Lower Bounds for Multiplication via Network Coding

Peyman Afshani[*] Casper Freksen[†] Lior Kamma[†] Kasper Green Larsen[‡]

Abstract

Multiplication is one of the most fundamental computational problems, yet its true complexity remains elusive. The best known upper bound, by Fürer, shows that two n-bit numbers can be multiplied via a boolean circuit of size $O(n \lg n \cdot 4^{\lg^* n})$, where $\lg^* n$ is the very slowly growing iterated logarithm. In this work, we prove that if a central conjecture in the area of network coding is true, then any constant degree boolean circuit for multiplication must have size $\Omega(n \lg n)$, thus almost completely settling the complexity of multiplication circuits. We additionally revisit classic conjectures in circuit complexity, due to Valiant, and show that the network coding conjecture also implies one of Valiant's conjectures.

1 Introduction

Multiplication is one of the most fundamental computational problems and the simple "long multiplication" $O(n^2)$-time algorithm for multiplying two n-digit numbers is taught to elementary school pupils around the world. Despite its centrality, the true complexity of multiplication remains elusive. In 1960, Kolmogorov conjectured that the thousands of years old $O(n^2)$-time algorithm is optimal and he arranged a seminar at Moscow State University with the goal of proving this conjecture. However only a week into the seminar, the student Karatsuba came up with an $O(n^{\lg_2 3}) \approx O(n^{1.585})$ time algorithm [KO62]. The algorithm was presented at the next seminar meeting and the seminar was terminated. This sparked a sequence of improved algorithm such as the Toom-Cook algorithm [Too63, Coo66] and the Schönhage-Strassen algorithm [SS71]. The Schönhage-Strassen algorithm, as well as the current fastest algorithm by Fürer [Fü09], are both based on the Fast Fourier Transform (FFT). Fürer's algorithm can be shown to run in time $O(n \lg n \cdot 4^{\lg^* n})$ when multiplying two n-bit numbers [HvdH18]. It can even be implemented as a constant degree Boolean circuit of the same size. Here $\lg^* n$ is the very slowly growing iterated logarithm.

But what is the true complexity of multiplying two n-bit numbers? Can it be done via e.g. a Boolean circuit of size $O(n)$ like addition? Or is multiplication strictly harder? Our main contribution is to show a connection between multiplication and a central conjecture by Li and Li [LL04] in the area of *network coding*. Our results show that if the conjecture by Li and Li [LL04] is true, then any constant degree Boolean circuit for computing the product of two n-bit numbers must have size $\Omega(n \lg n)$. This establishes a conditional lower bound

[*] Aarhus University. Email: peyman@cs.au.dk.

[†] Aarhus University. Supported by a Villum Young Investigator Grant. Email: {cfreksen, lior.kamma}@cs.au.dk.

[‡] Aarhus University. Supported by a Villum Young Investigator Grant and an AUFF Starting Grant. Email: larsen@cs.au.dk.

for multiplication that comes within a $4^{\lg^* n}$ factor of Fürer's upper bound and implies that multiplication is strictly harder than addition.

Before diving into the details of our results, we first give a brief introduction to network coding.

Network Coding. Network coding studies communication problems in graphs. Given a graph G with capacity constraints on the edges and k data streams, each with a designated source-sink pair of nodes (s_i, t_i) in G, what is the maximum rate at which data can be transmitted concurrently between the source-sink pairs? One solution is to just forward the data, which reduces the problem to a *multicommodity flow* problem. The central question in network coding is whether one can achieve a higher rate by using coding/bit tricks. This question is known to have a positive answer in directed graphs, where the rate increase may be as high as a factor $\Omega(|G|)$ (by sending XOR's of carefully chosen input bits), see e.g. [AHJ$^+$06]. However the question remains wide open for undirected graphs where there are no known examples for which network coding can do better than the multicommodity flow rate. A central conjecture in network coding, due to Li an Li [LL04], says that coding yields no advantage in undirected graphs.

Conjecture 1 (Undirected k-pairs Conjecture [LL04])**.** *The coding rate is equal to the Multicommodity-Flow rate in undirected graphs.*

Despite the centrality of this conjecture, it has heretofore resisted all attempts at either proving or refuting it. Conjecture 1 has been used twice before for proving lower bounds for computational problems. Adler *et al.* [AHJ$^+$06] were the first to initiate this line of study. They presented conditional lower bounds for computing the transpose of a matrix via an *oblivious algorithm*. Here oblivious means that the memory access pattern is fixed and independent of the input. Since a circuit is oblivious, they also obtain circuit lower bounds for matrix transpose. Very recently Farhadi *et al.* [FHLS19] showed how to remove the *obliviousness* assumption for external memory problems. Their main result was a tight lower bound for external memory integer sorting, conditioned on Conjecture 1 being true.

1.1 Our Results

Our main result is an exciting new connection between network coding and the complexity of multiplication. Formally, we prove the following theorem:

Theorem 1. *Assuming Conjecture 1, every boolean circuit with arbitrary gates and bounded in and out degrees that computes the product of two numbers given as two n-bit strings has size $\Omega(n \lg n)$.*

In fact, we prove our $\Omega(n \lg n)$ lower bound for an even simpler problem than multiplication, namely the *shift problem*: In the shift problem, we are given an n-bit string x and an index $j \in [n]$. The goal is to construct a circuit that outputs the $2n$-bit string y whose ith bit equals the $(i - j + 1)$th bit of x for every $j \leq i \leq j + n - 1$. Here we think of the index j as being given in binary using $\lceil \lg n \rceil$ bits. We prove the following result:

Theorem 2. *Assuming Conjecture 1, every boolean circuit with arbitrary gates and bounded in and out degrees that computes the shift problem has size $\Omega(n \lg n)$.*

Theorem 1 follows as a corollary of Theorem 2 by observing that shifting x by j positions is equivalent to multiplication by 2^j. Moreover, it is not hard to see that there is a linear sized circuit that has $\lceil \lg n \rceil$ input gates and n output gates, where on an index $j \in [n]$, it outputs the number 2^j in binary (i.e. a single 1-bit at position j).

We find it quite fascinating that even a simple instruction such as shifting requires circuits of size $\Omega(n \lg n)$, at least if we believe Conjecture 1.

Valiant's Depth Reduction and Circuit Complexity Lower Bounds. In addition to our main lower bound results for multiplication, we also demonstrate that the network coding conjecture sheds new light on some fundamental conjectures by Valiant. In a 1977 survey Valiant [Val77] outlined potentially plausible attacks on the problem of proving a lower bound for the size of any circuit that can compute a permutation or even shifts of a given input. The goal was to prove that achieving both $O(n)$ size and $O(\lg n)$ depth for such circuits is impossible. While most of his attacks were rebuffed due to existence of complex and highly connected graphs that only had $O(n)$ edges (superconcentrators), Valiant outlined one last potential approach that could still be fruitful. His main brilliant idea was to start with a circuit of some depth and by applying graph theoretical approaches reducing the depth of the circuit while eliminating only a small number of edges. The hope was that information theoretical approaches could finish the job once the depth of the circuit was very low and once the (graph theoretical) complexity of the circuit was peeled away.

More formally, Valiant showed that for every circuit C with n input and output gates, of size $O(n)$, depth $O(\lg n)$ and fan-in 2, and for every $\varepsilon > 0$, the function computed by C can be computed by a boolean circuit with arbitrary gates C' of depth 3 with n input and output gates and εn extra nodes. Moreover, the number of input gates directly connected to an output gate is bounded. That is, if we denote the set of input and output gates by X and Y respectively, then for every $y \in Y$, there are at most $O(n^\varepsilon)$ wires connecting y and X.

In turn, this reduction shows that it is enough to prove lower bounds on such depth 3 circuits. Almost 20 years later and based on these ideas, Valiant [Val92] put forward several conjectures that if resolved could open the way for proving circuit complexity lower bounds. Loosely speaking, Valiant conjectured that if $\varepsilon \leq 1/2$ then such depth 3 circuits cannot compute cyclic-shift permutation. Before discussing Valiant's conjectures more formally, we first state our second main result, which essentially shows that Conjecture 1 implies one of Valiant's conjectures, albeit with a smaller (but still constant) bound on ε.

Theorem 3. *Let C be a depth 3 circuit that computes multiplication such that the following holds.*

1. *The number of gates in the second layer of C is at most εn for $\varepsilon \leq 1/300$; and*
2. *for every output gate y of C, the number of input gates directly connected to y is at most c.*

Then assuming Conjecture 1, $c = \Omega\left(\frac{\lg n}{\lg \lg n}\right)$.

As with Theorem 1, we prove Theorem 3 on an even restricted set of circuits, namely circuits that compute the shift function. We now turn to give a formal description of Valiant's Conjectures, and demonstrate how Theorem 3 brings us closer to settling them.

Valiant's Conjectures. Let Γ be a bipartite graph on two independent sets X and Y such that $X = \{x_1, \dots, x_n\}$ denotes a set of inputs and $Y = \{y_1, \dots, y_n\}$ denotes a set of outputs. Furthermore assume, let $f_1, \dots f_{\varepsilon n}$ be εn extra nodes and connect them by edges to all the nodes in Γ. Denoting the resulting graph by G consider all possible boolean circuits with arbitrary gates whose underlying topology is G. We say such a circuit computes a permutation $\pi \colon Y \to X$ if for every assignment $x_1, \dots, x_n \in \{0,1\}^n$ to the input gates, after the evaluation of the circuit y_j is assigned $\pi(y_j)$ for every $j \in [n]$. Valiant conjectured that this should be impossible if ε is too small or if Γ has too few edges. In particular, he proposed the following.

Conjecture 2. *If Γ has maximum degree at most 3 and if $\varepsilon \leq 1/2$, then there exists a permutation π such that no circuit that has G as its underlying topology can compute the permutation π. Moreover, there exists such π that is a cyclic shift.*

Theorem 3 shows that conditioned on Conjecture 1, if $\varepsilon \leq 1/300$ then Valiant's first conjecture holds. We note that our proof for Theorem 3 continues to hold even if the gates' boolean functions are fixed after the shift offset is given. That is, if only the topology is fixed in advance. This coincides exactly with the formulation of Valiant's conjecture. Valiant further conjectured the following.

Conjecture 3. *If Γ has at most $n^{2-\delta}$ edges for some constant $\delta > 0$, and if $\varepsilon \leq 1/2$, then there exists a permutation π such that no circuit that has G as its underlying topology can compute the permutation π. Moreover, there exists such π that is a cyclic shift.*

1.2 Related Work

Lower Bounds for Multiplication. There are a number of previous lower bounds for multiplication in various restricted models of computation. Clifford and Jalsenius [CJ11] considered a streaming variant of multiplication, where one number is fixed and the other is revealed one digit at a time. They require that a digit of the output is reported before the next digit of the input is revealed. In this streaming setting, they prove an $\Omega((\delta/w)n \lg n)$ lower bound, where δ is the number of bits in a digit and w is the word size. For $\delta = 1$ and $w = O(1)$, this is $\Omega(n \lg n)$. Ponzio [Pon98] considered multiplication via read-once branching programs, i.e. programs that have bounded working memory and may only read each input bit exactly once. He proved that any read-once branching program for computing the middle bit of the product of two n-bit numbers, must use $\Omega(\sqrt{n})$ bits of working memory. Finally, we also mention the work of Morgenstern [Mor73] who proved lower bounds for computing the related FFT. Morgenstern proved an $\Omega(n \lg n)$ lower bound for computing the *unnormalied* FFT via an arithmetic circuit when all constants used in the circuit are bounded. Unfortunately this doesn't say anything about the complexity of multiplying two n-bit numbers.

Valiant's Conjectures. Despite their importance, Valiant's conjectures are still mostly open. One interesting development by Riis [Rii07], shows that Conjecture 3 as stated is incorrect. Riis proved that all cyclic shifts are realizable for $\varepsilon = \frac{1}{2} - \frac{1}{2n^{1-\delta}}$ where $n^{1+\delta}$ is the total number of edges of Γ. Riis further conjectured that replacing the bound on ε by a slightly stricter bound should result in a correct conjecture. Specifically, Riis suggest bounding $\varepsilon = \Theta\left(\frac{1}{\lg \lg n}\right)$.

2 Preliminaries

We now give a formal definition of Boolean circuits with arbitrary gates, followed by definitions of the k-pairs communication problem, the multicommodity flow problem. In the two latter problems we reuse some of the definitions used by Farhadi *et al.* [FHLS19], which have been simplified a bit compared to the more general definition by Adler *et al.* [AHJ$^+$06]. In particular, we have forced communication networks to be directed acyclic graphs. This is sufficient to prove our lower bounds and simplifies the definitions considerably.

Boolean Circuits with Arbitrary Gates. A *Boolean Circuit with Arbitrary Gates* with n source or input nodes and m target or output nodes is a directed acyclic graph C with n nodes of in-degree 0, which are called *input gates*, and are labeled with input variables $X = \{x_i\}_{i\in[n]}$ and m nodes out-degree 0, which are called *output gates* and are labeled with output variables $Y = \{y_i\}_{i\in[m]}$. All other nodes are simply called *gates*. For every gate u of in-degree $k \geq 1$, u is labeled with an arbitrary function $f_u : \{0,1\}^k \to \{0,1\}$. The circuit is also equipped with a topological ordering $v_1, \ldots, v_t$ of C, in which $v_i = x_i$ for $i \in [n]$ and $v_{t-i+1} = y_{m-i+1}$ for all $i \in [m]$. The *depth* of a circuit C is the length of the longest path in C. An *evaluation* of a circuit on an n bit input $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ is conducted as follows. For every $i \in [n]$, assign x_j to v_j. For every $j \geq n+1$, assign to v_j the value $f_{v_j}(u_1, \ldots, u_k)$, where $u_1, \ldots, u_k$ are the nodes of C with edges going into v_j in the order induced by the topological ordering. The *output* of C on an n bit input $x = (x_1, \ldots, x_n)$, denoted $C(x_1, \ldots, x_n)$ is the value assigned to $(y_1, \ldots, y_m)$ in the evaluation. We say a circuit computes a function $f : \{0,1\}^n \to \{0,1\}^m$ if for every $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, $f(x_1, \ldots, x_n) = C(x_1, \ldots, x_n)$.

For every $j \in [t]$ and $b \in \{0,1\}$, we *hardwire* b for v_j in C by removing v_j and all adjacent edges from C, and replacing v_j for b in the evaluation of f_{v_i} for every $i > j$ such that $v_j v_i$ is an edge in C.

k-Pairs Communication Problem. The input to the k-pairs communication problem is a directed acyclic graph $G = (V, E)$ where each edge $e \in E$ has a capacity $c(e) \in \mathbb{R}^+$. There are k sources $s_1, \ldots, s_k \in V$ and k sinks $t_1, \ldots, t_k \in V$.

Each source s_i receives a message A_i from a predefined set of messages $A(i)$. It will be convenient to think of this message as arriving on an in-edge. Hence we add an extra node S_i for each source, which has a single out-edge to s_i. The edge has infinite capacity.

A network coding solution specifies for each edge $e \in E$ an alphabet $\Gamma(e)$ representing the set of possible messages that can be sent along the edge. For a node $v \in V$, define $\mathrm{In}(u)$ as the set of in-edges at u. A network coding solution also specifies, for each edge $e = (u, v) \in E$, a function $f_e : \prod_{e' \in \mathrm{In}(u)} \Gamma(e') \to \Gamma(e)$ which determines the message to be sent along the edge e as a function of all incoming messages at node u. Finally, a network coding solution specifies for each sink t_i a decoding function $\sigma_i : \prod_{e \in \mathrm{In}(t_i)} \Gamma(e) \to M(i)$. The network coding solution is correct if, for all inputs $A_1, \ldots, A_k \in \prod_i A(i)$, it holds that σ_i applied to the incoming messages at t_i equals A_i, i.e. each source must receive the intended message.

In an execution of a network coding solution, each of the extra nodes S_i starts by transmitting the message A_i to s_i along the edge (S_i, s_i). Then, whenever a node u has received a message a_e along all incoming edges $e = (v, u)$, it evaluates $f_{e'}(\prod_{e \in \mathrm{In}(u)} a_e)$ on all out-edges and forwards the message along the edge e'.

We define the *rate* of a network coding solution as follows: Let each source receive a uniform random and independently chosen message A_i from $A(i)$. For each edge e, let A_e denote the random variable giving the message sent on the edge e when executing the network coding solution with the given inputs. The network coding solution achieves rate r if:

- $H(A_i) \geq r$ for all i.
- For each edge $e \in E$, we have $H(A_e) \leq c(e)$.

Here $H(\cdot)$ denotes binary Shannon entropy. The intuition is that the rate is r, if the solution can handle sending a message of entropy r bits between every source-sink pair.

Multicommodity Flow. A multicommodity flow problem in an undirected graph $G = (V, E)$ is specified by a set of k source-sink pairs (s_i, t_i) of nodes in G. We say that s_i is the source of commodity i and t_i is the sink of commodity i. Each edge $e \in E$ has an associated capacity $c(e) \in \mathbb{R}^+$. A (fractional) solution to the multicommodity flow problem specifies for each pair of nodes (u, v) and commodity i, a flow $f^i(u, v) \in [0, 1]$. Intuitively $f^i(u, v)$ specifies how much of commodity i that is to be sent from u to v. The flow satisfies *flow conservation*, meaning that:

- For all nodes u that is not a source or sink, we have $\sum_{w \in V} f^i(u, w) - \sum_{w \in V} f^i(w, u) = 0$.
- For all sources s_i, we have $\sum_{w \in V} f^i(s_i, w) - \sum_{w \in V} f^i(w, s_i) = 1$.
- For all sinks we have $\sum_{w \in V} f^i(w, t_i) - \sum_{w \in V} f^i(t_i, w) = 1$.

The flow also satisfies that for any pair of nodes (u, v) and commodity i, there is only flow in one direction, i.e. either $f^i(u, v) = 0$ or $f^i(v, u) = 0$. Furthermore, if (u, v) is not an edge in E, then $f^i(u, v) = f^i(v, u) = 0$. A solution to the multicommodity flow problem achieves a rate of r if:

- For all edges $e = (u, v) \in E$, we have $r \cdot \sum_i (f^i(u, v) + f^i(v, u)) \leq c(e)$.

Intuitively, the rate is r if we can handle a demand of r for every commodity.

The Undirected k-Pairs Conjecture. Conjecture 1 implies the following for our setting: Given an input to the k-pairs communication problem, specified by a directed acyclic graph G with edge capacities and a set of k source-sink pairs, let r be the best achievable network coding rate for G. Similarly, let G' denote the undirected graph resulting from making each directed edge in G undirected (and keeping the capacities and source-sink pairs). Let r' be the best achievable flow rate in G'. Conjecture 1 implies that $r \leq r'$.

Having defined coding rate and flow rate formally, we also mention that a result of Braverman *et al.* [BGS17] implies that if there exists a graph G where the network coding rate r, and the flow rate r' in the corresponding undirected graph G', satisfies $r \geq (1 + \varepsilon)r'$ for a constant $\varepsilon > 0$, then there exists an infinite family of graphs $\{G^*\}$ for which the corresponding gap is at least $(\lg |G^*|)^c$ for a constant $c > 0$. So far, all evidence suggest that no such gap exists, as formalized in Conjecture 1.

3 Key Tools and Techniques

The main idea in the heart of both proofs is the simple fact that in a graph with t vertices and maximum degree at most c, most node pairs lie far away from one another. Specifically, for every node u in G, at least $t - \sqrt{t}$ nodes have distance $\geq \frac{1}{2}\log_c t$ from u. While this key observation is almost enough to prove Theorem 2, the proof of Theorem 3 requires a much more subtle approach, as there is no bound on the maximum degree in the circuits in question. The only bound we have is on the number of wires going directly between from input gates into output gates. Specifically, every two nodes in the underlying undirected graph are at distance ≤ 3 (see figure 1).

In order to overcome this obstacle, we present a construction of a communication network based on the circuit C that essentially eliminates the middle layer in the depth-3 circuit C, thus leaving a bipartite graph with bounded maximum degree. To this end, we observe that since the size of the middle layer is bounded by εn, then there exists a large set $\mathcal{F}$ of inputs in $\{0,1\}^n$ such that on all inputs from $\mathcal{F}$, the gates $f_1, \ldots, f_{\varepsilon n}$ attain the same values. By hardwiring these values to the circuit, we can evaluate the circuit for all inputs in $\mathcal{F}$ on a depth-2 circuit Γ obtained from C by removing $f_1, \ldots, f_{\varepsilon n}$. We next turn to construct the communication network. Employing ideas recently presented by Farhadi *et al.* [FHLS19], we "wrap" the depth-2 circuit by adding source and target nodes. In order to cope with inputs that do not belong to $\mathcal{F}$, we add a designated *supervisor* node u (see figure 2). Loosely speaking, the source nodes transmit their input to u, and u sends back the information needed to "edit" the input string x and construct an input string $x' \in \mathcal{F}$, which is then transferred to the circuit Γ as blackbox.

The Correction Game. In order to bound the edge capacities of the network G in a way that the supervisor node can transmit enough information to achieve a high communication rate, but then again not allow to much flow to go through the supervisor when considering G as a multicommodity flow instance, Farhadi *et al.* [FHLS19] defined a game between a set of m players and a supervisor, where given a fixed set $\mathcal{F} \subseteq \{0,1\}^n$ and a random string $\beta \in \{0,1\}^n$ given as a concatenation of m strings $\beta_1, \ldots, \beta_m$ of length n/m each, the goal is to "correct" x and produce a string $\chi \in \{0,1\}^n$ such that $\beta \oplus \chi \in \mathcal{F}$. The caveat is that the only communication allowed is between the players and the supervisor. That is, no communication, and thus no cooperation, is allowed between the m players. Formally, the game is defined as follows.

Definition 1. *Let $\mathcal{F} \subseteq \{0,1\}^n$. The* $\mathcal{F}$-correction game *with $m+1$ players is defined as follows. The game is played by m ordinary players $p_1, \ldots, p_m$ and one designated* supervisor *player u. The supervisor u receives m strings $\beta_1, \ldots, \beta_m \in \{0,1\}^{n/m}$ chosen independently at random. For every $\ell \in [m]$, u then sends p_ℓ a message R_ℓ. Given R_ℓ, the player p_ℓ produces a string $\chi_\ell \in \{0,1\}^{n/m}$ such that $(\beta_1 \oplus \chi_1) \circ (\beta_2 \oplus \chi_2) \circ (\beta_m \oplus \chi_m) \in \mathcal{F}$.*

Farhadi *et al.* additionally present a protocol for the $\mathcal{F}$-correction game in which the supervisor player sends prefix-free messages to the m players, and moreover, they give a bound on the amount of communication needed as a function of the number of players and the size of $\mathcal{F}$.

Lemma 4 ([FHLS19]). *If $|\mathcal{F}| \geq 2^{(1-\varepsilon)n}$, then there exists a protocol for the $\mathcal{F}$-correction game*

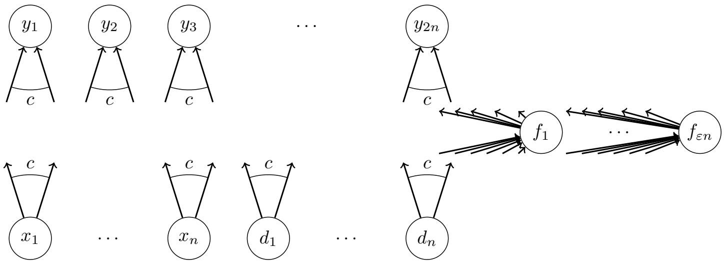

Figure 1: The depth 3 circuit C.

with $m+1$ *players such that the messages* $\{R_\ell\}_{\ell\in[m]}$ *are prefix-free and*

$$\sum_{\ell\in[m]} \mathbb{E}[|R_\ell|] \le 3m + 2m \lg\left(\sqrt{\frac{\varepsilon}{2}} \cdot \frac{n}{m} + 1\right) + \sqrt{\frac{\varepsilon}{8}} \cdot n \lg \frac{2}{\varepsilon} \ ,$$

4 A Lower Bound for Boolean Circuits Computing Multiplication

In this section we show that conditioned on Conjecture 1, every bounded degree circuit computing multiplication must have size at least $\Omega(n \lg n)$, thus proving Theorems 1 and 2. In fact, we will prove something slightly stronger. Define the shift function $s : \{0,1\}^n \times [n] \to \{0,1\}^{2n}$ as follows. For every $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $\ell \in [n]$, $s(x, \ell) = (y_1, \ldots, y_{2n})$ where $y_j = x_{j-\ell+1}$ if $\ell \le j \le \ell + n - 1$ and $y_j = 0$ otherwise. We will show that every circuit with bounded in and out degrees that computes the shift function on n-bit numbers has size $\Omega(n \lg n)$. Clearly, a circuit that can compute the product of two n-bit numbers can also compute the shift function. Let c denote the maximum in and out degree in C, and let $j \in [n]$. Then in the undirected graph induced by C, there are at most $\sqrt{n}$ nodes whose distance from x_j is at most $\frac{1}{2}\log_{2c} n$. Therefore among $y_j, \ldots, y_{j+n-1}$, at least $n - \sqrt{n} - 1 \ge n - 2\sqrt{n}$ are at distance at least $\frac{1}{2}\log_{2c} n$. In other words, $\Pr_{\ell\in[n]}[d_{\hat{C}}(x_j, y_{j+\ell-1}) \ge \frac{1}{2}\log_{2c} n] \ge 1 - \frac{2}{\sqrt{n}}$, where $\hat{C}$ denotes the undirected graph induced by C (by removing edge directions). Therefore there exists a shift $\ell_0 \in [n]$ such that $|\{j \in [n] : d_{\bar{C}}(x_j, y_{j+\ell_0-1}) \ge \frac{1}{2}\log_{2c} n\}| \ge n - 2\sqrt{n} \ge n/2$.

Fixing ℓ_0, let consider the following communication problem. For each $j \in [n]$, $s_j = x_j \in_R \{0,1\}$ and $t_j = y_{j+\ell_0-1}$. The circuit C equipped with 1-uniform edge capacities is a network coding solution to this problem with rate $r \ge 1$. By the undirected n-pairs conjecture, there is a multicommodity flow in $\hat{C}$ that transfers one unit of flow from each source to its corresponding sink. For every j, let $f^j : E \to [0,1]$ be the flow associated with commodity j. Then

$$|E| = \sum_{e\in E} c_e \ge \sum_{e\in E}\sum_{j\in[n]} f^j(e) \ge \Omega(n \log_c n) \ .$$

5 A Lower Bound for Depth 3 Boolean Circuits Computing Multiplication

Let C be a depth 3 circuit that computes multiplication such that the number of gates in the second layer of C is at most εn for some small $\varepsilon \in (0,1)$ and for every $u \in Y$, $deg_{\bar{C}[X\cup Y]}(u) \le c$,

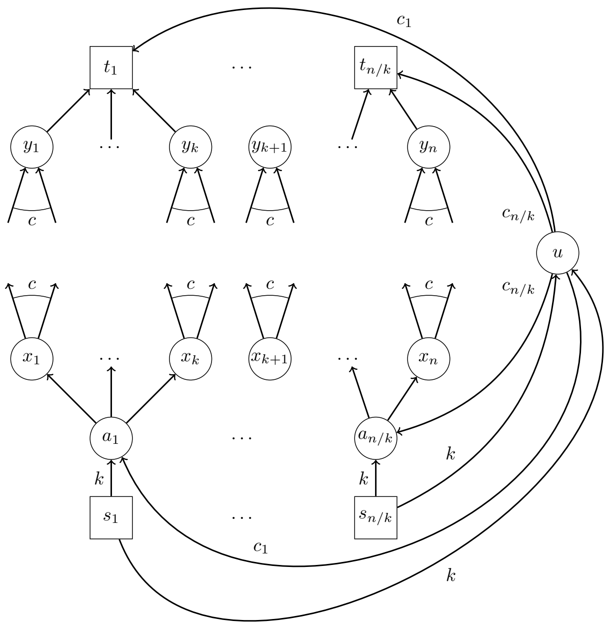

Figure 2: Given the 2-layer circuit Γ spanned by $x_1, \dots, x_n, y_1, \dots, y_n$, we construct the communication network graph G.

where once again $\bar{C}$ denotes the undirected graph induced by C, and $\bar{C}[X \cup Y]$ is the subgraph of $\bar{C}$ induced by $X \cup Y$. By slightly increasing c and ε (by a small constant factor) and without loss of generality, we can assume that this applies for all $u \in X$ as well.

Denote the input and output gates of C by $X = \{x_1, \dots, x_n, \hat{x}_1, \dots, \hat{x}_n\}$ and $Y = \{y_1, \dots, y_{2n}\}$ respectively, and denote the set of the middle-layer gates by $F = \{f_1, \dots, f_{\varepsilon n}\}$ (see Figure 1).

As before, we focus on computing the shift function, thus limiting the input to $(\hat{x}_1, \dots, \hat{x}_n)$ to have exactly one 1-entry. We next partition $(x_1, \dots, x_n)$ into consecutive blocks of size $k = 20$ bits each. For every $\ell \in [n/k]$ let $B_\ell = \{k(\ell-1)+1, \dots, k\ell\}$ be the set of indices belonging to the ℓth block.

Definition 2. *For every $\alpha \in [n]$ and $\ell \in [n/k]$, we say B_ℓ is* far from all targets *(with respect to α) if for all sources in the block are at distance at least $\frac{1}{2}\log_{2c} n$ from all respective destinations in $\bar{C}[X \cup Y]$. That is for every $u, v \in B_\ell$, $d_{\bar{C}[X \cup Y]}(x_u, y_{v+\alpha-1}) \geq \frac{1}{2}\log_{2c} n$.*

Let $\alpha \in_R [n]$. By the constraint on the degrees, for every $j \in [n]$, there are at most $\sqrt{n}$ nodes whose distance from x_j is at most $\frac{1}{2}\log_{2c} n$ in $\bar{C}[X \cup Y]$. Therefore for every $\ell \in [n/k]$,

$$\Pr_{\alpha \in_R [n]} [B_\ell \text{ is far from all targets}] \geq 1 - \frac{k^2}{\sqrt{n}} .$$

By averaging we get that for large enough n there is some $\alpha_0 \in [n]$ such that there are at least $\frac{n}{k} - k\sqrt{n} \geq \frac{9n}{10k}$ blocks which are far from all targets. Without loss of generality, we may assume for ease of notation that $\alpha_0 = 1$. By hardwiring 1 for α_0 into the circuit C, the circuit now simply transfers $(x_1, \ldots, x_n)$ to $(y_1, \ldots, y_n)$.

Reduction to Network Coding. Let $x = (x_1, \ldots, x_n)$ and $i \in [\varepsilon n]$. By slightly abusing notation, we denote the value of the gate f_i when evaluating the circuit by $f_i(x_1, \ldots, x_n)$. By averaging, there exist a string $(\hat{f}_1, \ldots, \hat{f}_{\varepsilon n})$ and a set $\mathcal{F} \subseteq \{0,1\}^n$ such that $|\mathcal{F}| \geq 2^{(1-\varepsilon)n}$ and such that for every $x = (x_1, \ldots, x_n) \in \mathcal{F}$ and $i \in [\varepsilon n]$, $f_i(x_1, \ldots, x_n) = \hat{f}_i$. By hardwiring $(\hat{f}_1, \ldots, \hat{f}_{\varepsilon n})$ for $(f_1, \ldots, f_n)$ into the circuit C, we get a new circuit denoted Γ that contains only the input and output gates of C, and transfers $(x_1, \ldots, x_n)$ to $(y_1, \ldots, y_n)$ for every $(x_1, \ldots, x_n) \in \mathcal{F}$. Moreover, the set of edges between X and Y in Γ is equal to the set of edges between X and Y in C.

Next, we construct a communication network G by adding some nodes and edges to Γ, as demonstrated also in Figure 2. We add a new set of nodes $\{s_j, a_j, t_j\}_{j=1}^{n/k} \cup \{u\}$. For every $\ell \in [n/k]$, add edges $s_\ell a_\ell$ and $s_\ell u$ of capacity k and edges $u a_\ell$ and $u t_\ell$ of capacity $c_\ell = \mathbb{E}[|R_\ell|]$, where R_ℓ is the message sent to player p_ℓ by the supervisor player in the $\mathcal{F}$-correction game protocol for $n/k + 1$ players guaranteed in Lemma 4. In addition, for every $\ell \in [n/k]$ and every $j \in B_\ell$ add edges $a_\ell x_j$ and $y_j t_\ell$ of capacity 1. All edges of Γ are assigned capacity of 1.

Transmitting Data. In what follows, we will lower bound the communication rate of the newly constructed network G.

Lemma 5. *There exists a network coding solution on G that achieves rate k.*

To this end, let $A_1, \ldots, A_{n/k} \in \{0,1\}^k$ be independent uniform random variables. We next give a protocol by which the sources $s_1, \ldots, s_{n/k}$ transmit $A_1, \ldots, A_{n/k}$ to the targets $t_1, \ldots, t_{n/k}$. The protocol employs as a an intermediate step the correction game protocol guaranteed by Lemma 4.

1. For every $\ell \in [n/k]$, s_ℓ sends A_ℓ to a_ℓ over the edge $s_\ell a_\ell$ and to u over the edge $s_\ell u$.

2. Employing the $\mathcal{F}$-correction game protocol with $n/k + 1$ players, for every $\ell \in [n/k]$, u sends a message R_ℓ to a_ℓ over the edge $u a_\ell$ and to t_ℓ over the edge $u t_\ell$. Following the correction game protocol, for every ℓ, given R_ℓ, a_ℓ and t_ℓ produce a string χ_ℓ satisfying that $(A_1 \oplus \chi_1) \circ \ldots \circ (A_{n/k} \oplus \chi_{n/k}) \in \mathcal{F}$.

3. For every $\ell \in [n/k]$ and every $i \in [k]$, a_ℓ transmits the ith bit of $A_\ell \oplus \chi_\ell$ to the ith gate in the ℓth block, namely $x_{(\ell-1)k+i}$. Note that $(x_1, \ldots, x_n) = (A_1 \oplus \chi_1) \circ \ldots \circ (A_{n/k} \oplus \chi_{n/k}) \in \mathcal{F}$.

4. Next, the communication network employs the circuit Γ and transmits $(x_1, \ldots, x_n)$ to $(y_1, \ldots, y_n)$. For every $\ell \in [n/k]$ and every $i \in B_\ell$, y_i transmits x_i to t_ℓ.

5. Finally, for every $\ell \in [n/k]$, t_ℓ now holds both $A_\ell \oplus \chi_\ell$ and χ_ℓ. Therefore t_ℓ can recover A_ℓ.

By invoking the protocol described above, every one of the n/k sources sends k bits to the corresponding target. For every edge $e \in G$, let A_e denote the random variable giving the message sent on the edge e when executing the protocol.

Claim 6. *For every $e \in G$, $H(A_e) \le c_e$.*

Proof. First note that for every $\ell \in [n/k]$, every edge e leaving s_ℓ has capacity k and transmits A_ℓ. Therefore $H(A_\ell) = k \le c_e$. Every edge e that is not leaving any source nor u has capacity 1 and transmits exactly one bit (not necessarily uniformly random) of information. Therefore $c_e = 1 \ge H(A_e)$. Finally, let e be an edge leaving u. Then there exists some $\ell \in [n/k]$ such that $e = ua_\ell$ or $e = ut_\ell$. In both cases the message transmitted on e is R_ℓ and the capacity c_e of e satisfies $c_e = c_\ell = \mathbb{E}[|R_\ell|] \ge H(R_\ell)$, where the last inequality follows from Shannon's Source Coding theorem, as all messages are prefix-free. □

We can therefore conclude that the network G achieves rate $\ge k$, and the proof of Lemma 5 is complete.

Deriving the Lower Bound. By Conjecture 1, the underlying undirected graph $\bar{G}$ achieves a multicommodity-flow rate $\ge k$. Therefore there exists a multicommodity flow $\{f^\ell\}_{\ell \in [n/k]} \subseteq [0,1]^{E(\bar{G})}$ that achieves rate k. We first observe that at most a constant fraction of the flow can go through the supervisor node u. To see this, we note that as $|\mathcal{F}| \ge 2^{(1-\varepsilon)n}$, then by Lemma 4 the expected total information sent by the supervisor in the $\mathcal{F}$-correction game with n/k players is at most

$$\frac{3n}{k} + \frac{2n}{k} \lg\left(k\sqrt{\frac{\varepsilon}{2}} + 1\right) + \sqrt{\frac{\varepsilon}{8}} \cdot n \lg \frac{2}{\varepsilon} \le \frac{5n}{k} \tag{1}$$

Therefore by the definition of the capacities $\{c_\ell\}_{\ell \in [n/k]}$ we get that for small enough (constant) ε,

$$\sum_{\ell \in [n/k]} c_{ua_\ell} = \sum_{\ell \in [n/k]} c_{ut_\ell} = \sum_{\ell \in [n/k]} c_\ell \le \frac{5n}{k} \tag{2}$$

Since $\{f^\ell\}_{\ell \in [n/k]}$ achieves rate k we conclude that

$$\begin{aligned} k \cdot \sum_{v \in V(\bar{G}) : uv \in E(\bar{G})} \sum_{\ell \in [n/k]} (f^\ell(u,v) + f^\ell(v,u)) &\le \sum_{v \in V(\bar{G}) : uv \in E(\bar{G})} c_e \\ &= \sum_{\ell \in [n/k]} c_{us_\ell} + \sum_{\ell \in [n/k]} (c_{ua_\ell} + c_{ut_\ell}) \le n + \frac{10n}{k}\ , \end{aligned}$$

and therefore

$$\sum_{v \in V(\bar{G}) : uv \in E(\bar{G})} \sum_{\ell \in [n/k]} (f^\ell(u,v) + f^\ell(v,u)) \le \frac{n}{k} + \frac{10n}{k^2} \le 1.5\frac{n}{k}\ . \tag{3}$$

By the flow-conservation constraint, we know that therefore the total amount of flow that can go through u is $\le 0.75\frac{n}{k}$. By averaging, at least a 1/6 fraction of the sources send at least 1/10 units of flow through $\bar{G} - u$. By the choice of α_0, in $\bar{G} - u$, at least a 1/15 of the sources are at least $\frac{1}{2}\log_{2c}(n)$ away from their targets. Without loss of generality, assume these are the first $\frac{n}{15k}$ sources. We conclude that

$$\begin{aligned} cn \ge |E[X \cup Y]| = \sum_{e \in E[X \cup Y]} c_e &\ge k \cdot \sum_{e = vw \in E[X \cup Y]} \sum_{\ell \in [n/k]} f^\ell(v,w) + f^\ell(w,v) \\ &\ge k \cdot \sum_{\ell \in [n/15k]} \sum_{e = vw \in E[X \cup Y]} f^\ell(v,w) + f^\ell(w,v) \ge \frac{n}{30}\log_{2c}(n)\ , \end{aligned} \tag{4}$$

and therefore $c \geq \Omega\left(\frac{\lg n}{\lg \lg n}\right)$, and the proof of Theorem 3 is now complete.

5.1 Remarks and Extensions

For sake of fluency, some minor remarks and extensions were intentionally left out of the text, and will be discussed now.

Circuits with Bounded Average Degree. Our results still hold if we relax the second requirement of Theorem 3 and require instead that the number of edges in $\bar{C}[X \cup Y]$ is at most cn. That is, the average degree in $\bar{C}[X \cup Y]$ is at most c. To see this, note that under this assumption, there are at most $0.001n$ gates in $X \cup Y$ whose degree in $\bar{C}[X \cup Y]$ is larger than $1000c$. For each such gate v, add a new node f in the middle layer, and connect v and all the neighbours of v in $\bar{C}[X \cup Y]$ to f. Then delete all the edges adjacent to v in $\bar{C}[X \cup Y]$. The number of nodes added to the middle layer is at most $0.001n$, and the degree of all nodes in $\bar{C}[X \cup Y]$ is now bounded by $1000c$. The rest of our proof continues as before.

Shifts vs. Cyclic Shifts. In order to prove lower bounds for circuits computing multiplication, our results are stated in terms of shifts (which are a special case of products, as mentioned). This is in contrast to Valiant's conjectures, which are stated in terms of cyclic shifts. However, we draw the readers attention to the fact that our proofs work for cyclic shifts as well. The exact same arguments apply, and the proofs remain unchanged.

References

[AHJ+06] M. Adler, N. J. A. Harvey, K. Jain, R. Kleinberg, and A. R. Lehman. On the capacity of information networks. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*, SODA '06, pages 241–250. Society for Industrial and Applied Mathematics, 2006. Available from: http://dl.acm.org/citation.cfm?id=1109557.1109585.

[BGS17] M. Braverman, S. Garg, and A. Schvartzman. Coding in undirected graphs is either very helpful or not helpful at all. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 18:1–18:18, 2017.

[CJ11] R. Clifford and M. Jalsenius. Lower bounds for online integer multiplication and convolution in the cell-probe model. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part I*, pages 593–604, 2011.

[Coo66] S. A. Cook. *On the minimum computation time of functions.* PhD thesis, Harvard University, 1966.

[Fü09] M. Fürer. Faster integer multiplication. *SIAM Journal on Computing*, 39(3):979–1005, 2009. doi:10.1137/070711761.

[FHLS19] A. Farhadi, M. Hajiaghayi, K. G. Larsen, and E. Shi. Lower bounds for external memory integer sorting via network coding. In *Proceedings of the 52st Symposium on Theory of Computing, STOC 2019*, 2019. To appear.

[HvdH18] D. Harvey and J. van der Hoeven. Faster integer multiplication using short lattice vectors. *CoRR*, 2018. `arXiv:1802.07932`.

[KO62] A. A. Karatsuba and Y. P. Ofman. Multiplication of many-digital numbers by automatic computers. *Proceedings of the USSR Academy of Sciences*, 145:293–294, 1962.

[LL04] Z. Li and B. Li. Network coding: The case of multiple unicast sessions. In *Proceedings of the 42nd Annual Allerton Conference on Communication, Control, and Computing*, 2004.

[Mor73] J. Morgenstern. Note on a lower bound on the linear complexity of the fast Fourier transform. *Journal of the ACM*, 20(2):305–306, 1973. `doi:10.1145/321752.321761`.

[Pon98] S. Ponzio. A lower bound for integer multiplication with read-once branching programs. *SIAM J. Comput.*, 28(3):798–815, 1998.

[Rii07] S. Riis. Information flows, graphs and their guessing numbers. *The Electronic Journal of Combinatorics*, 14(1), 2007.

[SS71] A. Schönhage and V. Strassen. Schnelle multiplikation großer zahlen. *Computing*, 7(3):281–292, Sep 1971. `doi:10.1007/BF02242355`.

[Too63] A. L. Toom. The complexity of a scheme of functional elements realizing the multiplication of integers. *Proceedings of the USSR Academy of Sciences*, 150(3):496–498, 1963.

[Val77] L. G. Valiant. Graph-theoretic arguments in low-level complexity. In *Mathematical Foundations of Computer Science 1977*, pages 162–176, 1977.

[Val92] L. G. Valiant. Why is boolean complexity theory difficult? In *Proceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, pages 84–94, 1992.