A parallel spatial quantum search algorithm applied to the 3-SAT problem

M. Barreto, G. Abal, and S. Nesmachnow

Facultad de Ingeniería, Universidad de la República
Herrera y Reissig 565, CP 11300, Montevideo, Uruguay

Abstract This work presents a quantum search algorithm to solve the 3-SAT problem. An improvement over one of the best known classical algorithms for this problem is proposed, replacing the local search with a quantum search algorithm. The performance of the improved algorithm is assessed by simulating it using parallel programming techniques with shared memory. The experimental analysis demonstrate that the parallel simulation of the algorithm takes advantage of the available computing resources to improve over the efficiency of the sequential version, thus allowing to perform realistic simulations in reduced execution times.

1 Introduction

Quantum computing has emerged as a new computing model that can offer significantly improvement over classical computation. Many quantum algorithms have been proposed, that, when executed on a quantum computer, provide significant efficiency improvements with respect to their best-known analog classical methods [3]. The most remarkable are the Shor's algorithm which finds the prime factors of large integers exponentially faster than the best-known classical algorithm, and the quantum search algorithm for unsorted databases by Grover [13], which provides a quadratic improvement over classical search algorithms.

The development of new quantum algorithms has continuously stimulated the search for a practical quantum information processing machine. As far as a quantum machine is not available, a quantum algorithm is often tested by simulating it on classical computers. These simulations are often challenging, as simulating quantum systems in classical machines demands exponential resources and cannot be done efficiently. Parallel computing techniques in classical machines can be used to make these simulations tractable in some cases, by sharing the large amount of calculations and memory needed to emulate the behavior of quantum algorithms.

The 3-SAT is a case of the Boolean satisfiability problem (with m clauses and 3 variables in each clause), a classical optimization problem with important applications in many areas. The 3-SAT is NP-hard [9] and it is used to prove that many other problems are in the NP-complete class [12]. Since no polynomial-time algorithm is known to solve it, heuristic and metaheuristic techniques are useful methods for obtaining approximate solutions, as they allow to compute quasi-optimal solutions in reasonable times [29].

A clever use of quantum computing may improve the performance of existing classical algorithms. For instance, it has been suggested that quantum search may be used to speed up existing methods for solving NP-hard problems by using the ideas from the algorithm by Grover [3], achieving at most a quadratic improvement over classical unstructured search. In this work, we consider one of the best known algorithms for the 3-SAT problem [25] and propose a concrete improvement, using a variant of Shenvi's quantum search algorithm [26]. The performance of the improved algorithm is assessed by simulation using parallel computing techniques.

The article is organized as follows. Section 2 presents some basic concepts about quantum computing. The 3-SAT problem is formulated in Section 3. Section 4 describes the proposed quantum algorithm for the 3-SAT. The experimental analysis and the results of simulating this algorithm using parallel computing techniques are reported in Section 5. Finally, Section 6 summarizes the conclusions and discusses the main lines for future work.

2 Quantum information processing

Quantum mechanics describes the behavior of nature at small scales. It is a linear theory, and thus a superposition of allowed states is an allowed state. A quantum state belongs to a Hilbert space, so the superposition has complex coefficients and relative phases are important. Over the last twenty years, quantum ideas have been successfully applied to several information processing problems. In quantum computing the basic unit of information is the qubit, a superposition of the two states of a bit, $|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle$.

A key feature of quantum mechanics is its non-deterministic nature. When observed, quantum superpositions stochastically collapse to one of the basis states, with probabilities given by their coefficients. For instance, a series of observations of qubits prepared in state $|\varphi\rangle$ will result in $|0\rangle$ with probability $|\alpha|^2$ and in $|1\rangle$ with probability $|\beta|^2$, assuming that the state was properly normalized so that $|\alpha|^2 + |\beta|^2 = 1$.

The tensor product of single qubits is used to construct n-qubit basis states $(|0\ldots00\rangle, |0\ldots01\rangle, \ldots)$, allowing to build the 2^n basis states of a n-dimensional Hilbert space. These states are denoted as $|j\rangle$, where $j \in [0, 2^n - 1]$. A generic, normalized, n-qubit state is expressed as a superposition of 2^n basis states, $|\Psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} c_j|j\rangle$, where $\sum_j |c_j|^2 = 1$. The positive numbers $|c_j|^2$ are the probabilities of obtaining the result $|j\rangle$ when measuring the system in state $|\Psi\rangle$.

A one-step evolution is obtained from the unitary operator of the system, so that $|\Psi(t+1)\rangle = U|\Psi(t)\rangle$, where $t = 0, 1, 2, \ldots$ and U is a $2^n \times 2^n$ unitary operation ($U^\dagger U = I$). This evolution is time-reversible and quantum logic is a reversible logic. Irreversibility enters the picture when an observer makes a measurement to access information. A single application of U updates the n-qubit state, or its 2^n complex coefficients, in what has been called intrinsic parallelism of quantum mechanics. A classical simulation of this quantum process requires to evolve in 2^n channels, and this cannot be done efficiently.

The other quantum resource which sets apart quantum systems from classical ones is *entanglement*. This term designates quantum correlations between two (or more) quantum states, which differ from classical correlations and give a genuine non-local character to quantum mechanics. Entanglement is a valuable resource for quantum computing, and it is the key resource in several proposed applications, such as quantum teleportation or quantum cryptography [21].

Quantum algorithms are often built by modifying well-known classical algorithms to take advantage of the quantum resources. Many successful stochastic algorithms have emerged from classical Markovian processes, and quantum analogs of this process have been studied. The Quantum Walk (QW) is the most well studied of these quantum analogs [16]. The discrete-time version [2] requires an auxiliary *quantum coin* Hilbert space, whose states determine the directions of motion. Each step is a translation to a neighboring site, along a path determined by the previous coin state. According to the intrinsic parallelism implicit in quantum mechanics, the walker moves simultaneously in all available directions.

QW are useful tools to generate new quantum algorithms. The optimal algorithm for solving the element distinctness problem, which aims to determine whether a set has repeated elements or not, is based on QWs [4]. An optimal search algorithm equivalent to Grover's algorithm [13] uses a modified QW on an n-dimensional hypercube to find an element among N sites after $O(\sqrt{N})$ steps [26]; QW algorithms are the best methods for the problem of spatial search in two-dimensional regular networks [1,5]. Ambainis [3] suggested that using a quantum search might provide a quadratic efficiency improvement with respect to classical algorithms designed to solve the 3-SAT problem.

In this work, we implement such a quantum search as an extension to Schöning's algorithm [25] and test its performance using parallel simulations.

3 The 3-SAT problem

The 3-SAT is a classical optimization problem in the field of computing theory, with important applications in many other research fields such as electronic design and verification planning [19], scheduling [10] and cryptography [27]. This section introduces the 3-SAT problem, presents its mathematical formulation, and briefly summarizes related work.

3.1 Boolean satisfiability

In propositional logic, a literal is either a logical variable or its negation, and a Boolean expression in its conjunctive normal form (CNF) is a conjunction of a set of m clauses, each of whom is a disjunction of literals. The Boolean satisfiability (SAT) problem consists of determining, if it exists, a truth assignment for the variables that makes a given Boolean expression true. The k-SAT is the Boolean satisfiability problem restricted to clauses with k literals, and the 3-SAT problem is a special case for which $k = 3$. The k-SAT for $k > 3$ can always be mapped to an instance of a 3-SAT [12].

The mathematical formulation of the 3-SAT problem is as follows:

- Let there be a set of n literals $X \equiv \{x_1, \ldots, x_n\}$, where $x_r = \{0, 1\}$.
- Let there be a Boolean expression $\Phi = \bigwedge_{i=1}^{i=m} C_i$, formed by a set of m clauses $C = \{C_1, \ldots, C_m\}$, with $C_i = \bigvee_{j=1}^{3} l_{ij}$, where l_{ij} is either a literal x_r or its negation $\neg x_r$.
- The 3-SAT problem consists in determining the set of values for the literals $X \equiv \{x_1, \ldots, x_n\}$ that makes Φ true.

When $k = 2$ the problem is in the complexity class P, as it can be solved in polynomial time [6]. On the other hand, the k-SAT problem is NP-Complete when $k \geq 3$ [9]; in fact, it was the first problem proved to be NP-Complete and many NP-complete problems have been proven so, by reducing them to an instance of 3-SAT. Thus, if a polynomial time algorithm to solve the k-SAT for $k \geq 3$ is known, then every NP–complete problem can be solved in polynomial time. However, no such efficient algorithm to solve the 3-SAT is known.

3.2 Related work: algorithms to solve the 3-SAT

In the last decade several methods to solve the 3-SAT problem were proposed. Some quantum algorithms have been presented for this problem, despite that due to the unstructured search space, it is not among the most suitable to be solved using quantum techniques. A brief review is presented next.

Classical algorithms. The exponential time hypothesis by Impagliazzo et al. [14] states that no algorithm can solve the 3-SAT in subexponential time in the worst case. One of the best classical algorithms known to solve the 3-SAT is the non-deterministic algorithm by Schöning [25], which applies (an exponential number of times) a local search of $3n$ steps on randomly selected points from the search space. The algorithm by Schöning has a computational complexity of $(1.329)^n$ where n is the number of literals in search space and succeeds with high probability to solve the 3-SAT.

The PPSZ algorithm by Paturi et al. [23] is able to find a 3-SAT solution in $O(1,307^n)$. It is the best known algorithm for solving the 3-SAT with a unique solution, but its computational cost increases as the number of solutions grows.

Iwama and Tamaki [15] combined the Schöning and PPSZ algorithms, and reached a complexity order of $O(1,324^n)$. Rolf [24] slightly improved the complexity order, developing a $O(1,322^n)$ algorithm to solve the 3-SAT problem. Nevertheless, due to its conceptual simplicity, the algorithm by Schöning is still commonly used as a benchmark to compare algorithmic proposals for the 3-SAT.

Quantum algorithms. Algorithms based on QW have been successfully applied to solve search problems [3,18,28]. In this line of work, an abstract search algorithm on a generic regular graph based on a QW has been formulated by Ambainis et al. [5], and later applied to the spatial search problem in specific regular graphs by Abal et al. [1].

Ambainis [3] also proposed the theoretical possibility of using the quantum amplitude amplification technique [21] to quadratically improve a broad class of classical non-deterministic algorithms. Regarding the 3-SAT problem, according to Ambainis proposal the computational complexity of a quantum version of Schöning's algorithm may be reduced to $O(\sqrt{1.329^n}) = O(1.153^n)$, a significant improvement over classical methods for solving the 3-SAT.

The quantum algorithm to solve the 3-SAT problem by Ohya and Masuda [22] incorporated concepts from the chaos theory. This proposal was controversial, as it assumed certain hypothesis which are not satisfied by any of the currently accepted quantum computing models. For instance, Dugic [11] showed that the algorithm presented by Ohya and Masuda transforms a coherent superposition of states in an incoherent mixed state, and this can make the algorithm to fail.

Leporati and Felloni [17] proposed three quantum algorithms for the 3-SAT by using a quantum registry machine with Fredkin gates, where each circuit depends on the 3-SAT instance to solve This work assumed that after a measure an external observer can differentiate between a null and non-null vector.

Cheng et al. [8] combined Grover's search algorithm with hill-climbing and an evolutionary algorithm. Some qubits in Grover's algorithm are replaced by classical bits, which are assigned using classical 3-SAT algorithms. Quantum simulations were performed for problem instances up to 80 variables, by effectively reducing the number of qubits needed to represent the problem instance (18 qubits were used to represent a problem instance with 80 variables).

In this work, we propose to include a spatial quantum search with controlled neighborhood in Schöning's algorithm for the 3-SAT, and simulate the resulting algorithm using parallel computing techniques to evaluate its performance.

4 Quantum-search applied to the 3-SAT

The state-space X relevant to the 3-SAT problem forms an hypercube of n dimensions. A state in this space is identified by a binary string x of n boolean literals or, alternatively, by an integer label $k \in [0, 2^n - 1]$. One can move a step in any of n directions by flipping the value of the corresponding literal in the string x. The search of a specific state in this structured space is a case of spatial search problem, since at every time step it is possible to query an oracle to see if we are at the searched site, or to move to an adjacente site.

4.1 The spatial quantum search problem

A classical walk is a prescription of how a walker should move conditioned to the value of a random variable. On a regular graph of degree n the random variable may take n values, each with some probability p_n. The edges of the graph incident to a vertex v must have labels from 0 to $n - 1$. If the walker is in vertex v and the result of the random variable is j, then the walker moves to the vertex v' that is connected to v by an edge of label j. This procedure is repeated again and again. The result is a random walk on the graph.

In a quantum setting, both the toss of a coin and the shift of the walker must be performed by unitary operators. In a regular graph of degree n, the vector space where the walk takes place is $\mathcal{H}_C \otimes \mathcal{H}_V$, where $\mathcal{H}_C$ is the Hilbert space spanned by $\{|0\rangle, \cdots, |n-1\rangle\}$ representing the coin space and $\mathcal{H}_V$ is the Hilbert space spanned by $\{|0\rangle, \cdots, |2^n - 1\rangle\}$ representing the vertex space.

The form of the evolution operator is $U = S\ (C \otimes I)$, where C is a $n \times n$ matrix that acts only on the coin subspace and S is the shift operator given by $S|j\rangle|v\rangle = |j\rangle|v'\rangle$, where v' is the vertex that is connected to v by edge j. The coin operator is the same for all vertices. The walker starts at some initial configuration $|\psi_0\rangle$ and after r steps its state is $U^r|\psi_0\rangle$.

In quantum walk search algorithms, if we would like to search for vertex v_0, it must be distinguished, usually using an oracle function, such that $f(v) = 0$ unless $v = v_0$ and in that case $f(v_0) = 1$. In practice, one marks this vertex by applying a site-dependent coin operator. If the walker is not in the marked vertex, the coin of the original walk (C) is used. If the walker is in the marked vertex, the coin $-I$ is used. The new coin is defined by $C' = (-I) \otimes |v_0\rangle\langle v_0| + C \otimes (I - |v_0\rangle\langle v_0|)$. This new coin defines a new evolution operator given by $U' = S\ C'$. The most used coin is the *Grover coin*, which is the real unitary operator farthest from the identity. It is defined as $C = 2|s\rangle\langle s| - I$, where $|s\rangle = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} |i\rangle$. For this coin, all directions have the same weight. Using the Grover coin, $U' = U \cdot (I - 2|s, v_0\rangle\langle s, v_0|)$ is obtained. As mentioned before, it is possible to perform a very general analysis of search algorithms on graphs by requiring some properties for U [5].

The search method consists in constructing an initial superposition state and applying the algorithm a specific number of times (of order $\sqrt{2^n}$) and then making a measurement. The state will collapse on the searched state with probability $O(1)$. If the search state is not found, the process can be repeated a few number of times. The algorithm by Grover is a simple example of quantum search, and several other algorithms have been proposed [5,1]. Most relevant to our purposes is the algorithm by Shenvi [26], which applies to a quantum search in a hypercube.

4.2 A quantum search algorithm for the 3-SAT problem

The proposed method combines the ideas in the algorithms by Schöning and Shenvi, applying a quantum search using neighborhoods, a common procedure in heuristic and metaheuristic optimization methods [29].

Two elements in the search space are in the same neighborhood if they are "close", considering the number of movements needed to get to one element from the other. In the 3-SAT problem, a neighborhood is a subspace of qubits: let be the element $x = [x_1, x_2, ..., x_n], x_i \in \{0, 1\}$ and $r \leq l \leq n$. Two strings t_1, t_2 belong to the same neighborhood if and only if $x_k^{t_1} = x_k^{t_2}\ \forall\, k \notin [r, l]$ (these are the *fixed* qubits that defines the neighborhood). For example, $|00011\rangle$, $|00010\rangle$ and $|00001\rangle$ belong to the neighborhood defined by the elements in the search space that have x_5, x_4 and x_3 fixed in $|0\rangle$. The neighborhood is $N = |000**\rangle$, a subspace of size 2 qubits ($*$ stands for a 'don't care' boolean value).

Algorithm 1 describes the proposed method, named Shenvi with Local Search in Neighborhood (SLSN). It randomly chooses a string x^* and searches for a solution using Shenvi's algorithm in a neighborhood of x^*. If a solution is not found in this subset of elements, the algorithm jumps randomly to another location in the search space and repeats the process.

Algorithm 1 Description of the SLSN algorithm.

```
while not solution found do
  x* = generate random state
  define neighborhood N as the closest elements to x*
  if (N is satisfiable) then
    apply Shenvi(x*, N)
    measure quantum state
    if (solution found) then
      finalize
    end if
  end if
end while
```

Suppose that $x^* = |x_5x_4x_3x_2x_1\rangle$, and that the neighborhood is defined as the elements with qubits x_1 and x_2 fixed. All elements in this neighborhood have the pattern $|***x_2x_1\rangle$. A given neighborhood is satisfiable if the fixed variables do not make any clause evaluate to false. This concept is applied in order to avoid searching for a solution in subsets where it is impossible to find one. SLSN uses a classic function to evaluate if the neighborhood is satisfiable or not; in case of not being satisfiable, the algorithm repeats the process considering another candidate x^* and another neighborhood. Figure 1 shows an example of how SLSN solves a problem with 5 variables and a neighborhood of 3 qubits.

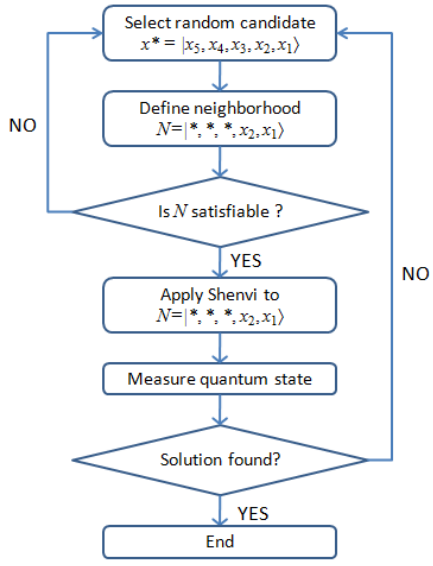

Figure 1. Example of the SLSN algorithm (5 variables, neighborhood size: 3 qubits).

5 Experimental analysis

This section presents the experimental analysis of the proposed quantum search algorithm to solve the 3-SAT problem. 3-SAT instances with few solutions were generated using the G3 algorithm by Motoki and Watanabe [20]. The theoretical relationship $m^* = 4.26n + 6.24$ defines a phase transition in which the 3-SAT problem goes from being an almost trivial problem to an extremely hard to solve problem [10]. This relation was considered in order to generate hard to solve 3-SAT instances with few solutions.

The experimental analysis was performed on a Dell Power Edge (Quad-core Xeon E5430 at 2.66GHz, 8 GB RAM) from Cluster FING (cluster website: `http://www.fing.edu.uy/cluster`). The simulation of a quantum algorithm requires an exponentially large amount of memory to represent the search space and large execution times are needed because a classical computer operates sequentially on a set of elements that grow exponentially with the number of qubits. Applying parallel computing techniques allows to significantly improve the performance of the quantum simulations.

5.1 Theoretical performance

The evaluation of the computational cost of executing the quantum SLSN algorithm is done by estimating the execution cost of each step in the algorithm as it if had been executed on a quantum computer.

Given the neighborhood size s as a parameter, the method first chooses an element $x^* \in X$ at random and generates a random neighborhood. Then it checks that the neighborhood is in fact satisfiable. The computational cost of the satisfiability evaluation is $O(n - s)$. The cost of applying Shenvi's search algorithm to this neighborhood is $O(\sqrt{2^s})$. Then, the estimated cost of applying the SLSN algorithm is $O(n - s) + O(\sqrt{2^s})$.

A certain neighborhood is satisfiable if the fixed qubits do not make any clause evaluate to false. If the problem has a unique solution, since there are n possible satisfiable neighborhoods of size s, the probability that the chosen element x^* has as a substring a part of a solution to the problem is $n/2^{n-s}$. Thus, the average number of times that the SLSN algorithm must be executed until a solution is found will be $O\left(\frac{2^{n-s}}{n}\right)$. The overall cost for the SLSN algorithm is then

$$O(SLSN) = O\left(\frac{2^{n-s}}{n}\right) \times \left[O(n-s) + O(\sqrt{2^s})\right]. \quad (1)$$

For neighborhoods scaling as $s \sim \log_2 n$ such as those used in Schonning's algorithm, the expression in equation 1 implies that an exponential number of local searches are made. Schöning uses $3n$ steps in the local search before jumping to another subspace if no solution was found. Shenvi's algorithm needs $O(\sqrt{2^s})$ steps to search a neighborhood of size s. Thus, the default size of the neighborhood is set to $s = 2\log_2(3n)$. In this case, the local search requires $3n$ steps as in Schöning's algorithm.

5.2 Parallel simulations

Parallel computing techniques are a useful approach to speedup the simulation of quantum simulations by cooperatively using several computing elements. This work uses *shared-memory* parallelism, an easy to develop choice that have had remarkable results when solving complex problems. Shared-memory versions of the proposed algorithms were developed using the OpenMP library [7], in order to evaluate the benefits of applying parallel computing techniques.

Parallel Shenvi. A parallel version of the algorithm by Shenvi was implemented and used as a baseline for comparing the efficiency of the new SLSN algorithm. The efficiency analysis results are shown in the Table 1, reporting the average execution times and speedup values obtained in the 10 executions of the Shenvi simulation performed for each problem instance. Acceptable speedup values are obtained when using 8 computing elements.

n	**m**	**Shenvi execution time (in s.)**		**speedup**
		sequential	**parallel**	
15	52	78	23	3.39
16	55	316	76	4.15
17	60	1258	309	4.07
18	64	2989	818	3.65
19	67	10904 (3 hours)	2705 (45 minutes)	4.03
20	70	57308 (16 hours)	11428 (3 hours)	5.01
21	73	84730 (23.5 hours)	21239 (5.9 hours)	3.99

Table 1. Comparison of the sequential and parallel simulations of Shenvi.

Parallel SNLS. The shared-memory parallel version of the SLSN algorithm is one of the main contributions of this work. Table 2 shows the experimental results of the parallel SLSN evaluation, reporting the problem dimensions and the average values for the execution time, for the number of steps until a solution was found, and for the measures required, obtained in 10 independent executions performed for each problem instance. SLSN succesfully solved all the problem instances, and the number of steps significantly reduced as the number of variables grows. When solving 3-SAT instances with more than 16 variables and using neighborhoods with more than 9 qubits, SLSN requires less steps than the traditional Shenvi.

Table 2 indicates that the configuration that computed the best results was always the one with the lower number of qubits. The default value of qubits in the neighborhood was $2\log_2(3n)$ (performing $3n$ steps as in Schöning's algorithm), but the best efficiency values were obtained when using fewer qubits since the neighborhood size notably impacts in the performance of simulation of the SLSN algorithm.

n	m	# sol.	s	time(s)	steps	measures	n	m	# sol.	s	time(s)	steps	measures
15	52	14	2	0.2	3822	3.5	19	67	53	5	0.9	5970	6.0
			5	0.1	1041	9.3				7	2.2	2324	7.5
			7	0.5	1142	8.7				9	13.1	1469	4.5
			9	4.7	1155	5.0				12	91.8	316	5.2
			11	18.5	369	4.6				14	479.9	273	6.0
			13	140.8	226	10.1							
16	55	27	2	0.1	2646	5.3	20	70	233	5	0.2	510	3.6
			5	0.2	1286	4.2				7	1.2	506	3.2
			7	0.1	250	3.3				10	4.4	73	8.6
			9	2.6	479	2.9				11	10.4	97	2.4
			11	9.5	181	6.4				12	16.2	36	3.0
			13	93.6	237	4.4				14	131.4	32	6.4
										16	1225.2	40	2.0
17	60	55	2	0.2	3966	3.2	22	78	54	5	4.2	13688	4.2
			5	0.1	1370	6.1				7	16.2	11498	5.4
			7	0.2	639	4.7				10	75.6	1529	6.6
			9	1.9	370	7.7				11	319.0	2741	8.6
			11	15.1	290	4.5				12	894.0	2375	8.6
			13	43.7	111	4.1				14	1982.6	454	4.8
			15	409.5	86	4.1				16	11676.8	353	9.8
18	64	13	9	15.0	530	2.6	23	83	50	10	137.4	2653	2.0
			12	353.8	791	4.2				12	353.8	791	4.2
			11	26.8	160	4.0							
			13	108.0	107	2.4							
			15	370.6	28	8.6							

Table 2. SLSN results for the 3-SAT.

Figure 2 compare the execution times (in logarithmic scale) of the parallel SLSN simulations using the best and the worst neighborhood size, and the Shenvi simulation. Even when using the worst neighborhood size, SLSN significantly improves the efficiency results obtained with the parallel Shenvi.

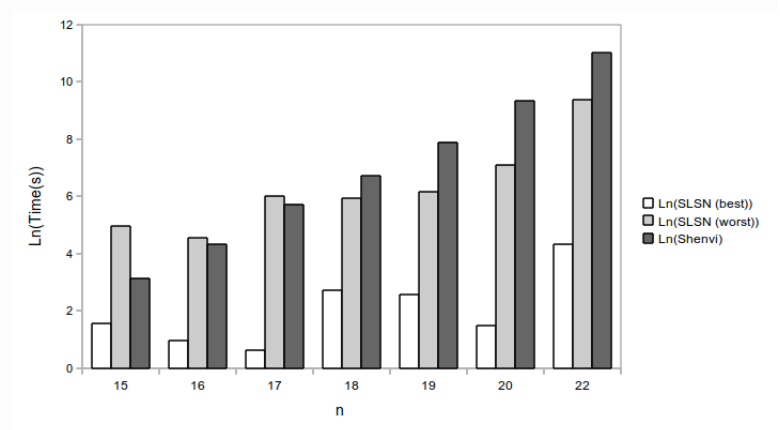

Figure 2. Execution time comparison: Shenvi, SLSN (best) and SLSN (worst).

6 Conclusions and future work

This work has presented a new quantum algorithm to solve the 3-SAT problem. The main contributions of the article are the new SLSN algorithm, which combines the search method proposed by Shenvi with a local search in neighborhoods as proposed in the classical algorithm by Schöning. Moreover, since quantum methods to solve the 3-SAT problem have been previously proposed but not simulated, a specific contribution of this work is the development of parallel simulations on shared-memory computers.

The experimental analysis performed in hard-to-solve 3-SAT instances generated using a well-known methodology demonstrated that the proposed method and its parallel implementation significantly speeds up the execution time required to perform the quantum simulations. The execution times results confirmed that by combining the approaches by Shenvi and Schöning, an efficient new method for solving the 3-SAT was devised. The local search in neighborhoods, a common procedure used in heuristic and metaheuristic optimization methods, is a useful tool to improve the efficacy of the 3-SAT solver. Mainly due to the computational complexity of the simulation, the best efficiency results were obtained when using small neighborhoods in the SLSN algorithm.

The main lines for future work are related with further studying the behavior of the SLSN algorithm and the parallel simulations. Regarding the first line, the impact of the neighborhood size on the quality of results/computational efficiency shall be better analyzed; and it would be interesting to investigate the distortion effect of noise in the quantum algorithm. On the other hand, other alternatives for designing efficient implementations of the quantum simulation can be explored (i.e. by using a mixed parallel model that combines the shared-memory the and distributed-memory approaches). Last, the research could be also extended to other known NP-hard problems, taking advantage of the quantum parallelism to build better alternatives to the best known classic algorithms.

References

1. Abal, G., Donagelo, R., Marquezino, F., Portugal, R.: Spatial search on a honeycomb network. Mathematical Structures in Computer Science 21, 1–11 (2010)
2. Aharonov, Y., Davidovich, L., Zagury, N.: Quantum random walks. Phys. Rev. A 48, 1687 (1993)
3. Ambainis, A.: Quantum search algorithms. SIGACT News 35, 22–35 (2004)
4. Ambainis, A.: Quantum walk algorithm for element distinctness. SIAM Journal on Computing 37, 210–239 (2007)
5. Ambainis, A., Kempe, J., Rivosh, A.: Coins make quantum walks faster. In: Proc. of the 16^{th} annual ACM-SIAM symposium on discrete algorithms. pp. 1099–1108 (2005)
6. Aspvall, B., Plass, M., Tarja, R.: A linear-time algorithm for testing the truth of certain quantified boolean formulas. Inf. Proc. Letters 8, 121–123 (1979)
7. Chapman, B., Jost, G., der Pas, R.V.: Using OpenMP: Portable Shared Memory Parallel Programming. MIT Press (1999)

8. Cheng, S., Tao, M.: Quantum cooperative search algorithm for 3-SAT. J. Comput. Syst. Sci. 73, 123–136 (2007)
9. Cook, S.: The complexity of theorem proving procedures. In: Proc. of the 3^{rd} Annual ACM Symposium on Theory of Computing. pp. 151–158 (1971)
10. Crawford, J., Auton, L.: Experimental results on the cross-over point in satisfiability problems. In: Proc. of the 11^{th} National Conference of the American Association for Artificial Intelligence. pp. 21–27 (1993)
11. Dugi, M.: A note on the ohya-masuda quantum algorithm. Open Systems & Information Dynamics 9, 115–123 (2002)
12. Garey, M., Johnson, D.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)
13. Grover, L.: A fast quantum mechanical algorithm for database search. In: Proc. of the 28^{th} ACM Symposium in the Theory of Computation. pp. 212–219 (1996)
14. Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? Journal of Computer and System Sciences 63, 759–764 (2001)
15. Iwama, K., Tamaki, S.: Improved upper bounds for 3-SAT. In: Proc. of the 15^{th} annual ACM-SIAM symposium on Discrete algorithms. pp. 328–328 (2004)
16. Kempe, J.: Quantum random walks - an introductory overview. Contemp. Phys. 44, 307 (2003)
17. Leporati, A., Felloni, S.: Three "quantum" algorithms to solve 3-SAT. Theor. Comput. Sci. 372, 218–241 (March 2007)
18. Magniez, F., Santha, M., Szegedy, M.: Quantum algorithms for the triangle problem. SIAM Journal on Computing 37(2), 413–424 (2007)
19. Marques-Silva, J.P., Sakallah, K.A.: Boolean satisfiability in electronic design automation. In: Proc. of the 37^{th} Annual Design Automation Conference. pp. 675–680 (2000)
20. Motoki, M., Watanabe, O.: Random generation of unique–solution 3SAT instances. Tech. rep., Tokyo Institute of Technology (2011), available at `http://www.is.titech.ac.jp/~watanabe/gensat/a1`. Retrieved April 2011
21. Nielsen, M., Chuang, I.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)
22. Ohya, M., Masuda, N.: Np problem in quantum algorithm. Open Systems & Information Dynamics 7, 33–39 (April 2000)
23. Paturi, R., Pudlak, P., Saks, M., Zane, F.: An improved exponential-time algorithm for k-SAT. In: Proc. of the 39th Annual IEEE Symposium on Foundations of Computer Science. pp. 337–364 (1998)
24. Rolf, D.: 3-SAT in RTIME(O(1.32793^{n})) - Improving randomized local search by initializing strings of 3-clauses. Electronic Colloquium on Computational Complexity 054 (2003)
25. Schöning, U.: A probabilistic algorithm for k-SAT and constraint satisfaction problems. In: 40^{th} Ann. Symp. Found. Comp. Sci. p. 410 (1999)
26. Shenvi, N., Kempe, J., Whaley, B.: Quantum random walk search algorithm. Phys. Rev. A 67, (052307) (2003)
27. Soos, M., Nohl, K., Castelluccia, C.: Extending SAT solvers to cryptographic problems. In: Proc. of the 12^{th} International Conference on Theory and Applications of Satisfiability Testing. pp. 244–257 (2009)
28. Szegedy, M.: Quantum speed-up of markov chain based algorithms. Annual IEEE Symposium on Foundations of Computer Science 0, 32–41 (2004)
29. Talbi, E.: Metaheuristics: From Design to Implementation. Wiley (2009)