- 教材讨论
  - JH第5章第3节第4小节

问题1：Neq-Pol

- Describing the idea of following algorithm.

Algorithm 5.3.4.4. NEQ-POL

Input: Two polynomials $p_1(x_1, \ldots, x_m)$ and $p_2(x_1, \ldots, x_m)$ over $\mathbb{Z}_n$ with at most degree d, where n is a prime and $n > 2dm$.

Step 1: Choose uniformly $a_1, a_2, \ldots, a_m \in \mathbb{Z}_n$ at random.

Step 2: Evaluate $I := p_1(a_1, a_2, \ldots, a_m) - p_2(a_1, a_2, \ldots, a_m)$.

Step 3: **if** $I \neq 0$ **then output**$(p_1 \not\equiv p_2)$ {accept}
 else output$(p_1 \equiv p_2)$ {reject}.

- What does step 2 means? Why?

Neq-Pol is one-sided error Monte Carlo algorithm

- Explain the basic idea of proving the following theorem.

Theorem 5.3.4.5. *Algorithm* NEQ-POL *is a polynomial time one-sided-error Monte Carlo algorithm that decides the nonequivalence of two polynomials.*

- If $p_1 = p_2$ $\quad Prob(\text{NEQ-POL rejects } (p_1, p_2)) = 1.$
- If $p_1 \neq p_2$

$$Prob(\text{NEQ-POL accepts } (p_1, p_2)) =$$

$$Prob(p_1(a_1, \ldots, a_m) - p_2(a_1, \ldots, a_m) \neq 0) \geq 1 - \frac{m \cdot d}{n} \geq \frac{1}{2}.$$

问题2：Fingerprinting

- Why is NEQ-POL an application of fingerprinting?

we test whether the fingerprint $p_1(a_1, \ldots, a_n)$ of p_1 is identical to the fingerprint $p_2(a_1, \ldots, a_2)$ of p_2 for random $a_1, \ldots, a_n$.

- What is the concrete meanings of error Prob. for NEQ-POL?

问题3：EQ-1BPs

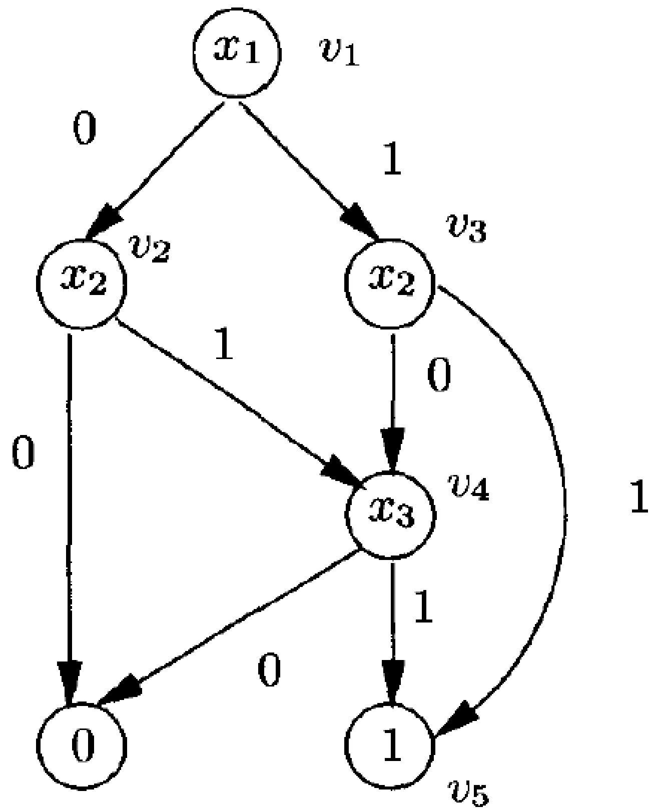
- What is 1BPs?
  - Equivalence problem for one-time-only branching programs.

The equivalence problem for one-time-only branching programs, EQ-1BP, is to decide, for two given one-time-only branching programs B_1 and B_2, whether B_1 and B_2 represent the same Boolean function. One can represent a branching program in a similar way as a directed weighted graph[28] and so we omit the formal description of branching program representation.[29]

EQ-1BP

Input:	One-time-only branching program B_1 and B_2 over a set of Boolean variables $X = \{x_1, x_2, x_3, \ldots\}$.	
Output:	"yes"	if B_1 and B_2 are equivalent (represent the same Boolean function),
	"no"	otherwise.

Constructing a Polynomial for a 1BP



The Properties

Observation 5.3.4.7. For every 1BP A over the set of variables $\{x_1, x_2, \ldots, x_m\}$,

(i) $p_A(x_1, \ldots, x_m)$ is a polynomial of degree at most 1 for every variable,
(ii) $p_A(a_1, \ldots, a_m) = A(a_1, \ldots, a_m)$ for every Boolean input $(a_1, \ldots, a_m) \in \{0, 1\}^m$.

Lemma 5.3.4.8. *For every two* 1BP*s* A *and* B,
A *and* B *are equivalent if and only if* p_A *and* p_B *are identical.*

Algorithm: NEQ-1BP

- What is the idea of the following algorithm?

Algorithm 5.3.4.9. NEQ-1BP

Input: Two 1BPs A and B over the set of variables $\{x_1, x_2, \ldots, x_m\}$, $m \in \mathbb{N}$.

Step 1: Construct the polynomials p_A and p_B.

Step 2: Apply the algorithm NEQ-POL on $p_A(x_1, \ldots, x_m)$ and $p_B(x_1, \ldots, x_m)$ over some $\mathbb{Z}_n$, where n is a prime that is larger than $2m$.

Output: The output of NEQ-POL.

- What is the essential strategy of NEQ-1BP?