Invariant Quantum Algorithms for Insertion into an Ordered List

Edward Farhi, Jeffrey Goldstone*
Center for Theoretical Physics
Massachusetts Institute of Technology
Cambridge, MA 02139

Sam Gutmann†
Department of Mathematics
Northeastern University
Boston, MA 02115

Michael Sipser‡
Department of Mathematics
Massachusetts Institute of Technology
Cambridge, MA 02139

MIT CTP # 2815

January 19, 1999

Abstract

We consider the problem of inserting one item into a list of $N-1$ ordered items. We previously showed that no quantum algorithm could solve this problem in fewer than $\log N/(2\log\log N)$ queries, for N large. We transform the problem into a "translationally invariant" problem and restrict attention to invariant algorithms. We construct the "greedy" invariant algorithm and show numerically that it outperforms the best classical algorithm for various N. We also find invariant algorithms that succeed exactly in fewer queries than is classically possible, and iterating one of them shows that the insertion problem can be solved in fewer than $0.53\log N$ quantum queries for large N (where $\log N$ is the classical lower bound). We don't know whether a $o(\log N)$ algorithm exists.

*farhi@mit.edu ; goldstone@mitlns.mit.edu
†sgutm@nuhub.neu.edu
‡sipser@math.mit.edu
This work was supported in part by The Department of Energy under cooperative agreement DE–FC02–94ER40818 and by the National Science Foundation under grant NSF 95–03322 CCR.

1 Introduction

We consider the problem of inserting a new item into an ordered list of $N-1$ items. A single classical query consists of comparing the new item with any chosen item on the list to see if the new item comes before or after the chosen item. Classically, the best algorithm for determining the point of insertion is binary search, which uses $\lceil \log_2 N \rceil$ queries. In [1] we showed that quantum mechanically, for large N, an algorithm that succeeds after k quantum queries must have

$$k > \frac{\log_2 N}{2 \log_2 \log_2 N} \ . \tag{1.1}$$

The same bound holds for algorithms that succeed with probability $\varepsilon > 0$ (independent of N).

In this paper we transform the insertion problem into an equivalent "translationally invariant" problem and restrict our attention to translationally invariant algorithms. In the next section we spell out what we mean by a translationally invariant algorithm. We derive a lower bound on the number of quantum queries needed for a successful translationally invariant algorithm. This bound turns out to coincide with (1.1), which suggests to us that the best algorithm may in fact be translationally invariant.

In Section 3 we construct the greedy translationally invariant algorithm for the insertion problem. By a "greedy" algorithm we mean an algorithm in which each step is chosen to maximize the probability of success after all preceding steps have been chosen. We present some numerical results for the greedy algorithm. For example, if $N = 2048$, after 5 quantum queries the probability of success is 0.9939 compared to the best possible classical probability of 1/64. However, we have not been able to analyze the large N behavior of the greedy algorithm.

The greedy algorithm can achieve a high probability of success but is not exact ("Exact" means that the correct answer is guaranteed.) In Section 4 we present a method for exploring whether an exact k-quantum-query translationally invariant algorithm exists for a given N. Using this method we find a 2-query algorithm for $N = 6$. A self-contained presentation of this algorithm is given at the end of Section 4. Furthermore, we find that no 2-query translationally invariant algorithm exists for $N \geq 7$. With 3 quantum queries we can construct a translationally invariant algorithm for $N = 52$ but we do not know how large a value of N can be attained with $k = 3$.

Starting with a k-quantum-query algorithm that exactly solves the insertion problem for some M, one can solve the insertion problem for $N = M^h$ with hk quantum queries for any positive integer h. To do this first pick out $M-1$ items, equally spaced in the list of $M^h - 1$ items. Running the k-quantum-query algorithm determines the point of insertion to lie in a range of M^{h-1} items. Iterate this procedure a total of h times to exactly determine the point of insertion in the original list of $M^h - 1$ items. Note that the overall algorithm with hk queries is not translationally invariant although the k-query subroutine is.

The result of the previous paragraph and our exact $N = 52$ in $k = 3$ algorithm (see Section 4) shows that one can construct a quantum algorithm for solving the insertion problem with $N - 1$ items where the number of queries grows like

$$\left(\frac{3}{\log_2 52}\right) \log_2 N \ . \tag{1.2}$$

Further exploration of the methods in Section 4 will certainly lead to a better constant than $3/\log_2 52$ and perhaps even an $o(\log N)$ algorithm.

Recently, Röhrig [2] published an algorithm that uses an average of $(3/4)\log_2 N + O(1)$ queries to solve the insert problem with probability $1/2$. This is not attainable classically, but iterating the algorithm to improve the $1/2$ probability involves more queries than are required to solve the insertion problem classically.

Our results carry over immediately to sorting. Classically, in the comparison model, n items can be sorted in $n \log_2 n + O(n)$ queries using binary-search insertion for each $N = 2, 3, \ldots, n$. Using our exact quantum insertion algorithm as a subroutine, the number of required queries can be cut by a constant factor, beating the classical lower bound of $n \log_2 n$.

2 Translationally Invariant Algorithms

The classical problem of inserting one item into an ordered list of $N - 1$ items is equivalent to the following oracular problem: Consider the N functions f_j defined on the set $\{0, 1, \ldots, N - 1\}$ by

$$f_j(x) = \begin{cases} -1 \ , & x < j \\ +1 \ , & x \geq j \end{cases} \tag{2.1}$$

for $j = 0, 1, \ldots, N - 1$. A query consists of giving the oracle a value of x with the oracle returning $f_j(x)$ for some fixed but unknown j. The problem is to determine j. (Note that $f_j(N - 1) = +1$ for all j, so querying the oracle at $x = N - 1$ is of no help. However, it is convenient for us to include this value of x.)

In order to construct our quantum algorithms we double the domain of the functions f_j and define

$$F_j(x) = \begin{cases} f_j(x) \ , & 0 \leq x \leq N - 1 \\ -f_j(x - N) \ , & N \leq x \leq 2N - 1 \ . \end{cases} \tag{2.2}$$

The problem is still to determine the value of j. Counting queries of F_j is equivalent to counting queries of f_j. Doubling the domain of the functions is of no help classically but is of use to us in the quantum setting. Note that $F_{j+1}(x) = F_j(x-1)$ for $j = 0, 1, \ldots, N-2$ if we make the identification that $x = -1$ is $x = 2N-1$. In this sense the F_j's are translates of each other.

We work in a Hilbert space of dimension $2N$ with basis vectors $|x\rangle$ with $x = 0, 1, \ldots, 2N-1$. A quantum query is an application of the unitary operator

$$\widehat{F}_j \,|x\rangle = F_j(x)\,|x\rangle \tag{2.3}$$

when the oracle holds the function F_j. (The workbits necessary for constructing (2.3) have been suppressed.) A k-query quantum algorithm starts in a state $|s\rangle$ and alternately applies $\widehat{F}_j$ and j-independent unitary operators V_ℓ to produce the state

$$V_k \widehat{F}_j V_{k-1} \cdots V_1 \widehat{F}_j \,|s\rangle \ . \tag{2.4}$$

(In our algorithms, all of the operators in (2.4) act as the identity in the suppressed work space.) An algorithm succeeds if the states in (2.4) are an orthogonal set for $j = 0, 1, \ldots, N-1$. Because the last unitary operator, V_k, is at our disposal we are free to choose the orthogonal states of a successful algorithm to be any orthogonal set. Corresponding to F_j, we choose

$$|j+\rangle = \frac{1}{\sqrt{2}}\big(|j\rangle + |j+N\rangle\big) \quad \text{for } j = 0, 1, \ldots, N-1 \tag{2.5}$$

to be the target state of a successful k-query algorithm for k even and

$$|j-\rangle = \frac{1}{\sqrt{2}}\big(|j\rangle - |j+N\rangle\big) \quad \text{for } j = 0, 1, \ldots, N-1 \tag{2.6}$$

to be the target state of a successful k-query algorithm for k odd. (We defer the explanation for the odd/even distinction until later.)

We now note that the $\widehat{F}_j$ are translates of each other in the following sense. Let the translation operator T be defined by

$$\begin{aligned} &T\,|x\rangle = |x+1\rangle \quad \text{for } x = 0, 1, \ldots, 2N-2 \\ &T\,|2N-1\rangle = |0\rangle \ . \end{aligned} \tag{2.7}$$

Then we have

$$T \widehat{F}_j T^{-1} = \widehat{F}_{j+1} \tag{2.8}$$

for $j = 0, 1, \ldots, N-2$ and equivalently

$$T^j \widehat{F}_0 T^{-j} = \widehat{F}_j \tag{2.9}$$

for $j = 1, 2, \ldots, N-1$. Furthermore

$$T^j \left|0\pm\right\rangle = \left|j\pm\right\rangle \tag{2.10}$$

for $j = 1, 2, \ldots, N-1$.

Suppose we pick the starting state of our algorithm to be

$$\left|s\right\rangle = \frac{1}{\sqrt{2N}} \sum_{x=0}^{2N-1} \left|x\right\rangle \tag{2.11}$$

which is translationally invariant, that is,

$$T\left|s\right\rangle = \left|s\right\rangle \ . \tag{2.12}$$

Furthermore, suppose we limit ourselves to translationally invariant unitary operators V_ℓ, that is, we require

$$TV_\ell T^{-1} = V_\ell \quad \text{for } \ell = 1, 2, \ldots, k. \tag{2.13}$$

Then if a k-query algorithm succeeds for $j = 0$, that is,

$$\begin{aligned} \left|0+\right\rangle &= V_k \widehat{F}_0 V_{k-1} \cdots V_1 \widehat{F}_0 \left|s\right\rangle \quad \text{when } k \text{ is even, or} \\ \left|0-\right\rangle &= V_k \widehat{F}_0 V_{k-1} \cdots V_1 \widehat{F}_0 \left|s\right\rangle \quad \text{when } k \text{ is odd} \end{aligned} \tag{2.14}$$

then because of (2.9), (2.10), (2.12), and (2.13) it follows that

$$\begin{aligned} \left|j+\right\rangle &= V_k \widehat{F}_j V_{k-1} \cdots V_1 \widehat{F}_j \left|s\right\rangle \quad \text{when } k \text{ is even, or} \\ \left|j-\right\rangle &= V_k \widehat{F}_j V_{k-1} \cdots V_1 \widehat{F}_j \left|s\right\rangle \quad \text{when } k \text{ is odd.} \end{aligned} \tag{2.15}$$

A clear advantage of this translationally invariant *ansatz* is that finding a set of V's which makes the single j-independent condition (2.14) hold guarantees that the algorithm succeeds for all j.

To understand which operators V are translationally invariant, that is, satisfy $TVT^{-1} = V$, we work in the momentum basis

$$\left|\mathbf{p}\right\rangle = \frac{1}{\sqrt{2N}} \sum_{x=0}^{2N-1} e^{i\mathbf{p}x\,\pi/N} \left|x\right\rangle \quad \text{for } \mathbf{p} = 0, 1, \ldots, 2N-1 \tag{2.16}$$

for which

$$|x\rangle = \frac{1}{\sqrt{2N}} \sum_{\mathbf{p}=0}^{2N-1} e^{-i\mathbf{p}x\,\pi/N} \, |\mathbf{p}\rangle \quad \text{for } x = 0, 1, \ldots, 2N-1. \tag{2.17}$$

Kets with boldfaced labels always denote momentum basis vectors. Note that

$$T\,|\mathbf{p}\rangle = e^{-i\mathbf{p}\,\pi/N}\,|\mathbf{p}\rangle \tag{2.18}$$

so we see that T is diagonal in the momentum basis. Thus if V_ℓ is diagonal in the momentum basis, that is,

$$V_\ell\,|\mathbf{p}\rangle = e^{i\alpha_\ell(\mathbf{p})}\,|\mathbf{p}\rangle \tag{2.19}$$

where $\alpha_\ell(\mathbf{p})$ is real, then each V_ℓ is both unitary and translationally invariant.

Constructing a successful k-query translationally invariant algorithm is equivalent to finding phases $\alpha_\ell(\mathbf{p})$, for $\ell = 1, 2, \ldots, k$ to make (2.14) hold. Because of (1.1), for a given N, we know that this cannot be done if k is too small. Strategies for choosing the phases $\alpha_\ell(\mathbf{p})$ for the greedy algorithm and for exactly successful algorithms are the subjects of the next two sections.

Because the translationally invariant *ansatz* has led to the momentum basis, all the elements of (2.14) are best expressed in the momentum basis. The V_ℓ's are defined in the momentum basis by (2.19). By (2.11) and (2.16) we have

$$|s\rangle = |\mathbf{0}\rangle \ . \tag{2.20}$$

(Recall that the boldface $\mathbf{0}$ denotes the momentum basis vector with $\mathbf{p} = \mathbf{0}$.) By (2.5), (2.6), and (2.17) we have

$$|0+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |N\rangle) = \frac{1}{\sqrt{N}} \sum_{\mathbf{p}\text{ even}} |\mathbf{p}\rangle \tag{2.21}$$

and

$$|0-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |N\rangle) = \frac{1}{\sqrt{N}} \sum_{\mathbf{p}\text{ odd}} |\mathbf{p}\rangle \ .$$

(The nonboldfaced kets $|0\rangle$ and $|N\rangle$ are in the $|x\rangle$ basis.) We also need the matrix elements of $\widehat{F}_0$ in the momentum basis,

$$\begin{aligned} \langle \mathbf{p}| \, \widehat{F}_0 \, |\mathbf{q}\rangle &= \sum_{x=0}^{2N-1} \langle \mathbf{p}| \, \widehat{F}_0 \, |x\rangle \, \langle x \,|\, \mathbf{q}\rangle \\ &= \sum_{x=0}^{2N-1} \langle \mathbf{p} \,|\, x\rangle \, F_0(x) \, \langle x \,|\, \mathbf{q}\rangle \\ &= \frac{1}{2N}\Big(\sum_{x=0}^{N-1} - \sum_{x=N}^{2N-1}\Big) e^{i\pi(\mathbf{q}-\mathbf{p})\,x/N} \ . \end{aligned} \tag{2.22}$$

So

$$\langle \mathbf{p}| \, \widehat{F}_0 \, |\mathbf{q}\rangle = \begin{cases} \dfrac{ie^{-i\pi(\mathbf{q}-\mathbf{p})/2N}}{N \sin \pi(\mathbf{q}-\mathbf{p})/2N} \ , & \mathbf{q}-\mathbf{p} \text{ odd} \\ 0 \ , & \mathbf{q}-\mathbf{p} \text{ even}. \end{cases} \tag{2.23}$$

After ℓ queries, a translationally invariant algorithm produces the state

$$|\psi_\ell\rangle = V_\ell \widehat{F}_0 V_{\ell-1} \cdots V_1 \widehat{F}_0 \, |\mathbf{0}\rangle \ . \tag{2.24}$$

Here and throughout, k is the fixed total number of queries, and ℓ, with $1 \le \ell \le k$, indexes a stage of the algorithm. Expressed in the momentum basis, for ℓ even, using (2.19) we have

$$\begin{aligned} |\psi_\ell\rangle = \sum_{\mathbf{p}_\ell \text{ even}} \ \sum_{\mathbf{p}_{\ell-1} \text{ odd}} \cdots \sum_{\mathbf{p}_1 \text{ odd}} & \\ |\mathbf{p}_\ell\rangle \, e^{i\alpha_\ell(\mathbf{p}_\ell)} \, \langle \mathbf{p}_\ell| \, \widehat{F}_0 \, |\mathbf{p}_{\ell-1}\rangle \, e^{i\alpha_{\ell-1}(\mathbf{p}_{\ell-1})} & \ldots e^{i\alpha_1(\mathbf{p}_1)} \, \langle \mathbf{p}_1| \, \widehat{F}_0 \, |\mathbf{0}\rangle \end{aligned} \tag{2.25}$$

where we need only include $\mathbf{p}_1$ odd, $\mathbf{p}_2$ even, etc. because of (2.23). This means that at each stage there are N, not $2N$, phases to choose.

The goal of an algorithm is to produce the state $|0+\rangle$ after an even number of queries (or $|0-\rangle$ after an odd number). We can judge how close to success we are at the ℓ-th stage by evaluating the overlap with $|0+\rangle$ (for ℓ even),

$$\langle 0+ \,|\, \psi_\ell\rangle = \frac{1}{\sqrt{N}} \sum_{\mathbf{p} \text{ even}} \langle \mathbf{p} \,|\, \psi_\ell\rangle \tag{2.26}$$

by (2.21). For these translationally invariant algorithms, the probability of success if we stop at the k-th stage is the same whichever F_j the oracle holds and equals

$\left|\langle 0+\,|\,\psi_k\rangle\right|^2$. To find a lower bound on the number of queries required for success we note, using (2.25) and (2.26) that

$$\left|\langle 0+\,|\,\psi_\ell\rangle\right| \le \frac{1}{\sqrt{N}} \sum_{\mathbf{p}_\ell \text{ even}} \sum_{\mathbf{p}_{\ell-1} \text{ odd}} \cdots \sum_{\mathbf{p}_1 \text{ odd}} \left|\langle \mathbf{p}_\ell|\, \widehat{F}_0\, |\mathbf{p}_{\ell-1}\rangle \langle \mathbf{p}_{\ell-1}|\, \widehat{F}_0\, |\mathbf{p}_{\ell-2}\rangle \cdots \langle \mathbf{p}_1|\, \widehat{F}_0\, |\mathbf{0}\rangle\right| . \tag{2.27}$$

Because $\langle \mathbf{p}|\, \widehat{F}_0\, |\mathbf{q}\rangle$ only depends on $(\mathbf{p}-\mathbf{q}) \bmod 2N$ the righthand side of (2.27) consists of ℓ identical factors and we have

$$\left|\langle 0+\,|\,\psi_\ell\rangle\right| \le \frac{1}{\sqrt{N}} \Big[\sum_{\mathbf{p} \text{ odd}} \left|\langle \mathbf{p}|\, \widehat{F}_0\, |\mathbf{0}\rangle\right|\Big]^\ell . \tag{2.28}$$

By (2.23) we then have

$$\left|\langle 0+\,|\,\psi_\ell\rangle\right| \le \frac{1}{\sqrt{N}} \Big[\frac{1}{N} \sum_{\mathbf{p} \text{ odd}} \frac{1}{\sin \pi\mathbf{p}/2N}\Big]^\ell . \tag{2.29}$$

(Recall that $0 \le \mathbf{p} \le 2N-1$.) Approximating the sum, we have

$$\begin{aligned} \frac{1}{N} \sum_{\mathbf{p} \text{ odd}} \frac{1}{\sin \pi\mathbf{p}/2N} &= \frac{4}{\pi}\Big(1 + \frac{1}{3} + \cdots + \frac{1}{N-1}\Big) \\ &\quad + \frac{2}{\pi}\int_0^{\pi/2} \mathrm{d}\theta \Big(\frac{1}{\sin\theta} - \frac{1}{\theta}\Big) + O(1/N) \\ &= \frac{2}{\pi}\Big(\ln N + \gamma + \ln\frac{8}{\pi}\Big) + O(1/N) . \end{aligned} \tag{2.30}$$

(The approximation (2.30) is already correct at $N = 3$ to 1 part in 1000.) A k-query algorithm that succeeds with probability ε must have

$$\varepsilon \le \left|\langle 0+\,|\,\psi_k\rangle\right|^2 \le \frac{1}{N}\Big[\frac{2}{\pi}\ln N + O(1)\Big]^{2k} \tag{2.31}$$

which implies, for N large, that

$$k > \frac{\ln N}{2\ln\ln N} . \tag{2.32}$$

Note that the ratio of the righthand sides of the bounds (1.1) and (2.32) converges to 1 as $N \to \infty$. Since the bound (2.32) was derived under the assumption of translation invariance, it is only a special case of the fully general bound (1.1).

The idea of invariance makes sense in other computing problems. For example, Grover's search problem [3] is invariant under the group of permutations. Requiring an algorithm in this problem to be permutation invariant is extremely restrictive. There is only one phase to choose for each V, and it is easy to see that the choice of -1 at each stage, which corresponds to Grover's algorithm, is optimal.

3 The Greedy Algorithm

The state produced after ℓ queries of a translationally invariant algorithm, given in (2.24), can be related to the state produced after $\ell - 1$ queries by

$$|\psi_\ell\rangle = V_\ell \widehat{F}_0 \,|\psi_{\ell-1}\rangle \tag{3.1}$$

where $|\psi_0\rangle = |\mathbf{0}\rangle$. We define the greedy algorithm inductively. Given $|\psi_{\ell-1}\rangle$ we choose V_ℓ to maximize the overlap of $|\psi_\ell\rangle$ with $|0+\rangle$ if ℓ is even or with $|0-\rangle$ if ℓ is odd. At each stage the overlap increases and hence the probability of success if we stop at the kth stage increases with k. As we will see below, the greedy algorithm is never perfect, but we provide numerical evidence that it converges rapidly. For selected values of N up to 4096 we see that the greedy algorithm outperforms the best classical algorithm.

We begin by showing how well the greedy algorithm does with one query. In this case

$$\begin{aligned} |\psi_1\rangle &= V_1 \widehat{F}_0 \,|\mathbf{0}\rangle \\ &= \sum_{\mathbf{p}\ \mathrm{odd}} |\mathbf{p}\rangle\, e^{i\alpha_1(\mathbf{p})} \langle \mathbf{p}|\, \widehat{F}_0 \,|\mathbf{0}\rangle \end{aligned} \tag{3.2}$$

where we have inserted a complete set, used (2.19) and used the fact that $\langle \mathbf{p}|\, \widehat{F}_0 \,|\mathbf{0}\rangle$ vanishes for $\mathbf{p}$ even. By (2.21)

$$\Big|\langle 0- \,|\, \psi_1\rangle\Big| = \frac{1}{\sqrt{N}} \Big| \sum_{\mathbf{p}\ \mathrm{odd}} e^{i\alpha_1(\mathbf{p})} \langle \mathbf{p}|\, \widehat{F}_0 \,|\mathbf{0}\rangle \Big|\ . \tag{3.3}$$

Choosing V_1 is equivalent to choosing the phases $\alpha_1(\mathbf{p})$. To maximize (3.3) we choose $\alpha_1(\mathbf{p})$ to make each term in the sum real and positive. With this choice

$$\Big|\langle 0- \,|\, \psi_1\rangle\Big| = \frac{1}{\sqrt{N}} \sum_{\mathbf{p}\ \mathrm{odd}} \Big| \langle \mathbf{p}|\, \widehat{F}_0 \,|\mathbf{0}\rangle \Big|\ . \tag{3.4}$$

Using (2.23) we have

$$\Big|\langle 0- \,|\, \psi_1\rangle\Big| = \frac{1}{N^{3/2}} \sum_{\mathbf{p}\ \mathrm{odd}} \frac{1}{\sin(\pi \mathbf{p}/2N)}\ . \tag{3.5}$$

Approximating the sum, for N large, as we did in (2.30), gives

$$\Big|\langle 0- \,|\, \psi_1\rangle\Big|^2 \sim \frac{4}{\pi^2 N} \Big[\ln N + \gamma + \ln \frac{8}{\pi}\Big]^2 \tag{3.6}$$

which is the probability of success after running a 1-query greedy algorithm. This beats the classically best possible, which is $2/N$.

To see how the greedy algorithm works at the ℓ-th stage (ℓ even, for example) first note that by (2.21)

$$\begin{aligned} \Big|\langle 0+\,|\,\psi_\ell\rangle\Big| &= \frac{1}{\sqrt{N}}\Big|\sum_{\mathbf{p}\text{ even}}\langle \mathbf{p}\,|\,\psi_\ell\rangle\Big| \\ &= \frac{1}{\sqrt{N}}\Big|\sum_{\mathbf{p}\text{ even}} e^{i\alpha_\ell(\mathbf{p})}\,\langle \mathbf{p}|\,\widehat{F}_0\,|\psi_{\ell-1}\rangle\Big| \end{aligned} \tag{3.7}$$

by (3.1) and (2.19). To maximize (3.7) we choose the phases $\alpha_\ell(\mathbf{p})$ to make each term in the sum real and nonnegative, that is, each $\langle \mathbf{p}\,|\,\psi_\ell\rangle$ is real and nonnegative. Now

$$\langle \mathbf{p}\,|\,\psi_\ell\rangle = e^{i\alpha_\ell(\mathbf{p})}\sum_{\mathbf{q}\text{ odd}}\langle \mathbf{p}|\,\widehat{F}_0\,|\mathbf{q}\rangle\,\langle \mathbf{q}\,|\,\psi_{\ell-1}\rangle \tag{3.8}$$

and by the choice of $\alpha_\ell(\mathbf{p})$ and (2.23)

$$\begin{aligned} \langle \mathbf{p}\,|\,\psi_\ell\rangle &= \frac{1}{N}\Big|\sum_{\mathbf{q}\text{ odd}}\frac{e^{i\pi(\mathbf{p}-\mathbf{q})/2N}}{\sin\pi(\mathbf{p}-\mathbf{q})/2N}\,\langle \mathbf{q}\,|\,\psi_{\ell-1}\rangle\Big| \\ &= \frac{1}{N}\Big|\sum_{\mathbf{q}\text{ odd}}\Big(\cot\big(\pi(\mathbf{p}-\mathbf{q})/2N\big)+i\Big)\,\langle \mathbf{q}\,|\,\psi_{\ell-1}\rangle\Big|\,. \end{aligned} \tag{3.9}$$

This last formula, together with its virtually identical ℓ-odd analogue, explicitly determines $|\psi_\ell\rangle$ from $|\psi_{\ell-1}\rangle$, providing a complete description of the greedy algorithm. The choice of k, the number of queries before stopping and measuring, depends on the probability of success desired.

The probability of success after ℓ queries is

$$\mathrm{Prob}(\ell) = \Big|\langle 0+\,|\,\psi_\ell\rangle\Big|^2 = \frac{1}{N}\Big|\sum_{\mathbf{p}\text{ even}}\langle \mathbf{p}\,|\,\psi_\ell\rangle\Big|^2\,. \tag{3.10}$$

Now we can rewrite (3.9) as

$$\langle \mathbf{p}\,|\,\psi_\ell\rangle = \frac{1}{N}\Big|\sum_{\mathbf{q}\text{ odd}}\cot\big(\pi(\mathbf{p}-\mathbf{q})/2N\big)\,\langle \mathbf{q}\,|\,\psi_{\ell-1}\rangle + i\sqrt{N}\,\mathrm{Prob}^{\frac{1}{2}}(\ell-1)\Big| \tag{3.11}$$

and accordingly

$$\mathrm{Prob}(\ell) = \left[\frac{1}{N}\sum_{\mathbf{p}\text{ even}}\left\{\mathrm{Prob}(\ell-1)+\frac{1}{N}\Big(\sum_{\mathbf{q}\text{ odd}}\cot\frac{\pi(\mathbf{p}-\mathbf{q})}{2N}\,\langle \mathbf{q}\,|\,\psi_{\ell-1}\rangle\Big)^2\right\}^{\frac{1}{2}}\right]^2. \tag{3.12}$$

This formula shows that $\mathrm{Prob}(\ell) \geq \mathrm{Prob}(\ell-1)$. Furthermore, if $\langle \mathbf{q} \,|\, \psi_{\ell-1} \rangle = 1/\sqrt{N}$ for all $\mathbf{q}$ odd (which is equivalent to $\mathrm{Prob}(\ell-1) = 1$) then by (3.11) we have $\langle \mathbf{p} \,|\, \psi_\ell \rangle = 1/\sqrt{N}$ for all $\mathbf{p}$ even and $\mathrm{Prob}(\ell) = 1$. The greedy algorithm tends toward this fixed point.

Table 1: Probability of success of the greedy algorithm, stopping after k quantum queries.

N	$k=1$	2	3	4	5	6
64	.2036	.6495	.9615	.9997	1.000	1.000
256	.0788	.3886	.8221	.9907	.9999	1.000
1024	.0282	.2000	.5981	.9324	.9983	1.000
2048	.0165	.1374	.4818	.8690	.9939	.9997
4096	.0096	.0922	.3755	.7834	.9819	.9992

Numbers are given to 4 significant figures, so the 1.000's do not mean exact performance.

We have some numerical results for the greedy algorithm, which are presented in Table 1. For these calculations we also need the formulas analogous to (3.9) and (3.10) for ℓ odd. Starting in the state $|\psi_0\rangle = |\mathbf{0}\rangle$ it is then straightforward to calculate $\langle \mathbf{p} \,|\, \psi_\ell \rangle$ and the associated probability of success. Clearly the greedy quantum algorithm does much better than the best classical algorithm, which has a probability of success of $2^k/N$.

4 Exact Algorithms

An exactly successful k-query algorithm is a choice of $V_1, V_2, \ldots, V_k$ for which (2.14) holds. In this section, we recast this condition in a form that allows us to determine, in certain cases, if such a choice of V's exists.

For any k-query algorithm, successful or not, we define, as before,

$$|\psi_0\rangle = |\mathbf{0}\rangle \tag{4.1}$$

$$|\psi_\ell\rangle = V_\ell \widehat{F}_0 V_{\ell-1} \cdots V_1 \widehat{F}_0 \,|\mathbf{0}\rangle \tag{4.2}$$

where $1 \leq \ell \leq k$. The form (2.19) for each V_ℓ implies by (4.2)

$$\Big| \langle \mathbf{p} \,|\, \psi_\ell \rangle \Big| = \Big| \langle \mathbf{p} |\, \widehat{F}_0 \,|\psi_{\ell-1}\rangle \Big| \,. \tag{4.3}$$

Conversely, given any sequence $|\psi_0\rangle, |\psi_1\rangle, \ldots, |\psi_k\rangle$ satisfying (4.1) and (4.3) there is a sequence $V_1, V_2, \ldots, V_k$ of the form (2.19) such that (4.2) holds with

$$e^{i\alpha_\ell(\mathbf{p})} = \begin{cases} \dfrac{\langle \mathbf{p} \,|\, \psi_\ell\rangle}{\langle \mathbf{p}|\, \widehat{F}_0 \,|\psi_{\ell-1}\rangle} \;, & \mathbf{p}+\ell \text{ even} \\ \quad 1 \;, & \mathbf{p}+\ell \text{ odd} \end{cases} \tag{4.4}$$

where the choice of 1 for $\mathbf{p}+\ell$ odd is arbitrary.

If $|\psi_\ell\rangle$ satisfy (4.2) and (4.1) [or equivalently (4.3) and (4.1)] then as before $|\psi_\ell\rangle$ is a superposition of momentum basis states with $\mathbf{p}$ even for ℓ even and $\mathbf{p}$ odd for ℓ odd. The corresponding statement in the x basis is

$$\langle x+N \,|\, \psi_\ell\rangle = (-1)^\ell \langle x \,|\, \psi_\ell\rangle \;. \tag{4.5}$$

Using (4.5) and (2.16) we have

$$\langle \mathbf{p} \,|\, \psi_\ell\rangle = \begin{cases} \sqrt{\dfrac{2}{N}} \displaystyle\sum_{x=0}^{N-1} \langle x \,|\, \psi_\ell\rangle \, e^{-i\mathbf{p}x\pi/N} \;, & \mathbf{p}+\ell \text{ even} \\ \quad 0 \;, & \mathbf{p}+\ell \text{ odd} \end{cases} \tag{4.6}$$

and with (2.2) and (2.3) we have

$$\langle \mathbf{p}|\, \widehat{F}_0 \,|\psi_{\ell-1}\rangle = \begin{cases} \sqrt{\dfrac{2}{N}} \displaystyle\sum_{x=0}^{N-1} \langle x \,|\, \psi_{\ell-1}\rangle \, e^{-i\mathbf{p}x\pi/N} \;, & \mathbf{p}+\ell \text{ even} \\ \quad 0 \;, & \mathbf{p}+\ell \text{ odd.} \end{cases} \tag{4.7}$$

Thus (4.3) can be reformulated as

$$\left| \sum_{x=0}^{N-1} \langle x \,|\, \psi_\ell\rangle \, z^{-x} \right| = \left| \sum_{x=0}^{N-1} \langle x \,|\, \psi_{\ell-1}\rangle \, z^{-x} \right| \quad \text{at } z^N = (-1)^\ell. \tag{4.8}$$

Define polynomials of degree $N-1$ in the complex variable z by

$$P_\ell(z) = \sqrt{2} \sum_{x=0}^{N-1} \langle x \,|\, \psi_\ell\rangle \, z^{N-1-x} \;. \tag{4.9}$$

In terms of polynomials (4.9) the condition (4.8) is

$$\Big| P_\ell(z) \Big| = \Big| P_{\ell-1}(z) \Big| \quad \text{at } z^N = (-1)^\ell. \tag{4.10}$$

and

$$P_0(z) = \sqrt{2} \sum_{x=0}^{N-1} \langle x \,|\, \mathbf{0} \rangle \, z^{N-1-x} \qquad (4.11)$$
$$= \frac{1}{\sqrt{N}} (z^{N-1} + z^{N-2} + \cdots + 1)$$

Any sequence of degree $N - 1$ polynomials, $P_0, P_1, P_2, \ldots, P_k$ satisfying (4.10) and (4.11) corresponds to a k-query algorithm. For a k-query algorithm to be exactly successful we require, by (2.14), that

$$|\psi_k\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |N\rangle) \quad \text{when } k \text{ is even} \qquad (4.12)$$

or

$$|\psi_k\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |N\rangle) \quad \text{when } k \text{ is odd}$$

or equivalently

$$P_k(z) = z^{N-1} \; . \qquad (4.13)$$

To summarize, an exactly successful k-query translationally invariant algorithm exists if and only if a sequence of degree $N-1$ polynomials $P_0, P_1, \ldots, P_k$ exists that satisfies (4.10), with P_0 given by (4.11) and P_k given by (4.13).

Now define Q_ℓ by

$$Q_\ell(z) = P_\ell(z) \left[P_\ell\Big(\frac{1}{z^*}\Big) \right]^* \qquad (4.14)$$

and its coefficients $q_{\ell r}$ by

$$Q_\ell(z) = \sum_{r=-(N-1)}^{N-1} q_{\ell r} z^r \; . \qquad (4.15)$$

Note that

$$Q_\ell(z) = \left[Q_\ell\Big(\frac{1}{z^*}\Big) \right]^* \; , \text{ that is, } q_{\ell r} = q^*_{\ell,-r} \qquad (4.16)$$

and

$$Q_\ell(z) \geq 0 \qquad \text{on } |z| = 1. \qquad (4.17)$$

Now (4.10) is the same as

$$Q_\ell(z) = Q_{\ell-1}(z) \qquad \text{at } z^N = (-1)^\ell \qquad (4.18)$$

and (4.11) gives

$$Q_0(z) = \frac{1}{N}\left[z^{N-1} + 2z^{N-2} + \cdots + (N-1)z + N + (N-1)z^{-1} + \cdots + z^{1-N}\right] \quad (4.19)$$

and (4.13) gives

$$Q_k(z) = 1 \ . \quad (4.20)$$

One of the reasons we have introduced the Q's is that the condition (4.18) will turn out to be more tractable than (4.10).

Given a sequence $Q_0, Q_1, \ldots, Q_k$ defined by (4.14) where the P_ℓ's satisfy (4.10), (4.11), and(4.13) it is immediate that (4.16)–(4.20) are satisfied. We now establish the converse: given a sequence $Q_0, Q_1, \ldots, Q_k$ of the form (4.15) satisfying (4.16)–(4.20), each Q_ℓ can be factored as $P_\ell(z)[P_\ell(1/z^*)]^*$ where the polynomials P_ℓ satisfy (4.10), (4.11), and(4.13).

The key ingredient in establishing this converse is to prove that any $Q(z)$ of the form

$$Q(z) = \sum_{r=-M}^{M} q_r z^r \ , \quad q_M \neq 0 \quad (4.21)$$

satisfying

$$Q(z) = \left[Q\left(\frac{1}{z^*}\right)\right]^* \ , \text{ that is, } q_r = q_{-r}^* \quad (4.22)$$

and

$$Q(z) \geq 0 \quad \text{on } |z| = 1 \quad (4.23)$$

can be factored as

$$Q(z) = P(z)\left[P\left(\frac{1}{z^*}\right)\right]^* \quad (4.24)$$

for some polynomial P of degree M.

Proof: $z^M Q(z)$ is a polynomial of degree $2M$. Because of (4.22) its zeros occur in pairs, $z = ae^{i\alpha}$ and $z = \frac{1}{a}e^{i\alpha}$ (a real and positive). The only exception might be a zero on $|z| = 1$ but (4.23) implies that such zeros have *even* multiplicity. Thus we can factor

$$z^M Q(z) = C \prod_{t=1}^{M} (z - a_t e^{i\alpha_t})\left(z - \frac{1}{a_t}e^{i\alpha_t}\right) \quad (4.25)$$

or

$$Q(z) = D \prod_{t=1}^{M} (z - a_t e^{i\alpha_t}) \Big(\frac{1}{z} - a_t e^{-i\alpha_t} \Big) \ . \tag{4.26}$$

Now (4.22) shows that $D = D^*$ and (4.23) shows that $D > 0$. Now take

$$P(z) = \sqrt{D} \prod_{t=1}^{M} (z - a_t e^{i\alpha_t}) \tag{4.27}$$

establishing (4.24).

Having established (4.24) for each Q_ℓ obeying (4.16) and (4.17) it then follows immediately that the corresponding P_ℓ's obey (4.10) if the Q_ℓ's obey (4.18). We have thus shown that the existence of a sequence $Q_0, Q_1, \ldots, Q_k$ satisfying (4.15)–(4.20) is equivalent to the existence of an exactly successful k-query translationally invariant algorithm. Our goal is now to try to determine for which values of N and k such a sequence exists.

Formula (4.16) implies that

$$Q_\ell(e^{i\theta}) = \sum_{r=0}^{N-1} C_{\ell r} \cos r\theta + \sum_{r=1}^{N-1} S_{\ell r} \sin r\theta \tag{4.28}$$

where $C_{\ell r}$ and $S_{\ell r}$ are real. The matching condition (4.18), $Q_\ell(z) = Q_{\ell-1}(z)$ at $z^N = (-1)^\ell$, in terms of (4.28) is

$$\sum_{r=0}^{N-1} (C_{\ell r} - C_{\ell-1,r}) \cos\Big(r \frac{2\pi m}{N}\Big) + \sum_{r=1}^{N-1} (S_{\ell r} - S_{\ell-1,r}) \sin\Big(r \frac{2\pi m}{N}\Big) = 0 \tag{4.29}$$

for $m = 0, 1, \ldots, N-1$ in the case that ℓ is even. By taking the sum and difference of (4.29) with m and m replaced by $N - m$ we see that the matching conditions on the $C_{\ell r}$ decouple from the matching conditions on the $S_{\ell r}$ (and similarly for ℓ odd). Furthermore, $Q_\ell(e^{i\theta}) \geq 0$ along with $Q_\ell(e^{-i\theta}) \geq 0$, which follow from (4.17), combine to give

$$\sum_{r=0}^{N-1} C_{\ell r} \cos r\theta \geq \Big| \sum_{r=1}^{N-1} S_{\ell r} \sin r\theta \Big| \ . \tag{4.30}$$

By (4.19) and (4.20) there are no $\sin r\theta$ terms in Q_0 and Q_k, that is, $S_{0r} = S_{kr} = 0$. Thus we see that without loss of generality we can set all $S_{\ell r} = 0$ in (4.28) while attempting to determine if a sequence $Q_0, Q_1, \ldots, Q_k$ obeying (4.15)–(4.20) exists.

Now by (4.9) and (4.14),

$$Q_\ell(z) = 2 \sum_{x,y=0}^{N-1} \langle \psi_\ell \,|\, y\rangle \, \langle x \,|\, \psi_\ell\rangle \, z^{y-x} \tag{4.31}$$

and we see that the z^0 term is $2\sum_{x=0}^{N-1} \big|\langle x \,|\, \psi_\ell\rangle\big|^2$, which is 1 by (4.5). This implies that $C_{\ell 0}$ in (4.28) is 1. Now decompose (4.28) as

$$Q_\ell(e^{i\theta}) = 1 + A_\ell(\theta) + B_\ell(\theta) \tag{4.32}$$

where

$$A_\ell(\theta) = \sum_{r=1}^{N-1} a_{\ell r} \cos r\theta \ , \quad a_{\ell r} = a_{\ell, N-r} \tag{4.33}$$

$$B_\ell(\theta) = \sum_{r=1}^{N-1} b_{\ell r} \cos r\theta \ , \quad b_{\ell r} = -b_{\ell, N-r} \ . \tag{4.34}$$

Because $A_\ell(\theta) = 0$ when $e^{iN\theta} = -1$ and $B_\ell(\theta) = 0$ when $e^{iN\theta} = 1$, the matching conditions (4.18) become

$$\begin{aligned} B_1 &= B_0 \\ A_2 &= A_1 \\ B_3 &= B_2 \\ A_4 &= A_3 \\ &\ \vdots \end{aligned} \tag{4.35}$$

From (4.19) we have

$$Q_0 = 1 + \frac{2}{N}\big[(N-1)\cos\theta + (N-2)\cos 2\theta + \cdots + \cos(N-1)\theta\big] \tag{4.36}$$

and by (4.32)–(4.34)

$$A_0 = \cos\theta + \cos 2\theta + \cdots + \cos(N-1)\theta \tag{4.37}$$

and B_0 is given as

$$B_0 = \Big(1 - \frac{2}{N}\Big)\cos\theta + \Big(1 - \frac{4}{N}\Big)\cos 2\theta + \cdots - \Big(1 - \frac{2}{N}\Big)\cos(N-1)\theta \ . \tag{4.38}$$

By (4.20)

$$A_k = B_k = 0\ . \tag{4.39}$$

Finally, we can state the equivalence that we actually use. The existence of an exactly successful k-query translationally invariant algorithm is equivalent to the existence of a sequence of functions $A_0, B_0, A_1, B_1, \ldots, A_k, B_k$ of the form (4.33) and (4.34) with A_0, B_0 given by (4.37), (4.38) and A_k, B_k given by (4.39) with the matching conditions (4.35) and the positivity condition

$$1 + A_\ell(\theta) + B_\ell(\theta) \geq 0 \tag{4.40}$$

for $0 \leq \theta \leq \pi$ and $0 \leq \ell \leq k$.

We now apply the machinery developed above to the 2-query case to see for which N an exactly successful translationally invariant algorithm exists. The 2-query algorithm corresponds to the sequence $A_0, B_0, A_1, B_1, A_2, B_2$, with $A_2 = B_2 = 0$ by (4.39), A_0 and B_0 given by (4.37) and (4.38), and $B_1 = B_0$ and $A_1 = A_2 = 0$ by the matching conditions (4.35). The condition (4.40) for $\ell = 1$ becomes

$$1 + B_0(\theta) \geq 0 \tag{4.41}$$

with B_0 given by (4.38). Numerical examination of (4.41) shows that this inequality holds for $N \leq 6$ and we have shown that it does not for $N \geq 7$. Later in this section we will explicitly show the $k = 2$, $N = 6$ algorithm.

The $k = 2$ case was particularly straightforward because the matching conditions left no freedom to choose the A's and B's. For $k = 3$ the matching conditions leave a single undetermined function A_1. The two constraints that must be satisfied are (4.40) for $\ell = 1$ and $\ell = 2$, that is

$$1 + A_1(\theta) + B_0(\theta) \geq 0 \tag{4.42}$$

and

$$1 + A_1(\theta) \geq 0$$

where B_0 is given by (4.38) and A_1 is of the form (4.33). A 3-query translationally invariant algorithm exists for a given N if and only if such an A_1 can be found. By (4.33), $N/2$ (for N even) real parameters are needed to specify A_1. By numerically searching we have been able to find an A_1 that satisfies (4.42) for $N = 52$. This search was done on a laptop without heroic effort and we are not claiming that 52 is best possible.

We have shown that the existence of a sequence $Q_0, Q_1, \ldots, Q_k$ satisfying (4.15)–(4.20) implies the existence of an exactly successful k-query translationally invariant algorithm. Now we show explicitly how a given sequence $Q_0, Q_1, \ldots, Q_k$ determines the sequence of unitary operators $V_1, V_2, \ldots, V_k$ that comprise the actual algorithm via (2.15).

First each Q_ℓ is factored as in (4.26), and (4.27) is used to find each P_ℓ. Now (4.9) is used to find $\langle x \,|\, \psi_\ell \rangle$ for $x = 0, 1, \ldots, N-1$, and together with (4.5) yields all the $\langle x \,|\, \psi_\ell \rangle$. Next we use (2.16) to obtain $\langle \mathbf{p} \,|\, \psi_\ell \rangle$ from $\langle x \,|\, \psi_\ell \rangle$.

Determining the V_ℓ's means determining the phases $\alpha_\ell(\mathbf{p})$ for each $\mathbf{p}$. To use (4.4), we need $\langle \mathbf{p} |\, \widehat{F}_0 \,| \psi_{\ell-1} \rangle$, which can be found from $|\psi_{\ell-1}\rangle$ by inserting a complete set of $|x\rangle$ states,

$$\begin{aligned} \langle \mathbf{p} |\, \widehat{F}_0 \,| \psi_{\ell-1} \rangle &= \sum_{x=0}^{2N-1} \langle \mathbf{p} |\, \widehat{F}_0 \,| x \rangle \, \langle x \,|\, \psi_{\ell-1} \rangle \\ &= \sum_{x=0}^{2N-1} \langle \mathbf{p} \,|\, x \rangle \, F_0(x) \, \langle x \,|\, \psi_{\ell-1} \rangle \ . \end{aligned} \tag{4.43}$$

For $k = 2$ and $N = 6$ we numerically carried out the program just outlined. The sequence is Q_0, Q_1, Q_2 with Q_0 and Q_2 fixed. As before, to get Q_1 we set $S_{\ell r} = 0$ in (4.28). This means that $Q_1(e^{i\theta}) = 1 + B_0(\theta)$. To obtain $Q_1(z)$ we go to (4.38) with $N = 6$ and set $\cos r\theta = (z^r + z^{-r})/2$. We then numerically factor the 10-th-degree polynomial $z^5 Q_1(z)$ and continue following the procedure given above to obtain $\alpha_1(\mathbf{p})$ and $\alpha_2(\mathbf{p})$. We convert to the $|x\rangle$ basis, where translation invariance means

$$\langle x |\, V_\ell \,| y \rangle = \langle x - y |\, V_\ell \,| 0 \rangle \tag{4.44}$$

with $x - y < 0$ replaced by $x - y + 12$. We find

x	$\langle x \| V_1 \| 0 \rangle$	$\langle x \| V_2 \| 0 \rangle$
0	.7572	.9122
1	−.3473	−.2022
2	−.0034	−.0380
3	−.0640	.0736
4	−.1367	.1258
5	−.2011	.1286
6	.2428	−.0878
7	.3473	−.2022
8	.0034	−.0380
9	.0640	.0736
10	.1367	.1258
11	.2011	.1286

(4.45)

Note that $\langle x| V_\ell |y\rangle$ are all real whenever the $S_{\ell r}$ in (4.28) are 0.

The interested reader can now check, using (2.3) for $\widehat{F}_0$, which is diagonal in the x basis, that

$$\frac{1}{\sqrt{2}}\big(|0\rangle + |6\rangle\big) = V_2\widehat{F}_0 V_1 \widehat{F}_0 \,|s\rangle \tag{4.46}$$

where $|s\rangle = \dfrac{1}{\sqrt{12}}\displaystyle\sum_{x=0}^{11} |x\rangle$. By the translation invariance of V_1 and V_2 it then follows that

$$\frac{1}{\sqrt{2}}\big(|j\rangle + |j+6\rangle\big) = V_2\widehat{F}_j V_1 \widehat{F}_j \,|s\rangle \tag{4.47}$$

for $j = 0, 1, 2, 3, 4, 5$. The 6 states in (4.47) are an orthogonal set, so (4.45) along with (4.44) is an explicit construction of an exact algorithm for the $N = 6$ insertion problem in 2 queries.

5 Conclusion

Symmetry plays a crucial role in quantum physics. We have shown that there are problems in which symmetry is useful in constructing quantum algorithms that outperform the best classical algorithm.

Acknowledgments

One of us (E.F.) thanks Sean Robinson for useful discussions. We thank Ron Rubin for stimulating discussions and Martin Stock for LaTeX-ing beyond the call of duty.

References

[1] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, `quant-ph/9812057` .

[2] H. Röhrig, `quant-ph/9812061` .

[3] L. K. Grover, `quant-ph/9605043` .