19 November 2001

PHYSICS LETTERS A

Physics Letters A 290 (2001) 227–233

www.elsevier.com/locate/pla

Number partitioning on a quantum computer

H. De Raedt [a,*], K. Michielsen [a], K. De Raedt [b], S. Miyashita [c]

[a] *Centre for Theoretical Physics and Materials Science Centre, University of Groningen, Nijenborgh 4, NL-9747 AG Groningen, The Netherlands*

[b] *EMS, Vlasakker 21, B-2610 Wommelgem, Belgium*

[c] *Department of Applied Physics, School of Engineering, University of Tokyo, Bunkyo-ku, Tokyo 113, Japan*

Received 17 September 2001; accepted 10 October 2001

Communicated by A. Lagendijk

Abstract

We present an algorithm to compute the number of solutions of the (constrained) number partitioning problem. A concrete implementation of the algorithm on an Ising-type quantum computer is given. © 2001 Elsevier Science B.V. All rights reserved.

PACS: 03.67.Lx; 89.70.+c

1. Introduction

The discovery of quantum algorithms (QAs) that, when executed on a quantum computer (QC), give significant speedup over their classical counterparts [1,2] has given strong impetus to recent developments in the field of quantum computation. In theory an ideal QC is a universal computer. This means that for a given problem there exists an algorithm to solve this problem on a QC. The fact that a QC is a universal computer does not tell us the algorithm itself, nor the computational resources that are required to solve the problem. In general it is not easy to construct algorithms for an ideal QC. The purpose of this Letter is to present a new QA for a problem of combinatorial optimization: Counting the number of solutions of the (constrained) number partitioning problem.

2. Number partitioning

The number partitioning problem (NPP) is defined as follows [3–5]: Does there exist a partitioning of the set $A=\{a_1,\ldots,a_n\}$ of n positive integers a_j into two disjoint sets A_1 and $A_2=A-A_1$ such that $\sum_{a_j\in A_1}a_j=\sum_{a_j\in A_2}a_j$? The answer to this question is trivially no if the sum of all a_j, $B\equiv\sum_{a_j\in A}a_j$, is odd. More generally, the case of even or odd B can be treated on the same footing by asking if there exists a partition such that $|\sum_{a_j\in A_1}a_j-\sum_{a_j\in A_2}a_j|\leqslant\Delta$, where $\Delta=1$ (0) if B is odd (even).

For certain applications there may be additional constraints on the partitioning of the set A. A common one is to fix the difference C between the number of elements in A_1 and A_2: $C\equiv\sum_{a_j\in A_1}1-\sum_{a_j\in A_2}1$. For instance, if $C=0$ we ask if there is a partitioning

[*] Corresponding author.

E-mail addresses: deraedt@phys.rug.nl (H. De Raedt), kristel@phys.rug.nl (K. Michielsen), miya@yuragi.t.u-tokyo.ac.jp (S. Miyashita).

URL address: http://rugth30.phys.rug.nl/compphys.

PII: S0375-9601(01)00680-6

such that the number of elements in A_1 and A_2 are the same.

For a given instance of $A = \{a_1, \ldots, a_n\}$, we may encode the whole problem using only $n \log_2 B$ bits. The NPP can be solved by dynamic programming, in a time bounded by a low-order polynomial in nB [4]. As nB is not bounded by any polynomial of the input size $n \log_2 B$, the dynamic programming algorithm does not solve the NPP with polynomial computational resources[4].

Number partitioning is one of Garey and Johnson's six basic NP-complete problems [4]. Number partitioning is a key problem in the theory of computational complexity and has a number of important practical applications such as job scheduling, task distribution on multiprocessor machines, VLSI circuit design to name a few.

The computation time to solve a NPP depends on the number of bits $b = \log_2 B$ needed to represent the integers a_j and B. Numerical simulations using random instances of A show that the solution time grows exponentially with n for $n \ll b$ and polynomially for $n \gg b$ [6–9]. For random instances A, the NPP can be mapped onto a hard problem of statistical mechanics, namely that of finding the ground state of an infinite-range Ising spin glass [10–12]. The transition from the computationally "hard" (exponential) to "easy" (polynomial) has been related to the phase transition in the statistical mechanical system [10,12].

Although the transition between easy and hard problems is important from conceptual point of view, it is good to keep in mind that most real-life problems are of the easy type [4]. For instance, if the a_j's represent the weight of boxes that are to be distributed over several trucks, it is highly unlikely that the weight of these boxes will vary between say 1 and 2^{32} kg, or that it is important to know the weights of the boxes with a precision of, e.g., ten digits.

3. Quantum algorithm

The potential power of a QC stems from the fact that a QC operates on superpositions of states [13–19]. The interference of these states allows exponentially many computations to be done in parallel [13–19]. A QA consists of a sequence of unitary transformations that change the state of the QC [13–19]. Therefore to solve a NPP on a QC, we first have to develop an algorithm that can be expressed entirely in terms of unitary operations.

A generic n-qubit QC can be modeled by a collection of n two-state systems, represented by n Pauli-spin matrices $\{\vec{\sigma}_1, \ldots, \vec{\sigma}_n\}$ [13–19]. The two eigenstates of σ_j^z will be denoted by $|\uparrow\rangle_j$ and $|\downarrow\rangle_j$, corresponding to the states $|0\rangle_j$ and $|1\rangle_j$ of the jth qubit respectively. The eigenvalues corresponding to $|\uparrow\rangle_j$ and $|\downarrow\rangle_j$ are $S_j = +1$ and $S_j = -1$. They can be used to represent a partitioning of A: We assign a_j to A_1 (A_2) if $S_j = +1$ ($S_j = -1$). If we can find a set $\{S_1, \ldots, S_n\}$ such that $\Delta - \sum_{j=1}^n a_j S_j = 0$, we have found one solution of the NPP.

It is known that the most simple class of spin systems, i.e., those involving interactions of the Ising type only, can be used to build universal QCs [14,18, 20]. For our purposes it is, at this stage, sufficient to consider a system of n non-interacting Ising spins. The Hamiltonian is given by

$$H = \Delta - \sum_{j=1}^{n} a_j \sigma_j^z, \tag{1}$$

where the a_j's represent external fields acting on the spins. From (1) it follows directly that an eigenstate of H with energy zero corresponds to a solution of the NPP. We will use Hamiltonian (1) to define the time evolution of the QC, i.e., the QA that solves NPPs.

The second key to the construction of the quantum algorithm is the observation that the number of solutions n_s of a NPP is given by

$$n_s \equiv \frac{1}{M} \sum_{m=0}^{M-1} \operatorname{Tr} e^{-2\pi i m H/M}, \tag{2}$$

where $M \equiv B + \Delta + 1$ and $\operatorname{Tr} U$ denotes the trace of the matrix U [21]. Using the representation that diagonalizes the spin operators σ_j^z, we find

$$\begin{aligned} n_s &= \sum_{\{S_j=\pm1\}} \frac{1}{M} \sum_{m=0}^{M-1} \exp\left[\frac{2\pi i m}{M}\left(\sum_{j=1}^{n} a_j S_j - \Delta\right)\right] \\ &= \frac{1}{M} \sum_{\{S_j=\pm1\}} \frac{1 - \exp\left[2\pi i\left(\sum_{j=1}^{n} a_j S_j - \Delta\right)\right]}{1 - \exp\left[2\pi i\left(\sum_{j=1}^{n} a_j S_j - \Delta\right)/M\right]} \\ &= \sum_{\{S_j=\pm1\}} \delta_{\Delta, \sum_{j=1}^{n} a_j S_j}. \end{aligned} \tag{3}$$

As $|\Delta - \sum_{j=1}^{n} a_j S_j| < M$ for any choice of $\{S_j\}$, the sum over m in (2) will be zero unless $\Delta - \sum_{j=1}^{n} a_j S_j = 0$, in which case the configuration $\{S_1, \ldots, S_n\}$ is a solution of the NPP (note that there can be exponentially many solutions, for instance, in the case that all the a_j's are equal). Performing the sum over all spin configurations as indicated in (3), it follows immediately that n_s is the number of solutions of the NPP. Note that (2) gives the number of solutions of a NPP, which is more than just a yes or no answer to the question if a partition of A exists [4].

Formally expression (2) is the density of states at zero energy of the physical system described by Hamiltonian (1). Elsewhere we have shown that, for a large class of models H, the density of states can be calculated efficiently on a QC [22]. The algorithm presented below, although related to the one described in [22], is specifically tuned to solve NPPs.

The equivalence of (2) and the solution of the NPP can also be demonstrated by explicit calculation of the trace over all spin configurations. This is easy because the spins do not interact. The result is

$$n_s = 2^n M^{-1} \sum_{m=0}^{M-1} e^{-2\pi i m \Delta/M} \prod_{j=1}^{n} \cos(2\pi m a_j/M). \tag{4}$$

For $\Delta = 0$ and in the limit $M \to \infty$ we have $n_s = 2^n I_s$, where

$$I_s = \frac{1}{2\pi} \int_0^{2\pi} \cos(a_1\theta)\ldots\cos(a_n\theta)\, d\theta. \tag{5}$$

The question whether $I_s = 0$ or not is known to be equivalent to the (non-) existence of a solution of a NPP [4,23].

If $n_s > 0$ we can find a partitioning in the following manner. Assume we already know the values of the first $0 < l - 1 < n$ spins. We make a guess for S_l and compute $n_s^{(l)} \equiv M^{-1} \sum_{m=0}^{M-1} \operatorname{tr} e^{-2\pi i m H/M}$, where the use of the symbol tr instead of Tr indicates that in calculating the trace, the values of the variables $S_1, \ldots, S_l$ are fixed. If $n_s^{(l)} > 0$ our guess for S_l was correct, if not we reverse S_l. Then we increase l by one and repeat the procedure.

The algorithm outlined above is easily generalized to handle constraints. Introducing another Hamiltonian

$$H' = C - \sum_{j=1}^{n} \sigma_j^z, \tag{6}$$

the number of solutions $n_s(C)$ of the constrained NPP is given by

$$n_s(C) \equiv \frac{1}{MK} \sum_{k=0}^{K-1} \sum_{m=0}^{M-1} \operatorname{Tr} e^{-2\pi i m H/M} e^{-2\pi i k H'/K}, \tag{7}$$

where $K = n + |C| + 1$. Repeating the same steps as above we find that the sum over k yields zero unless $C = \sum_{j=1}^{N} S_j = \sum_{a_j \in A_1} 1 - \sum_{a_j \in A_2} 1$. The expression corresponding to (4) reads

$$n_s(C) = \frac{2^n}{MK} \sum_{k=0}^{K-1} \sum_{m=0}^{M-1} e^{-2\pi i m \Delta/M - 2\pi i k C/K} \times \prod_{j=1}^{n} \cos\left(\frac{2\pi m a_j}{M} + \frac{2\pi k}{K}\right). \tag{8}$$

The procedure to find a partitioning itself is the same as in the unconstrained case.

The algorithms defined by (1), (2) and (6), (7) solve NPPs and constrained NPPs without recourse to dynamic programming. This follows directly from explicit expressions (4) and (8). On a conventional computer the computation time required is bounded by nM (or nMK for the constrained case). Hence also these algorithms do not solve the (constrained) NPP in polynomial time (space). The conceptual difference between these algorithms and the dynamic-programming approach is that the former directly compute the number of solutions of the NPP whereas the latter performs a search for a solution of the NPP.

As we now show, (2) (or (7)) is a convenient starting point to construct a QA that solves the (constrained) NPP on a QC. As will be clear from the discussion below, it will be sufficient to concentrate on (2), the algorithm for (7) is almost identical.

The first step is to introduce a "number operator" X with eigenstates $|x\rangle$, $X|x\rangle = x|x\rangle$, $x = 0, 1, \ldots, M - 1$. We modify the Hamiltonian that governs the time-evolution of the QC as follows:

$$\tilde{H} = \Delta X - \sum_{j=1}^{n} a_j \sigma_j^z X. \tag{9}$$

Calculating the trace in the basis that diagonalizes $\sigma_1^z \ldots \sigma_n^z$ and X we find that $n_s = M^{-1}\operatorname{Tr} e^{-2\pi i \tilde{H}/M}$. Because $\tilde{H}$ is diagonal in this basis $\operatorname{Tr} e^{-2\pi i \tilde{H}/M}$ is proportional to one matrix element, namely

$$\begin{aligned}&\operatorname{Tr} e^{-2\pi i \tilde{H}/M}\\ &\quad = 2^n M \langle U_1 \ldots U_n U_x | e^{-2\pi i \tilde{H}/M} | U_1 \ldots U_n U_x \rangle, \quad (10)\end{aligned}$$

where $|U_j\rangle \equiv (|{\uparrow}\rangle_j + |{\downarrow}\rangle_j)/\sqrt{2}$ is the uniform superposition of the spin up and down state of spin j, and $|U_x\rangle \equiv (|0\rangle + |1\rangle + \cdots + |M-1\rangle)/\sqrt{M}$ is the uniform superposition of all the eigenstates of the number operator X. To derive expression (10) we made use of

$$e^{-ia\sigma_j^z}|U_j\rangle = \cos(a)|U_j\rangle - i\sin(a)\big|\bar{U}_j\big\rangle \qquad (11)$$

and $\langle U_j|\bar{U}_j\rangle = 0$, where $|\bar{U}_j\rangle = (|{\uparrow}\rangle_j - |{\downarrow}\rangle_j)/\sqrt{2}$.

From (2) it follows that

$$n_s = 2^n \langle U_1 \ldots U_n U_x | e^{-2\pi i \tilde{H}/M} | U_1 \ldots U_n U_x \rangle. \qquad (12)$$

As a QC can compute $e^{-itH}|\psi\rangle$ with one operation (for arbitrary input $|\psi\rangle$) [13], (12) shows that once the QC is in the state of uniform superposition $|U_1 \ldots U_N U_x\rangle$, one time-evolution step of the QC will solve the NPP.

The initial state $|{\uparrow}, \ldots, {\uparrow}, x = 0\rangle$ can be transformed into the state of uniform superposition $|U_1 \ldots U_N U_x\rangle$ by the standard procedure [16,17]: The states $|U\rangle_j$ can be obtained from the initial state $|{\uparrow}\rangle_j$ by a rotation of the spin j about the y-axis, i.e., $|U\rangle_j = e^{-i\pi\sigma_j^y/4}|{\uparrow}\rangle_j$ for $j = 1, \ldots, n$. On an Ising-type QC the states $|x\rangle$ can be implemented by adding new two-state systems. We denote the corresponding Pauli-spin operators and eigenvalues by $\vec{\mu}_p$ and s_p, respectively. We use these spins to represent $x = \sum_{l=1}^{p} 2^{l-2}(1 - s_l)$ in binary form. As $0 \leqslant x < M$ the number of spins p required to represent x is the smallest integer p for which $M \leqslant 2^p$. Using this binary representation for $|x\rangle$, the uniform superposition $|U_x\rangle$ can be obtained by p rotations of the initial state:

$$|U_x\rangle = e^{-i\pi\mu_1^y/4}|{\uparrow}\rangle_1 \otimes \cdots \otimes e^{-i\pi\mu_p^y/4}|{\uparrow}\rangle_p, \qquad (13)$$

where $\otimes$ denotes the direct product operation. The system now comprises $n + p$ spins and its Hamiltonian reads

$$\mathcal{H} = -\sum_{l=1}^{p}\sum_{j=1}^{n} J_{j,l}\sigma_j^z\mu_l^z - \sum_{j=1}^{n} b_j\sigma_j^z - \sum_{l=1}^{p} c_l\mu_l^z + d, \qquad (14)$$

where $J_{j,l} = -a_j 2^{l-2}$, $b_j = a_j(2^p - 1)/2$, $c_l = \Delta 2^{l-2}$, and $d = \Delta(2^p - 1)/2$.

The complete QA for computing n_s, i.e., for solving NPPs, can be summarized as follows: The initial state of the QC (all spins up by convention) is transformed into the state of uniform superposition. This takes $n + p$ one-qubit operations. Next the QC makes one time-evolution step $\exp(-i\pi\mathcal{H}/2^{p-1})$. The matrix element in (12) is obtained by applying the inverse of the $n + p$ rotations, followed by a projection onto the initial state. Clearly the total number of QC operations is only $2n + 2p + 1$ while the amount of memory used is $\mathcal{O}(\log_2 M + \log_2 n)$.

The constrained NPP can be solved in the same way: Add qubits to represent the variable k in (7) and repeat the steps that lead to (12). Note that once the uniform superposition has been prepared, the QA also solves the constrained NPP with one time-evolution step.

4. Implementation on a quantum computer emulator

For the purpose of demonstration we have implemented the QA that solves the unconstrained NPP on a quantum computer emulator (QCE), a software tool for simulating physical models of QCs [24]. A subtle point thereby is that (12) is not directly measurable because $e^{-2\pi i \tilde{H}/M}$ is not a physical observable. However, it is not difficult to express n_s in terms of an expectation value of a physical observable.

Let us write the number of solutions (7) as $n_s = 2^n\langle 0|\Phi\rangle$, where $|\Phi\rangle = U^{-1}e^{-i\pi\mathcal{H}/2^{p-1}}U|0\rangle$ and $U \equiv e^{-i\pi\sigma_1^y/4} \ldots e^{-i\pi\sigma_n^y/4} e^{-i\pi\mu_1^y/4} \ldots e^{-i\pi\mu_p^y/4}$. Our aim is to replace the projection onto the initial state $|0\rangle$, a shorthand notation for the state with all spins up, by the measurement of some observable. This can be accomplished by introducing another spin $\vec{\kappa}$, initially in the state of spin up, to the system and flip this spin if the other $n + p$ are all up, i.e., by performing an AND operation on the $n + p$ spins. With V denoting the unitary transformation that performs this AND operation, we have in the language of qubits instead of spins:

$$\begin{aligned}|\Psi\rangle &\equiv VU^{-1}e^{-i\pi\mathcal{H}/2^{p-1}}U|0\rangle \otimes |0\rangle_\kappa\\ &= V\big[2^{-n}n_s|0\rangle \otimes |0\rangle_\kappa + (\ldots) \otimes |0\rangle_\kappa\big]\end{aligned}$$

$$= 2^{-n} n_s |0\rangle \otimes |1\rangle_\kappa + (\ldots) \otimes |0\rangle_\kappa, \tag{15}$$

where $|\Psi\rangle$ is an element of the direct product of the Hilbert spaces spanned by the $n+p$ spins and the auxiliary spin $\vec{\kappa}$. We use the abbreviation $(\ldots)$ to represent the sum of all states of the $n+p$ spins that have at least one spin down. From (15) it immediately follows that

$$n_s = 2^n \langle \Psi | (1-\kappa^z)/2 | \Psi \rangle^{1/2}. \tag{16}$$

It is well-known how to implement the AND operation on a QC [25]. In our practical implementation [26], we have chosen to use a three-bit network, the Toffoli-gate, as a building block for realizing the AND operation on the $n+p$ qubits [25]. By adding extra work qubits the complete network requires of the order of $\log_2(n+p)$ steps and $n+p$ extra qubits to perform the AND operation. Clearly this does not change the polynomial time and space character of the QA that solves NPPs. A block diagram of the complete quantum program is shown in Fig. 1. We have implemented the QA on a 15-qubit QC and used it to solve the NPPs $A = \{1,2,3,4\}$, $A = \{1,1,1,4\}$ and $A = \{2,2,2,4\}$ (these examples are included in the software distribution [26]). In the final state the expectation values of the 15th qubit are 0.015625, 0.00390625, and 0, respectively. The corresponding number of solutions is $n_s = 2$, $n_s = 1$ and $n_s = 0$. Clearly the demonstration program correctly solves NPP problems.

5. Alternative implementation

The implementation described above has the same logical structure as other QAs [1,2,19]: Prepare the QC in a state of uniform superposition, perform some unitary transformation to encode information and then apply a filter to extract the answer. We now show that there is another QA that solves the NPP but does not fit into this general scheme in that the first step is missing.

Consider the time-dependent n-spin correlation function

$$C(t) = \langle \Phi | e^{iH_x t} \sigma_1^z \ldots \sigma_n^z e^{-iH_x t} \sigma_1^z \ldots \sigma_n^z | \Phi \rangle, \tag{17}$$

where $H_x = -\sum_{j=1}^n a_j \sigma_j^x / 2$. The state $|\Phi\rangle$ can be any n-spin state that is an eigenstate of $\sigma_1^z \ldots \sigma_n^z$, e.g., the state with all spins up. As the σ_j^z's are unitary operators, it is a simple matter to write down a QA that computes $C(t)$ on a QC. Obviously $C(t)$ is a physically observable quantity but it may require a rather complicated experimental setup to measure this n-spin correlation function.

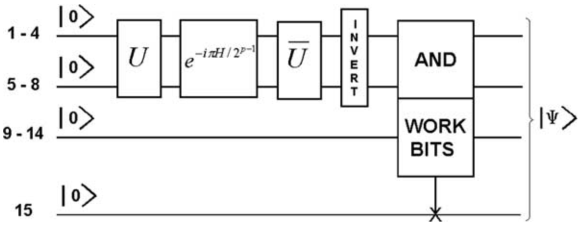

Fig. 1. Block diagram of the quantum algorithm that solves the number partitioning problem. In this example the first $n = 4$ qubits are used to represent the integers to be partitioned. The $p = 4$ qubits 5–8 are used to determine the number of solutions of the number partitioning problem. The remaining 7 qubits are used to relate n_s to a physically measurable quantity: The expectation value of the 15th qubit. The unitary transformation U prepares the uniform superposition of the first 8 qubits, $\bar{U}$ is the inverse of U, and the combination of INVERT and AND gates sets the 15th qubit to one if and only if the first eight qubits are all one.

Substituting into (17) the equation of motion for each spin, i.e., $e^{iH_x t} \sigma_j^z e^{-iH_x t} = \sigma_j^z \cos(a_j t) - \sigma_j^y \times \sin(a_j t)$, we find

$$C(t) = \langle \Phi | \prod_{j=1}^n \bigl[\mathbf{1} \cos(a_j t) - i \sigma_j^x \sin(a_j t) \bigr] | \Phi \rangle$$
$$= \prod_{j=1}^n \cos(a_k t). \tag{18}$$

The Fourier transform of $C(t)$ at zero frequency is directly proportional to I_s and hence to n_s:

$$S(\omega) = \int_{-\infty}^{\infty} e^{i\omega t} C(t)\, dt = 2^{-n} n_s \delta(\omega) + R(\omega), \tag{19}$$

where the regular part $R(\omega)$ is zero at $\omega = 0$. From (19) it is clear that the NPP has a solution if $S(\omega)$ shows a peak at zero frequency. Detection of the central peak in the dynamic correlation function $S(\omega)$ may require a very long observation time T. To distinguish between $n_s = 0$ and $n_s = 1$ the observation time T must be larger than $2^n \pi$.

6. Discussion

The essence of the algorithm proposed above is that it yields the number of solutions n_s without actually listing all solutions. This is made possible by expressing n_s in terms of the density of states of a physical system (Ising spins in our example). Clearly this QA certainly has its weakness if n_s is close to zero and n is large. Of the order of 2^n measurements of κ_z are required to distinguish between $n_s = 0$ and $n_s = 1$. This is tantamount to random sampling. By formulating, as we did, the outcome of the calculation in terms of (an average of) physical observables (see (16) or (19)) instead of a (collapsed) state, this problem of efficiency is difficult to overlook [27].

The NP-completeness of the NPP depends strongly on the fact that extremely large numbers a_j are allowed [4]. For a given instance of $A = \{a_1, \dots, a_n\}$, the whole problem can be encoded using only $n \log_2 B$ bits [4] and running the algorithms defined by (1), (2) or (6), (7) on a conventional computer requires a computation time that is proportional to nB. On a QC it takes only of the order of $n \log_2 B$ steps to complete our QA but in the hard case ($b \gg n$) and for large n, obtaining a yes-or-no answer requires tremendous precision. In the absence of any information of the a_j's other than that they are positive integers, the range of n_s extends from zero to $\binom{n}{n/2}$. Any algorithm that computes n_s (on a conventional or QC) should be able to cover this range (otherwise it can never return the correct n_s). This implies that the whole computation has to be done with at least the same (high) precision.

As the NPP example shows, a realistic assessment of the potential power of a QA should include a quantitative estimate of the precision and other computational resources (e.g., energy) that are required to obtain the correct answer. For our QA an estimate of the required precision follows from (16). Note that the alternative implementation yields a similar, physically equivalent, estimate for the observation time T.

The range of numbers a physically realizable QC will be able to handle is directly related to the energy range in which the QC operates ($-B$ to $+B$ in the NPP case). Although not a problem of principle, the physics of QC hardware will definitely impose some constraints on the range of numbers.

There are two other, potentially large, numbers involved in an NPP problem: The number of states $N_s = 2^n$ and the number of computational units N_{cu} (of microscopic size) in the physical realization of the QC. There are two cases to consider. (1) Theoretical (computer science): We have to examine the worst case. Then $N_s \gg N_{\text{cu}}$ so that our NPP algorithm has little merit. (2) In practice: In numerical experiments [6–9] $n \leqslant 32$ ($N_s < 2^{32}$) whereas, for instance, in NMR QC experiments $N_{\text{cu}} \approx 10^{18} \gg N_s$ [28]. Using a sufficiently large number of computational units and efficient detectors it should be possible to distinguish between $n_s = 0$ and $n_s = 1$. Assuming that clever engineering can produce spin systems such as (14), our QA might be used to demonstrate that a physical QC can solve a non-trivial problem.

Acknowledgement

This work is partially supported by the Dutch 'Stichting Nationale Computer Faciliteiten' (NCF).

References

[1] P.W. Shor, in: S. Goldwasser (Ed.), Proc. 35th Ann. Symp. on Foundations of Computer Science, IEEE Computer Society, Los Alamitos, CA, 1994, p. 124.

[2] L.K. Grover, Phys. Rev. Lett. 79 (1997) 4709.

[3] R.M. Karp, in: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, 1972, p. 85.

[4] M.R. Garey, D.S. Johnson, Computers and Intractability. A Guide to the Theory of NP-Completeness, W.H. Freeman, New York, 1999.

[5] T.H. Cormen, C.E. Leiserson, R.L. Rivest, Introduction to Algorithms, MIT Press, Cambridge, 1994.

[6] I.P. Gent, T. Walsh, in: Proceedings of ECAI-96, Wiley, New York, 1996, p. 170.

[7] I.P. Gent, T. Walsh, Comp. Intell. 14 (1998) 430.

[8] R.E. Korf, Artif. Intell. 106 (1998) 181.

[9] S. Mertens, cs.DS/9903011, Elsevier preprint (2000).

[10] S. Mertens, Phys. Rev. Lett. 81 (1998) 4281.

[11] F.F. Ferreira, J.F. Fontanari, J. Phys. A: Math. Gen. 31 (1998) 3417.

[12] S. Mertens, Phys. Rev. Lett. 84 (2000) 1347.

[13] R.P. Feynman, Int. J. Theor. Phys. 21 (1982) 467.

[14] S. Lloyd, Science 261 (1993) 1569.

[15] D.P. DiVincenzo, Science 270 (1995) 255.

[16] A. Ekert, R. Jozsa, Rev. Mod. Phys. 68 (1996) 733.

[17] V. Vedral, M. Plenio, Prog. Quantum Electron. 22 (1998) 1.

[18] G.P. Berman, G.D. Doolen, R. Mainieri, V.I. Tsifrinovich, Introduction to Quantum Computers, World Scientific, Singapore, 1998.

[19] P.W. Shor, SIAM Rev. 41 (1999) 303.

[20] G.P. Berman, G.D. Doolen, D.D. Holm, V.I. Tsifrinovich, Phys. Lett. A 193 (1994) 444.

[21] This value of M is not optimal and can be reduced by a factor of about two. However, this is irrelevant for what follows and we therefore omit the discussion of this technical point.

[22] H. De Raedt, A. Hams, K. Michielsen, S. Miyashita, K. Saito, Prog. Theor. Phys. (Suppl.) 138 (2000) 489.

[23] D.A. Plaisted, in: Proc. 17th Ann. Symp. on Foundations of Computer Science, IEEE Computer Society, Long Beach, CA, 1976, p. 264.

[24] H. De Raedt, A.H. Hams, K. Michielsen, K. De Raedt, Comp. Phys. Comm. 132 (2000) 94.

[25] A. Barenco, C.H. Bennet, R. Cleve, D.P. DiVincenzo, N. Margolus, P.W. Shor, T. Sleator, J.A. Smolin, H. Weinfurther, Phys. Rev. A 52 (1995) 3457.

[26] The quantum program described in the Letter is included in the QCE (version 7.0.1 and higher) software distribution which can be downloaded from http://rugth30.phys.rug.nl/compphys/qce.htm.

[27] Shor's algorithm for finding the period of a function is no different in this respect, see also M. Nielsen, I. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000.

[28] I.L. Chuang, L.M.K. Vandersypen, X. Zhou, D.W. Leung, S. Lloyd, Nature 393 (1998) 143.