Book Title
Book Editors
IOS Press, 2003

A Quantum Algorithm for Closest Pattern Matching

Paulo Mateus [a] and Yasser Omar [b,1]
[a] *Centro de Lógica e Computação, Instituto Superior Técnico*
[b] *CEMAPRE, ISEG, Universidade Técnica de Lisboa*

Abstract.
We propose a quantum algorithm for closest pattern matching which allows us to search for as many distinct patterns as we wish in a given string (database), requiring a query function per symbol of the pattern alphabet. This represents a significant practical advantage when compared to Grover's search algorithm as well as to other quantum pattern matching methods [7], which rely on building specific queries for particular patterns. Our method makes arbitrary searches on long static databases much more realistic and implementable. Our algorithm, inspired by Grover's, returns the position of the closest substring to a given pattern with high probability in $O(\sqrt{N})$ queries, where N is the size of the string. The analysis and quantum circuit construction of our algorithm are presented in [4].

Keywords. Quantum Algorithms, Database Search.

Search in databases is nowadays a common and fundamental application in computer science, one that we use daily to find a word in a text, or a site in Google. Currently we are also living the quantum information revolution, where the idea to encode information in quantum systems offers us a radically new type of information which allows for much securer communications and much faster computations than what we were able to achieve so far using (now-called) classical information [6]. There have been a few quantum algorithms proposed (the most significant probably being Shor's efficient factorization algorithm [8] of 1994, solving a problem that classically is believed to be intractable) even though the construction of a scalable quantum computer is still a challenge, presently being tackled with a plethora of different technologies [1]. Yet, should quantum computation become a reality, there is still no implementable efficient quantum algorithm to search a given database (in the current models of quantum computer), despite Grover's celebrated quantum search algorithm proposed in 1996 [3]. Grover's work, which now constitutes a paradigm for quantum search algorithms, offers a quadratic speed-up in query complexity (i.e. calls of a query function) when compared to the classical case. However, in the real execution of these search algorithms, we must distinguish the *compile time* and the *run time*. The compile time is essentially the construction of the query function on which the

[1]Correspondence to: Departamento de Matemática, Instituto Superior de Economia e Gestão, P-1200-781 Lisbon, Portugal; E-mail: yasser at qubit.org.

algorithm relies to identify the element being searched. But this construction is, in general, not a negligible task. In particular, for database search, we must go through all the database elements to build the so-called oracle, so that we can then implement the search. Note that this makes the quantum search irrelevant in practical terms, since you need to know the solution to run it. Moreover, given a query function built to find a particular element, it can only be used again to find that very same element. The search for a different item in the same database would require building a new specific query function. All this represents a serious obstacle to the application of current search algorithms [1].

To address this problem, we propose a quantum algorithm for pattern matching which allows us to search for as many distinct patterns as we wish in a given unsorted string (database), and moreover returns the position of the closest substring to a given pattern with high probability in $O(\sqrt{N})$ queries, where N is the size of the string. This means that the time to find the *closest* match (a much harder problem than to find the *exact* match, as we shall see) does not depend on the size of the pattern itself, a result with no classical equivalent. Another crucial point is that our quantum algorithm is actually useful and implementable to perform searches in (unsorted) databases. For this, we introduce a query function per symbol of the pattern alphabet, which will require a significant (though clearly efficient) pre-processing, but will allow us to perform an arbitrary amount of different searches in a static database. A *compile once, run many* approach yielding a new search algorithm that not only settles the previously existing implementation problems, but even offers the solution of a more general problem, and with a very interesting speed-up. The analysis of the algorithm as well as the recipe for its full implementation as a quantum circuit are given in [4].

In the classical setting, the best algorithm, to our knowledge, for the closest substring problem takes $\mathrm{O}(MN)$ queries where M is the size of the pattern. This result follows from adapting the best known algorithm for approximate pattern matching [5], which takes $\mathrm{O}(eN + M)$ where e is the number of allowed errors. One should not compare the closest match to (exact) pattern match, where the problem consists in determining if a certain word (pattern) is a substring of a text. For exact pattern matching it is proven that the best algorithm can achieve $\mathrm{O}(M + N)$ [5]. However, in practical cases where data can mutate over time, like DNA, or is store in a faulty systems, the closest match problem is a much more relevant, since sometimes, only approximates of the pattern exist, but nevertheless need to be found.

Our algorithm is based on the modified Grover search algorithm proposed in [2] for the case of multiple solutions. It uses the techniques originally introduced by Grover [3]: a query function that marks the state encoding the database element being searched by changing its phase; followed by an amplitude amplification of the marked state. The state can be detected with high probability by iterating this process $\sqrt{N}$ times where N is the size of the database.

[1] Yet, Grover's algorithm can be extremely useful and represent an effective speed-up for other search problems, as for instance in checking if the elements of a list are solutions of a given NP-complete problem, where the query function plays the role of the verifier and can be easily implemented.

Let us now describe our closest pattern matching algorithm. Given a string w of size N over an alphabet Σ, we want to know if a certain pattern p of size M occurs in w, or at least obtain the closest match to p in w. In particular we want to find the position $i \in \{1, \ldots, N\}$ where a certain symbol of p occurs in w. To this end, we encode position i in a unit vector $|i\rangle$ of a Hilbert space $\mathcal{H}$ of dimension N (where the set $B = \{|1\rangle, \ldots, |N\rangle\}$ constitutes an orthonormal basis of $\mathcal{H}$).

The initial state of the total system reflects the fact that we want the second symbol of p to occur just after the first, and the third to occur just after the second, and so on. For this reason we consider the following initial state, which consists of a uniform superposition of all possible states fulfilling this property: $|\psi_0\rangle = \sum_{k=1}^{N-M+1} \frac{1}{\sqrt{N-M+1}} |k\rangle$,

To perform the search, we now need to define a query function q_σ for each symbol σ of the alphabet Σ. We will thus have $|\Sigma|$ different query functions[2], each given by:

$$q_\sigma(i) = \begin{cases} 1 & \text{if the } i\text{-th letter of } w \text{ is } \sigma \\ 0 & \text{otherwise} \end{cases} . \tag{1}$$

As in Grover's algorithm, we want to use the query to mark states where there is a match for the individual symbol, in particular by shifting the phase of the respective state, as given by the following unitary transformation: $U_\sigma|k\rangle = (-1)^{q_\sigma(k)}|k\rangle$, where $|k\rangle \in B$. In our algorithm, we start by randomly choosing an element of the pattern p. Imagine we pick the j-th element of p ($j \in \{1, \ldots, M\}$). Then, whenever the $(k + j - 1)$-th letter of the string matches the j-th element of the pattern, we mark state $|k\rangle$ by shifting its phase. To produce this marking, we define the transposition operators T_j given by: $T_j|k\rangle = |k + j - 1 \bmod N\rangle$. The desired phase shifting is then obtained with $T_j^{-1} U_{p_j} T_j$.

On average, a position with a partial match, say of M' out of M matches of individual symbols, will have the phase shift applied $\frac{M'}{M}$ times. Note that the more matches we obtain, the more the phase will be shifted, and consequently the more the amplitude will be amplified. Observe that for a given string there might be full and partial matches, leading to larger and smaller amplitude amplifications respectively.

The amplitude amplification is obtained by applying the usual Grover diffusion D to the total state, over the $N - M + 1$ relevant basis:

$$D = (2(|\psi_0\rangle\langle\psi_0|) - I) + 2\left(\sum_{k=N-M+2}^{N} |k\rangle\langle k|\right), \tag{2}$$

where I is the identity operator of dimension N.

The algorithm is then constituted by iterating the phase shift induced by the query followed by amplitude amplification. The final step is to measure the state of a symbol of the pattern over the predefined basis B, yielding the position of the closest match of the pattern in the string with a high probability.

[2]Note that, in view of implementations, it is straightforward to define the corresponding unitary operator Q_σ, acting over $\mathcal{H} \otimes \mathcal{H}_2$ (where $\mathcal{H}_2$ is the Hilbert space of dimension 2), as follows: $Q_\sigma(|i\rangle \otimes |b\rangle) = |i\rangle \otimes |q_\sigma(i) \oplus b\rangle$, where $|i\rangle$ encodes position i and $|b\rangle$ is a auxiliary qubit.

In summary, the algorithm will be as follows:

Input: $w \in \Sigma^*$ and $p \in \Sigma^*$
Output: $m \in \mathbb{N}$
Quantum variables: $|\psi\rangle \in \mathcal{H}(\{1, \ldots, N\})$
Classical variables: $r, i, j \in \mathbb{N}$

1. choose $r \in [0, \llcorner\sqrt{N - M + 1}\lrcorner]$ uniformly,
2. set $|\psi\rangle = \sum_{k=1}^{N-M+1} \frac{1}{\sqrt{N-M+1}}|k\rangle$;
3. for $i = 1$ to r
   (a) choose $j \in [1, M]$ uniformly
   (b) set $|\psi\rangle = T_j^{-1} U_{p_j} T_j |\psi\rangle$;
   (c) set $|\psi\rangle = D|\psi\rangle$
4. set m to the result of the measurement of $|\psi\rangle$ over the base $\{|1\rangle, \ldots, |N\rangle\}$.

The analysis of the algorithm is inspired by [2] and is detailed in [4], where we conclude that, in the quantum circuit model, our algorithm has an efficient compile time of $O(N \log^2(N) \times |\Sigma|)$ and a total run time of $O(N^{3/2} \log^2(N) \log(M))$.

In summary, we have presented a quantum algorithm for closest pattern matching that not only makes the quantum search in (long unsorted) static databases realistic, but even interesting, as it is a strong candidate to offer a faster solution than what is known classically for this important problem. Based on a *compile once, run many times* approach, our algorithm allows for an arbitrary amount of different searches on the same string, and offers a query complexity of $O(\sqrt{N})$ in the most relevant limit where the size M of the pattern is much smaller than the size N of the database. In [4], we give the details on how to obtain the full quantum circuit that implements our algorithm, thus offering an oracle-based quantum algorithm ready to be implemented.

The authors would like to thank A. Ambainis, L. Grover, E. Kashefi, J.I. Latorre, U. Vazirani and A. Sernadas for useful discussions, and gratefully acknowledge the support from Fundação para a Ciência e a Tecnologia (Portugal), namely through programs POCTI/POCI and project POCI/MAT/55796/2004 QuantLog, partially funded by FEDER (EU).

References

[1] ARDA, Quantum Information Science and Technology Roadmap, *qist.lanl.gov* .
[2] M. Boyer, J. Brassard, P. Høyer and A. Tapp, *Fort. Phys.*, 46(1-5):493–505, 1998.
[3] L. Grover, *Phys. Rev. Lett.*, 79:325–328, 1997.
[4] P. Mateus and Y. Omar, *arXiv.org*, quant-ph/0508237, 2005.
[5] G. Navarro, *ACM Comput. Surv.*, 33(1):31–88, 2001.
[6] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, UK, 2000.
[7] H. Ramesh and V. Vinay, *J. Discrete Algorithms*, 1(1):103–110, 2003.
[8] P. W. Shor, *SIAM J. Comp.*, 26(5):1484–1509, 1996.