Quantum and Classical Strong Direct Product Theorems and Optimal Time-Space Tradeoffs

Hartmut Klauck[*]
University of Calgary
klauckh@cpsc.ucalgary.ca

Robert Špalek[†]
CWI, Amsterdam
sr@cwi.nl

Ronald de Wolf[†]
CWI, Amsterdam
rdewolf@cwi.nl

Abstract

A strong direct product theorem says that if we want to compute k independent instances of a function, using less than k times the resources needed for one instance, then our overall success probability will be exponentially small in k. We establish such theorems for the classical as well as quantum query complexity of the OR *function. This implies slightly weaker direct product results for all total functions. We prove a similar result for quantum communication protocols computing k instances of the Disjointness function. These results imply a time-space tradeoff* $T^2S = \Omega(N^3)$ *for sorting N items on a quantum computer, which is optimal up to polylog factors. They also give several tight time-space and communication-space tradeoffs for the problems of Boolean matrix-vector multiplication and matrix multiplication.*

1. Introduction

1.1. Direct product theorems

For every reasonable model of computation one can ask the following fundamental question:

> How do the resources needed for computing k independent instances of f scale with the resources needed for one instance and with k?

Here "resource" needs to be specified. It could refer to time, space, queries, communication etc. Similarly we need to define what we mean by "computing f", for instance whether we allow the algorithm some probability of error, and whether this probability of error is average-case or worst-case.

In this paper we consider two kinds of resources, queries and communication, and allow our algorithms some error probability. An algorithm is given k inputs $x^1, \ldots, x^k$, and has to output the vector of k answers $f(x^1), \ldots, f(x^k)$. The issue is how the algorithm can optimally distribute its resources among the k instances it needs to compute. We focus on the relation between the total amount T of resources available and the best-achievable success probability σ (which could be average or worst-case). Intuitively, if every algorithm with t resources must have some constant error probability when computing *one* instance of f, then for computing k instances we expect a constant error on each instance and hence an exponentially small success probability for the k-vector as a whole. Such a statement is known as a *weak* direct product theorem:

$$\text{If } T \approx t, \text{ then } \sigma = 2^{-\Omega(k)}$$

However, even if we give our algorithm roughly kt resources, on average it still has only t resources available per instance. So even here we expect a constant error per instance and an exponentially small success probability overall. Such a statement is known as a *strong* direct product theorem:

$$\text{If } T \approx kt, \text{ then } \sigma = 2^{-\Omega(k)}$$

Strong direct product theorems, though intuitively very plausible, are generally hard to prove and sometimes not even true. Shaltiel [41] exhibits a general class of examples where strong direct product theorems fail. This applies for instance to query complexity, communication complexity, and circuit complexity. In his examples, success probability is taken under the uniform probability distribution on inputs. The function is chosen such that for most inputs, most of the k instances can be computed quickly and without any error probability. This leaves enough resources to solve the few hard instances with high success probability. Hence for his functions, with $T \approx tk$, one can achieve average success probability close to 1.

Accordingly, we can only establish direct product theorems in special cases. Examples are Nisan

[*] Supported by Canada's NSERC and MITACS and by DFG grant KL 1470/1.
[†] Supported in part by EU project RESQ, IST-2001-37559.

et al.'s [34] strong direct product theorem for "decision forests", Parnafes et al.'s [36] direct product theorem for "forests" of communication protocols, Shaltiel's strong direct product theorems for "fair" decision trees and the discrepancy bound for communication complexity [41]. In the quantum case, Aaronson [2] established a result for the unordered search problem that lies in between the weak and the strong theorems: every T-query quantum algorithm for searching k marked items among $N = kn$ input bits will have success probability $\sigma \leq O(T^2/N)^k$. In particular, if $T \ll \sqrt{kn}$, then $\sigma = 2^{-\Omega(k)}$.

Our main contributions in this paper are strong direct product theorems for the OR-function in various settings. First consider the case of classical randomized algorithms. Let OR_n denote the n-bit OR-function, and let $f^{(k)}$ denote k independent instances of a function f. Any randomized algorithm with less than, say, $n/2$ queries will have a constant error when computing OR_n. Hence we expect an exponentially small success probability when computing $\mathrm{OR}_n^{(k)}$ using $\ll kn$ queries. We prove this in Section 3:

> **SDPT for classical query complexity:** Every randomized algorithm that computes $\mathrm{OR}_n^{(k)}$ using $T \leq \alpha kn$ queries has worst-case success probability $\sigma = 2^{-\Omega(k)}$ (for $\alpha > 0$ a sufficiently small constant).

For simplicity we stated this with σ being the *worst-case* success probability, but the statement is also valid for the *average* probability under a k-fold product distribution that is implicit in our proof.

This DPT for OR implies a weaker DPT for all *total* functions f, via the notion of *block sensitivity* $bs(f)$. Using techniques of Nisan and Szegedy [35], we can embed $\mathrm{OR}_{bs(f)}$ in f (with the promise that the weight of the input is 0 or 1). On the other hand, the classical bounded-error query complexity $R_2(f)$ is upper bounded by $bs(f)^3$ [7]. This implies:

> Every randomized algorithm that computes $f^{(k)}$ using $T \leq \alpha k R_2(f)^{1/3}$ queries has worst-case success probability $\sigma = 2^{-\Omega(k)}$.

This theorem falls short of a true strong direct product theorem in having $R_2^{1/3}(f)$ instead of $R_2(f)$ in the resource bound. However, the other two important aspects of a SDPT remain valid: the linear dependence of the resources on k and the exponential decay of the success probability.

Next we turn our attention to *quantum* algorithms. Buhrman et al. [16] actually proved that roughly k times the resources for one instance suffices to compute $f^{(k)}$ with success probability *close to 1*, rather than exponentially small: $Q_2(f^{(k)}) = O(kQ_2(f))$, where $Q_2(f)$ denotes the quantum bounded-error query complexity of f (such a result is not known to hold in the classical world). For instance, $Q_2(\mathrm{OR}_n) = \Theta(\sqrt{n})$ by Grover's search algorithm, so $O(k\sqrt{n})$ quantum queries suffice to compute $\mathrm{OR}_n^{(k)}$ with high success probability. In Section 4 we show that if we make the number of queries slightly smaller, the best-achievable success probability suddenly becomes exponentially small:

> **SDPT for quantum query complexity:** Every quantum algorithm that computes $\mathrm{OR}_n^{(k)}$ using $T \leq \alpha k\sqrt{n}$ queries has worst-case success probability $\sigma = 2^{-\Omega(k)}$ (for $\alpha > 0$ a sufficiently small constant).

Our proof uses the polynomial method [7] and is completely different from the classical proof. The polynomial method was also used by Aaronson [2] in his proof of a weaker version, mentioned above. Our proof takes its starting point from his proof, analyzing the degree of a single-variate polynomial that is 0 on $\{0, \dots, k-1\}$, at least σ on k, and between 0 and 1 on $\{0, \dots, kn\}$. The difference between his proof and ours is that we partially factor this polynomial, which gives us some nice extra properties over Aaronson's approach of differentiating the polynomial. In addition, we use a strong result of Coppersmith and Rivlin [20]. In both cases (different) extremal properties of Chebyshev polynomials finish the proofs. Again, we also get a weaker result for all total functions:

> Every quantum algorithm that computes $f^{(k)}$ using $T \leq \alpha k Q_2(f)^{1/6}$ queries has worst-case success probability $\sigma = 2^{-\Omega(k)}$.

The third and last setting where we establish a strong direct product theorem is quantum communication complexity. Suppose Alice has an n-bit input x and Bob has an n-bit input y. These x and y represent sets, and $\mathrm{DISJ}_n(x, y) = 1$ iff those sets are disjoint. Note that DISJ_n is the negation of $\mathrm{OR}_n(x \wedge y)$, where $x \wedge y$ is the n-bit string obtained by bitwise AND-ing x and y. In many ways, DISJ_n has the same central role in communication complexity as OR_n has in query complexity. In particular, it is "co-NP complete" [6]. The communication complexity of DISJ_n has been well studied: it takes $\Theta(n)$ bits of communication in the classical world [24, 38] and $\Theta(\sqrt{n})$ in the quantum world [13, 23, 3, 39]. For the case where Alice and Bob want to compute k instances of Disjointness, we establish a strong direct product theorem in Section 5:

> **SDPT for q. communication complexity:** Every quantum protocol that computes

$\mathrm{DISJ}_n^{(k)}$ communicating $T \leq \alpha k\sqrt{n}$ qubits has worst-case success probability $\sigma = 2^{-\Omega(k)}$.

Our proof uses Razborov's [39] lower bound technique to translate the quantum protocol to a polynomial, at which point the polynomial results established for the quantum query SDPT take over. We can obtain similar results for other symmetric predicates.

One may also consider computing the *parity* of the k outcomes instead of all k outcomes. This issue has been well studied, particularly in circuit complexity, and generally goes under the name of *XOR lemmas* [44, 21]. In this paper we focus on the vector version, but we can prove similar strong bounds for the parity version. In particular, we can get a strong XOR lemma for the quantum case using the technique of Cleve et al. [19, Section 3]. They show how the ability to compute the parity of any subset of k bits with probability $1/2 + \varepsilon$, suffices to compute the full k-vector with probability $4\varepsilon^2$. Hence our strong quantum direct product theorems imply strong quantum XOR lemmas.

1.2. Time-Space and Communication-Space tradeoffs

Apart from answering a fundamental question about the computational models of (quantum) query complexity and communication complexity, our direct product theorems also imply a number of new and optimal time-space tradeoffs.

First, we consider the tradeoff between the time T and space S that a quantum circuit needs for *sorting* N numbers. Classically, it is well known that $TS = \Omega(N^2)$, and this tradeoff is achievable [8]. In the quantum case, Klauck [26] constructed a bounded-error quantum algorithm that runs in time $T = O((N \log N)^{3/2}/\sqrt{S})$ for all $(\log N)^3 \leq S \leq N/\log N$. He also showed[1] a lower bound $TS = \Omega(N^{3/2})$, which is close to optimal for small S but not for large S. We use our strong direct product theorem to prove $T^2S = \Omega(N^3)$. This is tight up to polylog factors.

Secondly, we consider time-space and communication-space tradeoffs for the problems of *Boolean matrix-vector product* and *Boolean matrix product.* In the first problem there are an $N \times N$ matrix A and a vector b of dimension N, and the goal is to compute the vector $c = Ab$, where $c_i = \vee_{j=1}^{N} (A[i,j] \wedge b_j)$. In the setting of time-space tradeoffs, the matrix A is fixed and the input is the vector b. In the problem of matrix multiplication two matrices have to be multiplied with the same type of Boolean product, and both are inputs. Time-space tradeoffs for Boolean matrix-vector multiplication have been analyzed in an average-case scenario by Abrahamson [4], whose results give a worst-case lower bound of $TS = \Omega(N^{3/2})$ for classical algorithms. He conjectured that a worst-case lower bound of $TS = \Omega(N^2)$ holds. Using our classical direct product result we are able to confirm this, i.e., there is a matrix A, such that computing Ab requires $TS = \Omega(N^2)$. We also show a lower bound of $T^2S = \Omega(N^3)$ for this problem in the quantum case. Both bounds are tight (the second within a logarithmic factor) if T is taken to be the number of queries to the inputs. We also get a lower bound of $T^2S = \Omega(N^5)$ for the problem of multiplying two matrices in the quantum case. This bound is close to optimal for small S.

Research on communication-space tradeoffs in the classical setting has been initiated by Lam et al. [31] in a restricted setting, and by Beame et al. [9] in a general model of space-bounded communication complexity. In the setting of communication-space tradeoffs, players Alice and Bob are modeled as space-bounded circuits, and we are interested in the communication cost when given particular space bounds. For the problem of computing the matrix-vector product Alice receives the matrix A (now an input) and Bob the vector b. Beame et al. gave tight lower bounds e.g. for the matrix-vector product and matrix product over GF(2), but stated the complexity of Boolean matrix-vector multiplication as an open problem. Using our direct product result for quantum communication complexity we are able to show that any quantum protocol for this problem satisfies $C^2S = \Omega(N^3)$. This is tight within a polylogarithmic factor. We also get a lower bound of $C^2S = \Omega(N^5)$ for computing the product of two matrices, which again is tight.

No classical lower bounds for these problems were known previously, and finding better classical bounds than these remains open. The possibility to show good quantum bounds comes from the deep relation between quantum protocols and polynomials implicit in [39].

2. Preliminaries

We assume familiarity with quantum computing [32] and sketch the model of query complexity, referring to [18] for more details, also on the close relation between query complexity and degrees of multivariate polynomials. Suppose we want to compute some function f. For input $x \in \{0,1\}^N$, a *query* gives us access

[1] Unfortunately there is an error in the proof presented in [26], namely Lemma 5 appears to be wrong.

to the input bits. It corresponds to the unitary map $O : |i, b, z\rangle \mapsto |i, b \oplus x_i, z\rangle$. Here $i \in [N] = \{1, \ldots, N\}$ and $b \in \{0, 1\}$; the z-part is workspace, which is not affected by the query. We assume the input can be accessed only via such queries. A T-query quantum algorithm has the form $A = U_T O U_{T-1} \cdots O U_1 O U_0$, where the U_k are fixed unitaries, independent of x. This A depends on x via the T applications of O. The algorithm starts in initial S-qubit state $|0\rangle$ and its *output* is the result of measuring a dedicated part of the final state $A|0\rangle$. For a Boolean function f, the output of A is obtained by observing the leftmost qubit of the final superposition $A|0\rangle$, and its *acceptance probability* on input x is its probability of outputting 1. One of the most interesting quantum query algorithms is Grover's search algorithm [22, 10]. It can find an index of a 1-bit in an n-bit input in expected number of $\mathrm{O}\left(\sqrt{n/(|x|+1)}\right)$ queries, where $|x|$ is the Hamming weight (number of ones) in the input. If we know that $|x| \leq 1$, we can solve the search problem exactly using $\frac{\pi}{4}\sqrt{n}$ queries [11].

For investigating time-space tradeoffs we use the circuit model. A circuit accesses its input via an oracle like a query algorithm. Time corresponds to the number of gates in the circuit. We will, however, usually consider the number of queries to the input, which is obviously a lower bound on time. A quantum circuit uses space S if it works with S qubits only. We require that the outputs are made at predefined gates in the circuit, by writing their value to some extra qubits that may not be used later on. Similar definitions are made for classical circuits.

In the model of quantum communication complexity, two players Alice and Bob compute a function f on distributed inputs x and y. The complexity measure of interest in this setting is the amount of communication. The players follow some predefined protocol that consists of local unitary operations, and the exchange of qubits. The communication cost of a protocol is the maximal number of qubits exchanged for any input. In the standard model of communication complexity, Alice and Bob are computationally unbounded, but we are also interested in what happens if they have bounded memory, i.e., they work with a bounded number of qubits. To this end we model Alice and Bob as communicating quantum circuits, following Yao [45].

A pair of communicating quantum circuits is actually a single quantum circuit partitioned into two parts. The allowed operations are local unitary operations and access to the inputs that are given by oracles. Alice's part of the circuit may use oracle gates to read single bits from her input, and Bob's part of the circuit may do so for his input. The communication C between the two parties is simply the number of wires carrying qubits that cross between the two parts of the circuit. A pair of communicating quantum circuits uses space S, if the whole circuit works on S qubits.

In the problems we consider, the number of outputs is much larger than the memory of the players. Therefore we use the following output convention. The player who computes the value of an output sends this value to the other player at a predetermined point in the protocol. In order to make our models as general as possible, we furthermore allow the players to do local measurements, and to throw qubits away as well as pick up some fresh qubits. The space requirement only demands that at any given time no more than S qubits are in use in the whole circuit.

A final comment regarding upper bounds: Buhrman et al. [13] showed how to run a query algorithm in a distributed fashion with small overhead in the communication. In particular, if there is a T-query quantum algorithm computing N-bit function f, then there is a pair of communicating quantum circuits with $O(T \log N)$ communication that computes $f(x \wedge y)$ with the same success probability. We refer to the book of Kushilevitz and Nisan [30] for more on communication complexity in general, and to the surveys [25, 12, 42] for more on its quantum variety.

3. SDPT for Classical Queries

In this section we give the strong direct product theorem for randomized algorithms computing k independent instances of OR_n. Unlike the quantum case, the proof (sketched in Appendix A) is quite straightforward, proving a direct product theorem for non-adaptive algorithms as an intermediate.

Theorem 1 (SDPT for OR**)** *For every $0 < \gamma < 1$, there exists an $\alpha > 0$ such that every randomized algorithm for $\mathrm{OR}_n^{(k)}$ with $T \leq \alpha k n$ queries has success probability $\sigma \leq 2^{-\gamma k}$.*

The strong direct product theorem for OR implies a weaker direct product theorem for all functions. In this weaker version, the success probability of computing k instances still goes down exponentially with k, but we need to start from a polynomially smaller bound on the overall number of queries. For $x \in \{0,1\}^n$ and $S \subseteq [n]$, we use x^S to denote the n-bit string obtained from x by flipping the bits in S. Consider a (possibly partial) function $f : \mathcal{D} \to Z$, with $\mathcal{D} \subseteq \{0,1\}^n$. The *block sensitivity* $bs_x(f)$ of $x \in \mathcal{D}$ is the maximal b for which there are disjoint sets $S_1, \ldots, S_b$ such that $f(x) \neq f(x^{S_i})$. The *block sensitivity* of f is $bs(f) = \max_{x \in \mathcal{D}} bs_x(f)$. Block sensitivity is closely related to deterministic and bounded-error classical query complexity:

Theorem 2 ([33, 7]) *$R_2(f) = \Omega(bs(f))$ for all f, $D(f) \leq bs(f)^3$ for all total Boolean f.*

Nisan and Szegedy [35] showed how to embed a $bs(f)$-bit OR-function (with the promise that the input has weight ≤ 1) into f. Combined with our strong direct product theorem for OR, this implies a direct product theorem for all functions f in terms of $bs(f)$:

Theorem 3 *For every $0 < \gamma < 1$, there exists an $\alpha > 0$ such that for every f, every classical algorithm for $f^{(k)}$ with $T \leq \alpha k bs(f)$ queries has success probability $\sigma \leq 2^{-\gamma k}$.*

This is optimal if $R_2(f) = \Theta(bs(f))$, which is the case for most functions. For total functions, the gap between $R_2(f)$ and $bs(f)$ is not more than cubic, hence

Corollary 4 *For every $0 < \gamma < 1$, there exists an $\alpha > 0$ such that for every total Boolean f, every classical algorithm for $f^{(k)}$ with $T \leq \alpha k R_2(f)^{1/3}$ queries has success probability $\sigma \leq 2^{-\gamma k}$.*

4. SDPT for Quantum Queries

In this section we prove a strong direct product theorem for quantum algorithms computing k independent instances of OR. Our proof relies on the polynomial method of [7]. The following key lemma is proved in Appendix B.

Lemma 5 *Suppose p is a single-variate degree-D polynomial such that for some $\delta \geq 0$*

$-\delta \leq p(i) \leq \delta$ *for all* $i \in \{0, \ldots, k-1\}$,
$p(k) = \sigma$,
$p(i) \in [-\delta, 1+\delta]$ *for all* $i \in \{0, \ldots, N\}$.

Then for every integer $C \in [1, N-k)$ and $\mu = 2C/(N-k-C)$ we have

$$\sigma \leq \delta k 2^{k-1} + a\left(1 + \delta + \frac{\delta(2N)^k}{(k-1)!}\right) \cdot$$

$$\exp\left(\frac{b(D-k)^2}{(N-k-C)} + 2(D-k)\sqrt{2\mu + \mu^2} - k\ln(C/k)\right),$$

where a, b are the constants of Theorem 23 (Appendix B).

We will apply this lemma with δ negligibly small, $D = \alpha\sqrt{kN}$ for small α, and $C = ke^{\gamma+1}$, giving

$$\sigma \leq \exp\left((b\alpha^2 + 4\alpha e^{\gamma/2+1/2} - 1 - \gamma)k\right) \leq e^{-\gamma k} \leq 2^{-\gamma k}.$$

This will imply a strong tradeoff between queries and success probability for quantum algorithms that have to find k ones in an N-bit input. A *k-threshold algorithm with success probability σ* is an algorithm on N-bit input x, that outputs 0 with certainty if $|x| < k$, and outputs 1 with probability at least σ if $|x| = k$.

Theorem 6 *For every $\gamma > 0$, there exists an $\alpha > 0$ such that every quantum k-threshold algorithm with $T \leq \alpha\sqrt{kN}$ queries has success probability $\sigma \leq 2^{-\gamma k}$.*

Proof. Fix $\gamma > 0$ and consider a T-query k-threshold algorithm. By [7], its acceptance probability is an N-variate polynomial of degree $D \leq 2T \leq 2\alpha\sqrt{kN}$ and can be symmetrized to a single-variate polynomial p with the properties

$p(i) = 0$ if $i \in \{0, \ldots, k-1\}$
$p(k) \geq \sigma$
$p(i) \in [0, 1]$ for all $i \in \{0, \ldots, N\}$

Choosing $\alpha > 0$ sufficiently small and $\delta = 0$, the result follows from Lemma 5. □

This implies a strong direct product theorem for k instances of the n-bit search problem:

Theorem 7 (SQDPT for Search) *For every $\gamma > 0$, there exists an $\alpha > 0$ such that every quantum algorithm for $\mathrm{Search}_n^{(k)}$ with $T \leq \alpha k\sqrt{n}$ queries has success probability $\sigma \leq 2^{-\gamma k}$.*

Proof. Set $N = kn$, fix a $\gamma > 0$ and a T-query algorithm A for $\mathrm{Search}_n^{(k)}$ with success probability σ. Now consider the following algorithm on N-bit input x:

1. Apply a random permutation π to x.
2. Run A on $\pi(x)$.
3. Query each of the k positions that A outputs, return 1 iff at least $k/2$ of those bits are 1.

This uses $T + k$ queries. We will show that it is a $k/2$-threshold algorithm. If $|x| < k/2$, it always outputs 0. If $|x| = k/2$, the probability that π puts all $k/2$ ones in distinct n-bit blocks is

$$\frac{N}{N} \cdot \frac{N-n}{N-1} \cdots \frac{N - \frac{k}{2}n}{N - \frac{k}{2}} \geq \left(\frac{N - \frac{k}{2}n}{N}\right)^{k/2} = 2^{-k/2}.$$

Hence our algorithm outputs 1 with probability at least $\sigma 2^{-k/2}$. Choosing α sufficiently small, the previous theorem implies $\sigma 2^{-k/2} \leq 2^{-(\gamma+1/2)k}$, hence $\sigma \leq 2^{-\gamma k}$. □

Our bounds are quite precise for $\alpha \ll 1$. We can choose $\gamma = 2\ln(1/\alpha) - O(1)$ and ignore some lower-order terms to get roughly $\sigma \leq \alpha^{2k}$. On the other hand, it is known that Grover's search algorithm with $\alpha\sqrt{n}$ queries on an n-bit input has success probability roughly α^2 [10]. Doing such a search on all k instances gives overall success probability α^{2k}.

Theorem 8 (SQDPT for OR) *There exist $\alpha, \gamma > 0$ such that every quantum algorithm for $\mathrm{OR}_n^{(k)}$ with $T \leq \alpha k\sqrt{n}$ queries has success probability $\sigma \leq 2^{-\gamma k}$.*

Proof. An algorithm A for $\mathrm{OR}_n^{(k)}$ with success probability σ can be used to build an algorithm A' for $\mathrm{Search}_n^{(k)}$ with slightly worse success probability:

1. Run A on the original input and remember which blocks contain a 1.
2. Run simultaneously (at most k) binary searches on the nonzero blocks. Iterate this $s = 2\log(1/\alpha)$ times. Each iteration runs A on the parts of the blocks that are known to contain a 1, halving the remaining instance size each time.
3. Run the exact version of Grover's algorithm on each of the remaining parts of the instances to look for a one there (each part has size $n/2^s$).

This new algorithm A' uses $(s+1)T + \frac{\pi}{4}k\sqrt{n/2^s} = \mathrm{O}(\alpha\log(1/\alpha)k\sqrt{n})$ queries. With probability at least σ^{s+1}, A succeeds in all iterations, in which case A' solves $\mathrm{Search}_n^{(k)}$. By Theorem 7, for every $\gamma' > 0$ there is an $\alpha > 0$ such that $\sigma^{s+1} \le 2^{-\gamma' k}$. This gives the theorem with $\gamma = \gamma'/(s+1)$. □

Choosing parameters carefully, we can show that for every $\gamma < 1$ there is an α such that $\alpha k\sqrt{n}$ queries give $\sigma \le 2^{-\gamma k}$. Clearly, $\sigma = 2^{-k}$ is achievable without any queries by random guessing.

As in the classical case, we also get weaker bounds for all functions, using the following results from [7]: $Q_2(f) = \Omega\left(\sqrt{bs(f)}\right)$ for all f and $D(f) \le bs(f)^3$ for all total Boolean f.

Theorem 9 *There exist $\alpha, \gamma > 0$ such that for every f, every quantum algorithm for $f^{(k)}$ with $T \le \alpha k\sqrt{bs(f)}$ queries has success probability $\sigma \le 2^{-\gamma k}$.*

Corollary 10 *There exist $\alpha, \gamma > 0$ such that for every total Boolean f, every quantum algorithm for $f^{(k)}$ with $T \le \alpha k Q_2(f)^{1/6}$ queries has success probability $\sigma \le 2^{-\gamma k}$.*

5. SDPT for Quantum Communication

Here we establish a strong direct product theorem for quantum communication, specifically for protocols that compute k independent instances of the Disjointness problem. Our proof relies crucially on the beautiful technique that Razborov introduced to lower bound the quantum communication complexity of one instance of Disjointness [39]. It allows us to translate a quantum communication protocol to a single-variate polynomial that represents, roughly speaking, the protocol's acceptance probability as a function of the size of the intersection of x and y. The following lemma is implicit in Razborov's paper (see our long version [1]).

Lemma 11 *Consider a Q-qubit quantum communication protocol on N-bit inputs x and y, with acceptance probabilities $P(x, y)$. Define $P(i) = \mathrm{E}_{|x|=|y|=N/4, |x\wedge y|=i}[P(x, y)]$, with expectation taken uniformly over all x, y that each have weight $N/4$ and that have intersection i. For every $d \le N/4$ there exists a degree-d polynomial q such that $|P(i) - q(i)| \le 2^{-d/4+2Q}$ for all $i \in \{0, \ldots, N/8\}$.*

Theorem 12 (SQDPT for Disjointness) *There exist $\alpha, \gamma > 0$ such that every quantum protocol for $\mathrm{DISJ}_n^{(k)}$ with $Q \le \alpha k\sqrt{n}$ qubits of communication has success probability $p \le 2^{-\gamma k}$.*

Proof (sketch). By doing the same trick with $s = 2\log(1/\alpha)$ rounds of binary search as for Theorem 8, we can tweak a protocol for $\mathrm{DISJ}_n^{(k)}$ to a protocol that satisfies (with $P(i)$ defined as in Lemma 11, $N = kn$ and $\sigma = p^{s+1}$) $P(i) = 0$ if $i \in \{0, \ldots, k-1\}$; $P(k) \ge \sigma$; $P(i) \in [0, 1]$ for all $i \in \{0, \ldots, N\}$. Instead of exact Grover we use an exact version of the $\mathrm{O}(\sqrt{n})$-qubit Disjointness protocol of [3] (the [13]-protocol would lose a $\log n$-factor). Lemma 11, using $d = 12Q$, then gives a degree-d polynomial q that differs from P by at most $\delta \le 2^{-Q}$ on all $i \in \{0, \ldots, N/8\}$. This δ is sufficiently small to apply Lemma 5, which in turn upper bounds σ and hence p. □

6. Time-Space Tradeoff for Quantum Sorting

We will now use our strong direct product theorem to get near-optimal time-space tradeoffs for quantum circuits for sorting. This follows Klauck [26], who described an upper bound $T^2S = \mathrm{O}\big((N\log N)^3\big)$ and a lower bound $TS = \Omega\big(N^{3/2}\big)$. In our model, the numbers $a_1, \ldots, a_N$ that we want to sort can be accessed by means of queries, and the number of queries lower bounds the actual time taken by the circuit. The circuit has N output gates and in the course of its computation outputs the N numbers in sorted (say, descending) order, with success probability at least 2/3.

Theorem 13 *Every bounded-error quantum circuit for sorting N numbers that uses T queries and S qubits of workspace satisfies $T^2S = \Omega\big(N^3\big)$.*

Proof. We "slice" the circuit along the time-axis into $L = T/\alpha\sqrt{SN}$ slices, each containing $T/L = \alpha\sqrt{SN}$ queries. Each such slice has a number of output gates. Consider any slice. Suppose it contains output gates $i, i+1, \ldots, i+k-1$, for $i \le N/2$, so it is supposed to output the i-th up to $i+k-1$-th largest elements of its input. We want to show that $k = \mathrm{O}(S)$. If $k \le S$

then we are done, so assume $k > S$. We can use the slice as a k-threshold algorithm on $N/2$ bits, as follows. For an $N/2$-bit input x, construct a sorting input by taking $i-1$ copies of the number 2, the $N/2$ bits in x, and $N/2 - i + 1$ copies of the number 0, and append their position behind the numbers.

Consider the behavior of the sorting circuit on this input. The first part of the circuit has to output the $i-1$ largest numbers, which all start with 2. We condition on the event that the circuit succeeds in this. It then passes on an S-qubit state (possibly mixed) as the starting state of the particular slice we are considering. This slice then outputs the k largest numbers in x with probability at least $2/3$. Now, consider an algorithm that runs just this slice, starting with the completely mixed state on S-qubits, and that outputs 1 if it finds k numbers starting with 1, and outputs 0 otherwise. If $|x| < k$ this new algorithm always outputs 0 (note that it can verify finding a 1 since its position is appended), but if $|x| = k$ then it outputs 1 with probability at least $\sigma \geq \frac{2}{3} \cdot 2^{-S}$, because the completely mixed state has “overlap” 2^{-S} with the “good” S-qubit state that would have been the starting state of the slice in the run of the sorting circuit. On the other hand, the slice has only $\alpha\sqrt{SN} < \alpha\sqrt{kN}$ queries, so by choosing α sufficiently small, Theorem 6 implies $\sigma \leq 2^{-\Omega(k)}$. Combining our upper and lower bounds on σ gives $k = O(S)$. Thus we need $L = \Omega(N/S)$ slices, so $T = L\alpha\sqrt{SN} = \Omega\left(N^{3/2}/\sqrt{S}\right)$. □

As mentioned, our tradeoff is achievable up to polylog factors [26]. Interestingly, the near-optimal algorithm uses only a polylogarithmic number of qubits and otherwise just classical memory. For simplicity we have shown the lower bound for the case when the outputs have to be made in their natural ordering only, but we can show the same lower bound for any ordering of the outputs that does not depend on the input using a slightly different proof.

7. Time-Space Tradeoffs for Boolean Matrix Products

First we give a lower bound on the time-space tradeoff for Boolean matrix-vector multiplication on *classical* machines. For reasons of space we omit the proofs in this section and the next. They use the same approach as before: slice the circuit into small slices, and use a strong direct product theorem to show that each slice can only produce few outputs (hence we need many slices). Details may be found in our long version [1].

Theorem 14 *There is a matrix A such that every classical bounded-error circuit that computes the Boolean matrix-vector product Ab with T queries and space $S = o(N/\log N)$ satisfies $TS = \Omega(N^2)$.*

The bound is tight if T measures queries to the input. An absolutely analogous construction can be done in the quantum case.

Theorem 15 *There is a matrix A such that every quantum bounded-error circuit that computes the Boolean matrix-vector product Ab with T queries and space $S = o(N/\log N)$ satisfies $T^2S = \Omega(N^3)$.*

This is tight within a log-factor (needed to improve the success probability of Grover search).

Theorem 16 *Every classical bounded-error circuit that computes the Boolean matrix product AB with T queries and space S satisfies $TS = \Omega(N^3)$.*

While this is near-optimal for small S, it is probably not tight for large S, a likely tight tradeoff being $T^2S = \Omega(N^6)$. It is also no improvement compared to the average-case bounds of [4]. The application to the quantum case is analogous.

Theorem 17 *Every quantum bounded-error circuit that computes the Boolean matrix product AB with T queries and space S satisfies $T^2S = \Omega(N^5)$.*

If $S = O(\log N)$, then N^2 applications of Grover can compute AB with $T = O\left(N^{2.5}\log N\right)$. Hence our tradeoff is near-optimal for small S. We do not know whether it is optimal for large S.

8. Quantum Communication-Space Tradeoffs for Matrix Products

In this section we use the strong direct product result for quantum communication (Theorem 12) to prove tight communication-space tradeoffs.

Theorem 18 *Every quantum bounded-error protocol in which Alice and Bob have bounded space S and that computes the Boolean matrix-vector product, satisfies $C^2S = \Omega(N^3)$.*

Theorem 19 *Every quantum bounded-error protocol in which Alice and Bob have bounded space S and that computes the Boolean matrix product, satisfies $C^2S = \Omega(N^5)$.*

Theorem 20 *There is a quantum bounded-error protocol with space S that computes the Boolean product between a matrix and a vector within communication $C = O((N^{3/2}\log^2 N)/\sqrt{S})$. There is a quantum bounded-error protocol with space S that computes the Boolean product between two matrices within communication $C = O((N^{5/2}\log^2 N)/\sqrt{S})$.*

9. Open Problems

We mention some open problems. The first is to determine tight time-space tradeoffs for Boolean matrix product on both classical and quantum computers. Second, regarding communication-space tradeoffs for Boolean matrix-vector and matrix product, we did not prove any classical bounds that were better than our quantum bounds. Klauck [27] recently proved classical tradeoffs $CS^2 = \Omega\big(N^3\big)$ and $CS^2 = \Omega\big(N^2\big)$ for Boolean matrix product and matrix-vector product, respectively, by means of a *weak* direct product theorem for Disjointness. A classical *strong* direct product theorem for Disjointness would imply optimal tradeoffs, but we do not know how to prove this at the moment. Finally, it would be interesting to get any lower bounds on time-space or communication-space tradeoffs for decision problems in the quantum case, for example for Element Distinctness [15, 5] or the verification of matrix multiplication [17].

Acknowledgments

We thank Scott Aaronson for email discussions about the evolving results in his [2] that motivated some of our proofs, and Harry Buhrman for useful discussions.

References

[1] http://arxiv.org/abs/quant-ph/0402123.

[2] S. Aaronson. Limitations of quantum advice and one-way communication. In *Proc. of 19th Conf. on Computational Complexity*, 2004. To appear. quant-ph/0402095.

[3] S. Aaronson and A. Ambainis. Quantum search of spatial regions. In *Proc. of 44th FOCS*, pages 200–209, 2003. quant-ph/0303041.

[4] K. Abrahamson. A time-space tradeoff for Boolean matrix multiplication. In *Proc. of 31st FOCS*, pages 412–419, 1990.

[5] A. Ambainis. Quantum walk algorithm for element distinctness. These Proceedings. quant-ph/0311001.

[6] L. Babai, P. Frankl, and J. Simon. Complexity classes in communication complexity theory. In *Proc. of 27th FOCS*, pages 337–347, 1986.

[7] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. Earlier version in FOCS'98. quant-ph/9802049.

[8] P. Beame. A general sequential time-space tradeoff for finding unique elements. *SIAM Journal on Computing*, 20(2):270–277, 1991. Earlier version in STOC'89.

[9] P. Beame, M. Tompa, and P. Yan. Communication-space tradeoffs for unrestricted protocols. *SIAM Journal on Computing*, 23(3):652–661, 1994. Earlier version in FOCS'90.

[10] M. Boyer, G. Brassard, P. Høyer, and A. Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46(4–5):493–505, 1998. Earlier version in Physcomp'96. quant-ph/9605034.

[11] G. Brassard, P. Høyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Quantum Information: A Millennium Volume*, volume 305 of *AMS Contemporary Mathematics Series*, pages 53–74. 2002. quant-ph/0005055.

[12] H. Buhrman. Quantum computing and communication complexity. *EATCS Bulletin*, 70:131–141, 2000.

[13] H. Buhrman, R. Cleve, and A. Wigderson. Quantum vs. classical communication and computation. In *Proc. of 30th STOC*, pages 63–68, 1998. quant-ph/9802040.

[14] H. Buhrman, R. Cleve, R. de Wolf, and C. Zalka. Bounds for small-error and zero-error quantum algorithms. In *Proc. of 40th FOCS*, pages 358–368, 1999. cs.CC/9904019.

[15] H. Buhrman, C. Dürr, M. Heiligman, P. Høyer, F. Magniez, M. Santha, and R. de Wolf. Quantum algorithms for element distinctness. In *Proc. of 16th Conf. on Computational Complexity*, pages 131–137, 2001. quant-ph/0007016.

[16] H. Buhrman, I. Newman, H. Röhrig, and R. de Wolf. Robust quantum algorithms and polynomials. quant-ph/0309220, 2003.

[17] H. Buhrman and R. Špalek. Quantum verification of matrix products. Manuscript, 2004.

[18] H. Buhrman and R. de Wolf. Complexity measures and decision tree complexity: A survey. *Theoretical Computer Science*, 288(1):21–43, 2002.

[19] R. Cleve, W. van Dam, M. Nielsen, and A. Tapp. Quantum entanglement and the communication complexity of the inner product function. In *Proc. of 1st NASA QCQC conference*, volume 1509 of *LNCS*, pages 61–74. Springer, 1998. quant-ph/9708019.

[20] D. Coppersmith and T. J. Rivlin. The growth of polynomials bounded at equally spaced points. *SIAM Journal on Mathematical Analysis*, 23(4):970–983, 1992.

[21] O. Goldreich, N. Nisan, and A. Wigderson. On Yao's XOR lemma. Technical report, ECCC TR–95–050, 1995. Available at http://www.eccc.uni-trier.de/eccc/.

[22] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proc. of 28th STOC*, pages 212–219, 1996. quant-ph/9605043.

[23] P. Høyer and R. de Wolf. Improved quantum communication complexity bounds for disjointness and equality. In *Proc. of 19th STACS*, LNCS 2285, pages 299–310. Springer, 2002. quant-ph/0109068.

[24] B. Kalyanasundaram and G. Schnitger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992. Earlier version in Structures'87.

[25] H. Klauck. Quantum communication complexity. In *Workshop on Boolean Functions and Applications at 27th ICALP*, pages 241–252, 2000. quant-ph/0005032.

[26] H. Klauck. Quantum time-space tradeoffs for sorting. In *Proc. of 35th STOC*, pages 69–76, 2003. quant-ph/0211174.

[27] H. Klauck. Quantum and classical communication-space tradeoffs from rectangle bounds. Manuscript, 2004.

[28] D. E. Knuth. Combinatorial matrices. In *Selected Papers on Discrete Mathematics*, volume 106 of *CSLI Lecture Notes*. Stanford University, 2003.

[29] I. Kremer. Quantum communication. Master's thesis, Hebrew University, Computer Science Depart., 1995.

[30] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[31] T. Lam, P. Tiwari, and M. Tompa. Trade-offs between communication and space. *Journal of Computer and Systems Sciences*, 45(3):296–315, 1992. Earlier version in STOC'89.

[32] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[33] N. Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991. Earlier version in STOC'89.

[34] N. Nisan, S. Rudich, and M. Saks. Products and help bits in decision trees. In *Proc. of 35th FOCS*, pages 318–329, 1994.

[35] N. Nisan and M. Szegedy. On the degree of Boolean functions as real polynomials. *Computational Complexity*, 4(4):301–313, 1994. Earlier version in STOC'92.

[36] I. Parnafes, R. Raz, and A. Wigderson. Direct product results and the GCD problem, in old and new communication models. In *Proc. of 29th STOC*, pages 363–372, 1997.

[37] R. Paturi. On the degree of polynomials that approximate symmetric Boolean functions. In *Proc. of 24th STOC*, pages 468–474, 1992.

[38] A. Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992.

[39] A. Razborov. Quantum communication complexity of symmetric predicates. *Izvestiya of the Russian Academy of Science*, 67(1):159–176, 2003. quant-ph/0204025.

[40] T. J. Rivlin. *Chebyshev Polynomials: From Approximation Theory to Algebra and Number Theory*. Wiley-Interscience, second edition, 1990.

[41] R. Shaltiel. Towards proving strong direct product theorems. In *Proc. of 16th Conf. on Computational Complexity*, pages 107–119, 2001.

[42] R. de Wolf. Quantum communication and complexity. *Theoretical Computer Science*, 287(1):337–353, 2002.

[43] A. C.-C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proc. of 18th FOCS*, pages 222–227, 1977.

[44] A. C.-C. Yao. Theory and applications of trapdoor functions. In *Proc. of 23rd FOCS*, pages 80–91, 1982.

[45] A. C.-C. Yao. Quantum circuit complexity. In *Proc. of 34th FOCS*, pages 352–360, 1993.

A. Proofs from Section 3

Here we sketch the strong direct product theorem for classical randomized algorithms that compute k independent instances of OR_n, referring to [1] for a more detailed proof. By Yao's principle, it is sufficient to prove it for deterministic algorithms under a fixed hard input distribution. Let $\mathrm{Suc}_{t,\mu}(f)$ be the success probability of the best algorithm for f under μ that queries $\leq t$ input bits. We call an algorithm *non-adaptive* if, for each of the k input blocks, the maximum number of queries in that block is fixed before the first query. By induction, as in [41], we can prove:

Lemma 21 *Let $f : \{0,1\}^n \to \{0,1\}$ and μ be an input distribution. Every non-adaptive deterministic algorithm for $f^{(k)}$ under μ^k with $T \leq kt$ queries has success probability $\sigma \leq \mathrm{Suc}_{t,\mu}(f)^k$.*

Remark. A similar statement is not always true for adaptive algorithms. Following [41], define $h(x) = x_1 \vee (x_2 \oplus \ldots \oplus x_n)$. Clearly $\mathrm{Suc}_{\frac{2}{3}n,\mu}(h) = 3/4$ for μ uniform. By a Chernoff bound, $\mathrm{Suc}_{\frac{2}{3}nk,\mu^k}(h^{(k)}) = 1-2^{-\Omega(k)}$, because approximately half of the blocks can be solved using just 1 query and the unused queries can be used to answer exactly also the other half of the blocks.

However, the SDPT is valid for $\mathrm{OR}_n^{(k)}$ under ν^k, where $\nu(0^n) = 1/2$ and $\nu(e_i) = 1/2n$ for e_i an n-bit string that contains a 1 only at the i-th position. It is simple to prove that $\mathrm{Suc}_{\alpha n,\nu}(\mathrm{OR}_n) = \frac{\alpha+1}{2}$. Non-adaptive algorithms for $\mathrm{OR}_n^{(k)}$ under ν^k with αkn queries thus have $\sigma \leq (\frac{\alpha+1}{2})^k = 2^{-\log(\frac{2}{\alpha+1})k}$. We can achieve any $\gamma < 1$ by choosing α sufficiently small. We prove that adaptive algorithms cannot be much better. Without loss of generality, we assume: (1) The adaptive algorithm is deterministic. (2) Whenever the algorithm finds a 1 in some input block, it stops querying that block. (3) The algorithm spends the same number of queries in all blocks where it does not find a 1. This is optimal due to the symmetry between the blocks, and implies that the algorithm spends at least as many queries in each "empty" input block as in each "non-empty" block.

Lemma 22 *If there is an adaptive T-query algorithm A computing $\mathrm{OR}_n^{(k)}$ under ν^k with success probability σ, then there is a non-adaptive $3T$-query algorithm A' computing it with success probability $\sigma - 2^{-\Omega(k)}$.*

Proof. Let Z be the number of empty blocks. $\mathrm{E}[Z] = k/2$ and, by a Chernoff bound, $\delta = \Pr[Z < k/3] = 2^{-\Omega(k)}$. If $Z \geq k/3$, then A spends at most $3T/k$ queries in each empty block. Define non-adaptive A' that spends $3T/k$ queries in *each* block. Then A' queries all the positions that A queries, and maybe some more. Let us compare the overall success probabilities of A and A':

$$\begin{aligned}\sigma_A &= \Pr[Z < k/3] \cdot \Pr[A \text{ succeeds} \mid Z < k/3] \\ &\quad + \Pr[Z \geq k/3] \cdot \Pr[A \text{ succeeds} \mid Z \geq k/3] \\ &\leq \delta \cdot 1 + \Pr[Z \geq k/3] \cdot \Pr[A' \text{ succeeds} \mid Z \geq k/3] \\ &\leq \delta + \sigma_{A'}.\end{aligned}$$

We conclude that $\sigma_{A'} \geq \sigma_A - \delta$. (*Remark.* By replacing the $k/3$-bound on Z by a βk-bound for some $\beta > 0$, we can obtain arbitrary $\gamma < 1$ in the exponent $\delta = 2^{-\gamma k}$, while the number of queries of A' becomes T/β.) □

Combining the two lemmas establishes Theorem 1.

B. Proofs from Section 4

We use three results about polynomials, also used in [14]. The first is by Coppersmith and Rivlin [20, p. 980] and gives a general bound for polynomials bounded by 1 at integer points:

Theorem 23 (Coppersmith & Rivlin [20]) *Every polynomial p of degree $d \leq n$ that has absolute value $|p(i)| \leq 1$ for all integers $i \in [0, n]$, satisfies $|p(x)| < ae^{bd^2/n}$ for all real $x \in [0, n]$, where $a, b > 0$ are universal constants (no explicit values for a and b are given in [20]).*

The other two results concern the Chebyshev polynomials T_d, defined as in [40]:

$$T_d(x) = \frac{1}{2}\left(\left(x + \sqrt{x^2-1}\right)^d + \left(x - \sqrt{x^2-1}\right)^d\right).$$

T_d has degree d and its absolute value $|T_d(x)|$ is bounded by 1 if $x \in [-1, 1]$. On the interval $[1, \infty)$, T_d exceeds all others polynomials with those two properties ([40, p.108] and [37, Fact 2]):

Theorem 24 *If q is a polynomial of degree d such that $|q(x)| \leq 1$ for all $x \in [-1, 1]$ then $|q(x)| \leq |T_d(x)|$ for all $x \geq 1$.*

Paturi [37, before Fact 2] proved

Lemma 25 (Paturi [37]) *$T_d(1+\mu) \leq e^{2d\sqrt{2\mu+\mu^2}}$ for all $\mu \geq 0$.*

Proof. For $x = 1 + \mu$: $T_d(x) \leq (x + \sqrt{x^2-1})^d = (1+\mu+\sqrt{2\mu+\mu^2})^d \leq (1+2\sqrt{2\mu+\mu^2})^d \leq e^{2d\sqrt{2\mu+\mu^2}}$ (using that $1 + z \leq e^z$ for all real z). □

These tools allow us to establish the key lemma.

Proof of Lemma 5. Divide p with remainder by $\prod_{j=0}^{k-1}(x-j)$ to obtain

$$p(x) = q(x)\prod_{j=0}^{k-1}(x-j) + r(x),$$

where $d = \deg(q) = D - k$ and $\deg(r) \leq k - 1$. We know that $r(x) = p(x) \in [-\delta, \delta]$ for all $x \in \{0, \ldots, k-1\}$. Decompose r as a linear combination of polynomials e_i, where $e_i(i) = 1$ and $e_i(x) = 0$ for $x \in \{0, \ldots, k-1\} - \{i\}$:

$$r(x) \;=\; \sum_{i=0}^{k-1} p(i)e_i(x) = \sum_{i=0}^{k-1} p(i) \prod_{\substack{j=0\\ j\neq i}}^{k-1} \frac{x-j}{i-j}.$$

We bound the values of r for all real $x \in [0, N]$ by

$$\begin{aligned}|r(x)| &\leq \sum_{i=0}^{k-1} \frac{|p(i)|}{i!(k-1-i)!} \prod_{\substack{j=0\\ j\neq i}}^{k-1} |x-j| \\ &\leq \frac{\delta}{(k-1)!} \sum_{i=0}^{k-1} \binom{k-1}{i} N^k \leq \frac{\delta(2N)^k}{(k-1)!}, \\ |r(k)| &\leq \delta k 2^{k-1}.\end{aligned}$$

This implies the following about the values of q:

$$\begin{aligned}|q(k)| &\geq (\sigma - \delta k 2^{k-1})/k! \\ |q(i)| &\leq \frac{(i-k)!}{i!}\left(1 + \delta + \frac{\delta(2N)^k}{(k-1)!}\right) \\ &\quad \text{for } i \in \{k, \ldots, N\}\end{aligned}$$

In particular:

$$\begin{aligned}|q(i)| &\leq C^{-k}\left(1 + \delta + \frac{\delta(2N)^k}{(k-1)!}\right) = A \\ &\quad \text{for } i \in \{k + C, \ldots, N\}\end{aligned}$$

Theorem 23 implies that there are $a, b > 0$ such that

$$|q(x)| \leq A \cdot ae^{bd^2/(N-k-C)} = B$$

for all real $x \in [k + C, N]$. We now divide q by B to normalize it, and rescale the interval $[k + C, N]$ to $[1, -1]$ to get a degree-d polynomial t satisfying

$$\begin{aligned}|t(x)| &\leq 1 && \text{for all } x \in [-1, 1] \\ t(1+\mu) &= q(k)/B && \text{for } \mu = 2C/(N-k-C)\end{aligned}$$

Since t cannot grow faster than the degree-d Chebyshev polynomial, we get

$$t(1+\mu) \leq T_d(1+\mu) \leq e^{2d\sqrt{2\mu+\mu^2}}.$$

Combining our upper and lower bounds on $t(1 + \mu)$:

$$\frac{(\sigma - \delta k 2^{k-1})/k!}{C^{-k}\left(1 + \delta + \frac{\delta(2N)^k}{(k-1)!}\right) ae^{bd^2/(N-k-C)}} \leq e^{2d\sqrt{2\mu+\mu^2}}.$$

Rearranging gives the bound. □