Compression Genetic Algorithms and Quantum Genetic Algorithms

M.R. Rakhshani¹, S. Mousavi²

1. Department of Mathematics, Zahedan branch, Islamic Azad University, Zahedan, Iran
2. Department of Electrical Engineering, Hafez Higher Education Institute, Zahedan, Iran

Corresponding Author email: bbmoosavi@gmail.com

ABSTRACT: A genetic algorithm is a class of adaptive stochastic optimization algorithms involving search and optimization. Generally, Evolutionary computation has already been known in computer science since more than 4 decades. More recently, another alternative of evolutionary algorithms was invented: Quantum Genetic Algorithms (QGA). In this paper, we outline the approach of QGA by giving a comparison with Conventional Genetic Algorithm (CGA). Our results have shown that QGA can be a very promising tool for exploring search spaces.

Keywords: Genetic Algorithms, Quantum Computing, Evolutionary Strategies.

INTRODUCTION

A growing interest is observed in Quantum Computation and Quantum Information due to the possibility to efficiently solve hard problems for conventional computer science paradigms. Quantum computation and quantum information encompass processing and transmission of data stored in quantum states. In these fields, the computation is viewed as affected by the evolution of a physical system, which is governed by unitary operators, according to the Laws of Quantum Mechanics [9]. The basic unity information is the qubit, the counterpart in quantum computing to the classical 0 – 1 bit. Quantum Computation and Quantum Information explore quantum effects, like quantum parallelism, superposition of states and entanglement in order to achieve a computational theory more efficient than the classical ones. This has been demonstrated through quantum factoring and Grover’s algorithm for database search [14].

However, Genetic Algorithms (GAs) is a rapidly expanding area of current research. They were invented by John Holland in the 1960s [13]. Simply stated, GAs are stochastic search algorithms based on the mechanics of natural selection and natural genetics [9, 16, 15]. They have attracted people from a wide variety of disciplines, mainly due to its capabilities for searching large and non-linear spaces where traditional methods are not efficient [9].

Applying GAs, people are attracted by their capabilities for searching a solution non-usual spaces. That is why people investigate the application of GAs for learning quantum operators [6, 8] and in the designing of quantum circuits [31, 29, 23].

Those works rely on the fundamental result for quantum computing that all the computation can be expanded in a circuit which nodes are the universal gates [9]. These gates offer an expansion of an unitary operator U that evolves the system in order to perform some computation [19, 14]. Thus, we are naturally in the face of two classes of problems: (1) Given a set of functional points \( S = \{(x, y)\} \) find the operator \( U \) such that \( y = U \cdot x \); (2) Given a problem, find a quantum circuit that solves it. The former was formulated in the context of GAs for learning algorithms [6, 8] while the latter through evolutionary strategies [31, 29, 23].

In 1996, Narayan and Moore [17] introduced a novel evolutionary computing method where concepts and principles of quantum computing are used to inspire evolutionary strategies. It was the first attempt towards quantum evolutionary programming. The basic approach is inspired on the multiple universes view of quantum theory: each universe contains its own population of chromosomes. The
populations in each universe obey identical classical rules and evolve in parallel. However, just after classical crossover within each universe, the universes can interfere with one another which induces some kind of crossover involving the chromosomes. The lack of a more formal analysis of the physical concepts used brings difficulties to make the correlation between the physics and the genetic algorithm itself. Consequently it does not offer clues of the advantages of a quantum implementation for GAs within th current implementations of quantum computers [20]. Thus, we are not going to consider it in the following sections.

This paper is organized as follows. The QIGA proposed in [11] is presented on section 2.1. We continue the review by presenting on section 2.2 the QGA proposed in [25]. Finally, we present the conclusions on section 3.

Quantum Evolutionary Computation
Quantum-Inspired Genetic Algorithms

The analysis of genetic algorithms based on quantum computing concepts is presented here. This is an important step towards the implementation of genetic algorithms in a quantum hardware. We start with the Quantum-Inspired Genetic Algorithm (QIGA) proposed in [11]. The QIGA is characterized by principles of quantum computing including qubits and probability amplitude. It uses a qubit representation instead of the usual binary, numeric, or symbolic representations [15, 16]. More specifically, QIGA uses a m-qubit representation, defined as:

\[
\left(\begin{array}{c}
\alpha_{i0} \\
\alpha_{i1}
\end{array}\right), \left(\begin{array}{c}
\alpha_{20} \\
\alpha_{21}
\end{array}\right), ..., \left(\begin{array}{c}
\alpha_{m0} \\
\alpha_{m1}
\end{array}\right),
\]

(1)

where each pair \((\alpha_{i0}, \alpha_{i1})\); i = 1,...,m, indicates a qubit.

Now, we must explain how convergence can be obtained with the qubit representation. Let us consider the following scheme, which is proposed in [11].

For each m-qubit chromosome of the form (1), a binary string \(x_1, x_2, ..., x_m\) is defined, where each bit is selected using the corresponding qubit probability, \(|\alpha_i|^2\) or \(|\alpha_i|^2\). Observe that if \(|\alpha_i|^2\) or \(|\alpha_i|^2\) approaches to 1 or 0, the qubit chromosome converges to a single state and the diversity given by the superposition of states disappears gradually.

An application dependent fitness function is used to evaluate the solution \(x_1, x_2, ..., x_m\). Another step is to design efficient evolutionary strategies. This would be accomplished through crossover and mutations but their implementations are not explained in [11]. Obviously, as usual, we can suppose a one-point crossover between parent chromosomes as well as unitary operators to change a randomly chosen qubit \((\alpha_{i0}, \alpha_{i1})\). However, as the QIGA has diversity caused by the qubit representation, the role of genetic operators is not clear. Also, it is stated in [11] that, if the probabilities of mutation and crossover are high, the performance of the QIGA can be decreased notably.

At the beginning of the algorithm, a population \(Q(t) = q_1, q_2, ..., q_m\) of m-qubit chromosomes is instantiated. Given a m-qubit chromosome in \(Q(t)\) we can find the corresponding binary string through the rule stated above. The so obtained binary string population will be denoted by \(P(t)\). Besides, there is an update step which aims to increase the probability of some states. Henceforth, given a qubit \((\alpha_{i0}, \alpha_{i1})\) of a m-qubit chromosome, it is updated by using the rotation gate \(U(\theta_i)\):

\[
U(\theta_i) = \begin{bmatrix}
\cos(\theta_i) & -\sin(\theta_i) \\
\sin(\theta_i) & \cos(\theta_i)
\end{bmatrix}, \quad \begin{bmatrix}
\alpha_{i0}' \\
\alpha_{i1}'
\end{bmatrix} = U(\theta_i) \begin{bmatrix}
\alpha_{i0} \\
\alpha_{i1}
\end{bmatrix},
\]

(2)

where \(\theta_i\) is formed through the binary solutions \(P(t)\) and the best solution found.

Let us present a pseudo-code of the QIGA developed in [11]:

Procedure QIGA
begin
  t ← 0
  Initialize Q(t)
  Make P(t) by observing Q(t)
  Evaluate P(t)
  Store the best solution b among P(t)
  while (not termination-condition) do
    begin
      t ← t + 1
      Make P(t) by observing Q(t−1)
        Evaluate P(t)
        Update Q(t) using quantum gates U(t)
        end
      Store the best solution b among P(t)
    end
end
The quantum gates $U(t)$ are application dependent. This step aims to improve the convergence. After updating $Q(t)$, the best solution among $P(t)$ is selected, and if the solution is fitter than the stored best solution, the stored solution is replaced by the new one. The binary solutions $P(t)$ are discarded at the end of the loop. A parallel version of the QIGA is presented in [12].

Quantum Genetic Algorithms

The work reported on section 2.1 shows that the application of quantum computing concepts to evolutionary programming is a promising research. The results presented points out that a quantum genetic algorithm (QGA) would outperform the classical ones. Besides, such implementation would take advantage of quantum parallelism as well as GAs parallelism. The obvious question is how to implement genetic algorithms in quantum computers?

The reference [25] is an effort to produce a QGA. Despite of the fact that there are several open points, it is the first effort in the direction of such algorithm.

The QGA proposed in [25] uses two registers for each quantum individual; the first one stores an individual while the second one stores the individual’s fitness. These two registers are referred as individual register and the fitness register, respectively. A population of $N$ quantum individuals is stored through pairs of registers $\{\text{individual register, fitness register}\}, i = 1, \ldots, N$.

At different times during the QGA the fitness register would store a single fitness value or a quantum superposition of fitness values. Identically for the individual register. Once a new population is generated, the fitness for each individual would be calculated and the result stored in the individual’s fitness register.

The effect of the fitness measurement process reduces each quantum individual to a superposition of classical individuals with a common fitness. It is a key step in the QGA [25]. Then, crossover and mutation would be applied. The whole algorithm can be written as follows:

Quantum Genetic Algorithm
Generate a population of quantum individuals.
Calculate the fitness of the individuals.
Measure the fitness of each individual (collapse).
while (termination-condition) do
Selection based on the observed fitness.
Crossover and Mutations are applied.
Calculate the fitness of the individuals.
Measure the fitness of each individual (collapse). end while

According to [25], the more significant advantage of QGA’s will be an increase in the production of good building blocks (schemata [13, 16]) because, during the crossover, the building block is crossed with a superposition of many individuals instead of with only one in the classical GAs.

One can also view the evolutionary process as a dynamic map in which populations tend to converge on fixed points in the population space. From this viewpoint the advantage of QGA is that the large effective size allows the population to sample from more basins of attraction. Thus, it is much more likely that the population will include members in the basins of attraction for the higher fitness solutions.

Another advantage is the quantum computer’s ability to generate true random numbers. By applying Kolmogorov complexity analysis, it has been shown that the output of classical implementations in genetic programming, which use a pseudo random number generator, are bounded above by the genetic programming itself, whereas with the benefit of a true random number generator there is no such bound [25, 24].

Despite of these promising features, fundamental points are not addressed in [25]. Firstly, it is not clear how to implement crossover in a quantum computer. Besides, how to perform the fitness function calculation in quantum hardware? Even a much more fundamental problem is that to explore the superposition of quantum individuals the correlation individual $\leftrightarrow$ fitness must be kept during the whole computation. Entanglement seems to be the only possibility to accomplish this task. But, in this case, things must be formally described to avoid misunderstandings and wrong interpretations.

CONCLUSIONS

Quantum genetic algorithm is a more wonderful optimization process than the conventional genetic algorithm, and its encoding mode is more complex, and each generation of the evolution can cover a wider area.
Quantum genetic algorithm combines some characteristics of quantum computation with the genetic algorithm whose individuals are chosen by the natural selection. The fine species better adapt to environment of quantum genetic algorithm are produced by the operation of quantum rotating gates with the goal of the best individual of current generation. The main features of quantum genetic algorithm can be summarized as follows:

1. QGA is self-organizing, self-adaptive, and self-learning. 2. The object handling by QGA is not the parameter, but the chromosome strings encoded according to parameter variable. The encoding operation enables the quantum genetic algorithm to operate directly on the structure of the object. 3. QGA searches a group of points rather than a single point in the solution space at the same time. QGA samples in different areas of space simultaneously. 4. In QGA, the knowledge of the searching space or other ancillary information is not needed. Only the fitness function is used to evaluate the individuals.

REFERENCES

Aiden H. Wright. Genetic algorithms for real parameter optimization. In Gregory J.

1710