理論計算機科学特論 (2026 年前学期)

計算複雑性の基礎

第 14 回

対話証明系 (2) : IP $\subseteq$ PSPACE

岡本 吉央　(電気通信大学)

okamotoy@uec.ac.jp

2026 年 7 月 14 日

最終更新：2026 年 7 月 13 日　22:24

スケジュール (前半)

1. 計算理論の復習 (4/7)
2. 時間計算量：P, NP, coNP (4/14)
3. 帰着と完全性：NP 完全 (4/21)
4. 領域計算量：L, NL, PSPACE (4/28)

＊ 休み (祝日) (5/5)

5. 時間と領域の関係：$\mathrm{P} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXPTIME}$ (5/12)
6. 階層定理：$\mathrm{P} \neq \mathrm{EXPTIME}$ (5/19)
7. Ladner の定理：$\mathrm{NP} - \mathrm{P} = \mathrm{NPC} \Rightarrow \mathrm{P} = \mathrm{NP}$ (5/26)

スケジュール (後半)

8. Savitch の定理：PSPACE = NPSPACE (6/2)
9. Immerman-Szelepcsényi の定理：NL = coNL (6/9)
10. 交代性計算：AP = PSPACE (6/16)
11. 多項式階層：P = NP $\Rightarrow$ P = PH (6/23)
12. 確率的計算：PP, RP, BPP, ZPP (6/30)
13. 対話証明系 (1)：NP $\subseteq$ MA $\subseteq$ AM (7/7)
14. **対話証明系 (2)：IP $\subseteq$ PSPACE** (7/14)
15. 対話証明系 (3)：PSPACE $\subseteq$ IP (7/21)

* 休み (授業のない日) (7/28)

[復習] 対話証明系ってなに？

(interactive proof system)

アーサー (Arthur)

王様

検証者 (verifier)

マーリン (Merlin)

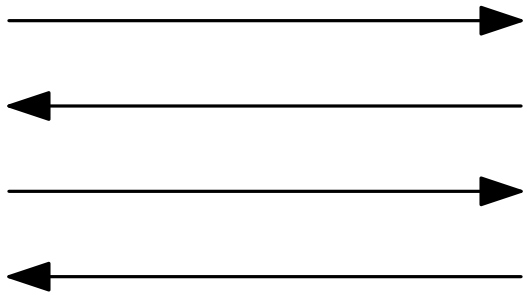
魔法使い

証明者 (prover)

[復習] 対話証明系ってなに？

(interactive proof system)

メッセージ



アーサー (Arthur)

王様

検証者 (verifier)

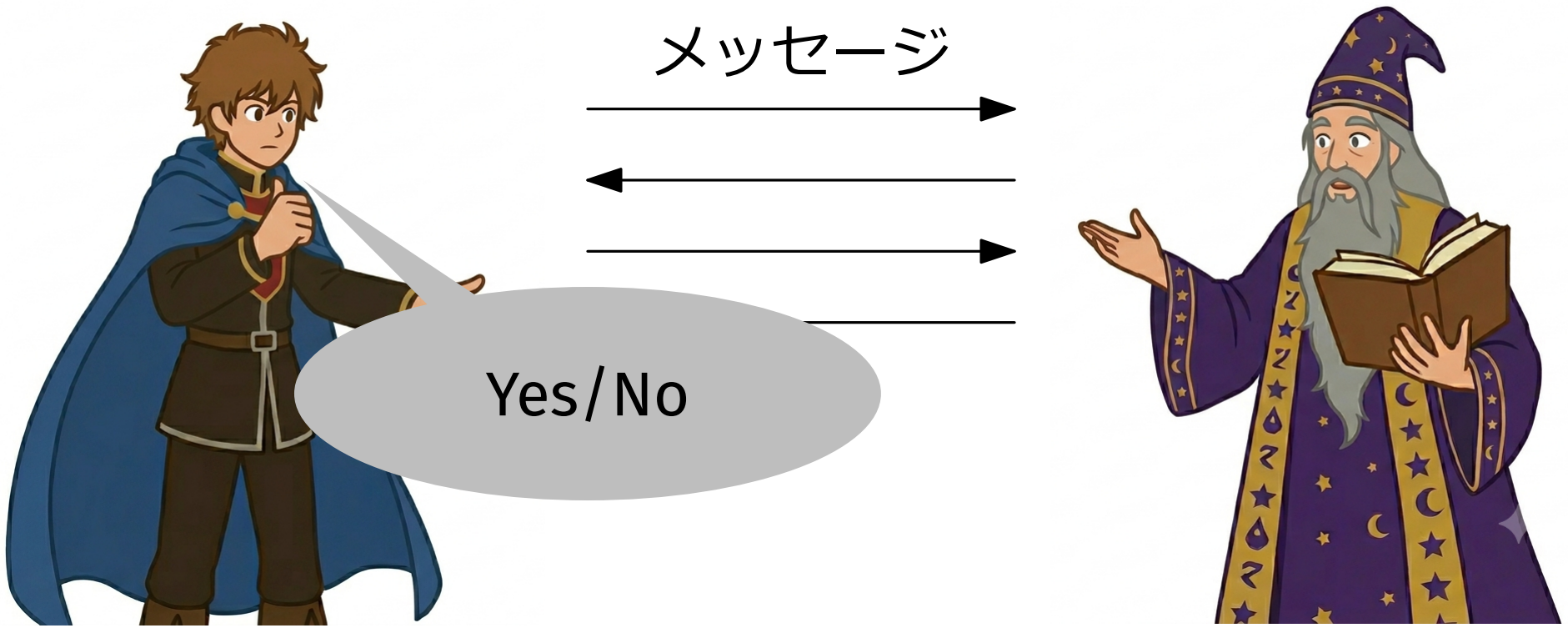
マーリン (Merlin)

魔法使い

証明者 (prover)

[復習] 対話証明系ってなに？

(interactive proof system)



アーサー (Arthur)

王様

検証者 (verifier)

マーリン (Merlin)

魔法使い

証明者 (prover)

[復習] 対話証明系ってなに？ (続)

判定問題 P を解きたい

入力 I

計算能力：低い

計算能力：高い

信頼できない

[復習] 対話証明系ってなに？ (続)

判定問題 P を解きたい

入力 I

正しく Yes/No を
言いたい

アーサーに Yes と
言わせたい

計算能力：低い

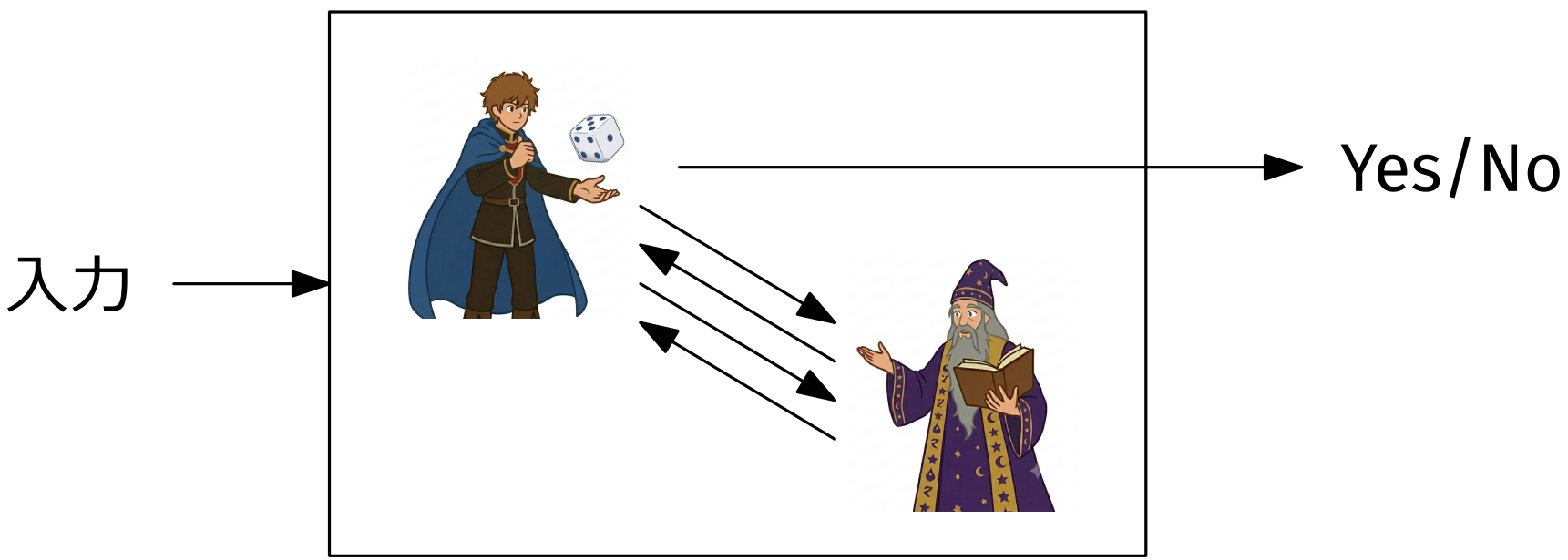
計算能力：高い

信頼できない

[復習] 対話証明系：設定

対話証明系の設定

- 登場人物
  検証者 多項式時間アルゴリズム + 乱数使用
  証明者 なんでも解ける (計算不能問題も解ける)
- 入力は両方に与えられる
- 出力は検証者が行う

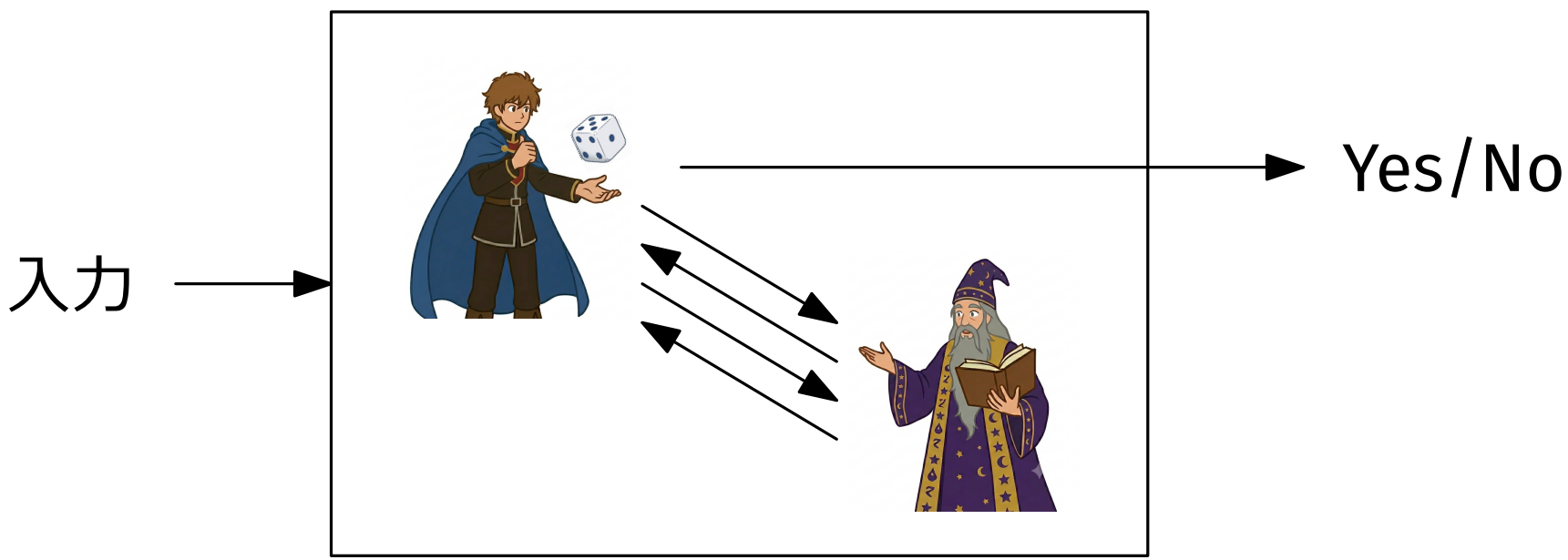


[復習] 対話証明系：プロトコル

定義 (非形式)：プロトコル

対話証明系における **プロトコル** (protocol) とは，
検証者と証明者が行う動作を定めたもの

検証者・証明者はプロトコルから逸脱しない



[復習] 対話証明系で問題を解くとは

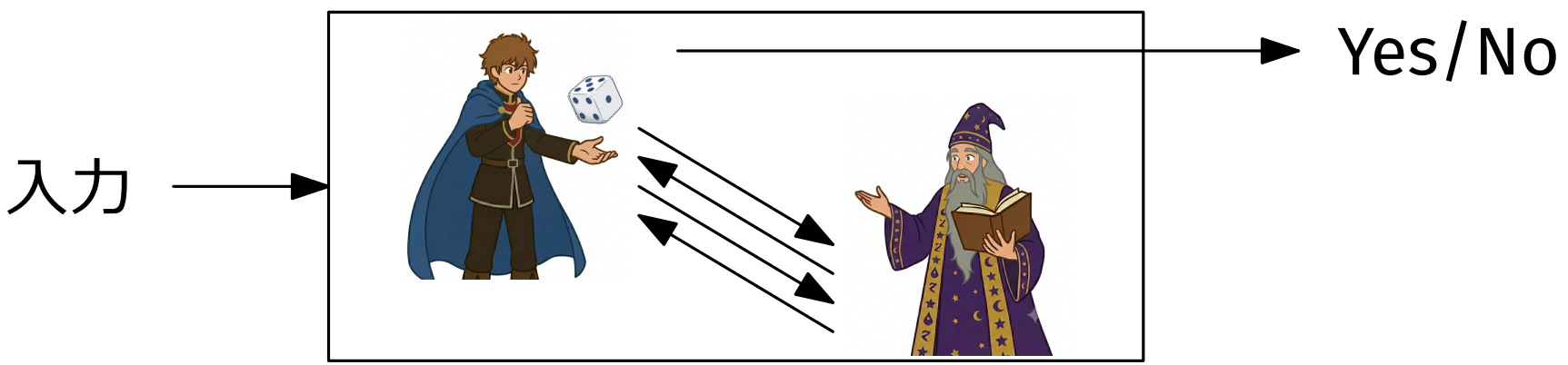
プロトコル A が判定問題 P を解くとは

- 入力が P の Yes インスタンス ⇒
  証明者がうまく動作することで，次を満たす

$$\Pr_r(\text{検証者が Yes を出力}) \geq \frac{2}{3}$$

- 入力が P の No インスタンス ⇒
  証明者がどのように動作しても，次を満たす

$$\Pr_r(\text{検証者が No を出力}) \geq \frac{2}{3}$$



[復習] 対話証明系で問題を解くとは

プロトコル A が判定問題 P を解くとは

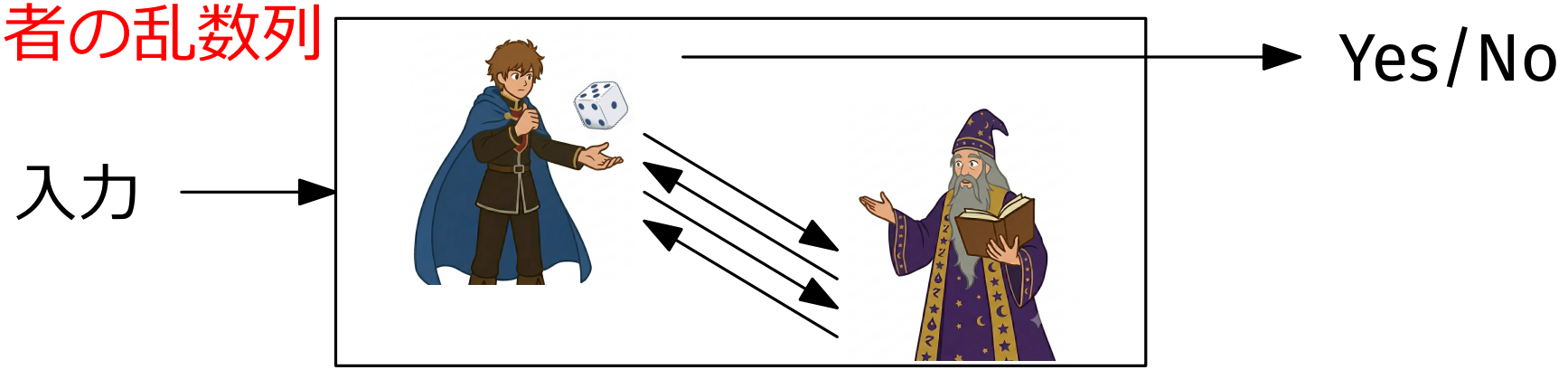
- 入力が P の Yes インスタンス ⇒
  証明者がうまく動作することで，次を満たす

$$\Pr_r(\text{検証者が Yes を出力}) \geq \frac{2}{3}$$

- 入力が P の No インスタンス ⇒
  証明者がどのように動作しても，次を満たす

$$\Pr_r(\text{検証者が No を出力}) \geq \frac{2}{3}$$

検証者の乱数列



[復習] 複雑性クラス IP$[k]$

正整数 k

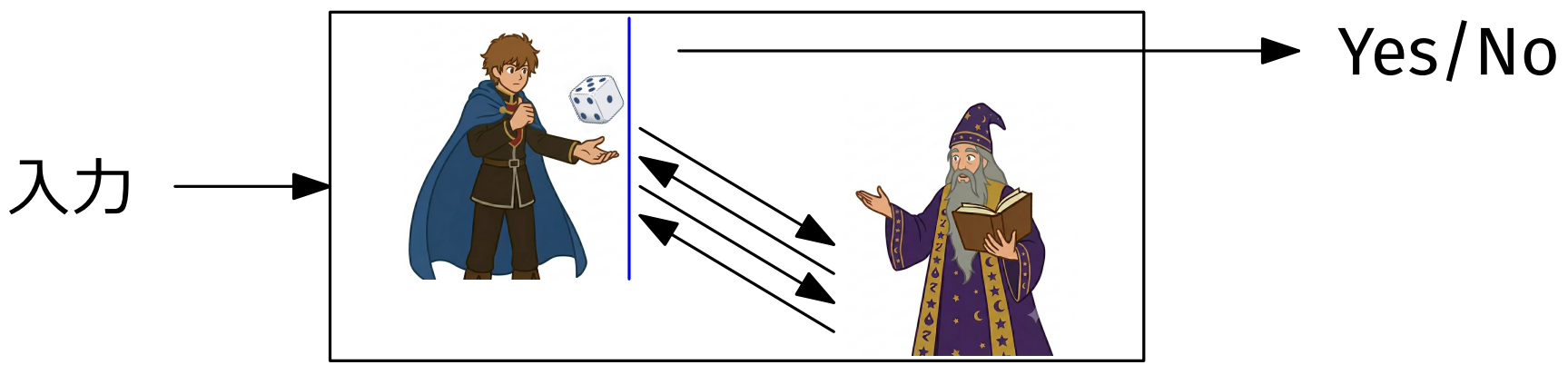
複雑性クラス IP$[k]$ (Goldwasser, Micali, Rackoff '89)

クラス IP$[k]$ は次を満たすプロトコルで解ける問題全体

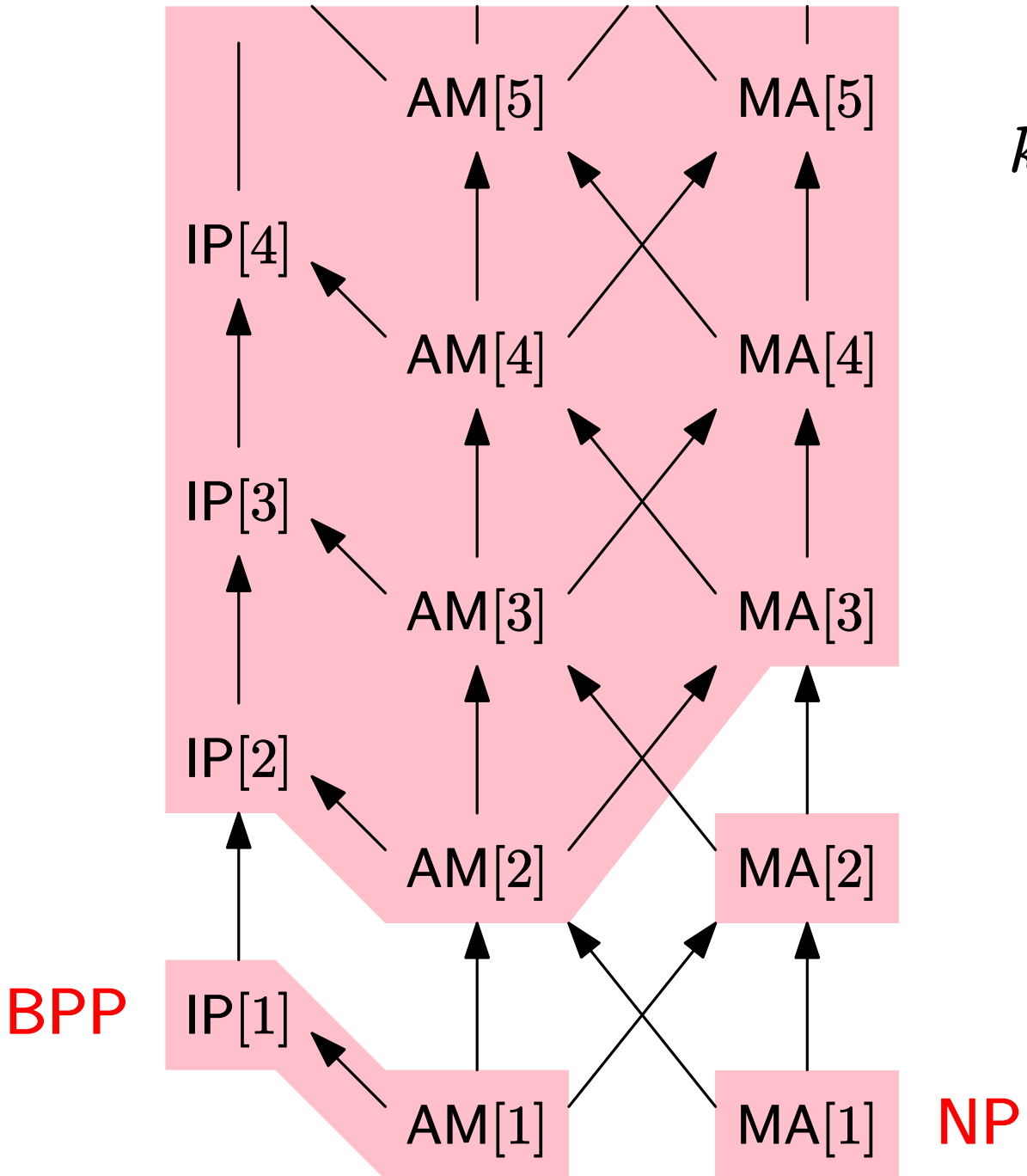
- プロトコルは 検証者 の動作から始まる
- 検証者と証明者が順に計 k ラウンド の動作を行う
- 検証者の乱数は証明者から 見えない (private coin)

IP = Interactive Proof

IP$[4]$ のプロトコル

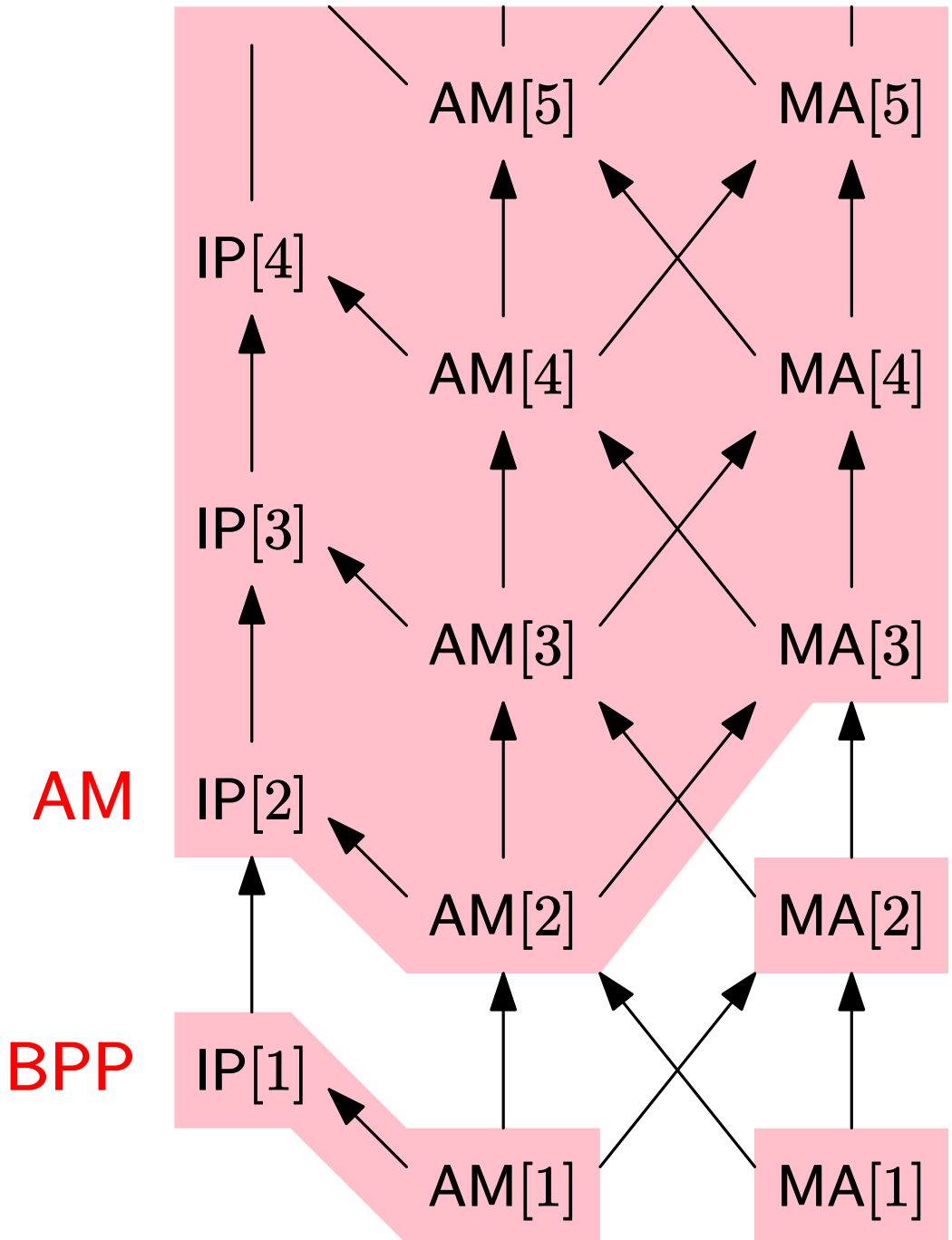


[復習] 複雑性クラスの関係



k が定数である限り

[復習] 複雑性クラスの関係



定義

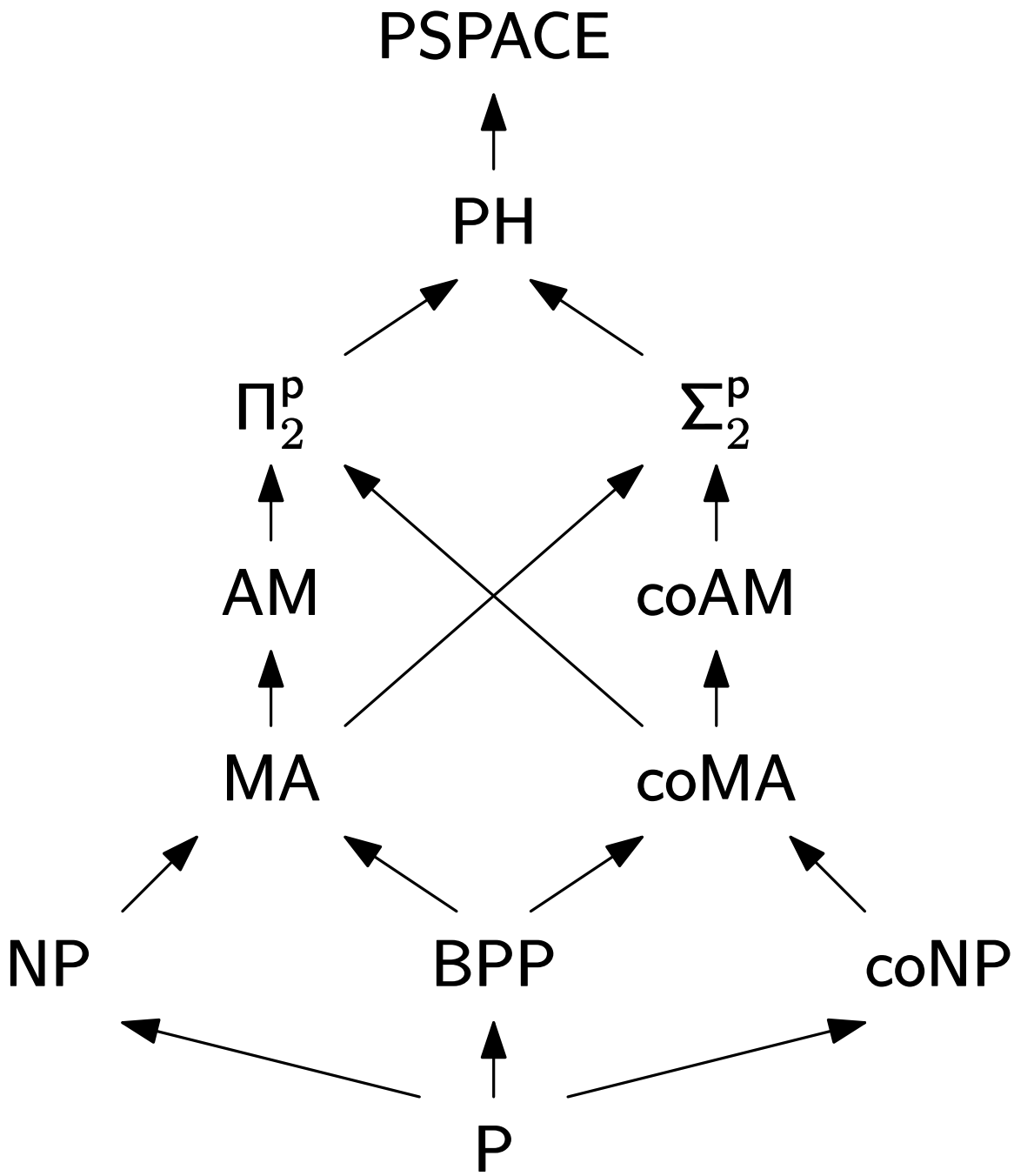
- AM := AM[2]
- MA := MA[2]

性質：まとめ

- NP ⊆ MA ⊆ AM
- BPP ⊆ MA ⊆ AM

「=」の成立は未解決

[復習] 複雑性クラスの関係 (続)



本日の内容

内容

- 複雑性クラス IP を導入する
- IP $\subseteq$ PSPACE を証明する
- PSPACE $\subseteq$ IP の証明の準備をする

結論：IP $=$ PSPACE

本日の内容

1. **複雑性クラス IP**
2. IP $\subseteq$ PSPACE の証明
3. PSPACE $\subseteq$ IP の証明：準備

複雑性クラス IP

定義：複雑性クラス IP

クラス IP とは次で定義される複雑性クラス

$$\mathsf{IP} = \bigcup_{d=1}^{\infty} \mathsf{IP}[n^d]$$

ただし，n は入力の符号長

つまり，

IP とはラウンド数が入力符号長の多項式である対話証明系

注 1：前回の内容の帰結

[復習] 前回の内容

- $\mathsf{AM}[k] \subseteq \mathsf{IP}[k]$ (for $k \geq 1$)
- $\mathsf{MA}[k] \subseteq \mathsf{AM}[k]$ (for $k \geq 2$)
- $\mathsf{AM}[k] = \mathsf{AM}[k+1] = \mathsf{MA}[k+1]$ (for $k \geq 2$) (Babai '85)
- $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$ (Goldwasser, Sipser '87)

つまり，$\mathsf{IP}[2] \subseteq \mathsf{IP}[k] \subseteq \mathsf{AM}[k+2] = \mathsf{AM}[2] \subseteq \mathsf{IP}[2]$

注意

$\mathsf{IP} = \mathsf{IP}[2]$ と言えない

注 1：前回の内容

[復習] 前回の内容

- $\mathsf{AM}[k] \subseteq \mathsf{IP}[k]$ (for $k \geq 1$)
- $\mathsf{MA}[k] \subseteq \mathsf{AM}[k]$ (for $k \geq 2$)
- $\mathsf{AM}[k] = \mathsf{AM}[k+1] = \mathsf{MA}[k+1]$ (for $k \geq 2$) (Babai '85)
- $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$ (Goldwasser, Sipser '87)

つまり，$\mathsf{IP}[2] \subseteq \mathsf{IP}[k] \subseteq \mathsf{AM}[k+2] = \mathsf{AM}[2] \subseteq \mathsf{IP}[2]$

注意

$\mathsf{IP} = \mathsf{IP}[2]$ と言えない

メッセージの符号長が k の指数関数倍だけ増える

注 2：前回の内容の帰結

[復習] 前回の内容

- $\mathsf{AM}[k] \subseteq \mathsf{IP}[k]$ (for $k \geq 1$)
- $\mathsf{MA}[k] \subseteq \mathsf{AM}[k]$ (for $k \geq 2$)
- $\mathsf{AM}[k] = \mathsf{AM}[k+1] = \mathsf{MA}[k+1]$ (for $k \geq 2$) (Babai '85)
- $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$ (Goldwasser, Sipser '87)

帰結

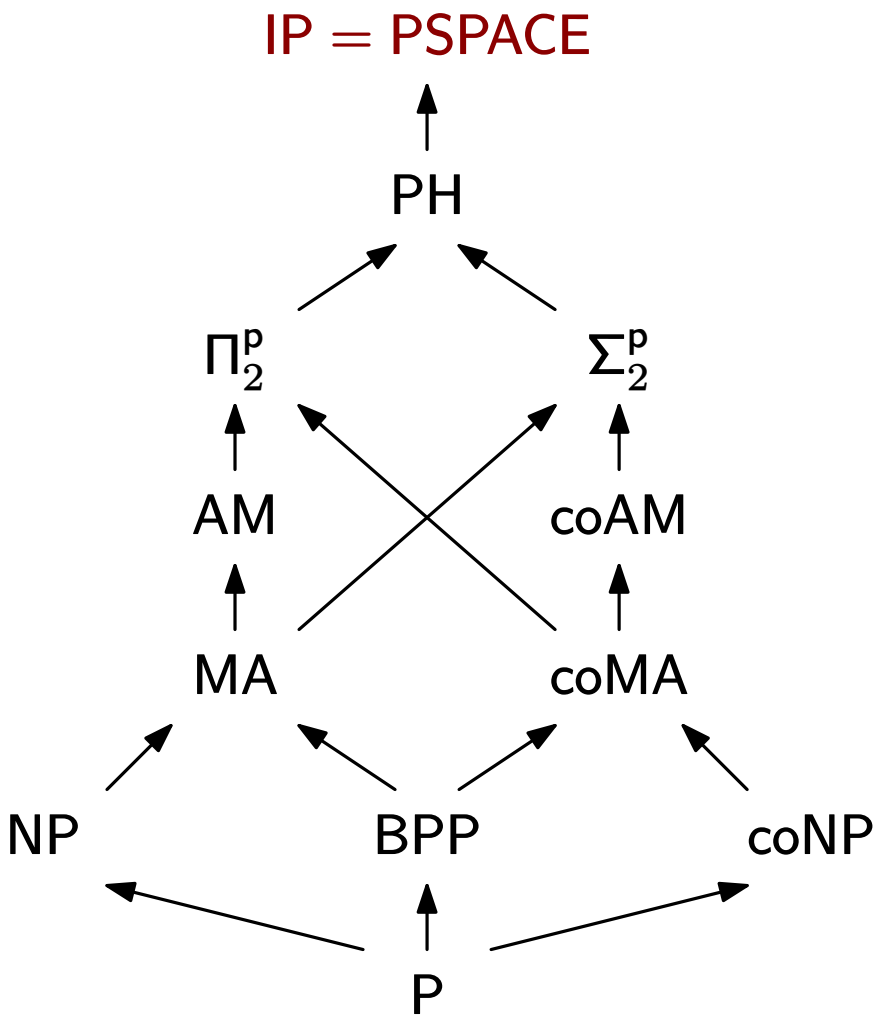
$$\mathsf{IP} = \bigcup_{d=1}^{\infty} \mathsf{AM}[n^d] = \bigcup_{d=1}^{\infty} \mathsf{MA}[n^d]$$

ただし，n は入力の符号長

IP = PSPACE

定理 (Shamir '92)

IP = PSPACE



PSPACE の四態

PSPACE の捉え方

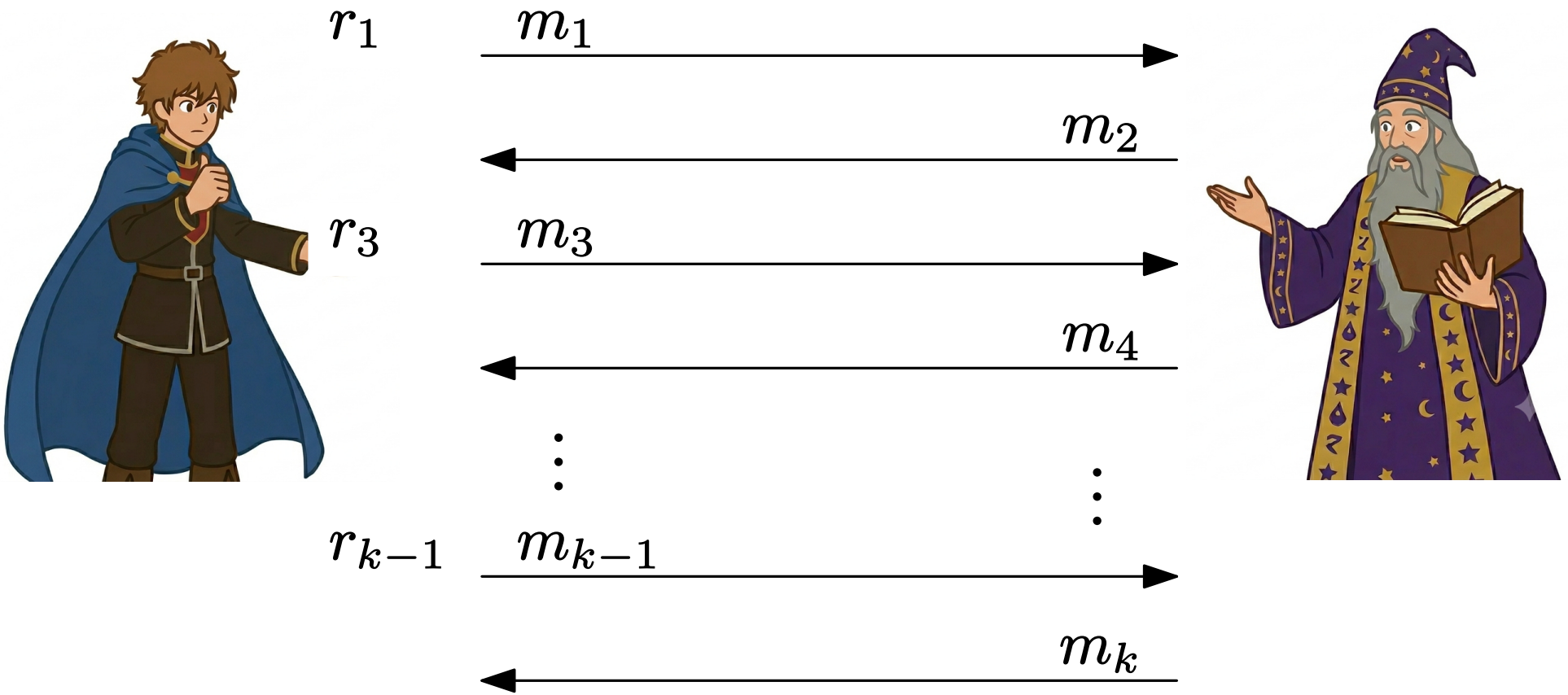
1. 多項式領域アルゴリズムで解ける
   (定義)
2. 非決定性多項式領域アルゴリズムで解ける
   (PSPACE = NPSPACE)
3. 交代性多項式時間アルゴリズムで解ける
   (AP = PSPACE)
4. 多項式ラウンド対話証明系で解ける
   (IP = PSPACE)

本日の内容

1. 複雑性クラス IP
2. IP $\subseteq$ PSPACE **の証明**
3. PSPACE $\subseteq$ IP の証明：準備

IP $\subseteq$ PSPACE (1)：設定

プロトコル A：ラウンド数 k は入力の符号長の多項式



検証者は，メッセージ m_i を作るために 乱数列 r_i を使う

注：$|m_i|$ も $|r_i|$ も入力の符号長の多項式

IP $\subseteq$ PSPACE (1)：設定

プロトコル A：ラウンド数 k は入力の符号長の多項式

r_1 m_1 →

← m_2

r_3 m_3 →

← m_4

⋮ ⋮

r_{k-1} m_{k-1} →

← m_k

成功確率 $\geq 2/3$

検証者は，メッセージ m_i を作るために 乱数列 r_i を使う

注：$|m_i|$ も $|r_i|$ も入力の符号長の多項式

IP ⊆ PSPACE (2)：証明の方針

証明の方針：非決定性多項式領域アルゴリズム B を作る

1. B では「A が Yes を出力する確率」を計算する
2. A が Yes を出力する確率 $\geq 2/3 \Rightarrow B$ の出力は Yes
   A が Yes を出力する確率 $\leq 1/3 \Rightarrow B$ の出力は No
3. $\therefore$ IP ⊆ NPSPACE $=$ PSPACE

Savitch の定理

注：多くの教科書では，上の方針に従わず，
「Yes の出力確率を最大化する」ような証明者を計算する
(決定性) 多項式領域アルゴリズムを与えることで
IP ⊆ PSPACE を証明している

IP $\subseteq$ PSPACE (3)：考え方

r_1 m_1 →

m_2 ←

r_3 m_3 →

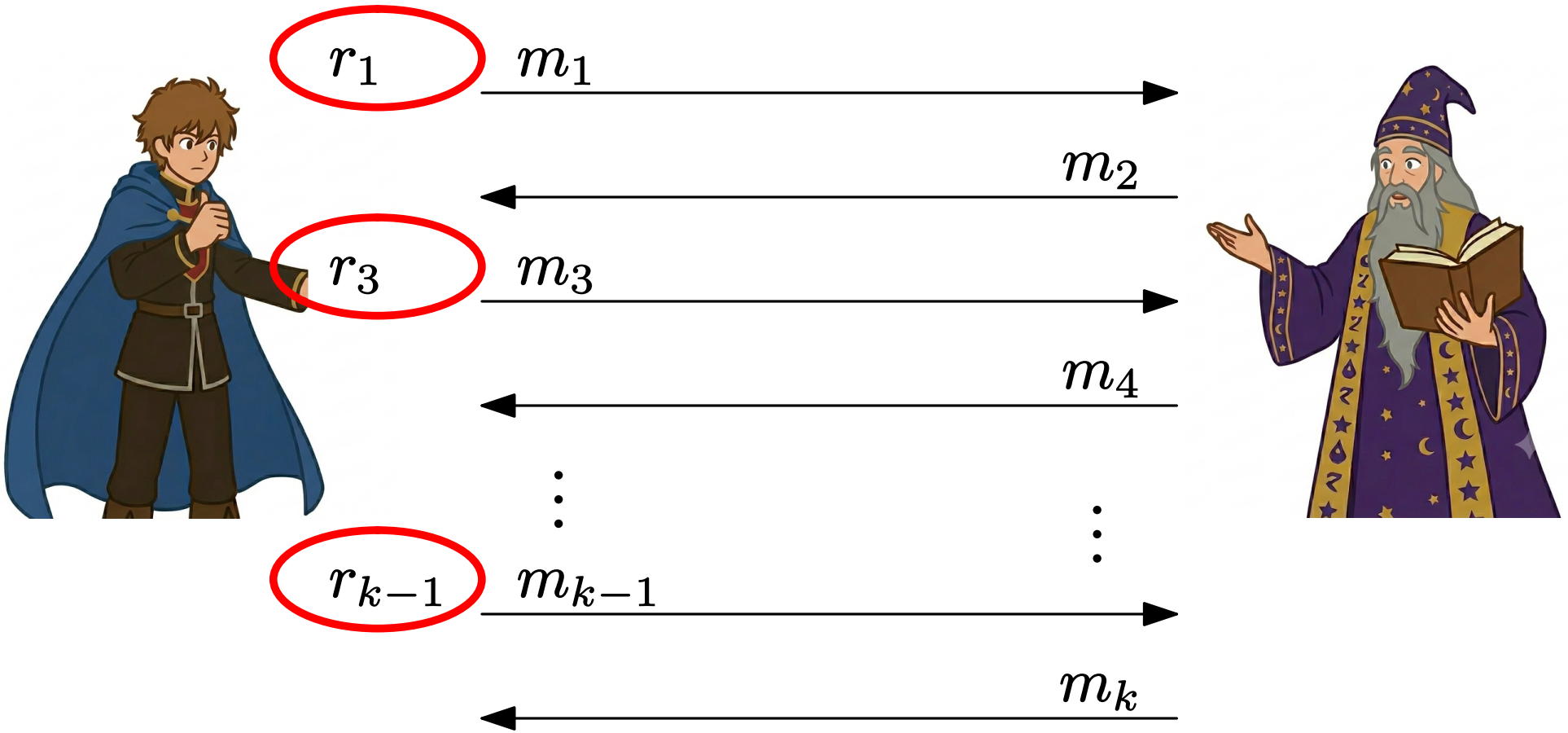
m_4 ←

$\vdots$ $\vdots$

r_{k-1} m_{k-1} →

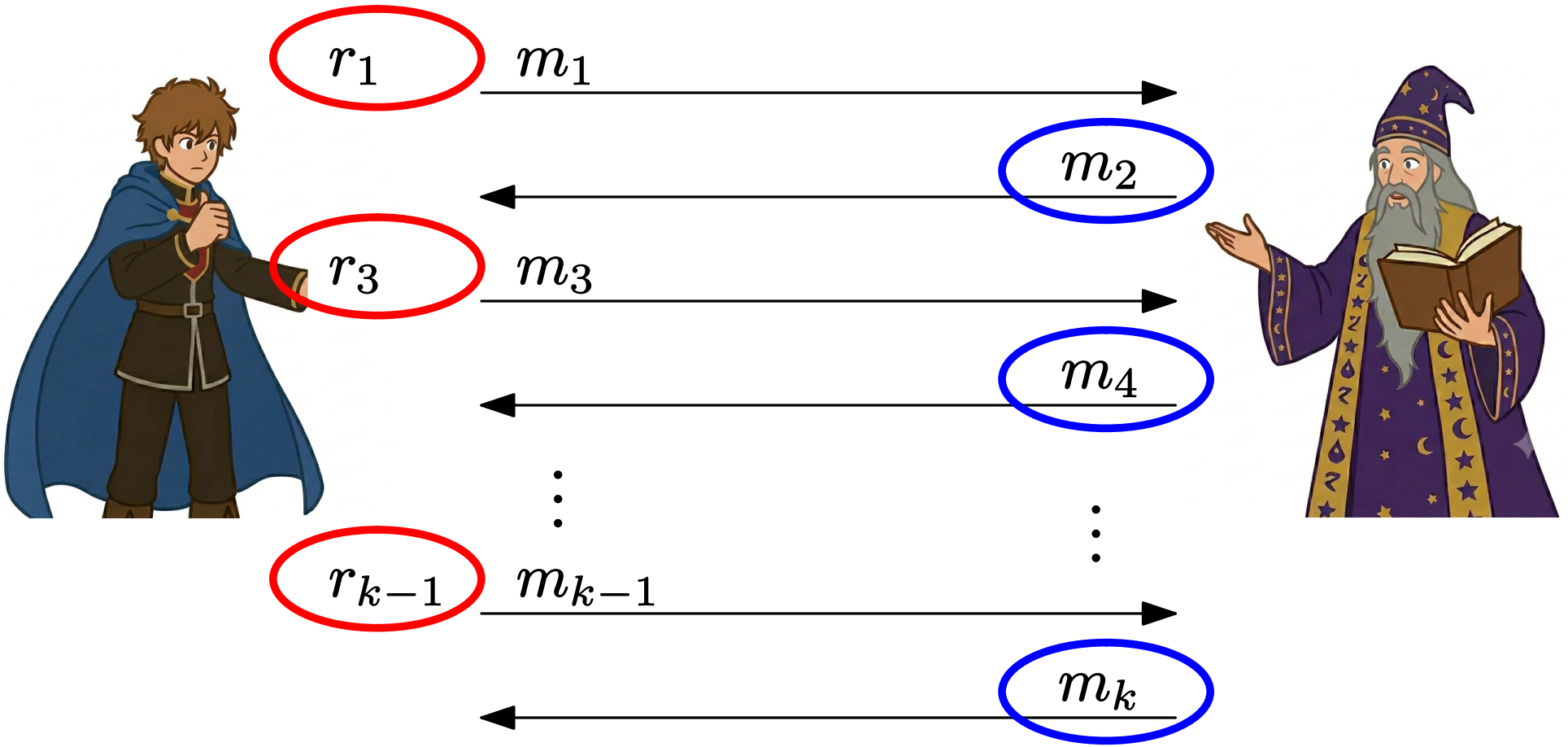
m_k ←

IP ⊆ PSPACE (3)：考え方



すべての可能な乱数列を考える

IP ⊆ PSPACE (3)：考え方



すべての可能な乱数列を考える

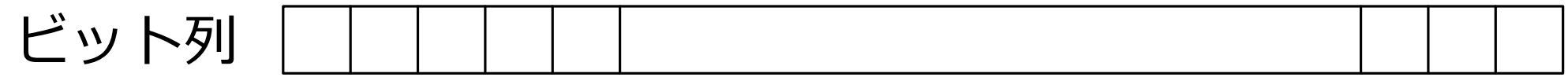
guess をする

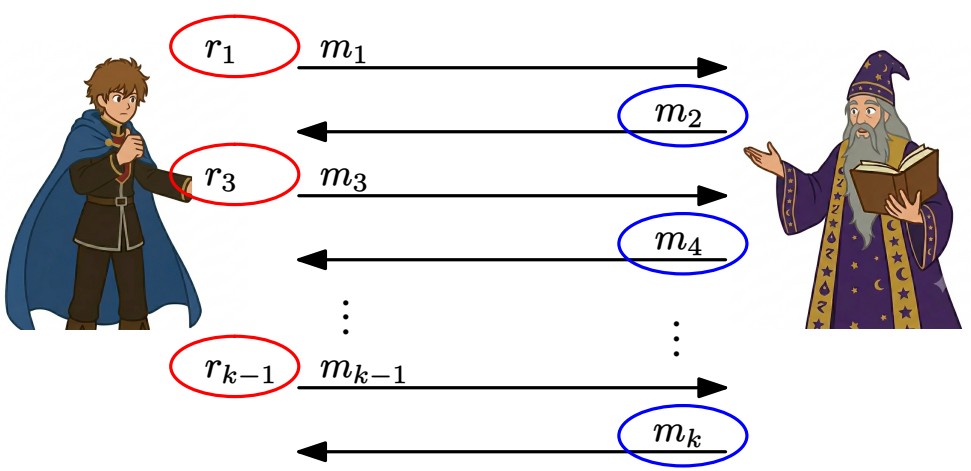
IP ⊆ PSPACE (4)

アルゴリズム B の動き

長さ L (入力から上界が分かる)

ビット列





すべての可能な乱数列を考える

guess をする

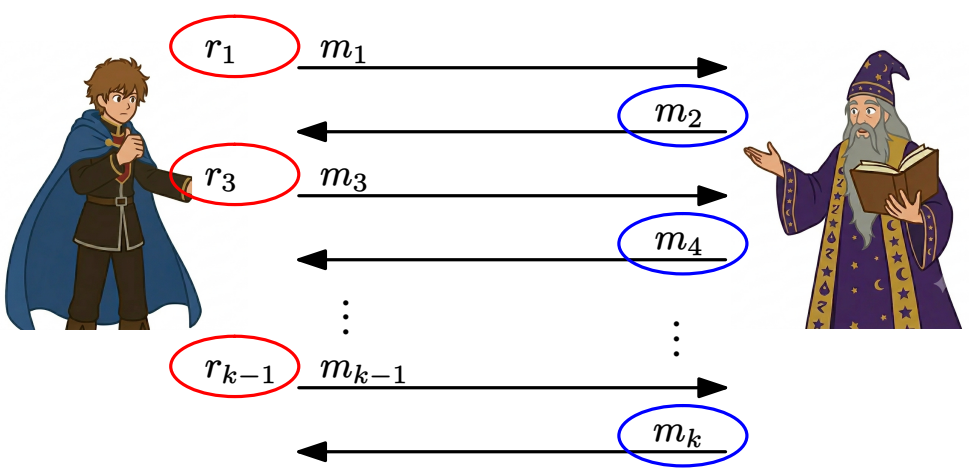
IP ⊆ PSPACE (4)

アルゴリズム B の動き

長さ L (入力から上界が分かる)

ビット列

0	0	1	0	1		1	1	0



すべての可能な乱数列を考える

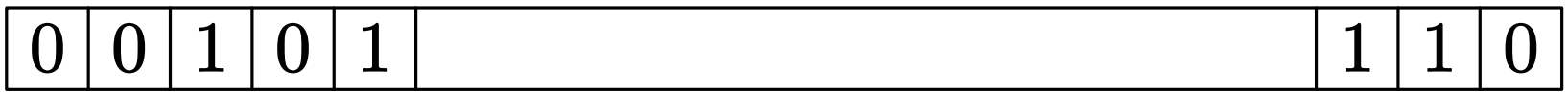
guess をする

IP ⊆ PSPACE (4)

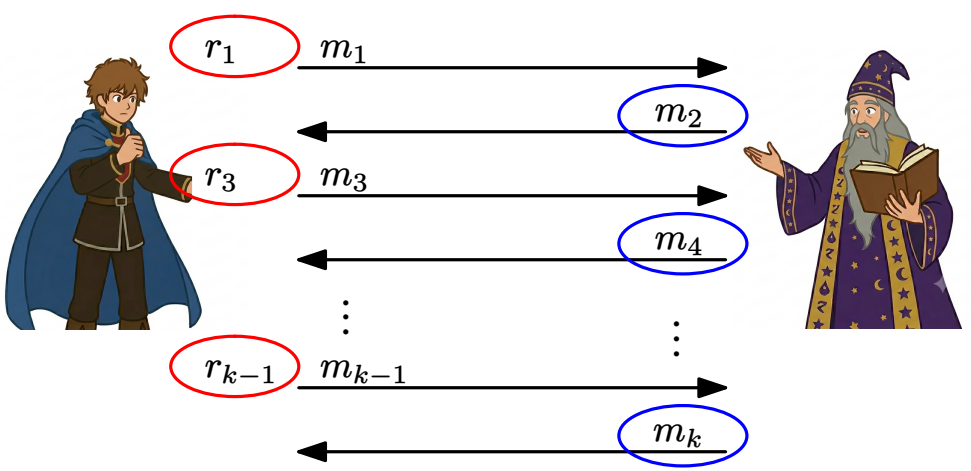
アルゴリズム B の動き

長さ L (入力から上界が分かる)

ビット列



r_1



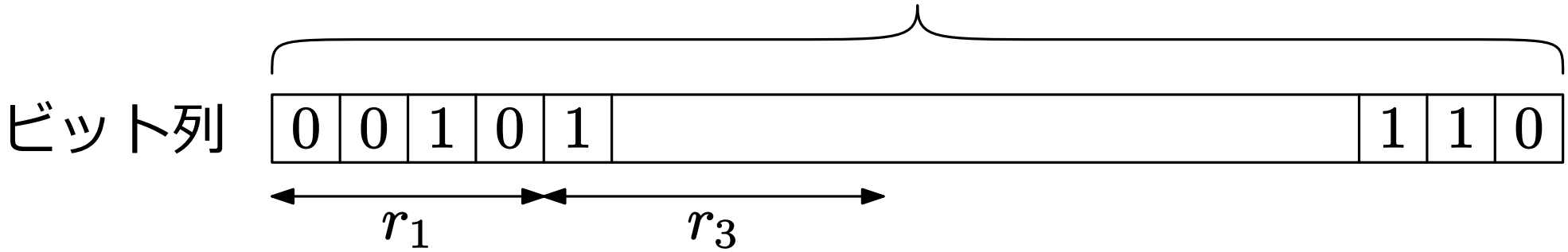
すべての可能な乱数列を考える

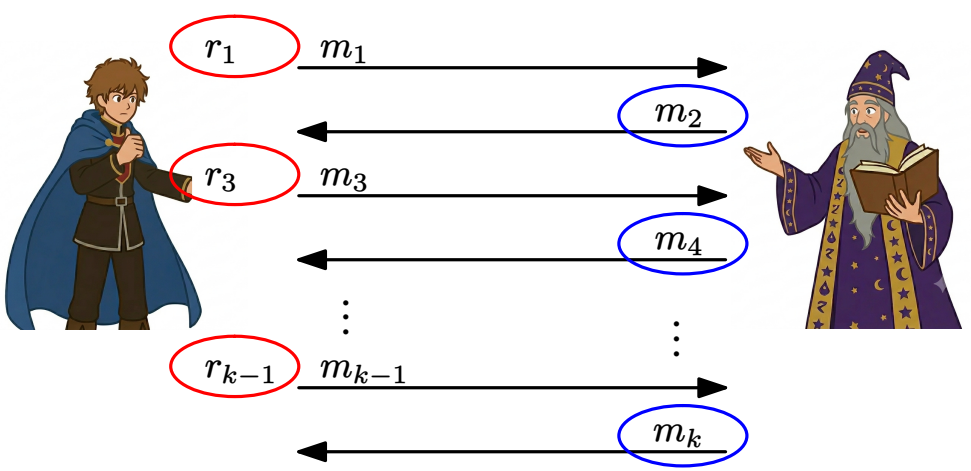
guess をする

IP ⊆ PSPACE (4)

アルゴリズム B の動き

長さ L (入力から上界が分かる)





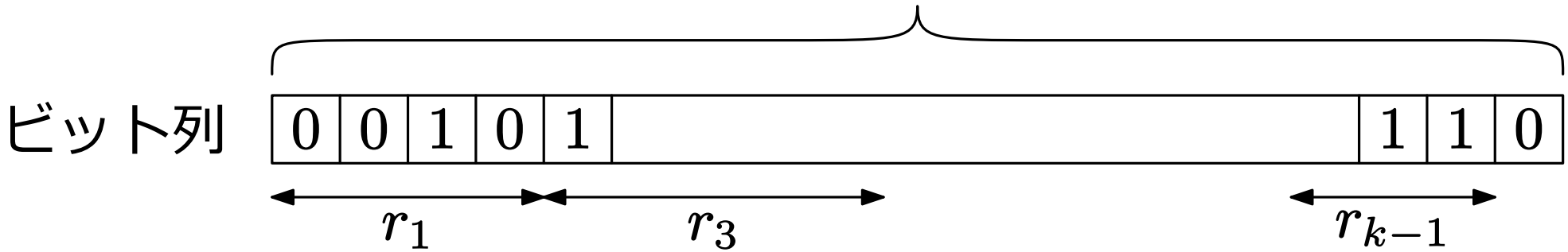
すべての可能な乱数列を考える

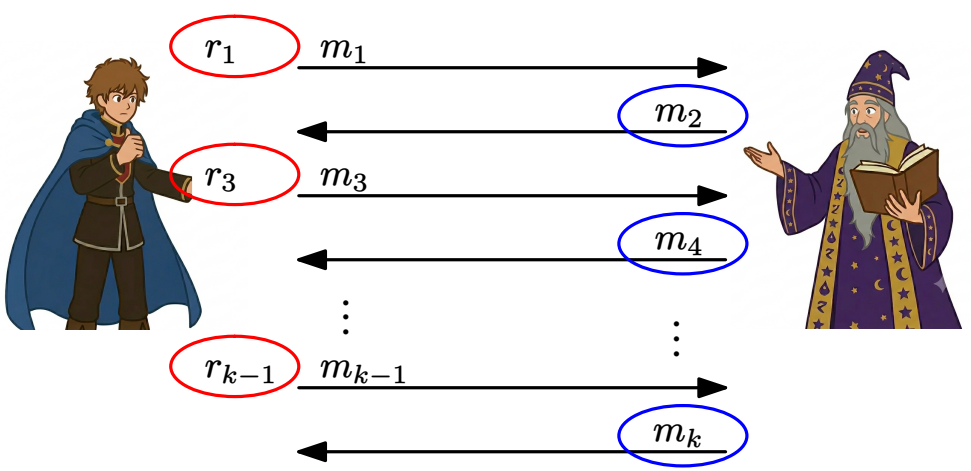
guess をする

IP ⊆ PSPACE (4)

アルゴリズム B の動き

長さ L (入力から上界が分かる)





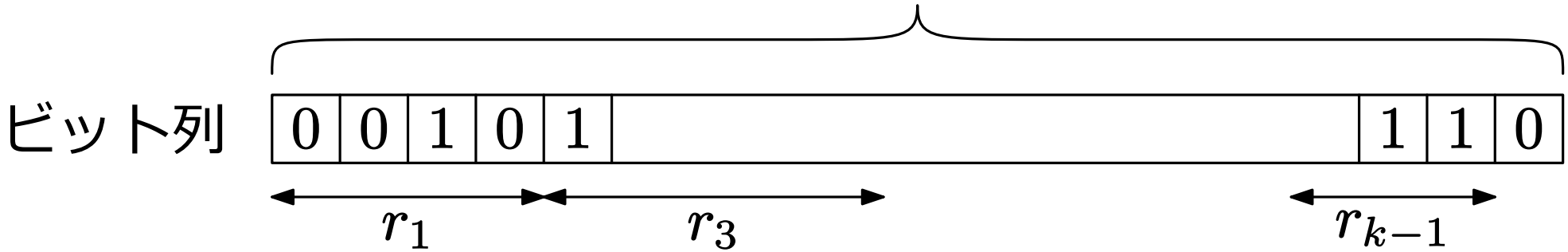
すべての可能な乱数列を考える

guess をする

IP ⊆ PSPACE (4)

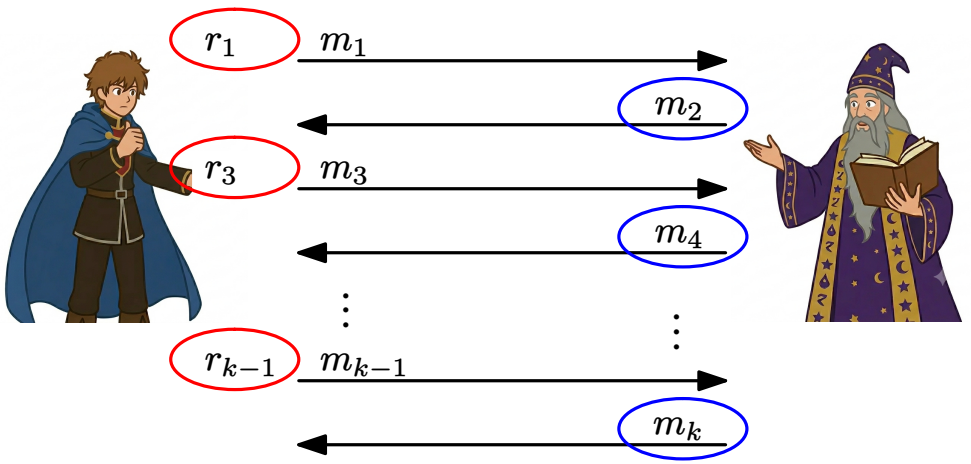
アルゴリズム B の動き

長さ L (入力から上界が分かる)



$\rightsquigarrow$ A が Yes を出力 $\Rightarrow$ Yes の出力確率 $+= \dfrac{1}{2^L}$

A が No を出力 $\Rightarrow$ Yes の出力確率 $+= 0$

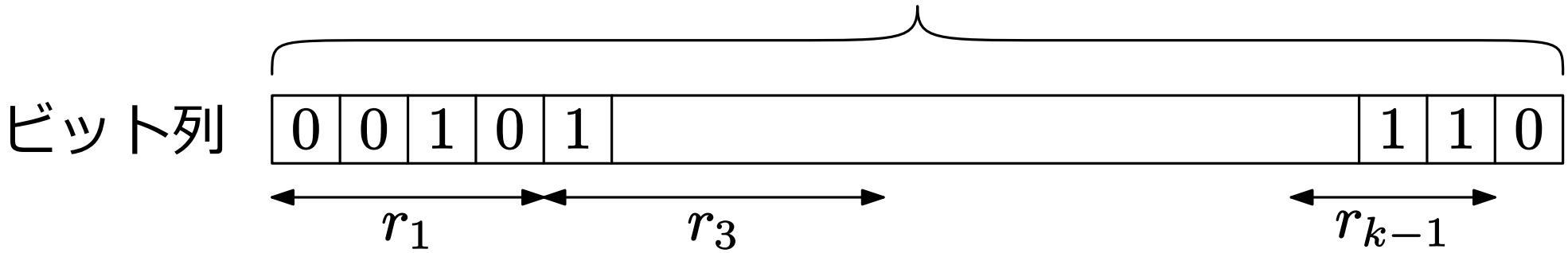


すべての可能な乱数列を考える

guess をする

IP $\subseteq$ PSPACE (4)

アルゴリズム B の動き



$\rightsquigarrow$ A が Yes を出力 $\Rightarrow$ Yes の出力確率 $+= \dfrac{1}{2^L}$

A が No を出力 $\Rightarrow$ Yes の出力確率 $+= 0$

- これで，A が Yes を出力する確率を正しく計算できる
- また，B は非決定性多項式領域アルゴリズム
- $\therefore$ IP $\subseteq$ NPSPACE $=$ PSPACE $\square$

本日の内容

1. 複雑性クラス IP
2. IP $\subseteq$ PSPACE の証明
3. PSPACE $\subseteq$ IP **の証明：準備**

PSPACE ⊆ IP：アイディア

基本的なアイディア

- ある PSPACE 完全問題に対して，
  多項式ラウンドの対話証明系を構成する
- PSPACE 完全問題として QBF を考える
- QBF を解くために，**算術化** を行う

算術化 = arithmetization

以下，算術化の例として，次を行う

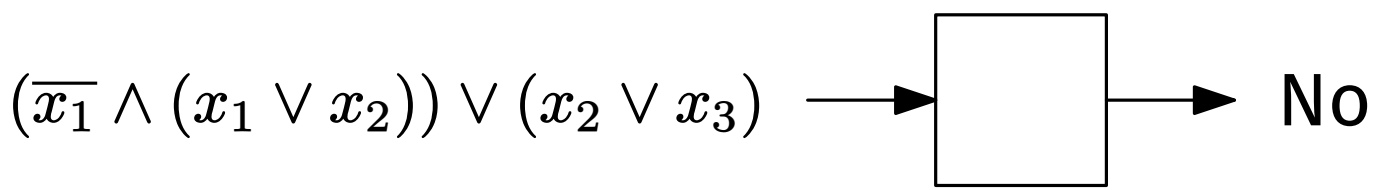
- UNSAT に対して
  多項式ラウンドの対話証明系を構成する

復習：UNSAT

論理式の充足不能性問題 (FORMULA-UNSAT)

入力： 論理式 φ
出力： φ が充足割当を持たない $\Rightarrow$ Yes
φ が充足割当を持つ $\Rightarrow$ No

以下，「FORMULA」は省略し，単に「UNSAT」と呼ぶ



性質 (復習)：UNSAT は coNP 完全

UNSAT を解くためには，次の問題が解ければよい

問題：#SAT

入力： 論理式 φ，非負整数 k
出力： φ の充足割当の数 $= k \Rightarrow$ Yes
φ の充足割当の数 $\neq k \Rightarrow$ No

例：$\varphi(x_1, x_2, x_3) = (\overline{x_1} \wedge (x_1 \vee x_2)) \vee (x_2 \vee x_3)$

UNSAT を解くためには，次の問題が解ければよい

問題：#SAT

入力： 論理式 φ，非負整数 k
出力： φ の充足割当の数 $= k \Rightarrow$ Yes
φ の充足割当の数 $\neq k \Rightarrow$ No

例：$\varphi(x_1, x_2, x_3) = (\overline{x_1} \wedge (x_1 \vee x_2)) \vee (x_2 \vee x_3)$

- $\varphi(0,0,0) = 0$ ・ $\varphi(0,0,1) = 1$
- $\varphi(0,1,0) = 1$ ・ $\varphi(0,1,1) = 1$
- $\varphi(1,0,0) = 0$ ・ $\varphi(1,0,1) = 1$
- $\varphi(1,1,0) = 1$ ・ $\varphi(1,1,1) = 1$ $\therefore \varphi$ の充足割当の数 $= 6$

注：φ が UNSAT の Yes インスタンス
$\Leftrightarrow \langle \varphi, 0 \rangle$ が #SAT の Yes インスタンス

#SAT $\in$ IP (1)：算術化の考え方

例：$\varphi(x_1, x_2, x_3) = (\overline{x_1} \wedge (x_1 \vee x_2)) \vee (x_2 \vee x_3)$

論理演算

$f(x_1, x_2, x_3) = x_2 + x_3 - x_2 x_3$

算術演算

#SAT ∈ IP (1)：算術化の考え方

例：$\varphi(x_1, x_2, x_3) = (\overline{x_1} \wedge (x_1 \vee x_2)) \vee (x_2 \vee x_3)$

論理演算

- $\varphi(0,0,0) = 0$
- $\varphi(0,0,1) = 1$
- $\varphi(0,1,0) = 1$
- $\varphi(0,1,1) = 1$
- $\varphi(1,0,0) = 0$
- $\varphi(1,0,1) = 1$
- $\varphi(1,1,0) = 1$
- $\varphi(1,1,1) = 1$

$f(x_1, x_2, x_3) = x_2 + x_3 - x_2 x_3$

算術演算

- $f(0,0,0) = 0$
- $f(0,0,1) = 1$
- $f(0,1,0) = 1$
- $f(0,1,1) = 1$
- $f(1,0,0) = 0$
- $f(1,0,1) = 1$
- $f(1,1,0) = 1$
- $f(1,1,1) = 1$

#SAT ∈ IP (1)：算術化の考え方

例：$\varphi(x_1, x_2, x_3) = (\overline{x_1} \wedge (x_1 \vee x_2)) \vee (x_2 \vee x_3)$

算術化

論理演算

- $\varphi(0,0,0) = 0$
- $\varphi(0,0,1) = 1$
- $\varphi(0,1,0) = 1$
- $\varphi(0,1,1) = 1$
- $\varphi(1,0,0) = 0$
- $\varphi(1,0,1) = 1$
- $\varphi(1,1,0) = 1$
- $\varphi(1,1,1) = 1$

$f(x_1, x_2, x_3) = x_2 + x_3 - x_2 x_3$

算術演算

- $f(0,0,0) = 0$
- $f(0,0,1) = 1$
- $f(0,1,0) = 1$
- $f(0,1,1) = 1$
- $f(1,0,0) = 0$
- $f(1,0,1) = 1$
- $f(1,1,0) = 1$
- $f(1,1,1) = 1$

#SAT $\in$ IP (2)：算術化の手順

論理演算	$\longrightarrow$	算術演算
$x \wedge y$		xy
$x \vee y$		$1-(1-x)(1-y)$
$\overline{x}$		$1-x$

この手順を
再帰的に
適用する

#SAT $\in$ IP (2)：算術化の手順

論理演算	$\longrightarrow$	算術演算
$x \wedge y$		xy
$x \vee y \equiv \overline{\overline{x} \wedge \overline{y}}$		$1-(1-x)(1-y)$
$\overline{x}$		$1-x$

この手順を
再帰的に
適用する

#SAT ∈ IP (2)：算術化の手順

論理演算		算術演算
$x \wedge y$		xy
$x \vee y$	$\equiv \overline{\overline{x} \wedge \overline{y}}$	$1-(1-x)(1-y)$
$\overline{x}$		$1-x$

この手順を
再帰的に
適用する

例：$(\overline{x_1} \wedge (x_1 \vee x_2)) \vee (x_2 \vee x_3)$

$$1-(1-(1-x_1)(1-(1-x_1)(1-x_2)))(1-(1-(1-x_2)(1-x_3)))$$

#SAT ∈ IP (2)：算術化の手順

論理演算 ⟶	算術演算
$x \wedge y$	xy
$x \vee y \equiv \overline{\overline{x} \wedge \overline{y}}$	$1-(1-x)(1-y)$
$\overline{x}$	$1-x$

この手順を
再帰的に
適用する

例：$(\overline{x_1} \wedge (x_1 \vee x_2)) \vee (x_2 \vee x_3)$

↓

$$1-(1-(1-x_1)(1-(1-x_1)(1-x_2)))(1-(1-(1-x_2)(1-x_3)))$$

これは x_1, x_2, x_3 に関する多項式

- 変数の数 = 元の論理式 φ の変数の数
- 各変数の次数 $\leq |\varphi|$ (論理式 φ の符号長)

#SAT ∈ IP (3)：算術化で解く

論理式 $\varphi(x_1, x_2, \ldots, x_n)$ $\xrightarrow{\text{算術化}}$ 多項式 $f(x_1, x_2, \ldots, x_n)$

$\varphi(x_1, x_2, \ldots, x_n)$ の充足割当の数

$$= \sum_{a_1 \in \{0,1\}} \sum_{a_2 \in \{0,1\}} \cdots \sum_{a_n \in \{0,1\}} f(a_1, a_2, \ldots, a_n)$$

#SAT $\in$ IP (3)：算術化で解く

論理式 $\varphi(x_1, x_2, \ldots, x_n)$ $\xrightarrow{\text{算術化}}$ 多項式 $f(x_1, x_2, \ldots, x_n)$

$\varphi(x_1, x_2, \ldots, x_n)$ の充足割当の数

$$= \sum_{a_1 \in \{0,1\}} \sum_{a_2 \in \{0,1\}} \cdots \sum_{a_n \in \{0,1\}} f(a_1, a_2, \ldots, a_n)$$

検証者：(確率的多項式時間で) 計算できない

証明者：計算できる

#SAT ∈ IP (4) : プロトコル (準備)

記法 : 各 $i \in \{0, 1, \ldots, n\}$ に対して

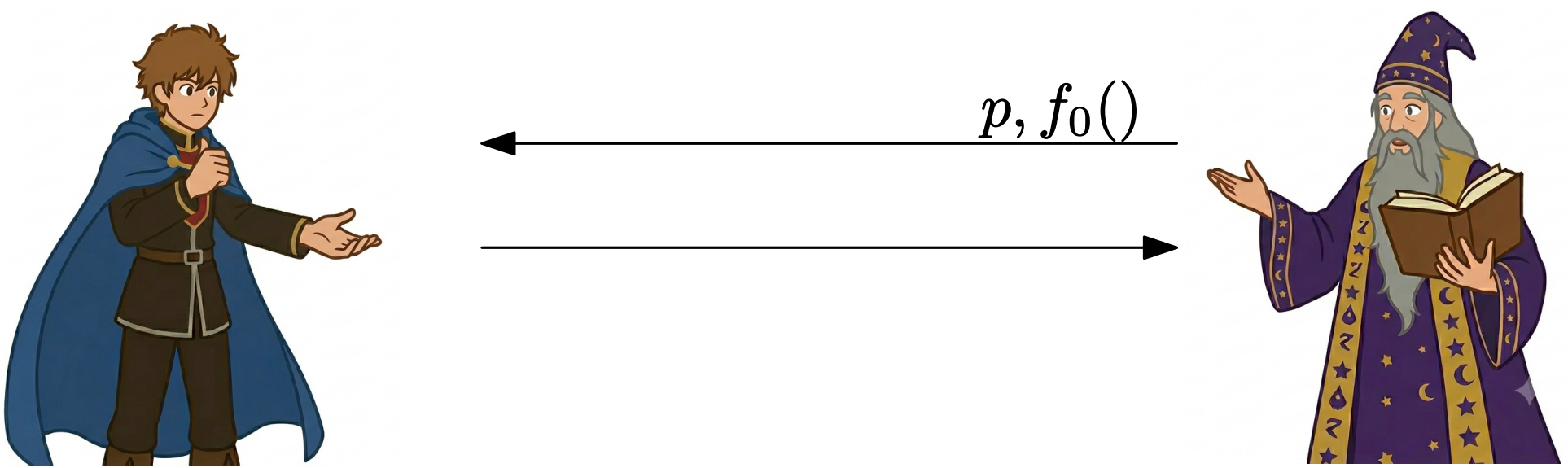
$$f_i(x_1, \ldots, x_i) = \sum_{a_{i+1} \in \{0,1\}} \cdots \sum_{a_n \in \{0,1\}} f(x_1, \ldots, x_i, a_{i+1}, \ldots, a_n)$$

注意

- $f_i(x_1, \ldots, x_i)$ の変数の数は i
- $f_n(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$
- $f_i(x_1, \ldots, x_i) = f_{i+1}(x_1, \ldots, x_i, 0) + f_{i+1}(x_1, \ldots, x_i, 1)$
  (多項式として等しい)
- $f_0() = \varphi$ の充足割当の数

#SAT $\in$ IP (5) : プロトコル (1)

0. (検証者) スキップ
1. (証明者) 十分大きな素数 p と $f_0()$ を送付
2. (検証者) p が素数であることと $f_0() = k$ を確認
   ($\rightsquigarrow$ そうでないとき，ただちに No を出力)
   証明者に次のメッセージを要求

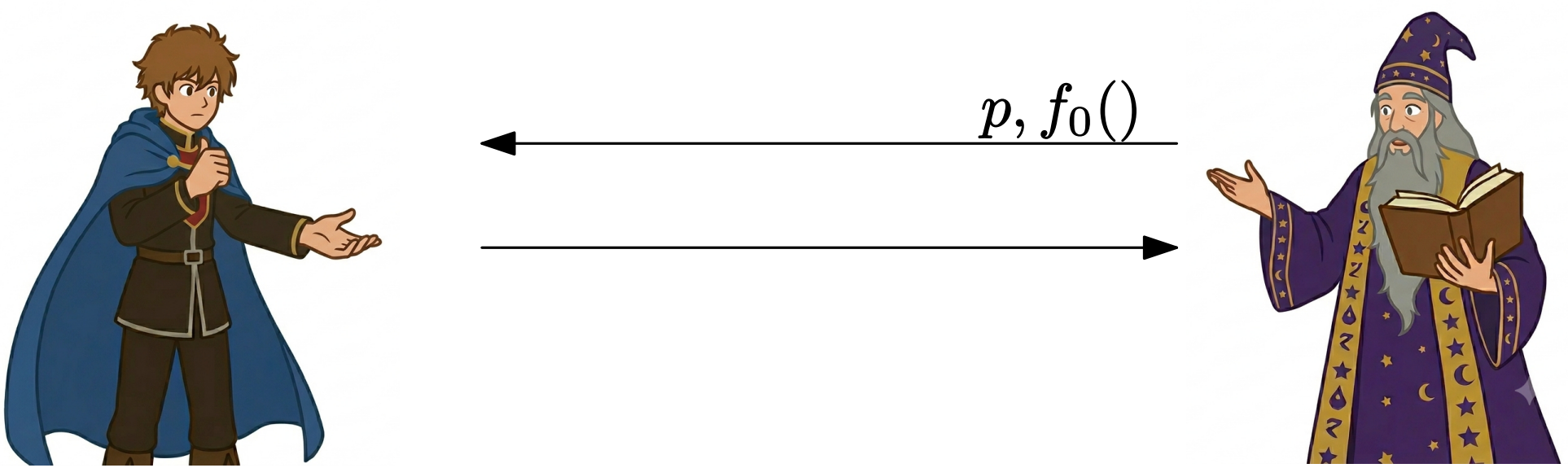


#SAT ∈ IP (5) : プロトコル (1)

0. (検証者) スキップ
1. (証明者) 十分大きな素数 p と $f_0()$ を送付
2. (検証者) p が素数であることと $f_0() = k$ を確認
   (↝ そうでないとき，ただちに No を出力)
   証明者に次のメッセージを要求

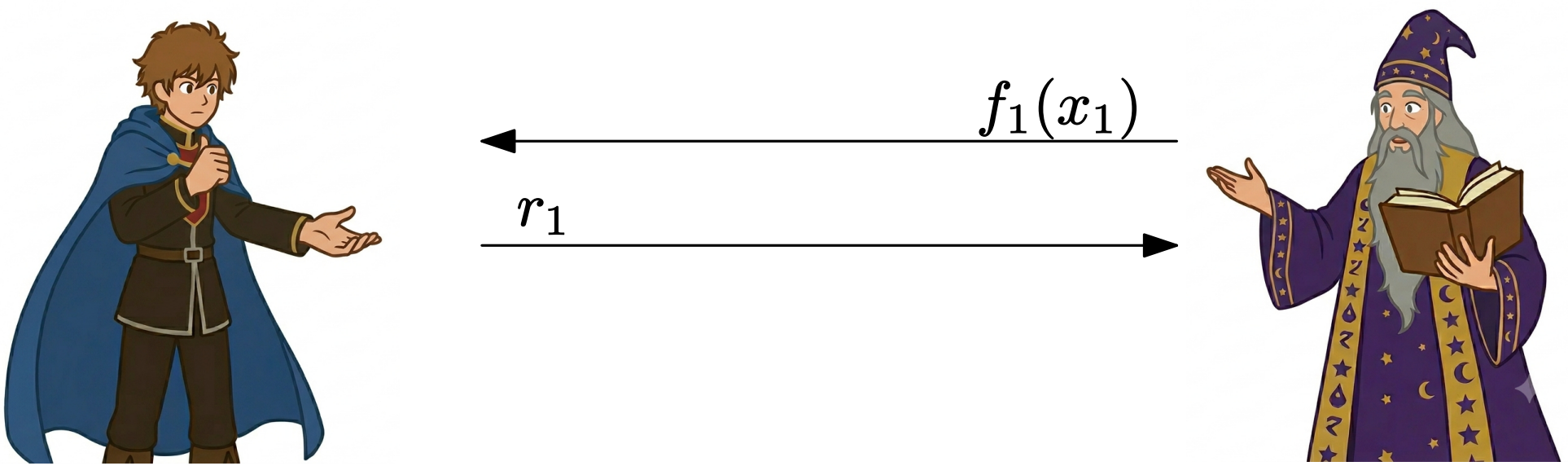
$2^{|\varphi|} < p \leq 2 \cdot 2^{|\varphi|}$ とする (ベルトランの仮説)

注：素数判定 ∈ RP (Miller '76, Rabin '80)
素数判定 ∈ P (Agrawal, Kayal, Saxena '04)



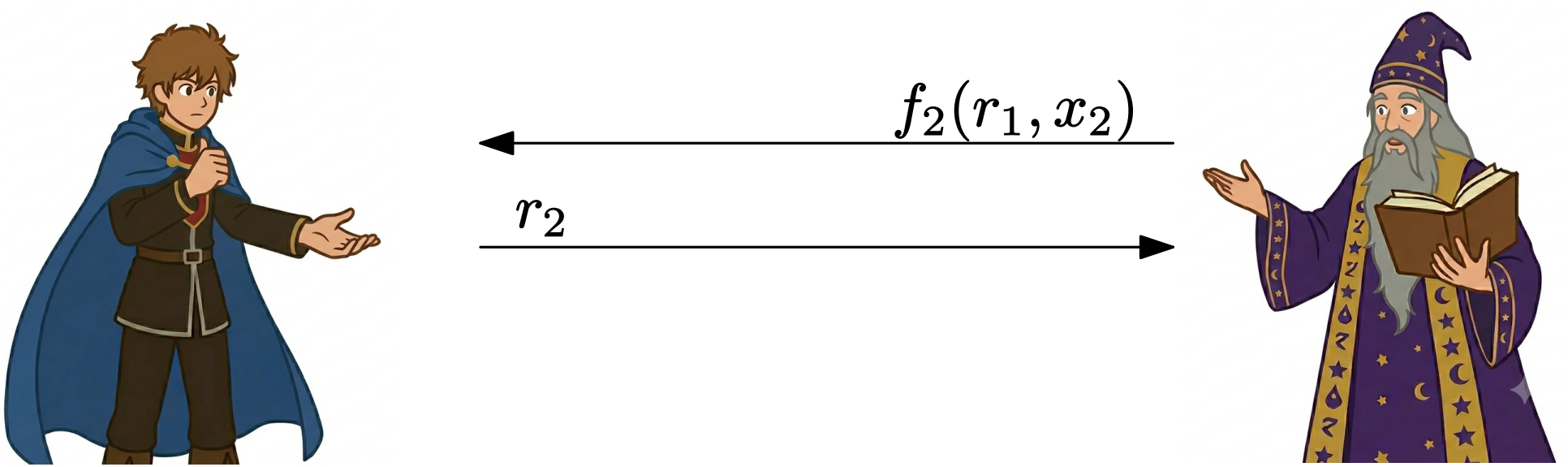
#SAT ∈ IP (5) : プロトコル (2)

3. (証明者) $f_1(x_1)$ を送付
4. (検証者) $f_0() = f_1(0) + f_1(1)$ を確認
   ($\rightsquigarrow$「$\neq$」のとき，ただちに No を出力)
   $r_1 \in \{0, 1, \ldots, p-1\}$ をランダムに作成し，送付



#SAT ∈ IP (5) : プロトコル (3)

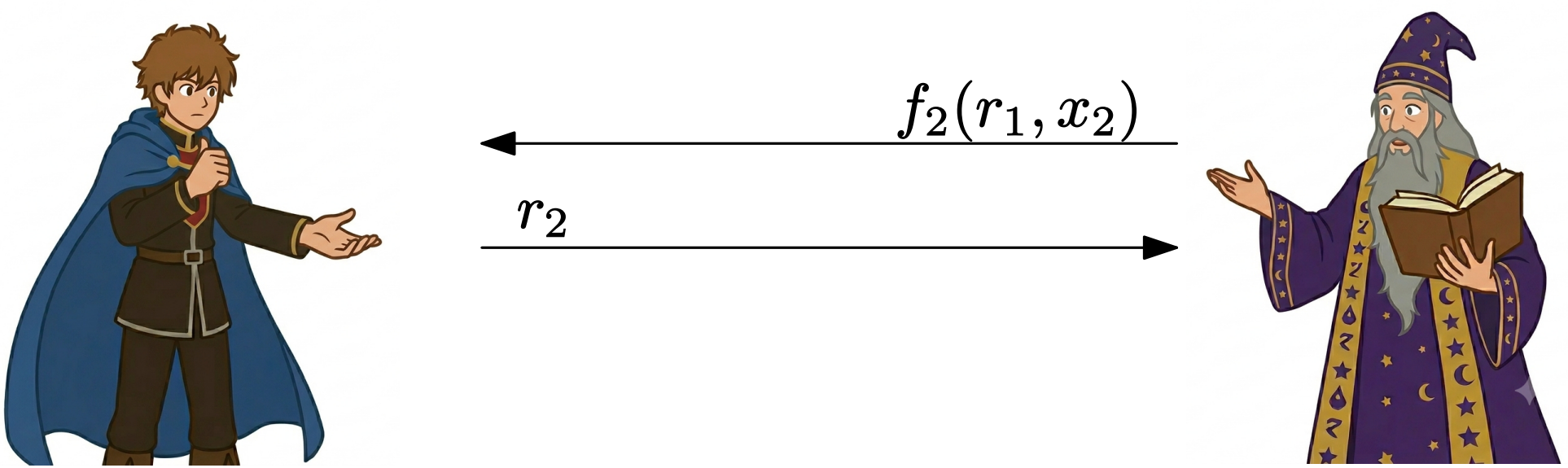
5. (証明者) $f_2(r_1, x_2)$ を送付
6. (検証者) $f_1(r_1) = f_2(r_1, 0) + f_2(r_1, 1)$ を確認
   ($\rightsquigarrow$「$\neq$」のとき，ただちに No を出力)
   $r_2 \in \{0, 1, \ldots, p-1\}$ をランダムに作成し，送付



#SAT ∈ IP (5) : プロトコル (3)

5. (証明者) $f_2(r_1, x_2)$ を送付
6. (検証者) $f_1(r_1) = f_2(r_1, 0) + f_2(r_1, 1)$ を確認
   ($\rightsquigarrow$「$\neq$」のとき，ただちに No を出力)
   $r_2 \in \{0, 1, \ldots, p-1\}$ をランダムに作成し，送付

算術演算は $\mathrm{mod}\ p$ で行う



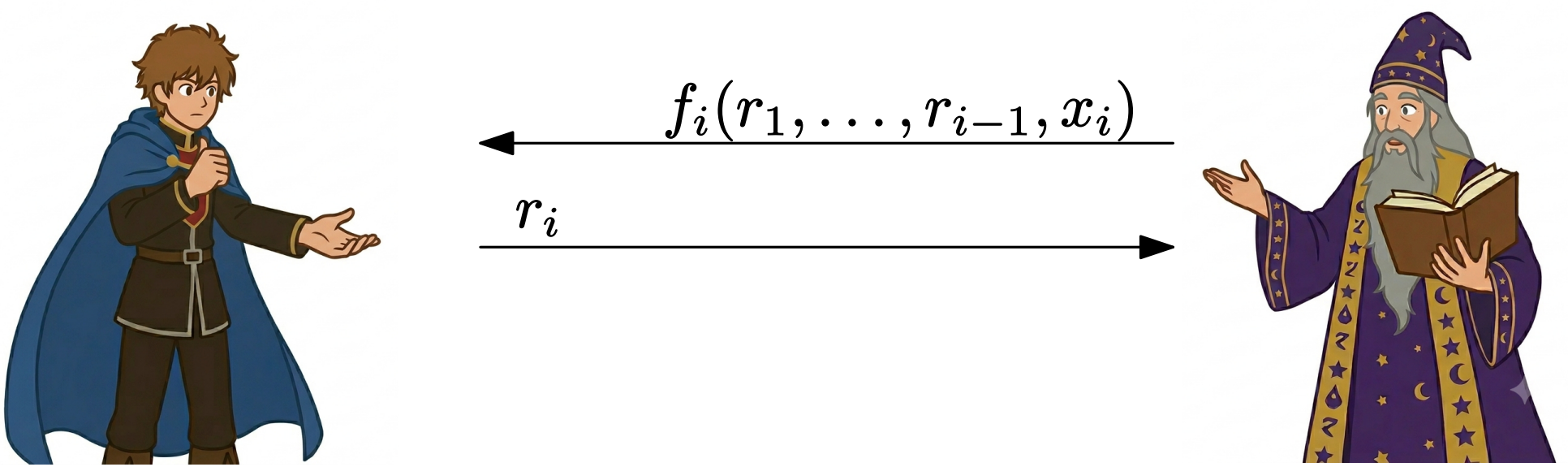
#SAT ∈ IP (6) : プロトコル (4)

$2i+1$. (証明者) $f_i(r_1, \ldots, r_{i-1}, x_i)$ を送付

$2i+2$. (検証者)

$f_{i-1}(r_1, \ldots, r_{i-1}) = f_i(r_1, \ldots, r_{i-1}, 0) + f_i(r_1, \ldots, r_{i-1}1)$ を確認 ($\rightsquigarrow$「$\neq$」のとき，ただちに No を出力)

$r_i \in \{0, 1, \ldots, p-1\}$ をランダムに作成し，送付



#SAT ∈ IP (6) : プロトコル (5)

$2n+1$. (証明者) $f_n(r_1,\ldots,r_{n-1},x_n)$ を送付

$2n+2$. (検証者)

$f_{n-1}(r_1,\ldots,r_{n-1}) = f_n(r_1,\ldots,r_{n-1},0)+f_n(r_1,\ldots,r_{n-1},1)$ を確認 ($\rightsquigarrow$「$\neq$」のとき，ただちに No を出力)

$r_n \in \{0,1,\ldots,p-1\}$ をランダムに作成し，

$f_n(r_1,\ldots,r_n) = f(r_1,\ldots,r_n)$ を確認

$\rightsquigarrow$「$=$」のとき Yes，「$\neq$」のとき No を出力

$f_n(r_1,\ldots,r_{n-1},x_n)$

#SAT ∈ IP (7)：正当性 (1)

プロトコルが Yes を出力する ⇔

- $f_0() = k$
- 任意の $i = 1, 2, \ldots, n$ に対して
  $f_{i-1}(r_1, \ldots, r_{i-1}) = f_i(r_1, \ldots, r_{i-1}, 0) + f_i(r_1, \ldots, r_{i-1}, 1)$
- $f_n(r_1, \ldots, r_n) = f(r_1, \ldots, r_n)$

f と f_i の作り方から，次がただちに分かる

$$\Pr_{r_1,\ldots,r_n}(\text{Yes を出力} \mid \text{入力が Yes インスタンス}) = 1$$

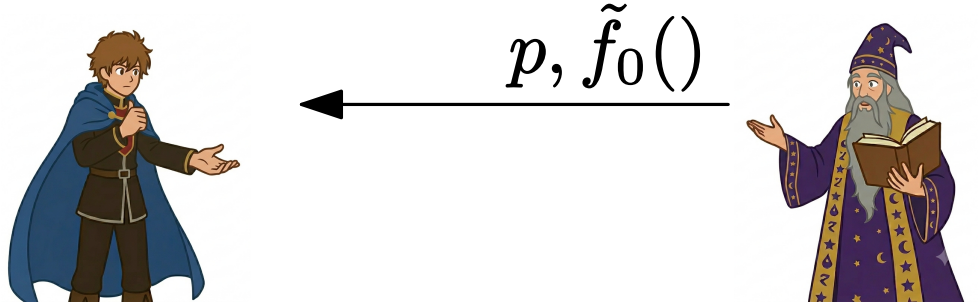
#SAT ∈ IP (8)：正当性 (2)

プロトコルが Yes を出力する ⇔

- $f_0() = k$
- 任意の $i = 1, 2, \ldots, n$ に対して
  $f_{i-1}(r_1, \ldots, r_{i-1}) = f_i(r_1, \ldots, r_{i-1}, 0) + f_i(r_1, \ldots, r_{i-1}, 1)$
- $f_n(r_1, \ldots, r_n) = f(r_1, \ldots, r_n)$

入力が No インスタンスであるときを考える

- 証明者は 検証者に Yes と言わせたいので，
  $\tilde{f}_0() = k$ を満たす嘘メッセージ $\tilde{f}_0()$ を送る
- (以下，証明者のメッセージに「~」をつける)



#SAT ∈ IP (9)：正当性 (3)

プロトコルが Yes を出力する ⇔

- $f_0() = k$
- 任意の $i = 1, 2, \ldots, n$ に対して
  $f_{i-1}(r_1, \ldots, r_{i-1}) = f_i(r_1, \ldots, r_{i-1}, 0) + f_i(r_1, \ldots, r_{i-1}, 1)$
- $f_n(r_1, \ldots, r_n) = f(r_1, \ldots, r_n)$

入力が No インスタンスであるときを考える

- $\tilde{f}_0() \neq f_0()$ かつ $\tilde{f}_1(0) + \tilde{f}_1(1) = \tilde{f}_0()$ なので，$\tilde{f}_1(x_1)$ は嘘
- $\tilde{f}_1(r_1)$ は高確率で嘘 $\rightsquigarrow$ 確率を抑える

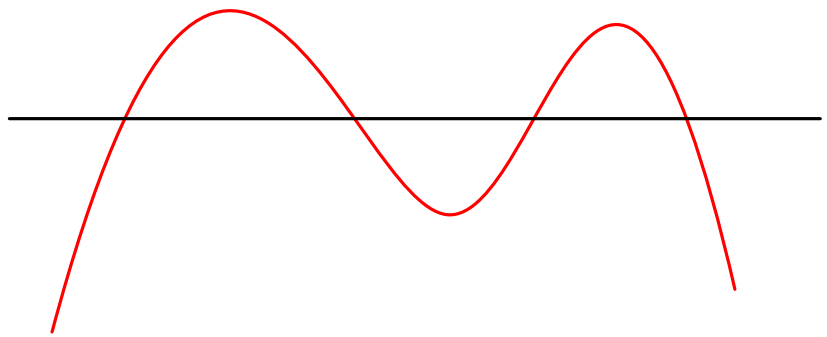
$$\Pr_{r_1}(\tilde{f}_1(r_1) = f_1(r_1)) \leq \text{???}$$

#SAT ∈ IP (10)：正当性 (4)

観察

No インスタンス ⇒

$$\Pr_{r_1}(\tilde{f}_1(r_1) = f_1(r_1)) \leq \frac{1}{|\varphi|^2}$$



- 多項式 $f_1(x_1)$ の次数 $\leq |\varphi|$
- $\therefore$ 多項式 $\tilde{f}_1(x_1)$ の次数 $\leq |\varphi|$ ($\because$ o.w. すぐに No 出力)
- $\therefore \tilde{f}_1(x_1) - f_1(x_1)$ は次数 $\leq |\varphi|$ の多項式
- $\therefore \tilde{f}_1(a) - f_1(a) = 0$ を満たす a は高々 $|\varphi|$ 個
- $\therefore \Pr_{r_1}(\tilde{f}_1(r_1) = f_1(r_1)) \leq \frac{|\varphi|}{p} \leq \frac{|\varphi|}{2^{|\varphi|}} \leq \frac{1}{|\varphi|^2}$

#SAT ∈ IP (11)：正当性 (5)

プロトコルが Yes を出力する ⇔

- $f_0() = k$
- 任意の $i = 1, 2, \ldots, n$ に対して
  $f_{i-1}(r_1, \ldots, r_{i-1}) = f_i(r_1, \ldots, r_{i-1}, 0) + f_i(r_1, \ldots, r_{i-1}, 1)$
- $f_n(r_1, \ldots, r_n) = f(r_1, \ldots, r_n)$

入力が No インスタンスであるときを考える

- 高確率で $\tilde{f}_1(r_1) \neq f_1(r_1)$ であり，かつ，
  $\tilde{f}_2(r_1, 0) + \tilde{f}_2(r_1, 1) = \tilde{f}_1(r_1)$ なので，
  $\tilde{f}_2(r_1, x_2)$ は高確率で嘘
- $\tilde{f}_2(r_1, r_2)$ は高確率で嘘 $\rightsquigarrow$ 前と同じ計算で

$$\Pr_{r_2}(\tilde{f}_2(r_1, r_2) = f_2(r_1, r_2) \mid \tilde{f}_1(r_1) \neq f_1(r_1)) \leq \frac{1}{|\varphi|^2}$$

まとめると，No インスタンスに対して，

$$\begin{aligned}
&\Pr_{r_1}(\tilde{f}_1(r_1) = f_1(r_1)) && \leq \frac{1}{|\varphi|^2} \\
&\Pr_{r_2}(\tilde{f}_2(r_1, r_2) = f_2(r_1, r_2) \mid \tilde{f}_1(r_1) \neq f_1(r_1)) && \leq \frac{1}{|\varphi|^2} \\
&\qquad\vdots \\
&\Pr_{r_n}(\tilde{f}_n(r_1, \ldots, r_n) = f_n(r_1, \ldots, r_n) \\
&\qquad \mid \tilde{f}_{n-1}(r_1, \ldots, r_{n-1}) \neq f_{n-1}(r_1, \ldots, r_{n-1})) && \leq \frac{1}{|\varphi|^2}
\end{aligned}$$

したがって，

$$\Pr_{r_1, \ldots, r_n}(\text{No を出力}) \geq \left(1 - \frac{1}{|\varphi|^2}\right)^n \geq \frac{2}{3}$$

□

本日のまとめと次回の予告

本日のまとめ

- 複雑性クラス IP を導入する
- IP $\subseteq$ PSPACE を証明する
- PSPACE $\subseteq$ IP の証明の準備をする

次回の予告

- PSPACE $\subseteq$ IP の証明をする

クイズ

Q

1.

2.

3.

4.