Bounded structured quantum search for 3-SAT

José Javier Paulet González
Universidad Complutense de Madrid, Madrid, Spain

Luis Fernando Llana Díaz
Universidad Complutense de Madrid, Madrid, Spain

Manuel Nuñez García
Universidad Complutense de Madrid, Madrid, Spain

Hernán Indíbil de la Cruz Calvo
Departmento de Sistemas Informáticos, Universidad de Castilla-La Mancha, Albacete, Spain

Fernando López Pelayo
Departmento de Sistemas Informáticos, Universidad de Castilla-La Mancha, Albacete, Spain

Abstract

The most famous quantum search algorithm is Grover's algorithm [2]. This algorithm makes an unstructured search to find the solution to the problem in time $\sqrt{N}$ where $N = 2^n$ and n is the number of qubits. As in classical computing, in quantum computing we can take advantage of the structure of the problem to make searches faster as it can be seen in [1]. The contribution of this paper is mainly focused on dealing with constraints over the problem domain with the aim of improving the efficiency of these quantum searches.

2012 ACM Subject Classification Theory of computation → Quantum query complexity

Keywords and phrases Quantum computing, 3-SAT, Structured quantum search

Funding This work has been supported by the Spanish MINECO/FEDER project FAME (RTI2018-093608-B-C31); the Region of Madrid project FORTE-CM (S2018/TCS-4314) co-funded by EIE Funds of the European Union.

1 Structured quantum search in 3-SAT

A Boolean satisfiability problem stands for determining whether a truth assignment to the variables of a given Boolean formula $f(x_{n-1}, \ldots, x_0)$ exists. The size of the problem is usually the number of different variables appearing in the formula to be studied. The searching domain grows exponentially as 2^n where n is the number of different variables. This problem domain can be seen as a binary tree where each floor represents a new variable included and branches are either True or False.

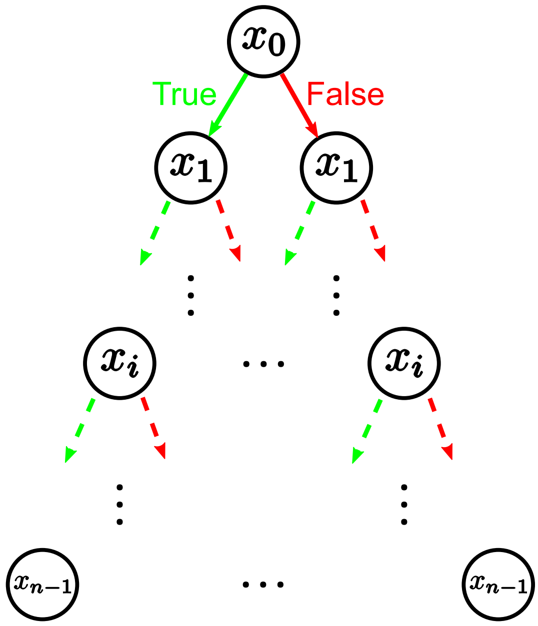


Figure 1 SAT problem domain in binary tree structure

Structured Quantum searching is somehow inspired on both usual Backtracking and on former Grover's algorithm to amplitude amplifying solution states. The Boolean function to be satisfied, in order to keep the condition of 'live node', at each floor of the binary tree is just a formula made of a subset of the former set of clauses $f(x_{n-1}, \ldots, x_0)$ in which all the literals belong to this floor. This way allows the binary tree only being unfolded on the branches corresponding to 'live nodes'. We can state that the number of iterations to perform all the above are:

$$k_{i,j} \approx \begin{cases} \textit{First search level } (j = 0) & \sqrt{\frac{2^{i+1}}{M_i}} \\ \textit{Intermediate search levels} & \sqrt{\frac{2^{i-j}}{M_i/M_j}} \\ \textit{Last search level } (i = n-1) & \sqrt{2^{n-j}} \end{cases}$$

Where i is the level at work for Grover's, j es the level of the previous search and M_i is the amount of possible solutions at i^{th} level.

Once all these Grover's searches have been performed (gathered by an operator called U) r iterations of the following circuit are required:

$$\left(\left(2U \left|0^{\otimes n}\right\rangle \left\langle 0^{\otimes n}\right| U^\dagger - I\right) \cdot Oracle\right)^r$$

This r number of iterations, assuming l levels of searching, is:

$$r = \sqrt{\frac{M_{i_{l-1}}}{M_f}} \tag{1}$$

Where i_{l-1} is the level before the last and M_f is the total number of solutions of the corresponding 3-SAT problem.

Therefore the total number of iterations is given by:

$$Iterations = r \cdot \left(\sum k_{i,j}\right)$$

The resulting quantum circuit would remain this way:

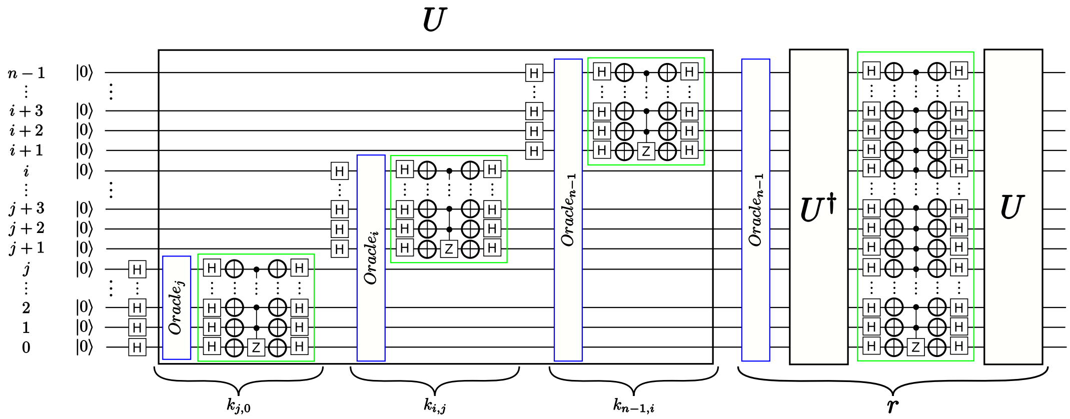


Figure 2 Structured quantum search circuit

Section 3 includes an example of this algorithm performing over a 15-variables 3-SAT problem.

2 Bounded search

Similarly to the ordinary quantum Grover's algorithm (non-structured search), the algorithm we are dealing with, also depends in an strong way on the size of the search domain. As expected, reducing the search domain speeds the search up in a remarkable way.

This section is devoted both to pursuit this idea and to estimate to what extent the saving on the number of iterations would be affected.

This first modification consist of instead of superposing all the range of states in each subspace, just reducing this range which, of course, will reduce the number of iterations at this point.

Firstly, we modify/reduce the amount of qubits superposed at the beginning which also reduces the inversion about the mean part.

For this task a sort of initial random superposition operator U_{S_i} will be built as defined in [3]. Afterwards, the new *Inversion about the mean* operator will be $\left(2U_{S_i}\,|0^{\otimes n}\rangle\,\langle 0^{\otimes n}|\,U^{\dagger}_{S_i} - I\right)$ so generating the circuit:

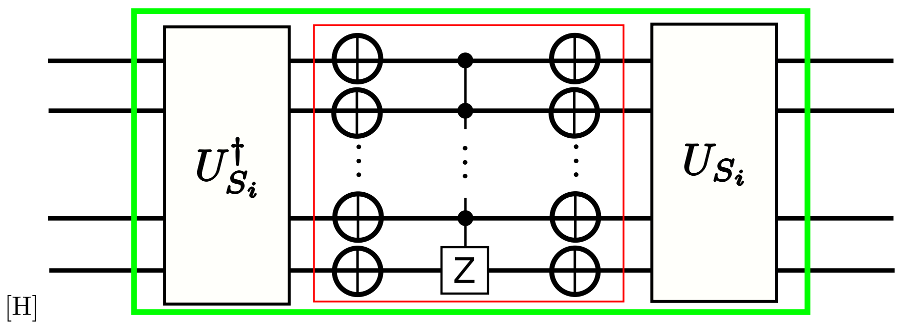


Figure 3 Inversion about the mean for bounded search

Let us elaborate on detail how to build U operator. It only differs with respect to the original one in the size of the sub-space's domains. Now each sub-space i is made of N_i states. Taking these numbers into account the number of iterations at each level is:

$$k'_{i,j} \approx \begin{cases} \textit{First search level } (j=0) & \sqrt{\frac{N_i}{M'_i}} \\ \textit{Intermediate search levels} & \sqrt{\frac{N_i - N_j}{M'_i / M'_j}} \\ \textit{Last search level } (i = n-1) & \sqrt{N_i - N_j} \end{cases}$$

Where N_j is the number of states in the sub-space of the previous search and M'_i is the number of solutions at i^{th} level within the range of states.

The following step gathers all these searches into a $U_{Bounded}$ operator and then r iterations of the following circuit are required:

$$\left(\left(2U_{Bounded}\left|0^{\otimes n}\right\rangle\left\langle 0^{\otimes n}\right| U^{\dagger}_{Bounded} - I\right) \cdot Oracle\right)^{r}$$

Again, the number of iterations r to be performed, assuming there are l levels to search, are:

$$r = \sqrt{\frac{M'_{i_{l-1}}}{M'_f}} \quad (2)$$

Where $M'_{i_{l-1}}$ and M'_f are the solution at the level before the last, and the total number of solutions of the 3-SAT problem within the corresponding range, respectively. The resulting quantum circuit would remain this way:

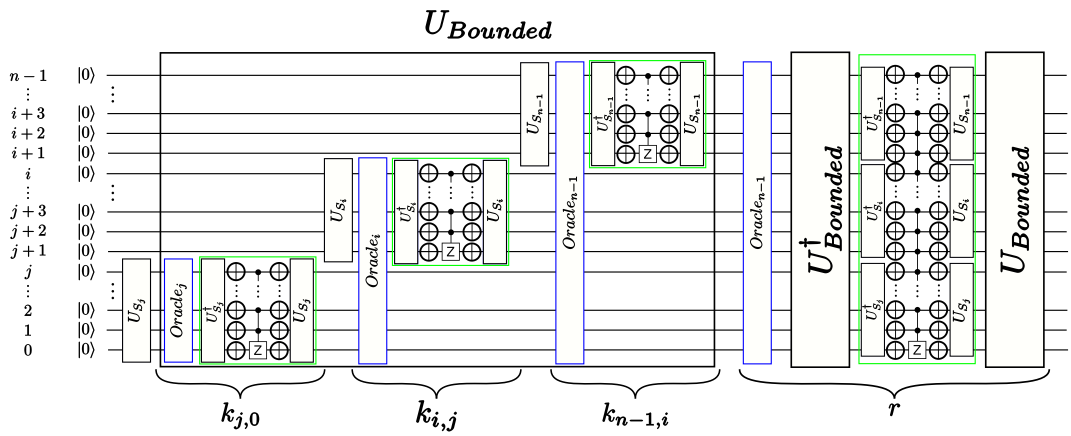


Figure 4 Structured quantum search bounded circuit

Section 3 includes an example of this bounded algorithm performing over a 15-variables 3-SAT problem.

3 Example

In this section, we are going to show both searches solving the next 3-SAT problem:

$$
\begin{aligned}
& f\left(x_{14}, x_{13}, x_{12}, x_{11}, x_{10}, x_{9}, x_{8}, x_{7}, x_{6}, x_{5}, x_{4}, x_{3}, x_{2}, x_{1}, x_{0}\right) = \\
& \quad (x_7 \vee \overline{x_{10}} \vee x_{12}) \wedge (x_4 \vee x_{13} \vee \overline{x_{14}}) \wedge (x_3 \vee \overline{x_{10}} \vee x_{14}) \wedge (x_1 \vee x_3 \vee x_7) \wedge (\overline{x_7} \vee \overline{x_{11}} \vee \overline{x_{13}}) \wedge \\
& \quad (\overline{x_5} \vee \overline{x_7} \vee \overline{x_{13}}) \wedge (x_3 \vee x_9 \vee x_{14}) \wedge (x_4 \vee x_7 \vee x_8) \wedge (\overline{x_3} \vee \overline{x_6} \vee x_{10}) \wedge (\overline{x_0} \vee \overline{x_4} \vee \overline{x_{14}}) \wedge \\
& \quad (x_5 \vee x_7 \vee x_{13}) \wedge (x_1 \vee \overline{x_9} \vee \overline{x_{12}}) \wedge (\overline{x_4} \vee \overline{x_8} \vee x_{13}) \wedge (\overline{x_4} \vee \overline{x_9} \vee \overline{x_{12}}) \wedge (\overline{x_0} \vee x_3 \vee \overline{x_6}) \wedge \\
& \quad (x_2 \vee \overline{x_{12}} \vee \overline{x_{13}}) \wedge (x_5 \vee x_{11} \vee x_{13}) \wedge (x_5 \vee x_6 \vee \overline{x_{14}}) \wedge (\overline{x_3} \vee \overline{x_8} \vee \overline{x_{14}}) \wedge (\overline{x_3} \vee \overline{x_8} \vee \overline{x_{10}}) \wedge \\
& \quad (\overline{x_1} \vee x_3 \vee x_9) \wedge (\overline{x_4} \vee x_6 \vee \overline{x_{11}}) \wedge (\overline{x_2} \vee \overline{x_{10}} \vee x_{11}) \wedge (x_0 \vee x_3 \vee \overline{x_{14}}) \wedge (\overline{x_3} \vee x_4 \vee \overline{x_6}) \wedge \\
& \quad (\overline{x_1} \vee \overline{x_6} \vee \overline{x_{13}}) \wedge (\overline{x_2} \vee x_{10} \vee x_{14}) \wedge (\overline{x_1} \vee \overline{x_5} \vee \overline{x_{14}}) \wedge (\overline{x_2} \vee \overline{x_5} \vee x_6) \wedge (\overline{x_1} \vee x_5 \vee \overline{x_{12}}) \wedge \\
& \quad (\overline{x_9} \vee x_{11} \vee \overline{x_{12}}) \wedge (x_7 \vee x_{11} \vee \overline{x_{14}}) \wedge (\overline{x_3} \vee x_4 \vee \overline{x_{12}}) \wedge (\overline{x_5} \vee \overline{x_8} \vee \overline{x_{12}}) \wedge (x_3 \vee \overline{x_8} \vee \overline{x_{14}}) \wedge \\
& \quad (x_1 \vee x_2 \vee x_{14}) \wedge (\overline{x_5} \vee \overline{x_9} \vee x_{10}) \wedge (\overline{x_6} \vee \overline{x_{10}} \vee \overline{x_{11}}) \wedge (x_0 \vee \overline{x_3} \vee x_7) \wedge (x_6 \vee \overline{x_8} \vee \overline{x_{10}}) \wedge \\
& \quad (\overline{x_0} \vee \overline{x_2} \vee \overline{x_{10}}) \wedge (\overline{x_3} \vee \overline{x_4} \vee \overline{x_{13}}) \wedge (x_8 \vee x_9 \vee x_{11}) \wedge (\overline{x_2} \vee x_7 \vee x_8) \wedge (x_3 \vee x_4 \vee \overline{x_7}) \wedge \\
& \quad (x_2 \vee x_7 \vee x_{14}) \wedge (x_3 \vee x_7 \vee x_{11}) \wedge (\overline{x_2} \vee x_7 \vee \overline{x_8}) \wedge (x_5 \vee x_{12} \vee \overline{x_{14}}) \wedge (\overline{x_0} \vee x_6 \vee x_{11}) \wedge \\
& \quad (x_4 \vee x_6 \vee x_8) \wedge (\overline{x_6} \vee \overline{x_{12}} \vee x_{14}) \wedge (\overline{x_6} \vee \overline{x_{10}} \vee \overline{x_{13}}) \wedge (\overline{x_2} \vee \overline{x_9} \vee \overline{x_{11}}) \wedge (x_0 \vee \overline{x_7} \vee x_{14}) \wedge \\
& \quad (x_4 \vee x_6 \vee \overline{x_{12}}) \wedge (\overline{x_5} \vee \overline{x_7} \vee \overline{x_9}) \wedge (\overline{x_9} \vee x_{10} \vee x_{11}) \wedge (\overline{x_0} \vee x_1 \vee \overline{x_{14}}) \wedge (\overline{x_3} \vee \overline{x_4} \vee x_9) \wedge \\
& \quad (\overline{x_2} \vee x_4 \vee x_5) \wedge (\overline{x_0} \vee x_4 \vee \overline{x_{14}}) \wedge (\overline{x_2} \vee x_3 \vee x_7) \wedge (x_1 \vee x_3 \vee \overline{x_5}) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_{12}}) \wedge \\
& \quad (\overline{x_0} \vee x_9 \vee \overline{x_{14}}) \wedge (\overline{x_1} \vee \overline{x_{10}} \vee \overline{x_{11}}) \wedge (\overline{x_4} \vee \overline{x_6} \vee \overline{x_9}) \wedge (x_2 \vee \overline{x_3} \vee \overline{x_6}) \wedge (x_1 \vee \overline{x_7} \vee \overline{x_{14}}) \wedge \\
& \quad (x_1 \vee x_3 \vee x_9) \wedge (x_0 \vee \overline{x_6} \vee \overline{x_8}) \wedge (\overline{x_7} \vee \overline{x_8} \vee \overline{x_{13}}) \wedge (x_0 \vee x_{12} \vee x_{14}) \wedge (\overline{x_5} \vee x_{10} \vee \overline{x_{13}}) \wedge \\
& \quad (x_2 \vee x_4 \vee x_5) \wedge (\overline{x_0} \vee \overline{x_1} \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3} \vee \overline{x_4}) \wedge (\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}) \wedge (\overline{x_2} \vee x_3 \vee \overline{x_4}) \wedge \\
& \quad (x_6 \vee x_7 \vee \overline{x_8})
\end{aligned}
$$

This problem has the following characteristics:

- 15 variables
- 81 clauses
- density = 5.4
- 1 solution

3.1 Structured quantum search

Let us consider two different levels of search for the sake of better illustrating how this strategy works:

1. One with the first level in i=8 $(x_7, \ldots, x_0)$ and the last level in i=14 $(x_{14}, \ldots, x_0)$.
2. Another one with the first level in i=6 $(x_5, \ldots, x_0)$, an intermediate one in i=9 $(x_8, \ldots, x_0)$ and the last in i=14 $(x_{14}, \ldots, x_0)$.

Qiskit[] is a well known SDK open-source to work on quantum computers that has been used to run our circuits. In particular *Aer* modulus together with the back-end *statevector_simulator* have been used both to simulate and to get probability amplitudes of the quantum states.

3.1.1 Case 1

For this first case the number iterations to execute at each level are:

- $k_8 = 2$
- $k_{14} = 8$
- $r = 3$

- $Iterations = 3 \cdot (2 + 8) = 30$

The resulting circuit for U operator can be seen here:

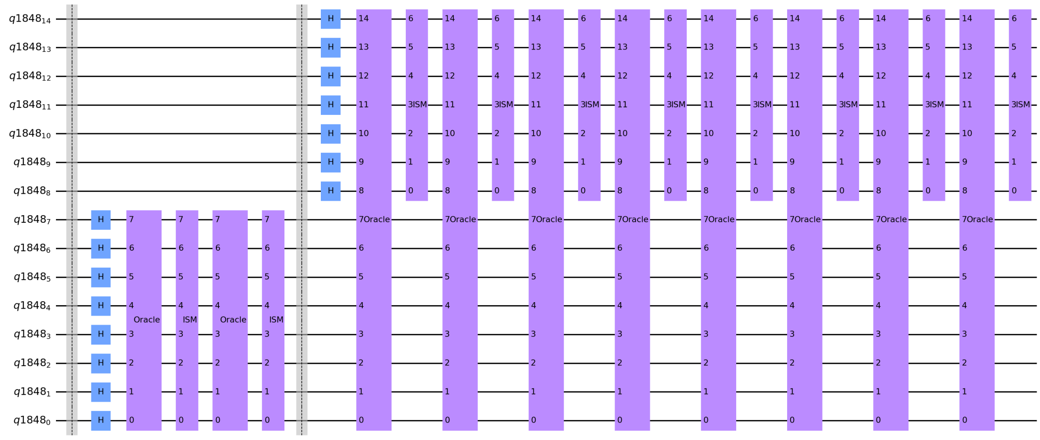


Figure 5 Quantum circuit of U (Case 1)

The whole structured quantum search circuit remains this way:

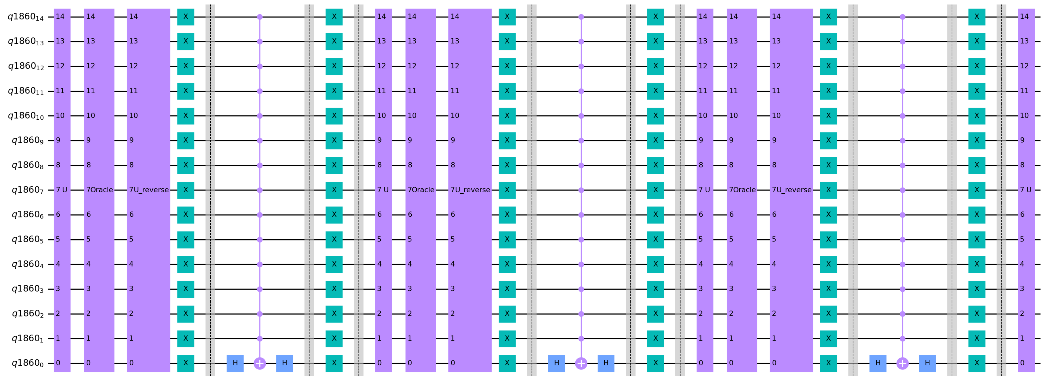
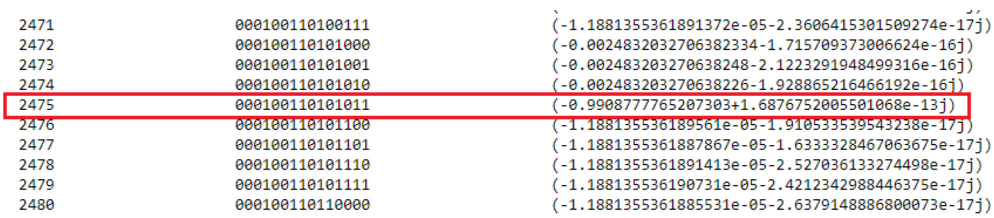


Figure 6 Structured search circuit (Case 1)

The outcome of this quantum circuit shows that with an almost 1 probability the solution state will be:



2471	000100110100111	(-1.1881355361891372e-05-2.3606415301509274e-17j)
2472	000100110101000	(-0.002483203270638234-1.715709373006624e-16j)
2473	000100110101001	(-0.002483203270638248-2.1223291948499316e-16j)
2474	000100110101010	(-0.002483203270638226-1.928865216466192e-16j)
2475	000100110101011	(-0.9908777765207303+1.6876752005501068e-13j)
2476	000100110101100	(-1.188135536189561e-05-1.910533539543238e-17j)
2477	000100110101101	(-1.1881355361887867e-05-1.6333328467063675e-17j)
2478	000100110101110	(-1.1881355361891413e-05-2.527036133274498e-17j)
2479	000100110101111	(-1.188135536190731e-05-2.4212342988446375e-17j)
2480	000100110110000	(-1.1881355361885531e-05-2.6379148886800073e-17j)

Figure 7 Probability amplitude of the solution state (Case 1)

3.1.2 Case 2

In this second case the number of iterations per level would be:

- $k_6 = 1$
- $k_9 = 1$
- $k_{14} = 6$
- $r = 3$
- $Iterations = 3 \cdot (1 + 1 + 6) = 24$

The corresponding U operator:

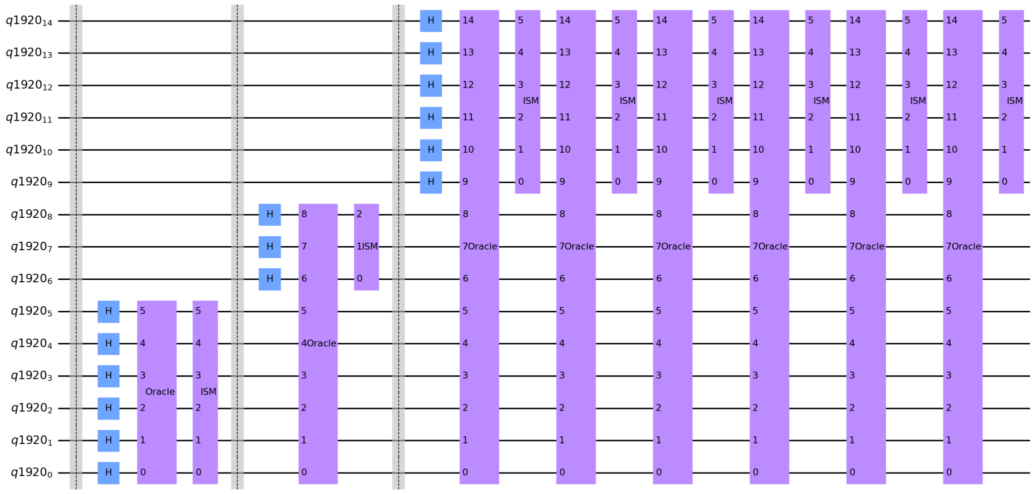


Figure 8 Quantum circuit of U (Case 2)

Finally the whole structured quantum search circuit is:

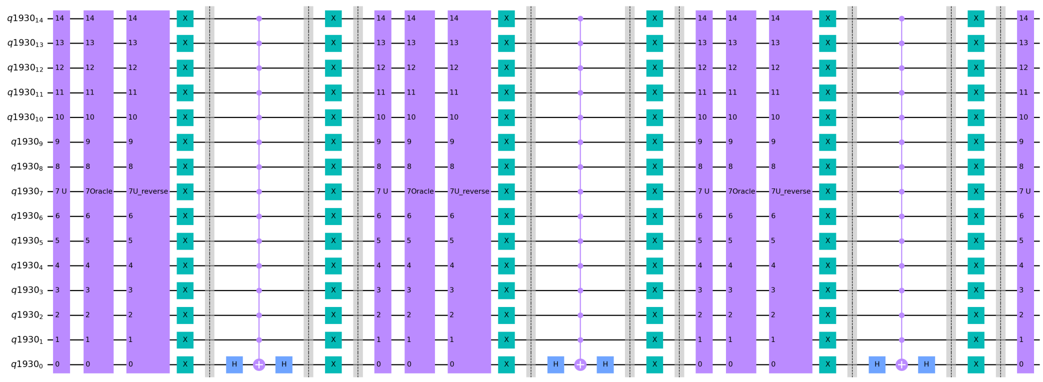


Figure 9 Structured search circuit (Case 2)

Once it has been executed, again with an almost 1 probability, the solution state will be:

2471	000100110100111	(-2.7555110398844728e-17+3.253643265445749e-17j)
2472	000100110101000	(-0.0003918437148030211-2.862817494515362e-16j)
2473	000100110101001	(-0.00039184371480300635-2.8987844085412176e-16j)
2474	000100110101010	(-0.00039184371480307855-2.925830924377076e-16j)
2475	000100110101011	(-0.9999042670145771+1.4585895119581795e-14j)
2476	000100110101100	(-0.00015673748592135808-6.294519078929736e-17j)
2477	000100110101101	(-0.00015673748592137082-6.737780376091153e-17j)
2478	000100110101110	(-0.0001567374859213013-7.482274941320633e-17j)
2479	000100110101111	(-2.364301810319745e-17+3.584469054025546e-17j)
2480	000100110110000	(-1.0967299683808878e-17+3.292148512807732e-17j)

Figure 10 Probability amplitude of the solution state (Case 2)

3.2 Bounded structured quantum search

In this example we have shared the levels with previous second case. Ranges per sub-space are so defined:

- First sub-space $(x_5, \ldots, x_0) \longrightarrow [20, 51]$
- Second sub-space $(x_8, \ldots, x_0) \longrightarrow [0, 7]$
- Third sub-space $(x_{14}, \ldots, x_0) \longrightarrow [0, 31]$

The number of iterations per level to be executed are:

- $k_6 = 1$
- $k_9 = 2$
- $k_{14} = 4$
- $r = 2$
- $\mathit{Iterations} = 2 \cdot (1 + 2 + 4) = 14$

The circuit for U operator is:

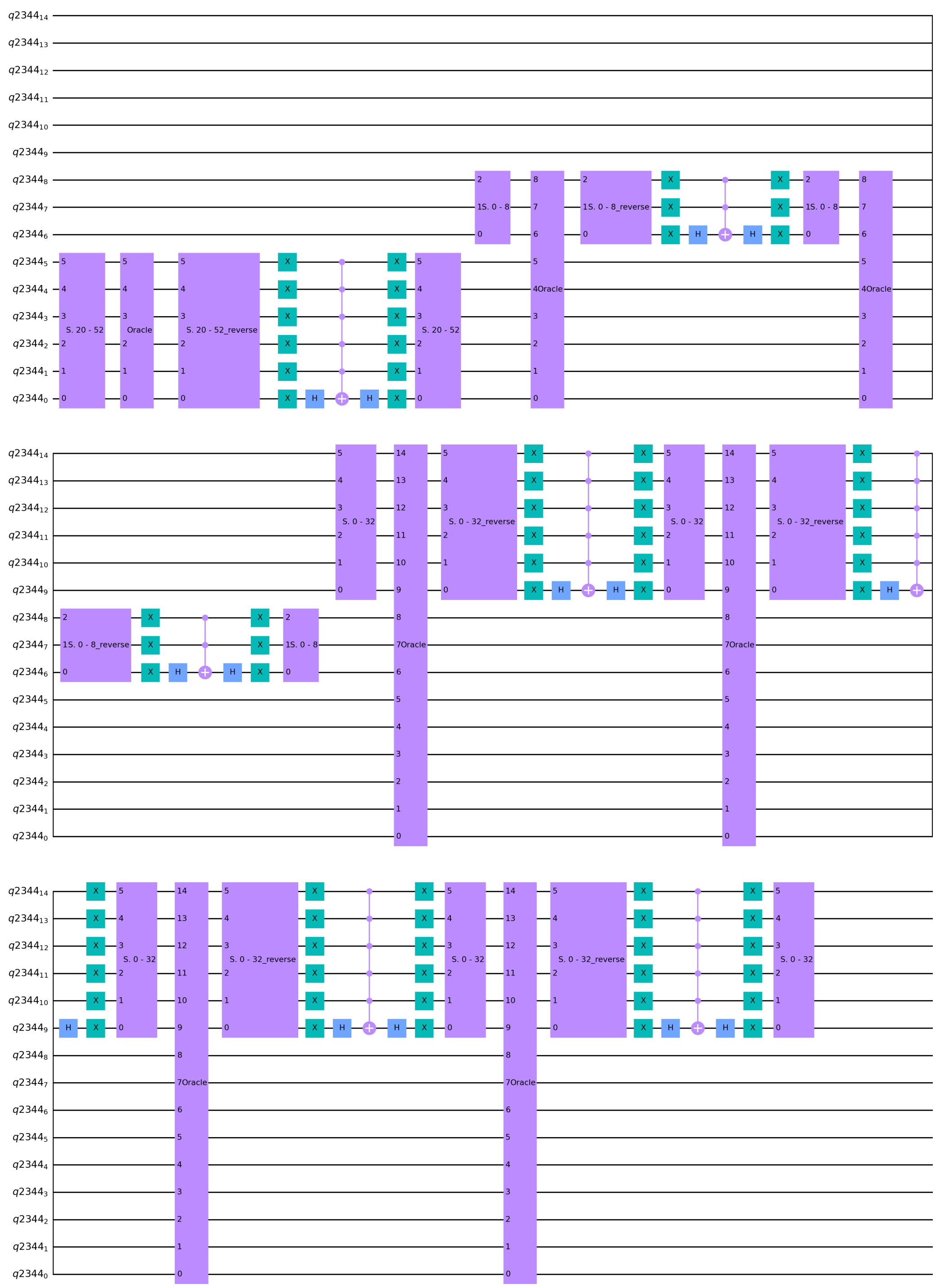


Figure 11 Quantum circuit of U (Bounded)

The whole quantum bounded structured circuit is:

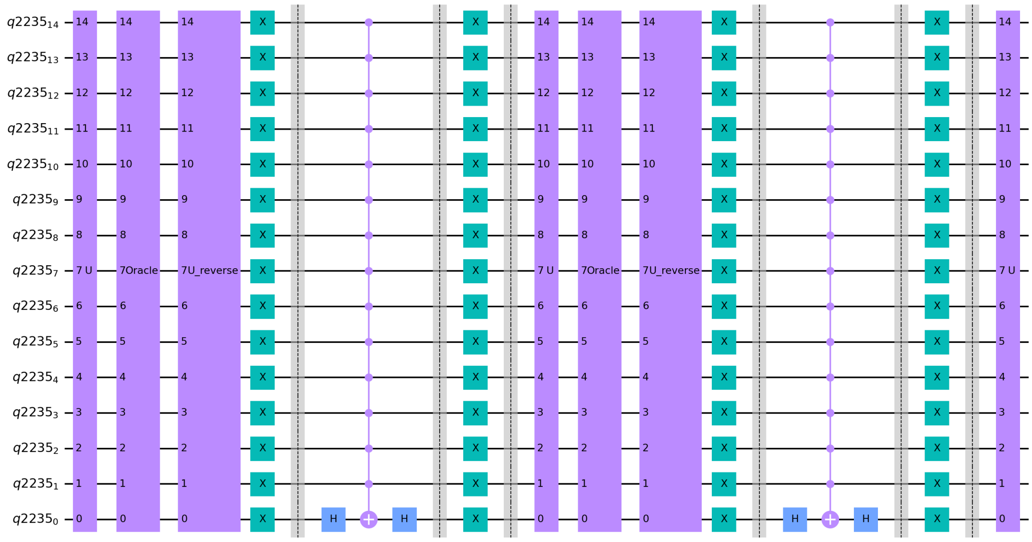


Figure 12 Structured search circuit (Bounded)

To finish with, . . . with an almost 1 probability the solution state will be:

2471	000100110100111	(0.0015115270238208174-8.518014884068187e-17j)
2472	000100110101000	(-0.01247009794652182+3.0649526650641827e-16j)
2473	000100110101001	(-0.012470097946521855+2.8206686330160475e-16j)
2474	000100110101010	(-0.012470097946521833+2.9487005406664363e-16j)
2475	000100110101011	(-0.964420308566682+4.2519036962786826e-14j)
2476	000100110101100	(-0.004534581071462367+1.0234478560557043e-16j)
2477	000100110101101	(-0.004534581071462441+1.0617475835458448e-16j)
2478	000100110101110	(-0.004534581071462499+1.0037436262505072e-16j)
2479	000100110101111	(0.0015115270238208118-9.474642203849064e-17j)
2480	000100110110000	(0.001511527023820837-9.705538552587256e-17j)

Figure 13 Probability amplitude of the solution state (Bounded)

References

1 N. Cerf, Lov Grover, and Colin Williams. Nested quantum search and np-hard problems. *Appl. Algebra Eng. Commun. Comput.*, 10:311–338, 05 2000. doi:10.1007/s002000050134.

2 Lov K Grover. A Fast Quantum Mechanical Algorithm for Database Search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 212–219, New York, NY, USA, 1996. Association for Computing Machinery. doi:10.1145/237814.237866.

3 Gui Long and Yang Sun. Efficient Scheme for Initializing a Quantum Register with an Arbitrary Superposed State. *Physical Review A*, 64, 10 2001. doi:10.1103/PhysRevA.64.014303.