Grover's Algorithm and the Evolutionary Approach of Quantum Computation

Mihai Udrescu[1], Lucian Prodan[1], Mircea Vlăduțiu[1]

[1] Advanced Computer Systems & Architectures Laboratory, Computer Science Department, "Politehnica" University , 2 Parvan Blvd, 300223 Timişoara, Romania
{mudrescu, lprodan, mvlad}@cs.utt.ro
http://www.acsa.utt.ro

Abstract. In the recent years we were witnesses of an intense research activity, trying to find some common ground for quantum computation and genetic programming [4][19]. The use of genetic algorithms provides outstanding means for automated synthesis of quantum circuits. In this paper we focus on the opportunity of designing so-called *Quantum Genetic Algorithms* or QGAs. As the qubit representation of the chromosomes provides the possibility of representing the entire population with just one quantum superposition state, a quantum state can be generated as a superposition of all the corresponding fitness values. By marking the superposed chromosome that dictates the best fitness value, one only needs to apply Grover's algorithm in order to return the solution of the search. We also present the implementation details of the proposed procedure, along with an illustrating example.

1 Introduction

By clearly identifying its most major problems and limitations, computer science has become mature [10]. The research community has put a lot of effort in the attempt to solve these problems and further pushing the computing frontiers; however, by using the means of what is now called *classical computation*, it seems that one can hardly expect more than marginal improvements, even with sophisticated approaches.

In this context, inspiration was mainly found in biology and physics: bio-inspired computing and quantum computing are considered as possible solutions. The optimism is fed by theoretical and practical achievements. Genetic algorithms and evolvable hardware are already successfully used in a wide range of applications, spanning from image compression, robotics and other artificial intelligence related issues, to engineering problems like fault tolerance and reliability in critical environments. Moreover, quantum computing seems to draw even more power from its exponential parallelism: Peter Shor has proven that a classical exponential problem (integer factorization) can be solved in polynomial time [16].

The above considerations indicate that the merge between the two novel computing promises, namely genetic algorithms (GAs) and quantum computing (QC) would be natural and benefic [19]. Researchers already follow the path of so-called Quantum Evolutionary Programming (QEP) [4] with outstanding results [17]. For instance, the best approach for automated synthesis of quantum circuits uses genetic programming [8]. Also, quantum algorithm design can be approached by evolutionary means [18]. In fact, the majority of such applications are addressing quantum computation design issues regarding quantum algorithms and implementations [17]; they are all part of QEP's sub-area called Quantum Inspired Genetic Algorithms (QIGAs) [4][9]. The other sub-area, called Quantum Genetic Algorithms (QGAs), tries to implement genetic algorithms in a quantum computation environment [4][14][15]. This paper proposes a new perspective on QGAs, by showing that no genetic algorithm strategy is necessary in quantum computation, because it can be reduced to Grover's algorithm [5].

1.1 Motivation

The QGAs rely on qubit representations for the chromosomes and the use of quantum operators in order to process them during the quest for the optimal solution of the search problem. In principle, this approach redefines the GA operators in quantum terms; these new operators will perform better due to the exploit of the quantum parallelism [15]. Nevertheless, approaching specific applications this way will result in a significant performance enhancement [6][7].

Because the chromosome represented by qubits, just one quantum chromosome register would be able to store the entire population as a superposition of all the possible classical states. The function that evaluates the fitness of the initial population (which could also be the entire population) would take the *chromosome register* as input and the output would be stored in a *fitness register*. This would store a superposition of all the fitness values, corresponding to the superposition of the individuals from the chromosome register.

The key observation that led us to this new perspective is the fact that if the best fitness value can be *marked* (i.e. by changing the phase of the corresponding eigenstate) without destroying the superposition of the registers, then Grover's algorithm will find the solution in $\mathcal{O}(\sqrt{n})$. Therefore, all the quantum versions of GA operators, like crossover or mutation, would become useless if we figured out a way to mark the best fitness, inside the fitness superposition state.

1.2 Quantum Computation Background

For quantum computing, the information storage unit is the quantum bit or *qubit*, which is presented here in *bra-ket* notation [10]. The qubit is a normalized vector in the $\mathcal{H}^2$ Hilbert space, with $\{ |0\rangle, |1\rangle \}$ as the orthonormal basis: $|\psi\rangle = a_0 |0\rangle + a_1 |1\rangle$. Here, a_0, $a_1 \in \mathbb{C}$ are the so-called quantum amplitudes, which represent the square root of the associated measurement probabilities for the superposed states $|0\rangle$ and $|1\rangle$ respectively, with $\|a_0\|^2 + \|a_1\|^2 = 1$. The qubits can be organized in linear structures called *quantum registers*, encoding a superposition of all possible classical states. For an *n*-

qubit quantum register, its corresponding state is a normalized vector in a 2^n-dimensional Hilbert space, $|\psi_r\rangle = \sum_{i=0}^{2^n-1} a_i |i\rangle$, where $\sum_{i=0}^{2^n-1} |a_i|^2 = 1$, $i \in \mathbb{N}$. When the individual qubit states are known, the tensor product provides the register overall state [10]. For instance, 2 qubits – $|\psi_A\rangle = a_0|0\rangle + a_1|1\rangle$ and $|\psi_B\rangle = a_2|0\rangle + a_3|1\rangle$ – give $|\psi_A\rangle \otimes |\psi_B\rangle = a_0a_2|00\rangle + a_0a_3|01\rangle + a_1a_2|10\rangle + a_1a_3|11\rangle$ as the overall state. A better handling of the quantum states is possible with the matrix representation. Thus, the 2-qubit tensor product from above will have the following matrix form: $\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} \otimes \begin{bmatrix} a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} a_0a_2 & a_0a_3 & a_1a_2 & a_1a_3 \end{bmatrix}^\dagger$.

• **Entanglement**
For a quantum register state, entanglement occurs iff it cannot be represented as a tensor product of its parts. In a 2-qubit example, we say that state $|\psi_1\rangle$ is not entangled while $|\psi_2\rangle$ is entangled, because $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) = |0\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, but there are no $|\phi_1\rangle$ and $|\phi_2\rangle$ qubits with $|\psi_2\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = |\phi_1\rangle \otimes |\phi_2\rangle$.

• **Quantum circuits**
The quantum circuits are constrained networks of gates, with no cloning and no feedback allowed [2]. The quantum gate is the physical device implementing a unitary operator [10], which represents the quantum state transform. Due to the unitary property, all quantum operators are reversible. In the circuit model of quantum computation, which is a rippling of consecutive quantum networks of gates and registers, the register measurement is the only non-unitary operation.

It was proven that the set of gates $\{XOR, \wedge_0(U)\}$ is universal in quantum computation [2]. The *n*+1 qubit transform $\wedge_n(U)$ is a conditional operator, applying 1-qubit unitary operator *U* on the target qubit iff the other *n* input qubits are '1'. If $U = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \sigma_x$, then the conditional transform only negates the target qubit for the other input bits being '1'. The *XOR* gate is $\wedge_1(\sigma_x)$, while the Toffoli gate (universal in classical reversible computation) is $\wedge_2(\sigma_x)$.

2 Quantum Genetic Algorithms

As part of Quantum Evolutionary Programming, QGAs have the ingredients of a substantial algorithmic speedup, due to the inherited properties from both QC and GA. However, there still are questions as to how would it be possible to implement a genetic algorithm on a quantum computer. The attempts made in this particular direction suggest there is room left for taking advantage from the massive quantum computation parallelism [15]. Moreover, some questions were left open, as pointed out in [4].

2.1 Running GAs in a Quantum Computational Environment

For the first time, the possibility (and the advantages) of the QGAs were indicated in [15]. The approach described here contains hard evidence for QGA speedup, but there still are some unanswered questions [4]. The proposed algorithm uses a number of *m* register pairs:

$$|\psi\rangle_i = |\phi\rangle_i^{individual} \otimes |\varphi\rangle_i^{fitness} \qquad (1)$$

where $i = \overline{0, m-1}$. The first (left) register contains the individual, while the second contains its corresponding fitness. Because we are dealing with quantum registers, both $|\phi\rangle$ and $|\varphi\rangle$ can encode a superposition of exponentially many individuals and their corresponding superposed fitness values. Each time a new population (set of individuals) is generated in the *individual* register, the corresponding fitness is computed and stored in the *fitness* register. Of course, if the fitness register is measured then, due to entanglement [10], the result is only one of the superposed values; in the individual register will remain superposed the individuals that give the measured fitness. Fitness register measurement is a crucial element in developing QGAs [15]. For the general expression of the pair register (*N*-qubit for the individual register and *M*-qubit for the fitness register) given in (2) the measurement of the second register ($|y\rangle$) will have *r* as result with probability (3).

$$|\psi\rangle_i = \sum_{x=0}^{2^N-1} \sum_{y=0}^{2^M-1} c_{x,y} |x, y\rangle, \text{ with } \sum_{x=0}^{2^N-1} \sum_{y=0}^{2^M-1} \|c_{x,y}\|^2 = 1 \qquad (2)$$

$$P(r) = \sum_{x=0}^{2^N-1} \|c_{x,r}\|^2 \qquad (3)$$

The post-measurement state of the pair register will be:

$$|\psi_r\rangle_i = \frac{1}{\sqrt{P(r)}} \sum_{x=0}^{2^N-1} c_{x,r} |x, r\rangle \qquad (4)$$

Due to the fact that an individual cannot have more than one fitness, it is obvious that, if individual *u* has a fitness value *v*, then $c_{u,y} = 0$ for all $y \neq v$.

The QGA, as described in [15], is presented in the following pseudo code:

Genetic Algorithm Running on a Quantum Computer (QGA)

```
1. For i:=1 to m prepare $|\phi\rangle_i^{individual}$ as superpositions of individuals and compute the corresponding fitness pair register $|\varphi\rangle_i^{fitness}$ (the outcome will be a set of m superpositions of fitness values).
2. Measure all fitness registers.
3. Repeat
      a. Selection according to the m measured fitness values.
      b. Crossover and mutation are employed in order to prepare a new population (setting the m individual registers).
```

```
        c. For the new population, the corresponding fit-
           ness values will be computed and then stored
           in the fitness registers.
        d. Measure all fitness registers
    Until the condition for termination is satisfied.
```

Reference [4] provides analysis and critique for the above presented algorithm. The identified advantages of using QGAs over the classical GAs, which are drawn from the quantum computational features, are :

– Due to the superposition of individuals (i.e. basis states) that is stored in the individual register, the building block [4] could be crossed not by just one individual, but by a superposition of exponentially many individuals. Thus, the selection of a new population is made with the contribution of many attraction pools.
– In quantum computation *true random numbers* can be generated. It was proven that a GA with a true random number generator will outperform a pseudo-random solution, which is the only possibility in classical computation [14].

The questions that remain open are:

– How is it possible to build the crossover operator in quantum computation?
– How is it possible to implement the fitness function on a quantum computer?
– How can the correlation be maintained – by using the entanglement – between the individual register (superposed) basis states and the (superposed) fitness values from the fitness register?

Although the advantages appear to be substantial, one can easily argue that the power of quantum computation is not sufficiently used by this approach. However, some of the opened questions have been addressed in reference [4]. Giraldi *et al.* developed a mathematical formalism in order to avoid misinterpretations regarding the last question. The second question is also addressed by defining quantum genetic operators. The proposed formalism establishes the necessary correlation between the fitness and the individual registers, which cannot be accomplished with the QGA construction provided in [15].

2.2 Mathematical Formalism

The QGA formalism uses m quantum register pairs (N-qubit individual register and M-qubit fitness register,) as presented in Section 2.1. Also, in order to achieve proper correlation between the individual and its fitness value, the fitness function must be chosen so that it is a "quantum function" as defined by [11], hence a pseudo-classical operator with a corresponding Boolean function: $f:\{0,1\}^N \rightarrow \{0,1\}^M$, $U_f : |x\rangle \otimes |0\rangle \rightarrow |x\rangle \otimes |f(x)\rangle$ if $|x\rangle$ is a basis state. When acting on a superposition, the unitary operator corresponding to function f will dictate the following mapping:

$$U_f : \sum_{x=0}^{2^N-1} a_x |x\rangle \otimes |0\rangle \rightarrow \sum_{x=0}^{2^N-1} a_x |x\rangle \otimes |f(x)\rangle = \sum_{x=0}^{2^N-1} a_x |x, f(x)\rangle \qquad (5)$$

An important aspect regarding the pseudo-classical Boolean functions is that they are universal (i.e. any computational function can be represented in such a form), and easy to be implemented as gate networks. In fact, due to their universality, Boolean functions form the backbone of the classical computation's *circuit model*.

The QGA algorithm, after adopting Giraldi's formalism can be rewritten as in the pseudo code below.

Genetic Algorithm Running on a Quantum Computer (QGA) – with proper formalism

```
1. For i:=1 to $m$ set the individual-fitness pair registers as $|\psi\rangle_i^1 = \frac{1}{\sqrt{n}}\sum_{u=0}^{n-1}|u\rangle_i^{ind}\otimes|0\rangle_i^{fit}$ (a superposition of $n$ individuals with $0 \le n \le 2^N$).
2. Compute the fitness values corresponding to the individual superposition, by applying a unitary transformation $U_{f_{\text{fit}}}$ (corresponding to pseudo-classical Boolean operator $f_{\text{fit}}:\{0,1\}^N \to \{0,1\}^M$). For i:=1 to $m$ do $|\psi\rangle_i^2 = U_{f_{\text{fit}}}|\psi\rangle_i^1 = \frac{1}{\sqrt{n}}\sum_{u=0}^{n-1}|u\rangle_i^{ind}\otimes|f_{\text{fit}}(u)\rangle_i^{fit}$
3. For i:=1 to $m$ measure the fitness registers, obtaining the post-measurement states (we suppose that $|y\rangle_i$ is measured): $|\psi\rangle_i^3 = \frac{1}{\sqrt{k_i}}\sum_{v\in\{0,1,\ldots,n-1\}}|v\rangle_i^{ind}\otimes|y\rangle_i^{fit}$ with $k_i$ values in $\{0,\ldots,n-1\}$ to satisfy $f_{\text{fit}}(v) = y$.
4. Repeat
    a. Selection according to the $m$ measured fitness values $|y\rangle_i$.
    b. Crossover and mutation are employed in order to prepare a new population (setting the $m$ individual registers $|u\rangle_i^{ind}$).
    c. For the new population, the corresponding fitness values will be computed and then stored in the fitness registers ($|f_{\text{fit}}(u)\rangle_i^{fit}$).
    d. Measure all fitness registers
   Until the condition for termination is satisfied.
```

Besides the necessary formalism, reference [4] also provides some insight regarding the implementation of the genetic operators in the quantum computational environment. These considerations lead towards two main implementation problems:

α) the number of all valid individuals is not always a power of 2, which is the total number of basis states;

β) crossover implementation is difficult and requires a much thoroughly investigation, including quantum computation architectural aspects [12].

3 A New Approach

An observation concerning the individual-fitness quantum register pair is that all the possible valid individuals (n) can be encoded in the same quantum state superposition, which has a total of 2^N possible basis states ($n \leq 2^N$). If we can figure out a method of measuring the highest fitness value from the fitness register, then by measuring the individual register we will get that corresponding individual (or one of them, if several have the same highest fitness value).

Approaching the QGAs in this manner renders genetic operators as no longer necessary, as long as finding the maximum has an efficient solution. This effectively leads to solving problem β).

Because the individual is encoded on N qubits, we have a total of 2^N basis states which can participate in the superposition. It is possible that not all of these basis states will encode valid individuals (problem α); the proposed method relies on defining some constrains regarding the fitness function and the fitness value format, without losing the generality of the solution. We will consider the fitness function as a Boolean pseudo-classical unitary operator U_f (characterized by $f:\{0,1\}^N \to \{0,1\}^M$) which can be also applied to non-valid individuals. The fitness value space $\{0,1\}^M$ can be split, so that a distinct subspace is allocated to the fitness values corresponding to valid individuals and another distinct subspace corresponds only to non-valid individuals. This enables us to concentrate only on processing states that correspond to valid individuals (Section 3.2 further elaborates on this particular aspect).

The method of finding the highest fitness value is inspired from efficient quantum algorithms for finding the maximum [1][3]. Finding the best fitness value is equivalent to marking the highest classical state that is superposed in the fitness register state or, in other words, the highest basis state with non-zero amplitude. Basically, the proposed methodology relies on reducing the highest fitness value problem to Grover's algorithm. In order to do so, special oracle and fitness value format are defined. Section 3.1 presents the quantum algorithm for finding the maximum [1], Section 3.2 presents details for oracle implementation and fitness register structure, while Section 3.3 provides our adaptation of the algorithm in order to find the best value in the fitness register.

3.1 Computing the Maximum

The minimum/maximum finding quantum algorithms [1][3] are inspired from the classical "bubble sort" algorithm, but their complexity in quantum version is $\mathcal{O}(\sqrt{n})$.

The quantum algorithm for finding the maximum takes an unsorted table of *m* elements as input, in order to return the index of the maximum value element. By adopting the formalism from [1], we have a pool $P[i]$ of *m* elements ($i = \overline{0, m-1}$) which will be processed in order to obtain the index k of the maximum element ($P[k]$). The Grover's algorithm can be used with a special oracle that "marks" all the basis states greater than some given value *j*:

$$O_j(i) = \begin{cases} 1 & \text{if } P[i] > P[j] \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Therefore, the resulted algorithm will have the form of the following pseudo code:

Quantum Algorithm for finding the maximum from an unsorted table of *m* elements

```
1. Initialize $k$:=$P[r]$; 0≤$r$≤$m$-1 and is randomly chosen
2. Repeat $\mathcal{O}(\sqrt{m})$ times
      a. Set two quantum registers as $|\psi\rangle = 1/\sqrt{m}\sum_{i=0}^{m-1}|i\rangle|k\rangle$
      b. Use Grover's algorithm for finding marked
         states from the first register (i.e. those
         which make $O_k(i)$=1)
      c. Measure the first register. The outcome will
         be one of the basis states that are > $k$. Let
         the measurement result be $x$. Make $k$:=$x$.
3. Return $k$ as result. It is the index of the maximum.
```

The complexity analysis performed in [1] reveals the fact that this algorithm will find the index of the maximum in $13.6\sqrt{m}$ steps, with an error rate smaller than $1/2$.

3.2 The Oracle

In order to deal with problem α) (see Section 2.2), we have to adopt a constraint, which does not restrict the generality of the fitness functions. If we consider the ordinary fitness function f_{fit} (which applies only on the valid individuals) $f_{\text{fit}} : \{0,1\}^N \to \{0,1\}^M$, it is Boolean (and therefore universal), with a straightforward correspondence to the unitary representation $U_{f_{\text{fit}}}$ [11]. The modified fitness function will accept invalid individuals as argument, and the returned values will belong to distinct areas, corresponding to valid or invalid individuals. This can be achieved by defining $f_{\text{fit}}^{\text{mod}} : \{0,1\}^N \to \{0,1\}^{M+1}$ as:

$$f_{\text{fit}}^{\text{mod}}(x) \in \begin{cases} 0 \times \{0,1\}^M & \text{if} \quad x \text{ is a non-valid individual} \\ 1 \times \{0,1\}^M & \text{if} \quad x \text{ is a valid individual} \end{cases} \tag{7}$$

The fitness values are encoded by the qubits in a modified fitness register, which has a (M+1)-qubit size. The valid individuals always produce fitness values with the most significant qubit being ‘1’; a ‘0’ value for the most significant qubit in the fitness register indicates the correspondence to a non-valid individual, as presented in Fig. 1 (quantum state matrix representation is used).

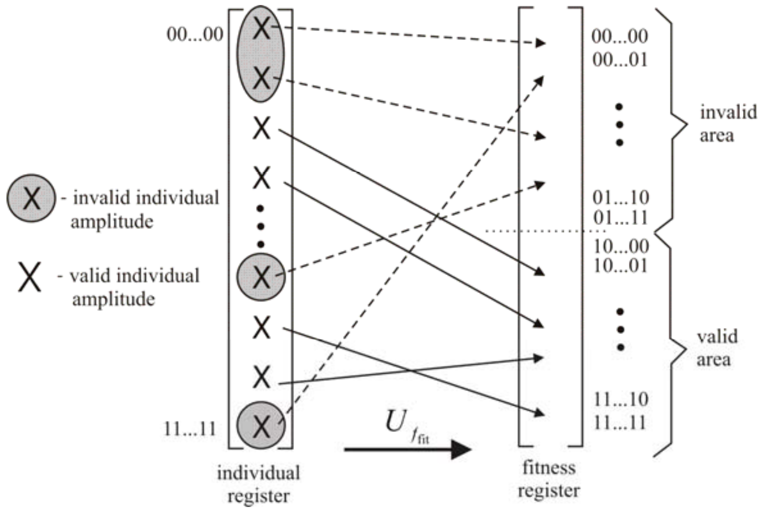

Fig. 1. Basic idea for fitness function construction: when is applied to valid individuals it produces a value in the valid area (upper half – $|10...00\rangle \div |11...11\rangle$) of the fitness register, whereas when applied to invalid individuals the corresponding values in the fitness register will always be in the invalid area (lower half – $|00...00\rangle \div |01...11\rangle$).

Another implementation-related problem concerns the oracle described by (6). We propose a solution that uses two’s complement number representation [13] for marking the states that have a value greater than a given $j \in \mathbb{N},\ j > 0$. As a consequence, the fitness register will have the form from Figure 2. The oracle processes all the fitness register qubits except the most significant (v), which indicates if the value represented by the other qubits belongs to a valid individual or not. All value qubits ($f_m \cdots f_0$) in the fitness register encode two’s complement positive integers as fitness values. The oracle adds $-(j+1)$ to the fitness register, therefore the basis states (from the state output by the quantum adder [20]) greater than j will always have $f'_M = 0$ (see the oracle implementation from Fig. 3.) For the solution in Fig. 3 we used 2 negation gates (denoted with ‘x’) and 1 *XOR* gate [2][10]. The architectures for the quantum arithmetic circuits, including the adder/subtractor, are presented in [20]. Only the qubits containing the result of the arithmetic function ($f'_0 \cdots f'_M$)

are used by the Grover iteration circuit [5][10] in order to find one of the marked basis states.

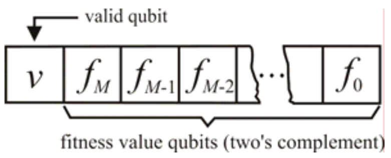

Fig. 2. The format of the fitness register, for the two's complement approach of oracle implementation.

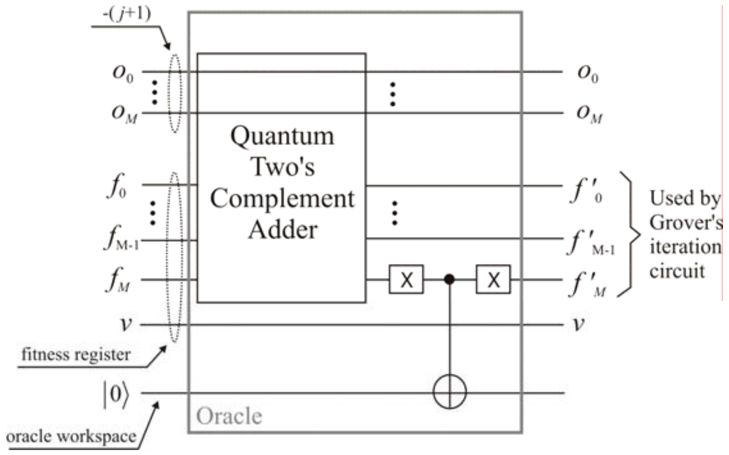

Fig. 3. Oracle implementation for a fitness register having the structure from Fig. 2.

Although the oracle uses two's complement addition (which means that we will have to change the fitness values in the superposition), the correlation between the individual and the fitness registers is not destroyed, because the addition is a pseudo-classical permutation function [11][20]. However, the Grover iteration will find as a marked basis state $|p\rangle = |f'_M \ldots f'_0\rangle$, with $p \in \mathbb{N}, f'_M, \ldots f'_0 \in \{0,1\}$, which is given by $|p\rangle = |q-(j+1)\rangle$, for $l \in \mathbb{N}$ and $|l\rangle = |f_M \ldots f_0\rangle$, with $f_M \ldots f_0 \in \{0,1\}$. This means that, after measuring $|p\rangle$, we have to add *j*+1 to this value in order to have the correct desired basis state (>*j*).

3.3 Reduced Quantum Genetic Algorithm

Having a fitness register as defined in the previous subsection, the corresponding fitness function, and the specially defined oracle, we are able to provide the pseudo-code that corresponds to running a Genetic Algorithm in the quantum computational environment. It is called *reduced Quantum Genetic Algorithm* (rQGA) because it uses only one population (encoded in just one quantum state), consisting of all possible individual binary representations (that correspond to valid and invalid individuals).

Crossover and mutation operators are not used for finding the highest fitness value (they are not required in a quantum context), which is obtained by employing Grover's algorithm.

The algorithm listed below is inspired from the quantum maximum algorithm from Section 3.1. The initial *max* value must obey the $2^{M+1} \le \max \le 2^{M+2} - 1$ relation, so that the search for the highest fitness value will take place only in the valid fitness area. We have a number of $m \in \mathcal{O}(\sqrt{N})$ pair registers (individual-fitness), where the individual register is on *N* qubits, and the fitness register on *M*+2 qubits.

Reduced Quantum Genetic Algorithm

```
1. For i:= 0 to m-1 set the pair registers as
   $|\psi\rangle_i^1 = \frac{1}{\sqrt{2^N}} \sum_{u=0}^{2^N-1} |u\rangle_i^{ind} \otimes |0\rangle_i^{fit}$
2. For i:= 0 to m-1 compute the unitary operation cor-
   responding to fitness calculation
   $|\psi\rangle_i^2 = U_{f_{\mathrm{fit}}} |\psi\rangle_i^1 = \frac{1}{\sqrt{2^N}} \sum_{u=0}^{2^N-1} |u\rangle_i^{ind} \otimes | f_{\mathrm{fit}}(u)\rangle_i^{fit}$
3. max:= random integer, so that $2^{M+1} \le \max \le 2^{M+2}$-1
4. For i:=0 to m-1 loop
      a. Apply the oracle. Therefore, if
         $| f_{\mathrm{fit}}(u)\rangle_i^{fit} > \max$ then corresponding $| f_{\mathrm{fit}}(u) - \max\rangle_i^{fit}$
         basis states are marked
      b. Use Grover's algorithm for finding marked
         states in the fitness register after applying
         the oracle. We find one of the marked basis
         states $|p\rangle = | f_{\mathrm{fit}}(u) - \max\rangle_i^{fit}$, with $f_{\mathrm{fit}}(u) - \max \ge 0$
      c. max:=p+max+1
5. Having the highest fitness value in the $|\bullet\rangle_{m-1}^{fit}$ regis-
   ter, we measure the $|\bullet\rangle_{m-1}^{ind}$ register in order to obtain
   the corresponding individual (or one of the corre-
   sponding individuals)
```

4 Conclusions

This paper described a methodology for running Genetic Algorithms on a Quantum Computer. By taking advantage of the quantum computation features, all the possible chromosome binary representations can be encoded in just one individual quantum register. This register is correlated with its pair (fitness) register, which contains a superposition of all corresponding fitness values. Due to quantum mechanical properties, measuring the highest fitness value in the fitness register, leads to a post-measurement state of the corresponding individual register that contains superposed basis state(s) encoding the individual(s) with the highest fitness.

Therefore, the initial problem is reduced to finding the best fitness value without destroying the individual-fitness register correlation. This objective is achieved by adapting an existing quantum algorithm for finding the maximum. Without loosing the generality of the solution, the adaptation requires that a specific structure be adopted for the fitness register, and a special oracle be defined by employing two's complement integer representation. As a result, the problem of finding the highest fitness value can be solved by Grover's algorithm without employing any genetic operators such as crossover and mutation.

Because the complexity of our algorithm adaptation is identical with its original form, and based on the analysis provided by [1], we reached the conclusion that any GA can be performed on a Quantum Computer in $\mathcal{O}(\sqrt{N})$ steps (Grover iterations in our case). This consequence broadens the area of computational problems where the quantum solutions outperform the classical ones.

References

1. Ahuja, A., Kapoor, S: A Quantum Algorithm for Finding the Maximum. ArXiv:quant-ph/9911082 (November 1999)
2. Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D., Margolus, N., Shor, P., Sleator, T., Smolin, J., Weinfurter, H. Elementary gates for quantum computation. Phys. Rev. A 52, 3457-3467 (1995).
3. Durr, C., Hoyer, P.: A Quantum Algorithm for Finding the Minimum. ArXiv:quant-ph/9607014 (July 1996)
4. Giraldi, G.A., Portugal, R., Thess, R.N.: Genetic Algorithms and Quantum Computation. ArXiv:cs.NE/0403003 (March 2004)
5. Grover, L.: Quantum mechanics helps in searching for a needle in a haystack, Phys. Rev. Lett. (79), pp. 325-328, (1997).
6. Han, K-H., Kim, J-H.: Genetic Quantum Algorithm and its Application to Combinatorial Optimization Problem. In Proc. of the 2000 Congress on Evolutionary Computation, citeseer.nj.nec.com/han00genetic.html, (2000)
7. Han, K-H., Kim, J-H.: Quantum-Inspired Evolutionary Algorithm for a Class of Combinatorial Optimization. IEEE Transactions on Evolutionary Computation, vol. 6, No. 6, pp.580-593 (2002)
8. Lukac, M., Perkowski, M., Goi, H., Pivtoraiko, M., Chung, H.-Y., Chung, K., Jeech, H., Byung-Guk, K., Yong-Duk, K.: Evolutionary Approach to Quantum and Reversible Circuits Synthesis, Artificial Intelligence Review, Vol. 20 , Issue 3-4, pp. 361-417 (2003)
9. Narayanan, A., Moore, M.: Quantum-Inspired Genetic Algorithms. IEEE International Conference on Evolutionary Computation (ICEC-96), Nagoya, Japan (May 1996) 61-66
10. Nielsen, M.A., Chuang, I.L. Quantum Computation and Quantum Information. Cambridge University Press (2000)
11. Omer, B.: Quantum programming in QCL. Technical Report, Institute of Information Systems, Technical University of Vienna (2000)
12. Oskin, M., Chong, F., Chuang, I.: A Practical Architecture for Reliable Quantum Computers. IEEE Computer, 35(1): 79-87 (2002).

13. Parhami, B.: Computer Arithmetic. Algorithms and Hardware Designs., Oxford University Press (2000)
14. Rylander, B., Soule, T., Foster, J.: Computational complexity,genetic programming, and implications. Proc. 4th EuroGP, pp . 348-360 (2001).
15. Rylander, B., Soule, T., Foster, J., Alves-Foss, J.: Quantum Evolutionary Programming. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001), Morgan Kaufmann (2001) 1005–1011.
16. Shor, P.W.: Algorithms for Quantum Computation: Discrete Logarithms and Factoring. Proc. 35th Symp. on Foundations of Computer Science, pp.124-134 (1994).
17. Spector, L., Barnum, H., Bernstein, K.J., Swamy, N.: Quantum Computing Applications of Genetic Programming. In Advances in Genetic Programming, volume 3. (1999)
18. Spector, L., Barnum, H., Bernstein, K.J., Swamy, N.: Finding a Better-than-Classical Quantum AND/OR Algorithm Using Genetic Programming. In Proceedings of 1999 Congress of Evolutionary Computation, Piscataway, NJ, IEEE Press vol 3 (1999) 2239-2246
19. Spector, L.: Automatic Quantum Computer Programming: A Genetic Programming Approach, Kluwer Academic Publishers, Boston, MA, (2004).
20. Vedral, V., Barenco, A., Ekert, A: Quantum Networks for Elementary Arithmetic Operations. quant-ph/9511018, (1996).