Quantum Genetic Algorithms

Bart Rylander[1]
kerryr@pacifier.com
360-887-4424

Terry Soule[2]
tsoule@stcloudstate.edu
208-885-7062

James Foster[1]
foster@cs.uidaho.edu
208-885-6905

Jim Alves-Foss[3]
jimaf@cs.uidaho.edu
208-885-7232

[1]Initiative for Bioinformatics and Evolutionary Studies (IBEST)
Department of Computer Science
University of Idaho
Moscow, ID 83844-1008

[2]Department of Computer Science
St. Cloud State University
St. Cloud, MN 56301

[3]Center for Secure and Dependable Software
University of Idaho
Moscow, ID 83844-1008

Abstract

Recent developments in quantum technology have shown that quantum computers can provide a dramatic advantage over classical computers for some algorithms. In particular, a polynomial-time algorithm for factoring, a problem which was previously thought to be hard for classical computers, has recently been developed [Shor, 1994]. Similarly, a quantum algorithm for searching an unsorted database in square root of the time it would take on a classical computer has also been described (Grover, 1996]. Both algorithms rely upon the inherent parallel qualities of quantum computers to achieve their improvement. Unfortunately, not all problems can benefit so dramatically from quantum application [Jozsa, 1991]. Since most problems of real interest for genetic algorithms (GAs) have a vast search space [Holland, 1975], it seems appropriate to consider how quantum parallelism can be applied to GAs. In this paper we provide a brief background of quantum computers. We explain how quantum algorithms can provide a fundamental improvement over classical ones for some problems. We present a simple quantum approach to genetic algorithms and analyze its benefits and drawbacks. This is significant because to date there are only a handful of quantum algorithms that take advantage of quantum parallelism [Williams and Clearwater, 1997]. Finally, we provide ideas for directions of future research.

1 INTRODUCTION

The first major breakthrough in quantum computing came in 1985 with the development of the first true Quantum Turing Machine (QTM) [Deutsch, 1985]. In 1996, two researchers independently proved that a Universal Quantum Simulator was possible [Lloyd, 1996], [Zalka, 1996]. As such, anything computable by a classical computer would be computable by a quantum computer in the same time, if and when such a computer was in fact built. This cannot be said of the converse however. In the same paper Deutsch introduced the QTM, he also introduced a feature of quantum computers that was not replicable by classical computers, without an exponential slowdown, namely quantum parallelism. This unique feature turns out to be the key to most successful quantum algorithms.

Quantum parallelism refers to the process of evaluating a function once on a "superposition" of all possible inputs to produce a superposition of all possible outputs. In other words, all possible outputs are computed in the time required to calculate just one output with a classical computer. Unfortunately, all of these outputs cannot be as easily obtained. Once a measurement is taken, the superposition collapses. Consequently, the promise of massive parallelism is offset by the inability to take advantage of it.

This interesting but insufficient state of affairs was where quantum computation rested until recently. Very few and relatively impractical algorithms had been designed for quantum computers. This situation changed in 1994 with

the development of a fast, hybrid algorithm (part QTM and part TM) for factoring that took advantage of quantum parallelism by using a Fourier transform [Shor, 1994]. With this algorithm and a suitably sized quantum computer it is possible to provide a solution for factoring in polynomial time. This is an important development because the security of most public-key encryption methods relies upon the difficulty of factoring large numbers. Though not proven intractable, factoring had previously seemed secure. Consequently, quantum technology has already made a very tangible impact on some communities. Likewise, Grover's algorithm for searching a database shows that the benefits of quantum computing are not isolated to only one problem. Given the nature of GAs and their own form of parallelism, it seems natural to explore the possibility of implementing GAs *on* a quantum computer.

2 QUANTUM VS. CLASSICAL

There are two significant differences between a classical computer and a quantum computer. The first is in storing information, classical bits versus quantum *q-bits*. The second is the quantum mechanical feature known as *entanglement*, which allows a measurement on some q-bits to effect the value of other q-bits.

A classical bit is in one of two states, 0 or 1. A quantum q-bit can be in a *superposition* of the 0 and 1 states. This is often written as α |0> +β |1> where α and β are the *probability amplitudes* associated with the 0 state and the 1 state. Therefore, the values α^2 and β^2 represent the probability of seeing a 0 (1) respectively when the value of the q-bit is measured. As such, the equation $\alpha^2 + \beta^2 = 1$ is a physical requirement. The interesting part is that until the q-bit is measured it is effectively in *both* states. For example, any calculation using this q-bit produces as an answer a superposition combining the results of the calculation having been applied to a 0 and to a 1. Thus, the calculation for both the 0 and the 1 is performed simultaneously. Unfortunately, when the result is examined (i.e. measured) only one value can be seen. This is the "collapse" of the superposition. The probability of measuring the answer corresponding to an original 0 bit is α^2 and the probability of measuring the answer corresponding to an original 1 bit is β^2.

Superposition enables a quantum register to store exponentially more data than a classical register of the same size. Whereas a classical register with N bits can store one value out of 2^N , a quantum register can be in a superposition of all 2^N values. An operation applied to the classical register produces one result. An operation applied to the quantum register produces a superposition of all possible results. This is what is meant by the term "quantum parallelism."

Again, the difficulty is that a measurement of the quantum result collapses the superposition so that only one result is measured. At this point, it seems as if we have gained nothing. However, depending upon the function being applied, the superposition of answers may have common features. If these features can be ascertained by taking a measurement and then repeating the algorithm, it may be possible to divine the answer you're searching for probabilistically. Essentially, this is how Shor's algorithm works. First, you produce a superposition and apply the desired functions. Then, take a Fourier transform of the superposition to deduce the commonalities. Finally, repeat these steps to pump up your confidence in the information that was deduced from the transform.

The next key feature to understand is entanglement. Entanglement is a quantum connection between superimposed states. In the previous example we began with a q-bit in a superposition of the 0 and 1 states. We applied a calculation producing an answer that was a superposition of the two possible answers. Measuring the superimposed answer collapses that answer into a single classical result. Entanglement produces a quantum connection between the original superimposed q-bit and the final superimposed answer, so that when the answer is measured, collapsing the superposition into one answer or the other, the original q-bit also collapses into the value (0 or 1) that produces the measured answer. In fact, it collapses to *all* possible values that produce the measured answer. Given this very brief introduction to superposition and entanglement, we can begin to address our GA. (Interested researchers may refer to Williams and Clearwater [Williams and Clearwater, 1997] for a more detailed description of quantum computing.)

3 A QUANTUM GENETIC ALGORITHM

We now present a quantum genetic algorithm (QGA) that exploits the quantum effects of superposition and entanglement. To begin, we start with N quantum registers, labeled $reg1_0$ through $reg1_{n-1}$, where N will be the population size. Each of these registers is then placed in a superposition of *all* possible individuals. Thus, each register actually stores all possible individuals. Next, the fitness function is applied to each of the N quantum registers with the result stored in a second set of N quantum registers, labeled $reg2_0$ through $reg2_{n-1}$. The fitness function is applied to produce an entanglement between the original registers and the second set of registers.

Each of the original registers contain a superposition of *all* possible individuals. Thus, each of the second registers contain a superposition of all possible *fitnesses*. Although all individuals were evaluated, resulting in all fitnesses, only one application of the fitness function was necessary for each register. This is the quantum

parallelism.

Now we measure each of the second registers. This collapses the superpositions so that only a single fitness is seen in each register ($reg2_0$ through $reg2_{n-1}$). Because of the entanglement, this measurement also collapses $reg1_0$ through $reg1_{n-1}$. However, these registers do not collapse to a single individual, rather they collapse to a smaller superposition that only includes the individuals that would produce the measured fitness. For example, if the measurement of $reg2_i$ produces a result of 7, then $reg1_i$ will collapse to a superposition of all the individuals whose fitness is 7.

Clearly, the ideal solution would be to measure the maximum fitness in register $reg2_i$ thereby causing register $reg1_i$ to collapse into a superposition of perfect individuals. In fact, this would clearly identify the optimum and our task would be done. Unfortunately, there is no way to force a particular measurement. The result of a measurement is random, with probabilities determined by the probability amplitudes. Therefore, the probability of measuring a maximum fitness is exactly equal to the probability of creating a perfect individual randomly.

The set of N paired registers, half containing fitness values and half containing the superposition of individuals of those fitnesses, will represent our initial population. Crossover is applied normally. The contents of register $reg1_i$ are crossed with the contents of register $reg1_j$. Because these registers contain a superposition of individuals, the result is two new superpositions. In particular, if $reg1_i$ contains all individuals of fitness f_i and $reg1_j$ contains all individuals of fitness f_j then the result is the superposition of *all* individuals that can be produced through crossover at the chosen location.

Next, we apply the fitness function to the N registers ($reg1_0$ through $reg1_{n-1}$). The results are stored in the second set of registers and entangled with the original registers, just as in creating the initial population. A measurement is then taken. This collapses the superimposed fitnesses to a single value and collapses the individuals in $reg1_0$ through $reg1_{n-1}$ to only those individuals with the measured fitness. Selection is applied based on the measured fitnesses, thus finishing the generation. Mutations may be included if desired.

The final step is to obtain a result when the termination condition is reached. The final result will be N pair of registers, the first register of each pair will contain a set of superimposed individuals, all of the same fitness, entangled with the second register of the pair, which contains the measured fitness. A measurement of the first register will detect one of the individuals of the given fitness. This produces the desired result, a single individual of the predetermined fitness. Figure 1 outlines the QGA.

Set $Reg1_0$ through $Reg1_{n-1}$ into superpositions

apply fitness function to $Reg1_{0,n-1}$
storing result in $Reg2_{0,n-1}$
(this produces the entanglement)

measure each of $Reg2_{0,n-1}$
(all registers $Reg1_i$ and $Reg2_i$ collapse)

repeat

apply crossover to selected elements of $Reg1_i$
to generate the new population
(Reg2i is still entangled)

apply fitness function as before

measure each of $Reg2_{0,n-1}$
(all registers $Reg1_i$ and $Reg2_i$ collapse)

until population converges or termination condition met

Figure 1: QGA

4 ANALYSIS OF THE QGA

One may ask, what is gained by the quantum genetic algorithm? Currently this answer cannot be quantified. Only recently has there been a way to characterize the complexity of problems related to *classical* GAs. By using the Minimum Chromosome Length (MCL) method [Rylander and Foster, 2000] and applying it to the classical GA convergence proofs [Ankenbrandt, 1991], it is now possible to firmly establish the complexity of a problem instance for a GA. While the MCL method can be applied to the QGA as well as the GA, the mathematics for the probabilistic convergence times for the QGA have yet to be completed. Since it is currently unknown how the Schema Theorem [Goldberg, 1989b] will react to this form of probabilistic selection, it would be disingenuous to firmly assert any type of quantitative advantage. Still, it may be possible to gain insights through a discussion of the pros and cons of the QGA in an informal manner.

The apparent advantage for a QGA is the increased diversity of a quantum population. A quantum population can be exponentially larger than a classical population of the same "size". In a classical population each individual can only represent one potential solution. In a quantum population each "individual" is a superposition of multiple potential solutions. Depending on the particular stage of the generation these solutions share one of two traits. In the first case, they all have the same fitness. In the second, the same crossover created them all. Thus, a

quantum population is, at one level, much larger than a similar classical population.

However, it is unclear exactly how the additional diversity will influence the result. We consider the case of two individuals of relatively high fitness. If these are classical individuals, it is possible that these individuals are relatively incompatible, and that any crossover between them is unlikely to produce a very fit offspring. Thus, after crossover it is likely that the offspring of these individuals will not be selected.

If these are two quantum individuals then they are actually multiple individuals, of the same high fitness, in a superposition. As such, it is very unlikely that all of these individuals are incompatible. Thus, it is almost certain that some highly fit offspring will be produced during crossover. Unfortunately, the likelihood of measuring these individuals may not be very good. Therefore, it is possible that *on average* the quantum algorithm will not have a great advantage. However, at a minimum the good offspring are somewhere in the superposition, which is an improvement over the classical case. This is one advantage of the QGA.

It seems likely that the more significant advantage of QGA's will be an increase in the production of good building blocks. In classical GA theory good building blocks are encouraged because statistically they are more likely to produce fit offspring, which will survive and further propagate that building block. However, when a new building block appears in the population it only has one chance to 'prove itself'. Originally a good building block only exists in a single individual. It is crossed with another individual and to survive it must produce a fit offspring in that one crossover. If the individual containing the good building block happens to be paired with a relatively incompatible mate it is likely that the building block will vanish.

The situation is very different in the QGA. Consider the appearance of a new building block. During crossover the building block is not crossed with *only* one other individual. Instead, it is crossed with a superposition of many individuals. If that building block creates fit offspring with most of the individuals, then by definition, it is a good building block. It is therefore statistically likely that in measuring the superimposed fitnesses, one of the “good” fitnesses will be measured, thereby preserving that building block. By using superimposed individuals the QGA removes much of the randomness of the GA. Thus, the statistical advantage of good building blocks should be much greater in the QGA. This in turn, should cause the number of good building blocks to grow much more rapidly. This is clearly a *very* significant benefit.

Interestingly, the advantage of a QGA depends on the mapping from individual to fitness. If this mapping is one-to-one then measuring the fitness function collapses each quantum individual to a single solution and the benefits are lost. The greater the degree of "many to oneness" of the mapping from individual to fitness the greater the potential diversity advantage of a QGA.

5 DIFFICULTIES WITH THE QGA

There are some potential difficulties with the QGA presented here, even as a theoretical model. Some fitness functions may require "observing" the superimposed individuals in a quantum mechanical sense. This would destroy the superposition of the individuals and ruin the quantum nature of the algorithm. Clearly it is not possible to consider all fitness functions in this context. However, since mathematical operations can be applied without destroying a superposition, many common fitness functions will be usable.

As noted previously a one-to-one fitness function will also negate the advantages of the QGA. Another, more serious difficulty, is that it is not physically possible to exactly copy a superposition. This creates difficulties in both the crossover and reproduction stages of the algorithm. A possible solution for crossover is to use individuals consisting of a linked list rather than an array. Then crossover only requires moving the pointers between two list elements rather than copying array elements. However, without a physical model for our quantum computer it is unclear whether the notion of linked lists is compatible with maintaining a quantum superposition.

The difficulty for reproduction is more fundamental. However, while it is not possible to make an exact copy of a superposition, it is possible to make an inexact copy. If the copying errors are small enough they can be considered as a "natural" form of mutation. Thus, those researchers who favor using only mutation may have an advantage in the actual implementation of a QGA.

6 CONCLUSIONS

We have presented a quantum GA that uses the quantum features, superposition and entanglement. Our simple analysis of the algorithm suggests that it should have two advantages over a normal GA. First, because "individuals" in the QGA are actually the superposition of multiple individuals it is less likely that good individuals will be lost. Secondly, and more significantly, the effective statistical size of the population appears to be increased. This means that the advantage of good building blocks has been magnified. Presumably this will greatly increase the production and preservation of good building blocks thereby dramatically improving the search

process.

Unfortunately, these implied advantages can not be presently proven. Therefore, a good direction for future research would include providing a mathematical analysis of the convergence time of the QGA. Once this has been determined, it should be easy to evaluate the relative complexity of QGAs and their classical counterpart by applying the MCL method to each. Another potential fruitful direction would be to compare the ease of implementation of crossover methods versus mutation methods. Whereas with classical GAs applying only mutation is in effect a "numerative method" [Holland, 1975] which must contend with the time complexity involved in searching a vast search space, this problem may not exist for a QGA.

References

Ankenbrandt, C.A. (1991). An Extension To the Theory of Convergence and A Proof of the Time Complexity of Genetic Algorithms, Foundations of Genetic Algorithms, Morgan Kaufman. Pp. 53-68

Deutsch, D. (1985). Quantum Theory, the Church-Turing Principle, and the Universal Quantum Computer, Proceedings Royal Society London, Vol. A400 (1985), pp. 97-1117.

Goldberg, D.E. (1989b). Sizing Populations for Serial and Parallel Genetic Algorithms, Proceedings of the Third International Conference on Genetic Algorithms. San Mateo, CA: Morgan Kaufman. Pp. 70-79

Grover, L. (1996). A Fast Quantum Mechanical Algorithm for Database Search, Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (1996), pp. 212-219.

Holland, J. (1975). Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan Press.

Jozsa, R. (1991). Characterizing Classes of Functions Computable by Quantum Parallelism, Proceedings Royal Society London, Vol. A435 (1991), pp. 563-574.

Lloyd, S. (1996). Universal Quantum Simulators, Science, Vol. 273, 23 August (1996), pp. 1073-1078.

Shor, P. (1994). Algorithms for Quantum Computation: Discrete Logarithms and Factoring, Proceedings 35th Annual Symposium on Foundations of Computer Science, pp. 124-134.

Rylander, B., Foster, J. (2000). GA-Hard Problems, Manuscript, submitted to Genetic and Evolutionary Computation Conference (GECCO- 2000).

Williams, C., Clearwater, S. (1997). Explorations in Quantum Computing. Springer-Verlag New York, Inc.

Zalka, C. (1996). Efficient Simulation of Quantum Systems by Quantum Computers, Los Alamos National Laboratory, preprint archive, quant-ph/9603026