Tutorial Note: Quantum Machine Learning

by Xun Gao (Tsinghua University, IIIS)

June 29, KITS-ML2017

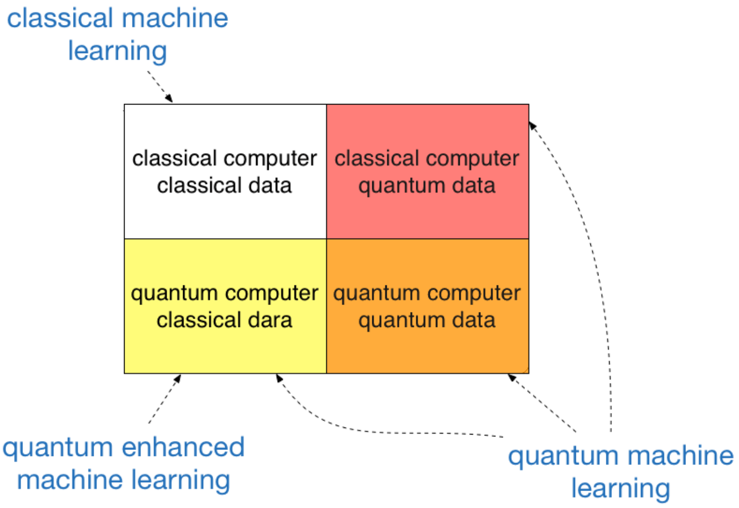
classical machine learning
classical computer classical data
classical computer quantum data
quantum computer classical dara
quantum computer quantum data
quantum enhanced machine learning
quantum machine learning

Figure 1.

We only focus on quantum enhanced machine learning: using quantum algorithm to solve traditional machine learning problem.

In this tutorial, we only give several simple examples. It's far away from a complete discussion of current research. The discussion is **lack of rigorous mathematical treatment**. Because we just want to convey the basic idea and get a feeling how quantum machine learning works. We only focus on the quantum algorithm which is expected to provide exponential speed-up. For more detail, please read the following references and the reference list in them:

- A recent review article on quantum machine learning: Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., & Lloyd, S. (2016). Quantum Machine Learning. arXiv preprint arXiv:1611.09347.
- The challenge of quantum machine learning: Aaronson, S. (2015). Read the fine print. Nature Physics, 11(4), 291-293.
- A brief introduction of quantum machine learning for general audience: Schuld, M. (2017). A Quantum Boost for Machine Learning. Physics World
- Quantum algorithm for solving linear equation (HHL algorithm): Harrow A W, Hassidim A, Lloyd S. Quantum algorithm for linear systems of equations. Physical review letters, 2009, 103(15): 150502.

1 A Brief Introduction of Quantum Computation

1.1 Classical Computation

A classical data can be described by a vector:

$$\boldsymbol{v} = (p_0, \cdots, p_{N-1}) \text{ where } N = 2^n, n \text{ is the number of bits.}$$

p_i is the probability to be the deterministic state $i = i_1 \cdots i_n$ (binary form of integer i), so

$$\|\boldsymbol{v}\|_1 = p_0 + \cdots + p_{N-1} = 1,$$

which means the l_1-distance is normalized.

An elementary operation for classical (probabilistic) computation is the operation preseving l_1-distance and acting non-trivially on constant number of bits. As shown in the following figure, the operation is a Markov process of which transtion probability is decribed by a coniditonal probability $p(kl|ij)$.

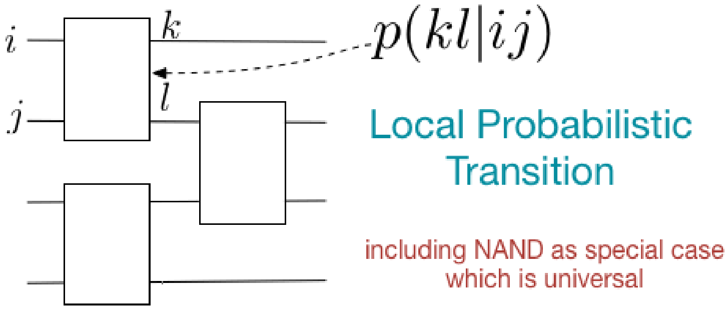
i
j
k
l
$p(kl|ij)$
Local Probabilistic Transition
including NAND as special case which is universal

Figure 2.

This figure shows a cirucit representation of classical computation which includes the ordinary logic circuit as its special case.

1.2 Quantum Computation

A quantum data can also be described by a vector:

$$|v\rangle = (\psi_0, \cdots, \psi_{N-1}) = \psi_0|0\rangle + \cdots + \psi_{N-1}|N-1\rangle,$$

In order to get useful information, we need to do measurement. For example, we measure the state on the $\{|0\rangle, \cdots, |N-1\rangle\}$ basis. With probability $|\psi_i|^2$, the outcome of measurement is $\psi_i/|\psi_i|\,|i\rangle$ (this is the collapse of wave function, this property is useful in quantum cryptography but puts strong limitations on quantum computation). So

$$\||v\rangle\|_2 = |\psi_0|^2 + \cdots + |\psi_{N-1}|^2 = 1 \,\text{or simply}\, \langle v|v\rangle = 1.$$

which means the l_2-distance is normalized.

An elementary operation for quantum computation is the operation preserving l_2-distance and acting non-trivially only on constant number of qubits. As shown in the following figure, the operation is a local unitary.

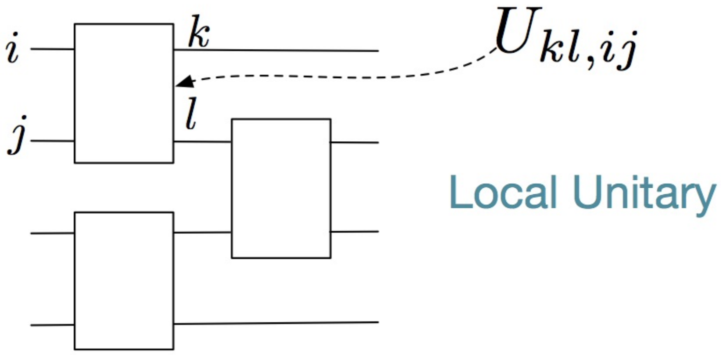

Figure 3.

This figure shows a quantum computation model: the quantum circuit model. After polynomial number of local unitaries, we need to do measurement to get useful information.

1.3 Quantum Parallelism?

Simply saying "quantum computer is powerful because quantum computer can do different computations at the same time or the so called quantum parallelism" can easily lead to misunderstanding. Here we give an example.

Suppose we have a unitary: $U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$, where $\oplus$ is bit-wise XOR here. We write the (classical) computation in this way since unitary is reversible thus the output state must contain all the information of input state. We usually use U_f in the followng way:

$$U_f|x\rangle|0\rangle = |x\rangle|f(x)\rangle$$

which means computing $f(x)$ and restore it in the second register. We apply it to a superposition state:

$$U_f\frac{\sum_x|x\rangle}{\sqrt{N}}|0\rangle = \frac{\sum_x|x\rangle|f(x)\rangle}{\sqrt{N}}$$

in order to get useful information, we have to do measurement. With probability $1/N$, we get the outcome $|x'\rangle|f(x')\rangle$ for some x'. In order to get another $f(x'')$, we need to compute f again.

It's easy to come up with a classical probabilistic computation to simulate the above quantum computation: randomly pick up an x with probability $1/N$, then compute $f(x)$. Simply using quantum superposition or quantum parallelism is not helpful. It needs carefully design of a quantum circuit for quantum speed-up.

1.4 Quantum Advantage

Expected exponential speed-up of quantum algorithm: e.g. Shor's algorithm. In the sense of query complexity, exponential speed-up can be proved. Here we give a toy model to show quantum advantage.

Suppose we can compute $f(x)$ given x, the problem is to decide $f(0) \overset{?}{=} f(1)$. It is clearly classical computer have to compute f twice. We will show quantum computer can solve this problem by computing f once.

$$U_f|x\rangle\frac{|0\rangle - |1\rangle}{\sqrt{2}} = |x\rangle\frac{|f(0)\rangle - |f(1)\rangle}{\sqrt{2}} = (-1)^{f(x)}|x\rangle\frac{|0\rangle - |1\rangle}{\sqrt{2}} \text{ or simply } V_f|x\rangle = (-1)^{f(x)}|x\rangle$$

$$V_f\frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}} = \begin{cases} \pm|+\rangle, f(0) = f(1) \\ \pm|-\rangle, f(0) \neq f(1) \end{cases} \text{ where } |\pm\rangle = \frac{|0\rangle \pm |1\rangle}{\sqrt{2}} \text{ and } \langle +|-\rangle = 0.$$

so we can measure it only once on $|\pm\rangle$ basis set to distinguish which case deterministicly. This is the Deutsch algorithm.

1.5 Other Quantum Cmoputation Model

Here we mention another model: adiabatic quantum computation. It is equivalent to the circuit model we mentioned above. We mention it here because it uses quite diferent strtegy to design algorithm and it seems to be quite useful in quantum machine learning. Please see

- Aharonov, D., Van Dam, W., Kempe, J., Landau, Z., Lloyd, S., & Regev, O. (2008). Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, *50*(4), 755-787.

for more detail.

2 A Brief Introduction of Quantum algorithm

Here we only give some very basic algorithm or subroutine in quantum computing and will not discuss them in detail. For more detail, please see

- Nielsen, M. A., & Chuang, I. (2002). Quantum computation and quantum information.

2.1 Time Evolution

Given a $2^n \times 2^n$ Hermitian matrix A, try to implement unitary operator e^{iAT} (time evolution operator). With some constraints on A, this can be implemented in a $\mathrm{poly}(n, T, 1/\varepsilon)$ size quantum circuit where ε is the precision. Classical computation seems need at least 2^n, but we should be careful and notice probabilistic computation.

For example, A is a local Hamiltonian $H = \sum_{<i,j>} H_{ij}$, where H_{ij} is a Hermitian matrix acting non-trivially only on qubit i and qubit j. Using Trotter decomposition,

$$e^{iH\delta t} = \prod_{<i,j>} e^{iH_{ij}\delta t} + O(\delta t^2)$$

each term in the product can be implemented by a local unitary. Then we repeat the above procedure $T/\delta t$ times:

$$\left(\prod_{<i,j>} e^{iH_{ij}\delta t}\right)^{\frac{T}{\delta t}} + O\left(\frac{T}{\delta t} \cdot \delta t\right) = e^{iHT} + O(T\delta t)$$

with a sufficiently small δt (but not too small), we can implement the time evolution operator with good precision.

Time evolution for some more general matrix A can be implemented efficiently, e.g. A is sparse: i. there are only $\mathrm{poly}(n)$ non-zero element in each row; ii. given the row index, the positions and values of non-zero elements can be listed in polynomial time. See

- Aharonov, Dorit, and Amnon Ta-Shma. "Adiabatic quantum state generation and statistical zero knowledge." *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*. ACM, 2003.

for more detail.

2.2 Phase Estimation

Suppose $A|\psi_i\rangle = \lambda_i|\psi_i\rangle$ and the time evolution operator e^{iAT} can be implemented efficiently. The eigenstate of A is given, then the phase estimation can use it to get an estimation of the corresponding eigenvalues:

$$|\psi_i\rangle|0\rangle \xrightarrow{\text{phase estimation}} |\psi_i\rangle|\tilde{\lambda_i}\rangle$$

where $\tilde{\lambda_i}$ is an estimation of λ_i s.t.

$$\left|\tilde{\lambda_i} - \lambda_i\right| \leqslant \varepsilon$$

and $|\tilde{\lambda_i}\rangle$ is the binary representation of $\tilde{\lambda_i}$ restored in quantum register. The evolution time and the precision has the following relation (Heisenberg Limit):

$$T\varepsilon = O(1).$$

2.3 SWAP-test

Given two quantum states $|\varphi_1\rangle$ and $|\varphi_2\rangle$ as input, SWAP-test can be used to estimate $|\langle\varphi_1|\varphi_2\rangle|^2$. It works in the following way:

$$\begin{aligned} |+\rangle|\varphi_1\rangle|\varphi_2\rangle & \xrightarrow{\text{control-SWAP}} \frac{|0\rangle|\varphi_1\rangle|\varphi_2\rangle + |1\rangle|\varphi_2\rangle|\varphi_1\rangle}{\sqrt{2}} \\ & \xrightarrow{\text{measure 1st qubit on } |\pm\rangle} \frac{|\varphi_1\rangle|\varphi_2\rangle \pm |\varphi_2\rangle|\varphi_1\rangle}{2} \end{aligned}$$

which means the probability to get the outcome $|\pm\rangle$ is

$$\left|\frac{|\varphi_1\rangle|\varphi_2\rangle \pm |\varphi_2\rangle|\varphi_1\rangle}{2}\right|^2 = \frac{1 \pm |\langle\varphi_1|\varphi_2\rangle|^2}{2}$$

repeat the measurement M times, we can get an estimation of $|\langle\varphi_1|\varphi_2\rangle|^2$ with precision $O\left(1/\sqrt{M}\right)$.

2.4 Control-Rotation

A simplified control-rotation can do the following operation:

$$|\theta\rangle|0\rangle \rightarrow |\theta\rangle e^{-i\theta\sigma_y}|0\rangle = |\theta\rangle(\cos\theta|0\rangle + \sin\theta|1\rangle)$$

where $|\theta\rangle$ is the binary representation of θ restored in a quantum register (the quantum circuit can be found in Nielsen & Chuang). Then we can generalize it to do the following:

$$|x\rangle|0\rangle \xrightarrow{\text{control-rotation}} |x\rangle\big(f(x)|0\rangle + \sqrt{1-f^2(x)}|1\rangle\big)$$

where wlog. we assume $|f(x)| \leqslant 1$ here for presentation simplicity. It works in the following way: define $f'(x) = \arccos f(x)$

$$\begin{aligned} |x\rangle|0\rangle|0\rangle &\xrightarrow{U_{f'}\text{ on first two registers}} |x\rangle|f'(x)\rangle|0\rangle \\ &\xrightarrow{\text{simplified control-rotation}} |x\rangle|\arccos f(x)\rangle\big(f(x)|0\rangle + \sqrt{1-f^2(x)}|1\rangle\big) \\ &\xrightarrow{U_{f'}^{\dagger}\text{ on first two registers}} |x\rangle|0\rangle\big(f(x)|0\rangle + \sqrt{1-f^2(x)}|1\rangle\big) \end{aligned}$$

3 HHL Algorithm

Solving $Ax = b$ in a quantum way (a little bit different from the original problem) with some contraint on A and b. Notice the dimension of the problem is $N = 2^n$, the quantum algorithm uses recouces scaling as poly(n).

We consider A is a Hermitian matrix. If A is not Hermitian, we solve the following problem instead:

$$\begin{pmatrix} 0 & A \\ A^{\dagger} & 0 \end{pmatrix}\begin{pmatrix} 0 \\ b \end{pmatrix} = \begin{pmatrix} x \\ 0 \end{pmatrix}$$

In this case, the matrix is Hermitian but we still solve the orignal problem.

Suppose $|b\rangle = \sum_{i=0}^{N-1} b_i|i\rangle$ can be prepared, the time evolution operator e^{iAt} can be implemented for $t \sim \text{poly}(n)$, and $\kappa = \text{poly}(n)$. HHL algorithm can be used to get the state:

$$|x\rangle = \frac{A^{-1}|b\rangle}{\|A^{-1}|b\rangle\|} \text{sometimes with an estimation of} \|A^{-1}|b\rangle\|.$$

It works in the following way: suppose $A|\psi_i\rangle = \lambda_i|\psi_i\rangle$ and $|b\rangle = \sum_i b_i'|\psi_i\rangle$, then

$$\begin{aligned} |b\rangle \quad &\xrightarrow{\text{phase estimation}} \quad \sum_i b_i'|\psi_i\rangle|\tilde{\lambda_i}\rangle \\ &\xrightarrow{\text{control-rotation (approximation result)}} \sum_i b_i'|\psi_i\rangle|\tilde{\lambda_i}\rangle\left(\frac{1}{\lambda_i\kappa}|0\rangle + \sqrt{1-\left(\frac{1}{\lambda_i\kappa}\right)^2}|1\rangle\right) \\ &\xrightarrow{\text{reverse of phase estimation}} \sum_i b_i'|\psi_i\rangle\left(\frac{1}{\lambda_i\kappa}|0\rangle + \sqrt{1-\left(\frac{1}{\lambda_i\kappa}\right)^2}|1\rangle\right) \\ &\xrightarrow{\text{measure last qubit to be }|0\rangle} \frac{1}{\kappa}\sum_i \frac{b_i'}{\lambda_i}|\psi_i\rangle \\ &= \frac{A^{-1}|b\rangle}{\kappa} \end{aligned}$$

the probability to get the result is

$$\frac{\|A^{-1}|b\rangle\|^2}{\kappa^2} \leqslant \frac{1}{\kappa^4}$$

(this is one of the reasons we require κ is not too small) and the final state is the normalization of the above state which is $|x\rangle$.

3.1 The Challenges of Making HHL Useful for Practical Problem

i. how to prepare $|b\rangle$ (N dimension) in poly($\log N$)? (very difficult, qRAM?) In the following we always assume we can do it;

ii. how to implement e^{iAt} for practical A? (Is sparse A useful? seems Yes)

iii. κ=poly($\log N$) (for a generic matrix, $\kappa \sim N^c$) (intrinsic property of the problem itself)

iv. how to get useful information from $|x\rangle$? Significant x_i or k-moment $\sum_i x_i^k$? Or more general $\langle x|F|x\rangle$ for some relatively simple F?

the following two papers give more discussion:

- Aaronson, S. (2015). Read the fine print. Nature Physics, 11(4), 291-293.
- Clader, B. David, Bryan C. Jacobs, and Chad R. Sprouse. Preconditioned quantum linear system algorithm. *Physical review letters* 110.25 (2013): 250504.

When applying HHL algorithm to machine learning problem, the last three issues are relatively easy to overcome, the first issue is the most difficult one.

3.2 Generalized HHL Algorithm

We can extend $A^{-1}|b\rangle$ to $f(A)|b\rangle$ for some “good enough” f. Just extend the control-rotation from $\left(\frac{1}{\lambda\kappa}|0\rangle + \sqrt{1-(\frac{1}{\lambda\kappa})^2}|1\rangle\right)$ to $\left(f(\lambda)|0\rangle + \sqrt{1-f^2(\lambda)}|1\rangle\right)$.

4 Some Simple Quantum Machine Learning Algorithms

In this seciton, we assume state preparetion can be done efficiently. This is the challenge for the HHL algorithm to be useful to machine learning task.

4.1 Data Fitting

Suppose

$$f(x) = \sum_j f_j(x)\lambda_j \text{ with some relatively simple fixed } f_j$$

given data points

$$f(x_i) = y_i$$

the problem is to find a set of λ_j fitting the data best. Define a matrix F such that $F_{ij} = f_j(x_i)$. The problem is reduced to the following quadratic problem:

$$\min_{\boldsymbol{\lambda}} \|F\boldsymbol{\lambda} - \boldsymbol{y}\|^2$$

where $\boldsymbol{\lambda}$=$(\cdots, \lambda_j, \cdots)^T$ and $\boldsymbol{y} = (\cdots, y_i, \cdots)^T$. Take derivative of $\boldsymbol{\lambda}$, the result is to solve the linear equation:

$$F^\dagger F\boldsymbol{\lambda} = F^\dagger \boldsymbol{y}.$$

The quantum algorithm works in the following way:

1. prepare $|y\rangle$ (**assumption**)
2. prepare $F^\dagger|y\rangle$ (generalized HHL, contraint on F e.g. sparse)
3. HHL algorithm to implement $(F^\dagger F)^{-1}$ (contraint on $F^\dagger F$ e.g. sparse)
4. get the result $|\lambda\rangle$, can be used to get useful information, e.g. significant λ_i for sparse representation, computing $f(x_{\text{new}})$ for a new data x_{new} if $\sum_j f_j(x_{\text{new}})|j\rangle$ can be prepared, then using SWAP-test.

4.2 Quantum Principle Component Analysis

Suppose the density matrix ρ is in register 1 and σ is in register 2, and S is the SWAP operation

$$\begin{aligned}\mathrm{tr}_1\, e^{-iS\delta t}\rho\otimes\sigma e^{iS\delta t} &= \sigma - i[\rho,\sigma]\delta t + O(\delta t^2)\\ &= e^{-i\rho\delta t}\sigma e^{i\rho\delta t} + O(\delta t^2)\end{aligned}$$

the first step is due to

$$\begin{aligned}\mathrm{tr}_1(\rho\otimes\sigma)\,S &= \sigma\rho\\ \mathrm{tr}_1 S(\rho\otimes\sigma) &= \rho\sigma\end{aligned}$$

This means we can implement time evolution operator $e^{-i\rho t}$ with a good approximation if we have multiple copies of ρ.

Then we give an example to show how to do principle component analysis. For simplicity, we assume the norm of each data is normalized, it is not difficult to extend the algorithm to general case. Suppose the data is $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_i, \cdots$, principle component analysis is to find the eigenstate of covariance matrix $\sum_i \boldsymbol{x}_i \boldsymbol{x}_i^T$ with maximum eigenvalue. Quantum PCA works in the following way (ignore normalization constant):

1. **assume the state preparetion can be done efficiently**:

$$|i\rangle|0\rangle \rightarrow |i\rangle|x_i\rangle \text{ where } |x_i\rangle \text{ is the quantum state encoding } \boldsymbol{x}_i;$$

2. start from equal weight superposition then apply the above state preparetion:

$$\sum_i |i\rangle|0\rangle \rightarrow \sum_i |i\rangle|x_i\rangle;$$

3. the density matrix is:

$$\sum_{ij} |i\rangle\langle j|\otimes|x_i\rangle\langle x_j|;$$

4. partial trace the first register:

$$\rho \propto \sum_{ij} \mathrm{tr}(|i\rangle\langle j|)\otimes|x_i\rangle\langle x_j| = \sum_i |x_i\rangle\langle x_i| = \text{covariance matrix};$$

5. write ρ in diagonal form:

$$\rho = \sum_i \lambda_i|\psi_i\rangle\langle\psi_i|;$$

6. using $e^{-i\rho t}$ to do phase estimation on ρ:

$$\sum_i \lambda_i|\psi_i\rangle\langle\psi_i|\otimes|\tilde{\lambda}_i\rangle\langle\tilde{\lambda}_i|;$$

7. measure the last register, with probability λ_i, the outcome is $|\psi_i\rangle|\tilde{\lambda}_i\rangle$;

8. so repeat the above procedure several times, the most frequently outcome is $|\psi_{\max}\rangle$, which is the principle component; this can be used to some useful things, e.g. give a new data $|x_{\text{new}}\rangle$, we can use SWAP-test to compute the inner product with $|\psi_{\max}\rangle$ which is the one-dimensional representation of $|x_{\text{new}}\rangle$.

4.3 Revisit Data Fitting

Using the technique in quantum PCA, we can relax the constraint on $F^\dagger F$ and reduce it to state preparetion. Assume we can prepare

$$\sum_{ij} F_{ij}|i\rangle|j\rangle$$

efficiently. Its density matrix is

$$\sum_{kilj} F_{ki}F_{lj}^*|k\rangle\langle l|\otimes|i\rangle\langle j|$$

partial trace the first register

$$\sum_{kilj} F_{ki}F^*_{lj}\mathrm{tr}(|k\rangle\langle l|)\otimes|i\rangle\langle j|=\sum_{kij} F_{ki}F^*_{kj}|i\rangle\langle j|=\sum_{ij} (F^\dagger F)_{ij}|i\rangle\langle j|$$

then use HHL algorithm to do data fitting without some contraint on $F^\dagger F$ like sparsity.

4.4 Other Algorithm Related to Quadratic Optimization Problem

Quantum Support Vector Machine, Quantum Independence Component Analysis, etc.

Since most quadratic optimization problems can be reduced to solving linear equations.

5 Other Types of Quantum Machine Learning Algorithms

Grover or its variation amplitude amplification, adiabatic quantum computation, quantum annealing, etc.

- maybe useful for more general optimization problem;
- Gibbs sampling, useful for inference on Probabilistic Graph Model and training Boltzmann Machine (Better than MCMC).

Quite different from those based on HHL algorithm:

- without assumption for state preparetion;
- at least polynomial speed-up;
- maybe or may not exponential speed-up, lack of theoretical analysis.