Comparison of Selection Strategies for Evolutionary Quantum Circuit Design

André Leier[1] and Wolfgang Banzhaf[2]

[1] Dept. of Computer Science, University of Dortmund, 44221 Dortmund, Germany
andre.leier@cs.uni-dortmund.de
[2] Dept. of Computer Science, Memorial University of Newfoundland, St. John's, NL, A1C 5S7 Canada
banzhaf@cs.mun.ca

Abstract. Evolution of quantum circuits faces two major challenges: complex and huge search spaces and the high costs of simulating quantum circuits on conventional computers. In this paper we analyze different selection strategies, which are applied to the Deutsch-Jozsa problem and the 1-SAT problem using our GP system. Furthermore, we show the effects of adding randomness to the selection mechanism of a (1,10) selection strategy. It can be demonstrated that this boosts the evolution of quantum algorithms on particular problems.

1 Introduction

Quantum Computing results from the link between quantum mechanics, computer science and classical information theory. It uses quantum mechanical effects, especially superposition, interference and entanglement, to perform new types of computation which promise to be more efficient than classical computation. However, up to now only a very narrow class of problems is known to be sped up by quantum algorithms, including factoring, (Shor's algorithm) [1], unstructured search (Grover's algorithm) [2] and also certain structured combinatorial search problems (Hogg's algorithm) [3] plus a few other number-theory problems [4, 5, 6]. Unfortunately, progress is still slow, yet the development of quantum algorithms seems to be crucial for future prospects of quantum computing.

Automatic quantum circuit design was motivated by the difficulties in manual quantum circuit design, because quantum algorithms are highly non-intuitive and practical quantum computer hardware is not yet available. Thus, quantum computers have to be simulated on classical hardware which naturally entails an exponential growth of computational costs and allows only to simulate small quantum systems (i. e. with only few qubits). Huge search spaces in even the simplest problems render evolutionary approaches nearly unable to achieve breakthrough solutions in the development of *new* quantum algorithms. At present we must be content to *evolve* essentially already existing quantum algorithms and to analyze the search space of these quantum circuits in order to improve

the efficiency of evolutionary search. Since the simulation of quantum circuits cannot be sped up, the only way to novel, complex quantum circuits leads via accelerated evolution.

This is our motivation for examining and comparing different selection strategies with respect to their effectiveness. Using our linear GP system, that comprises a quantum simulator for fitness evaluation of quantum programs, we implemented the (μ, λ) and $(\mu + \lambda)$ ES selection as part of a generational GP approach and tournament selection as part of a steady state GP. Moreover, we (i) combined the (1,10) selection strategy with additional randomness and, (ii) tested independently a step size adaptation. Our experiments were exemplarily performed on the Deutsch-Jozsa problem and the 1-SAT problem. Both problems were made suitable by choosing small instances.

This paper is organized as follows: An overview of related work on automatic quantum circuit design is given in Section 2. Our GP system, with the exception of the selection algorithms, is described in Section 3. Included are some basics of quantum computing, as far as they concern the understanding of this paper. The selection algorithms are separately treated in Section 4, while Section 5 explains the test problems. Section 6 presents experiments and their empirical results, before conclusions are drawn in the last section.

2 Related work

The idea of using genetic programming to design quantum circuits was discussed first by Williams and Gray in [7]. Given a unitary matrix U representing a desired quantum computation the aim was to find its decomposition into a sequence of elementary quantum gate operations. In contrast to subsequent GP schemes for the evolution of quantum circuits, a unitary operator solving a given problem had to be known in advance. Extensive investigations concerning the evolution of quantum algorithms were subsequently done by Spector et al. [8, 9, 10, 11]. In [8] the authors presented three different GP schemes for quantum circuit evolution: standard tree-based GP and both, stack-based and stackless linear genome GP. These were applied to evolve algorithms for Deutsch's two-bit problem, the scaling majority-on problem, the quantum four-item database search problem, and the two-bit-AND-OR problem. Better-than-classical algorithms could be evolved for all but the scaling majority-on problem. In [12] a linear-tree GP scheme was successfully applied to evolve a scalable quantum algorithm for 1-SAT, analogous to Hogg's algorithm. It was also found, that the mixing matrix can be implemented more efficiently by using simple Rx-Gates. In [13] we analyzed the structure of mutation landscapes of small instances of the Deutsch-Jozsa problem using autocorrelation functions and information measures for characterizing their behavior.

3 The GP System

3.1 The Quantum Computer Simulator

An integral component of our GP system is a quantum computer simulator. It simulates quantum algorithms, based on the idealized, noiseless (decoherence-free) quantum circuit model. Briefly speaking, quantum circuits are sequences of unitary matrices, so-called quantum gates, which are applied (i. e. multiplied) to an initial vector (or state) – usually a basis vector of the complex Hilbert space. This operation describes the transformation of a quantum mechanical, physical system – resulting in a final vector.

The dimension of the vector space grows exponentially in the number of quantum bits, qubits for short. A qubit is the basic unit of information in quantum computing. A basic set of quantum gates is sufficient to perform any arbitrary computations, i. e. it is, in the computational sense, universal. Some of the most elementary quantum gates are the Hadamard gate H, the single qubit rotation gates $Rx[\phi]$, $Ry[\phi]$, $Rz[\phi]$, the NOT and the controlled-NOT gate $CNOT$. The rotation gates need an angle parameter. In our system the resolution of the angle parameters was restricted to four bits allowing rotations as multiples of $1/8\pi$. Inputs to quantum algorithms are provided by certain unitary matrices (INP), sometimes known as oracles. They may change from instance to instance of a given problem, while the "surrounding" quantum algorithm remains unchanged. Our experiments were conducted using the universal gate set $\{H, NOT, CNOT, Rx[\phi], Ry[\phi], INP\}$. The initial vector is the first of the standard basis vectors.

At the end of a quantum computation always stands a measurement, otherwise we could not gather any information about the system. This can be seen just as a projection onto a basis vector. The probabilities for obtaining a certain basis vector are defined by and calculated from the vector coefficients.

To get a deeper insight into quantum computing and quantum algorithms [14, 15, 16] might be of interest.

3.2 Individual Structure and Genetic Operators

Because quantum circuits have a natural linear structure[3], it seems obvious to consider a linear genotypic representation for quantum circuits. The length or size of a quantum circuit, i. e. the number of quantum gates, should be limited to a reasonable value. It should be mentioned, that genotype and phenotype correspond directly to each other. The GP algorithm, generational or steady-state GP, decisively depend on the selection strategy that is described in detail in the next section.

In our GP system we exclusively considered mutation operators as evolutionary operators. These operators consist of random *deletion*, *insertion* and *replacement* of a single quantum gate, random *alteration* of parameters of a single gate and random *swap* of two neighboring quantum gates. Due to our experience, we refrained from using a crossover operator. Our observation showed

[3] Intermediate measurements providing a linear-tree structure [12] are disregarded.

that the efficiency of the search improves. This experience is confirmed by an analysis of the autocorrelation functions of time series obtained from random walks on the mutation [13] and crossover landscapes [17] of certain instances of the Deutsch-Jozsa problem[4]. For both, mutation and crossover landscapes the correlations are rather low. However, for crossover landscapes it is substantially lower (Fig. 1).

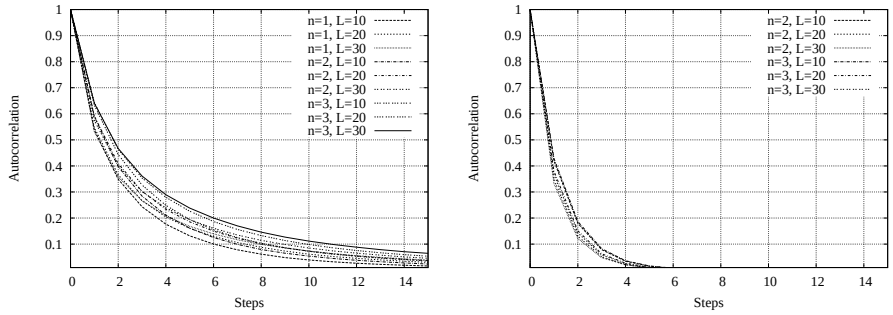


Fig. 1. Autocorrelation functions of mutation (left) and crossover landscapes (right) for the Deutsch-Jozsa problem with different values of n, the number of input bits of some Boolean function f (cf. Section 5.1), and L, the maximum length of a quantum circuit.

4 Selection Strategies

In general, selection is the result of a competition between individuals in a population. The better the individual's fitness in comparison with all other individuals in the population, the higher is its selection probability.

Some popular generational selection algorithms are the $(\mu + \lambda)$ and (μ, λ) selection as well as tournament selection. The (μ, λ) selection was originally used in ES-algorithms [18]. Here, μ parents are allowed to breed λ offspring, out of which the best μ are used as parents for the next generation. A variant of that method is the $(\mu + \lambda)$ selection [19], where offspring and parents participate in the selection. An algorithmic description of ES selection of type $(\mu + \lambda)$ and (μ, λ) follows:

[4] A detailed landscape analysis for landscapes regarding the 1-SAT problem has yet to be done.

1. Generate the initial population; population size is:
   - $\mu + \lambda$ (for $(\mu + \lambda)$ selection)
   - λ (for (μ, λ) selection)
2. Evaluate the fitness of each individual;
3. Select the winners:
   - the best μ individuals in the entire population (for $(\mu + \lambda)$ selection)
   - the best μ newly created individuals or offspring respectively (for (μ, λ) selection)
4. Perform one-point mutation on each winner to generate λ offspring (about λ/μ offspring per winner);
5. Evaluate the fitness of the offspring;
6. Go to step 3 unless termination criteria are met.

For our experiments we extended the $(1, 10)$ selection by adding randomness. After fitness evaluation of the offspring, instead of the best offspring, individuals were chosen randomly with a given probability p. In this case step 2 of the algorithm above looks as follows:

2. Let be $u = \text{unif}([0, 1])$ uniformly distributed; if $u < p$, then choose the winner randomly from the set of offspring and go ahead to step 4, otherwise evaluate the fitness of each individual;

Furthermore, we used (1,10) selection with self-adaptation [18] of the step-sizes. The self-adaptation routine is due to [20] and looks as follows:

$u = \text{unif}([0, 1])$
if $u < 0.5$
then $m* = 1.3$
else $m/ = 1.3$
$n_{mut} = \text{geo}(u, m)$

The parameter n_{mut} determines the step size, i. e. the number of mutations. It is drawn from a geometric distribution. In more detail,

$$\text{geo}(u, m) = \left\lfloor \frac{\ln(1-u)}{\ln(1-p)} \right\rfloor, \quad \text{where} \quad p = 1 - \frac{m}{1 + \sqrt{1+m^2}}.$$

The initial value of m has to be chosen appropriately.

Another important selection mechanism, *tournament selection*, does not belong to generational selection algorithms. Instead, it is based on competition within only a (usually small) subset of the population. A number of individuals taking part in the tournament is selected randomly according to the *tournament size*. In the smallest possible tournament, two individuals compete. The better individuals (the winners) are subject to genetic operators and replace the losers of the tournament. A higher selection pressure can be adjusted by a larger tournament size.

Independent of the selection strategy and the evolutionary process our GP system always stores the fitness of the best individual found so far. These data are the basis of all empirical results.

5 The Test problems

5.1 The Deutsch-Jozsa Problem

Given a function $f : \{0,1\}^n \to \{0,1\}$ (as a black box) promised to be either *constant*, $f(x) = c, \forall x \in \{0,1\}^n$ and $c \in \{0,1\}$, or *balanced*, i. e. as the result of f 0 occurs as many times as 1, the Deutsch-Jozsa problem is to determine which of the two properties f has. For $n = 1$ the task is also known as Deutsch's problem.

In classical computing f has to be evaluated $2^{n-1}+1$ times in the worst case, in quantum computing a single application of the corresponding input matrix, which is a Boolean function matrix defining f, is sufficient [21] (cf. [15, 13] for a detailed description of the input matrix). Using this matrix representation of a Boolean function, $n+1$ qubits are necessary to solve the problem on a quantum computer. The number of Boolean functions being either constant or balanced amounts to $2+\binom{2^n}{2^{n-1}}$. The general quantum algorithm solving the Deutsch-Jozsa problem is discussed in detail in [14, 15].

The fitness of a quantum program is measured by its ability to identify the property of a given function f. To obtain the fitness value of a quantum circuit, it is evaluated for each input matrix. Afterwards, the resulting vectors corresponding to balanced functions are compared with those corresponding to the two constant functions. On the basis of measurement probabilities for every base state the determinability of a correct classification is quantified. Misclassifications are penalized. A proper quantum algorithm, solving the Deutsch-Jozsa problem, classifies all functions correctly.

5.2 The 1-SAT Problem

The satisfiability problem (SAT) consists of a logical formula in n variables $v_1, \ldots, v_n$ and the requirement to find an assignment $a = (a_1, \ldots, a_n) \in \{0,1\}^n$ for the variables that makes the formula true. For k-SAT the formula is given as a conjunction of m clauses, where each clause is a disjunction of k literals v_i or $\bar{v_i}$ respectively with $i \in \{1 \ldots n\}$. Clauses which become false for a given assignment are called *conflicts*. The 1-SAT problem for n variables, solved by classical heuristics in $O(n)$ steps, can be solved even faster on a quantum computer. Hogg's quantum algorithm, presented in [22, 3], finds a solution in a single search step, using an input matrix, which encodes the number of conflicts for each assignment.

The number of fitness cases (the number of formulas) is $\sum_{k=1}^{n} \binom{n}{k} 2^k$ in total. Each fitness case consists of an input matrix for the formula and the desired

output. The fitness value is determined by the probabilities, that the final measurement will lead to a basis vector which corresponds to an assignment, making the formula true.

It applies to the fitness calculation of both problems, that the fitness function is also standardized and normalized. Moreover, quantum circuits with more than one input gate are strongly penalized.

6 Experiments and Empirical Results

We did experiments for the Deutsch-Joza problem with n=2 (3 qubits) and n=3 (4 qubits) and for the 1-SAT problem with n=3 (3 qubits) and n=4 (4 qubits). In this paper we present only plots of the problem instances with four qubits.

For these problem instances 20 evolutionary runs were performed for each selection method. The results were averaged over the runs. A GP run terminated when the number of single individual (quantum program) evaluations exceeded $1e + 07$ or the fitness of a new best individual under-ran a given threshold, which is close to zero. The length of the quantum circuits was limited to 15 gates. The best quantum circuits for the Deutsch-Jozsa (n=3) and the 1-SAT problem problem (n=4) need about nine gates each using the same gate set. The initial population was randomly created.

The selection strategies and parameter settings used in the GP system are:

- Comma- and Plus-strategies: (1,10), (5,20), (1+10), (5+20);
- (1,10) with step-size adaptation. The value of n_{mut} (the step size) was bounded to 5 mutations. The start value of m is set to 3. This might seem arbitrary, but experiments with larger values showed no or virtually no effect. A more detailed analysis is future work.
- (1,10), combined with additional random selection.
- Tournament selection with different population sizes (100, 500, 1000, 1500, 2000). Tournament size is 2.

Figure 2a shows four graphs illustrating the course of the evolutionary runs for quantum circuits of the Deutsch-Jozsa problem using pure comma- and plus-selection strategies. The (1,10) strategy achieved best results. However, for this strategy 25% of the runs did not lead to an acceptable result. For the other strategies the failure rate was even larger.

Tournament selection can outperform these runs when applied with a suitable population size of about 1500 to 2000 individuals, as shown in Figure 2b. For larger populations the performance decreases more and more. Yet even for tournament selection not every run was successfull within the required number of evaluations.

Performance of the (1,10) selection can be visibly boosted using step size adaptation, which was, all in all, the best selection strategy. In Figure 2c the plot essentially runs below the plots relating to all other selection strategies.

Couriously enough, comparable good performance was achieved by using the (1,10) strategy combined with 10% random selection. Further experiments

demonstrated that even higher percentages of random selection predominate over the pure comma- and plus-strategies. In all runs the resulting quantum circuit was a solution of the test problem. Thus, considering the number of successful runs, this strategy behaves even better than tournament selection on the given problem. Figure 2c illustrates the performance of all tested kinds of selection strategies.

This result confirms our analysis of the mutation landscape of the Deutsch-Joza problem. Because of the high ruggedness, the high modality and only a few paths to global optima the evolution is often caught in local optima. Larger steps or - what seems to help as much - additional randomness appears to compensate for this.

Applied to the 1-SAT problem instance we obtained a completely different result. Here, the best strategies are tournament selection with a population size of about 500 individuals, as shown in Figure 3a, and (1+10) selection, as shown in Figure 3b. Furthermore, Figure 3b illustrates, that plus-strategies perform far better than the comma-strategies. Step size adaptation does not improve evolution and additional randomness rather deteriorates the evolutionary search (Figure 3a). Moreover, for each of the tested selection strategies all 20 evolutionary runs were successful on this problem. This might be a hint that the fitness landscapes of the Deutsch-Jozsa and the 1-SAT problem clearly differ. The mutation landscape for 1-SAT should therefore be smoother and less complex, allowing evolution with comma- or plus-strategy to be more effective and independent of additional randomness and step-size adaptation. It remains to be seen, whether the fitness landscape analysis for the 1-SAT problem will confirm this prediction.

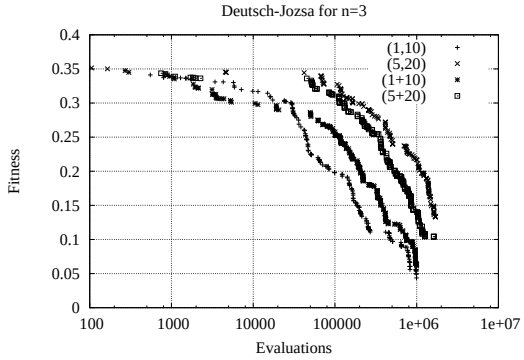
7 Conclusions

In this work we presented a comparison between different selection strategies for evolutionary quantum circuit design: tournament selection for different population sizes, simple comma- and plus-strategies, the (1,10) selection strategy combined with random selection or with step size adaptation. Our results were based on two test problems, the Deutsch-Jozsa and the 1-SAT problem, for which quantum algorithms already exist.

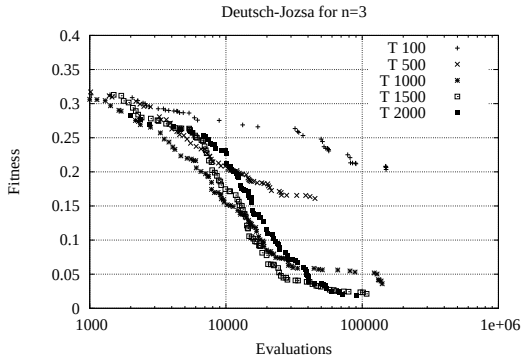
Depending on the population size, tournament selection can be very effective on either test problem. Self-adaptation is similarly effective and enables comma-selection to compete against tournament selection. Other parameter settings may even improve the search speed for the 1-SAT problem.

The comma-strategy with random selection seems to be useful on complex problems with rugged fitness landscapes, difficult for standard evolutionary selection mechanisms. Under those circumstances it can be effective or for smaller populations even better than tournament selection.

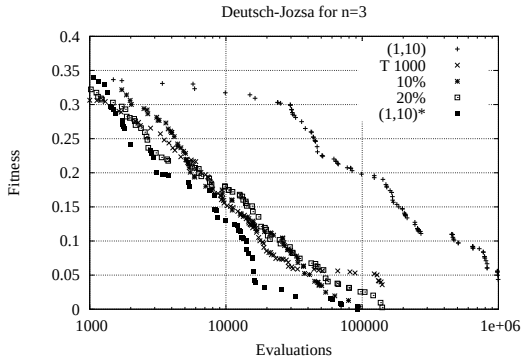
Further experiments on the influence of adding randomness for very difficult fitness landscapes would help to judge, whether this is more than just a pleasant side effect. Unfortunately there is only a small set of test problems available



(a)

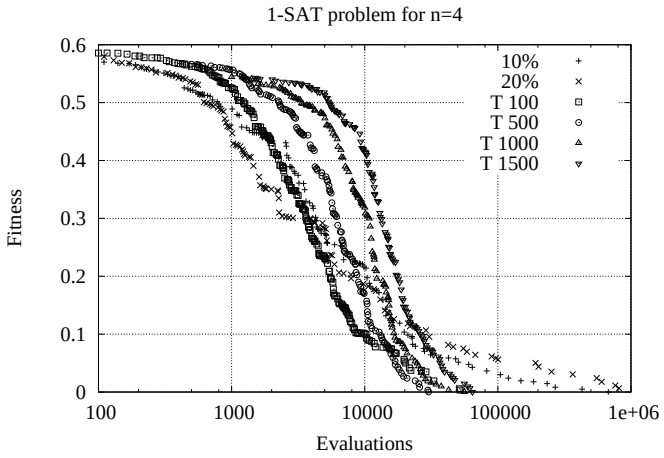


(b)

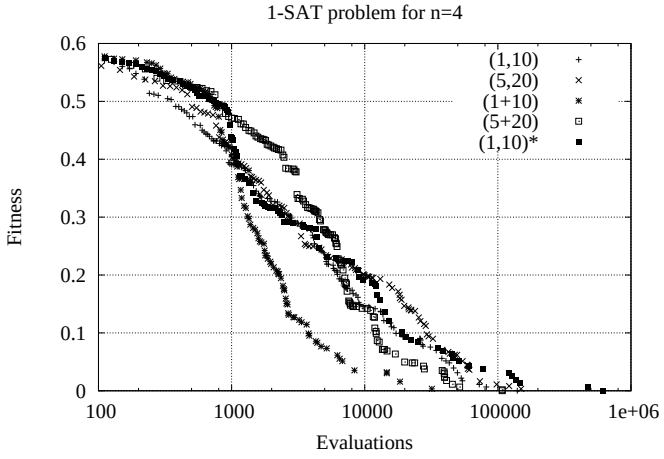


(c)

Fig. 2. Different selection strategies for the Deutsch-Jozsa problem with n=3: (a) comma- and plus-strategies; (b) tournament selection for different population sizes; (c) tournament selection, pure (1,10) selection, (1,10) selection with step size adaptation (marked with *) and (1,10) selection plus 10% or 20% random selection respectively. Note the logarithmic scaling on the x-axis.



(a)



(b)

Fig. 3. Different selection strategies for the 1-SAT problem with n=4: (a) Tournament selection for different population sizes and (1,10) selection strategy with 10% or 20% randomness, respectively. (b) (1+10), (5+20), (1,10) and (5,20) ES selection plus (1,10) selection with step size adaptation (marked with *). Note the logarithmic scaling on the x-axis.

and evolution of quantum circuits on larger state spaces is prohibitively time consuming.

References

[1] Shor, P.: Algorithms for quantum computation: discrete logarithm and factoring. In IEEE, ed.: Proceedings of the 35th Annual IEEE Symp. on Foundations of Computer Science (FOCS), Silver Spring, MD, USA, IEEE Computer Society Press (1994) 124–134

[2] Grover, L.: A fast quantum mechanical algorithm for database search. In ACM, ed.: Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC), New York, ACM Press (1996) 212–219

[3] Hogg, T.: Solving highly constrained search problems with quantum computers. J. Artificial Intelligence Res. **10** (1999) 39–66

[4] van Dam, W., Hallgren, S.: Efficient quantum algorithms for shifted quadratic character problems (2000)

[5] Hallgren, S.: Polynomial-time quantum algorithms for pell's equation and the principal ideal problem. In ACM, ed.: Proceedings of the 34rd Annual ACM Symposium on Theory of Computing (STOC), New York, ACM Press (2002)

[6] van Dam, W., Seroussi, G.: Efficient quantum algorithms for estimating Gauss sums (2002)

[7] Williams, C., Gray, A. In: Automated Design of Quantum Circuits. Springer, New York (1997) 113–125

[8] Spector, L., Barnum, H., Bernstein, H., Swamy, N. In: Quantum Computing Applications of Genetic Programming. Volume 3. MIT Press, Cambridge, MA, USA (1999) 135–160

[9] Spector, L., Barnum, H., Bernstein, H., Swamy, N.: Finding a better-than-classical quantum AND/OR algorithm using genetic programming. In Angeline, P., Michalewicz, Z., Schoenauer, M., Yao, X., Zalzala, A., eds.: Proceedings of the 1999 Congress on Evolutionary Computation, Silver Spring, MD, USA, IEEE Computer Society Press (1999) 2239–2246

[10] Barnum, H., Bernstein, H., Spector, L.: Better-than-classical circuits for OR and AND/OR found using genetic programming (1999)

[11] Barnum, H., Bernstein, H., Spector, L.: Quantum circuits for OR and AND of ORs. J. Phys. A: Math. Gen. **33** (2000) 8047–8057

[12] Leier, A., Banzhaf, W.: Evolving hogg's quantum algorithm using linear-tree gp. In Cantú-Paz, E., Wilson, S., Harman, M., Wegener, J., Dasgupta, D., Potter, M., Schultz, A., Jonoska, N., Dowsland, K., Miller, J., Foster, J., Deb, K., Lawrence, D., Roy, R., O'Reilly, U.M., Beyer, H.G., Standish, R., Kendall, G., eds.: GECCO-03: Proceedings of the Genetic and Evolutionary Computation Conference, Part I. Volume 2723 of LNCS., Springer (2003) 390–400

[13] Leier, A., Banzhaf, W.: Exploring the search space of quantum programs. In Sarker, R., Reynolds, R., Abbass, H., Tan, K., McKay, B., Essam, D., Gedeon, T., eds.: Proceedings of the 2003 Congress on Evolutionary Computation. Volume I., Piscataway, NJ, USA, IEEE Computer Society Press (2003) 170–177

[14] Gruska, J.: Quantum Computing. McGraw-Hill, London (1999)

[15] Nielsen, M., Chuang, I.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge, UK (2000)

[16] Hirvensalo, M.: Quantum Computing. Natural Computing Series. Springer, Berlin (2001)
[17] Wagner, G., Stadler, P.: Complex adaptations and the structure of recombination spaces. Technical Report 97-03-029, Santa Fe Institute (1998)
[18] Schwefel, H.P.: Evolution and Optimum Seeking. Sixth-Generation Computer Technology Series. John-Wiley & Sons, New York, USA (1995)
[19] Rechenberg, I.: Evolutionsstrategie '93. Frommann Verlag, Stuttgart, Germany (1994)
[20] Rudolph, G.: An evolutionary algorithm for integer programming. In Davidor, Y., Schwefel, H.P., Männer, R., eds.: Parallel Problem Solving from Nature – PPSN III, International Conference on Evolutionary Computation. Volume 866 of Lecture Notes in Computer Science., Berlin, Springer (1994) 139–148
[21] Cleve, R., Ekert, A., Macchiavello, C., Mosca, M.: Quantum algorithms revisited. Proc. R. Soc. London A **454** (1998) 339–354
[22] Hogg, T.: Highly structured searches with quantum computers. Phys. Rev. Lett. **80** (1998) 2473–2476