



ELSEVIER

6 March 2000

PHYSICS LETTERS A

Physics Letters A 267 (2000) 24–30

www.elsevier.nl/locate/physleta

Diagonalization of diffusion matrix in Grover's algorithm

L.C. Kwek, C.H. Oh^{*}, K. Singh, Xiang-Bin Wang

Department of Physics, Faculty of Science, National University of Singapore, Lower Kent Ridge Road, Singapore 119260, Singapore

Received 26 October 1999; received in revised form 24 January 2000; accepted 27 January 2000

Communicated by P.R. Holland

Abstract

An important process in Grover's quantum search algorithm is the iterative application of the inversion and diffusion matrix. This process can be represented by a unitary matrix, S . In this letter, we diagonalize this matrix and show how the matrix can be interpreted geometrically. © 2000 Elsevier Science B.V. All rights reserved.

In many computations, it is often essential to seek for efficient algorithms that can recover stored information from a large database quickly. Indeed, a good search algorithm always leads to substantial improvement in the speed of the computation [1]. Traditionally linked memory is a common way for achieving this speed. Specifically, one employs a mixture of trees or branching process and rummaged through a database using some hashing functions.

Recently, it has been shown that it is possible to achieve immense improvement in the speed of computation if we can realize our computation using a quantum mechanical system. A quantum mechanical system essentially ameliorates the computational process using a superposition of states. The algorithm therefore achieves some form of quantum parallelism. In particular, it has been shown recently that such quantum mechanical computations can drastically reduce the time needed to prime factorize a large number [2] and the time needed in a search algorithm [3].

In a series of seminal papers, Grover [3] introduced a quantum algorithm that could achieve a speed-up in the computational implementation with only $O(\sqrt{N})$ for a large structured database with N records. This compares favorably with the classical result which can only execute a search with $O(N)$ efficiency. Moreover, it has been shown that Grover's algorithm is optimal [4].

^{*} Corresponding author.

E-mail addresses: Scip6051@leonis.nus.edu.sg (L.C. Kwek), Phyohch@nus.edu.sg (C.H. Oh), Sciks@nus.edu.sg (K. Singh), Scip7236@leonis.nus.edu.sg (X.-B. Wang).

Grover's search algorithm can be summarized neatly into four main steps: (i) Initialization of the system into a superposition of states; (ii) Subject the system to a hashing function, $C(S)$, represented by a unitary operator, U , given by

$$U = \begin{pmatrix} -1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & 0 & 1 & \cdots & \vdots \\ \vdots & 0 & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \quad (1)$$

(assuming that the first entry satisfies the search criteria and therefore undergoes a rotation of π radians) followed by a diffusion matrix, D , defined by

$$D_{ij} = \frac{2}{N}, \quad \text{if } i \neq j \quad \text{and} \quad D_{ii} = -1 + \frac{2}{N}, \quad (2)$$

for $O(\sqrt{N})$ iterations; (iii) Measuring the resulting state. The heart of the process therefore hinges significantly on the on step (ii) and the $O(\sqrt{N})$ iterations of the matrix $DU \equiv S$. In this letter, we look closely into this important step and scrutinized the ideas behind the efficiency.

We first consider the eigenvalues of the matrix S . It is not difficult, albeit somewhat tedious, to shown that the eigenvalues of S all lie on the locus $|z| = 1$, unit circle, on the complex Argand plane and are explicitly

$$\left\{ \underbrace{-1, \cdots, -1}_{(N-2)}, \eta, \eta^* \right\},$$

where the root η and η^* satisfy the equation $z^2 - \frac{2(N-2)}{N}z + 1 = 0$ (see Appendix for details). Indeed, it can be further shown that if the matrix S is diagonalized as $P\Lambda P^{-1}$, with

$$\Lambda = \text{Diag} \left[\underbrace{-1, \cdots, -1}_{(N-2)}, \eta, \eta^* \right]$$

and

$$\eta = \frac{1}{N} (N-2 - 2i\sqrt{N-1}), \quad (3)$$

then the matrix P assumes the form

$$\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & i\sqrt{N-1} & -i\sqrt{N-1} \\ -1 & -1 & -1 & \cdots & -1 & 1 & 1 \\ 0 & 0 & 0 & \cdots & 1 & 1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & \cdots & 0 & 1 & 1 \\ 0 & 1 & 0 & \cdots & 0 & 1 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 1 & 1 \end{pmatrix}. \quad (4)$$

Since $|\eta| = 1$, we can rewrite the complex number η as $e^{-i\theta}$, where, using Eq. (3),

$$\cos \theta = \frac{N-2}{N} \quad \text{and} \quad \sin \theta = \frac{2\sqrt{N-1}}{N}. \quad (5)$$

In the quantum search, it is advantageous to seek the exact number of iterations needed so that in the measurement process, one can be assured of high probability. In general, most algorithms demand that the

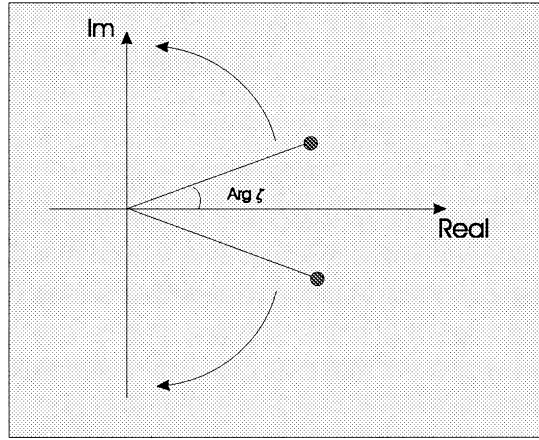


Fig. 1. Rotation of the complex number ζ about the origin in the Argand plane for the optimization.

probability of locating the required record be greater than half. However, once we have diagonalized the S matrix, we can assert a stronger condition. We can in fact seek for the number of iterations so that during the measurement process, one gets the required record with almost unity probability. Let us suppose that we need m iterations of the matrix S in order to achieve this optimal search. We therefore seek the value m such that

$$x' = U^m x_0 \quad (6)$$

$$= P \Lambda^m P^{-1} x_0, \quad (7)$$

where the final state or vector x' and the initial state x_0 are given by

$$x' = (1, 0, 0, \dots, 0)^T \quad (8)$$

$$x_0 = \frac{1}{\sqrt{N}} (1, 1, \dots, 1)^T. \quad (9)$$

(The superscript 'T' denotes transpose.)

Using Eq. (7), we see that $P^{-1}x' = \Lambda^m P^{-1}x_0$, and in the diagonalized basis, the problem reduces to finding the value of m so that the final state, $y' \equiv P^{-1}x'$, and the initial state, $y_0 \equiv P^{-1}x_0$ satisfy the relation $y' = \Lambda^m y_0$. It is not hard to compute explicitly and show that in the diagonalized basis, the states y' and y_0 are given by

$$y' = \left(0, 0, 0, \dots, 0, -\frac{i}{2\sqrt{N-1}}, \frac{i}{2\sqrt{N-1}} \right)^T, \quad (10)$$

$$y_0 = \left(0, 0, 0, \dots, 0, \frac{1}{2\sqrt{N}} - \frac{i}{2\sqrt{N-1}}, \frac{1}{2\sqrt{N}} + \frac{i}{2\sqrt{N-1}} \right)^T. \quad (11)$$

Note that Λ^m is still unitary and $\|P^{-1}x'\| = \|P^{-1}x_0\| = \frac{1}{\sqrt{2(N-1)}}$. Since y' has no real parts, geometrically in the Argand plane, an optimal search involves finding the amount of rotation $m\theta$ such that the value

$$\zeta \equiv \frac{1}{2\sqrt{N}} - \frac{i}{2\sqrt{N-1}}$$

in the state y_0 in the Argand plane is rotated, as shown in Fig. 1, to the purely imaginary number

$$\zeta' \equiv -\frac{i}{2\sqrt{N-1}}.$$

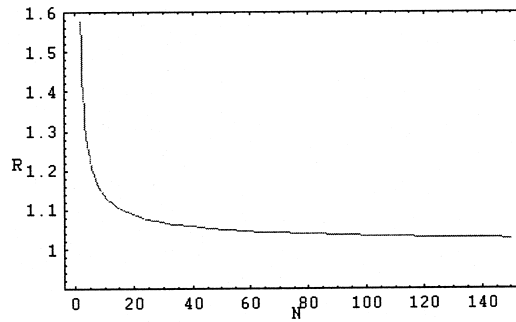


Fig. 2. Graph of the ratio R between the large N limit of $\frac{\pi\sqrt{N}}{4}$ and the value of m in Eq. (12) against the values of the database size, N .

Thus, optimization can be achieved geometrically with

$$m\theta + \text{Arg}(\zeta) = \frac{\pi}{2}. \quad (12)$$

Now, we know from Eq. (5) that

$$\theta = \tan^{-1} \frac{2\sqrt{N-1}}{N-2}, \quad (13)$$

and from Eq. (10) that

$$\text{Arg}(\zeta) = \tan^{-1} \frac{1}{\sqrt{N-1}}. \quad (14)$$

In the large N limit, that is $N \rightarrow \infty$, we see that Eq. (12) gives $m \sim \frac{1}{4}\pi\sqrt{N} \sim o(\sqrt{N})$ as expected. However, for finite N , using Eq. (12), we can work out exactly the number of steps needed to optimize the search algorithm. In particular, we have plotted the ratio R between the large N limit of $\frac{1}{4}\pi\sqrt{N}$ and the value of m in Eq. (12) in Fig. 2. We see that for finite N , as it should be at the initial testing stage in which the number of qubits is small, the difference can be quite significant.

Appendix A.

Consider the matrix $S = DU$ and rewrite the matrix, S , in the form

$$S = \begin{pmatrix} -x & y & y & \cdots & y \\ -y & x & y & \cdots & y \\ -y & y & x & \cdots & y \\ & & & \ddots & y \\ -y & y & y & \cdots & x \end{pmatrix}, \quad (\text{A.1})$$

where $x = -1 + \frac{2}{N}$ and $y = \frac{2}{N}$ so that $x - y = -1$.

where the determinants A and B are defined as

$$A = \left. \begin{array}{cccccc} -1-\lambda & 0 & 0 & 0 & \cdots & 0 \\ 1 & -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -1 \end{array} \right\} (N-1) \text{ rows,} \quad (\text{A.7a})$$

$$B = \left. \begin{array}{cccccc} y & y & y & y & \cdots & y \\ 1 & -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -1 \end{array} \right\} (N-1) \text{ rows.} \quad (\text{A.7b})$$

However, it is possible to evaluate the determinants A and B since

$$A = \left. \begin{array}{cccccc} -1-\lambda & 0 & 0 & 0 & \cdots & 0 \\ 1 & -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -1 \end{array} \right| \stackrel{\text{adding the columns}}{=} \left. \begin{array}{cccccc} -1-\lambda & 0 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 \end{array} \right\} (N-1) \text{ rows} = (1+\lambda)^{N-1}$$

and, using the same trick,

$$B = \left. \begin{array}{cccccc} y & y & y & y & \cdots & y \\ 1 & -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -1 \end{array} \right| \stackrel{\text{adding the columns}}{=} \left. \begin{array}{cccccc} (N-1)y & y & y & y & \cdots & y \\ 0 & -1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 \end{array} \right\} (N-1) \text{ rows} = (-1)^{N-2} (N-1)y.$$

Finally, putting everything together, we see that the secular equation can be simplified as

$$\begin{aligned} |S - \lambda I| &= (1 + \lambda)^{N-2} \{(-x - \lambda)A - (\lambda - 1)B\} \\ &= (1 + \lambda)^{N-2} \{(-x - \lambda)(1 + \lambda)(-1)^{N-1} + (1 - \lambda)(-1)^{N-2}(N-1)y\} \\ &= (1 + \lambda)^{N-2} (-1)^{N-2} \{(x + \lambda)(1 + \lambda) + (1 - \lambda)(N-1)y\} \\ &= (1 + \lambda)^{N-2} (-1)^{N-2} \{\lambda^2 + \lambda(1 + x - [N-1]y) + (x + [N-1]y)\} \\ &= (1 + \lambda)^{N-2} (-1)^{N-2} \left\{ \lambda^2 + \lambda \left(\frac{2}{N} - [N-1] \frac{2}{N} \right) + (-1 + Ny) \right\} \\ &= (1 + \lambda)^{N-2} (-1)^{N-2} \left(\lambda^2 - \frac{2(N-2)}{N} + 1 \right). \end{aligned} \quad (\text{A.8})$$

To obtain the eigenvectors, one only needs to consider the expression in (A.4) by substituting the appropriate eigenvalues. The result then follows immediately.

References

- [1] D.E. Knuth, *The Art of Computer Programming*, vol. 3, 2nd ed., Addison Wesley, Reading, Massachusetts, 1998. The importance of search algorithm cannot be underestimated. D.E. Knuth, the father of TeX and the METAFONT system, probably realizes its importance and devotes an entire volume, vol. 3, to search and sorting algorithm.
- [2] P.W. Shor, in: *Proc. Symposium on the Foundation of Computer Science*, 1994, Los Alamos, California, IEEE Computer Society Press, New York, 1994, p. 124.
- [3] L.K. Grover, *Phys. Rev. Lett.* 79 (1997) 325; 79 (1997) 4709; L.K. Grover, *Phys. Rev. Lett.* 80 (1998) 4329.
- [4] C.H. Bennett, E. Bernstein, G. Brassard, U. Vazirani, *SIAM J. Comput.* 26 (1997) 1510.