The Knowledge Complexity of Interactive Proof-Systems

Cauchy Huang

The National University of Singapore

Cauchy_0326xz@outlook.com

2024/01/08

Overview

1. Impact of GMR85
2. Definitions and Examples
   - Interactive Proof Systems based on Turing Machines (ITM)
   - Knowledge Complexity
3. Applications
   - P and NP and NPC
   - NIZK of Circuit Satisifiability Problem (Circuit SAT)
   - Related Concepts

Impact of GMR85

- Proposed a new theorem proving procedure, "Interactive Proof System".
- Proposed a new concept of knowledge, "The amount of knowledge", and defined "Knowledge Complexity".
- Proved that there exists Zero Knowledge Interactive Proof Systems.

Classical NP Proof Systems

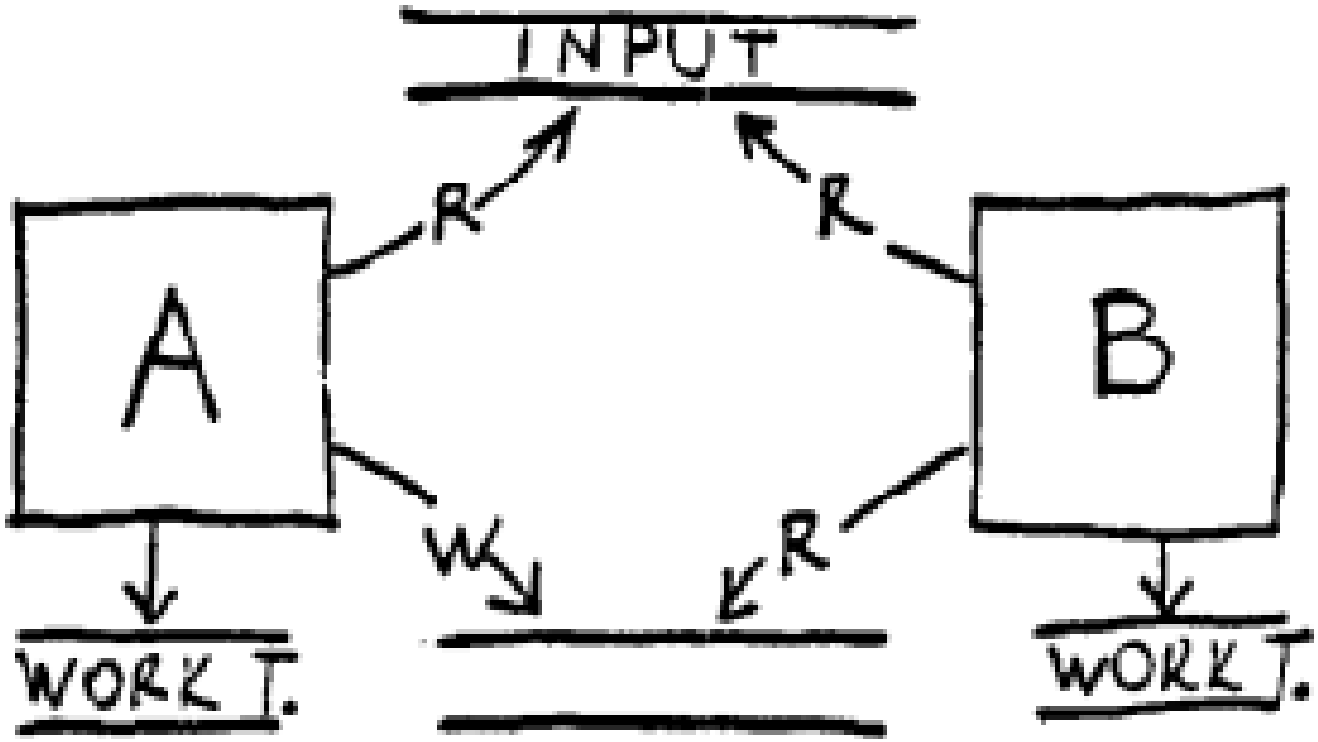

Fig. 1: The *NP* proof-system(*)

A(the prover) is exponential-time, and B(the verifier) is polynomial-time. Both A and B are deterministic, read a common input and interact in a very elementary way.

Interactive Proof Systems based on Turing Machines

2.1 Interactive Turing machines and interactive pairs of Turing machines

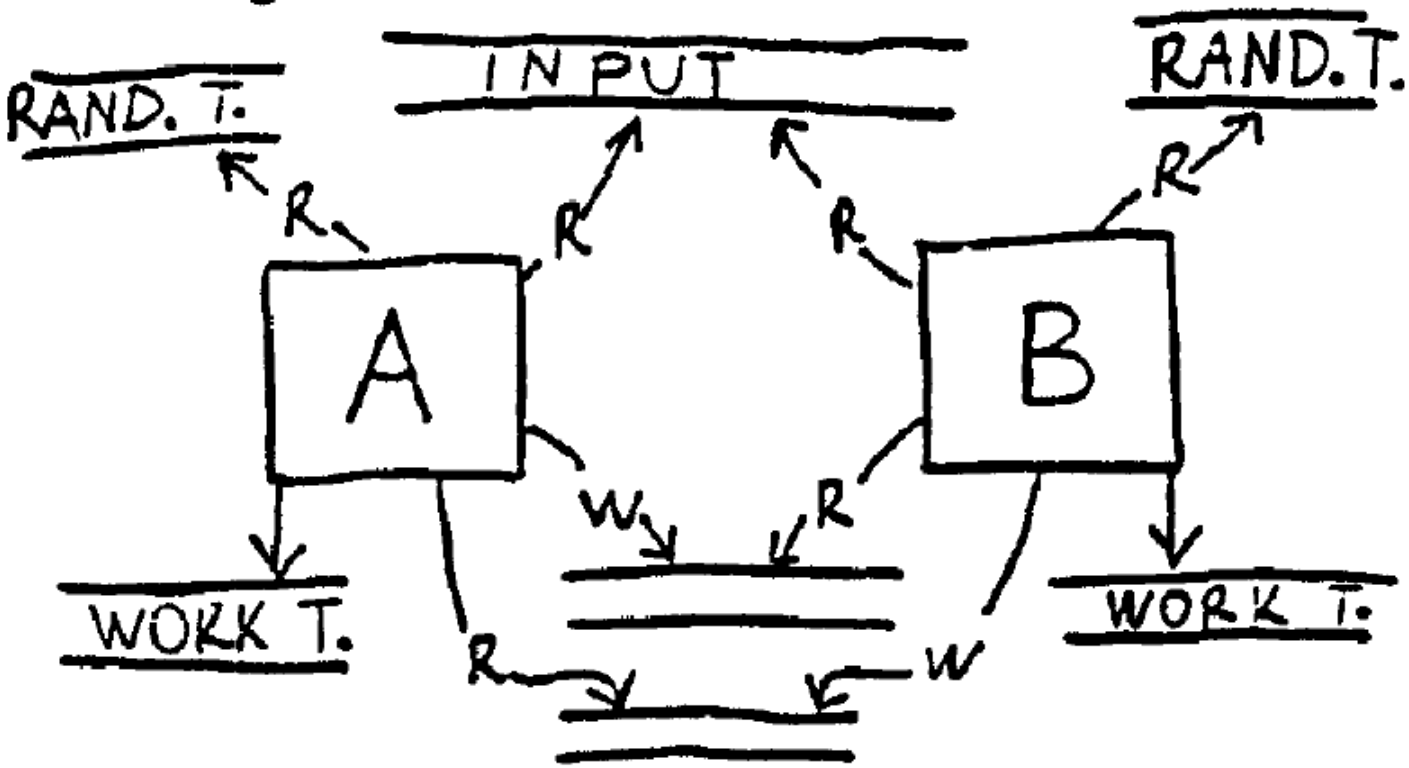

Fig. 2: an interactive pair of Turing machines

Interactive Proof Systems based on Turing Machines

- A and B are **probabilistic** algorithms.
- A and B can communicate in multiple turns.
- The system is probabilistic, which means at last B(the verifier) halts and accepts with probability less than 1. Actually, in GMR85, we require the probability should be at least $1 - \frac{1}{n^k}$ for each k and sufficiently large n.

Sketch

We will start the whole concept from an interesting phenomenon:

Highlight

For each of ITMs, the transcript will be a random bit stream, over the randomness of the input x and the flipping coins of A and B.

Then to define how much knowledge has been transferred in a communication, we can use the power of the degree of distinguishability for each distributions.

Degrees of distinguishability for probability distributions

Definition. I-c-Ensemble

Let I be an infinite set of strings and c a positive constant. For each $x \in I$ with length n, let Π_x be a probability distribution over the n^c-bit strings. Then we say that $\Pi = \{\Pi_x | x \in I\}$ is an I-c-ensemble.

Definition. Distinguishability

Let $p : N \mapsto [0, 1]$, We say that the ensembles Π_1 and Π_2 are at most $p - distinguishable$ if for all probabilistic algorithms D(maps strings $\rightarrow$ $\{0, 1\}$),

$$|p^D_{x,1} - p^D_{x,2}| < p(|x|) + \frac{1}{|x|^k}$$

for all k and sufficiently long x.
Where $p^D_{x,i}$ denotes the probability of the event "D outputs 1 on input a string s randomly chosen from distribution Π_i", and the probability is over the flipping coins of D itself and the distribution.

Knowledge Amount in a communication

Definition. "Communicates at most $f(n)$ bits of knowledge"

Let (A, B) be an interactive pair of Turing Machines, and I the set of its inputs.
Let B be a polynomial-time and $f : N \to N$ be non decreasing. We say that A *communicates at most f(n) bits of knowledge* to B if there exists a probabilistic polynomial-time machine M s.t. the I-ensembles $M[\cdot]$ and $(A, B)[\cdot]$ are at most $1 - \frac{1}{2^n}$-distinguashable.

Blank

Knowledge Complexity

Definition. Knowledge Complexity

Let L be a language possessing an interactive proof-system (A, B). Let $f : N \to N$ be non decreasing. We say that L has *knowledge complexity* $f(n)$ if, when restricting the inputs of (A, B) to the strings in L, A communicates at most $f(n)$ bits of knowledge. We denote this fact by $L \in KC(f(n))$.

P, NP, NPC and Circuit SAT

- P: problems that can be solved in polynomial time.
- NP: problems that can be verified in polynomial time.
- NP-Complete: NP problems, which all NP problems can be reduced to within polynomial time.

NPC Problem. Circuit Satisfiability Problem. (Circuit SAT)

Given a circuit C (consisting of gates and wires), check if there is an input(a set of wires) that makes C outputs 1.

NIZK of Circuit Satisifiability Problem (Circuit SAT)

Common reference string:

(1) $(ck, xk) \leftarrow K_{\text{binding}}(1^k)$
(2) The common reference string is $\sigma = ck$.

Statement: The statement is a circuit C built from NAND-gates. The claim is that there exist wires $w = (w_1, \ldots, w_{\text{out}})$ such that $C(w) = 1$.

Proof: Input (σ, C, w) such that $C(w) = 1$

(1) Commit to each bit w_i as $r_i \leftarrow \mathcal{R}; c_i = \text{com}_{ck}(w_i; r_i)$
(2) For the output wire let $r_{\text{out}} = 0$ and $c_{\text{out}} = \text{com}_{ck}(1; 0)$
(3) For all c_i make a WI proof π_i of existence of an opening (w_i, r_i) so $w_i \in \{0, 1\}$ and $c_i = \text{com}_{ck}(w_i; r_i)$
(4) For all NAND-gates do the following: It has input wires numbered i, j and output wire k. Using message $w_i + w_j + 2w_k - 2$ and randomness $r_i + r_j + 2r_k$ make a WI proof π_{ijk} for $c_i c_j c_k^2 \text{com}_{ck}(-2; 0)$ containing message 0 or 1.
(5) Return the proof π consisting of all the commitments and proofs

Verification: Input (σ, C, π).

(1) Check that all wires have a corresponding commitment and $c_{\text{out}} = \text{com}_{ck}(1; 0)$.
(2) Check that all commitments have a proof of the message being 0 or 1.
(3) Check that all NAND-gates with input wires i, j and output wire k have a proof π_{ijk} for $c_i c_j c_k^2 \text{com}_{ck}(-2; 0)$ containing 0 or 1
(4) Return 1 if all checks pass, else return 0

Fig. 3. Computational NIZK proof for Circuit SAT.

Note: $b_2 = \neg(b_0 \bigwedge b_1) \Leftrightarrow b_0 + b_1 + 2b_2 - 2 \in \{0, 1\}$

Blank

Identification Protocols

Definition. ID Protocol

An identification protocol is a triple $I = (G, P, V)$.

- G is a probabilistic, **key generation** algorithm, that takes no input, and outputs a pair (vk, sk), where vk is called the **verification key** and sk is called the **secret key**.
- P is an interactive protocol algorithm called the **prover**, which takes as input a secret key sk, as output by G.
- V is an interactive protocol algorithm called that **verifier**, which takes as input a verification key vk, as output by G, and which outputs **accept** or **reject**.

We require that when $P(sk)$ and $V(vk)$ interact with one another, $V(vk)$ always outputs **accepts**.

Identification Protocols

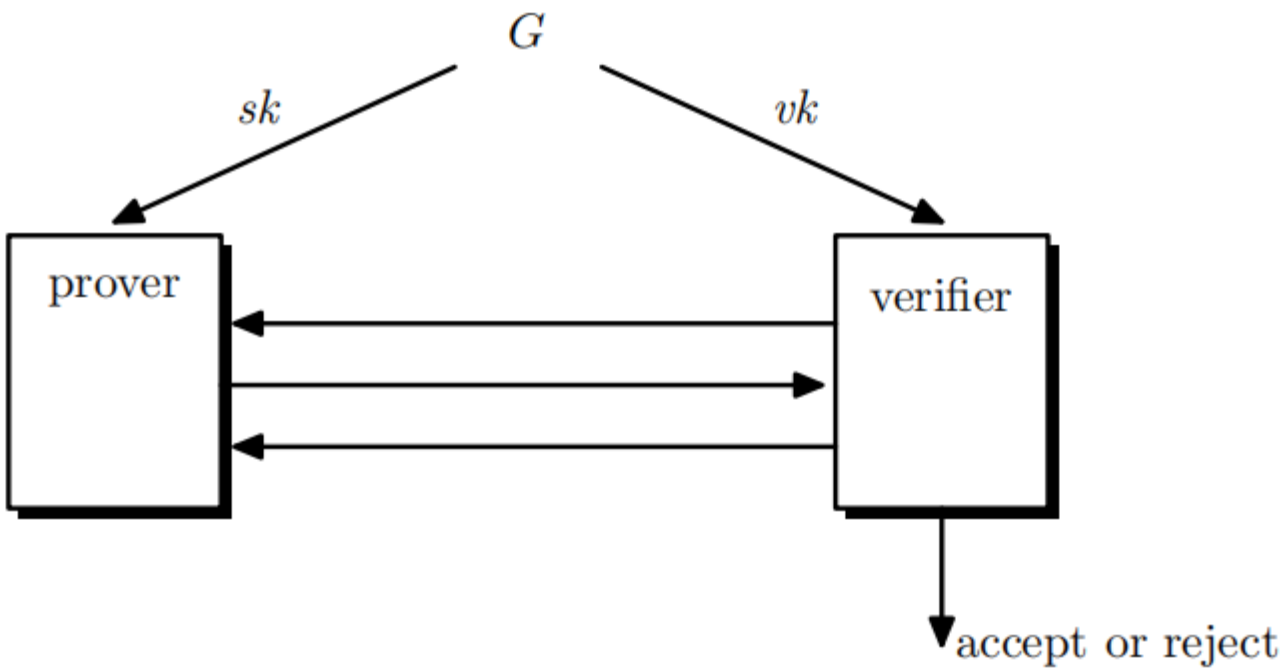

Figure 18.1: Prover and verifier in an ID protocol

Sigma Protocols

Definition. Sigma Protocol

Let $\mathcal{R} \subset \mathcal{X} \times \mathcal{Y}$ be an effective relation. A **SigmaProtocol** for $\mathcal{R}$ is a pair (P, V).

- P is an interactive protocol algorithm called the **prover**, which takes as input a witness-statement pair $(x, y) \in \mathcal{R}$.
- V is an interactive protocol algorithm called the **verifier**, which takes as input a statement $y \in \mathcal{Y}$, and which outputs **accept** or **reject**.
- P and V .....

We require that when $P(sk)$ and $V(vk)$ interact with one another, $V(vk)$ always outputs **accepts**.

Sigma Protocols

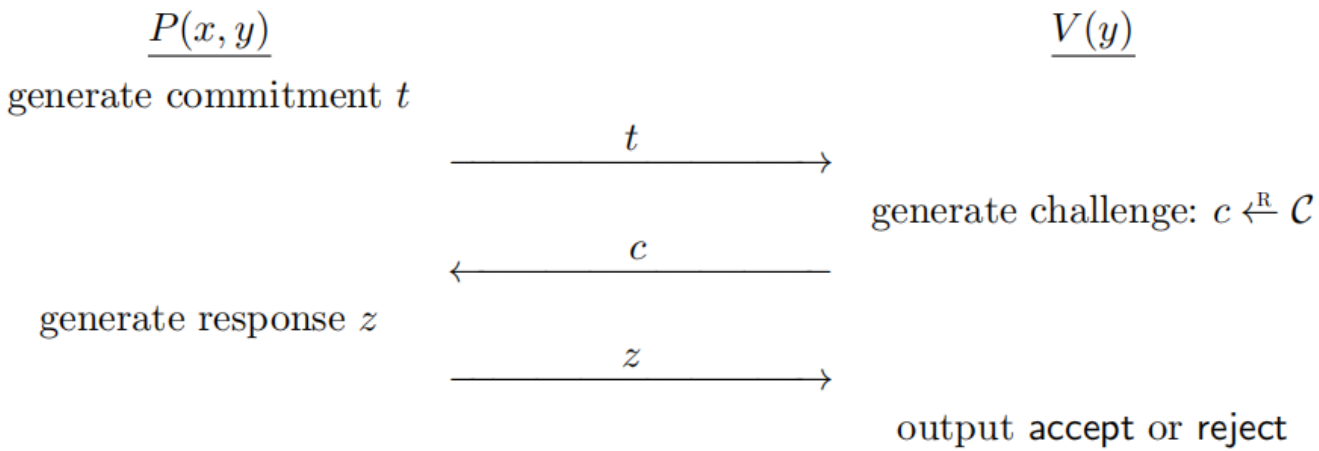

Figure 19.5: Execution of a Sigma protocol

The End