*In:* Tero T. Heikkilä, Mikio Nakahara, and Martti M. Salomaa (eds.) *Quantum Computing, Physical Realizations*, pages 73-82, Report TKK-F-A810, Helsinki University of Technology, Materials Physics Laboratory, 2001. Available from `ftp.uwasa.fi/cs/report01-2/TF.ps.Z`

# Optimisation by Quantum Computing and Genetic Algorithms

Jarmo T. Alander

During the last few years quantum computing has raised much enthusiasm among both computer scientists and physicists. In this brief critical review we deal with some aspects of quantum computing that are related to optimisation and especially to one heuristic optimisation method called genetic algorithm, whose functioning surprisingly closely resembles that of quantum computing. The specific problems dealt with are the following: comparison of quantum vs evolutionary computing in solving the famous traveling salesman (TSP) and Boolean satisfiability (SAT) problems, which a frequent test problems of search algorithms.

The simplest version of evolutionary computing is called genetic algorithm and it is based on a linear chromosome structure. However, the principle of GA processing is not limited to the linear chromosome structure, but also more complex structures like matrixes, trees, or even computer programs can be used as chromosomes. The last approach is called genetic programming (GP) and it is the most general approach used in evolutionary computing [1]. As an example of how genetic search approach can be utilised in quantum computing an example of how genetic programming can be used in the automatic i.e. computer aided design of quantum gate circuits.

In addition some general statistics of quantum computing research based on bibliographical information is given. Due to classification difficulties it may be never possible to reveal the exact number of quantum computing papers, but in any case in near future, if not already, the number is supposed to exceed 1,000, indicating the relatively high research activity of this new branch of science and engineering.

# 1 Introduction

## 1.1 Literature background

A general quantum computing (QC) and genetic algorithm (GA) search was done on web. A summary of the results of this query is given below:

October 10 2001: Query on www (Altavista) `quantum+computing` resulted in 11,076 fits while query `quantum+computer` resulted in 5,645 fits. As a comparison query `genetic+algorithm` resulted in 212,948 fits. The ratio QC/GA is thus about 1/20. In our genetic algorithm database we have 15,059 items, of which 125 [2] was classified as being more or less related to quantum computing giving ratio QC/GA = 1/120. This difference may be partly due to the fact that quantum computing is more recent than genetic algorithms. Web contains relatively more recent material than older. However, the author expects that a considerable portion of quantum computing references is missing from the database. From the ratio QC/GA we get an estimation for the total number of existing papers $N_{QC} \approx 125 \times 6 = 750$.

Further queries on web using the `dblp.uni-trier.de` reference database: `quantum computing` query resulted in 21 fits. This database contains mainly computer science references and is related to ACM. As a comparison `genetic algorithm` year = 1996 gave 49 fits (the absolute maximum limit of the system was 200 matches). (1997 N = 60, 1998 N = 66, 1999 N = 44, 2000 N = 49). Thus the ratio of this source to our database is about 1/40. From this we get another estimation for the total number of quantum computing papers $N_{QC} \approx 21 \times 40 = 840$.

A further query (November 9 2001) using ResearchIndex of NECI resulted in estimation of about 550 existing QC documents. The author assumes that the total number of references in QC is currently alreadyb about 1,000.

## 1.2 Quantum computing research activities

A brief summary statistics based on our quite complete bibliography of genetic algorithms, which currently contains also quantum computing references, is given next. For more information see [2] or [3]. Fig. 1 gives the annual distribution of the number of both genetic algorithm and quantum computing papers. As can be seen the growth has been nearly exponential so that the number of papers has been doubling approximately every second years. The growth rate is quite similar in both areas, while the lag between the histograms is about 12-14 years. The situation in quantum computing is much like that in genetic algorithms in the late 80's or early 90's, when GAs were arriving in Finland [4].

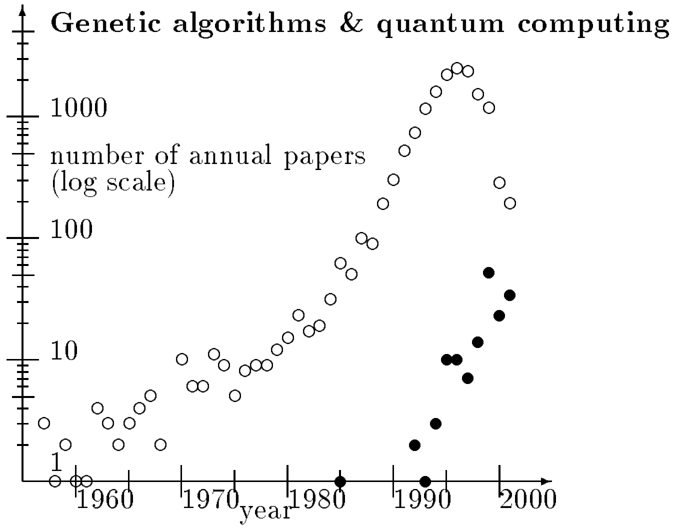


Figure 1: The number of papers in quantum computing (•, $N$ =157 ) and total GA papers (∘, $N$ =15102 ). Observe that the last few years are most incomplete in the database.

# 2 Solving SAT and TSP using QC

This section is based on the work done by Tad Hogg and Dmitriy Portnov, who have applied simulated quantum computing to solve two famous search or optimisation problems namely overconstrained satisfiability (SAT) and asymmetric traveling salesman problems (TSP). Both problems are known to be NP-hard i.e. the time required for the solution grows exponentially with the number of data points, because no known methods exists to solve them in polynomial time. Both problems are heavily used as test problems in optimisation studies.

The approach taken in the study by Tad Hogg and Dmitriy Portnov is as follows: The quantum algorithm mixes amplitudes among different states of qubits so that the bulk of amplitude gradually shifts towards states with relatively low costs. Tad Hogg has previously applied the same technique to a decision problem [5]. The approach, like e.g. genetic algorithms, is a heuristics i.e. there is e.g. no guarantee that it will really end up with the global optimum. In more details the algorithm is shown below [6]. For further details see [7].

1. **Initialisation**: initialise the amplitude equally among the states, giving $\psi_s^{(0)} = 2^{-n/2}$ for each of the $2^n$ states $s$.

2. **Mixing**: for steps 1 through $j$, adjust amplitude phases based on the

costs associated with the states and then mix them. These operations correspond to matrix multiplication of the state vector, with the final state vector given by:

$$\psi^{(j)} = U^{(j)} P^{(j)} \cdots U^{(1)} P^{(1)} \psi^{(0)},$$

where, for step $h$, $U^{(h)}$ is the mixing matrix and $P^{(h)}$ is the phase matrix, both to be described in more detail in the text.

3. **Fitness**: measure the final superposition, giving state $s$ probability $p(s) = |\psi_s^{(j)}|^2$.

Compare this to the genetic algorithm shown below. There are some obvious similarities.

1. **Initialisation** create a (random) starting population and evaluate fitness of each individual
2. **Recombination** recombine best individuals using crossover and mutation
3. **Evaluation** evaluate fitness
4. **Selection** select best ones for the next generation
5. **Termination** if solution found stop, otherwise continue at step 2

The mixing matrix $U^{(h)}$ in the above quantum algorithm is defined as follows:

$$U^{(h)} = W T^{(h)} W,$$

where

$$W_{rs} = 2^{-n/2} (-1)^{|r \wedge s|}$$

is the Walsh transform and $s$ and $r$ refers to states. The expression $|r \wedge s|$ denoted the number of 1-bits the states $s$ and $r$ have in common. The matrix $T^{(h)}$ is diagonal with elements depending on $|s|$, the number of 1-bits that the state $s$ contains:

$$T_{ss}^{(h)} = t_{|s|}^{(h)} = e^{i\pi\tau_h |s|},$$

where $\tau_h$ is a constant depending on the class of problems and the number of steps. Observe, that it is not dependent on the particular problem instance being solved. From the above definitions it follows that the elements $U_{rs}^{(h)}$ depend only on the Hamming distance $d(s,r)$ between the states i.e.

$$U_{rs}^{(h)} = u_{d(r,s)}^{(h)} = (-i \tan(\pi\tau_h/2))^{d(r,s)}.$$

Hamming distance $d(s,r)$ is number of bits differing between $s$ and $r$. The problem to be solved is implemented as the phase adjustment matrix $P^{(h)}$, which is a unitary diagonal matrix determined by the costs $c(r)$ associated with each state $r$:

$$P_{rr}^{(h)} = p_{c(r)} = e^{i\pi\rho_h c(r)},$$

where $\rho_h$ is a constant. The cost $c(r)$ of the search state is a measure of the fitness of the solution represented by $r$ i.e. it implements the objective function of the optimisation problem.

Observe that in optimisation problems the minimum cost is not usually known *a priori*, and thus amplitude amplification can not give any enchancement.

In order to be able to run the proposed quantum search algorithm the following values must be defined: $j$ the number of steps, and the phase parameters $\tau_h$ and $\rho_h$ for $h = 1, \ldots, j$. Tad Hogg and Dmitriy Portnov give two principal ways of optimising these parameters in addition they refer also to quantum computing as one alternative to do rigorous optimisation. From the view point of genetic algorithms the optimisation of these parameters would be at least worth considering. Here one must look the problem of algorithm fine tuning also from the view point of computing. Generally the optimisation of optimisation algorithm may sound great but the danger is that it may lead to a problem, well known in machine learning, called overlearning. It means that the finetuned algorithm is extremely fast to solve the problem that it was finetuned to solve to—the problem is that for other problems it may perform much worse.

Using GA we have once "succeeded" in finetuning the parameters of GA so successfully that the resulting algorithm was unbeatably fast to solve the given fixed problem [8]. The bug was that the set of parameters included also the random number generator seed. The result was that the genetic algorithm found the random number seed that immediately lead to the solution. As warning as this real life example is, it also demonstrates the power of genetic algorithm based optimisation.

Tad Hogg and Dmitriy Portnov have finetuned the above parameters and solved two famous problems, three variable satisfiability (3-SAT) problem and a six city asymmetric traveling salesman problem (TSP).

The SAT problem means the following. Given a Boolean expression of $n$ variables $B(v_1, \ldots, v_n)$ assign values `true` and `false` to all $v_i$ so that $B(v_1, \ldots, v_n) =$ `true`. This problem seems to be quite easy. However, the problem is that the number of cases is $2^n$, which grows exponentially and is intractable for values exceeding roughly fifty. There is not any known algorithm that can solve general SAT problems in polynomial time. Actually, it can be shown that any NP complete problem can be transformed into a SAT problem in polynomial time. Thus if one succeeds to solve SAT in polynomial time, also other NP complete problems can be solved in polynomial time. If quantum computing can dramatically speed up processing of SAT problems, then it can speed up the processing of all NP complete problems.

Tad Hogg and Dmitriy Portnov solve the so called 3-SAT problem, which is the following: Given a set of Boolean expressions $B_i(v_{i1}, v_{i2}, v_{i3})$ of three variables $v_{i1}$, $v_{i2}$, and $v_{i3}$ selected for each expression independently from a set of $K$ variables, $1 \leq i \leq N$ and $1 \leq i1, i2, i3 \leq K$. Find the assingments for variables $v_i$ so that the number of true expressions $B$ is maximum. It can be shown that also this SAT problem class is NP complete. The results obtained by Tad Hogg and Dmitriy Portnov when solving rather difficult ($N = 80, K = 20, \mu = N/K = 4$) random samples of 3-SAT problems is comparable to, actually even somewhat better than those got using a known efficient local search heuristics GSAT [9]. However, it is possible that part of the good performance is due to overlearning. That is difficult, if not impossible, to judge based solely on the information given in [6].

The six city asymmetric TSP problem is really a modest test problem. The state of the art in this field has been to attack problems consisting of millions of cities using linear programming during already over a decade [10]. In any case the results got were less striking for TSP than for the 3-SAT problem.

Because both problems are frequently used in testing search and optimisation methods, it is not any surprise that also the author has used them in some tests of genetic algorithms [11, 12, 13]. The results seem to indicate that a TSP type problem where the permutation of parameters is essential is difficult for genetic algorithm based optimisation while a 3-SAT type problem is considerably easier. Actually most of the problem cases of SAT problems are even trivial to solve by just counting the number of occurrences of variables and their negations, when the Boolean expressions $B_i$ are disjunctions. It is near the phase transition value $\mu \approx 4.25$, where the really difficult degenerate problem instances can be found. By phase transition we mean here a dramatic change in the average difficult of a problem. While most SAT problems cases seems to be easy, the cases near the phase transion tend to be extremely difficult causing exponential processing time consumption [14].

# 3 Comparing GA and QC

As said above there seems to be some indication that the performance of QC and GAs have some similarity. In table 1 is given a brief list of comparable properties of quantum computing *versus* genetic algorithms. For more information on genetic algorithms see e.g. [15] or references in [16].

| *property* | *QC* | *GA* |
|---|---:|---:|
| example in | quantum mechanics | evolution |
| nature | heuristics | heuristics |
| functioning | stochastic | stochastic |
| building block | qubit | gene / bit |
| theory | difficult | very difficult |
| scanning method | entanglement | crossover |
| structural parameters | several | several |
| dynamics | iterations | generations |
| applicability | unknown | high |
| speed | extremely fast | slow |
| implementation | extremely difficult | easy |
| generality | unknown | wide |

Table 1: Comparison of some properties of quantum computing (QC) and genetic algorithms (GA).

Quantum computing has also inspired to develop new genetic algorithms that have some quantum computing flavour [17, 18, 19].

# 4 Automatic design of quantum circuits

As at least the students of this course have perhaps noticed the design of quantum circuits is a non-trivial mental activity. This section describes a genetic algorithm based method to automatically design quantum circuits [20]. Other similar work is described in [21, 22, 23, 24].

## 4.1 Genetic programming

In the work by Colin P. Williams and Alexander G. Gray genetic programming has been used to design quantum circuits, more specifically quantum teleportation circuits. The resulting circuits have been simple, even simpler than those previously designed manually. [20]

### 4.2 Fitness function

In order to be able to utilise evolutionary computing we must define the target or fitness function. Williams and Gray defined their fitness function as follows:

$$f(S,U) = \sum_{i=1}^{2^N} \sum_{j=1}^{2^N} |U_{ij} - S_{ij}|, \quad S,U \in U(2^N),$$

where $U$ is the unitary matrix of the target circuit and $S$ is the unitary matrix of the proposed circuit generated by GP. When $f = 0$, $S$ implements exactly the given target circuit.

A predefined set of elementary quantum gates are given to the GP, which is also able to simulate the functioning of the circuits consisting of the given gates and their interconnections by evaluating the resulting unitary matrix $S$.

The GP method was able to automatically create both send and receive parts of the teleportation circuit. The send circuit consisted of two XOR and L and R gates and was comparable to the known minimum send circuits designed manually. The solution was found after 26.4 generations on the average, when using population size 100. Both figures are really modest for GP, from which we can safely deduce that designing a teleportation circuit is very easy for GP.

The minimum receive circuit consisted also of four gates, which was two gates less than in the best human designed teleportation receiver gate so far.

## 5 Conclusions and future

From the relative easiness of automatic design of basic quantum circuits and the fact that GAs has been widely used in VLSI design [25], we can deduce that GA has high application potential also in design of more elaborate quantum circuits. Personally the author is interested in applying GAs to more realistic and challenging quantum circuit design problems in collaboration with those more familiar with the technical details of quantum computing. One specific topic could be a comparison study of QC and GA performance on different problem types representing different fitness landscapes types [26].

## References

[1] Jarmo T. Alander. Indexed bibliography of genetic programming. Report 94-1-GP, University of Vaasa, Department of Information Technology and Production

Economics, 1995. (ftp.uwasa.fi/ cs/report94-1/ gaGPbib.ps.Z).

[2] Jarmo T. Alander. Indexed bibliography of quantum computing. Report 94-1-QC, University of Vaasa, Department of Information Technology and Production Economics, 2001. (ftp.uwasa.fi/ cs/report94-1/ gaQCbib.ps.Z).

[3] Jarmo T. Alander. Indexed bibliography of genetic algorithms in chemistry and physics. Report 94-1-CHEMPHYS, University of Vaasa, Department of Information Technology and Production Economics, 1995. (ftp.uwasa.fi/ cs/report94-1/ gaCHEMPHYSbib.ps.Z).

[4] Jarmo T. Alander. Indexed bibliography of genetic algorithms in the Nordic countries. Report 94-1-NORDIC, University of Vaasa, Department of Information Technology and Production Economics, 1995. (ftp.uwasa.fi/ cs/report94-1/ gaNORDICbib.ps.Z).

[5] Tad Hogg. Quantum search heuristics. *Physical Review A*, 61():052311, 2000.

[6] Tad Hogg and Dmitriy Portnov. Quantum optimization. *Information Sciences*, 128():181–197, 2000.

[7] Tad Hogg, Carlos Mochon, Wolfgang Polak, and Eleanor Rieffel. Tools for quantum algorithms. *Int. J. Mod. Phys. C*, 10():1347–1362, 1999.

[8] Jarmo T. Alander and Timo Mantere. Genetic algorithms in software testing - experiments with temporal target functions. In Pavel Ošmera, editor, *Proceedings of the 6th International Mendel Conference on Soft Computing (MENDEL 2000)*, pages 9–14, Brno (Czech Republic), 7.-9. June 2000. Technical University of Brno.

[9] Bart Selman, Hector Levesque, and David Mitchell. A new method for solving hard satisfiability problems. In *AAAI-92 Proceedings Tenth National Conference on Artificial Intelligence*, pages 440–446, Jan Jose, California, 12. - 16. July 1992. AAAI Press/ The MIT Press.

[10] Donald I. Miller and Joseph F. Pekny. Exact solution of large asymmetric traveling salesman problems. *Science*, 251:754–761, 1991.

[11] Jarmo T. Alander. On Boolean SAT and GA. [27], Chapter 17, pages 190–207.

[12] Jarmo T. Alander. On optimal population size of genetic algorithms. In Patrick Dewilde and Joos Vandewalle, editors, *CompEuro 1992 Proceedings, Computer Systems and Software Engineering, 6th Annual European Computer Conference*, pages 65–70, The Hague, 4.-8. May 1992. IEEE Computer Society.

[13] Jarmo T. Alander. In Lance D. Chambers, editor, *Practical Handbook of Genetic Algorithms: New Frontiers*, volume III of *New Frontiers*, chapter 13. Population size, building blocks, fitness landscape and genetic algorithm search efficiency in combinatorial optimization: An empirical study, pages 459–485. CRC Press, Inc., Boca Raton, FL, 1999.

[14] Bernardo A. Huberman and Tad Hogg. Phase transitions in artificial intelligence systems. *Artificial Intelligence*, 33(2):155–171, October 1987.

[15] Jarmo T. Alander. Geneettisten algoritmien mahdollisuudet [Potentials of genetic algorithms]. Teknologiakatsaus [Technology review] 59/98, Teknologian kehittämiskeskus [Finnish Technology Development Centre], 1998. (in Finnish; 100pages; abstract in English; available via www URL: www.uwasa.fi/cs/GA).

[16] Jarmo T. Alander. Indexed bibliography of genetic algorithms basics, reviews, and tutorials. Report 94-1-BASICS, University of Vaasa, Department of Information Technology and Production Economics, 1995. (`ftp.uwasa.fi/ cs/report94-1/ gaBASICSbib.ps.Z`).

[17] Ajit Narayanan and Mark Moore. Quantum-inspired genetic algorithms. In *Proceedings of IEEE International Conference on Evolutionary Computation, 1996*, pages 61–66, 1996. IEEE, Piscataway, NJ.

[18] Kuk-Hyun Han and Jong-Hwan Kim. Genetic quantum algorithm and its application to combinatorial optimization problem. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 2, pages 1354–1360, 2000. IEEE, Piscataway, NJ.

[19] Kuk-Hyun Han, Kui-Hong Park, Ci-Ho Lee, and Jong-Hwan Kim. Parallel quantum-inspired genetic algorithm for combinatorial optimization problems. In *Evolutionary Computation, 2001, Proceedings of the 2001 Congress on*, volume 2, pages 1422–1429, 2001. IEEE.

[20] Colin P. Williams and Alexander G. Gray. Automated design of quantum circuits. In Colin P. Williams, editor, *Quantum Computing and Quantum Communications, QCQC'98, First NASA International Conference*, volume 1509 of *Lecture Notes in Computer Science*, pages 113–125, Palm Springs, CA, February 1998 1999. Springer-Verlag, Berlin.

[21] Y. Z. Ge, L. T. Watson, and E. G. Collins. Genetic algorithms for optimization on a quantum computer. In *Unconventional Models of Computation*, pages 218–227, 1998.

[22] Lee Spector, Howard Barnum, Herbert J. Bernstein, and Nikhil Swamy. Finding a better-than-classical quantum AND/OR algorithm using genetic programming. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 3, pages 2239–2246, Washington D.C., 6.-9. July 1999. IEEE, Piscataway, NJ.

[23] Taro Yabuki and Hitoshi Iba. Genetic algorithms and quantum circuit design, evolving a simpler teleportation circuit. In *Late Breaking Papers at the 2000 Genetic and Evolutionary Computation Conference*, pages 421–425, 2000.

[24] B. I. P. Rubinstein. Evolving quantum circuits using genetic programming. In *Evolutionary Computation, 2001, Proceedings of the 2001 Congress on*, volume 1, pages 144–151, 2001. IEEE.

[25] Jarmo T. Alander. Indexed bibliography of genetic algorithms in electronics and VLSI design and testing. Report 94-1-VLSI, University of Vaasa, Department of Information Technology and Production Economics, 1995. (`ftp.uwasa.fi/ cs/report94-1/ gaVLSIbib.ps.Z`).

[26] Wim Hordijk and Peter F. Stadler. Amplitude spectra of fitness landscapes. *Journal of Complex Systems*, 1():39–66, 1998.

[27] Jarmo T. Alander, editor. *Geneettiset algoritmit – Genetic Algorithms*, Report number TKO-C53, 1992.