Debates with small transparent quantum verifiers

Abuzer Yakaryılmaz
National Laboratory for Scientific Computing, Brazil

A. C. Cem Say
Bogazici University, Turkey

H. Gokalp Demirci
University of Chicago, USA

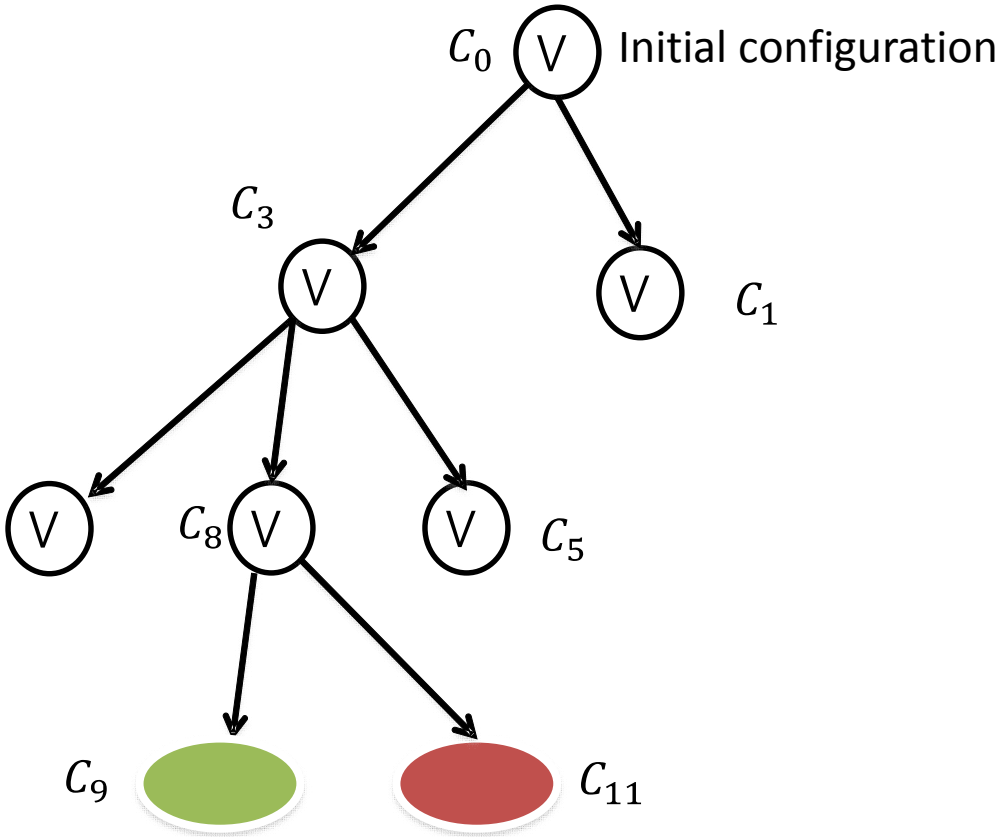
DLT 2014 (Ekaterinburg)
August 26, 2014

A short review of nondeterministic and alternating machines

Alternating finite automata

- There are four types of states:
  - Existential states
  - Universal states
  - Accepting states
  - Rejecting states
- The given input is written on a tape between two end-markers.
- The head is on the left end-marker and the state is the initial one at the beginning of the computation.
- The computation can split more than one branch.
  - $(s,\sigma) \rightarrow \mathit{Powerset}(s',d)$, where $d \in \{\leftarrow,\downarrow,\rightarrow\}$.
- The input is accepted (rejected) if the automaton enters an accepting (a rejecting) states

Computation tree for nondeterministic machines

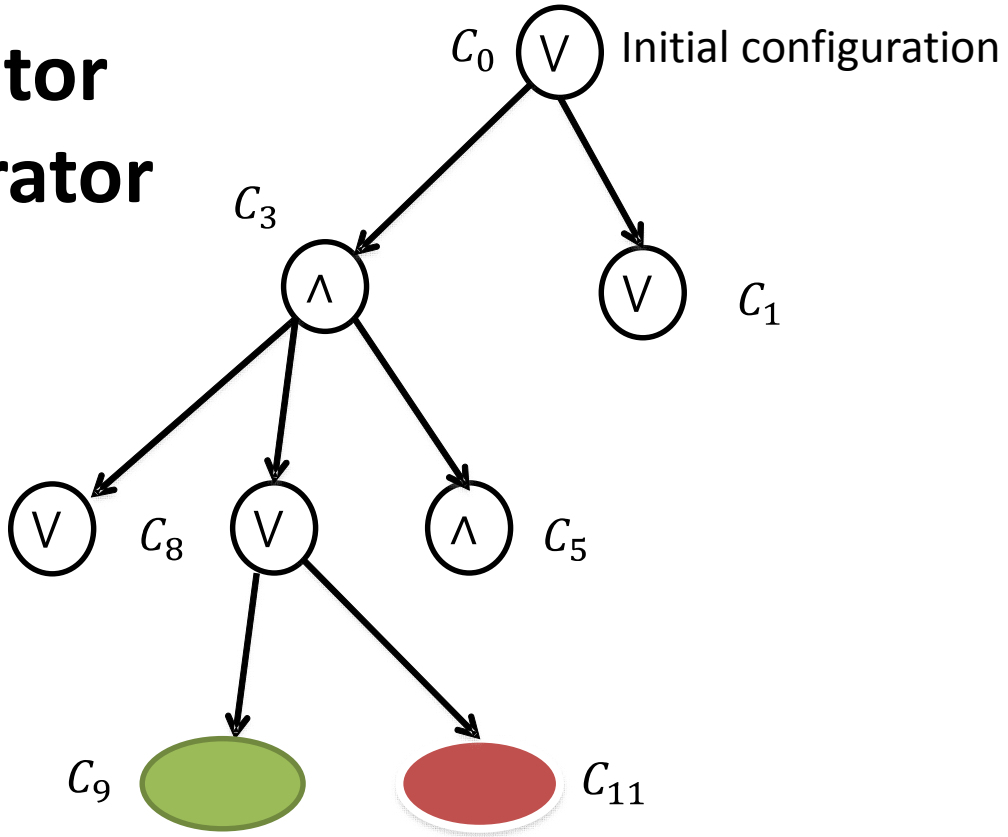


halting configurations

accepting or rejecting?

Computation tree for alternating machines

not only OR (∨) operator
but also AND (∧) operator



halting configurations
accepting or rejecting?

Additional property:

An AND node takes the value of 'TRUE' if all of its children have the value of 'TRUE'

The computation in each path is deterministic!

How can we define **bounded-error** nondeterminism or alternation?

nondeterministic machine

≡

deterministic verification

Similarly, we can define probabilistic verification or quantum verification!

a proof system (a two person game)

For a given language *L,*
the deterministic machine (we call it **verifier**) can interact with a prover (access a proof) to determine the membership of a given input string.

- For the members, the verifier accepts the input. (There exists a valid proof for a true statement!)

- For the non-members, none of the provers convince the verifier and so the verifier rejects input. (There exists no proof for a false statements!)

Probabilistic verification:

The verifier is a probabilistic machine!

- Depending on different branches, different interactions can happen with the prover.

- The verifier can hide its probabilistic choices:
  Private-coin interactive proof system
- Otherwise, it is called:
  Public-coin interactive proof systems
  or, **Arthur-Merlin games**

NP is the class of languages whose memberships are verified in polynomial time deterministically.

Polynomial time probabilistic (quantum) verification defines the class **PSPACE.**

Space-bounded verification:

There is no restriction on time but on space.

The “simplest” case: The verifier is a finite state automaton.

2PFA (two-way probabilistic finite automaton):

- It can recognize the following language with bounded-error.

$$\left\{ a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} \middle| k \geq 0 \right\}$$

Any Turing recognizable language can be recognized by a two-way deterministic finite automaton with two counters (2D2CA).

A proof system having a 2PFA as verifier can simulate the computation of a 2D2CA on a given input string:

- The prover sends the contents of the counters to the verifier and the 2PFA can execute a similar algorithm.
- The drawback: For the non-members, the computation might not be halted. Such proof systems are called “weak”.

Public-coin proof systems are not powerful as private-coin ones!

ArthurMerlin(2PFA) does not contain the language of palindromes and it is a proper subset of **P** even in weak case!

Two-way finite automaton having a constant size quantum register (2QCFA):

They can recognize the following language:

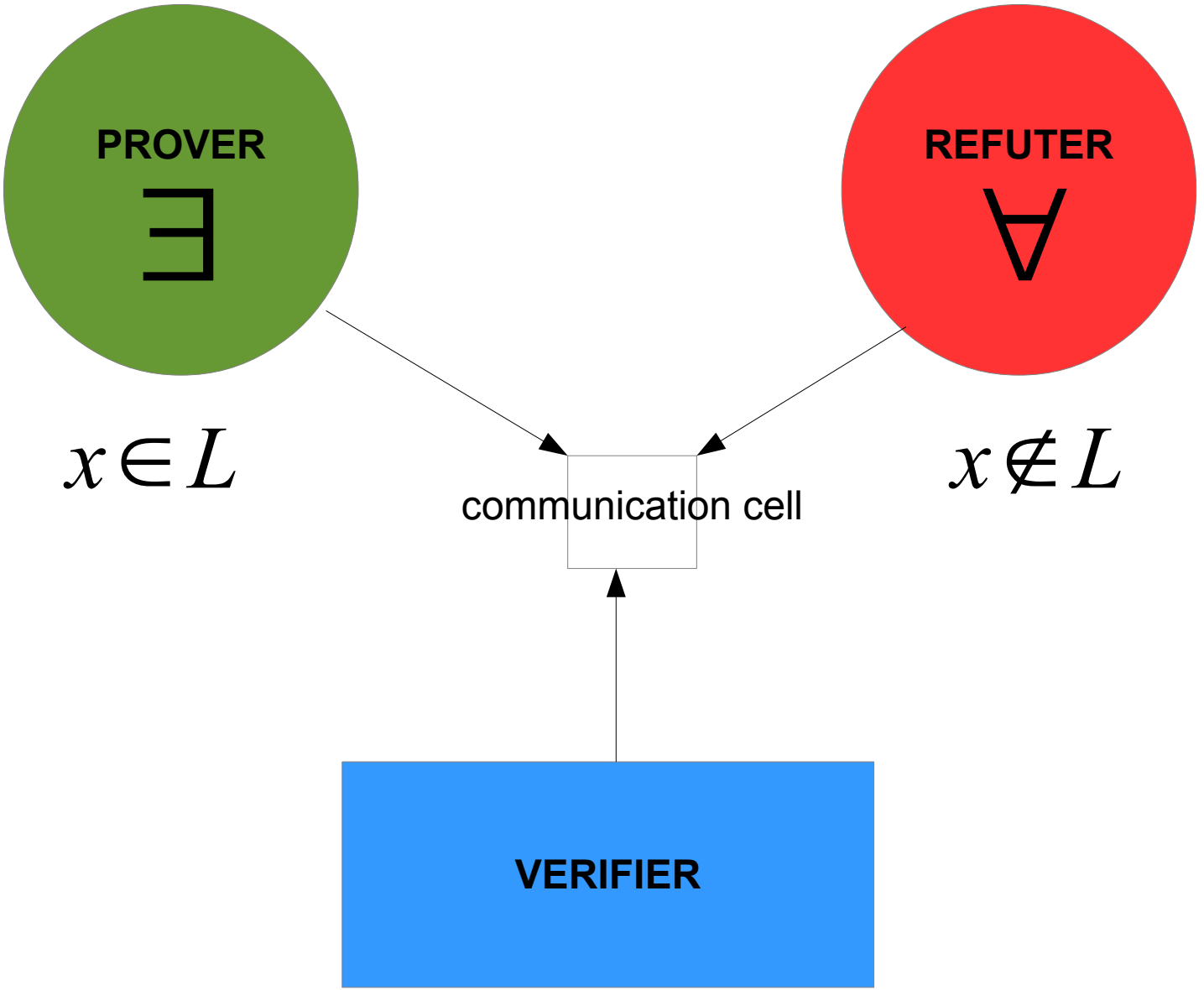
- The members are accepted exactly!

$$\left\{ w_1 c w_1 c w_2 w_2 c \cdots c w_k c w_k \middle| w_i \in \{a, b\}^*, k \geq 0 \right\}$$

Arthur Merlin proof system having 2QCFAs as verifiers:

- The prover can send the configurations of a Turing machine on a given input!
  - The members are accepted exactly!
  - The protocol is public!

Debate Systems (Alternation)


PROVER
∃
REFUTER
∀
$x \in L$
$x \notin L$
communication cell
VERIFIER

We say that language L has a debate chakable by a machine V with error bound $\epsilon \in [0, \frac{1}{2})$ if

- for each $w \in L$, the prover is able to make the verifier accept w with probability at least $1-\epsilon$, no matter what the refuter says in return,
- for each $w \notin L$, the refuter is able to make the verifier reject w with probability at least $1-\epsilon$, no matter what the prover says in return.

A language is said to be debatable if it has a debate checkable by some verifiers.

Note that the class of debatable languages are closed under complementation.

DebateArthurMerlin(2PFA)
is a subset of **NP.**

What about
DebateArthurMerlin(2QCFA)?

A very short review of quantum computation.

A quantum state of n-dimensional quantum register is a norm-1 vector over real (complex) numbers.

$$|\psi\rangle = \alpha_1|c_1\rangle + \cdots + \alpha_n|c_n\rangle, \text{ where } \sum_{j=1}^{n} |\alpha_j|^2 = 1.$$

$$|c_j\rangle = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \qquad |\psi\rangle = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_{j-1} \\ \alpha_j \\ \alpha_{j+1} \\ \vdots \\ \alpha_n \end{pmatrix}$$

Two fundamental quantum operators:

1. Unitary operators preserving the length:

$$|\psi'\rangle = U|\psi\rangle \quad (|\psi\rangle = \alpha_1|c_1\rangle + \cdots + \alpha_n|c_n\rangle)$$
$$= \alpha'_1|c_1\rangle + \cdots + \alpha'_n|c_n\rangle, \text{ where } \sum_{j=1}^{n} |\alpha'_j|^2 = 1.$$

2. Measurements:

$$|\psi'\rangle \rightarrow \left\{ |\psi'_1\rangle, \ldots, |\psi'_k\rangle \middle| \sum_{j=1}^{k} \langle\psi'_j|\psi'_j\rangle = 1 \right\}$$

If outcome j is observed, then system collapses to

$$\frac{|\psi'_j\rangle}{\sqrt{\langle\psi'_j|\psi'_j\rangle}}$$

Consider the following operations defined on rational numbers:

$$v_f = A \cdot A \cdot B \cdot B \cdot A \cdot A \cdot A \cdot A \cdot B \cdot B \cdot A \cdot A \cdot v_0$$

A quantum register simulates this computation and obtain a normalized version of the final vector with a very small probability!

The debate for a decidable language ***L***

Let **T** be the Turing machine deciding ***L*** and ***w*** be the given input.

- The verifier requests the computation history of **T** on **w** from the verifier and refuter: C1$C2$C3$...$CN.
- The verifier check the correctness of the history:
  - Deterministically check the initial one!
  - Deterministically check the format of each configuration.
  - Quantumly check whether Ci is a valid successor of C(i-1).
  - Give a parallel decision to the configuration history.

Decision without error

In the above protocol, the verifier knows that one of the player is lying after a conflict.

If the verifier detects this error immediately, then the verifier does need to the rest of the configuration.

The verifier can only focus on the error but the cheating player may send infinitely long configuration, and so, the verifier could not make a check.

But, the verifier can make such check if the length of the configurations are linear by using the input head.

So, this simulation works for linear space Turing machines (linear-space alternating TMs): $ASPACE(n) = TIME(2^n)$

Multi-head 2QCFAs as verifiers

The last simulation also works for log-space if the automaton has multi-head:

$$\cup_k ASPACE\left(n^k\right) = APSPACE = EXPTIME$$

THANK YOU!

QUESTIONS?