QUANTUM ALGORITHMS FOR ELEMENT DISTINCTNESS*

HARRY BUHRMAN‡, CHRISTOPH DÜRR†, MARK HEILIGMAN§, PETER HØYER¶, FRÉDÉRIC MAGNIEZ‖, MIKLOS SANTHA**, AND RONALD DE WOLF††

Abstract. We present several applications of quantum amplitude amplification for deciding whether all elements in the image of a given function are distinct, for finding an intersection of two sorted tables and for finding a triangle in a graph. Our techniques generalize and improve those of Brassard, Høyer, and Tapp. This shows that in the quantum world element distinctness is significantly easier than sorting, in contrast to the classical world.

1. Introduction. In the last decade, quantum computing has become a prominent and promising area of theoretical computer science. Realizing this promise requires two things: actually building a quantum computer and discovering tasks where a quantum computer is significantly faster than a classical computer. Here we are concerned with the second issue. Few good quantum algorithms are known to date. The two main examples are Shor's algorithm for factoring [24], which achieves an exponential speed-up over the best known classical factoring algorithms, and Grover's search algorithm [15], which achieves a quadratic speed-up over classical search algorithms. Whereas the first so far has remained a seminal but somewhat isolated result, the second has been applied as a building block in quite a few other quantum algorithms [6, 8, 9, 10, 21, 20, 7, 12].

The security of the widely used cryptosystem RSA is based on the assumption that it is hard to factor integers. Shor's algorithm solves precisely this task. In the same flavor, the security of digital signatures is based on the assumption that it is difficult to find two items which map to the same value for some particular function. This motivates the research on the quantum complexity of this task. We define different variants of this problem. Though we do not improve the bounds for the following problem, we define it first to start our explanation. We use $[N]$ to denote $\{1, \ldots, N\}$.

COLLISION PROBLEM
input $f : [N] \to [M]$ which is 2-to-1, i.e. $\forall i \in [N] \exists! j \in [N], i \neq j : f(i) = f(j)$
output $i, j \in [N]$ with $i \neq j$ and $f(i) = f(j)$
complexity Classically the bounded-error query complexity is $\Theta(N^{1/2})$. For a quantum computer the bounded-error query complexity is $\Theta(N^{1/3})$: In 1997 Brassard, Høyer, Tapp [8] gave a bounded-error quantum algorithm using $O(N^{1/3})$ queries to f and in 2002 Shi [23] showed the matching lower bound.

*A preliminary version of this paper appeared in *Proceedings of the 16th IEEE Conference on Computational Complexity*, pp. 131–137, 2001.

Research partially supported by the EU 5th framework programs QAIP IST-1999-11234, and RAND-APX IST-1999-14036.

‡CWI, P.O.Box 94079, Amsterdam, the Netherlands. Also affiliated with the University of Amsterdam. Email: buhrman@cwi.nl.

†Université Paris-Sud, LRI, 91405 Orsay, France. Email: durr@lri.fr.

§NSA, Suite 6111, Fort George G. Meade, MD 20755, USA. Email: miheili@nsa.gov.

¶Dept. of Comp. Sci., University of Calgary, Alberta, Canada T2N 1N4. Email: hoyer@cpsc.ucalgary.ca. Research conducted while at BRICS, University of Aarhus, Denmark.

‖CNRS–LRI, UMR 8623 Université Paris-Sud, 91405 Orsay, France. Email: magniez@lri.fr.

**CNRS–LRI, UMR 8623 Université Paris-Sud, 91405 Orsay, France. Email: santha@lri.fr.

††CWI, P.O.Box 94079, Amsterdam, the Netherlands. Email: rdewolf@cwi.nl.

In the following problem we remove the assumption about the input. The birthday paradox gives a simple relation between both problems. A random subset of size $\sqrt{N}$ of the domain of any 2-to-1 function contains with high probability a collision pair. Therefore any bounded-error algorithm for *Element Distinctness* using $O(N^\alpha)$ queries implies a bounded-error algorithm for the *Collision Problem* using $O(N^{\alpha/2})$ queries.

ELEMENT DISTINCTNESS
input $f : [N] \to [M]$
output $i, j \in [N]$ with $i \neq j$ and $f(i) = f(j)$, or "all distinct" if f is injective
complexity We present a bounded-error quantum algorithm which makes $O(N^{3/4})$ queries. It dates from early 2000, and first appeared in [11]. However, recently the bounded-error quantum query complexity was shown to be $\Theta(N^{2/3})$: The lower bound follows from Shi [23] by the observation above and an algorithm matching this bound was found in 2003 by Ambainis [3] using a quantum walk. The classical bounded-error query complexity is $\Theta(N)$ by a trivial reduction from the OR-problem: For an OR-instance $x \in \{0,1\}^N$ we define the function $f : [N+1] \to [N+1]$ where $f(N+1) = 0$ and for all $i \in [N]$ $f(i) = (1 - x_i)i$. Now $\mathrm{OR}(x) = 1$ iff f contains a collision pair.

The element distinctness problem has been well studied classically [25, 18, 14, 5]. It is particularly interesting because its classical complexity is related to that of sorting, which is well known to require $N \log N + \Theta(N)$ comparisons in the classical world. If we sort f, we can decide element distinctness by going through the sorted list once, which gives a classical upper bound of $N \log N + O(N)$ comparisons. Conversely, element distinctness requires $\Omega(N \log N)$ comparisons in case of classical bounded-error algorithms (even in a much stronger model [14]), so sorting and element distinctness are essentially equally hard classically. On a quantum computer, the best known upper bound for sorting is $0.53\ N \log N$ comparisons [13], and such a linear speed-up is best possible: quantum sorting requires $\Omega(N \log N)$ comparisons, even if one allows a small probability of error [16]. Accordingly, our $O(N^{3/4} \log N)$ upper bound shows that element distinctness is significantly easier than sorting for a quantum computer, in contrast to the classical case.

In this paper we also give algorithms for related problems. Typically, web search engines like Google associate to every word a list of pages containing it, sorted in order of its page rank, and when the query is "Rolling Stones" for example, then the search engine must output the intersection of the lists associated to the words "Rolling" and "Stones". Now imagine a search engine implemented on a quantum computer. This motivates the following problem.

LIST INTERSECTION
input $f, g : [N] \to [M]$ each is monotone increasing
output $i, j \in [N]$ with $i \neq j$ and $f(i) = f(j)$, or "lists disjoint" if the images of f and g are disjoint
complexity We present a bounded-error quantum algorithm which makes $O(\sqrt{N} c^{\log^* N})$ queries to f for some constant $c > 1$. A trivial lower bound $\Omega(\sqrt{N})$ can be obtained by a reduction from the OR-problem: given an OR-instance $x \in \{0,1\}^N$, define $f, g : [N] \to [2N+1]$ by $f(i) = 2i + 1$ and $g(i) = 2i + x_i$ for all $i \in [N]$. Then f and g are ordered, and $\mathrm{OR}(x) = 1$ iff the

LIST INTERSECTION problem has a solution. The same reduction shows that the classical bounded-error query complexity is $\Theta(N)$.

The function $\log^\star(N)$ is defined as the minimum number of iterated applications of the logarithm function necessary to obtain a number less than or equal to 1: $\log^\star(N) = \min\{i \geq 0 \mid \log^{(i)}(N) \leq 1\}$, where $\log^{(i)} = \log \circ \log^{(i-1)}$ denotes the ith iterated application of log, and $\log^{(0)}$ is the identity function. Even though $c^{\log^\star(N)}$ is exponential in $\log^\star(N)$, it is still very small in N, in particular $c^{\log^\star(N)} \in o(\log^{(i)}(N))$ for any constant $i \geq 1$.

To a function $f : [N] \to [M]$ we can associate a *collision graph* $G(V, E)$ with $V = [N]$ and $(i, j) \in E$ if $i \neq j$ and $f(i) = f(j)$. The *Element Distinctness* problem simply consists of finding an edge in G. An interesting problem is to ask whether G contains some fixed subgraph. A simple, yet non-trivial subgraph is the triangle, i.e. the complete graph on 3 vertices.

TRIANGLE FINDING

input the symmetric adjacency matrix $M : [n] \times [n] \to \{0, 1\}$ of a graph with m edges

output $u, v, w \in [N]$ such that $M(u, v) = M(v, w) = M(w, u) = 1$ or "failure" if the graph contains no triangle

complexity We present a bounded-error quantum algorithm which needs $O(n + \sqrt{nm})$ queries. A better algorithm has been found in 2003, with $O(n^{1.3})$ bounded-error query quantum complexity [19], while Yao [26] showed a lower bound of $\Omega(n^{2/3} \log^{1/6} n)$. Classically a simple reduction from the OR-problem shows that the bounded-error query complexity is $\Theta(n^2)$, even if $m = O(n)$.

2. Preliminaries. We assume the reader is familiar with the formalism of quantum computing, otherwise we refer to [22]. The quantum ingredient of our algorithms is *amplitude amplification* [7], which generalizes quantum search [15]. The essence of amplitude amplification can be summarized by the following theorem.

THEOREM 2.1 (Amplitude amplification). *There exists a quantum algorithm* ***QSearch*** *with the following property. Let $\mathcal{A}$ be any quantum algorithm that uses no measurements, and mapping $|0\rangle$ to a superposition $\sum_{x \in X} \alpha_x |x\rangle$, for some set X. Let $g : X \to \{0, 1\}$ be a function testing whether a basis state represents a* solution *or not. Let p be the success probability of $\mathcal{A}$, i.e. $p^2 = \sum_{x:g(x)=1} |\alpha_x|^2$. Let S_g be an operator implementing g s.t. $S_g|x\rangle = (-1)^{g(x)}|x\rangle$ for every $x \in X$. Then algorithm* ***QSearch*** *finds a solution using an expected number of $O(1/\sqrt{p})$ applications of $\mathcal{A}$, $\mathcal{A}^{-1}$ and S_g if $p > 0$, and otherwise runs forever.*

Note that when an algorithm $\mathcal{A}$ does make measurements during its computation then there is a standard trick which transforms it into an equivalent algorithm $\mathcal{A}'$ which does not. We replace every measurement with an operator writing the value, which would be the result of the measurement, in a new register, which initially was all zero. In the rest of the computation, every computation depending on the result of the measurement will depend rather on the content of this register.

QSearch works by iterating the unitary transformation $Q = -\mathcal{A}S_0\mathcal{A}^{-1}S_g$ a number of times, starting with initial state $\mathcal{A}|0\rangle$. The operator S_0 is defined as $S_0|0\rangle = -|0\rangle$ and $S_0|x\rangle = |x\rangle$ for all $x \neq 0$. The analysis of [7] shows that a measurement after $\Theta(1/\sqrt{p})$ iterations of Q yields a solution with probability close to 1. The algorithm **QSearch** does not need to know the value of p in advance, but if p is known, then a slight modification finds a solution *with certainty* using $O(1/\sqrt{p})$ applications of $\mathcal{A}$,

$\mathcal{A}^{-1}$ and S_g.

Grover's algorithm for searching a space of N items is a special case of amplitude amplification, where $\mathcal{A}$ is the Hadamard transform on each qubit. This $\mathcal{A}$ has probability $p \geq 1/N$ of finding a solution (if there is at least one), so amplitude amplification implies an $O(\sqrt{N})$ quantum algorithm for searching the space. We refer to this process as "quantum searching".

3. Element Distinctness.

Algorithm: Find a collision pair in $f : [N] \to [M]$

1. Partition the domain of f into disjoint sets $S_1, \ldots, S_{\sqrt{N}}$ of size $O(\sqrt{N})$ each.
2. Apply amplitude amplification to the following *inner block*
   (a) Select a random subset S_k of the partition.
   (b) Query all values $f(i)$ for $i \in S_k$, and build a binary search tree over the set $f(S_k) := \{f(i) : i \in S_k\}$. If S_k contains a collision pair, output it.
   (c) Otherwise search $j \in [N] \backslash S_k$ such that $f(j) \in f(S_k)$. Use the quantum search procedure which succeeds with probability at least $1/2$ provided S_k contains one element of a collision pair. In case of success, output the collision pair.

THEOREM 3.1. *If f has a collision pair i, j then the previous algorithm finds it after an expected number of $O(N^{3/4})$ queries to f.*

Proof. With probability at least $1/\sqrt{N}$, step 2a selects a subset containing i or j. Suppose this is the case. Then either the set contains a collision pair or it does not. If it does, then step 2b finds it, and if it does not, then with probability at least $1/2$, step 2c finds a collision pair. Therefore amplitude amplification will run $O(N^{1/4})$ expected number times the inner loop until success. Each of step 2b and step 2c use $O(\sqrt{N})$ queries, from which we conclude the claimed complexity. □

A weaker model is the comparison model, where we are only allowed to ask query $f(i) \leq f(j)$ for given indices i, j, rather than the actual values $f(i), f(j)$. The previous algorithm can be adapted to that model with the price of an $O(\log N)$ factor in steps 2b and 2c. In contrast, for classical (exact or bounded-error) algorithms, element distinctness is as hard as sorting and requires $\Theta(N \log N)$ comparisons.

4. List intersection. We are given two monotone increasing functions $f, g : [N] \to [M]$ and search for $i, j \in [N]$ such that $f(i) = g(j)$. A simple algorithm would be to make a quantum search for $i \in [N]$ such that there exists $j \in [N]$ with $f(i) = g(j)$. The quantum search of i will need $O(\sqrt{N})$ iterations and the binary search of j $O(\log N)$ queries. This gives a bounded-error quantum algorithm using $O(\sqrt{N} \log N)$ queries. We now show how to get rid of most of the log factor by exploiting the fact that both functions are monotone increasing.

Our quantum algorithm solves the problem using $O\left(\sqrt{N} c^{\log^*(N)}\right)$ comparisons for some constant $c > 0$. We define a set of subproblems such that the original problem (f, g) contains a collision pair if and only if at least one of the subproblems contains one. We then solve the original problem by running the subproblems in quantum parallel and applying amplitude amplification.

Let $1 \leq r < N$ be an integer. For the purpose of defining subproblems we extend the functions f and g to the domain $[1, N+r]$, mapping $f(N+i) = \max\{f(N), g(N)\} + i$ and $g(N+i) = f(N+i) + r$ for all $1 \leq i \leq r$, extending at the same time the range of f and g to $[M + 2r]$. We also define the *insertion point* of some integer $x < h(N+1)$ in a monotone increasing function $h : [N + r] \to [M + 2r]$ as the smallest index i such that $h(i) \geq x$.

We define $2\left\lceil \frac{N}{r} \right\rceil$ subproblems as follows. For each $0 \le i \le \lceil N/r \rceil - 1$, consider the subproblem (f_i, g_i') where f_i denotes the restriction of f to subdomain $[ir+1, (i+1)r]$, and g_i' the restriction of g to $[j, j+r-1]$ where j is the insertion point of $f(ir+1)$ in g.

Similarly, for each $0 \le j \le \lceil N/r \rceil - 1$, let be the subproblem (f_j', g_j) where g_j denotes the restriction of g to $[jr+1, (j+1)r]$, and f_j' the restriction of f to $[i, i+r-1]$ where i is the insertion point of $g(jr+1)$ in f.

LEMMA 4.1. *If $i, j \in [N]$ is a collision pair for (f, g) then it is also a collision pair for one of the subproblems.*

Proof. Let be $k = \lfloor i/r \rfloor + 1$ and k' be the insertion point of $f(k)$ in g. If $j \in [k', k'+r-1]$ then (i, j) is also a collision pair for the subproblem (f_k, g_k'). Otherwise let be $\ell = \lfloor j/r \rfloor + 1$. We have $f(k) \le g(\ell) \le f(i)$. Therefore the insertion point ℓ' of $g(\ell)$ in f satisfies $i \in [\ell', \ell'+r-1]$, from which we conclude that (i, j) is a collision pair for the subproblem (f_ℓ', g_ℓ). □

THEOREM 4.2. *There exists a quantum algorithm that outputs a collision pair between f and g with probability at least $\frac{2}{3}$ provided one exists, using $O\left(\sqrt{N} c^{\log^*(N)}\right)$ queries, for some constant $c > 1$.*

Proof. Let $T(N)$ denote the worst-case number of queries required if f and g have domain of size N. We show that

$$T(N) \le c' \sqrt{\frac{N}{r}} \left(\lceil \log(N+1) \rceil + T(r) \right), \tag{4.1}$$

for some (small) constant c'. Let $0 \le i \le \lceil N/r \rceil - 1$ and consider the subproblem (f_i, g_i'). To find the insertion point of $f(\lfloor i/r \rfloor + 1)$ in g we need $\lceil \log(N+1) \rceil$ queries by using binary search. Then we need additional $T(r)$ queries at most to find a collision pair for (f_i, g_i'). There are $2\left\lceil \frac{N}{r} \right\rceil$ subproblems, so by applying amplitude amplification we can find a collision pair among any one of them with probability at least $\frac{2}{3}$, provided there is one, using the number of queries claimed in equation (4.1).

We pick $r = \lceil \log^2(N) \rceil$. Since $T(r) \ge \Omega(\sqrt{r}) = \Omega(\log N)$, equation (4.1) implies

$$T(N) \le c'' \sqrt{\frac{N}{r}} T(r), \tag{4.2}$$

for some constant c''. Furthermore, our choice of r implies that the depth of the recursion defined by equation (4.2) is on the order of $\log^*(N)$, so unfolding the recursion gives the theorem. □

5. Triangle-finding. Finally we consider a related search problem. Consider an undirected graph $G = (V, E)$ on $|V| = n$ nodes with $|E| = m$ edges. There are $N = \binom{n}{2}$ edge slots in E, which we can query in a black box fashion (see also [10, Section 7]). The goal is now to find distinct vertices $a, b, c \in V$ such that (a, b), (a, c), $(b, c) \in E$. Since there are $\binom{n}{3}$ triples a, b, c, and we can decide whether a given triple is a triangle using 3 queries, we can use Grover's algorithm to find a triangle in $O(n^{3/2})$ queries. Below we give an algorithm which has the same complexity for dense graphs $m = O(n^2)$ but is more efficient for sparse graphs. In particular when $m = O(n)$, then the algorithm uses only $O(n)$ queries, while any classical bounded-error algorithm needs $\Omega(n^2)$ queries by a sensitivity argument for distinguishing the star graph, with the same graph augmented by a single edge.

Algorithm: Find a triangle

1. Use the bounded-error quantum counting procedure from [7, Theorem 18] to get a factor-2 estimation m' of the number of edges m, with $O(n)$ expected number of queries.
2. Apply amplitude amplification to the following *inner block*, interrupting if after $O(\sqrt{m'})$ calls to the inner block
   (a) Use quantum search to find an edge $(a,b) \in E$ among all $\binom{n}{2}$ potential edges, using at most $O(n/\sqrt{m'})$ queries.
   (b) Use quantum search to find a node $c \in V$ such that a,b,c is a triangle, using at most (n) queries.
3. Repeat until a triangle is found

Quantum search of an edge $(a,b) \in E$ succeeds after $O(n/\sqrt{m})$ expected number of queries. Since amplitude amplification forbids any observation in the inner block, we need step 2 to get an estimation of m, which determines the number of queries after which step 2a will be interrupted.

THEOREM 5.1. *If the graph contains a triangle, then the previous algorithm finds one after $O(n+\sqrt{nm})$ expected number of queries.*

Proof. Suppose step 2 finds the correct estimation of m. Suppose the graph contains a triangle. Let an edge be *golden* if it is part of a triangle. Then step 2a finds one with probability at least $1/2m$. Given this event step 2b finds a triangle with probability at least $1/2$. Therefore if amplitude amplification step succeeds with probability at least $1/2$.

Step 2 succeeds with probability at least $1/2$, so the total algorithm needs only a constant expected number of repetitions.

Each iteration costs $O(n+\sqrt{nm'})$ queries, where m' is the random outcome of step 2 with expectation m. This establishes the claimed complexity. □

6. Concluding remarks. An interesting related problem that is still wide open is the issue of *time-space tradeoffs* for element distinctness. Such tradeoffs have been studied for classical algorithms by Yao [25], Ajtai [2], Beame, Saks, Sun, and Vee [5], and others. In particular, Yao shows that the time-space product of any classical deterministic comparison-based branching program solving element distinctness satisfies $TS \geq \Omega(N^{2-\varepsilon(N)})$, where $\varepsilon(N) = 5/\sqrt{\ln N}$. An upper bound $TS = O((N \log N)^2)$ is achievable classically.

Ignoring logarithmic factors, the quantum algorithm presented here uses time $T = N^{3/4}$ and space $S = N^{1/2}$. An alternative quantum algorithm is to search the space of all $\binom{N}{2}$ (x,y)-pairs to try and find a collision. This algorithm has roughly $T = N$ and $S = \log N$. Thirdly, Ambainis's new algorithm has $T = N^{2/3}$ and $S = N^{2/3}$. All these algorithms satisfy $T^2S \approx N^2$. In fact, for every space bound S less than $N^{2/3}$, one can find an algorithm whose time (or query) complexity T satisfies $T^2S \approx N^2$. We conjecture that this close to optimal. Proving this would be very interesting, since no non-trivial quantum time-space tradeoff lower bounds are known for any decision problem (some tradeoffs for multiple-output problems may be found in [17]).

REFERENCES

[1] S. AARONSON, *Quantum lower bound for the collision problem*, in Proceedings of 34th ACM Symposium on Theory of Computing (STOC), pp. 635–642, 2002. quant-ph/0111102.

[2] M. Ajtai, *Determinism versus non-determinism for linear time RAMs*, in Proceedings of 31st ACM Symposium on Theory of Computing (STOC), pp. 632–641, 1999.

[3] A. Ambainis, *Quantum walk algorithm for element distinctness.* quant-ph/0311001, 1 Nov 2003.

[4] ———, *Quantum lower bounds for collision and element distinctness with small range.* quant-ph/0305179, 29 May 2003.

[5] P. Beame, M. Saks, X. Sun, and E. Vee, *Super-linear time-space tradeoff lower bounds for randomized computation*, in Proceedings of 41st IEEE Symposium on Foundations of Computer Science (FOCS), pp. 169–179, 2000. Available at ECCC TR00–025.

[6] G. Brassard and P. Høyer, *An exact quantum polynomial-time algorithm for Simon's problem*, in Proceedings of the 5th Israeli Symposium on Theory of Computing and Systems (ISTCS), pp. 12–23, 1997. quant-ph/9704027.

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

[8] G. Brassard, P. Høyer, and A. Tapp, *Quantum algorithm for the collision problem*, ACM SIGACT News (Cryptology Column), 28, pp. 14–19, 1997. quant-ph/9705002.

[9] H. Buhrman, R. Cleve, and A. Wigderson, *Quantum vs. classical communication and computation*, in Proceedings of 30th ACM Symposium on Theory of Computing (STOC), pp. 63–68, 1998. quant-ph/9802040.

[10] H. Buhrman, R. Cleve, R. de Wolf, and Ch. Zalka, *Bounds for small-error and zero-error quantum algorithms*, in Proceedings of 40th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 358–368, 1999. cs.CC/9904019.

[11] H. Buhrman, C. Dürr, M. Heiligman, P. Høyer, F. Magniez, M. Santha, and R. de Wolf, *Quantum algorithms for element distinctness*, in Proceedings of 16th IEEE Conference on Computational Complexity, pp. 131–137, 2001. quant-ph/0007016.

[12] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla, *Quantum query complexity of some graph problems*, in Proceedings of 31st ICALP, Lecture Notes in Computer Science, 2004.

[13] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, *Invariant quantum algorithms for insertion into an ordered list.* quant-ph/9901059, 19 Jan 1999.

[14] D. Grigoriev, *Randomized complexity lower bounds*, in Proceedings of 30th ACM Symposium on Theory of Computing (STOC), 1998, pp. 219–223.

[15] L. K. Grover, *A fast quantum mechanical algorithm for database search*, in Proceedings of 28th ACM Symposium on Theory of Computing (STOC), 1996, pp. 212–219. quant-ph/9605043.

[16] P. Høyer, J. Neerbek, and Y. Shi, *Quantum complexities of ordered searching, sorting, and element distinctness*, in Proceedings of 28th ICALP, vol. 2076 of Lecture Notes in Computer Science, Springer, pp. 346–357, 2001. quant-ph/0102078.

[17] H. Klauck, R. Špalek, and R. de Wolf, *Quantum and classical strong direct product theorems and optimal time-space tradeoffs.* quant-ph/0402123, 18 Feb 2004.

[18] A. Lubiw and A. Rácz, *A lower bound for the integer element distinctiveness problem*, Information and Control, 94, pp. 83–92, 1991.

[19] F. Magniez, M. Santha, and M. Szegedy, *An $O(n^{1.3})$ quantum algorithm for the triangle problem.* quant-ph/0310134, 21 Oct 2003.

[20] M. Mosca, *Quantum searching, counting and amplitude amplification by eigenvector analysis*, in MFCS workshop on Randomized Algorithms, pp. 90–100, 1998.

[21] A. Nayak and F. Wu, *The quantum query complexity of approximating the median and related statistics*, in Proceedings of 31st ACM Symposium on Theory of Computing (STOC), pp. 384–393, 1999. quant-ph/9804066.

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

[23] Y. Shi, *Quantum lower bounds for the collision and the element distinctness problems*, in Proceedings of 43rd IEEE Symposium on Foundations of Computer Science (FOCS), pp. 513–519, 2002. quant-ph/0112086.

[24] P. W. Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM Journal on Computing, 26, pp. 1484–1509, 1997. Earlier version in FOCS'94. quant-ph/9508027.

[25] A. C.-C. Yao, *Near-optimal time-space tradeoff for element distinctness*, SIAM Journal on Computing, 23, pp. 966–975, 1994. Earlier version in FOCS'88.

[26] ———, Personal communication, 2003.