

ISO Presentation

Quantum Programming

Tristan NEMOZ

Supervisor: Mario BERTA

6 May 2020

Plan

1 Introduction

2 Problem statement

3 Quantum Computing

Classical computing parallel

Quantum Computing formalism

4 Implementing a Quantum Recommendation system

5 Errata

6 Conclusion

7 Questions

Quantum Computing applied to Machine Learning

- Goal: exponential speed-up

Quantum Computing applied to Machine Learning

- Goal: exponential speed-up
- HHL¹ (2009): $O(n) \rightarrow O(\log(n) \kappa^2)$

¹Harrow, Hassidim, and Lloyd, “Quantum Algorithm for Linear Systems of Equations”.

Quantum Computing applied to Machine Learning

- Goal: exponential speed-up
- HHL¹ (2009): $O(n) \rightarrow O(\log(n) \kappa^2)$
- Quantum Recommendation Systems² (2016):
 $O(n^2) \rightarrow O(\text{poly}(k) \text{polylog}(mn))$

¹Harrow, Hassidim, and Lloyd, "Quantum Algorithm for Linear Systems of Equations".

²Kerenidis and Prakash, *Quantum Recommendation Systems*.

Quantum Computing applied to Machine Learning

- Goal: exponential speed-up
- HHL¹ (2009): $O(n) \rightarrow O(\log(n) \kappa^2)$
- Quantum Recommendation Systems² (2016):
 $O(n^2) \rightarrow O(\text{poly}(k) \text{polylog}(mn))$

¹Harrow, Hassidim, and Lloyd, "Quantum Algorithm for Linear Systems of Equations".

²Kerenidis and Prakash, *Quantum Recommendation Systems*.

Plan

- 1 Introduction
- 2 Problem statement**
- 3 Quantum Computing
 - Classical computing parallel
 - Quantum Computing formalism
- 4 Implementing a Quantum Recommendation system
- 5 Errata
- 6 Conclusion
- 7 Questions

Real-world problem overview

- m users can rate n products

Real-world problem overview

- m users can rate n products (binary voting)

Real-world problem overview

- m users can rate n products (binary voting)
- Not all ratings are known

Real-world problem overview

- m users can rate n products (binary voting)
- Not all ratings are known
- Can we predict whether user j will like product k ?

Real-world problem overview

- m users can rate n products (binary voting)
- Not all ratings are known
- Can we predict whether user j will like product k ? Can we **recommend** a product to this user?

Mathematical description

Binary recommendation system

Let $\mathbf{P} \in \{0; 1\}^{m \times n}$, whose coefficients are known with a given probability. A quantum recommendation system, given $\hat{\mathbf{P}}$ the incomplete representation of \mathbf{P} and an user j , predicts k such that $\mathbf{P}_{j,k} = 1$ with high probability.

Mathematical description

Binary recommendation system

Let $\mathbf{P} \in \{0; 1\}^{m \times n}$, whose coefficients are known with a given probability. A quantum recommendation system, given $\hat{\mathbf{P}}$ the incomplete representation of \mathbf{P} and an user j , predicts k such that $\mathbf{P}_{j,k} = 1$ with high probability.

Solved by Kerenidis and Prakash.

Mathematical description

Binary recommendation system

Let $\mathbf{P} \in \{0; 1\}^{m \times n}$, whose coefficients are known with a given probability. A quantum recommendation system, given $\hat{\mathbf{P}}$ the incomplete representation of \mathbf{P} and an user j , predicts k such that $\mathbf{P}_{j,k} = 1$ with high probability.

Solved by Kerenidis and Prakash **without practical implementation proposed**.

Mathematical description

Binary recommendation system

Let $\mathbf{P} \in \{0; 1\}^{m \times n}$, whose coefficients are known with a given probability. A quantum recommendation system, given $\hat{\mathbf{P}}$ the incomplete representation of \mathbf{P} and an user j , predicts k such that $\mathbf{P}_{j,k} = 1$ with high probability.

Solved by Kerenidis and Prakash **without practical implementation proposed**.

Problem statement

How does the real-world implementation of the Quantum Recommendation System algorithm differs from its theoretic implementation?

Plan

① Introduction

② Problem statement

③ **Quantum Computing**

Classical computing parallel

Quantum Computing formalism

④ Implementing a Quantum Recommendation system

⑤ Errata

⑥ Conclusion

⑦ Questions

Classical computing

- Operates on bit-strings 0110110001

Classical computing

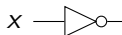
- Operates on bit-strings 0110110001
- Bits are elements of $\{0 ; 1\}$

Classical computing

- Operates on bit-strings 0110110001
- Bits are elements of $\{0; 1\}$
- Operates with a limited set of gates:

Classical computing

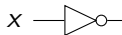
- Operates on bit-strings 0110110001
- Bits are elements of $\{0; 1\}$
- Operates with a limited set of gates:



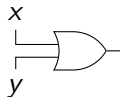
(a) NOT gate

Classical computing

- Operates on bit-strings 0110110001
- Bits are elements of $\{0; 1\}$
- Operates with a limited set of gates:



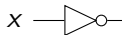
(a) NOT gate



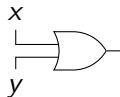
(b) OR gate

Classical computing

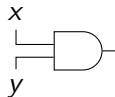
- Operates on bit-strings 0110110001
- Bits are elements of $\{0; 1\}$
- Operates with a limited set of gates:



(a) NOT gate



(b) OR gate

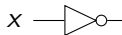


(c) AND gate

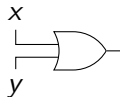
Figure: Classical gates

Classical computing

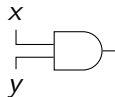
- Operates on bit-strings 0110110001
- Bits are elements of $\{0; 1\}$
- Operates with a limited set of gates:



(a) NOT gate



(b) OR gate

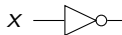


(c) AND gate

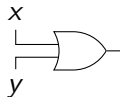
Figure: Classical gates

Classical computing

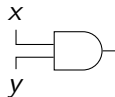
- Operates on bit-strings 0110110001
- Bits are elements of $\{0; 1\}$
- Operates with a limited set of gates:



(a) NOT gate



(b) OR gate

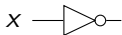


(c) AND gate

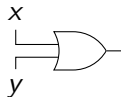
Figure: Classical gates

Classical computing

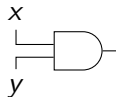
- Operates on bit-strings 0110110001
- Bits are elements of $\{0; 1\}$
- Operates with a limited set of gates:



(a) NOT gate



(b) OR gate



(c) AND gate

Figure: Classical gates

Quantum computing

- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$

Quantum computing

- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$
- Qubits are **normalized vectors of \mathbb{C}^2**

Quantum computing

- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$
- Qubits are **normalized vectors of \mathbb{C}^2**
- **All** operations are reversible

Quantum computing

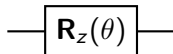
- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$
- Qubits are **normalized vectors of \mathbb{C}^2**
- **All** operations (aside from measuring) are reversible

Quantum computing

- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$
- Qubits are **normalized vectors of \mathbf{C}^2**
- **All** operations (aside from measuring) are reversible
- Operates with a limited set of matrices/gates:

Quantum computing

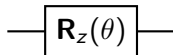
- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$
- Qubits are **normalized vectors of \mathbf{C}^2**
- **All** operations (aside from measuring) are reversible
- Operates with a limited set of matrices/gates:



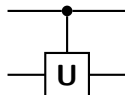
(a) The $R_z(\theta)$ gate

Quantum computing

- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$
- Qubits are **normalized vectors of \mathbb{C}^2**
- **All** operations (aside from measuring) are reversible
- Operates with a limited set of matrices/gates:



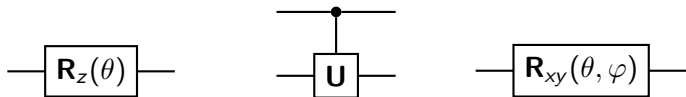
(a) The $R_z(\theta)$ gate



(b) A controlled gate

Quantum computing

- Operates on **qubits**-strings $|q_1\rangle |q_2\rangle |q_3\rangle |q_4\rangle |q_5\rangle$
- Qubits are **normalized vectors of \mathbb{C}^2**
- **All** operations (aside from measuring) are reversible
- Operates with a limited set of matrices/gates:



(a) The $R_z(\theta)$ gate (b) A controlled gate (c) The $R_{xy}(\theta, \varphi)$ gate

Figure: Quantum gates

Qubits

A qubit $|q\rangle$ is a normalized vector of \mathbf{C}^2 . We denote:

$$|q\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

Qubits

A qubit $|q\rangle$ is a normalized vector of \mathbf{C}^2 . We denote:

$$|q\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Qubits

A qubit $|q\rangle$ is a normalized vector of \mathbf{C}^2 . We denote:

$$|q\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \alpha |0\rangle + \beta |1\rangle .$$

Qubits

A qubit $|q\rangle$ is a normalized vector of \mathbf{C}^2 . We denote:

$$|q\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \alpha |0\rangle + \beta |1\rangle .$$

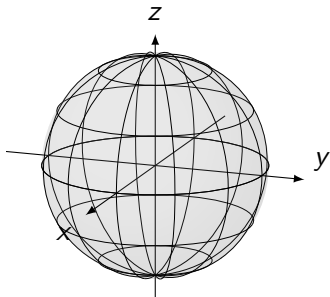
A qubits-string is called a **quantum register**.

The Bloch sphere

A qubit $|q\rangle$ can be represented on the so-called Bloch sphere:

The Bloch sphere

A qubit $|q\rangle$ can be represented on the so-called Bloch sphere:



The Bloch sphere

A qubit $|q\rangle$ can be represented on the so-called Bloch sphere:

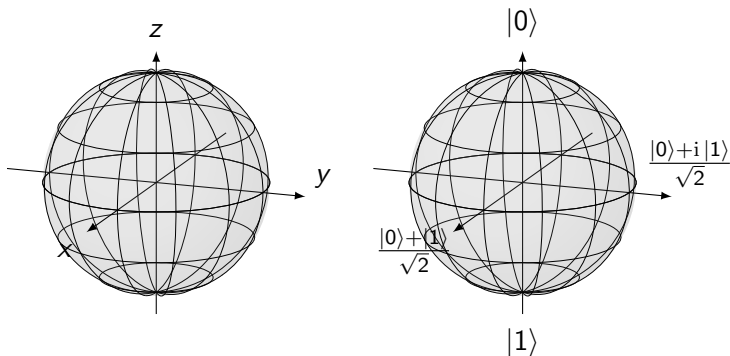


Figure: The Bloch Sphere

The Bloch sphere

A qubit $|q\rangle$ can be represented on the so-called Bloch sphere:

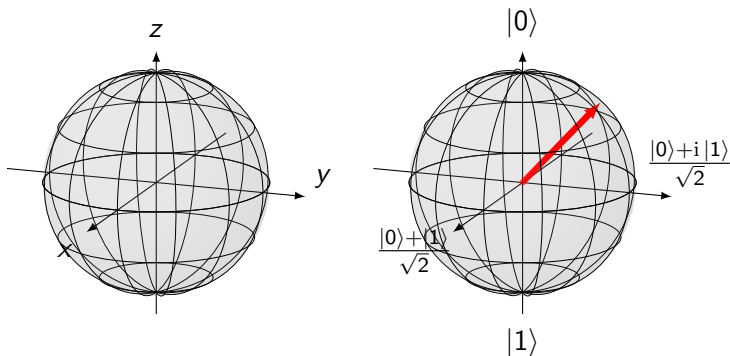


Figure: The Bloch Sphere

Quantum Computing principles

- Possibility to encode information with qubits-strings

Quantum Computing principles

- Possibility to encode information with qubits-strings
- Linearity allows to apply operations on several qubits-string at a time

Quantum Computing principles

- Possibility to encode information with qubits-strings
- Linearity allows to apply operations on several qubits-string at a time
- Possibility to get a probabilistic view of a qubit at the price of forcing it into a certain state.

Quantum Computing principles

- Possibility to encode information with qubits-strings
- Linearity allows to apply operations on several qubits-string at a time
- Possibility to get a probabilistic view of a qubit at the price of forcing it into a certain state. **Measurement destroys information.**

Plan

① Introduction

② Problem statement

③ Quantum Computing

Classical computing parallel

Quantum Computing formalism

④ Implementing a Quantum Recommendation system

⑤ Errata

⑥ Conclusion

⑦ Questions

Implementing a Quantum Recommendation system

- The real-world implementation of the algorithm is quite straight-forward, except for:

Implementing a Quantum Recommendation system

- The real-world implementation of the algorithm is quite straight-forward, except for:
 - Loading a vector stored in a classical structure as a quantum state

Implementing a Quantum Recommendation system

- The real-world implementation of the algorithm is quite straight-forward, except for:
 - Loading a vector stored in a classical structure as a quantum state
 - Applying the Quantum Phase Estimation subroutine

Implementing a Quantum Recommendation system

- The real-world implementation of the algorithm is quite straight-forward, except for:
 - Loading a vector stored in a classical structure as a quantum state
 - Applying the Quantum Phase Estimation subroutine
 - Comparing a qubits-string and a bits-string

Implementing a Quantum Recommendation system

- The real-world implementation of the algorithm is quite straight-forward, except for:
 - Loading a vector stored in a classical structure as a quantum state
 - Applying the Quantum Phase Estimation subroutine
 - Comparing a qubits-string and a bits-string

Loading a vector as a quantum state

Loading a vector as a quantum state

Let $\mathbf{x} \in \mathbf{R}^n$ be a normalized vector.

Loading a vector as a quantum state

Loading a vector as a quantum state

Let $\mathbf{x} \in \mathbf{R}^n$ be a normalized vector. Then, its associated quantum state is:

$$|x\rangle = \sum_{j \in \{0; 1\}^{\lceil \log_2(n) \rceil}} \mathbf{x}_j |j\rangle .$$

Loading a vector as a quantum state

Loading a vector as a quantum state

Let $\mathbf{x} \in \mathbf{R}^n$ be a normalized vector. Then, its associated quantum state is:

$$|x\rangle = \sum_{j \in \{0;1\}^{\lceil \log_2(n) \rceil}} \mathbf{x}_j |j\rangle .$$

Loading \mathbf{x} means creating $|x\rangle$ from $|0\rangle^{\otimes \lceil \log_2(n) \rceil}$ with a polylogarithmic number of gates in n^3 .

QRAM⁴

A QRAM is a binary tree whose leaves store the coefficients of a vector \mathbf{x} and whose nodes stores the sum of its leaves values.

⁴Prakash, "Quantum Algorithms for Linear Algebra and Machine Learning."

QRAM⁴

A QRAM is a binary tree whose leaves store the coefficients of a vector \mathbf{x} and whose nodes stores the sum of its leaves values.

1

Figure: An example of a QRAM tree

⁴Prakash, "Quantum Algorithms for Linear Algebra and Machine Learning."

QRAM⁴

A QRAM is a binary tree whose leaves store the coefficients of a vector \mathbf{x} and whose nodes stores the sum of its leaves values.

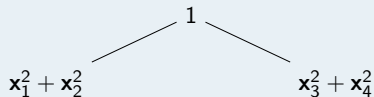


Figure: An example of a QRAM tree

QRAM⁴

A QRAM is a binary tree whose leaves store the coefficients of a vector \mathbf{x} and whose nodes stores the sum of its leaves values.

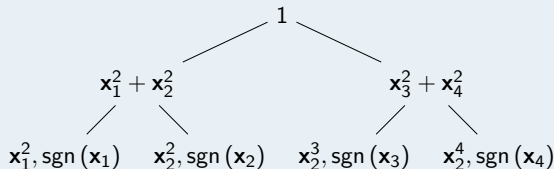


Figure: An example of a QRAM tree

Loading from QRAM

Dervovic et al.'s solution

It is possible, using QRAM, to load $|x\rangle$ using 2^k controlled rotations at level k of the QRAM tree in $O(n)$.

Loading from QRAM

Dervovic et al.'s solution

It is possible, using QRAM, to load $|x\rangle$ using 2^k controlled rotations at level k of the QRAM tree in $O(n)$.

If the gates were to be applied in parallel, the problem would be solved.

Loading from QRAM

Dervovic et al.'s solution

It is possible, using QRAM, to load $|x\rangle$ using 2^k controlled rotations at level k of the QRAM tree in $O(n)$.

If the gates were to be applied **in parallel**, the problem would be solved.

Parallel execution

Parallel execution of controlled rotations

Let $|x\rangle$ be a n -qubits quantum register and $|\text{target}\rangle$ be a target qubit.

Parallel execution

Parallel execution of controlled rotations

Let $|x\rangle$ be a n -qubits quantum register and $|\text{target}\rangle$ be a target qubit. Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y -axis of angle θ_k only if the first quantum register is in state $|k\rangle$

Parallel execution

Parallel execution of controlled rotations

Let $|x\rangle$ be a n -qubits quantum register and $|\text{target}\rangle$ be a target qubit. Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y -axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Parallel execution

Parallel execution of controlled rotations

Let $|x\rangle$ be a n -qubits quantum register and $|\text{target}\rangle$ be a target qubit. Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y -axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n . That is, we want to design a gate \mathbf{L}_x such that:

$$\mathbf{L} |\theta_k\rangle |x\rangle |\text{target}\rangle = \begin{cases} |\theta_k\rangle |x\rangle \mathbf{R}_y(\theta_k) |\text{target}\rangle & \text{if } |x\rangle = |k\rangle \\ |\theta_k\rangle |x\rangle |\text{target}\rangle & \text{otherwise} \end{cases}.$$

A simpler problem

Parallel execution of rotations

It is possible to get the state $|\theta\rangle e^{i\theta} |x\rangle$ from the state $|\theta\rangle |x\rangle$

A simpler problem

Parallel execution of rotations

It is possible to get the state $|\theta\rangle e^{i\theta} |x\rangle$ from the state $|\theta\rangle |x\rangle$ in time $O(1)$.

A simpler problem

Parallel execution of rotations

It is possible to get the state $|\theta\rangle e^{i\theta} |x\rangle$ from the state $|\theta\rangle |x\rangle$ in time $O(1)$.

The z -rotation

Given an angle θ , the rotation around the z -axis of the Bloch sphere is given by the gate:

$$\mathbf{R}_z(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}.$$

Parallel execution of rotations

Let $\theta_i \in \{0; 1\}$.

Parallel execution of rotations

Let $\theta_i \in \{0; 1\}$.

$$|\theta_i\rangle \longrightarrow \boxed{\mathbf{R}_z\left(\frac{1}{2^k}\right)} \longrightarrow |\psi\rangle$$

Figure: Effect of the z-rotation on a qubit

Parallel execution of rotations

Let $\theta_i \in \{0; 1\}$.

$$|\theta_i\rangle \longrightarrow \boxed{\mathbf{R}_z\left(\frac{1}{2^k}\right)} \longrightarrow |\psi\rangle$$

Figure: Effect of the z-rotation on a qubit

$$|\psi\rangle =$$

Parallel execution of rotations

Let $\theta_i \in \{0; 1\}$.

$$|\theta_i\rangle \text{ --- } \boxed{\mathbf{R}_z\left(\frac{1}{2^k}\right)} \text{ --- } |\psi\rangle$$

Figure: Effect of the z-rotation on a qubit

$$|\psi\rangle = \begin{cases} |0\rangle & \text{if } \theta_i = 0 \end{cases}$$

Parallel execution of rotations

Let $\theta_i \in \{0; 1\}$.

$$|\theta_i\rangle \longrightarrow \boxed{\mathbf{R}_z\left(\frac{1}{2^k}\right)} \longrightarrow |\psi\rangle$$

Figure: Effect of the z-rotation on a qubit

$$|\psi\rangle = \begin{cases} |0\rangle & \text{if } \theta_i = 0 \\ e^{i2^{-k}} |1\rangle & \text{if } \theta_i = 1 \end{cases}$$

Parallel execution of rotations

Let $\theta_i \in \{0; 1\}$.

$$|\theta_i\rangle \longrightarrow \boxed{\mathbf{R}_z\left(\frac{1}{2^k}\right)} \longrightarrow |\psi\rangle$$

Figure: Effect of the z-rotation on a qubit

$$|\psi\rangle = \begin{cases} |0\rangle & \text{if } \theta_i = 0 \\ e^{i2^{-k}} |1\rangle & \text{if } \theta_i = 1 \end{cases} = e^{i\theta_i 2^{-k}} |\theta_i\rangle$$

Parallel execution of rotations

Let $\theta_i \in \{0; 1\}$.

$$|\theta_i\rangle \longrightarrow \boxed{\mathbf{R}_z\left(\frac{1}{2^k}\right)} \longrightarrow |\psi\rangle$$

Figure: Effect of the z-rotation on a qubit

$$|\psi\rangle = \begin{cases} |0\rangle & \text{if } \theta_i = 0 \\ e^{i2^{-k}} |1\rangle & \text{if } \theta_i = 1 \end{cases} = e^{i\theta_i 2^{-k}} |\theta_i\rangle$$

Parallel execution of rotations

$$\text{Let } \theta = \sum_{k=0}^{t-1} \theta_k 2^{-k}.$$

Parallel execution of rotations

$$\text{Let } \theta = \sum_{k=0}^{t-1} \theta_k 2^{-k}.$$

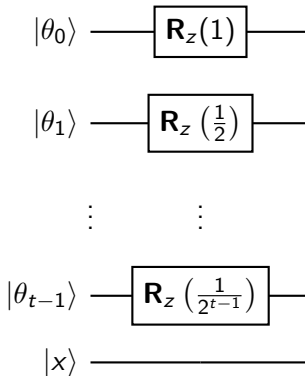


Figure: Parallel execution of z-rotations

Parallel execution of rotations

Let $\theta = \sum_{k=0}^{t-1} \theta_k 2^{-k}$.

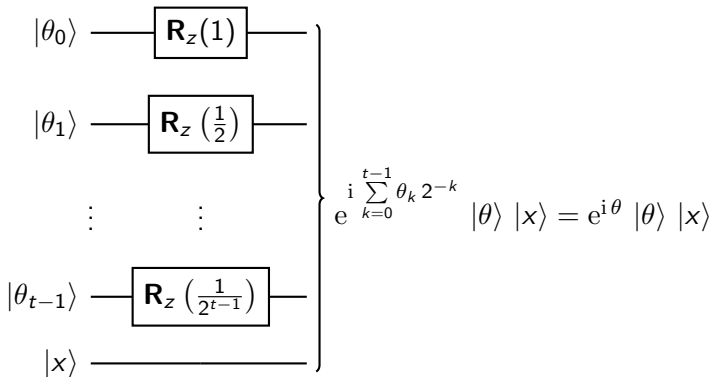


Figure: Parallel execution of z-rotations

Parallel execution of controlled z -rotations

As a recall:

Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y -axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Parallel execution of controlled z-rotations

As a recall:

Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Applying the z-rotation gates in the previous circuit applies a rotation on the target qubit around the z-axis of angle θ_k in time $O(1)$.

Parallel execution of controlled z-rotations

As a recall:

Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Applying the z-rotation gates in the previous circuit applies a rotation on the target qubit around the z-axis of angle θ_k in time $O(1)$.

Parallel execution of controlled z-rotations

As a recall:

Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Applying the z-rotation gates in the previous circuit applies a rotation on the target qubit around the z-axis of angle θ_k in time $O(1)$.

Parallel execution of controlled z-rotations

As a recall:

Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Controlling the z-rotation gates on the target qubit in the previous circuit applies a rotation on the target qubit around the z-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time $O(t)$.

Parallel execution of controlled z-rotations

As a recall:

Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Controlling the z-rotation gates on the target qubit in the previous circuit applies a rotation on the target qubit around the z-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time $O(t)$.

Parallel execution of controlled z-rotations

As a recall:

Parallel execution of controlled rotations consists in applying on the target qubit a rotation around the y-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .

Controlling the z-rotation gates on the target qubit in the previous circuit applies a rotation on the target qubit around the z-axis of angle θ_k only if the first quantum register is in state $|k\rangle$ in time $O(t)$.

Parallel execution of controlled z-rotations

As a recall:

*Parallel execution of controlled rotations consists in applying on the target qubit a rotation **around the y-axis** of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .*

Controlling the z-rotation gates on the target qubit in the previous circuit applies a rotation on the target qubit **around the z-axis** of angle θ_k only if the first quantum register is in state $|k\rangle$ in time $O(t)$.

Parallel execution of controlled z-rotations

As a recall:

*Parallel execution of controlled rotations consists in applying on the target qubit a rotation **around the y-axis** of angle θ_k only if the first quantum register is in state $|k\rangle$ in time constant with respect to n .*

Controlling the z-rotation gates on the target qubit in the previous circuit applies a rotation on the target qubit **around the z-axis** of angle θ_k only if the first quantum register is in state $|k\rangle$ in time $O(t)$.

Is it possible to transform a y-rotation into a z-rotation of the same angle?

Converting a z -rotation to a y -rotation

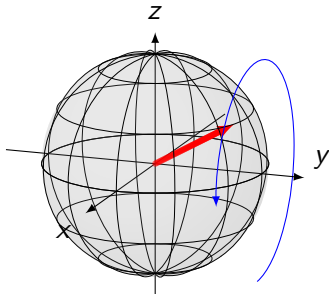


Figure: An example of a transformation of a y -rotation into a z -rotation

Converting a z -rotation to a y -rotation

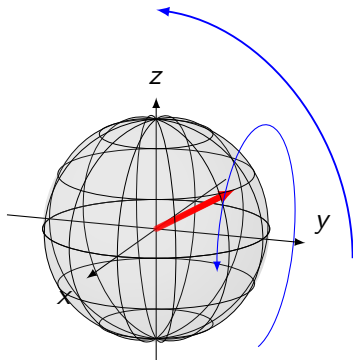


Figure: An example of a transformation of a y -rotation into a z -rotation

Converting a z -rotation to a y -rotation

Figure: An example of a transformation of a y -rotation into a z -rotation

Converting a z -rotation to a y -rotation

Figure: An example of a transformation of a y -rotation into a z -rotation

Converting a z -rotation to a y -rotation

Figure: An example of a transformation of a y -rotation into a z -rotation

Plan

- 1 Introduction
- 2 Problem statement
- 3 Quantum Computing
 - Classical computing parallel
 - Quantum Computing formalism
- 4 Implementing a Quantum Recommendation system
- 5 Errata**
- 6 Conclusion
- 7 Questions

- It **is** possible to apply Quantum Phase Estimation without measuring

- It **is** possible to apply Quantum Phase Estimation without measuring (the error analysis is complex though)

- It **is** possible to apply Quantum Phase Estimation without measuring (the error analysis is complex though)
- It is possible to implement the Threshold gate with a half as much gates

Errata

- It **is** possible to apply Quantum Phase Estimation without measuring (the error analysis is complex though)
- It is possible to implement the Threshold gate with a half as much gates (the complexity remains unchanged nevertheless)
- The implementation as described in the report of the Threshold Gate is wrong (it misses one gate)

Plan

- 1 Introduction
- 2 Problem statement
- 3 Quantum Computing
 - Classical computing parallel
 - Quantum Computing formalism
- 4 Implementing a Quantum Recommendation system
- 5 Errata
- 6 Conclusion**
- 7 Questions

Conclusion

- Other parts work

Conclusion

- Other parts work
- Have shown that the whole algorithm is implementable given some compromises

Conclusion

- Other parts work
- Have shown that the whole algorithm is implementable given some compromises
- Already replaced by its dequantized equivalent





Conclusion

- Other parts work
- Have shown that the whole algorithm is implementable given some compromises
- Already replaced by its dequantized equivalent
- Error-correction, number of qubits, \dots

Plan

- 1 Introduction
- 2 Problem statement
- 3 Quantum Computing
 - Classical computing parallel
 - Quantum Computing formalism
- 4 Implementing a Quantum Recommendation system
- 5 Errata
- 6 Conclusion
- 7 Questions

References I

-  Dervovic, Danial et al. *Quantum linear systems algorithms: a primer*. 2018. arXiv: 1802.08227 [quant-ph].
-  Harrow, Aram W., Avinatan Hassidim, and Seth Lloyd. "Quantum Algorithm for Linear Systems of Equations". In: *Physical Review Letters* 103.15 (2009). ISSN: 1079-7114. DOI: 10.1103/physrevlett.103.150502. URL: <http://dx.doi.org/10.1103/PhysRevLett.103.150502>.
-  Kerenidis, Iordanis and Anupam Prakash. *Quantum Recommendation Systems*. 2016. arXiv: 1603.08675 [quant-ph].
-  Prakash, Anupam. "Quantum Algorithms for Linear Algebra and Machine Learning." PhD thesis. EECS Department, University of California, Berkeley, 2014. URL: <http://www2.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-211.html>.