Exact Quantum Algorithms for the Leader Election Problem

Seiichiro Tani[*,†]

(Joint work with Hirotada Kobayashi[†] and Keiji Matsumoto[‡,†])

[*] NTT Communication Science Laboratories., NTT Corporation

[†] Quantum Computation and Information Project, ERATO, JST

[‡] Foundations of Information Research Division, NII

Outline

1. Introduction
   - Anonymous Leader Election Problem (LE)

2. Our results
   - Two distributed quantum algorithms
     that exactly elect a unique leader

3. Detailed description of our first algorithm

4. Overview of our second algorithm

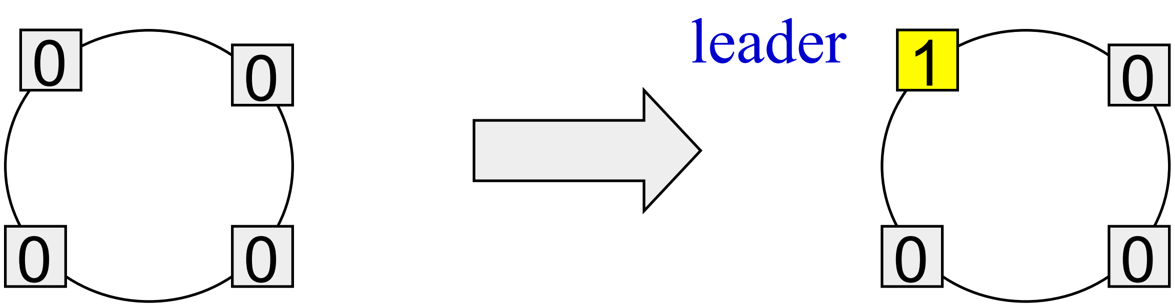
5. Summary

Anonymous Leader Election Problem (LE)

Given n parties connected by communication links, elect a unique leader from among n parties.

Under the Initial Condition:

- All parties are in the same state.
  - ⇒Each party performs the same algorithm.



Negative Results in Classical Cases

- Case 1: # of parties is given,

  No classical algorithm can solve LE exactly for many network topologies

  ("exact" = "zero-error" and "bounded time")

- Case 2: Only the upper bound of # of parties is given,

  No classical algorithm can solve LE even with zero-error for any network topology having cycles.

Our Results

For parties connected by quantum communication links:

- Case 1: n (# of parties) is given,

  LE can be solved exactly
  in poly (in n) time/communication complexity
  for any network topology.

- Case 2: Only N (the upper bound of # of parties) is given,

  LE can be solved exactly
  in poly (in N) time/communication complexity
  for any network topology.

Two proposed algorithms

- Algorithm I
  - More efficient in time and total (quantum + classical) communication complexity
- Algorithm II
  - Less quantum communication and fewer rounds

	Time (including local time steps)	Quantum Comm.	Quantum +Classical Comm.	# of rounds
Algo. I	$O(n^3)$	$O(n^4)$	$O(n^4)$	$O(n^2)$
Algo. II	$O(n^6(\log n)^2)$	$O(n^2 \log n)$	$O(n^6(\log n)^2)$	$O(n \log n)$

Case 1: # of parties (***n***) is given

Details of Algorithm I

Algorithm I Overview

1. Let all parties be eligible to be the leader.
2. For $m = n$ down to 2, repeat PartyReduction(m),

   which works such that:

   - If m equals # of eligible parties,
     # of eligible parties is decreased by at least 1
     (but not decreased to 0)
   - Otherwise, # of eligible parties is decreased or unchanged
3. The party still remaining eligible is the unique leader.

▼ In Step 2, always $m \geq$ (# of eligible parties)

$\Rightarrow$ After Step 2, only one party remains eligible

▼ Even if only the upper bound of n is given, the algorithm works well by using the bound instead of n.

Consistent/inconsistent over eligible parties

Each party has c bits

⇒All parties share cn-bit string s

- String s is inconsistent over eligible parties,
  if all eligible parties do not have the same c-bit values.

- State ϕ is inconsistent over eligible parties,
  If ϕ is a superposition of inconsistent strings

Key Observation used to construct PartyReduction (m)

All eligible parties share an inconsistent state.

⇓

Eligible parties can be reduced by at least one (but cannot be reduced into 0 party) by

1. Measuring qubits.
2. Letting only eligible parties having the maximum value among eligible parties remain eligible.

PartyReduction (m)

(1) Share an inconsistent state
with prob. 1 if m equals # of eligible parties.
(2) By measurement, parties obtain an inconsistent string.
(3) Only eligible parties that have the maximum value among eligible parties remain eligible.

PartyReduction (m) meets requirements:

- if m equals # of eligible parties,
  (3) reduces # of eligible parties by at least 1
  (but not to 0).
- Otherwise # of eligible parties does not increase.

Subgoal A

Share either an inconsistent state or $\left(|0^k\rangle+|1^k\rangle\right)/\sqrt{2}$ among eligible parties (*k*= # of eligible parties)

(1) Each party prepares two qubits.

(2) Each eligible party initializes them to $\dfrac{|0\rangle+|1\rangle}{\sqrt{2}}\otimes|\mathrm{flag}\rangle$

Each non-eligible party initializes them to $|0\rangle\otimes|\mathrm{flag}\rangle$

$$|\phi\rangle=\left(\sum_{i=0}^{2^k-1}|i\rangle\right)|0\rangle^{\otimes(n-k)}|\mathrm{flag}\rangle^{\otimes n}$$

Subgoal A

(3) Check inconsistency of a string corresponding to each basis state in the classical way

(and uncompute to erase all garbage).

$$|\phi\rangle = \left(\sum_{i=0}^{2^k-1} |i\rangle \right) |0\rangle^{\otimes(n-k)} |\text{flag}\rangle^{\otimes n}$$

$$\rightarrow \left(\sum_{i \neq 0, 2^k-1} |i\rangle \right) |0\rangle^{\otimes(n-k)} |\text{true}\rangle^{\otimes n} + \left(|0^k\rangle + |1^k\rangle \right) |0\rangle^{\otimes(n-k)} |\text{false}\rangle^{\otimes n}$$

(4) Measure the flag part.

Subgoal A: Check inconsistency (in the classical way)

Suppose each party *i* has a classical bit $b_i \in \{0,1\}$ of a string

1. Each eligible party initializes $x_i = b_i$,
   while each non-eligible party initializes $x_i = *$
2. Each party repeats the following *n-1* times:
   2.1 Send the current value x_i to all neighbors.
   2.2 Receive the current values $x_{i1}, \ldots, x_{i\deg(i)}$ from all neighbors
   and update $x_i := x_i \cdot x_{i1} \cdot \cdots \cdot x_{i\deg(i)}$.
3. Conclude the string is "inconsistent" iff $x_i \notin \{0, 1\}$.

deg(i): # of edges incident to party *i*

Subgoal A: X_i updating rule

- $x_i = 0$ iff $x_i, x_{i1}, \ldots, x_{i\deg(i)} \in \{0, *\}$ but $\notin \{*\}$
  - All eligible parties could possibly have 0.
- $x_i = 1$ iff $x_i, x_{i1}, \ldots, x_{i\deg(i)} \in \{1, *\}$ but $\notin \{*\}$
  - All eligible parties could possibly have 1.
- $x_i = *$ iff $x_i, x_{i1}, \ldots, x_{i\deg(i)} \in \{*\}$
  - No information on the values of eligible parties.
- Otherwise, x_i = "inconsistent"
  - Eligible parties have to share an inconsistent string.

Subgoal B

Subgoal(B): Transform $\left(|0^k\rangle + |1^k\rangle\right)/\sqrt{2}$ shared by eligible parties
into an inconsistent state with prob. 1,
given the number *k* of eligible parties.

- Case 1: *k* is even,

Each eligible party applies to its qubit

$$U_k = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & e^{-i\pi/k} \\ -e^{i\pi/k} & 1 \end{pmatrix}$$

⇒ In the resulting state,
both $|0\cdots0\rangle$ and $|1\cdots1\rangle$ have amplitude 0,
i.e., the resulting state is inconsistent.

Subgoal B

- Case 2: *k* is odd

  (1) Each eligible party transforms the *k*-cat state into a 2*k*-cat state by preparing a fresh ancilla qubit and applies CNOT.

  (2) Each eligible party then applies to its two qubits

$$V_k = \frac{1}{\sqrt{R_k + 1}} \begin{pmatrix} 1/\sqrt{2} & 0 & \sqrt{R_k} & \sqrt{R_k}\, e^{i\pi/k} \\ 1/\sqrt{2} & 0 & -\sqrt{R_k}\, e^{-i\frac{\pi}{k}} & \sqrt{R_k}\, e^{-i\pi/k} \\ \sqrt{R_k} & 0 & \frac{e^{-\frac{i}{2k}} I_k}{i\sqrt{2} R_{2k}} & -\sqrt{R_k} \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\text{where } R_k = (e^{i\pi/k} + e^{-i\pi/k})/2$$

$$I_k = (e^{i\pi/k} - e^{-i\pi/k})/(2i)$$

⇨ In the resulting state, all of $|00\cdots00\rangle, |01\cdots01\rangle, |10\cdots10\rangle, |11\cdots11\rangle$ have amplitude 0.

PartyReduction (m)

$$|\phi\rangle = \left(\sum_{i=0}^{2^k-1} |i\rangle\right)|0\rangle^{\otimes(n-k)}|\text{flag}\rangle^{\otimes n}$$

Check inconsistency of each string superposed in $|\phi\rangle$

$$\left(\sum_{i\neq 0,2^k-1} |i\rangle\right)|0\rangle^{\otimes(n-k)}|\text{true}\rangle^{\otimes n} + \left(|0^k\rangle + |1^k\rangle\right)|0\rangle^{\otimes(n-k)}|\text{false}\rangle^{\otimes n}$$

↓

Each party measures the flag part.

$\left(|0^k\rangle + |1^k\rangle\right)/\sqrt{2}$ ↓

Apply U_m or V_m

$|\phi_{\text{inconsistent}}\rangle$ ↓

$\left(\sum_{i\neq 0,2^k-1} |i\rangle\right)|0\rangle^{\otimes(n-k)}$ ↓

Measure all qubits and reduce eligible parties.

Algorithm II

Overview of algorithm II (1)

1. Quantum Stage
   - Share log *n* sets of *n*-qubit cat-like-states by one-time exchange of qubits and partial measurement.

$$\left|\phi_i\right\rangle = \left(\left|X_i\right\rangle + \left|\overline{X_i}\right\rangle\right)/\sqrt{2} \quad (i = 1 \ldots \log n)$$

X_i: *n*-bit binary string, which is determined probabilistically.

Overview of algorithm II (2)

2. LOCC Stage

1. Let all parties be eligible to be the leader, and set the number *k* of eligible parties to *n*
2. Repeat *PartyReduction II (k)* until *k*=1

log n times

(1) Transform $|\phi\rangle$ into an inconsistent state by using k with prob. 1

(2) Measure the qubits

(3) Reduce eligible parties by at least half by selecting minorities with resp. to the measurement results

(4) Count the # of eligible parties and set it to k

LOCC= Local quantum Operations and Classical Communication

Quantum Stage: Sharing a Cat-like State (1)

Suppose party *i* has *d* neighbors.

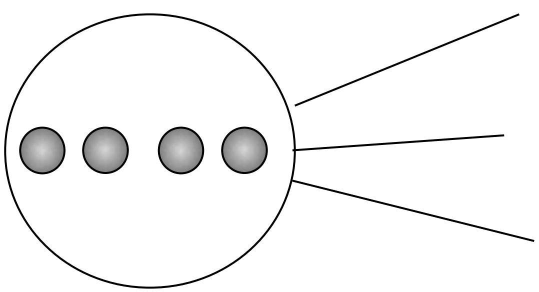
1. Each party *i* prepares a (*d* + 1)-cat state in register **R**

$$\left(\left|0^{d+1}\right\rangle + \left|1^{d+1}\right\rangle\right) / \sqrt{2}.$$

2. Exchange a qubit in **R** with each neighbor party (while keeping one of the qubit in **R** himself.)

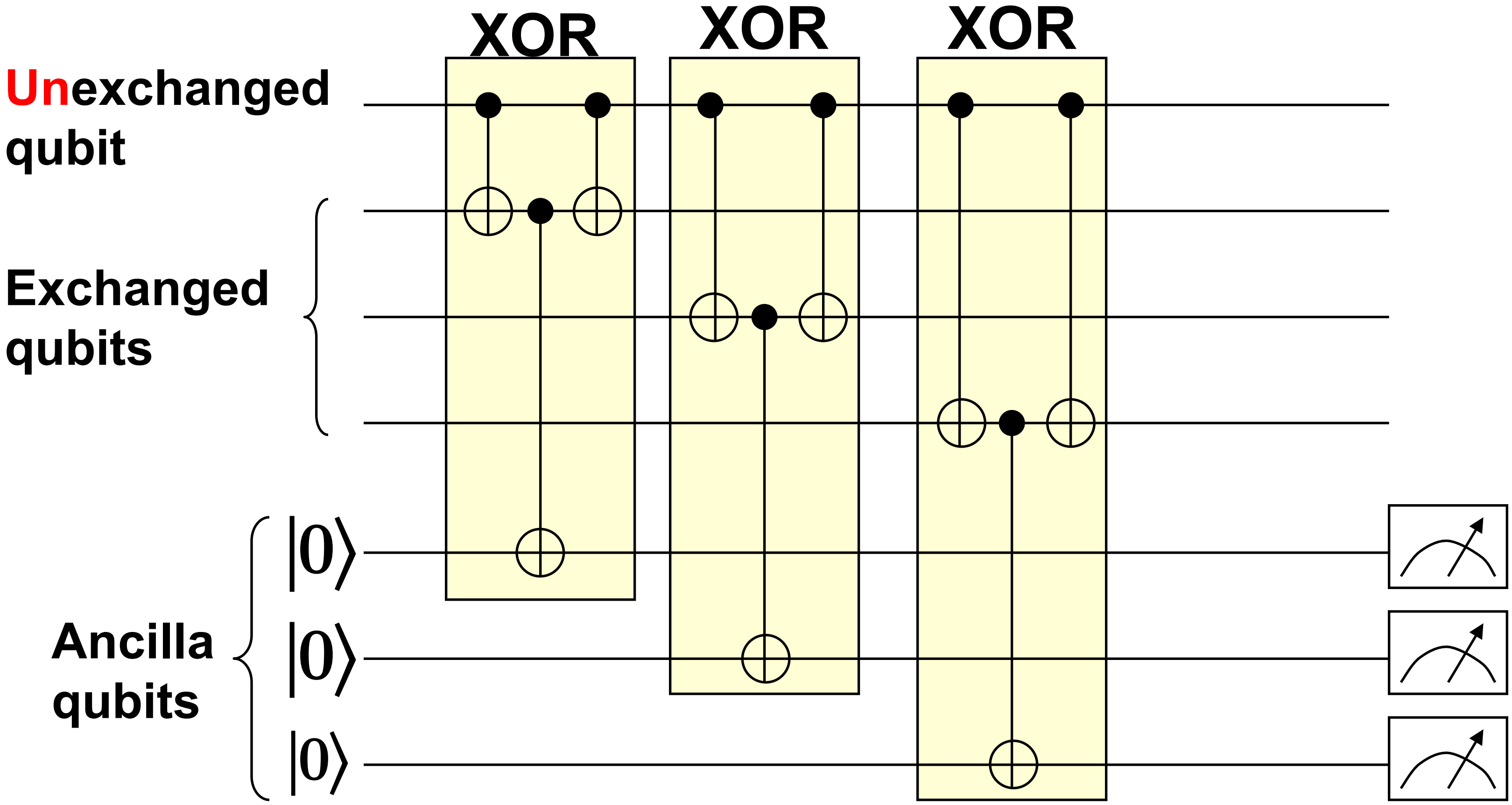
Ex. (*d*=3)

$$\frac{\left|0000\right\rangle + \left|1111\right\rangle}{\sqrt{2}}$$



Quantum Stage: Sharing a Cat-like State (2)

3. Compute an XOR of the unexchanged qubit and each exchanged qubit.
4. Measure the *d* XORs.



Quantum Stage: Sharing a Cat-like State (3)

5. Apply CNOT controlled by the unexchanged qubit targeting to each exchanged qubit

⇒ All exchanged qubits are disentangled.

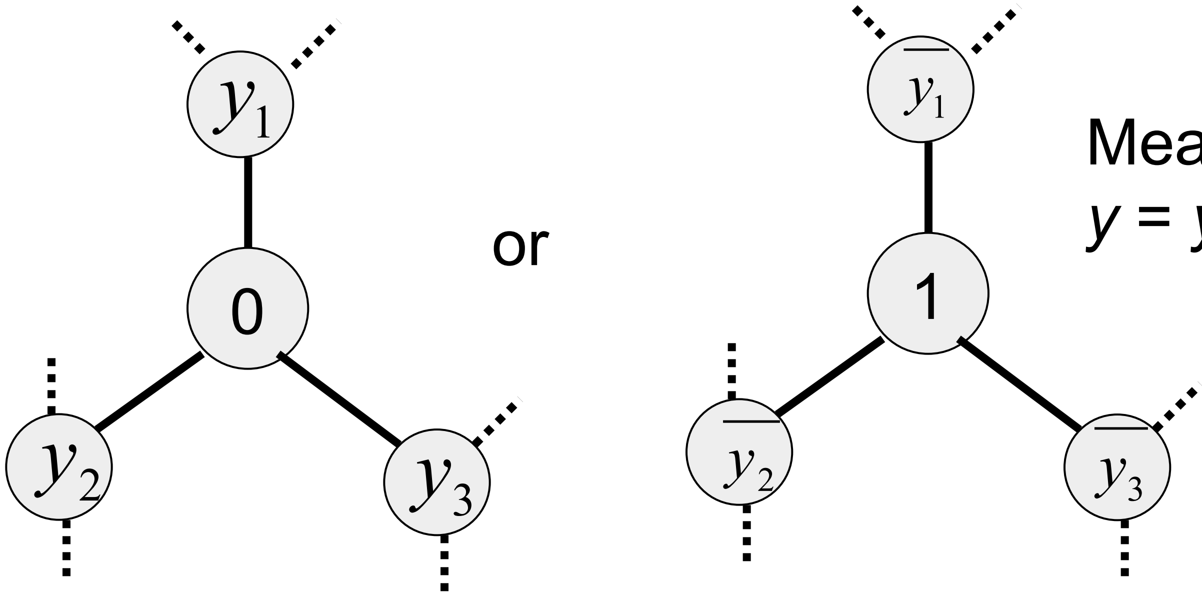
⇓

Lemma. After the procedure, the system state is *n*-qubit state:

$$|\phi\rangle = \left(|X\rangle + |\overline{X}\rangle\right)/\sqrt{2}$$

Quantum Stage: Key point

- By measuring the results of XORs, we fixes the local relations between the party *i*'s value and each neighbor's value.
- Only two basis states X and $\overline{X}$ satisfy all the local relations.



Measurement results $y = y_1 \ldots y_d$,

Summary

- Two distributed quantum algorithms that can exactly solve LE in polynomial time/communication complexity for any network topology.

- Modified versions of our algorithms can even solve the case where only the upper bound of the number of parties is given.

- Our second algorithm involves only one round of quantum communication at the beginning, and after that everything is done with only LOCCs.