Christian B. Mendl, Pedro Hack, Keefe Huang, Irene López Gutiérrez

**Tutorial 10** (Grover as a database search algorithm[1])
Grover's algorithm is sometimes referred to as a *database search algorithm*. In this tutorial we will examine how the algorithm could in principle be used to search in an unstructured database, and discuss the feasibility of this approach.

Assume we have a database containing $N = 2^n$ items, each of length $l$ bits: $\{d_1, d_2, \ldots, d_N\}$. We want to determine where a particular item, $s$, is in this database.

(a) Discuss how a classical computer (with a CPU and a memory) would approach this problem. How many queries to the memory are required on average? What is the worst-case scenario?

Now imagine we are given a "quantum processing unit" (QPU) containing four registers:

- An $n$ qubit register for the database index.
- An $l$ qubit register for our query $|s\rangle$.
- An $l$ qubit register for items loaded from the database, initialized as $|0\rangle$.
- A 1 qubit register initialized as $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

With this QPU, we can perform the following load operation: for an index $x$

$$|x\rangle\,|s\rangle\,|t\rangle\,|-\rangle \overset{\text{LOAD}}{\mapsto} |x\rangle\,|s\rangle\,|t \oplus d_x\rangle\,|-\rangle\,.$$

In particular, for $|t\rangle = |0\rangle$ the third register will contain $|d_x\rangle$. Then, the second and the third register are compared and, if they are the same, a bit flip is applied to the forth register.

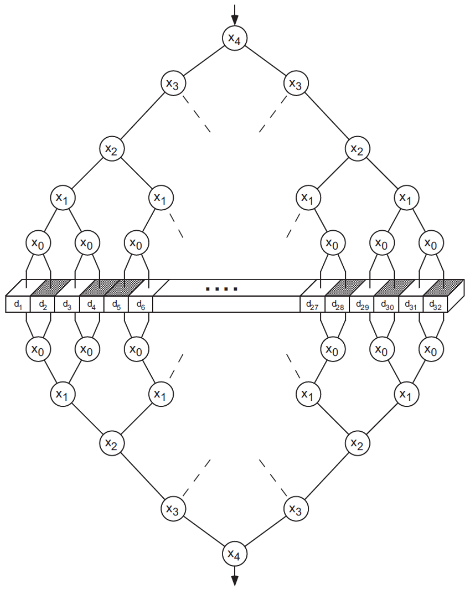
(b) What is the effect of this operation? What is its connection to Grover's algorithm?

Recall, however, that in Grover's algorithm we make use of superposition states. Thus, it may seem that in order to implement this we need a quantum memory besides our QPU. But in fact, we only need a classical memory which can be addressed by a quantum scheme.

The figure to the right shows a conceptual diagram of a 32 cell memory with a five qubit addressing scheme. The tree diagram represents the possible paths taken by an input qubit. At each node, the input qubit is sent left or right depending on the value of the qubit inside the circle. A superposition of paths is possible.

(c) What is the advantage of this scheme as opposed to a fully quantum memory?

(d) Discuss in general the practicality of this algorithm. Is the unstructured database problem common? Is the required hardware achievable?


$x_4$
$x_3$
$x_2$
$x_1$
$x_0$
$d_1$ $d_2$ $d_3$ $d_4$ $d_5$ $d_6$
....
$d_{27}$ $d_{28}$ $d_{29}$ $d_{30}$ $d_{31}$ $d_{32}$

## Solution

(a) The CPU must have enough memory to store a bit string of length $n$. One can then iterate through the indexes of the database entries. Each entry, $d_x$, is loaded from the memory into the CPU and the condition $d_x = s$ is checked until satisfied. On average, this would require $N/2$ queries to the memory. The worst case scenario is when $s$ is the very last item checked, i.e., one performs $N$ queries.

[1] M. A. Nielsen, I. L. Chuang: *Quantum Computation and Quantum Information*. Cambridge University Press (2010), section 6.5

(b) A conditional bit flip on the forth register results in

$$|x\rangle\,|s\rangle\,|d_x\rangle\,|-\rangle \mapsto \begin{cases} |x\rangle\,|s\rangle\,|d_x\rangle\,\frac{1}{\sqrt{2}}(|1\rangle - |0\rangle) = -\,|x\rangle\,|s\rangle\,|d_x\rangle\,|-\rangle\,, & \text{if } s = d_x \\ |x\rangle\,|s\rangle\,|d_x\rangle\,|-\rangle & \text{otherwise} \end{cases}$$

One can then perform the load operation again to reset the third qubit to zero, since $|d_x \oplus d_x\rangle = |0\rangle$. In summary, when the second and third register are the same, a phase of $-1$ is introduced. This is precisely the oracle in Grover's algorithm. Grover's algorithm will then only need $\mathcal{O}(\sqrt{N})$ such load and flip operations.

(c) Given that quantum states are very sensitive to noise, for long term storage it is preferable to use classical hardware.

(d) Unstructured databases are not common. Typically, the database is designed with some kind of structure to optimize the number of queries to memory needed (e.g., by storing items in alphabetical order). In terms of hardware, the quantum addressing scheme depicted above uses $\mathcal{O}(N\log(N))$ "switches" to send the input qubit left or right. At this point in time, such a setup would be too expensive, and not advantageous compared to distributed classical computing approaches.