On the optimality of Grover's algorithm

Nikolay Nahimov, Alexander Rivosh

EIROPAS SOCIĀLAIS FONDS
EIROPAS SAVIENĪBA
IEGULDĪJUMS TAVĀ NĀKOTNĒ

Eiropas Sociālā fonda projekts
“Datorzinātnes pielietojumi un tās saiknes ar kvantu fiziku”
Nr.2009/0216/1DP/1.1.1.2.0/09/APIA/VIAA/044

Unstructured search

- Search in the unsorted array

0	0	0	1	...	0	0

- We have a function given as a black-box:

$$f(x) : \{0,1\}^n \rightarrow \{0,1\}$$

- The unstructured search problem is to find $x \in \{0,1\}^n$ such that $f(x) = 1$, or to conclude that no such x exists.

Classical case

- Unsorted array

0	0	0	1	...	0	0

- Trivial algorithm: sequential check

Best case: 1 step

Worst case: N steps

Average case: N/2 steps

- “Clever algorithm”: use another fixed array element sequence. This does not change anything.

Probabilistic case

- Unsorted array

0	0	0	1	...	0	0

- Trivial algorithm: check k random array elements.

Probability of finding a solution p(k) = k/N

Run the algorithm until it finds a solution

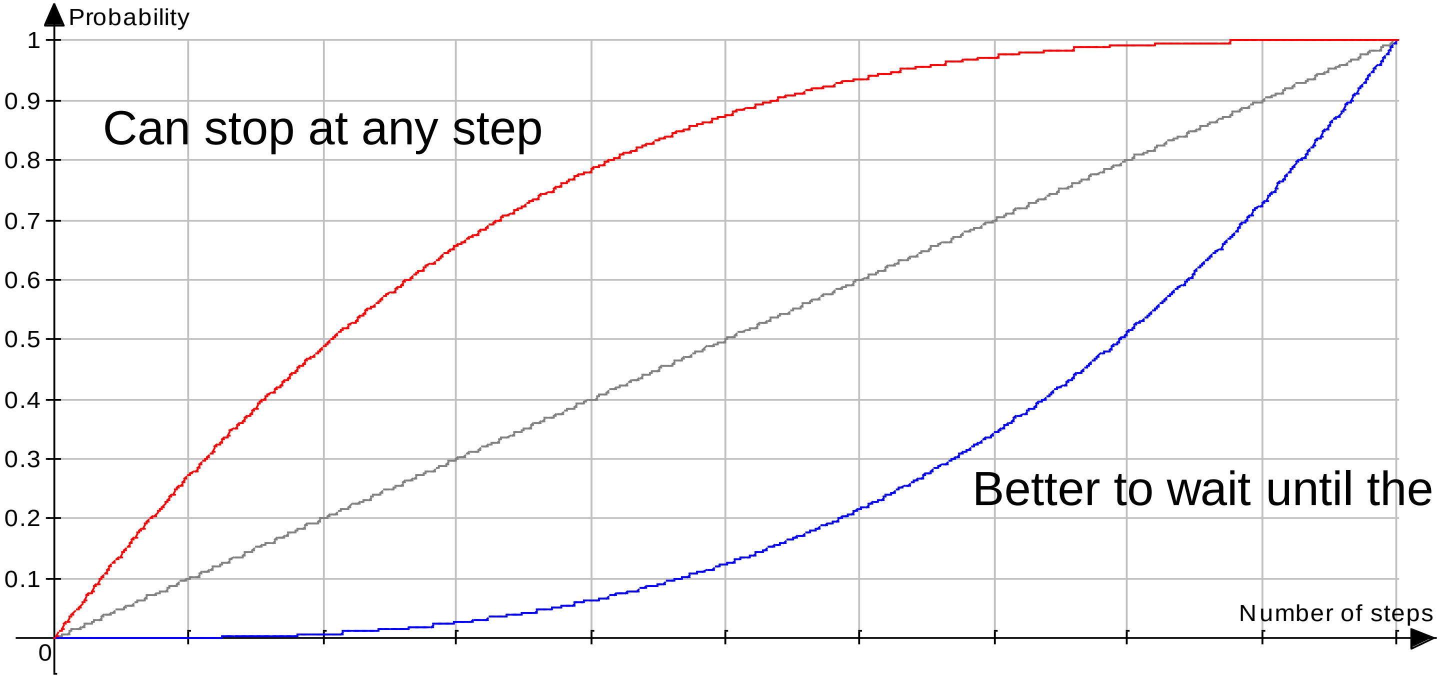
- How many we need to rerun the algorithm ?
- What is the optimal value of k ?

Probabilistic algorithms

- We have a probabilistic algorithm, which finds a solution with probability p.
- How many times we need to run the algorithm to find a solution with probability 1 ?
- On the average we should run the algorithm 1/p times.

Probabilistic algorithms

- If after k steps the probability of finding a solution is p(k), the average running time of the algorithm is k / p(k).



Probabilistic case

- The algorithm: check k random array elements.
- Probability of finding a solution p(k) = k/N grows linearly with k.
- To find a solution with probability 1 we should repeat the algorithm 1/p = 1 / (k/N) times on the average.
- The average running time = k / (k/N) = N.

  Does not depend on k.

Quantum case

- Unsorted array

0	0	0	1	...	0	0

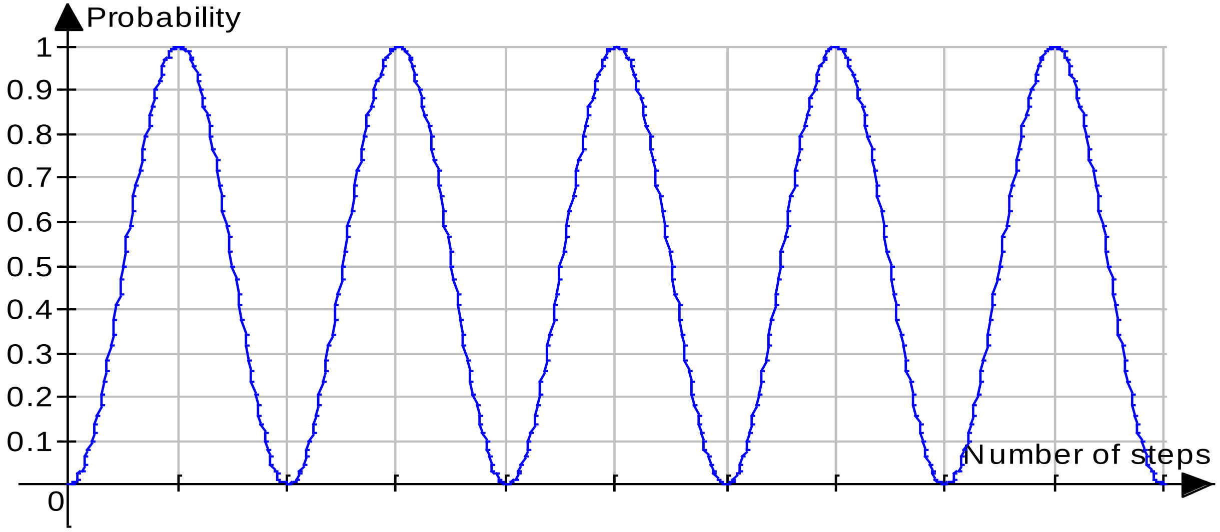
- Classical case: optimal algorithm performs O(N) checks.

- Quantum case: optimal algorithm performs O($\sqrt{N}$) checks.

  Let M be the number of steps of the algorithm.

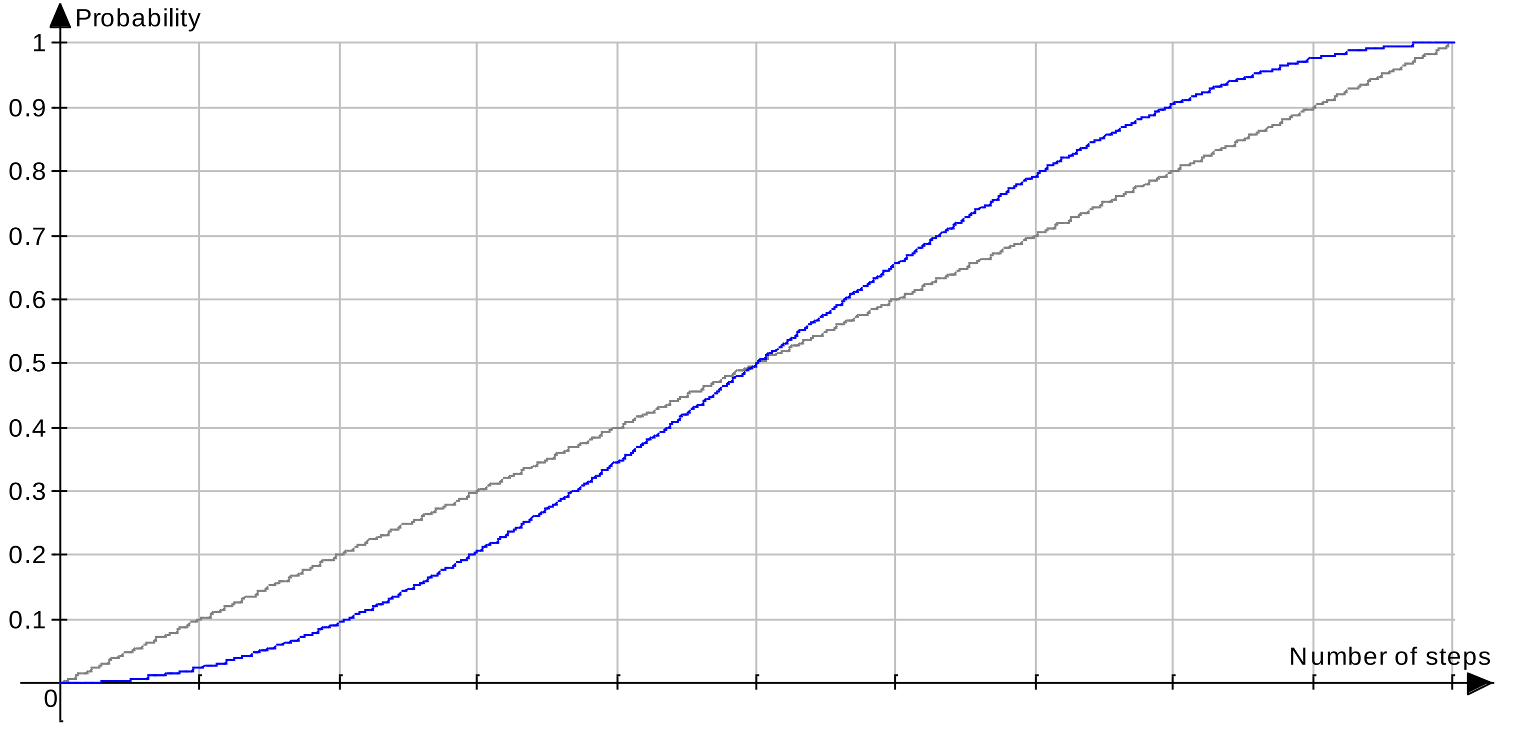
Quantum case

- The probability of finding a solution after k steps is $\sin^2(\pi k / 2M)$



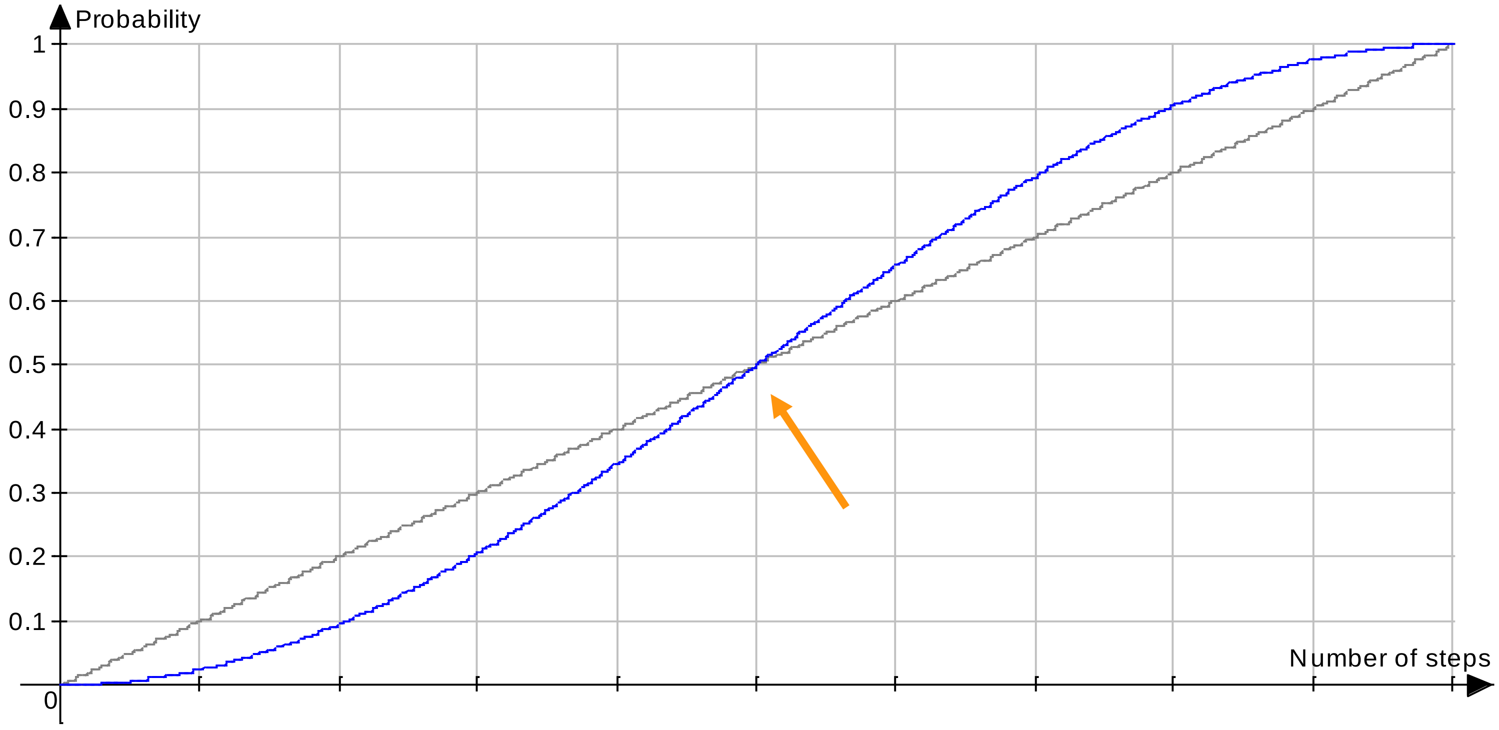
Quantum case

- If we stop the computation after k steps the average running time of the algorithm is k / p(k).



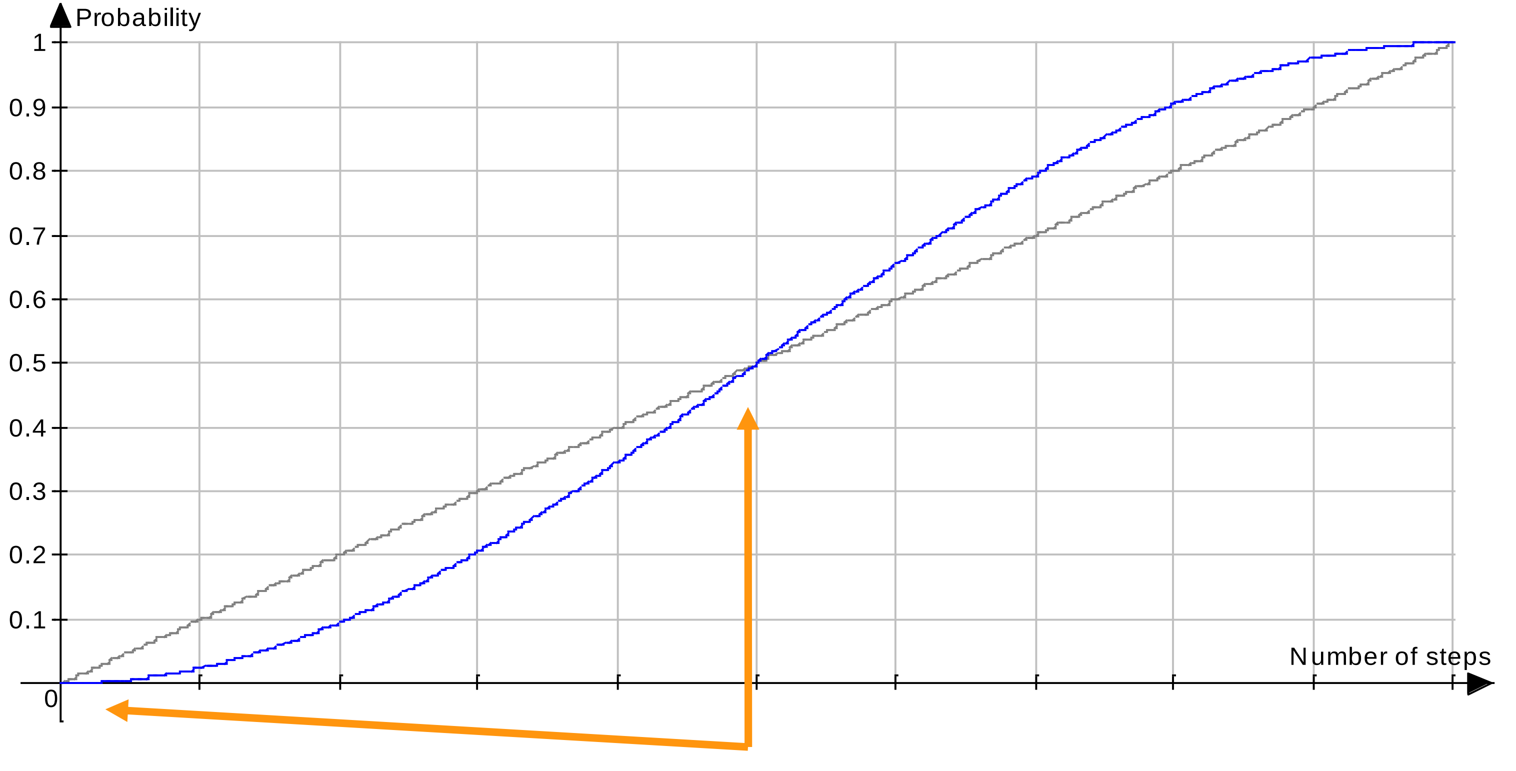
Quantum case

- If p(k) = k/M, the average running time is M.



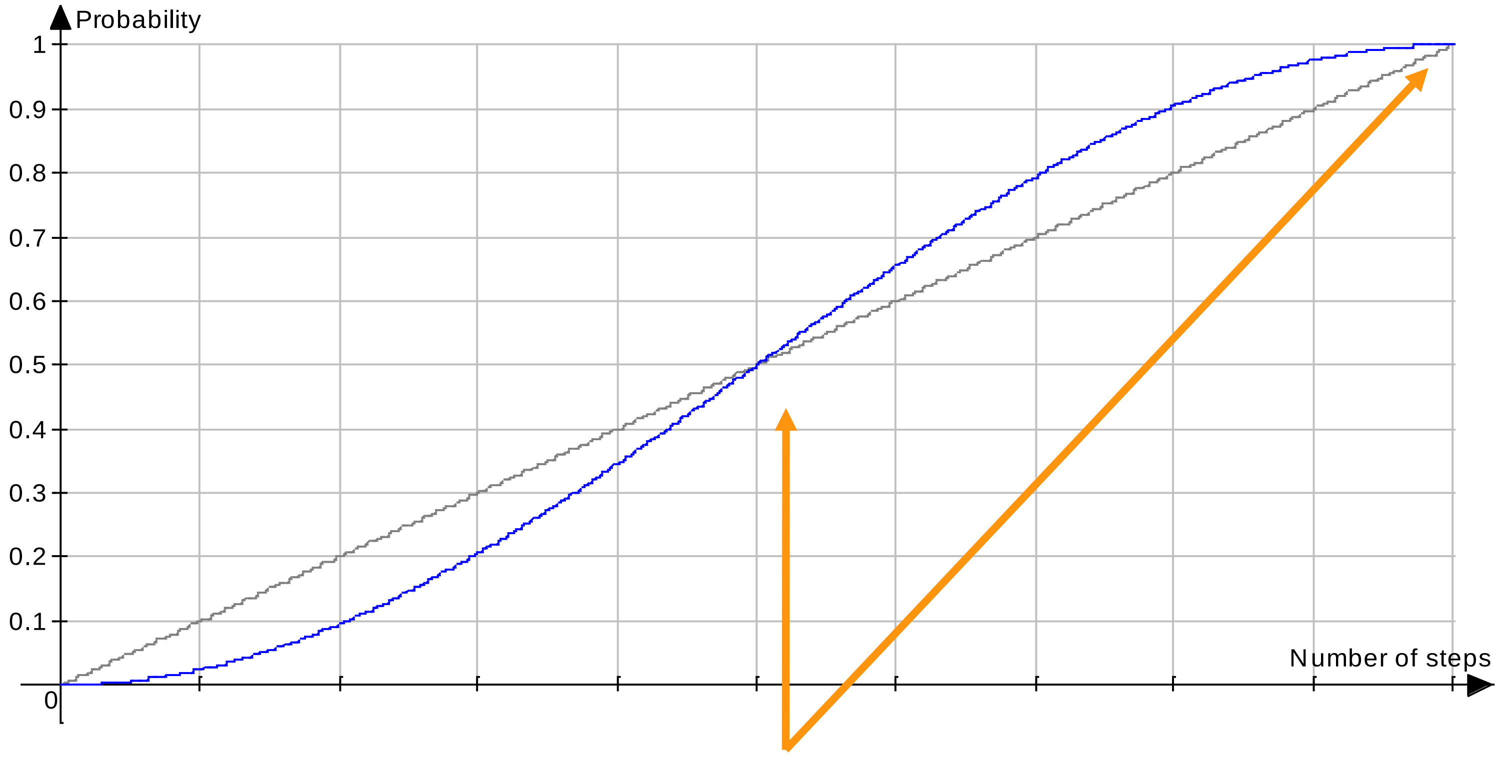
Quantum case

- If p(k) < k/M, the average running time > M.



Quantum case

- If p(k) > k/M, the average running time < M.



Quantum case

- The optimal moment to end the computation is the minimum of the $k/p(k) = k / \sin^2(\pi k / 2M)$ function.
- Calculation gives $k \approx 0.74202$ and the average running time $k/p(k) \approx 0.87857$.
- That is the average number of steps can be reduced by approximately 12.14%.

Conclusions

- The average number of Grover's algorithm steps can be reduced by approximately 12.14%.

- The same argument can be applied to a wide range of other quantum query algorithms, such as amplitude amplification, some variants of quantum walks and NAND formula evaluation, etc.

Thank you !