A quantum speedup in machine learning: finding an *N*-bit Boolean function for a classification

New Journal of Physics
The open access journal at the forefront of physics
Deutsche Physikalische Gesellschaft DPG | IOP Institute of Physics

A quantum speedup in machine learning: finding an *N*-bit Boolean function for a classification

Seokwon Yoo[1,2], Jeongho Bang[2,1], Changhyoup Lee[3,1] and Jinhyoung Lee[1,2]
[1] Department of Physics, Hanyang University, Seoul 133-791, Korea
[2] Center for Macroscopic Quantum Control & Department of Physics and Astronomy, Seoul National University, Seoul, 151-747, Korea
[3] Centre for Quantum Technologies, National University of Singapore, 3 Science Drive 2, 117543 Singapore
E-mail: hyoung@hanyang.ac.kr

Received 9 May 2014, revised 31 July 2014
Accepted for publication 1 September 2014
Published 9 October 2014
New Journal of Physics **16** (2014) 103014
doi:10.1088/1367-2630/16/10/103014

Abstract
We compare quantum and classical machines designed for learning an *N*-bit Boolean function in order to address how a quantum system improves the machine learning behavior. The machines of the two types consist of the same number of operations and control parameters, but only the quantum machines utilize the quantum coherence naturally induced by unitary operators. We show that quantum superposition enables quantum learning that is faster than classical learning by expanding the approximate solution regions, i.e., the acceptable regions. This is also demonstrated by means of numerical simulations with a standard feedback model, namely random search, and a practical model, namely differential evolution.

Keywords: quantum information, quantum learning, machine learning

1. Introduction

Over the past few decades, quantum physics has brought remarkable innovations into fields of various disciplines. For example, there are exponentially faster quantum algorithms, compared to their classical counterparts [1–3]. The physical limit of measurement precision has been improved in quantum metrology [4, 5], and a large number of protocols offering higher security have been proposed in quantum cryptography [6, 7]. These achievements are enabled by appropriate usage of quantum effects such as quantum superposition and quantum entanglement.

Another important scientific area is machine learning, which is a subfield of artificial intelligence and one of the most advanced automatic control techniques. While learning is usually regarded as a characteristic of humans or living things, machine learning enables a machine to learn a task [8]. Machine learning has been attracting great attention, with its novel ability to learn. On one hand, machine learning has been studied to provide an understanding of the learning of a real biological system, in a theoretical manner. On the other hand, machine learning is also expected to provide reliable control techniques for use in designing complex systems in a practical manner [8].

Recently, the hybridizing of the two scientific fields described above, quantum technology and machine learning, has received great interest [9–12]. One question naturally arises: can machine learning be improved by using favorable quantum effects? Several attempts to answer this question have been made in the past few years—for example, using quantum perceptrons [13], neural networks [14–16], and quantum-inspired evolutionary algorithms [17, 18]. Most recently, remarkable studies have been carried out [19–22]. In [19], a learning speedup for the quantum machine was observed with a lower memory requirement for a specific example, namely the kth-root NOT operation. In [20], a strategy for designing a quantum algorithm was introduced, establishing a link between the learning speedup and the speedup of the quantum algorithm found. In [21, 22], the authors showed quantum speedup for the task of classifying a large number of data. However, it is still unclear what quantum effects work in machine learning and how they work, particularly in the absence of a fair comparison between classical and quantum machines.

In this work, we consider a binary classification problem as a learning task. Such a classification can be realized for an N-bit Boolean function that maps a set of N-bit binary strings in $\{0, 1\}^N$ into $\{0, 1\}$ [23]. The main objective in this paper is to compare a quantum machine with a classical machine. These two machines are equivalent. The only differentiation is that the quantum machine can deal with quantum effects, whereas the classical machine cannot. The machines are analyzed in terms of the acceptable region, defined as a localized solution region of parameter space. In the analysis, it is shown that the quantum machine can learn faster due to the acceptable region being expanded by quantum superposition. Such a quantum learning speedup is understood in terms of an expansion of the acceptable region. In order to make the analysis more explicit, we analyze further by using random search, which is a standard model for use in learning performance analysis [24]. In such a primitive model, we validate the quantum speedup, showing that the overall number of iterations required to complete the learning is proportional to $O(e^{\alpha D})$, with $\alpha \simeq 3.065$ for the classical machine and $\alpha \simeq 0.238$ for the quantum machine. Here, D is the size of the search space. Differential evolution is employed as a learning model, taking into account more realistic circumstances. By means of numerical simulations, we show that the quantum speedup is still observed even in such a case.

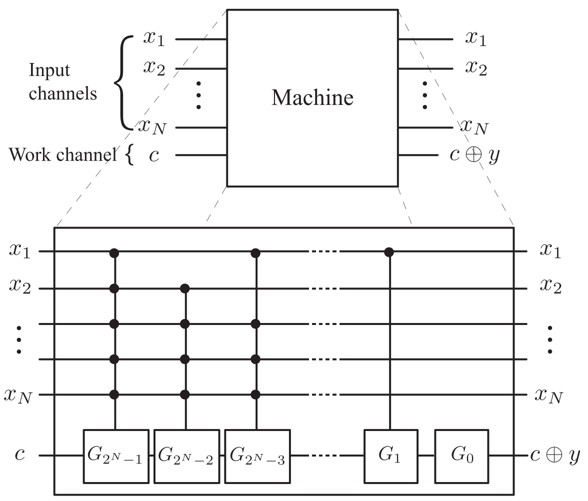

Figure 1. The N-bit Boolean function is implemented by a reversible circuit. The machine consists of N-bit input channels and a single-bit work channel, which contains 2^N operations: one single-bit operation G_0, and $2^N - 1$ operations G_k conditioned by the input bits $\boldsymbol{x}$. Here, the constant input c is set to be 0, which gives rise to an output bit y.

2. Classical and quantum machines

Machine learning can be decomposed into two parts: the machine and the feedback. The machine performs various tasks depending on its internal parameters, and the feedback adjusts the parameters of the machine in order for the machine to perform a required task called the target. Learning is a process involving finding suitable parameters for the machine, whereby the machine is expected to generate desired results working towards a target[4]. This concept of machine learning has been widely adopted in the context of machine learning at the fundamental level [8].

In this work, we assign to a machine a binary classification problem as a task, where the machine will learn a target N-bit Boolean function, defined as

$$f\colon \boldsymbol{x} \in \{0,\, 1\}^N \to y \in \{0,\, 1\}, \tag{1}$$

where $\boldsymbol{x} = x_N \dots x_2 x_1$ is represented as an N-bit string of $x_j \in \{0,\, 1\}$ $(j = 1,\, 2,\, \dots,\, N)$. This function can be written by using the positive polarity Reed–Muller expansion [25]:

$$f(\boldsymbol{x}) = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus a_3 x_1 x_2 \oplus \cdots \oplus a_{2^N-1} x_1 \cdots x_N = \bigoplus_{k=0}^{2^N-1} \left(a_k \prod_{j \in \mathrm{C}_k} x_j \right), \tag{2}$$

where $\oplus$ denotes modulo-2 addition, $\bigoplus$ means a direct sum of the moduli, and the Reed–Muller coefficients a_k are either 0 or 1. Here, C_k is an index set whose elements are given in such a way. The number j is then taken to be an element of C_k only if k_j is equal to 1 when k is written as an N-bit string, $k_N \dots k_2 k_1$. Thus, each set of $\{a_k\}$ corresponds to each of 2^{2^N} Boolean functions.

The Boolean function can be implemented by a reversible circuit as shown in figure 1, where an additional bit channel, called the work channel, and controlled operations are

[4] We consider the case of supervised learning, where the desired results of the task are given to the machine.

Table 1. Four possible one-bit Boolean functions are given with Reed–Muller coefficients (a_0 and a_1), and operations (G_0 and G_1). These are common to the classical and quantum cases.

Boolean function	a_0	a_1	G_0	G_1
$f_1: x \mapsto 0$	0	0	Identity	Identity
$f_2: x \mapsto 1$	1	0	NOT	Identity
$f_3: x \mapsto x$	0	1	Identity	NOT
$f_4: x \mapsto x \oplus 1$	1	1	NOT	NOT

employed [26, 27]. A single-bit operation G_0 is placed on the work channel and $(2^N - 1)$ controlled-G_k operations are caused to act on the work channel when all the control bits, x_j ($j \in \mathrm{C}_k$), are 1. The input signal c on the work channel is fixed at 0. The operation G_k is given as either the identity (i.e., doing nothing), if $a_k = 0$, or NOT (i.e., flipping an input bit to its complement bit), if $a_k = 1$. As an example, a one-bit Boolean function (i.e., with $N = 1$) has $2^{2^1} = 4$ sets of Reed–Muller coefficients (a_0, a_1), which determine all possible Boolean functions. Table 1 gives four possible one-bit Boolean functions with Reed–Muller coefficients and the corresponding operations.

With a reversible circuit model, we then define classical and quantum machines. The classical machine consists of classical channels and operations, and the Boolean function of the classical machine is described as

$$(\boldsymbol{x}, c) \xrightarrow{f} (\boldsymbol{x}, c \oplus y) \tag{3}$$

with classical bits $\boldsymbol{x}$, y, and c. We suppose the Reed–Muller coefficients a_k to be probabilistically determined by the internal parameters p_k, which implies that G_k performs the identity and NOT operations with probabilities p_k and $1 - p_k$, respectively. These probabilistic operations are primarily intended to provide a fair comparison with the quantum machine, which naturally employs a probabilistic operation. Now, we construct the quantum machine by setting only the work channel to be quantum. The input channels are left as classical, as the input information is classical in our work. Thus, the Boolean function of the quantum machine is described as

$$(\boldsymbol{x}, |c\rangle) \xrightarrow{f} (\boldsymbol{x}, |\psi\rangle), \tag{4}$$

where the signal on the work channel is encoded into a qubit state. The classical probabilistic operations G_k are also necessarily replaced by unitary operators:

$$\hat{G}_k = \begin{pmatrix} \sqrt{p_k} & \mathrm{e}^{\mathrm{i}\phi_k}\sqrt{1 - p_k} \\ \mathrm{e}^{-\mathrm{i}\phi_k}\sqrt{1 - p_k} & -\sqrt{p_k} \end{pmatrix}, \tag{5}$$

where p_k is the probability of $\hat{G}_k$ performing the identity operation, i.e., $|0\rangle \to |0\rangle, |1\rangle \to \mathrm{e}^{\mathrm{i}\pi}\,|1\rangle$, and $1 - p_k$ is that of $\hat{G}_k$ performing the NOT operation, i.e., $|0\rangle \to \mathrm{e}^{-\mathrm{i}\phi_k}\,|1\rangle, |1\rangle \to \mathrm{e}^{\mathrm{i}\phi_k}\,|0\rangle$. Note that the relative phases ϕ_k are free parameters suitably chosen before the learning. The feedback adjusts only the p_k parameters, controllable both in the classical and the quantum experimental setups [28, 29].

These classical and quantum machines are equivalent to each other. They have the same circuit structures and exactly the same number of control parameters, p_k. Moreover, the single classical operation G_k and the quantum operator $\hat{G}_k$ cannot be discriminated between by measuring the distribution of outcomes for the same input $\boldsymbol{x}$ and parameters p_k.

3. The acceptable region

A target Boolean function is represented by a point, $\mathrm{Q}_f = (p_0, p_1, \ldots, p_{2^N-1})$, in the 2^N -dimensional search space spanned by the probabilities, p_k. For example, four possible learning targets, f_j ($j = 1, 2, 3, 4$), for the one-bit Boolean function, correspond to four points in the search space: $\mathrm{Q}_{f_1} = (1, 1)$, $\mathrm{Q}_{f_2} = (0, 1)$, $\mathrm{Q}_{f_3} = (1, 0)$, and $\mathrm{Q}_{f_4} = (0, 0)$. Similarly, the machine behavior is also characterized as a point $\mathrm{Q}_\mathrm{m} = (p_0, p_1)$, i.e., the respective points lead to different probabilistic tasks that the machine performs. Learning is simply regarded as a process of moving Q_m to a given target point in the whole search space. It is, however, usually impractical (actually, impossible in real circumstances) to locate Q_m exactly at the target point. Instead, it is feasible to find approximate solutions near to the exact target, i.e. the learning is expected to lead the point Q_m into a region near to the target point [8]. We call such a region an acceptable region for the approximate target functions. As the learning time and convergence depend primarily on the size of the acceptable region, it is usually expected that a larger acceptable region will make the learning faster [30]. We examine, in this sense, the acceptable regions of classical and quantum machines.

The acceptable region is defined as a set of points which guarantee the errors, $\epsilon = 1 - \mathcal{F}$, to be less than or equal to a tolerable value, ϵ_t. Here, $\mathcal{F}$ is the figure of merit of the machine performance, called the task fidelity, and it quantifies how well the machine performs a target function, defined by

$$\mathcal{F}(p_0, p_1, \ldots, p_{2^N-1}) = \left(\prod_{\boldsymbol{x}} \sum_{y} \sqrt{P(y|\boldsymbol{x}) P_\tau(y|\boldsymbol{x})} \right)^{\frac{1}{2^N}}, \tag{6}$$

where $P(y|\boldsymbol{x})$ is a conditional probability of obtaining an output y given an input $\boldsymbol{x}$, and the target probabilities $P_\tau(y|\boldsymbol{x})$ are those for the target. For example, we have target probabilities for f_1 in table 1 written as

$$P_\tau(0|0) = 1,\ P_\tau(1|0) = 0,\ P_\tau(0|1) = 1,\ \text{and}\ P_\tau(1|1) = 0. \tag{7}$$

The term $\sum_y \sqrt{P(y|\boldsymbol{x}) P_\tau(y|\boldsymbol{x})}$ in equation (6) corresponds to the closeness of the two probability distributions $P(y|\boldsymbol{x})$ and $P_\tau(y|\boldsymbol{x})$ for the given $\boldsymbol{x}$ [31]. The task fidelity, $\mathcal{F}$, increases as the outputs get close to the required outputs; $\mathcal{F}$ becomes unity only when the machine gives the target for all $\boldsymbol{x}$, and otherwise is less than 1. The acceptable region can be seen as a set of probabilities, p_k, such that $1 - \epsilon_t \leqslant \mathcal{F}(p_1, \ldots, p_{2^N-1})$, and thus higher $\mathcal{F}$ guarantees a wider acceptable region for a given tolerance, ϵ_t.

Let us begin with, as the simplest case, the target function f_1[5], a one-bit Boolean function, whose task fidelity, $\mathcal{F}(p_0, p_1)$, is reduced to

[5] This constant function, f_1, is a trivial function; however, it is a considerable task for the machines to learn f_1.

Table 2. The task fidelities of the quantum and classical machines are given in terms of the probabilities (p_0 and p_1) for each target function of the one-bit Boolean function. The phase Δ is defined in the main text, and it plays an important role in quantum machine learning.

Function	$\mathcal{F}_c(p_0, p_1)$	$\mathcal{F}_q(p_0, p_1)$
f_1	$\sqrt[4]{p_0(p_0p_1 + q_0q_1)}$	$\sqrt[4]{\mathcal{F}_c^4 + p_0p_{\text{int}}\cos\Delta}$
f_2	$\sqrt[4]{q_0(q_0p_1 + p_0q_1)}$	$\sqrt[4]{\mathcal{F}_c^4 - q_0p_{\text{int}}\cos\Delta}$
f_3	$\sqrt[4]{p_0(q_0p_1 + p_0q_1)}$	$\sqrt[4]{\mathcal{F}_c^4 - p_0p_{\text{int}}\cos\Delta}$
f_4	$\sqrt[4]{q_0(p_0p_1 + q_0q_1)}$	$\sqrt[4]{\mathcal{F}_c^4 + q_0p_{\text{int}}\cos\Delta}$

$$\mathcal{F}(p_0, p_1) = \sqrt[4]{P(0|0)P(0|1)}, \tag{8}$$

which is common to the classical and the quantum machines. For the classical machine, equation (8) is evaluated as

$$\mathcal{F}_c(p_0, p_1) = \sqrt[4]{p_0\left(p_0p_1 + q_0q_1\right)}, \tag{9}$$

adopting the conditional probabilities $P_c(y|\boldsymbol{x})$ given by

$$P_c(0|0) = p_0p_1 + p_0q_1 = p_0, \quad P_c(0|1) = p_0p_1 + q_0q_1, \tag{10}$$

where $q_j = 1 - p_j$ (j = 0,1). For the quantum machine, the conditional probabilities $P_q(y|\boldsymbol{x})$ differ slightly from $P_c(y|\boldsymbol{x})$ due to the superposition of $\hat{G}_0$ and $\hat{G}_1$. The conditional probabilities $P_q(y|\boldsymbol{x})$ are given as

$$\begin{aligned} P_q(0|0) &= \left|\langle 0|\hat{G}_0|0\rangle\right|^2 = P_c(0|0), \\ P_q(0|1) &= \left|\langle 0|\hat{G}_1\hat{G}_0|0\rangle\right|^2 = P_c(0|1) + p_{\text{int}}\cos\Delta, \end{aligned} \tag{11}$$

where $p_{\text{int}} = 2\sqrt{p_0p_1q_0q_1}$, and $\Delta = \phi_1 - \phi_0$ is the difference of the phases of the two unitaries $\hat{G}_0$ and $\hat{G}_1$. Thus, the task fidelity $\mathcal{F}_q$ of the quantum machine is evaluated as

$$\mathcal{F}_q(p_0, p_1) = \sqrt[4]{\mathcal{F}_c^4 + p_0p_{\text{int}}\cos\Delta}, \tag{12}$$

where the additional term $\cos\Delta$ is apparently the result of quantum superposition. From the result of equation (12), we can see that

$$\begin{cases} \mathcal{F}_q > \mathcal{F}_c & \text{if } \cos\Delta > 0, \\ \mathcal{F}_q < \mathcal{F}_c & \text{if } \cos\Delta < 0, \end{cases} \tag{13}$$

provided that $0 < p_j < 0$ (j = 0,1). The phase Δ plays an important role in helping the quantum machine via constructive interference, leading to $\mathcal{F}_q > \mathcal{F}_c$. The task fidelities for the other three targets are also listed in table 2. Note here that, for all cases of the target function f_j, $\mathcal{F}_q$ can always be larger than $\mathcal{F}_c$ on choosing appropriate free parameters ϕ_1 and ϕ_2 before the learning. Therefore, the quantum machine has wider acceptable regions than the classical machine for a

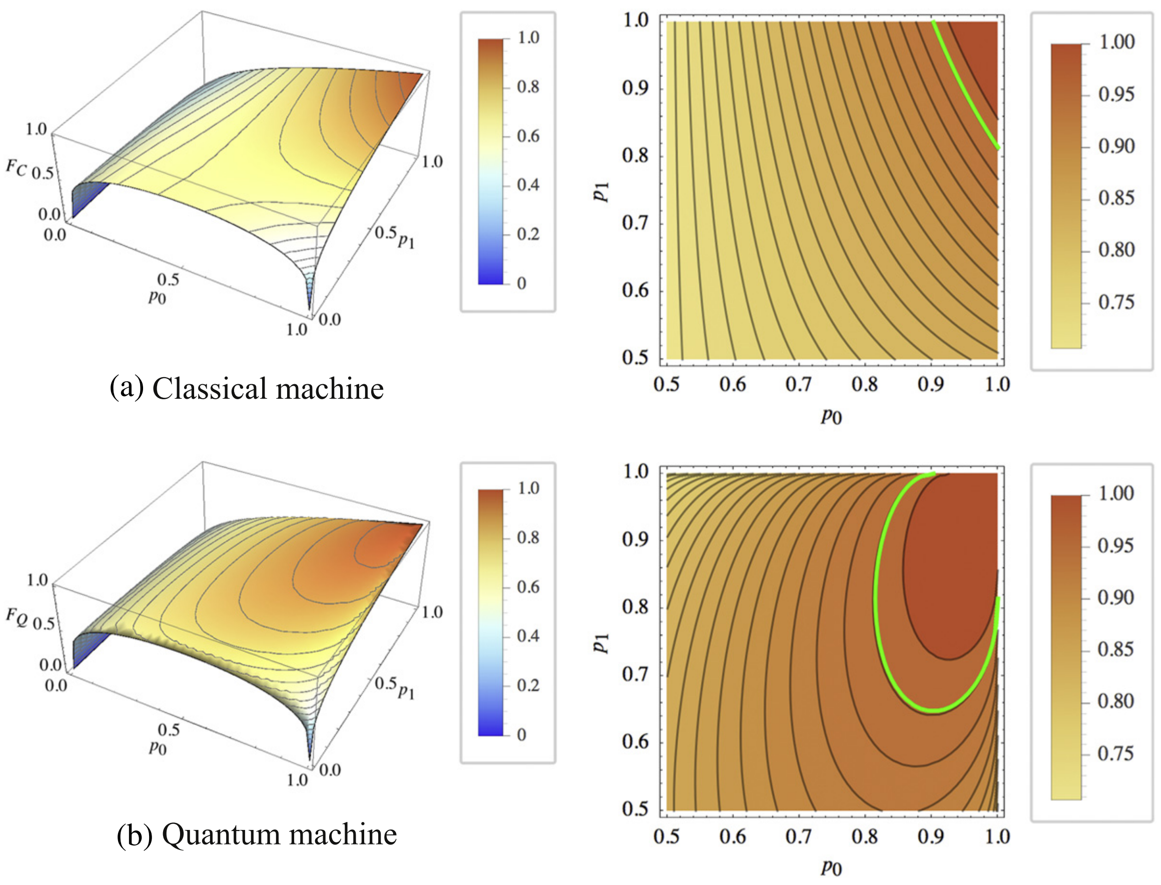

Figure 2. Left column: the task fidelities for classical and quantum machines. Right column: green lines in the magnified views indicate the acceptable regions for a given tolerable error $\epsilon_t = 0.05$ around the exact target point, $(p_0, p_1) = (1, 1)$. Here, we set $\Delta = 0$ to maximize the task fidelity of the quantum machine. It is found that the acceptable region of the quantum machine is about 5.6 times the size of that of the classical machine.

given tolerance. In figure 2, the task fidelity and the acceptable region for each machine are shown for the target f_1 when $\Delta = 0$ is chosen to maximize the difference between the two machines. We also found that the acceptable region of the quantum machine is about 5.6 times the size of that of the classical machine.

The optimal phase condition for improving the task fidelity, as in equation (13), can be generalized to an arbitrary N-bit Boolean function ($N > 1$). We provide one of the conditions as

$$\phi_k = \begin{cases} 0 & \text{if } s_k = 0 \\ \pi & \text{if } s_k = 1 \end{cases}. \tag{14}$$

where s_k is the kth component of a solution point $\mathrm{Q}_f(s_0, s_1, \ldots, s_{2^N-1})$ in the 2^N-dimensional search space (see appendix A). This condition yields $\mathcal{F}_\mathrm{q} \geqslant \mathcal{F}_\mathrm{c}$, so the acceptable region of the quantum machine can be wider than that of the classical machine for an arbitrary N-bit Boolean function.

Table 3. The learning time n_c is compared with the acceptable regions γ, and it is demonstrated that $n_c = \gamma^{-1}$. This implies that a larger acceptable region leads to a lesser learning time. Simulation failed for $N = 4$ and 5 in the classical case due to the finite computational resources and very long run time.

	Classical		Quantum	
N	γ^{-1}	n_c	γ^{-1}	n_c
1	1.0×10^2	1.03×10^2	1.8×10^1	1.74×10^1
2	1.4×10^4	1.39×10^4	2.6×10^1	2.68×10^1
3	4.4×10^8	4.67×10^8	5.5×10^1	5.36×10^1
4	9.8×10^{18}	—	3.5×10^2	3.48×10^2
5	7.1×10^{41}	—	2.5×10^4	2.48×10^4

4. Learning speedup via an expanded acceptable region

This section is devoted to the learning time in machine learning. For a numerical simulation, we employ random search as a feedback; this has often been considered for studying learning performance, rather than for any practical reasons [24]. Random search runs as follows. First, all 2^N control parameters p_k are randomly chosen, and then, the task fidelity is measured with the chosen p_k parameters. These two steps are thought of as a single iteration of the procedure. The iterations are repeated until the condition $\mathcal{F} \geqslant 1 - \epsilon_t$ is satisfied for a given ϵ_t. After a sufficient number of simulations have been performed, we then calculate the mean iteration number defined as $n_c = \sum nP(n)$, where $P(n)$ is the probability of completing learning at the nth iteration. This mean iteration number, n_c, can be used to quantify the learning time, and the results of numerical simulations for n_c are shown in table 3, where quantum learning is demonstrated to be faster than classical learning. This is a direct result of the wider acceptable region of the quantum machine, as n_c is inversely proportional to the size of the acceptable region in random search; $n_c = 1/\gamma$ is given by substituting in $P(n) = \gamma(1-\gamma)^{(n-1)}$, where γ is equal to the ratio of the acceptable region to the whole space in random search. We demonstrate this by comparing the results for n_c with the acceptable regions γ found from Monte Carlo simulation, given in table 3, and thereby we note that the acceptable region is the main feature directly influencing the learning time in random search.

Also in figure 3, the data for n_c in table 3 are well fitted to a function $\ln n_c = \alpha D + \beta$, implying that the size of the acceptable region is exponentially decreased as the dimension $D = 2^N$ of the parameter space increases, i.e. $n_c = \mathrm{O}(e^{\alpha D})$ [32]. The fitting parameters are given as

$$\begin{cases} \alpha \simeq 3.065 \pm 0.072, \ \beta \simeq -3.188 \pm 1.196 & \text{in the classical case,} \\ \alpha \simeq 0.238 \pm 0.008, \ \beta \simeq 2.267 \pm 0.127 & \text{in the quantum case.} \end{cases} \tag{15}$$

It is remarkable that the exponent α in the quantum case is much smaller than that in the classical case.

It follows from what has been shown that the acceptable region is the main feature directly influencing the learning time in random search. We have proved that we can always prepare a quantum machine which has an acceptable region larger than that of the classical one, in the

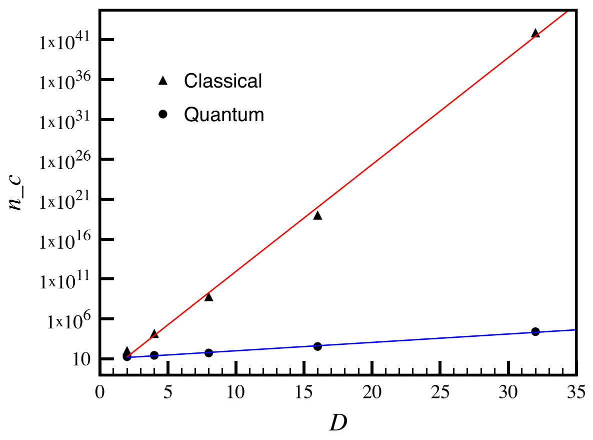

Figure 3. The learning time, n_c, with the dimension $D = 2^N$ of the parameter space for 1000 realizations. In this work, we consider a constant target function that yields 0 for all inputs $\boldsymbol{x}$, the optimal phase condition of equation (14) is chosen for the quantum machine, and the tolerable error ϵ_t is set as 0.05. The data are well fitted to $\ln n_c = \alpha D + \beta$ in the classical (red line) and quantum (blue line) cases, with the fitting parameters α and β as in equation (15).

previous section. Therefore, we finally conclude that the learning time can be shorter in the quantum case than in the classical case. The results of numerical simulation also support the assertion that the quantum machine learns much faster, particularly in a large search space. We clarify again that such a quantum speedup is enabled by the quantum superposition, and appropriately arranged phases.

5. Applying differential evolution

We consider a more practical learning model, taking into account real circumstances. A general analysis of the learning efficiency is very complicated, as so many factors are associated with the learning behavior. Furthermore, the most efficient learning algorithms tend to use heuristic rules and are problem-specific [33, 34]. Nevertheless, it is usually believed that the acceptable region is a key factor for the efficiency of learning in a heuristic manner [32]. We conjecture, in this sense, that the quantum machine offers the quantum speedup even in a practical learning method.

We apply differential evolution (DE), which is known as one of the most efficient learning methods for global optimization [30]. We start with M sets of control parameter vectors $\boldsymbol{p}_i = (p_0, p_1, ..., p_{2^N-1})_i$, for $i = 1, 2, ..., M$, whose components are the control parameters of the machine. In DE, these vectors, $\boldsymbol{p}_i$, are supposed to evolve by 'mating' their components p_k with each other. Equation (6) is used as a criterion for how well machines with $\boldsymbol{p}_i$ fit to the target. This process is iterated until the task fidelity reaches a certain level of accuracy $1 - \epsilon_t$ (see [30] or [20] for the detailed method of effecting differential evolution).

We perform the numerical simulations by increasing N from 1 to 7. The results are averaged over 1000 realizations for $M = 50$ and $\epsilon_t = 0.05$. The target function is a constant function: $f(\boldsymbol{x}) = 0$ for all $\boldsymbol{x}$. Free parameters in differential evolution (e.g., the crossover rate

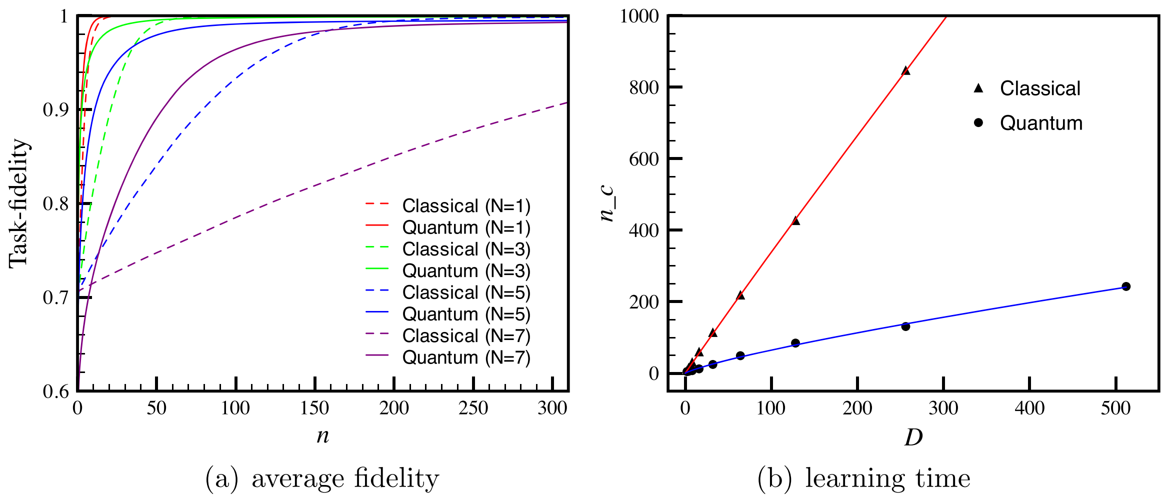

Figure 4. (a) The mean task fidelity is given with respect to the iteration n. The simulations are done increasing N from 1 to 7. It is readily observed that the increments of the task fidelities are faster in the quantum situation for all cases. (b) The learning time, n_{c}, as the dimension D of the parameter space increases is shown. The data are well fitted to a presumable function $n_{\mathrm{c}} \simeq \alpha D^{\beta}$, with $\alpha \simeq 3.82$, $\beta \simeq 0.97$ in the classical case (red line), and $\alpha \simeq 1.61$, $\beta \simeq 0.80$ in the quantum case (blue line). Note that the quantum machine still shows better convergence with the smaller α and β.

and differential weight) are chosen to achieve the best learning efficiency for a classical machine[6]. Nevertheless, we expect the quantum machine to still exhibit the quantum speedup, assisted by the quantum superposition, with the optimal phases in equation (14). We give the mean task fidelity averaged over M in figure 4(a). For both classical and quantum cases, the mean task fidelities are increased close to 1, but the quantum machine is much faster for all cases. We investigate the learning time n_c as we increase the dimension $D = 2^N$ of the parameter space, as depicted in figure 4(b). The data are well fitted to a presumable function $n_{\mathrm{c}} \simeq \alpha D^{\beta}$, with $\alpha \simeq 3.82$, $\beta \simeq 0.97$ for the classical machine, and $\alpha \simeq 1.61$, $\beta \simeq 0.80$ for the quantum machine[7]. We note that the quantum machine still exhibits the speedup with the smaller α and β. Therefore, we expect such quantum speedup to be achievable even in real circumstances.

6. Summary and discussion

We investigated the learning performances of two machines by considering the task of finding an N-bit Boolean function which can be used in a binary classification problem. The two machines were designed equivalently to make the comparison of these two machines as convincing as possible. The critical difference between the two machines was that the

[6] One may worry about the crossover point (for $N \geqslant 5$) in figure 4(a), associated with the validity of the quantum learning speedup for $\epsilon_t \to 0$. However, the appearance of the crossover is due to the DE optimization with the free parameters. Note here that the free parameters are optimized for the classical machine. The crossover can be removed by choosing the appropriate free parameters for each machine.

[7] Such a polynomial result shows much improvement from the differential evolution one—but this is quite distinct from the case of random search, which exhibits exponential dependence.

operations in the quantum machine are described by unitary operators, to deal with the quantum superposition. The learning processes of the two machines were characterized in terms of the acceptable region: the localized region of the parameter space including approximate solutions. We have found that the quantum machine has a wider acceptable region, induced by quantum superposition. We demonstrated a simulation with a standard feedback method, namely random search, to show that the sizes of the acceptable regions were inversely proportional to the learning time. Here, it was also shown that a wider acceptable region makes the learning faster; that is, the learning time is proportional to $\mathrm{O}(\mathrm{e}^{\alpha D})$, with $\alpha \simeq 3.065$ in the classical learning case and $\alpha \simeq 0.238$ for the quantum machine. We then applied a practical learning method, namely differential evolution, to our main task, and observed the learning speedup of the quantum machine.

Here, we wish to recall that the maximized learning speedup of the quantum machine is achieved by choosing suitable phases as in equation (14). From a practical perspective, one may consider that an additional task, such as finding the relative phases, is required to ensure the remarkable performance of the quantum learning machine for other N-bit Boolean function targets. Alternatively, such an issue could be resolved by synchronizing the relative phases with the control parameters in the quantum machine, still yielding the learning speedup (see appendix B for details).

We expect our work to motivate researchers to study the role of various quantum effects in machine learning, and to open up new possibilities for improving the machine learning performance. It is still open whether the quantum machine can be improved more by using other quantum effects, such as quantum entanglement.

Acknowledgments

We acknowledge the financial support of National Research Foundation of Korea (NRF) grants funded by the Ministry of Science, ICT & Future Planning (No. 2010–0018295 and No. 2010–0015059). We also thank T Ralph, M Żukowski and H J Briegel for discussions and advice.

Appendix A. Finding the optimal phase condition in equation (14)

Let us recall the general form of the task fidelity, as in equation (6). We suppose the target to be a deterministic function. Then, equation (6) is rewritten as

$$\mathcal{F}\left(p_0, p_1, \ldots, p_{2^N-1}\right) = \left(\prod_{\boldsymbol{x}} P\left(f\left(\boldsymbol{x}\right)|\boldsymbol{x}\right) \right)^{\frac{1}{2^{N+1}}}. \tag{A.1}$$

In deriving the above reduced form of equation (A.1), we used that $P_\tau\left(y|\boldsymbol{x}\right) = 1$ when y is equal to the desired value $f\left(\boldsymbol{x}\right)$ for a given target f, and otherwise $P_\tau\left(y|\boldsymbol{x}\right) = 0$. equation (A.1) shows that the task fidelity is enlarged if $P\left(f\left(\boldsymbol{x}\right)|\boldsymbol{x}\right)$ is maximized for all $\boldsymbol{x} \neq 0$.

To start, consider an ideal learning machine (either classical or quantum) that always generates the desired outcome results with perfect task fidelity, $\mathcal{F} = 1$. From our analysis in section 3, we can represent this machine as a point $\mathrm{S} = (s_0, s_1, \ldots, s_{2^N-1})$ in the 2^N-dimensional search space. In this sense, we consider this ideal machine the 'solution machine'. We then consider a 'near-solution machine' which is located at a point $\mathrm{Q} = (p_0, p_1, \ldots, p_{2^N-1})$ in the search space. More specifically, $d(\mathrm{Q}, \mathrm{S}) = \sqrt{\sum_{k=0}^{2^N-1}\left(s_k - p_k\right)^2} = \delta$, where $d(\mathrm{Q}, \mathrm{S})$ is the Euclidean distance. Here we assume further that the search space is isotropic around S, so the machines on the surface of the hypersphere $d(\mathrm{Q}, \mathrm{S}) = \delta$ have the same task fidelity. This assumption is physically reasonable for very small tolerance error. Thus, without loss of generality, we consider the near-solution machine corresponding to the point Q on the sphere $d(\mathrm{Q}, \mathrm{S}) = \delta$, satisfying $|s_k - p_k| = c$ for all k. Here, $c = \sqrt{\delta/2^N}$.

In these circumstances, $P(f(\boldsymbol{x})|\boldsymbol{x})$ for a classical near-solution machine is necessarily smaller than 1, depending on δ. On the other hand, if we choose the optimal phases ϕ_k, then $P(f(\boldsymbol{x})|\boldsymbol{x})$ can be 1 without any dependence on δ in the quantum machine. To show this, let us first write the conditional probability $P(f(\boldsymbol{x})|\boldsymbol{x})$ in equation (A.1) as

$$P(f(\boldsymbol{x})|\boldsymbol{x}) = \left| \langle f(\boldsymbol{x})| \left(\prod_{k \in A_{\boldsymbol{x}}} \hat{G}_k \right) |0\rangle \right|^2, \tag{A.2}$$

where $A_{\boldsymbol{x}}$ is the index set whose elements are indices of the actually applied operators conditioned on the input $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$. For example, if $\boldsymbol{x} = 1$ (i.e. $\{1, 0, 0 \ldots, 0\}$ in the binary representation), then we have $A_{\boldsymbol{x}} = \{0, 1\}$ because G_0 is always applied independently of the input, and the input signal $x_1 = 1$ activates G_1 (see figure 1). Thus, $\prod_{k \in A_1} \hat{G}_k = \hat{G}_1 \hat{G}_0$. On the basis of the above description, we can generalize the calculations as

$$\begin{cases} \prod_{k \in A_1} \hat{G}_k = \hat{G}_1 \hat{G}_0 & \text{for } \boldsymbol{x} = 1, \\ \prod_{k \in A_2} \hat{G}_k = \hat{G}_2 \hat{G}_0 & \text{for } \boldsymbol{x} = 2, \\ \prod_{k \in A_3} \hat{G}_k = \hat{G}_3 \hat{G}_2 \hat{G}_1 \hat{G}_0 & \text{for } \boldsymbol{x} = 3, \\ \quad \vdots \end{cases} \tag{A.3}$$

Here, equation (A.2) becomes 1 when $c = 0$ or equivalently $d(\mathrm{Q}, \mathrm{S}) = 0$, because it is nothing but the solution machine. The basic idea is to find a condition for which all terms of c vanish even though c is nonzero, i.e. the near-solution machine. Therefore, $P(f(\boldsymbol{x})|\boldsymbol{x})$ for the near-solution machine is mathematically equal to that of the solution machine. To carry out the task, we consider the product of two arbitrary unitaries $\hat{G}_k \hat{G}_l$ $(k \neq l)$, as

$$\begin{pmatrix} \sqrt{p_k} & \mathrm{e}^{\mathrm{i}\phi_k}\sqrt{q_k} \\ \mathrm{e}^{-\mathrm{i}\phi_k}\sqrt{q_k} & -\sqrt{p_k} \end{pmatrix} \begin{pmatrix} \sqrt{p_l} & \mathrm{e}^{\mathrm{i}\phi_l}\sqrt{q_l} \\ \mathrm{e}^{-\mathrm{i}\phi_l}\sqrt{q_l} & -\sqrt{p_l} \end{pmatrix}. \tag{A.4}$$

If we consider the near-solution machine, we can let $p_{k(l)} = |s_{k(l)} - c|$ and $q_{k(l)} = 1 - p_{k(l)}$. We then calculate $\hat{G}_k \hat{G}_l$, for the given s_k, s_l in S, as

$$\hat{G}_k \hat{G}_l = \begin{pmatrix} \mathrm{e}^{\mathrm{i}\Delta}\left(1 + \mathrm{e}^{-\mathrm{i}\Delta} c \Lambda_-\right) & g(c)\mathrm{e}^{\mathrm{i}\phi_l}\Lambda_- \\ g(c)\mathrm{e}^{-\mathrm{i}\phi_k}\Lambda_- & \mathrm{e}^{-\mathrm{i}\Delta}\left(1 - c\Lambda_-\right) \end{pmatrix} \quad \text{if } s_k = 0,\ s_l = 0,$$

$$\hat{G}_k\hat{G}_l = \begin{pmatrix} g(c)\Lambda_+ & e^{i\phi_l}\left(1 - c\Lambda_+\right) \\ -e^{-i\phi_l}\left(1 + ce^{-i\Delta}\Lambda_+\right) & g(c)e^{-i\Delta}\Lambda_+ \end{pmatrix} \text{if } s_k = 1,\ s_l = 0,$$

$$\hat{G}_k\hat{G}_l = \begin{pmatrix} g(c)\Lambda_+ & -e^{i\phi_k}\left(1 - ce^{-i\Delta}\Lambda_+\right) \\ e^{-i\phi_k}\left(1 - c\Lambda_+\right) & g(c)e^{-i\Delta}\Lambda_+ \end{pmatrix} \text{if } s_k = 0,\ s_l = 1,$$

$$\hat{G}_k\hat{G}_l = \begin{pmatrix} 1 - c\Lambda_- & -g(c)e^{i\phi_k}\Lambda_- \\ g(c)e^{-i\phi_k}\Lambda_- & 1 - ce^{i\Delta}\Lambda_- \end{pmatrix} \quad \text{if } s_k = 1,\ s_l = 1, \tag{A.5}$$

where $\Lambda_\pm = 1 \pm e^{i\Delta}$, $\Delta = \phi_k - \phi_l$, and $g(c) = \sqrt{c - c^2}$. In calculating equation (A.5), we assumed a deterministic target, i.e. $s_{k(l)}$ is to be either 0 or 1, as is usual in most tasks (but not necessarily the case). Here, the important thing is that we can cause the term associated with c to vanish, by letting

$$\Lambda_\pm = 0, \text{ or equivalently, } \begin{cases} \phi_k = \phi_l & \text{if } s_l = s_k, \\ \phi_k = \phi_l + \pi & \text{if } s_l \neq s_k. \end{cases} \tag{A.6}$$

The above condition in equation (A.6) can be applied for all $k \neq l$. Thus, we provide here a generalized condition:

$$\phi_k = \begin{cases} 0 & \text{if } s_k = 0, \\ \pi & \text{if } s_k = 1. \end{cases} \tag{A.7}$$

This is the optimal phase condition, as in (14). We can check that this condition gives the maximum task fidelity with $P(f(\boldsymbol{x})|\boldsymbol{x}) = 1$ (for all $\boldsymbol{x} \neq 0$).

Appendix B. A practical version of a quantum machine

The speedup introduced in this paper is enabled when a quantum machine uses suitable phases. Accordingly, the suitable phases are prerequisites for fast learning. In a practical case, the learning time has to include complexity to obtain suitable phases, and this is not very easy to achieve. We introduce a practical quantum machine that does not require the effort of finding optimal phases. To this end, we modify the unitary $\hat{G}_k$ in equation (5) by setting all the phases ϕ_k as πp_k, i.e., $\hat{G}_k$ is written as

$$\hat{G}_k = \begin{pmatrix} \sqrt{p_k} & e^{i\pi p_k}\sqrt{1 - p_k} \\ e^{-i\pi p_k}\sqrt{1 - p_k} & -\sqrt{p_k} \end{pmatrix}, \tag{B.1}$$

such that the phases πp_k are getting closer to the optimized phases πs_k as the machine approaches the solution point in the parameter space during the learning, since the optimized phase condition is given by equation (14). Thus, this guarantees wider acceptable regions than for the classical machine for any learning target.

Figure B1 (a) shows that the practical quantum machine has wider acceptable regions than the classical machine for all one-bit Boolean targets. The areas inside the solid and dashed lines represent the acceptable regions for the practical quantum machine and the classical machine,

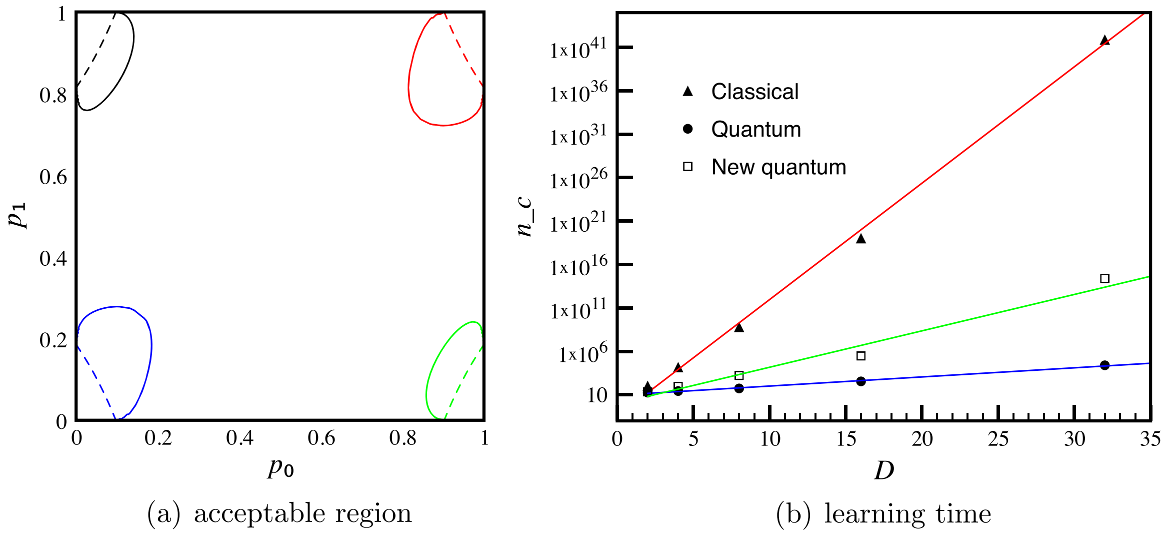

Figure B1. (a) We depict the boundaries of the acceptable regions for the one-bit learning machine with respect to all possible target functions, as in table 1: f_1 (red), f_2 (black), f_3 (green) and f_4 (blue). We set $\epsilon_t = 0.05$. It is directly seen that the quantum case (solid line) has larger acceptable regions than the classical case (dotted line), for all cases. (b) The learning time, n_c, with the dimension D of the parameter space. The data for classical (red) and quantum (blue) machines are exactly the same as for figure 3. The data for the new quantum machine (green line) are well fitted to $\ln n_c = \alpha D + \beta$, with the fitting parameters $\alpha \simeq 0.985$ and $\beta \simeq -0.200$.

respectively. This supports the assertion that the practical quantum machine always learns faster than the classical machine, while the performance of the original quantum machine depends on the target function.

We then obtained the learning time of the practical quantum machine, which is shown in figure B1(b). These data are also well fitted to $\ln n_c = \alpha D + \beta$, with the fitting parameters $\alpha \simeq 0.985 \pm 0.101$ and $\beta \simeq -0.200 \pm 1.662$. Thus, $n_c \sim \mathrm{O}(\mathrm{e}^{0.985D})$ for the practical quantum machine, whereas $n_c \simeq \mathrm{O}(\mathrm{e}^{3.065D})$ for the classical machine (see equation (15)). This result shows that a considerable learning speedup is still achieved with this practical quantum machine, even though it takes up a little more time as compared to an original machine available with the optimal relative phases ($n_c \sim \mathrm{O}(\mathrm{e}^{0.238D})$).

References

[1] Deutsch D and Jozsa R 1992 *Proc. R. Soc.* A **439** 553
[2] Grover L K Jul 1997 *Phys. Rev. Lett.* **79** 325
[3] Shor P W 1997 *Siam J. Comput.* **26** 1484
[4] Giovannetti V, Lloyd S and Maccone L 2004 *Science* **306** 1330
[5] Giovannetti V, Lloyd S and Maccone L 2011 *Nat. Photonics* **5** 222
[6] Bennett C H and Brassard G 1984 *Proc. IEEE Int. Conf. on Computers Systems, and Signal Processing, Bangalore* p 175
[7] Ekert A K Aug 1991 *Phys. Rev. Lett.* **67** 661
[8] Langley P 1996 *Elements of Machine Learning* (San Francisco, CA: Morgan Kaufmann)
[9] Pearson B J, White J L, Weinacht T C and Bucksbaum P H 2001 *Phys. Rev.* A **63** 063412

[10] Gammelmark S and Molmer K 2009 *New J. Phys.* **11** 033017
[11] Bang J, Lee S W, Jeong H and Lee J 2012 *Phys. Rev.* A **86** 062317
[12] Briegel H J and De las Cuevas G 2012 *Sci. Rep.* **2** 400
[13] Lewenstein M 1994 *J. Mod. Opt.* **41** 2491
[14] Kak S C 1995 *Inform. Sciences* **83** 143
[15] Chrisley R L 1997 *Brain, Mind and Physics* ed P Pylkkänen and P Pylkkö (Amsterdam: IOS Press) pp 126–39
[16] Narayanan A and Menneer T 2000 *Inform. Sciences* **128** 231
[17] Han K H and Dec Kim J H 2002 *IEEE T. Evolut. Comput* **6** 580
[18] Han K H and Kim J H 2006 *IEEE Int. Conf. on Evolutionary Computation (IEEE)* pp 2622–9
[19] Manzano D, Pawłowski M and Brukner C 2009 *New J. Phys.* **11** 113018
[20] Bang J, Ryu J, Yoo S, Pawłowski M and Lee J 2014 *New J. Phys.* **16** 073017
[21] Lloyd S, Mohseni M and Rebentrost P 2013 arXiv:1307.0411
[22] Rebentrost P, Mohseni M and Lloyd S 2013 arXiv:1307.0471
[23] Opper M and Haussler D 1991 *Phys. Rev. Lett.* **66** 2677
[24] Rastrigin L A 1963 *Automat. Rem. Contr.* **24** 1337
[25] Gupta P, Agrawal A and Jha N K 2006 *IEEE T. Comput. Aid.* D **25** 2317
[26] Maslov D and Mar Dueck G W 2003 *6th Int. Symp. on Representation and Methodology of Future Computing Technologies* pp 162–70
[27] Toffoli T 1980 Reversible Computing *Technical Memo MIT/LCS/TM-151 (MIT Lab for Computer Science)*
[28] Reck M, Zeilinger A, Bernstein H J and Bertani P 1994 *Phys. Rev. Lett.* **73** 58
[29] Kim J, Lee J S and Feb Lee S 2000 *Phys. Rev.* A **61** 032312
[30] Storn R and Price K 1997 *J. Global. Optim.* **11** 341
[31] Nielsen M A and Chuang I L 2010 *Quantum Computation and Quantum Information* 10th edn (Cambridge: Cambridge University Press)
[32] van den Bergh F and Engelbrecht A P 2004 *IEEE T. Evolut. Comput.* **8** 225
[33] Middleton A A 2004 *Phys. Rev.* E **69** 055701
[34] Pál K F 1996 *Physica* A **233** 60