# Quantum search of partially ordered sets

**Ashley Montanaro**

Department of Computer Science, University of Bristol, UK

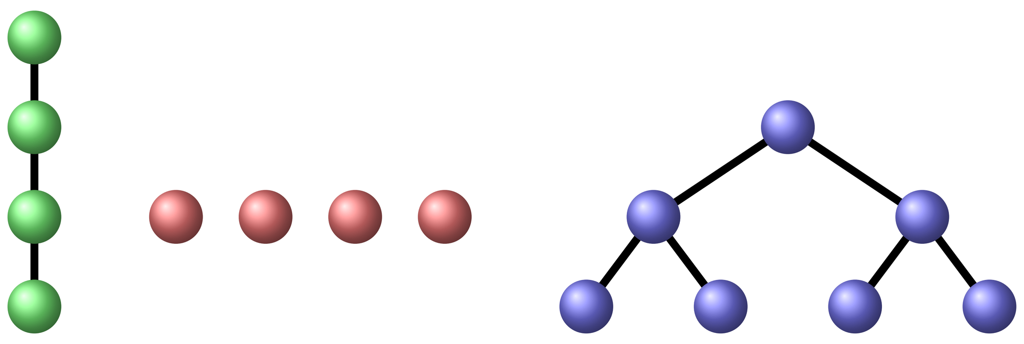
montanar@cs.bris.ac.uk

## Abstract


- *We investigate the generalisation of quantum search of unstructured and totally ordered sets to search of partially ordered sets (posets).*
- *In two models, we show that quantum algorithms can achieve at most a quadratic improvement in query complexity over classical algorithms, up to logarithmic factors; we also give quantum algorithms that almost achieve this bound.*
- *In one model, we give an almost optimal quantum algorithm for searching forest-like posets.*
- *In the other, we give an optimal $O(\sqrt{n})$ quantum algorithm for searching posets derived from $n \times n$ arrays sorted along rows and columns.*
- *This leads to an optimal $O(\sqrt{n})$ quantum algorithm for finding the intersection of two sorted lists of $n$ integers.*


## Partially ordered sets

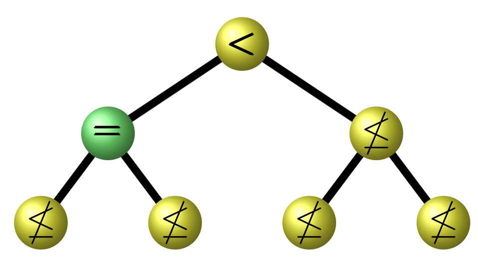
- A **partial order** on a set $S$ is a relation $\leq$ such that, for $a, b, c \in S$, $a \leq a$, $(a \leq b) \wedge (b \leq a) \Rightarrow a = b$, and $(a \leq b) \wedge (b \leq c) \Rightarrow a \leq c$.
- Posets can be expressed by **Hasse diagrams**:



## Two models for search

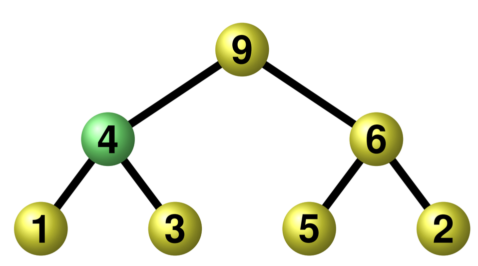
In the **abstract** model [4]:

- We are searching for an unknown "marked" element $a$.
- Querying element $x$ returns $<$, $=$, $\not\leq$ depending on whether $a < x$, $a = x$, or either $a > x$ or $a$ and $x$ are incomparable.




In the **concrete** model [6]:

- Each element $s \in S$ stores an unknown integer $S[s]$.
- For any given integer $a$, we want to find the unique $s$ such that $S[s] = a$.




## General bounds

**Theorem.** *Let $S$ be an $n$-element poset, and let $D(S)$, $Q_E(S)$ and $Q_2(S)$ be the number of queries required for an exact classical, exact quantum, or bounded-error quantum (respectively) algorithm to find the marked element in $S$. Then, in the abstract model,*

$$D(S) = O(Q_2(S)^2 \log n)$$
$$Q_2(S) = O(\sqrt{D(S)} \log n \sqrt{\log \log n})$$

*and in the concrete model,*

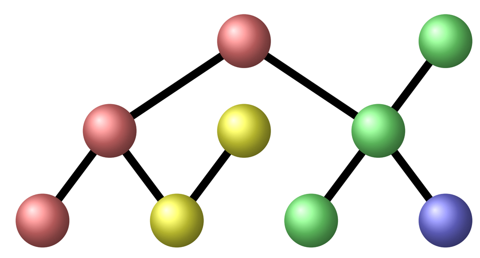
$$D(S) = O(Q_2(S)^2 \log n)$$
$$Q_E(S) = O(\sqrt{D(S)} \log n)$$

Proof idea (abstract model):

- Reduce poset search to an **oracle identification** problem (finding marked element ⇔ identifying an oracle).
- Lower bound: from a result of Servedio and Gortler [7]. Upper bound: from a result of Atici and Servedio [3].

Proof idea (concrete model):

- Lower bound: from the bound of Ambainis on inverting a permutation [2].
- Upper bound: from **Dilworth's Theorem** [5] giving a decomposition of posets into **chains** (sets of comparable elements). Perform binary search on each chain in quantum parallel.



## Recursive quantum search

**Theorem.** *Let $P_n$ be the problem of searching an abstract database, parametrised by an abstract size $n$, for a known element which may or may not be in the database. Let $T(n)$ be the time required for a bounded-error quantum algorithm to solve $P_n$, i.e. to find the element, or output "not found". Let $P_n$ satisfy the following conditions:*

- *If $n \leq n_0$ for some constant $n_0$, then there exists an algorithm to find the element, if it is contained in the database, in time $T(n) \leq t_0$, for some constant $t_0$.*
- *If $n > n_0$, then the database can be divided into $k$ sub-databases of size at most $\lceil n/k \rceil$, for some constant $k > 1$.*
- *If the element is contained in the original database, then it is contained in exactly one of these sub-databases.*
- *Each division into sub-databases uses time $f(n)$, where $f(n) = O(n^{1/2-\epsilon})$ for some $\epsilon > 0$.*

*Then $\boldsymbol{T(n) = O(\sqrt{n})}$.*

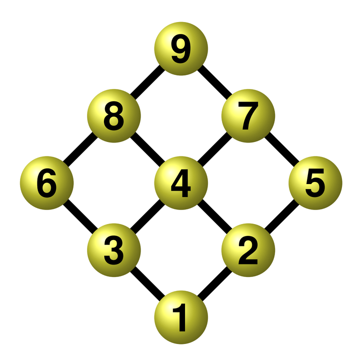
**Proof idea:**

- Based on a result of Aaronson and Ambainis [1] on quantum search of spatial regions.
- Split the database some number of times and pick a sub-database at random, then recurse.
- Perfom some number of iterations of amplitude amplification at each recursive step...
- ...then use amplitude amplification on the whole algorithm.

## 2-dimensional arrays

An interesting poset: an $n \times n$ array of distinct integers that are increasing along rows and columns.

| | | |
|---|---|---|
| 1 | 3 | 6 |
| 2 | 4 | 8 |
| 5 | 7 | 9 |




- We give an optimal quantum algorithm which finds an integer in such an array using $O(\sqrt{n})$ queries.
- Implies an optimal $O(n^{(d-1)/2})$ algorithm to find an integer in a $d$-dimensional $n \times n \times \cdots \times n$ array.
- Also implies an optimal $O(\sqrt{n})$ algorithm to find the intersection of two sorted lists of $n$ integers.
- The quantum algorithm is based on an asymptotically optimal $O(n)$ recursive classical algorithm.
- We use the **recursive quantum search theorem** above to convert this to an optimal quantum algorithm.

Sketch of the classical algorithm:

- Perform binary search on the central row (or column) of the array.
- Can discard at least half the elements, leaving two rectangular subarrays.
- Call this algorithm recursively on these subarrays.

**Example:** (where green: integer to search for, yellow: not searched yet, blue: currently being searched, red: discarded)

| | | | | |
|---|---|---|---|---|
| 1 | 3 | 5 | 10 | 13 |
| 2 | 4 | 7 | 11 | 14 |
| 6 | 8 | 9 | 15 | 21 |
| 12 | 16 | 17 | 20 | 24 |
| 18 | 19 | 22 | 23 | 25 |

| | | | | |
|---|---|---|---|---|
| 1 | 3 | 5 | 10 | 13 |
| 2 | 4 | 7 | 11 | 14 |
| 6 | 8 | 9 | 15 | 21 |
| 12 | 16 | 17 | 20 | 24 |
| 18 | 19 | 22 | 23 | 25 |

## Conclusions

- We have given general upper and lower bounds on quantum search of partially ordered sets, in two different models.
- The non-query transformations used by the algorithms given here are efficiently implementable.
- Given a poset $S$ to be searched, quantum circuits for these algorithms can be produced in time polynomial in the size of $S$.

**Open questions:**

- In the abstract model, is there a general lower bound of $Q_2(S) = \Omega(\log n)$?
- Can the logarithmic factors in the quantum upper bounds in both models be improved, e.g. by being changed into additive terms?
- In the concrete model, could the 2D search algorithm be extended to arrays that may contain duplicate elements?

## References


[1] S. Aaronson, A. Ambainis. Quantum search of spatial regions. *Theory of Computing* 1, pp. 47-79, quant-ph/0303041, 2005.

[2] A. Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences* 64, pp. 750-767, quant-ph/0002066, 2002.

[3] A. Atici, R. Servedio. Improved bounds on quantum learning algorithms. *Quantum Information Processing* 4, pp. 355-386, quant-ph/0411140, 2005.

[4] Y. Ben-Asher, E. Farchi, I. Newman. Optimal search in trees. *SIAM J. Comput.* 28, pp. 2090-2102, ECCC TR96-044, 1999.

[5] R. P. Dilworth. A decomposition theorem for partially ordered sets. *The Annals of Mathematics* 51, pp. 161-166, 1950.

[6] N. Linial, M. Saks. Searching ordered structures. *Journal of Algorithms* 6, pp. 86-103, 1985.

[7] R. A. Servedio, S. J. Gortler. Quantum versus classical learnability. *Proc. CCC'01*, pp. 138-148, quant-ph/0007036, 2001.


University of Bristol

quant-ph/0702196