-party protocols. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, STOC '83, pages 94–99, 1983.

[CKP16] Yi-Jun Chang, Tsvi Kopelowitz, and Seth Pettie. An exponential separation between randomized and deterministic complexity in the LOCAL model. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 615–624, 2016.

[CPS14] Kai-Min Chung, Seth Pettie, and Hsin-Hao Su. Distributed algorithms for the lovasz local lemma and graph coloring. In *PODC Proceedings of the 2014 ACM symposium on Principles of distributed computing*, pages 134–143, 2014.

[CRR14] Arkadev Chattopadhyay, Jaikumar Radhakrishnan, and Atri Rudra. Topology matters in communication. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 631–640, 2014.

[CSWY01] A. Chakrabarti, Yaoyun Shi, A. Wirth, and A. Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Proceedings of the Fourty-Second Annual Symposium on Foundations of Computer Science*, FOCS '01, pages 270–278, 2001.

[GS17] Mohsen Ghaffari and Hsin-Hao Su. Distributed degree splitting, edge coloring, and orientations. SODA 2017, 2017.

[JK09] Rahul Jain and Hartmut Klauck. New results in the simultaneous message passing model via information theoretic techniques. In *Proceedings of the Twenty-Fourth Annual IEEE Conference on Computational Complexity, 2009*, CCC '09, pages 369 – 378, 2009.

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[KNR95] Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. In *Proceedings of the Twenty-seventh Annual ACM Symposium on Theory of Computing*, STOC '95, pages 596–605, 1995.

[McG] Andrew McGregor. Open problem 65. List of Open Problems in Sublinear Algorithms. `http://sublinear.info/index.php?title=Open_Problems:65`.

[MS81] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error Correcting Codes*. North-Halland, 1981.

[MT10] Robin A. Moser and Gábor Tardos. A constructive proof of the general lovász local lemma. *J. ACM*, 57(2):11:1–11:15, 2010.

[NS96] Ilan Newman and Mario Szegedy. Public vs. private coin flips in one round communication games (extended abstract). In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 561–570, 1996.

[WW15] Omri Weinstein and David P. Woodruff. *Automata, Languages, and Programming: 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, chapter The Simultaneous Communication of Disjointness with Applications to Data Streams, pages 1082–1093. 2015.

[Yao79] Andrew Chi-Chih Yao. Some complexity questions related to distributive computing(preliminary report). In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 209–213, 1979.

תקציר

לסיבוכיות תקשורת תפקיד מרכזי בהוכחת חסמים תחתונים במודלים מבוזרים שבהם יש מגבלה על רוחב הפס של התקשורת בין השחקנים (CONGEST). הטכניקת הוכחה נפוצה היא רדוקציה לסיבוכיות תקשורת של שני שחקנים, אבל לפעמים שני שחקנים בלבד לא מספיקים בשביל החסם הרצוי. לפיכך, טבעי לחקור סיבוכיות תקשורת בין מספר רב של שחקנים. בעבודה הזאת נחקור סיבוכיות תקשורת בין מספר רב של שחקנים בדגש על סיבוכיות תקשורת סימולטנית.

עבור סיבוכיות תקשורת סימולטנית בין שני שחקנים זה ידוע שרנדומיות פומבית חזקה בהרבה מרנדומיות פרטית: פרוטוקולים עם גישה לרנדומיות פומבית יכולים להיות זולים באופן לא חסום על פני הפרוטוקולים הדטרמיניסטים היעילים ביותר לאותה בעיה, וידוע שרנדומיות פרטית שיכולה להיות רק טובה בצורה ריבועית מפרוטוקול דטרמיניסטי.(Babai & Kimmel 97)

בעבודה זאת אנחנו מרחיבים את התוצאה עבור מספר רב של שחקנים, ומראים שפונקציה f של k שחקנים עם סיבוכיות תקשורת דטרמיניסטית D(f) יש סיבוכיות תקשורת רנדומית פרטית של $\Omega(\sqrt{D(f)})$ לפחות לכל k גדול מ2. החסם הזה הדוק – בעזרת הפונקצית All-EQ (ששווה 1 אם יש שוויון בין כל השחקנים) ניתן להראות אלגוריתם שרץ בסיבוכיות $\tilde{O}(\sqrt{nk}+k)$. מכיוון שהסיבוכיות הדטרמיניסטית של All-Eq הוא $\Theta(nk)$ זה מראה שהחסם התחתון לא נחלש גם כשמספר השחקנים הוא גבוה.

בנושא מבוזר, מפורט החלק שלי בעבודה משותפת על בעיית Sinkless Orientation במודל הLOCAL, שכולל את הרדוקציה, ואלגוריתמים רנדומים ודטרמיניסטים שפותרים את הבעיה הזאת וכמה וריאציות על אחרות על הבעיה. עבודה ששיפרה באופן אקספוננציאלי את החסם התחתון שהיה ידוע על ה Lovasz Local Lemma. בנוסף אני מוסיף חלק נוסף שמסביר כיצד להשיג אלגוריתם דטרמיניסטי יעיל גם במודל הCONGEST.

אוניברסיטת תל-אביב TEL AVIV UNIVERSITY
הפקולטה למדעים מדויקים ע״ש ריימונד ובברלי סאקלר
בית הספר למדעי המחשב ע״ש בלבטניק

נושאים בסיבוכיות תקשורת מרובת משתתפים ומבוזר

חיבור זה הוגש כחלק מהדרישות לקבלת התואר

"מוסמך האוניברסיטה(M.Sc.)"

על ידי

מגיש אור פישר

עבודת המחקר בוצעה בהנחייתם של

ד"ר רותם אושמן ופרופ' אורי צוויק