マルチパーティ計算の理論

藤崎 英一郎

北陸先端科学技術大学院大学

2018 年 6 月 29 日 @ 金沢大学研究集会

もくじ

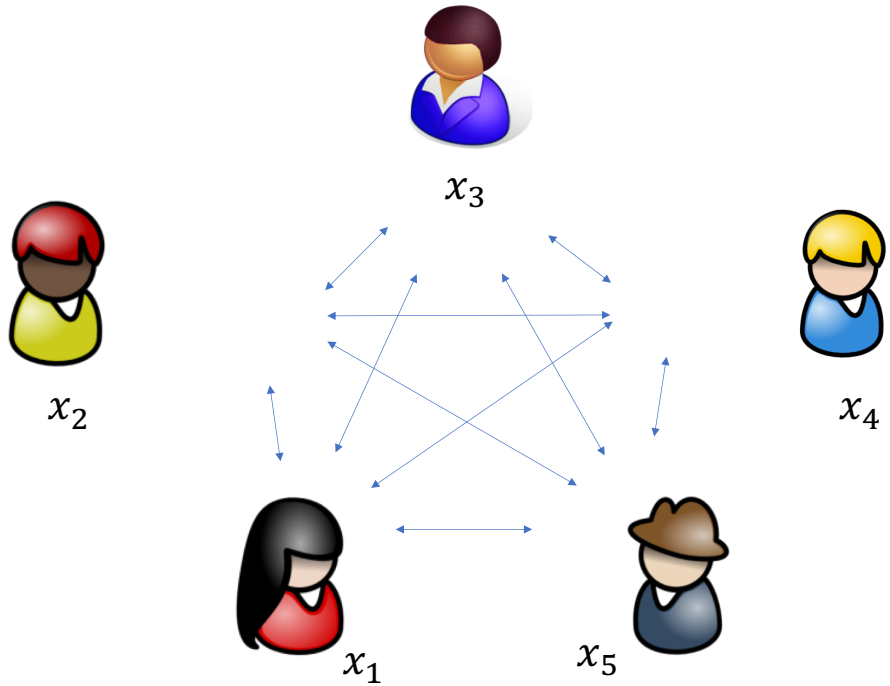
この発表について

暗号の予備知識なしで分かる（はずの）暗号のお話

- 理解するに必要な知識：学部程度の線型代数。
- ただし、時間が不足（前期の講義のダイジェスト版）＋ 話者の力不足

マルチパーティ計算

- 参加者: $P_1, \ldots, P_n$.
- 各 P_i への秘密の入力: $x_i \in \{0,1\}^\lambda$
- 全参加者への入力（公開情報）: 関数 $F : \{0,1\}^{n\lambda} \to \{0,1\}^*$.
- 各 P_i への出力：$F(x_1, \ldots, x_n)$. より一般的には、参加者ごとに違う出力をすることも許す.
- ネットワークモデル: synchronous network with peer-to-peer secure channel



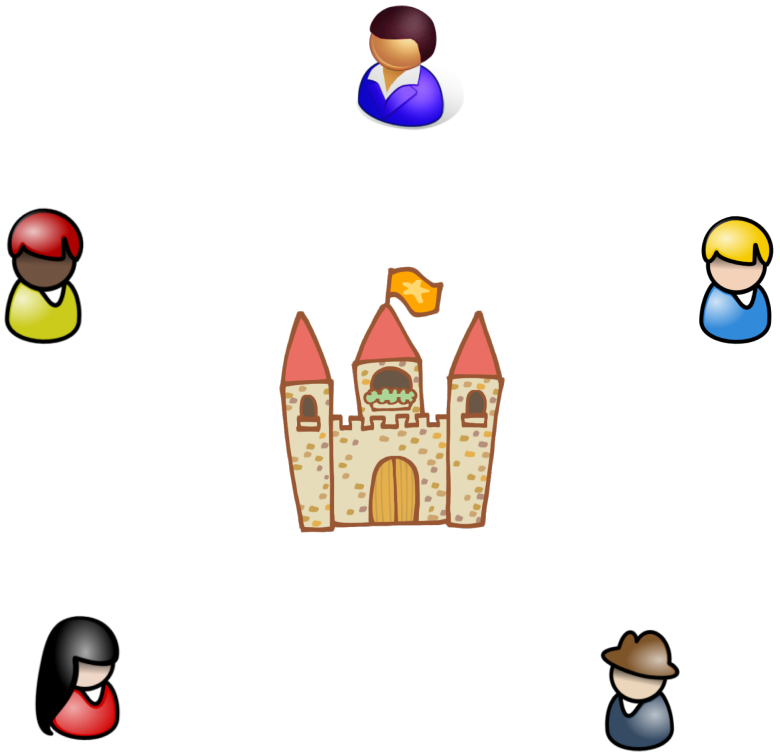
ネットワークモデル

- どの参加者同士も peer-to-peer で秘密通信可 (secure channel)
- 同期型ネットワーク (synchronous network)
  - 送られた情報は遅延なく、次のスロットで相手に必ず届く
  - cf. 非同期型ネットワーク (asynchronous network)
- 同報通信ができることを仮定する場合あり
  - 同報通信を仮定しない場合、Byzantine 合意を用いて同報通信を実現する（不正者の数が全体の参加者の 1/3 未満なら同期型ネットワークで実現できる）
  - 今回の話では同報通信を仮定しない。

Byzantine 合意問題

Byzantine 合意問題解法あり $\Longleftrightarrow$ peer-to-peer secure channel での Broadcast の問題の解法あり

定理 (Lamport): 同期型ネットワークなら、裏切り者の数が全体の 1/3 未満ならエラーなしで Byzantine 合意問題に解法あり.



将軍A~Eが、Byzantine 帝国の首都コンスタンチノープルを囲んでいる。

将軍達は、全員で、「攻撃」(attack) または「撤退」(retreat) ができるように各将軍に伝令を送る。

多数決で決まった方に全員一致で行動したい。

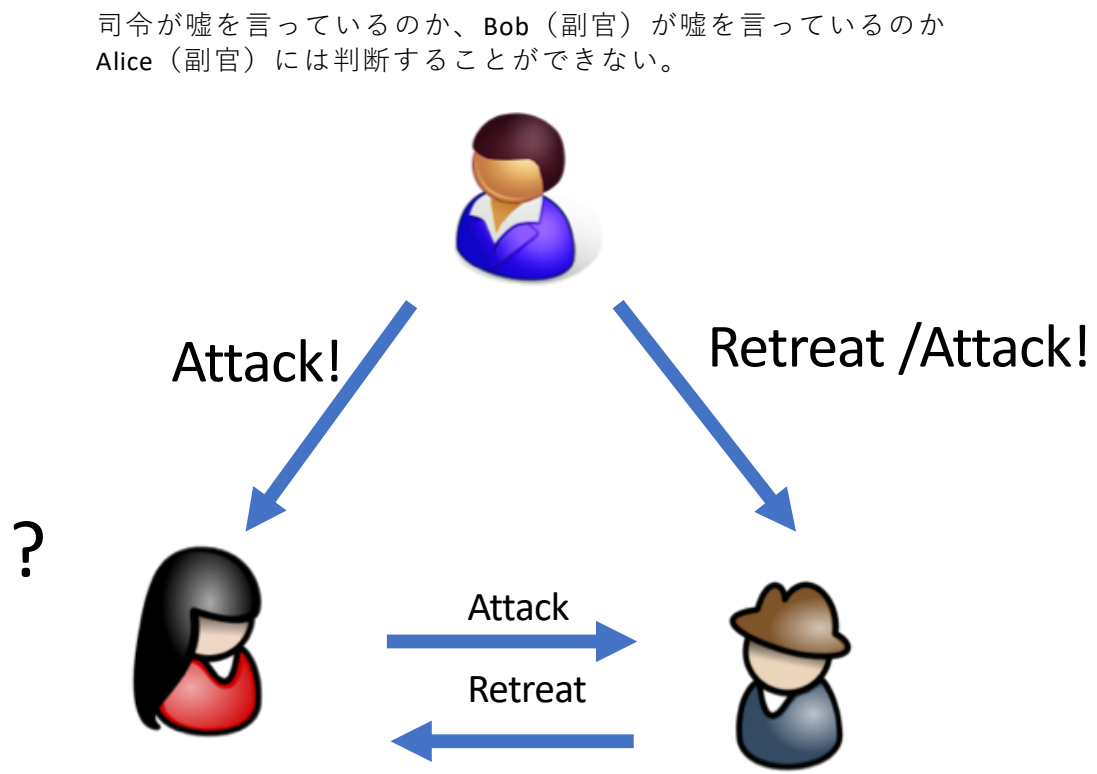
ただし、裏切り者がこの中にいる。

自分の情報を全員に正しく伝えるにどうしたら良いか？

司令官と副官の問題

Byzantine 合意問題は、司令官と副官の問題を解くことと同じになる。

司令と副官の中に 1/3 以上の裏切り者がいた場合、（エラーのない）解法は存在しない。
以下、$n = 3$ の例。

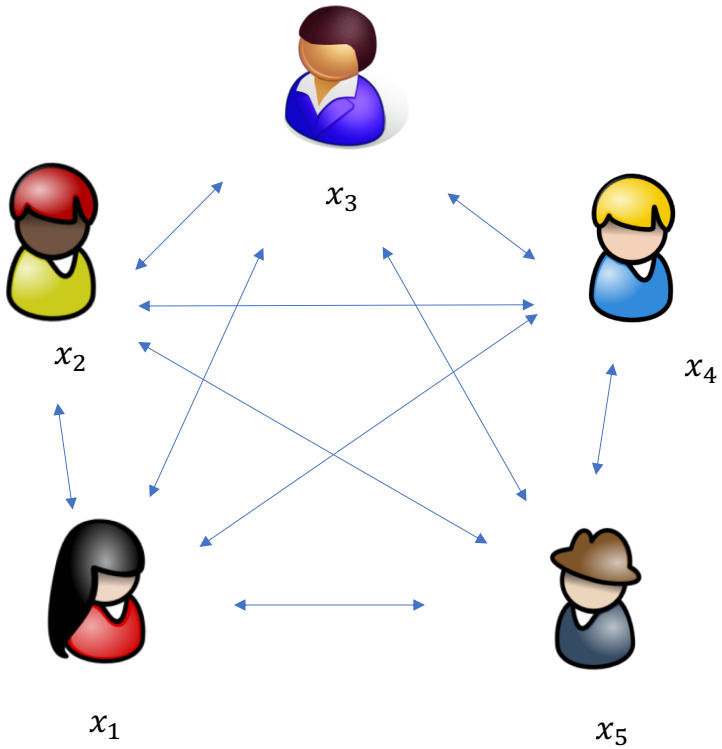


どのようにMPCの安全性を考えるか

各参加者が信頼できる第三者に自分の秘密を渡し、計算結果だけをもらえるのを理想のプロトコル。現実のプロトコルが理想と（ほぼ）同じになれば安全という。

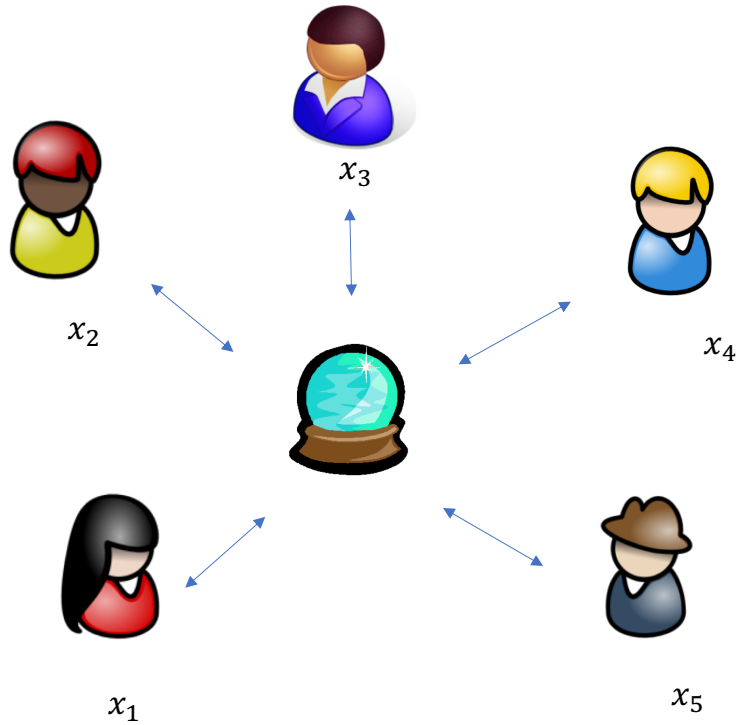
現実のプロトコル

互いに通信し計算結果を得る



理想状態

各自の秘密を  に預けると計算結果を返してくれる



不正者（攻撃者）に関する決め事

（多くの場合は）不正者の数を制限（この話でも）。不正者同士は常に結託して情報を共有できる。さらに、不正者のタイプを次のように分類する。

- Passive Adversary (aka semi-honest or honest-but-curious).
  - 受動的攻撃者
  - 正規のプロトコルからは逸脱しないが、プロトコルから得た情報から正直な参加者の秘密情報を得ようとする
- Active Adversary
  - 能動的攻撃者
  - プロトコルから逸脱しても良い。正直な参加者の秘密情報を得ようとすることと、プロトコルを失敗に終わらせようとする。

より細かい話はのちの安全性モデルのところで

知られている結果

不正者数に応じた結果

	Passive	Active w/ broadcast	Active w/o broadcast
情報理論的安全	$< \frac{n}{2}$	$< \frac{n}{3}$	$< \frac{n}{3}$
統計的安全	$< \frac{n}{2}$	$< \frac{n}{2}$	$< \frac{n}{3}$
計算量的安全	n	$< \frac{n}{2}$	$< \frac{n}{2}$

ただし、$n > 2$. 結果は全て optimal

- 情報理論的安全 = 現実と理想の差が全くない
- 統計的安全: 現実と理想の差がわずかな統計的差しかない。
- 計算量的安全: 多項式時間制限のアルゴリズムでは、識別できないぐらいしか、現実と理想の分布に違いはない。

ここで話す MPC

BGW [BGW88] / CCD [CCD88] 型の秘密計算方式.

- 情報理論的安全な三者以上の秘密計算プロトコル (MPC).
  - 不正者数 $< n/2$ としたときの Passive 安全な MPC
    [BGW88] に Michael Rabin の改良提案を組み合わせたもの
  - 不正者数 $< n/3$ としたときの Active 安全な MPC
    [BGW88] に [CDM00] のアイデアなどを混ぜ合わせ [CDN15] で解説されたもの
- 主要テクニック
  - 線型秘密分散 (Linear Secret Sharing (LSS))
  - 検証可能線型秘密分散 (Verifiable Linear Secret Sharing (VLSS))
  - (V)LSS での掛け算
    - 加減算は（線形性から）容易なので、乗算をするテクニックが重要。
    - 加減算、乗算ができると論理回路を計算できるので、任意の（多項式時間計算可能な）関数の MPC ができるようになる。

参考文献

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson.
Completeness theorems for non-cyptographic fault-tolerant distributed computation.
In Simon [Sim88], pages 1–10.

[CCD88] D. Chaum, C. Crépeau, and I. Damgård.
Multiparty unconditionally secure protocols.
In Simon [Sim88], pages 11–19.

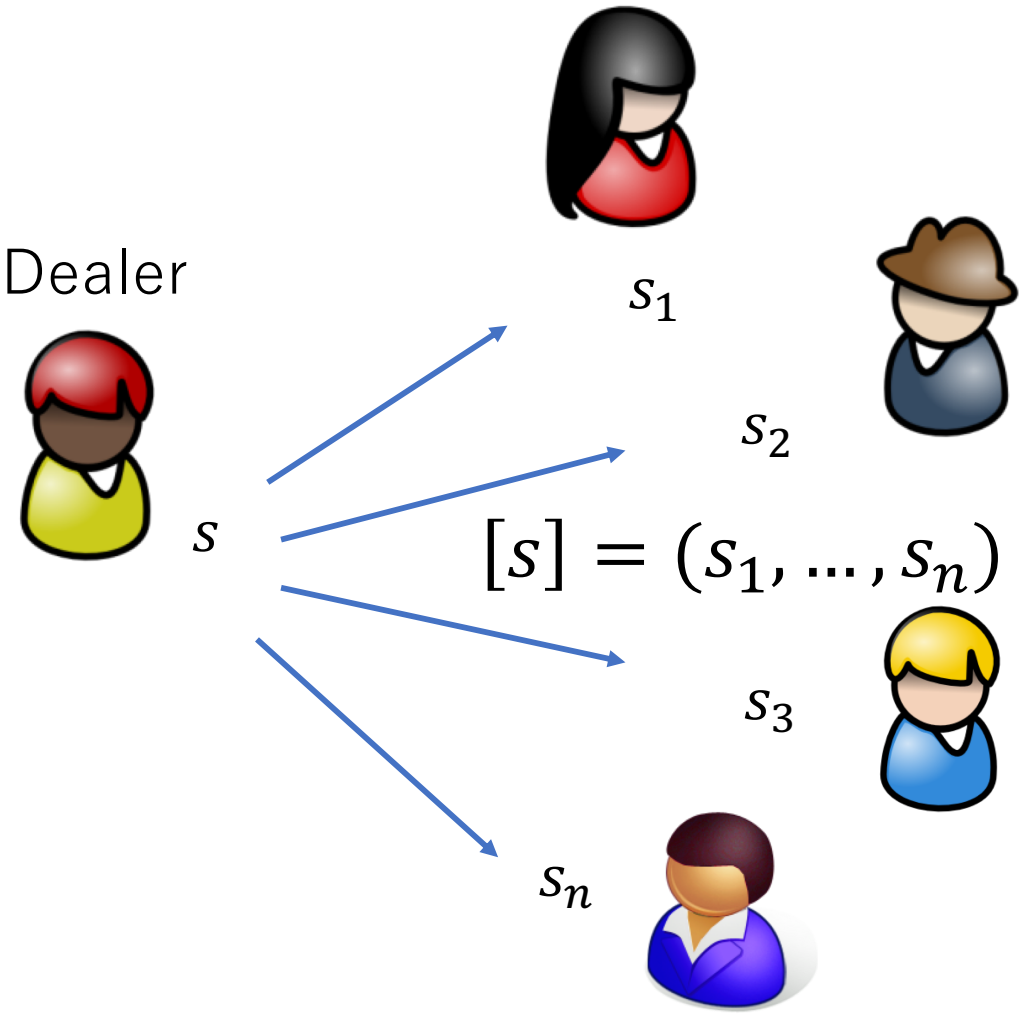
[CDM00] Ronald Cramer, Ivan Damgård, and Ueli M. Maurer.
General secure multi-party computation from any linear secret-sharing scheme.
In Bart Preneel, editor, EUROCRYPT 2000, volume 1807 of Lecture Notes in Computer Science, pages 316–334. Springer, Heidelberg, 2000.

[CDN15] Ronald Cramer, Ivan Damgård, and Jesper Nielsen.
Secure Multiparty Computation and Secret Sharing.
Cambridge University Press, 2015.

もくじ

秘密分散

情報理論的安全な MPC での基本技術。秘密を分割して参加者間で保持。
t 人では秘密が全く漏れない。$t+1$ 人以上で秘密を復元できる。



$$[s] = (s_1, \ldots, s_n)$$

Dealer により、（n人の）参加者にシェアが分配された状態を示す

t+1人以上でsを復元できる

(t+1,n)-秘密分散

秘密分散 (Secret Sharing)

$(t+1, n)$-秘密分散 $\mathsf{SS} = (\mathsf{Share}, \mathsf{Recon}, \Gamma_{t+1,n})$ とは次のスペックを満たすもの。

Specification

- Share は秘密 $s \in K$（K は体）を入力に取り、n 個の share からなるベクトル $[s] := (s_1, \ldots, s_n)$ を出力する確率的アルゴリズム

$$\mathsf{Share} : s \in K \mapsto [s] := (s_1, \ldots, s_n) \in K^n$$

- $\Gamma_{t+1,n} \triangleq \{Q \subset \{1, \ldots, n\} \mid \#Q \geq t+1\}$（閾値型アクセス構造）
- Recon は、$[s]$ の t 個以上の要素を含む部分集合 $S_Q = \{s_i | i \in Q\}$（$Q \in \Gamma_{t+1,n}$）を入力に取り、秘密 s を復元する確定的アルゴリズム

上記 $\mathsf{SS} = (\mathsf{Share}, \mathsf{Recon}, \Gamma_{t+1,n})$ が秘密分散とは、次の条件を満たすものである。

Security

- (Perfect Reconstructability) 全ての $s \in K$, $[s] \leftarrow \mathsf{Share}(s)$, $Q \in \Gamma_{t+1,n}$ となる S_Q に対して、$\mathsf{Recon}(S_Q) = s$.
- (Perfect Privacy) 全ての $s \in K$, $[s] \leftarrow \mathsf{Share}(s)$, $T \notin \Gamma_{t+1,n}$ となる S_T に対して、$H(s) = H(s|S_T)$。すなわち、確率変数 s と S_T は独立。

線型秘密分散 (Linear Secret Sharing)

SS $=$ (Share, Recon, $\Gamma_{t+1,n}$) が線型秘密分散とは、次の条件を満たすものである。

Linearlity

- SS が秘密分散かつ、
- (Linearlity) 全ての $a, b, \lambda, \rho \in K$, $[a] \leftarrow \mathrm{Share}(a)$, $[b] \leftarrow \mathrm{Share}(b)$, $[\lambda a + \rho b] \leftarrow \mathrm{Share}(\lambda a + \rho b)$ に対して、

$$[\lambda a + \rho b] = \lambda[a] + \rho[b]$$

が成り立つ。ここで、$\lambda[a] := (\lambda a_1, \ldots, \lambda a_n)$, $[a] + [b] := (a_1 + b_1, \ldots, a_n + b_n)$ と定義。

すなわち、$P_1, \ldots, P_n$ 間で秘密 a, b をシェア状態 $[a]$,$[b]$ であるとき、各 P_i がローカルに手元で $\lambda a_i + \rho b_i$ を計算して（λ, ρ は既知）新たにシェアされる $(\lambda a_1 + \rho b_1, \ldots, \lambda a_n + \rho b_n)$ が秘密 $\lambda a + \rho b$ をシェアした状態 $[\lambda a + \rho b]$ になることを意味する。

（蛇足）線形秘密分散

Reconstruction で $\sim$ を導入してみるとわかりやすいかも

$$\mathrm{Share} : K \longrightarrow K^n/\sim$$

$$\cup$$

$$a \longmapsto [a] = (a_1, \cdots, a_n)$$

$$(a_1, \cdots a_n) \sim (a_1', \cdots a_n')$$

$$\Leftrightarrow [a] = [a']$$

Reconstruction で同じ値になる

linear

$$[a+b] = [a] + [b]$$

$$[\lambda a] = \lambda [a]$$

自明な (n, n)-線型秘密分散

K を有限体。秘密 $s \in K$.

自明な (n, n)-線型秘密分散

- Share(s):
  - 独立で一様ランダムに $s_1, \ldots, s_{n-1}$ を K から選ぶ。すなわち、$s_1, \ldots, s_{n-1} \ldots_R K$.
  - $s_n = s - \sum_{i=1}^{n-1} s_i \in K$ を計算。
  - $[s] = (s_1, \ldots, s_n)$ を出力する。
- Recon$(s_1, \ldots, s_n)$: $s = \sum_{i=1}^{n} s_i \in K$ を出力。

- (Perfect Reconstruction) 自明。
- (Perfect Privacy) どの $(n-1)$ 個のシェア $(s_{i_1}, \ldots, s_{i_{n-1}})$ をとっても、s に無関係で独立一様ランダムであるため、s と $(s_{i_1}, \ldots, s_{i_{n-1}})$ は独立。よって、Perfect Privacy を満たす。
- (Linearlity) 明らかに $[\lambda s + \rho s'] = \lambda[s] + \rho[s']$ を満たす。

Shamir 秘密分散

Shamir 秘密分散 SS $= (\mathsf{Share}, \mathsf{Recon}, \Gamma_{t+1,n})$. $\alpha_1, \ldots, \alpha_n \in K$ はシステムで予め定められた異なる値。

Shamir SS

- Share(s):
  - $a_0 := s$ の条件のもと K 係数のランダムな t 次多項式 $f(X) = \sum_{i=0}^{t} a_i X^i$ を選ぶ。

$$f(X) \leftarrow_R K[X] \quad \text{such that} \quad \deg(f) = t \text{ and } a_0 = s.$$

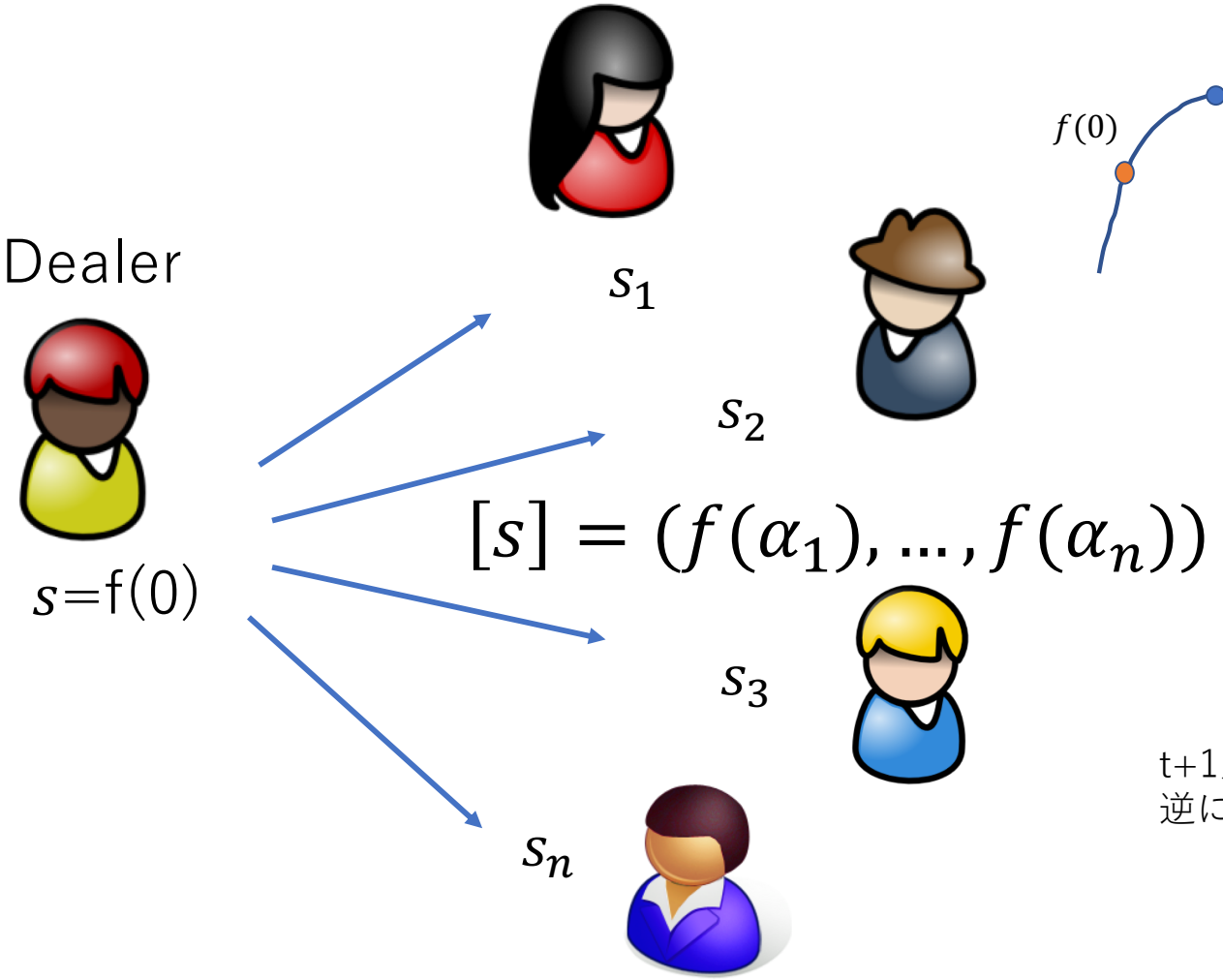
  - $[s] = (f(\alpha_1), \ldots, f(\alpha_n))$ を出力する。
- Recon(S_Q) ($S_Q = \{f(\alpha_i) | i \in Q \text{ s.t. } Q \in \Gamma_{t+1,n}\}$): s を出力。

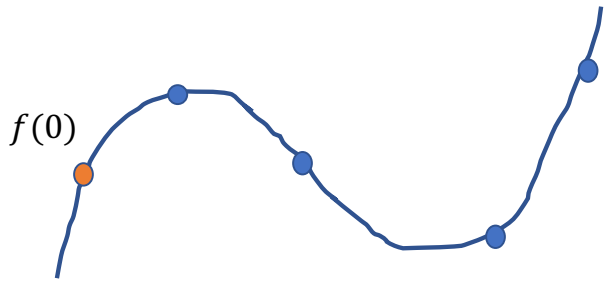
$$s = \sum_{i \in Q} \lambda_{i,Q} f(\alpha_i) \text{ where } \lambda_{i,Q} = \prod_{j \in Q \setminus \{i\}} \left(\frac{\alpha_j}{\alpha_j - \alpha_i} \right).$$

Reconstruction vector $(\lambda_{1,Q}, \ldots, \lambda_{\#Q,Q})$ は、S_Q のみで決定することに注意

Shamir 秘密分散 (2)

Dealer は、多項式 f を $s = f(0)$ かつ $\deg(f) = t$ となる条件でランダムに選ぶ。





t次曲線は、t+1点で決定
逆に、t点以下なら決定しない

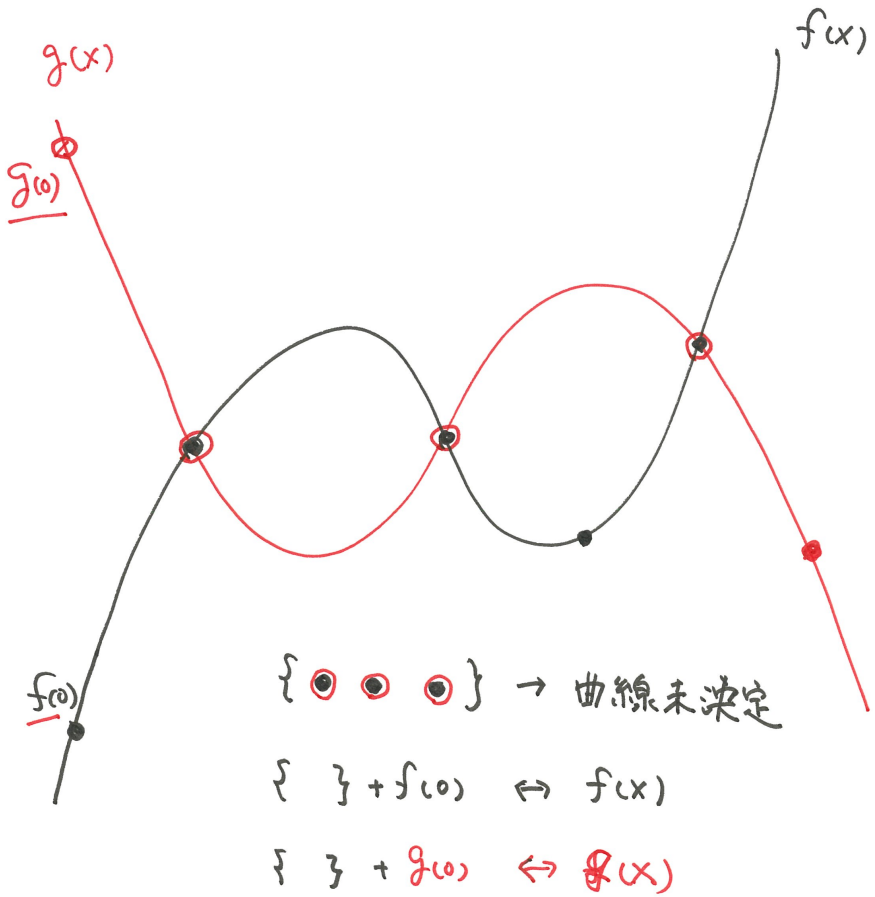
t+1人以上でs=f(0)を復元できる
逆に、t人では、sの情報漏れない

(t+1,n)-秘密分散

多項式の決定から

$f(X)$ の決定 $\Longleftrightarrow$ 曲線の $\deg(f)+1$ 点の決定

- （$f(0)$ 以外の）$\deg(f)$ 点が漏れる　$\Longrightarrow$ 秘密 $f(0)$ の情報全くなし
- $\deg(f)+1$ 点を公開　$\Longrightarrow$ $f(X)$ が決定するので $f(0)$ が決定



Perfect Privacy

以下の $(f(\alpha_1), \ldots, f(\alpha_n))$ は完全同分布.

- Original: D は $s := a_0$ とし、$a_1, \ldots, a_t$ をランダムに決定（$f(X)$ が決定）。D は、$f(\alpha_1), \ldots, f(\alpha_n)$ を $P_1, \ldots, P_n$ にそれぞれ配る。
- $\mathrm{Dist}_{(\alpha_{i_1}, \ldots, \alpha_{i_t})}$: D は $s := f(0)$ とし、$f(\alpha_{i_1}), \ldots, f(\alpha_{i_t})$ をランダムに決定（$f(X)$ が決定）. D は、$f(\alpha_1), \ldots, f(\alpha_n)$ を $P_1, \ldots, P_n$ にそれぞれ配る。

$\{\alpha_{i_1}, \ldots, \alpha_{i_t}\}$ はどの組み合わせであっても Original の share の配り方と同分布。

$(a_0, a_1, \ldots, a_t) \Leftrightarrow (f(0), f(\alpha_{i_1}), \ldots, f(\alpha_{i_t}))$

$f(X) = a_0 + a_1 X + \cdots + a_t X^t$. Vandermonde 行列より正則なので一対一.

$$\begin{pmatrix} f(0) \\ f(\alpha_{i_1}) \\ \vdots \\ f(\alpha_{i_t}) \end{pmatrix} = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ 1 & \alpha_{i_1}^1 & \ldots & \alpha_{i_1}^t \\ \ldots & \ldots & \ldots & \ldots \\ 1 & \alpha_{i_t}^1 & \ldots & \alpha_{i_t}^t \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_t \end{pmatrix}$$

Perfect Privacy: $S_T = \{f(\alpha_{i_1}), \ldots, f(\alpha_{i_t})\}$ は、$s = f(0)$ と無関係かつ独立に選ばれているので、これらの値が漏れても s の情報を含まない。

Perfect Reconstruction

Lagrange 補間公式から Reconstruction algorithm を構成。
$f(X)$ を次数 $\deg(f) = t$ の多項式とする。任意の $n(\geq t+1)$ の異なる点 $\alpha_1, \ldots, \alpha_n$ の関数値を $f(\alpha_1), \ldots, f(\alpha_n)$ とすると、$f(X)$ は

$$f(X) = \sum_{i=1}^{n} \lambda_{i,n}(X) \cdot f(\alpha_i) \text{ where } \lambda_{i,n}(X) = \prod_{j=1, j\neq i}^{n} \left(\frac{X - \alpha_j}{\alpha_i - \alpha_j} \right)$$

と、$n(\geq t+1)$ の個数と $\alpha_1, \ldots, \alpha_n$ の選び方によらず一意に復元される。

$$\lambda_i(\alpha_j) = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j. \end{cases}$$

かつ、

$$s = f(0) = \sum_{i=1}^{n} \lambda_{i,n}(0) f(\alpha_i)$$

に注意。$(\lambda_{1,n}, \ldots, \lambda_{n,n}) := (\lambda_{1,n}(0), \ldots, \lambda_{n,n}(0))$ を、Shamir SS の $\alpha_1, \ldots, \alpha_n$ での reconstruction vector と呼ぶ.

多項式補間

Theorem 1

K を体とする。任意の $\{(\alpha_i, y_i)\}_{i=1}^{n}$ $(\alpha_i, y_i \in K)$ に対して（ただし $\alpha_i \neq \alpha_j$）、

- $y_i = F(\alpha_i)$ *for* $i = 1, \ldots, n$
- $\deg(F(X)) \leq n - 1$

となる多項式関数 $Y = F(X)$ が一意に定まる。

この定理（と証明）から

- n 点から補間される Lagrange 多項式 $F(X)$ の次数は $n-1$ 以下（見せかけの次数は $n-1$）。
- $\deg(f(X)) = t < n - 1$ とする。$\{(\alpha_i, f(\alpha_i)\}_{i=1}^{n}$ から補間された Lagrange 多項式を $F(X)$ とすると、$F(X) \equiv f(X)$（Reconstruction の一意性）。

Theorem 1 の証明

$Q = \{1, \ldots, n\}$ とする。全ての x_i $(i \in Q)$ に対して、$F(\alpha_i) = f(\alpha_i)$ となるような多項式を一つ考える。$F(X) = \sum_i \lambda_{i,Q}(X) \cdot f(\alpha_i)$ とおくと

$$\lambda_{i,Q}(\alpha_j) = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j. \end{cases}$$

なら条件を満たす。そのような $\lambda_{i,Q}(X)$ は、

$$\begin{aligned} \lambda_{i,Q}(X) &= \frac{\prod_{j \in Q \setminus \{i\}} (X - \alpha_j)}{\prod_{j \in Q \setminus \{i\}} (\alpha_i - \alpha_j)} \\ &= \frac{(X - \alpha_1) \ldots (X - \alpha_{i-1}) \cdot \cancel{(X - \alpha_i)} \cdot (X - \alpha_{i+1}) \ldots (X - \alpha_n)}{(\alpha_i - \alpha_1) \ldots (\alpha_i - \alpha_{i-1}) \cdot \cancel{(\alpha_i - \alpha_i)} \cdot (\alpha_i - \alpha_{i+1}) \ldots (\alpha_i - \alpha_n)} \end{aligned}$$

が条件を満たす。さらに、$\deg(\lambda_{i,Q}) \leq n - 1$ より、$\deg(F) \leq n - 1$.

Theorem 1 の証明（続き）

$F(X)$ 以外にも条件を満たす t 次（以下の）多項式 $G(X)$ が存在すると仮定。すなわち

- $y_i = F(\alpha_i) = G(\alpha_i)$ for $i = 1, \ldots, n$
- $\deg(F(X)), \deg(G(X)) \leq n - 1$

今、$H(X) \triangleq F(X) - G(X)$ を考えると、全て $i \in Q$ に対して、$H(\alpha_i) = 0$. H の次数は高々 $n - 1$ であるから、n 個の α_i を根に持つためには、$H(X) \equiv 0$ が必要。よって、$F(X) \equiv G(X)$ であり、このような多項式は、$n - 1$ 次以下では一意に決定する。

Shamir 秘密分散は線型

$f, g \in K[X]$ $(\deg(f), \deg(g) \leq t)$ に対して次のことは明らか。

線型性

$$[\alpha f(0) + \beta g(0)] = \alpha[f(0)] + \beta[g(0)] \qquad \text{where } \alpha, \beta \in K.$$

$$f(X) = a_0 + a_1 X + \ldots a_t X^t \text{ および } g(X) = b_0 + b_1 X + \ldots b_t X^t$$

とする。$h(X) \triangleq \alpha f(X) + \beta g(X)$ と定義すると、$\deg(h) \leq t$ で、

$$\begin{aligned}
[h(0)] &= (h(\alpha_1), \ldots, h(\alpha_n)) \\
&= (\alpha f(\alpha_1) + \beta g(\alpha_1), \ldots, \alpha f(\alpha_n) + \beta g(\alpha_n)) \\
&= \alpha[f(0)] + \beta[g(0)]
\end{aligned}$$

Shamir SS の足し算

線形性から簡単に計算できる。

addition

$[a]$, $[b]$: $a, b \in K$ が $P_1, \ldots, P_n$ 間でシェアされた状態

$$[a] + [b] = [a + b]$$

f, g をそれぞれ a, b をシェアする時の t 次多項式とする。$f_0 = a$, $g_0 = b$ で、

$$f(X) = f_0 + f_1 X + \ldots + f_t X^t, \text{ and } g(X) = g_0 + g_1 X + \ldots + g_t X^t.$$

$h(X) = f(X) + g(X)$ と置くと、$h(0) = a + b$ と $\deg(h) = t$ より

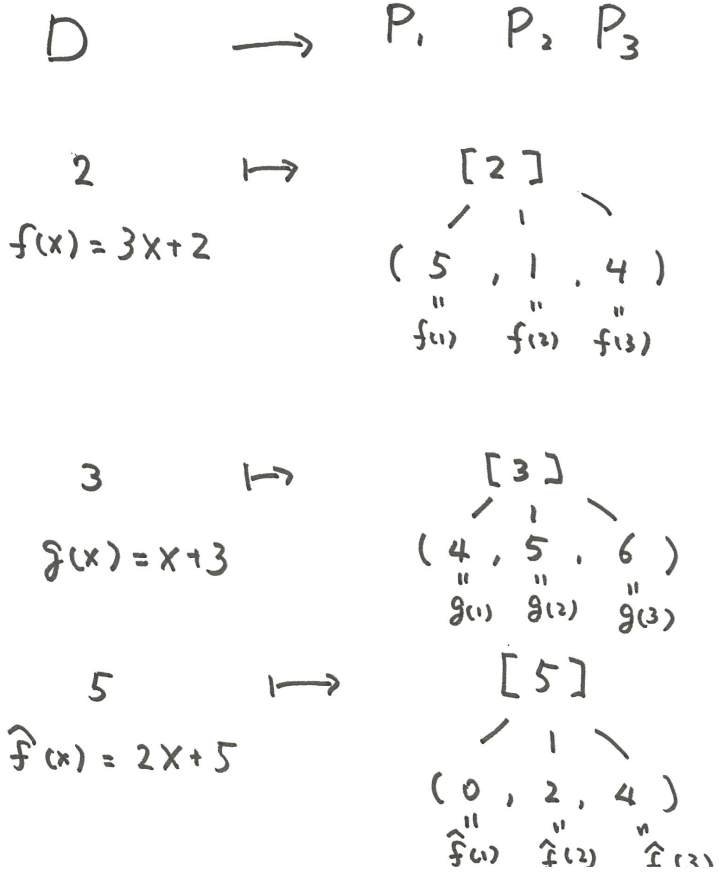
$$[a + b] = (h(\alpha_1), \ldots, h(\alpha_n))$$

$h(\alpha_i) = f(\alpha_i) + g(\alpha_i)$ であるから、各 P_i がローカルに自分のシェアを足し算すれば、$[a + b]$ と言う状態になる。

線型性

例) Shamir SS での線型性の確認

$K = \mathbb{Z}/7\mathbb{Z}$



[2]+[3]

def (5+4, 1+5, 4+6)

(f(1)+g(1), f(2)+g(2), f(3)+g(3))

2 6 3

11 ← 秘密を復元した時に同じという意味

[5]

= (0, 2, 4)

Shamir SS の掛け算（試み）

掛け算は少し工夫がいる。

multiplication

$[a]_{(t)}$, $[b]_{(t)}$: $a, b \in K$ が $P_1, \ldots, P_n$ 間で $(t+1, n)$-SS でシェアされた状態.

$$[a]_{(t)} \cdot [b]_{(t)} = [ab]_{(2t)}$$

が成り立つ。ただし、$[a] \cdot [b] := (a_1 b_1, \ldots, a_n b_n)$ と定義（各 P_i がローカルに自分のシェアを掛け合わせた状態）。

f, g をそれぞれ a, b をシェアする時の t 次多項式とする。$f_0 = a$, $g_0 = b$ で、

$$f(X) = f_0 + f_1 X + \ldots + f_t X^t, \text{ and } g(X) = g_0 + g_1 X + \ldots + g_t X^t.$$

$h(X) = f(X)g(X)$ と置くと、$h(0) = ab$ と $\deg(h) = 2t$ より

$$[ab]_{(2t)} = (f(\alpha_1)g(\alpha_1), \ldots, f(\alpha_n)g(\alpha_n))$$

よって、$P_1, \ldots, P_n$ 間で ab をシェアした状態になるが、$2t+1$ 個のシェアが集まらないと ab が復元できない。

Shamir SS の掛け算（アイデア）

multiplication

Lagrange と線型性により

$$[ab] = [h(0)] = [\sum_{i=1}^{n} \lambda_{i,n} h(\alpha_i)] = \sum_{i=1}^{n} \lambda_{i,n} [h(\alpha_i)]$$

$(\lambda_{1,n}, \ldots, \lambda_{n,n})$ は $\{\alpha_1, \ldots, \alpha_n\}$ のみから決まる。

$h(X) = f(X)g(X)$ であるから、$h(\alpha_i) = f(\alpha_i)g(\alpha_i)$. よって、$[a]$, $[b]$ の状態から、$[ab]$ を作るには各 P_i がローカルに $a_i b_i = f(\alpha_i)g(\alpha_i)$ を計算した後、その値のシェアを他の参加者に $(t+1, n)$-線型秘密分散で配り $[f(\alpha_i)g(\alpha_i)]$ という状態を作り出せば良い。後は線型性からローカルな計算で $[ab]$ の状態に持っていく。

Shamir SS の掛け算（まとめ）

- Input: $[a], [b]$: a, b がそれぞれ $P_1, \ldots, , P_n$ 間でシェアされている
- Output: $[ab]$: ab が $P_1, \ldots, , P_n$ 間でシェアされる

1. 各 P_i が $c_i = a_i b_i$ をローカルに計算
2. 各 P_i が c_i を $P_1, \ldots, P_n$ 間で線型秘密分散。$[c_1], \ldots, [c_n]$ の状態になる。P_i は、$c_{1,i}, \ldots, c_{i,i}, \ldots, c_{n,i}$ を、$[c_1], \ldots, [c_n]$ の自分のシェアとして持つ。
3. $i = 1, \ldots, n$ に対して、$\lambda_{i,n} = \prod_{j \neq i} \frac{\alpha_i}{\alpha_j - \alpha_i}$ を計算する。
4. 各 P_i が $d_i = \sum_{j=1} \lambda_{j,n} c_{j,i}$ を計算する。
5. $[ab] = (d_1, \ldots, d_n)$ なので、$[ab]$ の状態になる。

補足

$[a]$, $[b]$ の状態から

掛算

$[a] = (a_1, a_2, a_3)$ 　　 $[b] = (b_1, b_2, b_3)$

P_1: $[a_1 b_1] = (c_{11}, c_{12}, c_{13})$

P_2: $[a_2 b_2] = (c_{21}, c_{22}, c_{23})$

P_3: $[a_3 b_3] = (c_{31}, c_{32}, c_{33})$

各 P_i は $a_i b_i$ を SSする

$$[ab] = \lambda_{13}[a_1 b_1] + \lambda_{23}[a_2 b_2] + \lambda_{33}[a_3 b_3]$$

$$\left(-, \sum_{i=1}^{3} \lambda_{i3} c_{i2}, -\right) \quad (-, \lambda_{13} c_{12}, -) \quad (\lambda_{23} c_{21}, \lambda_{23} c_{22}, \lambda_{23} c_{23}) \quad (-, \lambda_{33} c_{32}, -)$$

各 P_j は $\sum_{i=1}^{3} \lambda_{i3} c_{ij}$ を保持 $\Leftrightarrow$ $[ab]$

Shamir SS による MPC の線型和と積

初期状態 $[a] = (a_1, \ldots, a_n)$, $[b] = (b_1, \ldots, b_n)$: すなわち $a, b \in K$ が $P_1, \ldots, P_n$ 間でシェアされた状態.

線形和と積

$$[a] + [b] = [a + b], \qquad -[a] = [-a]$$

線型性から、各 P_i がローカルに $a_i + b_i$ を計算すれば、$[a + b]$ という状態になる。また各 P_i が $-a_i$ を計算すれば $[-a]$ になる。

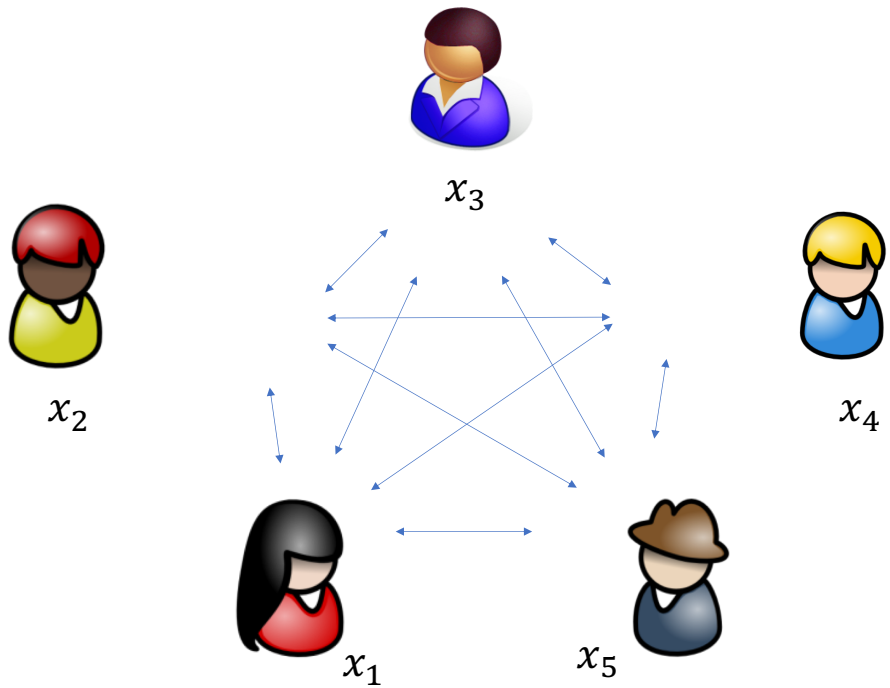
$$[ab] = [\sum_{i=1}^{n} \lambda_{i,n} a_i b_i] = \sum_{i=1}^{n} \lambda_{i,n} [a_i b_i].$$

$(\lambda_{1,n}, \ldots, \lambda_{n,n})$ は $\{\alpha_1, \ldots, \alpha_n\}$ のみから決まる。各 P_i がローカルに $a_i b_i$ を計算後、その値を $(t+1, n)$-線型秘密分散で $[a_i b_i]$ を行う。後は線型性からローカルな計算で $[ab]$ という状態にできる.

もくじ

マルチパーティ計算

- 参加者: $P_1, \ldots, P_n$.
- 各 P_i への秘密の入力: $x_i \in \{0,1\}^\lambda$
- 全参加者への入力（公開情報）: 関数 $F : \{0,1\}^{n\lambda} \to \{0,1\}^*$.
- 各 P_i への出力：$F(x_1, \ldots, x_n)$. より一般的には、参加者ごとに違う出力をすることも許す.
- ネットワーク: Pair-wise private & synchronized.



Secure MPC against Passive Adversaries

- 参加者: $P_1, \ldots, P_n$.
- 各 P_i への秘密の入力: $x_i \in \{0,1\}^\lambda$
- 全参加者への入力: 関数 $F : \{0,1\}^{n\lambda} \to \{0,1\}^*$.
- 各 P_i への出力：$F(x_1, \ldots, x_n)$.
- パラメータ：t, n $(t < n/2)$
- 不正者：passive, $\mathcal{A}_{t,n}(\triangleq \{A \subset \{1, \ldots, n\} \mid \#A \leq t\})$.

Theorem 2

There is an efficient MPC protocol to evaluate any efficiently computable function F such that the following conditions hold:

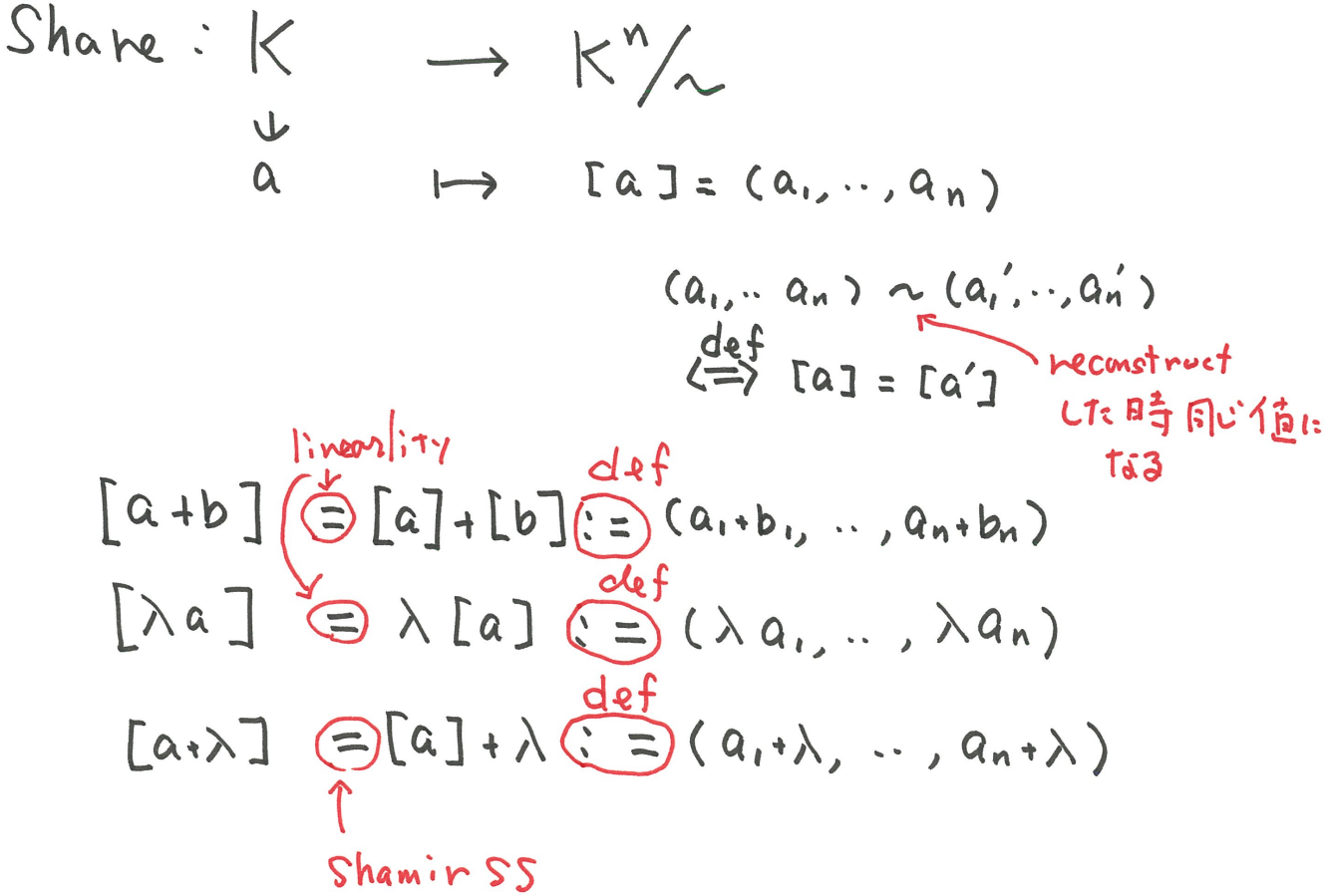
- *(Perfect Correctness) All players receive $F(x_1, \ldots, x_n)$ with prob. 1.*
- *(Perfect Privacy) Any passive $\mathcal{A}_{t,n}$-adversary with $t < n/2$ learns no information beyond $\{x_i\}_{i \in \mathcal{A}_{t,n}}$ and $F(x_1, \ldots, x_n)$ from executing the protocol regardless of their computing power and memory.*

MPC の構成（準備）

準備

- $F : \{0,1\}^{n\lambda} \to \{0,1\}^*$: 多項式時間計算可能関数
- C_F: F を計算する多項式サイズの AND, OR, NOT で構成された論理回路
- K: $n < \#K$ な有限体
- C_F の論理演算子への入力 $b \in \{0,1\}$ を $b \in \{0,1\} \subset K$ と K の元とみなす。
- AND, OR, NOT の論理ゲートを K 上の演算に置き換える。
  - $b \wedge b' \iff b \cdot b' \in K$.
  - $b \vee b' \iff b + b - b \cdot b' \in K$.
  - $\neg b \iff 1 - b \in K$.

線形性と ShamirSS の性質があると

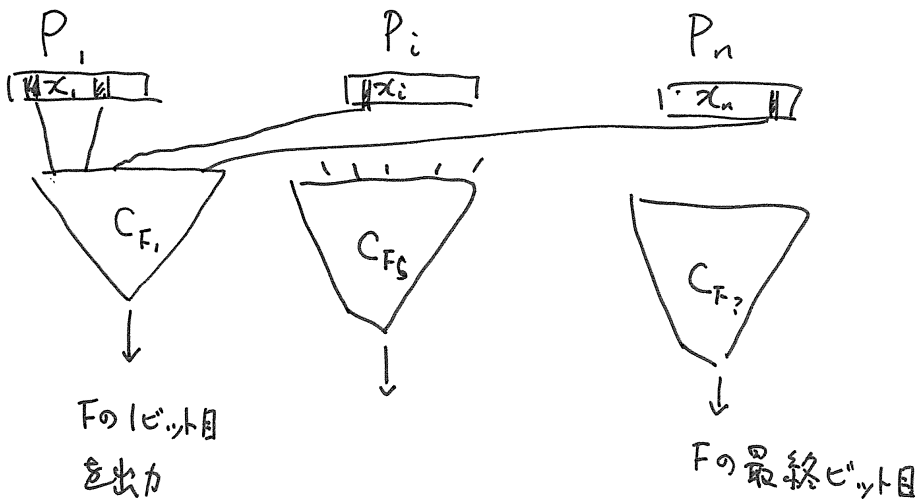


関数 F の回路

$F : \{0,1\}^{\lambda} \times \cdots \times \{0,1\}^{\lambda} \to \{0,1\}^{*}$ ← 任意の有限ビット列の集合

(n)

回路に分解



MPC の構成

- Input sharing phase: 各参加者 P_i は、x_i の各ビット b を $(t+1, n)$-Shamir SS を使い $[b]$ の状態にする。この結果、全ての参加者の秘密の全てのビットが（線型）秘密分散される。
- Computation phase: 各論理ゲートを秘密分散した形で実行
  - $[a] + [b] = [a + b]$
  - $1 - [a] = [1 - a]$
  - $[ab] = \sum_{i=1}^{n} \lambda_{i,n} [a_i b_i]$
- Output reconstruction phase: 出力ゲート結果が秘密分散された状態になっているので、各 P_i は自分の出力ゲート結果に関するシェアを他の参加者に公開（全員に送る）。各 P_i は、出力結果を復元する。

安全性

- Perfect Correctness: Passive adversary より、自明。
- Perfect Privacy: 不正者（達）は、次の二種類の情報*を正直な参加者から受け取る。
  - Type1: Input sharing phase と multiplication 計算時の、正直な参加者が持つ秘密 $[b]$ のシェア（$\{b_i\}_{i\in\mathcal{A}_{t,n}}$）
  - Type2: Output reconstruction phase での出力 y の全てのシェア。

プライバシーが保たれる直感的な説明。

- Type1 での情報 $\{b_i\}_{i\in\mathcal{A}_{t,n}}$ は、実際の秘密 b と独立。よってなにも秘密は漏れていない。
- Type2 の場合：ある t 次元多項式 f により、$y = f(0)$ となっており、不正者達は $\{f(\alpha_i)\}_{i\in\mathcal{A}_{t,n}}$ を共有している。今不正者の数がちょうど t とする。すると $\{f(\alpha_i)\}_{i\in\mathcal{A}_{t,n}}$ と $f(0)$ から、多項式 $f(X)$ が復元できるので、正直な参加者から送られてきたシェアは、「余剰」情報にはならない**。

*: 線形和の時は不正者は新たな情報を正直な参加者からもらってないので、なにも情報が増えないことに注意。

**: 不正者数が t 未満のときはどうなるか自分で考えよ。

$t \geq n/2$ の場合

$t < n/2$ という制限は optimal である。実際、$t \geq n/2$ の場合、実現できない関数が存在する。

Theorem 3

If $t \geq n/2$, there is a function that cannot be securely evaluated, even if the adversary is passive. Here, a function securely evaluated means that it satisfies perfect correctness and perfect privacy at the same time.

- *(Perfect Correctness) All players receive $F(x_1, \ldots, x_n)$ with prob. 1.*
- *(Perfect Privacy) Any passive $\mathcal{A}_{t,n}$-adversary with $t < n/2$ learns no information beyond $\{x_i\}_{i \in \mathcal{A}_{t,n}}$ and $F(x_1, \ldots, x_n)$ from executing the protocol regardless of their computing power and memory.*

実現できない関数

関数として、AND（または OR）は実現できない。 特別なケースとして $n = 2$, $t = 1$ の時を考える。

- P_1 の秘密を $b_1 \in \{0, 1\}$, P_2 の秘密を $b_2 \in \{0, 1\}$ とする。
- $b_1 = 0$ のとき、b_2 がなんであっても、計算結果は $0 = b_1 \wedge b_2$ であるので、もし安全なプロトコルがあるのであれば perfect privacy から、P_1 は b_2 の情報がなにもわからないはず。反対に、P_1 の perfect privacy もあるので、$b_1 = 1$ としても P_1 と P_2 の間でやりとりされた情報 τ にあう P_1 の内部乱数 ρ_1 がなければいけない。
- 計算結果は、τ のみから得られて今計算結果は 0 であったはず（もともと P_1 の入力が 0 だから）。しかし、$b_1 = 1$ でも対応する P_1 の内部乱数 ρ_1 があるので、perfect correctness から P_2 の入力は $b_2 = 0$ でなければ行けなくなって P_2 の perfect privacy がなくなってしまう。よって矛盾。
- 逆に perfect correctness から始めると、$b_1 = 1$ かつ τ にあう ρ_1 が存在しないことになって、P_1 の perfect privacy がないことになってしまう。

実現できない関数（その２）

$n > 2$ の場合は、2 人参加者の場合に帰着する。

- n 人の参加者を、$\#V_1, \#V_2 \leq t$ となる重ならない二つのグループ V_1 と V_2 に分ける。$t \geq n/2$ なのでこれは可能。
- V_1 が P_1 の V_2 が P_2 の役割を果たすことで、2 人参加者のケースに帰着できる。

AND ではなく、OR も実現できないことを示せ。

MPC 計算実用化への試み

某企業の例

- 関数を回路（＝専用チップ）に直して実現するのは不便。またループが可変にあるようなものに向かない.
- 回路で MPC が実現できる（feasibility result) から、高級言語で MPC を実現へ $\Longrightarrow$ 高級言語の関数一つ一つを（高速な）MPC 計算に作り直す. 関数の品揃えを増やしている.
- 実現で悩んでいる難しい関数がまだまだ多数存在する.
- 今の所、Passive Case を中心に開発が進んでいる.

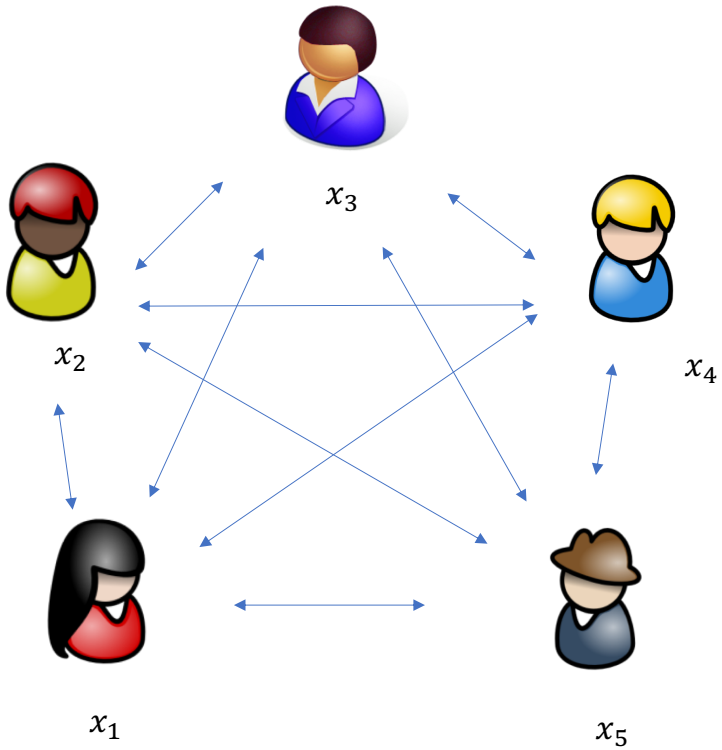
もくじ

安全性モデル

大雑把に言うと、現実が、理想と同じ、または十分近くなれば安全という。

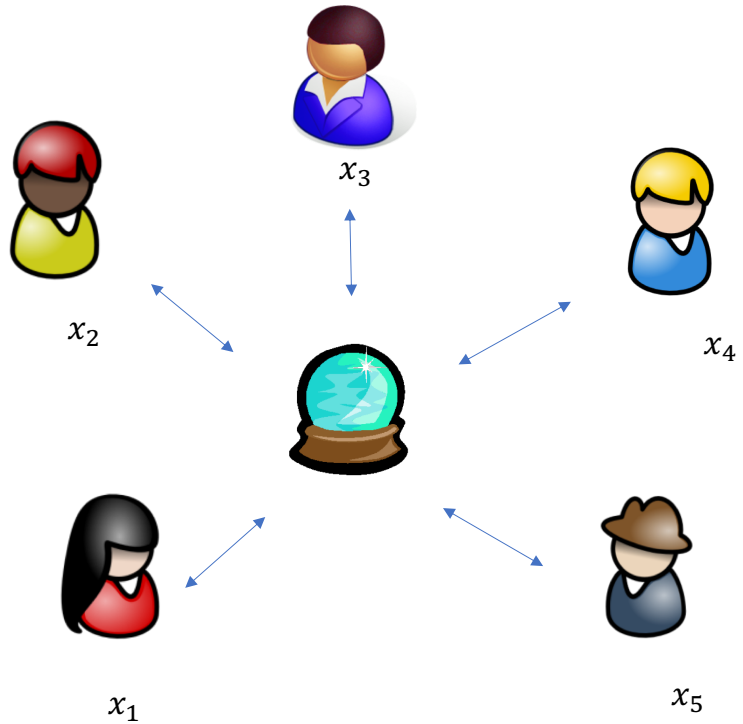
現実のプロトコル

互いに通信し計算結果を得る

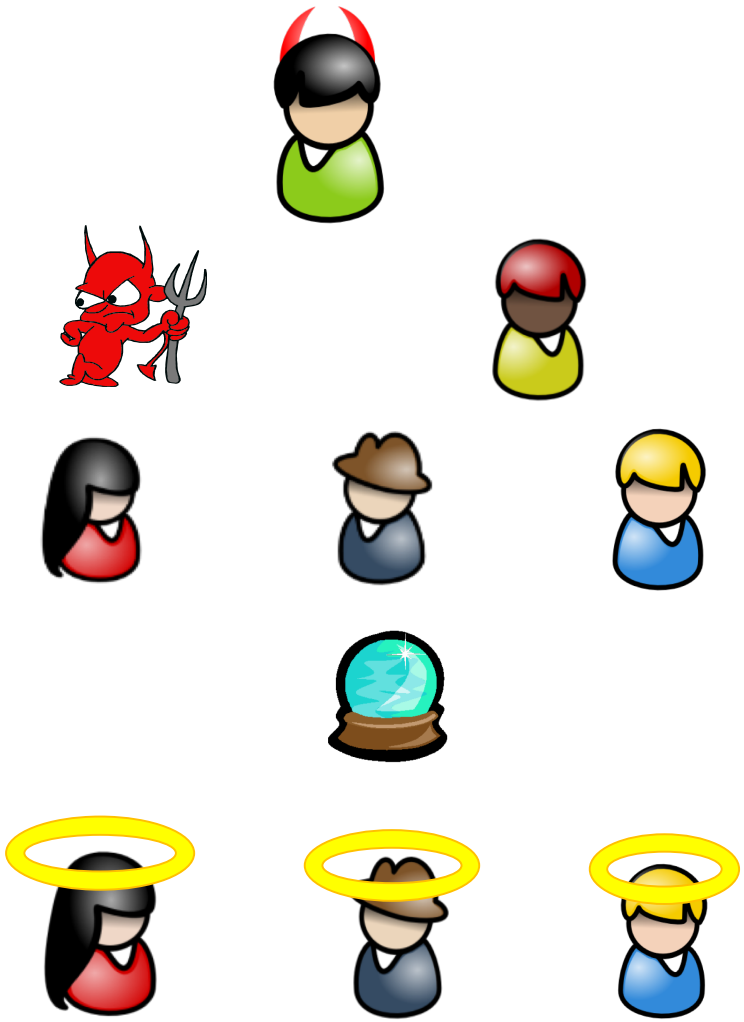


理想状態

各自の秘密を  に預けると計算結果を返してくれる



Environment Security



Environment Z

Adversary A / Simulator S

Parties (Alice, Bob, Charlie), who execute Protocol π

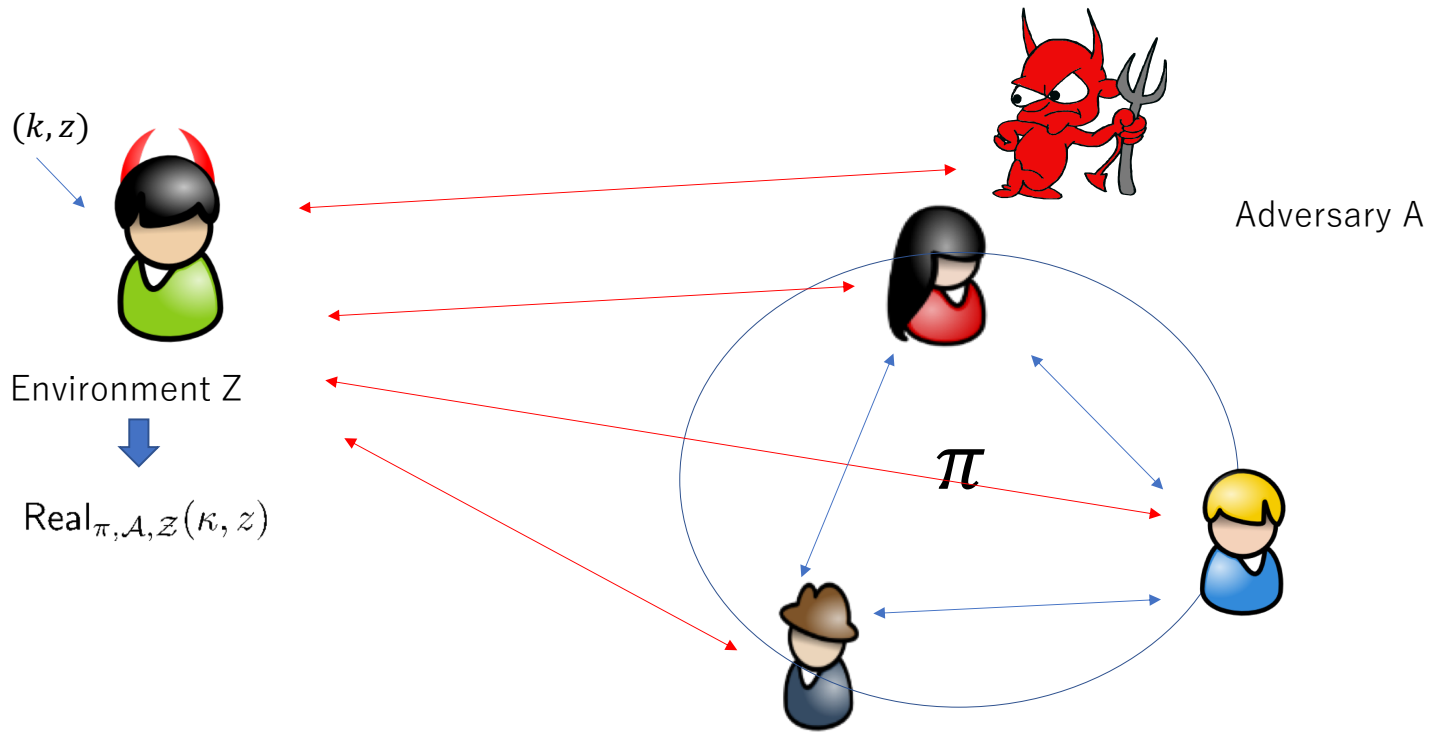
Ideal Functionality F

``functionality'' of F = ``functionality'' of π

Dummy Parties (Alice, Bob, Carlie)

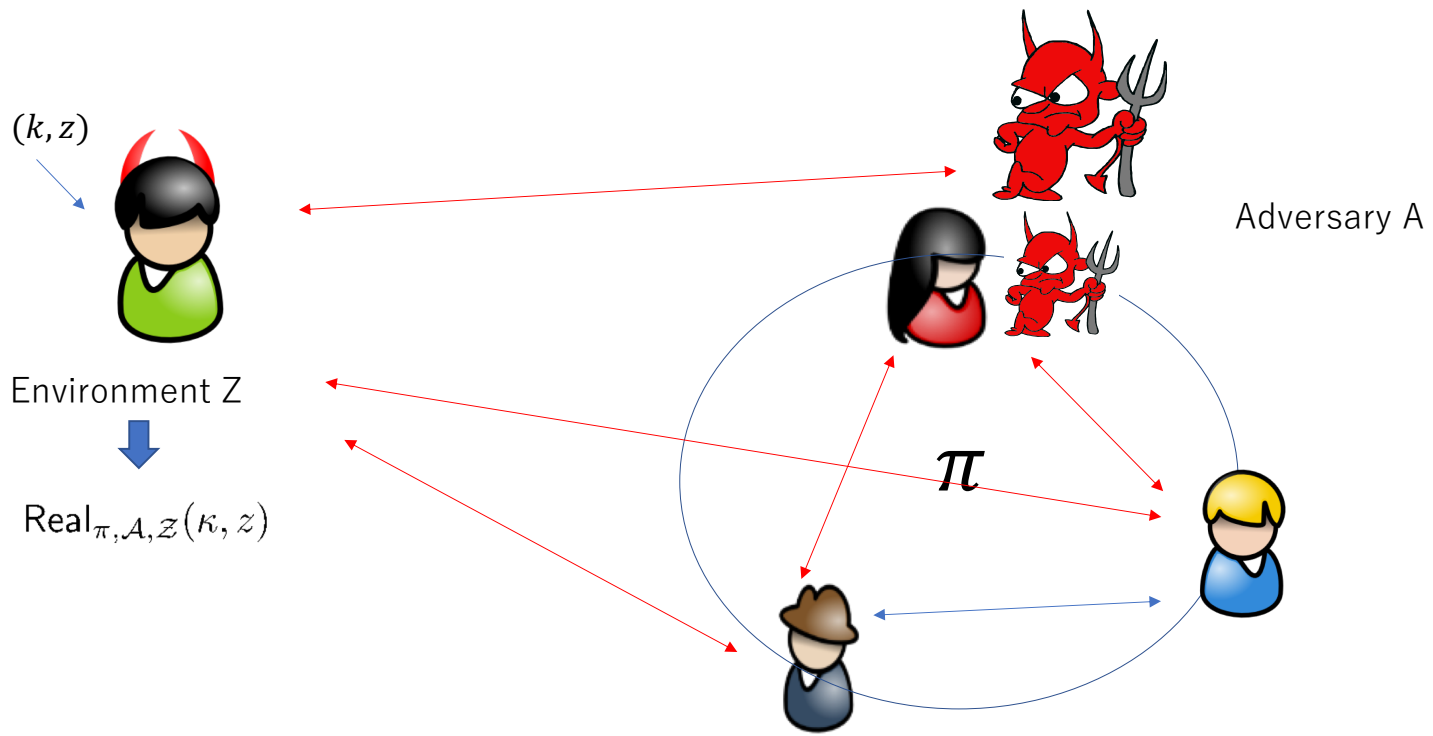
Real World

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。Adversary $\mathcal{A}$ は、$\mathcal{Z}$ の意のままに行動し、情報を $\mathcal{Z}$ に伝える。



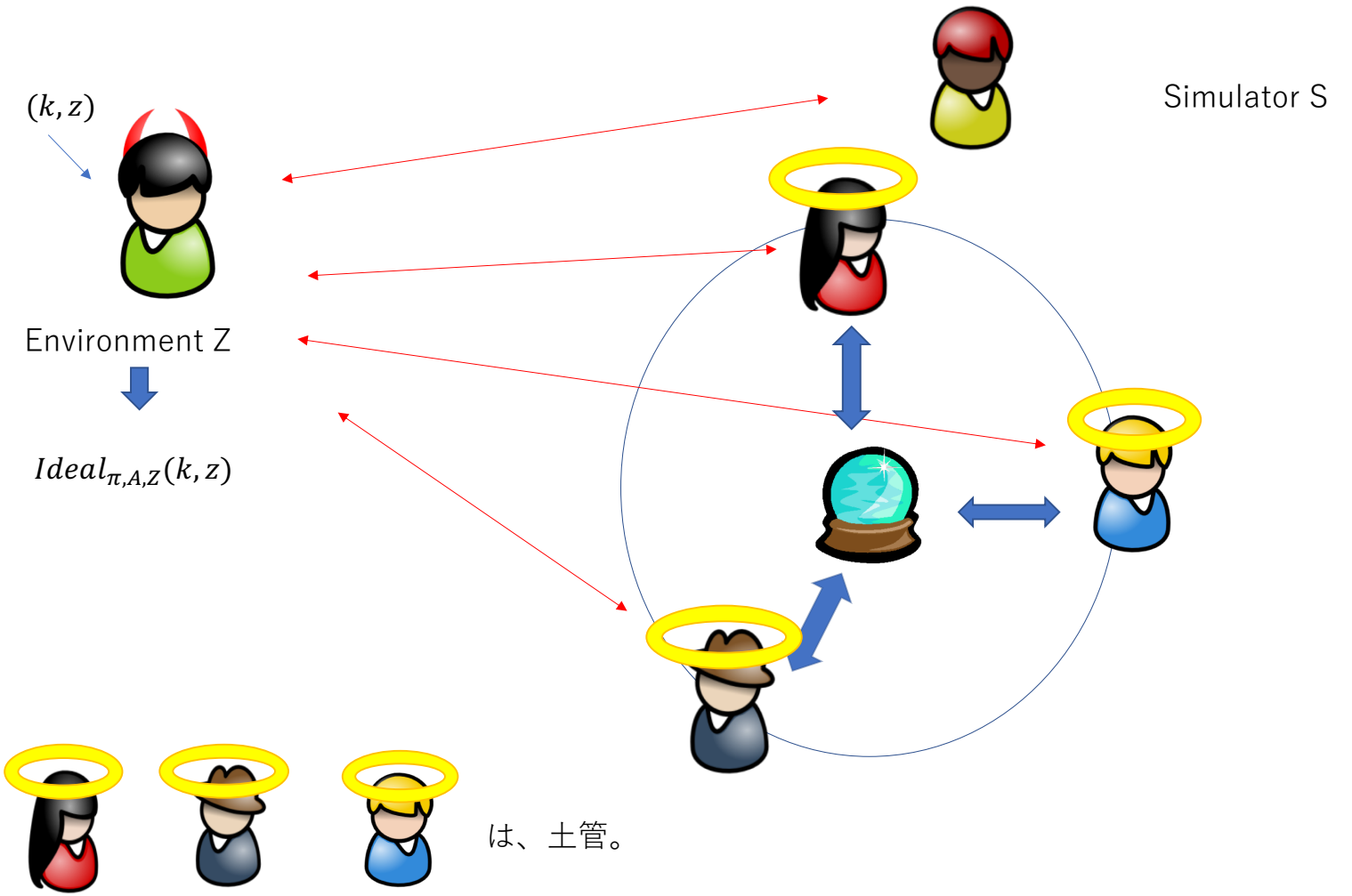
Real World (Alice Corrupted)

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。Adversary $\mathcal{A}$ は、$\mathcal{Z}$ の意のままに行動し、情報を $\mathcal{Z}$ に伝える。$\mathcal{A}$ は、Alice の（今までの内部情報を全て得て）代わりにプロトコルに参加する。



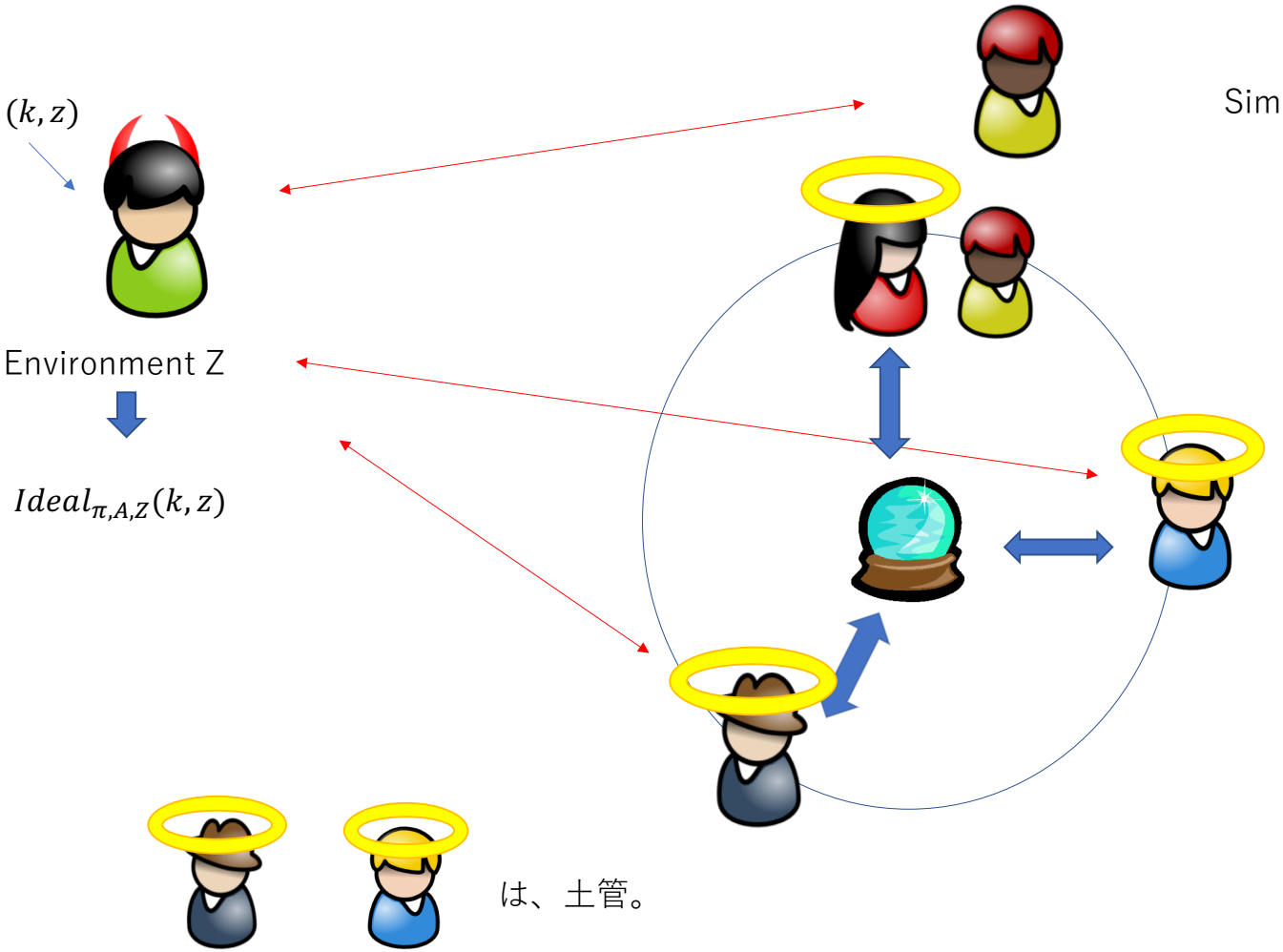
Ideal World

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。Simulator $\mathcal{S}$ は、$\mathcal{A}$ のふりをして、情報を $\mathcal{Z}$ に伝える。



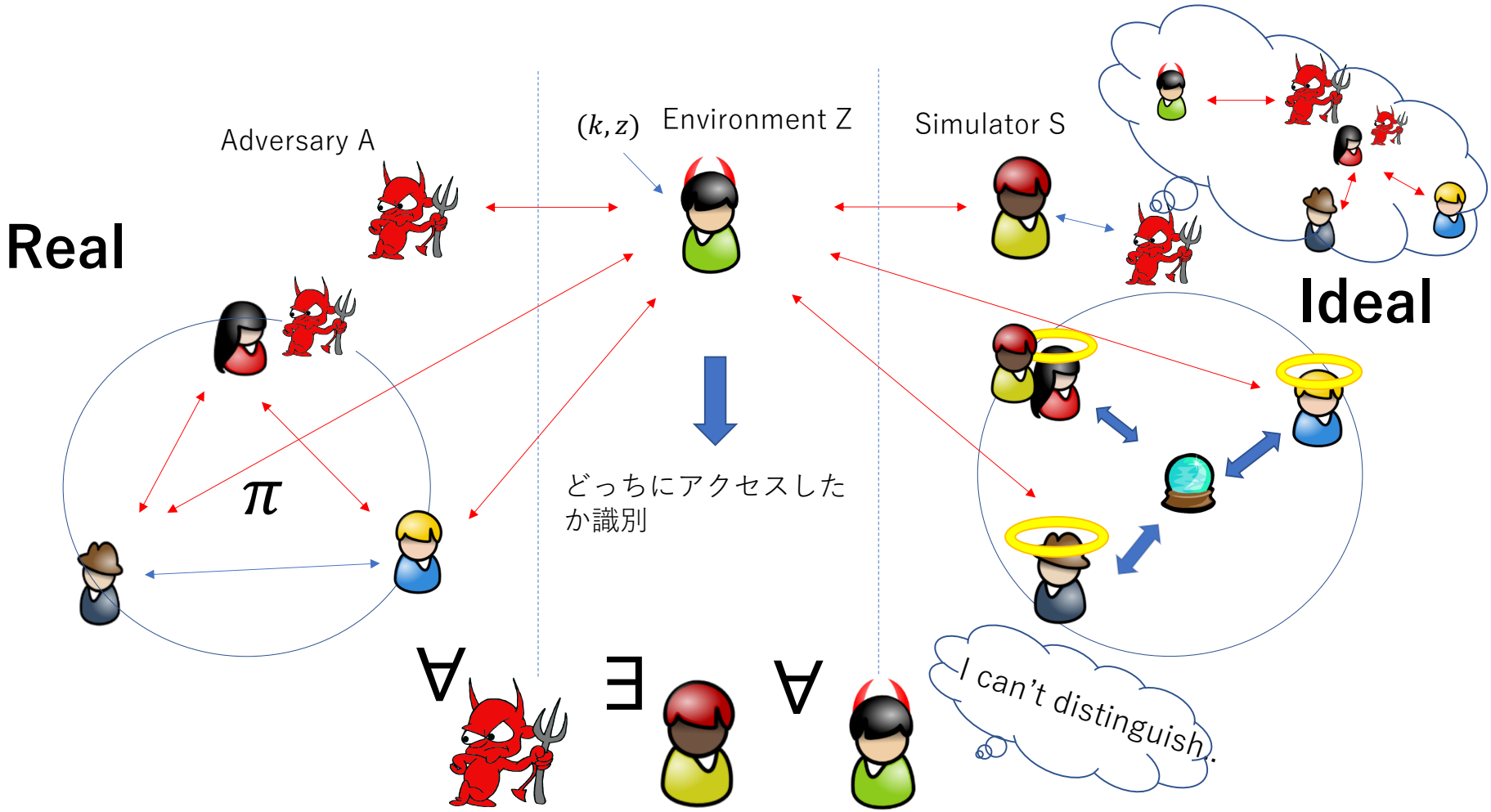
Ideal World (Alice Corrupted)

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。$\mathcal{S}$ は、Alice の（今までの内部情報を全て得て）代わりにプロトコルに参加する。



安全とは

Real と Ideal の世界を Environment $\mathcal{Z}$ が識別できなければ安全と定義



Adversary（攻撃者）

- Passive Adversary (aka semi-honest or honest-but-curious).
  - 受動的攻撃者
  - 正規のプロトコルからは逸脱しないが、プロトコルから得た情報から正直な参加者の秘密情報を得ようとする
- Active Adversary
  - 能動的攻撃者
  - プロトコルから逸脱しても良い。正直な参加者の秘密情報を得ようとすることと、プロトコルを失敗に終わらせようとする。
  - さらに、static case と adaptive case が存在する。
    - static adversary: プロトコルが始まる前に不正者が固定されている
    - adaptive adversary: プロトコル中に正直な参加者を corrupt して、今までの情報を得た上に不正者として操ることができる

知られている結果

不正者数（Adversary が corrupt できる参加者数）に応じた結果

	Passive	Active w/ broadcast	Active w/o broadcast
情報理論的安全	$< \frac{n}{2}$	$< \frac{n}{3}$	$< \frac{n}{3}$
統計的安全	$< \frac{n}{2}$	$< \frac{n}{2}$	$< \frac{n}{3}$
計算量的安全	n	$< \frac{n}{2}$	$< \frac{n}{2}$

ただし、$n > 2$.

- 情報理論的安全: Real と Ideal の分布が完全に同じ
- 統計的安全: Real と Ideal の分布の差がわずかな統計的差しかない。
- 計算量的安全: 多項式時間制限の Environment $\mathcal{Z}$ では、識別できないぐらいしか、Real と Ideal の分布に違いはない。

UC 安全性

- 特に Environment $\mathcal{Z}$ と Adversary $\mathcal{A}$ を「巻き戻さず」に Real の世界を Ideal で Simulator $\mathcal{S}$ が、シミュレートできるなら、プロトコル π を汎用結合安全性（UC 安全性）が実現できたという。

  “The protocol π UC realizes the ideal functionality $\mathcal{F}_\pi$”. のように使う
- すでに説明した BGW [BGW88] の受動的攻撃者に対する MPC は、参加者ごとの P2P の秘匿通信が仮定され、同期型のネットワークであれば、corrupt される参加者の数が半分未満であれば受動的攻撃者に対して情報理論的安全かつ UC 安全である。
- 今後説明する [CDN15] に記載される MPC（BGW のプロトコルの修正版）は、参加者ごとの P2P の秘匿通信が仮定され、同期型のネットワークであれば、corrupt される参加者の数が 1/3 未満であれば能動的攻撃者に対して情報理論的安全かつ UC 安全である。

もくじ

能動的攻撃者 (active adversary) 対策に向けて

Shamir の秘密分散では、Dealer が正直な場合、 $P_1, \ldots, P_n$ に $f(\alpha_1), \ldots, f(\alpha_n)$ がそれぞれ配られ、不正者が t 人以下 $(n \geq 3t+1)$ であれば、Reed-Solomon 符号の (Welch/Berlekamp) 復号によりもとの多項式 $f(X)$ が復元できる。

$$\hat{f}(\alpha_i) = f(\alpha_i) + e_i \text{ s.t. } W_H(\boldsymbol{e}) \leq t \quad \Longrightarrow \quad \text{Reconstruct } f(X)$$

しかし、Dealer が不正者であった場合これは成り立たない。Dealer が不正者であっても、プロトコルが受理された場合は、秘密分散を正しくやらざるを得ない方式を考える（計算量仮定無しで実現）。

$\Rightarrow$ Ben-Or/Goldwasser/Wigderson [BGW88] の $(t+1, n)$-検証可能秘密分散 (Verifiable Secret Sharing) 方式 $(n \geq 3t+1)$.

用語等の定義

- $[a]_i$:　秘密 a が、参加者間で正しく線型秘密分散されており、a を P_i が保持している状態。
- $[a]_\emptyset$: 誰も a を 1 人で保持していない状態
- $[a]$:　誰が a を 1 人で保持しているか興味ないので明記していない場合
- $P_i : a \rightarrow [a]_i$:　P_i が、秘密 a を正しく秘密分散する行為（コミットメント）。
- $P_i : [a]_i \Rightarrow a$:　P_i が、秘密 a を各参加者に配ったシェアを証拠として broadcast して開示する行為。
- $[a]_i \Rightarrow a$:　全参加者が a のシェアを broadcast して、全員が a を認識する行為。
- $P_i : a \leftarrow [a]_j$:　参加者が各自の $[a]_j$ のシェアを P_i に秘密通信で送り、P_i に a を教える行為。この行為により、$[a]_i$ という状態にもなることに注意。

不正者が全体の 1/3 未満であれば、BGW VSS の Distribution は、$[a]$ を正しくコミットし、Reconstruction は、開示を正しく行うこと常にできるので、上記の具体的な実現法の一つである。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.
- $[a], [b] \to [ab]$ をどうするか。$ab = \sum \lambda_{i,n} a_i b_i$ なので、各 P_i に a_i, b_i から自主的に $a_i b_i$ を作ってもらわなければいけない。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.
- $[a], [b] \to [ab]$ をどうするか。$ab = \sum \lambda_{i,n} a_i b_i$ なので、各 P_i に a_i, b_i から自主的に $a_i b_i$ を作ってもらわなければいけない。
- 回路への入力をビット $a \in \{0, 1\}$ にしなければいけない。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.
- $[a], [b] \to [ab]$ をどうするか。$ab = \sum \lambda_{i,n} a_i b_i$ なので、各 P_i に a_i, b_i から自主的に $a_i b_i$ を作ってもらわなければいけない。
- 回路への入力をビット $a \in \{0, 1\}$ にしなければいけない。
  $\Rightarrow [a(a-1)] = [0]$ を証明する！

能動的攻撃者モデルで使われる秘密分散

能動的攻撃者モデルで使われる秘密分散（コミットメント）：

$$[[a]]_D := ([a_1]_1, \ldots, [a_n]_n)$$

D の持つ秘密 a が参加者間で秘密分散され、さらに参加者 P_i の持つ a のシェア a_i がさらに参加者間で秘密分散されている状態。

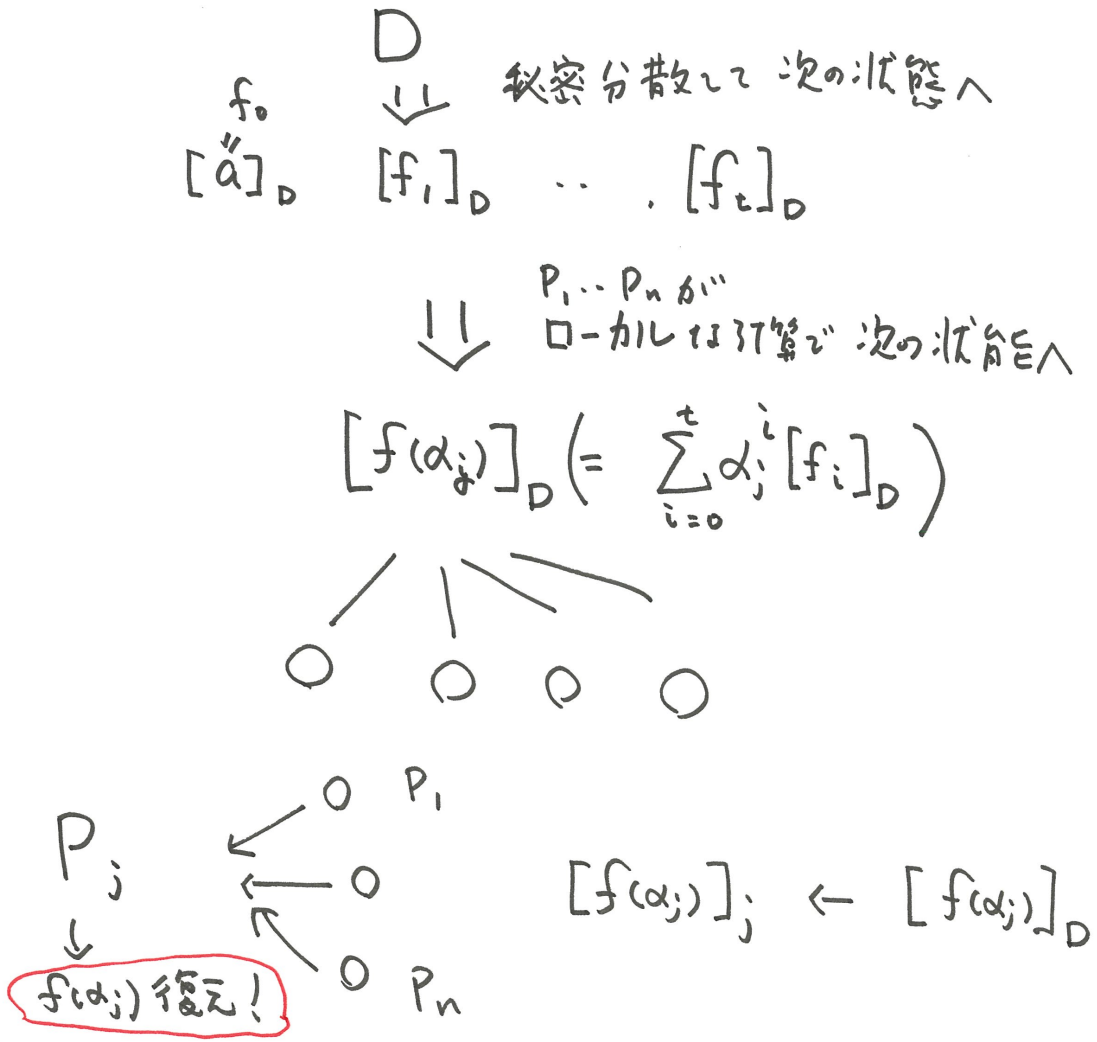
$D : a \to [[a]]_D$ は基本コマンドを使い次のように実現される。

1. $D : a \to [a]_D$.
2. D は、ランダムな $f_1, \ldots, f_t \in K$ を選び、$f_0 = a$ として多項式 $f(X) = \sum_{i=0}^{t} f_i X^i$ を定義する。
3. $D : f_1, \ldots, f_t \to [f_1]_D, \ldots, [f_t]_D$.
4. $P_1, \ldots, P_n$ は、線形性を生かしローカルな（互いの通信なし）計算で次の状態を作る。

$$[f(\alpha_1)]_D, \ldots, [f(\alpha_n)]_D \quad \text{where } [f(\alpha_j)]_D = \sum_{i=0}^{t} \alpha_j^i [f_i]_D.$$

5. 全ての P_i に対して、$P_i : f(\alpha_i) \leftarrow [f(\alpha_i)]_D$.
6. 各 P_i は、$a_i = f(\alpha_i)$ と設定する。

図で


D
秘密分散して次の状態へ
f_0
$[a]_D$
$[f_1]_D$ .. . $[f_t]_D$
$P_1 .. P_n$ が
ローカルな計算で次の状態へ
$[f(\alpha_j)]_D (= \sum_{i=0}^{t} \alpha_j^i [f_i]_D)$
P_1
P_j
P_n
$f(\alpha_j)$復元!
$[f(\alpha_j)]_j \leftarrow [f(\alpha_j)]_D$

足し算

次のように定義。

$$[[a]] + [[b]] := \Big([a_1]_1 + [b_1]_1, \ldots, [a_n]_n + [b_n]_n\Big)$$

$[\cdot]$ の線形性から、

$$[[a + b]] = [[a]] + [[b]]$$

が成立し、参加者間のローカルな計算で済む。実際の手順は以下の通り。

- Input: $[[a]] = ([a_1]_1, \ldots, [a_n]_n)$, $[[b]] = ([b_1]_1, \ldots, [b_n]_n)$.
- Output: $[[a + b]] = \Big([a_1 + b_1]_1, \ldots, [a_n + b_n]_n\Big)$.

1. $i = 1, \ldots, n$ に対して、$[a_i] + [b_i]$ を各参加者が実行。すなわち、各参加者が、$[a_i]$ と $[b_i]$ の自分のシェアをローカルに足し算し、$[a_i] + [b_i]$ の状態にする。$[\cdot]$ の線形性から、これは $[a_i + b_i]$ の状態と同じ。
2. $i = 1, \ldots, n$ に対して、各 P_i が $a_i + b_i$ をローカルに計算し保持する。その結果、$[a_i + b_i]_i$ の状態になる。

図で

$$[\![a]\!] + [\![b]\!] := ([a_1]_1 + [b_1]_1 , \ldots , [a_n]_n + [b_n]_n)$$

$[\![a]\!] = ([a_1]_1, \ldots [a_n]_n)$　　$[\![b]\!] = ([b_1]_1, \ldots [b_n]_n)$

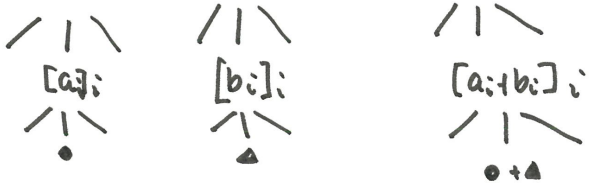
$[a_1]_1 + [b_1]_1 \overset{\text{linearity}}{=} [a_1+b_1]_1$

$[a_n]_n + [b_n]_n \overset{\text{linearity}}{=} [a_n+b_n]_n$

$P_i : a_i, b_i \to a_i + b_i$ 保持

$P_1, \ldots P_n : [a_i]_i + [b_i]_i \overset{\text{linearity}}{=} [a_i+b_i]_i$

$\Rightarrow [\![a]\!] + [\![b]\!] = [\![a+b]\!]$



ちょっとした注意

$[[a]]$ は、特に誰が秘密 a を保持しているか気にしていないことを示す表現。実際、

- $[[a]]_i + [[b]]_i = [[a+b]]_i$
- $[[a]]_i + [[b]]_j = [[a+b]]_\emptyset$ for $i \neq j$.
- $[[a]]_\emptyset + [[b]]_j = [[a+b]]_\emptyset$.
- $[[a]]_\emptyset + [[b]]_\emptyset = [[a+b]]_\emptyset$.

などのバリエーションが可能であるが、しかしいずれの場合も、

$$[[a+b]] = \Big([a_1 + b_1]_1, \ldots, [a_n + b_n]_n\Big)$$

は成り立っている。つまり、秘密を 1 人で保持している参加者がいるかどうかに関わらず、各シェア $a_i + b_i$ は、P_i により保持され、$a_i + b_i$ も参加者間で正しく秘密分散されている。

スカラー倍

$\lambda \in K$ に対して、次のように定義。

$$\lambda[[a]] := \left(\lambda[a_1]_1, \ldots, \lambda[a_n]_n\right)$$

$[\cdot]$ の線形性から、

$$[[\lambda a]] = \lambda[[a]]$$

が成立し、参加者間のローカルな計算で済む。実際の手順は以下の通り。

- Input: λ, $[[a]] = ([a_1]_1, \ldots, [a_n]_n)$
- Output: $[[\lambda a]] = \left([\lambda a_1]_1, \ldots, [\lambda a_n]_n\right)$.

1. $i = 1, \ldots, n$ に対して、$\lambda[a_i]$ を各参加者が実行。すなわち、各参加者が、$[a_i]$ の自分のシェアとスカラー λ をローカルに掛け算し、$\lambda[a_i]$ 状態を作る。$[\cdot]$ の線形性から $[\lambda a_i]$ の状態になる。
2. $i = 1, \ldots, n$ に対して、各 P_i が λa_i をローカルに計算し保持。その結果 $[\lambda a_i]_i$ の状態になる。

図で

$$\lambda\,[\![a]\!] \;:=\; (\lambda[a_1]_1, \ldots \lambda[a_n]_n)$$

$$[a_1]_1 \ldots [a_n]_n$$

$$P_i : \lambda, a_i \to \lambda a_i \;(\text{secret})$$

$$P_1 \ldots P_n : \lambda\,[a_i] = [\lambda a_i] \;(\text{linearity})$$

Commitment Multiplication Protocol (CMP) I

このプロトコル CMP は、$[[a]]$, $[[b]]$ から、$[[ab]]$ を構成するときに使われるサブプロトコルである。

- Input: $[a]_D$, $[b]_D$, $[c]_D$
- Output: accepts if $c = ab$; otherwise, rejects.

1. D は、ランダムな $f_1, \ldots, f_t, g_1, \ldots, g_t \in K$ を選び、多項式 $f(X) = a + \sum_{i=1}^{t} f_i X^i$, $g(X) = b + \sum_{i=1}^{t} g_i X^i$ を定義。D は、$h(X) = f(X)g(X) = \sum_{i=0}^{2t} h_i X^i$ を計算する。$h(0) = f(0)g(0) = c$ に注意。
2. $D : f_1, \ldots, f_t \to [f_1]_D, \ldots, [f_t]_D$.
3. $D : g_1, \ldots, g_t \to [g_1]_D, \ldots, [g_t]_D$.
4. $D : h_1, \ldots\ldots, h_{2t} \to [h_1]_D, \ldots\ldots, [h_{2t}]_D$.
5. $P_1, \ldots, P_n$ は、線形性を生かしローカルな計算で次の状態を作る。

$$[f(\alpha_1)]_D, \ldots, [f(\alpha_n)]_D, \quad [g(\alpha_1)]_D, \ldots, [g(\alpha_n)]_D, \quad [h(\alpha_1)]_D, \ldots, [h(\alpha_n)]_D$$

6. 全ての P_i に対して、$P_i : f(\alpha_i), g(\alpha_i), h(\alpha_i) \leftarrow [f(\alpha_i)]_D, [g(\alpha_i)]_D, [h(\alpha_i)]_D$.

Commitment Multiplication Protocol (CMP) II

7. P_i は、$f(\alpha_i)g(\alpha_i) = h(\alpha_i)$ が成り立つかチェックをして、もし成り立たなければ (accuse, P_i, D) を broadcast する。
8. 全ての (accuse, P_i, D) に対して、参加者全員が協力して $f(\alpha_i), g(\alpha_i), h(\alpha_i)$ を開示する。すなわち、

$$[f(\alpha_i)]_i, [g(\alpha_i)]_i, [h(\alpha_i)]_i \Rightarrow f(\alpha_i), g(\alpha_i), h(\alpha_i)$$

9. accuse していない P_j は、一つでも、$f(\alpha_i)g(\alpha_i) \neq h(\alpha_i)$ となるものがあれば、(accuse, P_j, D) を broadcast する。
10. Step 7 と Step 9 の合計 accuse 数が $t+1$ 以上であれば、プロトコルは拒否 (reject) される。そうでなければ、受理 (accept) される。

(Correctness) D が正直な場合

- 正直な P_i は、$f(\alpha_i)g(\alpha_i) = h(\alpha_i)$ が成り立つので、accuse を broadcast することはない。
- 不正直な P_i は、accuse をする可能性があるが、accuse の合計数は高々 t.
- (accuse, P_i, D) に対して、全参加者が協力して $f(\alpha_i)$, $g(\alpha_i)$, $h(\alpha_i)$ を開示した場合、$f(\alpha_i)g(\alpha_i) = h(\alpha_i)$ が成り立つことが分かるので、正直な参加者が新たに accuse することは無い。
- よって、プロトコルは accept される。

(Correctness) D が不正直な場合

- プロトコルが accept されたにも関わらず、$c \neq ab$ であったと仮定する。
- 正直な参加者の数は $n - t \geq 2t + 1$. もし、全ての正直な参加者の各自のシェアが、$f(\alpha_i)g(\alpha_i) = h(\alpha_i)$ が成立してしまったとする。すると $\deg(h) \leq 2t$ より、$[c]_D = [ab]_D = \sum_{i \in H} \lambda_{i,H}[f(\alpha_i)g(\alpha_i)]_D$ が成立してしまい、$c = ab$ に反する。
- よって、正直な参加者のシェアの中に $f(\alpha_i)g(\alpha_i) \neq h(\alpha_i)$ となるものを紛らわせなければならない。
- しかし、正直な P_i に $f(\alpha_i)g(\alpha_i) \neq h(\alpha_i)$ となる $f(\alpha_i)$, $g(\alpha_i)$, $h(\alpha_i)$ を送ると必ず Step 7 で $(\mathsf{accuse}, P_i, D)$ を broadcast され、Step 9 で、他の正直な参加者も全員 accuse を broadcast するため、accuse 数が t を超え、プロトコルは必ず reject されてしまう。
- よって、正直な参加者に不正なシェアを送った場合、プロトコルは必ず reject されるので、矛盾。

Commitment Sending Protocol (CSP)

このプロトコル CSP は、$D : a \to [[a]]_D$ にするプロトコルの中で使われていたサブプロトコル。必要なので定義しておく。

- Input: $[a]_D$,
- Output: $[[a]]_D$; otherwise, rejects.

- D は、ランダムな $f_1, \ldots, f_t \in K$ を選び、$f_0 = a$ として多項式 $f(X) = \sum_{i=0}^{t} f_i X^i$ を定義する。
- $D : f_1, \ldots, f_t \to [f_1]_i, \ldots, [f_t]_i$.
- $P_1, \ldots, P_n$ は、線形性を生かしローカルな（互いの通信なし）計算で次の状態を作る。

$$[f(\alpha_1)]_i, \ldots, [f(\alpha_n)]_i \quad \text{where } [f(\alpha_j)] = \sum_{i=0}^{t} \alpha_j^i [f_i].$$

- 全ての P_i に対して、$P_i : f(\alpha_i) \leftarrow [f(\alpha_i)]_D$.
- 各 P_i は、$a_i = f(\alpha_i)$ と設定する。
- $[[a]]_D = ([a]_1, \ldots, [a]_n)$ の状態になる。

掛け算（考え方）

$[[a]], [[b]]$ から、$[[ab]]$ を構成する掛け算プロトコルを考える。$[[a]] = \left([a_1]_1, \ldots, [a_n]_n\right)$, $[[b]] = \left([b_1]_1, \ldots, [b_n]_n\right)$ とする。$\hat{c}_i = a_i b_i$ の関係を満たす $\hat{c}_i$ を秘密分散して、$[[\hat{c}_i]]$ の状態にした参加者 P_i の集合を S とする。正直な人は正しく $[[\hat{c}_i]]$ をつくるので、$\#S \geq n - t = 2t + 1$. よって

$$ab = \sum_{i \in S} \lambda_{i,S} a_i b_i$$

$[[\cdot]]$ の線形性より、

$$[[ab]] = \sum_{i \in S} \lambda_{i,S} [[a_i b_i]]$$

となり、$[[ab]]$ を実現できることがわかる。

掛け算

- Input: $[[a]] = \left([a_1]_1, \ldots, [a_n]_n\right)$, $[[b]] = \left([b_1]_1, \ldots, [b_n]_n\right)$.
- Output: $[[c]] = ([c_1]_1, \ldots, [c_n]_n)$ where $c = ab$.

1. 各 P_i は、$a_i b_i$ を計算し、$\mathrm{CMP}([a_i]_i, [b_i]_i, [a_i b_i]_i)$ を実行する。このプロトコルが受理されると、$[a_i b_i]_i$ という状態になる。
2. 各 P_i は、$[a_i b_i]_i$ を $[[a_i b_i]]_i$ の状態にするために、$P_1, \ldots, P_n$ の間でサブプロトコル $\mathrm{CSP}([a_i b_i]_i)$ を行う。
3. CMP プロトコルを受理された P_i の集合を S とする。各参加者は各自がローカルに計算し
$$\sum_{i \in S} \lambda_{i,S} [[a_i b_i]]$$
という状態を作る。

$\#S \geq n - t \geq 2t + 1$ と、$[[\cdot]]$ の線形性より、

$$[[ab]] = [[\sum_{i \in S} \lambda_{i,S} a_i b_i]] = \sum_{i \in S} \lambda_{i,S} [[a_i b_i]]$$

という状態になっており目的を達している。

おしまい

より詳しいスライドは以下に

- I486S 暗号プロトコル理論
- URL:https://www.jaist.ac.jp/~fujisaki/i486S

Thanks for listening !

もくじ

検証可能秘密分散 (VSS)

Ben-Or/Goldwasser/Wigderson [BGW88] の $(t+1, n)$-検証可能秘密分散 (Verifiable Secret Sharing) 方式 $(n \geq 3t+1)$

目的: Dealer D に秘密分散

$$[s] = (s_1, \ldots, s_n) = (f(\alpha_1), \ldots, f(\alpha_n))$$

を正しく実行させたい。

- Dealer D
- Players $P_1, \ldots, P_n$
- Perfect Privacy: Dealer が正直な場合、Dealer の秘密情報 $s \in K$ は、t 人以下の攻撃者 $\mathcal{A}_{t,n}$ に対して一切情報を与えない。
- Perfect Reconstruction: VSS 方式が参加者間で受理 (accept) された場合、正直な参加者 $P_{h_1}, \ldots, P_{h_{n-t}}$ は、t 次のある多項式 f の点 $(f(\alpha_{h_1}), \ldots, f(\alpha_{h_{n-t}}))$ をそれぞれもつ。
  $\Longrightarrow$ すなわち、正直な $t+1$ 人以上の参加者のみで、または n 人の中に不正者が t 人以下混じっていて、それを正直な参加者が認識できていなくても秘密 $f(0)$ を正しく復元できる。

BGW VSS

Dealer は、次のような二変数対称多項式を（ランダムに）選ぶ。

Bivariable Symmetric Polynomial

$$F(X,Y) = \sum_{i,j=0}^{t} r_{i,j} X^i Y^j \text{ where } r_{00} = s,$$

such that $\deg_X(F) = \deg_Y(F) = t$ and, for all i, j, $r_{i,j} = r_{j,i}$.

Notes:

- $F(\alpha_i, \alpha_j) = F(\alpha_j, \alpha_i)$ for all i, j.
- $f(X) \triangleq F(X, 0)$: real sharing polynomial.
- $s = F(0,0)$: real secret.
- $s_i = f(\alpha_i) = F(\alpha_i, 0) = F(0, \alpha_i)$: P_i's real share on $f(X)$.
- $f_i(X) \triangleq F(X, \alpha_i)$: P_i's verification polynomial.

BGW VSS (Distribution Phase) I

プロトコル実行中、D への accuse が $t+1$ 以上となった時点でプロトコルは拒絶され強制終了するものとする。

1. D は、$s = F(0,0)$ とし、検証多項式 $f_i(X) = F(X, \alpha_i)$ を P_i $(i = 1, \ldots, n)$ にそれぞれ（秘匿通信で）送る。
2. 各 P_i は、$\deg(f_i) \neq t$ の場合、(accuse, i, D) を broadcast*し、Step 9 でプロトコルが受理されるまで復活しない。

   *: broadcast channel を仮定していないが、Byzantine algorithm で broadcast を実現する（$n > 3t$ より可能).
3. $\deg(f_i) = t$ を確認できた P_i は $s_i = f_i(0)$ を s の自らのシェアと設定する。各 P_i は、$s_{ij} = f_i(\alpha_j)$ を（秘匿通信で）P_j $(j \in \{1, \ldots, n\} \backslash \{i\})$ に送る。

   D が不正をしていなければ $s_i = f_i(0) = F(0, \alpha_i) = F(\alpha_i, 0) = f(\alpha_i)$.
4. 各 P_i は、P_j $(j \in \{1, \ldots, n\} \backslash \{i\})$ から送られてきた s_{ji} に対して、$s_{ij} = s_{ji}$ であるかチェックする。各 P_i は、送られてきた s_{ji} に対して、$s_{ij} \neq s_{ji}$ となるものを発見した時、P_i は、$(\text{dispute}, i, j)$ を broadcast する。

   D, P_i, P_j が全て正直なら、$s_{ij} = f_i(\alpha_j) = F(\alpha_j, \alpha_i) = F(\alpha_i, \alpha_j) = f_j(\alpha_i) = s_{ji}$.
5. D は、各 $(\text{dispute}, i, j)$ に対して、$\hat{s}_{ij}$ を broadcast する。

   D が正直であれば、$\hat{s_{ij}} = F(\alpha_i, \alpha_j)$ である。

BGW VSS (Distribution Phase) II

6. (dispute, i, j) に対して broadcast された $\hat{s}_{ij}$ を、P_i と P_j はそれぞれ $\hat{s}_{ij} = s_{ij}$ と $\hat{s}_{ij} = s_{ji}$ であるかチェックする。もし、自分のものと一致しないとわかったら、その参加者（P_i とする）は、(accuse, i, D) を broadcast し、Step 9 でプロトコルが受理されるまで復活しない。
7. D は、これまでの全ての (accuse, i, D) に対して、$\hat{f}_i(X)$ を broadcast する。
   D が正直であれば、$\hat{f}_i(X) = f_i(X)$ である。
8. $\hat{f}_i(X)$ が broadcast されたとき、これまで accuse していない P_j は、$\deg(\hat{f}_i) = t$ と
$$s_{ji} = f_j(\alpha_i) = \hat{f}_i(\alpha_j)$$
   が成立するかチェックし、成立しなければ (accuse, j, D) を broadcast する。
9. これまでの accuse 数が合わせて t 以下の場合は、プロトコルは受理 (accept) され、これまでに accuse をした P_i も、$\hat{f}_i(X)$ を新たな自分の検証多項式とみなし $f_i(X) \triangleq \hat{f}_i(X)$ とおき、$s_i = f_i(0)$ を自分のシェアとする。

BGW VSS (Reconstruction Phase)

Reconstruction

正直な $n - t(\geq 2t + 1)$ 人の参加者は、Reed-Solomon 符号の復号により、$f(X)$ を復元することができ、$s = f(0)$ を復元できる。

- Step 6 で合計 accuse 数が $t + 1$ 以上になりプロトコルが強制終了されなければ、D が正直であろうとなかろうと、少なくとも $t + 1$ 人以上の正直な参加者間で t 次多項式 $f(X)$ が補間（秘密分散）される。
- プロトコルが受理された時、D が正直であろうとなかろうと、正直な全ての参加者間で t 次多項式 $f(X)$ が補間（秘密分散）される。

Perfect Reconstruction

Theorem 4

BGW VSS 方式が受理されたとき、全ての正直な参加者 $P_i \in H$ は、ある（同じ）t 次多項式 $f(X)$ の各点 $f(\alpha_i)$ をシェアとしてもつ。よって、$\#H = n - t \geq 2t + 1$ より *Reconstruction* で秘密 $f(0)$ が常に正しく復元できる。

定理 4 の証明（D が正直な場合）

D が正直であれば、正直な参加者 P_i は dispute をすることはあっても、accuse をすることはない。不正な参加者の数は t であるから、プロトコルは常に受理され、各 P_i は、$f_i(X)$ を保持し、D は正直なため
$f_i(0) = F(0, \alpha_i) = F(\alpha_i, 0) = f(\alpha_i)$ が成り立つ。 □

定理 4 の証明（D が不正者の場合）

D を不正者とし、C を Step 2 と Step 6 で accuse しない正直な参加者の集合とする。Step 6 で、プロトコルが終了しないとしたら、その時点で accuse した参加者の数は高々 t. よって、$\#C \geq n - \#\{\text{不正者}\} - \#\text{accuse} \geq n - t - t = n - 2t \geq t + 1$.

$$g(X) = \sum_{i \in C} \lambda_{i,C} f_i(X)$$

とおく。任意の $P_j \in H$ に対して、

$$\begin{aligned} g(\alpha_j) &= \sum_{i \in C} \lambda_{i,C} f_i(\alpha_j) \\ &= \sum_{i \in C} \lambda_{i,C} f_j(\alpha_i) \quad (\text{At Step 9: } f_i(\alpha_j) = f_j(\alpha_i) \text{ for all } i \in C, j \in H.) \\ &= f_j(0) \quad (deg(f_j) = t, \#C \geq t + 1 \Longrightarrow f_j(0).) \\ &= s_j. \end{aligned}$$

C に属する参加者は全員正直なので、$deg(f_i) = t$ より $deg(g) = t$. よって、正直な参加者 P_j は共通の t 次多項式 $g(X)$ の各点 $g(\alpha_j)$ を保持する。

定理 4 の別証明（D が不正者の場合）

まず次の lemma を考える。

Lemma 5

$f_1(X), \ldots, f_p(X) \in K[X]$ を p 個の体 K を係数とする t 次多項式。
$\alpha_1, \ldots, \alpha_p \in K$ は互いに異なる値とする。$p \geq t+1$ で、全ての $i, j \in \{1, \ldots, p\}$ に対して、$f_i(\alpha_j) = f_j(\alpha_i)$ が成立するとすると、ある $\deg_X(F) = \deg_Y(F) = t$ である二変数対称多項式 $F(X, Y)$ が存在して、

$$f_i(X) = F(X, \alpha_i) \quad \textit{for } i \in \{1, \ldots, p\}$$

となる。

$p = t + 1$ の場合

列ベクトル $\boldsymbol{x}$, $\boldsymbol{\mu_i}$ を $\boldsymbol{x} = (1, x, \ldots, x^t)^T$, $\boldsymbol{\mu_i} = (1, \alpha_i, \ldots, \alpha_i^t)^T$ と定義する。$f_i(X)$ の係数を要素とする列ベクトルを $\boldsymbol{f_i}$ とすると、$f_i(X) = \boldsymbol{x}^T \boldsymbol{f_i}$ と表せる。$\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_{t+1}}$ が $t+1$ 次元ベクトル空間の独立なベクトルであり、$M \triangleq (\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_{t+1}})$ は $(t+1) \times (t+1)$ の Van de monde 行列だから正則。よって、$(t+1) \times (t+1)$ 行列 $R = (\boldsymbol{f_1}, \ldots, \boldsymbol{f_{t+1}})M^{-1}$ とすると $(\boldsymbol{f_1}, \ldots, \boldsymbol{f_{t+1}}) = RM$. よって $\boldsymbol{f_i} = R\boldsymbol{\mu_i}$. すなわち

$$f_i(X) = \boldsymbol{x}^T R \boldsymbol{\mu_i} \qquad \text{where } i = 1, \ldots, t+1 \tag{1}$$

$f_i(\alpha_j) = f_j(\alpha_i)$ と、$f_j(\alpha_i) = f_j(\alpha_i)^T = (\boldsymbol{\mu_i}^T R \boldsymbol{\mu_j})^T = \boldsymbol{\mu_j}^T R^T \boldsymbol{\mu_i}$ より、

$$\boldsymbol{\mu_j}^T R \boldsymbol{\mu_i} = \boldsymbol{\mu_j}^T R^T \boldsymbol{\mu_i}. \tag{2}$$

これが、全ての $i, j \in \{1, \ldots, t+1\}$ に対して成り立つから、$M^T R M = M^T R^T M$. M は正則だから、$R = R^T$. これより、二変数対称多項式 $F(X, Y) = \boldsymbol{x}^T R \boldsymbol{y}$（ただし、$\boldsymbol{y} = (1, Y, \ldots, Y^t)^T$）が存在して $f_i(X) = F(X, \alpha_i) (= \boldsymbol{x}^T R \boldsymbol{\mu_i})$ となる。 □

$p > t+1$ の時

$Q = \{1, \ldots, t+1\}$ とする。$i \in Q$ に対しては、$p = t+1$ の場合から二変数対称多項式 $F(X, Y) = \boldsymbol{x}^T R \boldsymbol{y}$ $(R = R^T)$ が存在して $f_i(X) = \boldsymbol{x}^T R \boldsymbol{\mu_i}$ となる。ここで、

$$f_i(\alpha_j) = \boldsymbol{\mu_j}^T R \boldsymbol{\mu_i} = (\boldsymbol{\mu_j}^T R \boldsymbol{\mu_i})^T = \boldsymbol{\mu_i}^T R \boldsymbol{\mu_j}.$$

$j \notin Q$ の場合、$f_j(X) = \boldsymbol{x}^T \boldsymbol{f_j}$ とおく。

$$f_j(\alpha_i) = f_i(\alpha_j) \qquad \text{for all } i \in Q. \tag{3}$$

から、

$$\boldsymbol{\mu_i}^T \boldsymbol{f_j} = \boldsymbol{\mu_i}^T R \boldsymbol{\mu_j} \qquad \text{for all } i \in Q. \tag{4}$$

$M = (\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_{t+1}})$ として、(4) から $M^T \boldsymbol{f_j} = M^T R \boldsymbol{\mu_j}$. M は正則なので、$\boldsymbol{f_j} = R \boldsymbol{\mu_j}$.
よって、$f_j(X) = \boldsymbol{x}^T R \boldsymbol{\mu_j}$ $(j \notin Q)$. □

D が不正者の場合の証明

- C: Step 2 と Step 6 で accuse しない正直な参加者の集合
- $\bar{C}$: Step 2 または Step 6 で accuse した正直な参加者の集合。すなわち、$H = C \sqcup \bar{C}$.

Step 6 で、プロトコルが終了しないとしたら、その時点で accuse した参加者の数は高々 t. よって、

$$\#C \geq n - \#\{\text{不正者}\} - \#\text{accuse} \geq n - t - t = n - 2t \geq t + 1.$$

Lemma 5 より、$i, j \in C$ に対して、$f_i(\alpha_j) = f_j(\alpha_i)$ から、全ての $i \in C$ に対して、$f_i(X) = \boldsymbol{x}^T R \boldsymbol{\mu_i}$ となる。$j \in \bar{C}$ は、$\hat{f}_i(X)$ を D から受け取るが、プロトコルが最終的に受理するならば、

$$\hat{f}_j(\alpha_i) = f_i(\alpha_j) \qquad \text{for all } i \in Q(\subset C).$$

Q は、C の任意の $\#Q = t + 1$ となる部分集合。すると $p > t + 1$ の場合と同様で、$\hat{f}_j(X) = \boldsymbol{x}^T R \boldsymbol{\mu_j}$ $(j \in \bar{C})$. これより、プロトコルが受理されるなら正直な参加者全員に対して、

$$f_i(X) = \boldsymbol{x}^T R \boldsymbol{\mu_i} = F(X, \alpha_i) \qquad \text{for all } i \in H.$$

よって、$f_i(0) = F(0, \alpha_i) = F(\alpha_i, 0) = f(\alpha_i)$. □

Perfect Privacy

Theorem 6

D が正直な場合、*Distribution Phase* を実行することで、不正者にもれる正直な参加者のシェア多項式 $f_i(X)$ $(P_i \in H)$ に関する情報は、$\{f_i(\alpha_j) \mid P_i \in H,\ P_j \in \mathcal{A}_{t,n}\}$ のみで、また D の秘密情報 s は一切漏れない。

不正者の共有できる情報は $\{f_j(X) = F(X, \alpha_j) \mid P_j \in \mathcal{A}_{t,n}\}$ である。この情報から、正直な参加者の多項式 $f_i(X)$ の点 $f_i(\alpha_j)$ $(P_i \in H,,\ P_j \in \mathcal{A}_{t,n})$ の情報は得られるが、それ以上の情報はなにも得られない。

Theorem 6 の証明 I

$k(X)$ を任意の

$$k(0) = 1 \quad \text{and} \quad k(\alpha_j) = 0 \ \text{ for all } P_j \in \mathcal{A}_{t,n}$$

となる t 次多項式とする。$k(X, Y) = k(X)k(Y)$ と定義する。すると、$k(X, Y)$ は二変数対称多項式で、

$$\deg_X(k) = \deg_Y(k) = t, \quad k(0, 0) = 1, \ \text{ and } k(X, \alpha_j) = 0 \ \text{ for all } P_j \in \mathcal{A}_{t,n}$$

任意の $\hat{s}$ に対して $\hat{F}$ を

$$\hat{F}(X, Y) = F(X, Y) + (\hat{s} - s)k(X, Y) \tag{5}$$

と定義すると、

$$f_j(X) = F(X, \alpha_j) = \hat{F}(X, \alpha_j) = \hat{f}_j(X) \qquad \text{for all } P_j \in \mathcal{A}_{t,n}$$

が成り立つ。$\mathcal{A}_{t,n}$ からみると、参加者間で多項式 $f(X)$ を補間している場合と、$\hat{f}(X)$ を補間している場合に全く情報の違いがない。実際、$\hat{F}(X, Y)$ を $F(X, Y)$ を $F(0, 0) = s$ の条件のもとランダムに選んでから、(5) のように計算したものは、$(\hat{s} - s)k(X, Y)$ の定数項以外は、$F(X, Y)$ の係数で one-time pad された状態

Theorem 6 の証明 II

$$\hat{F}(X,Y) = \boldsymbol{x}^T \Big(r_{ij} + (\hat{s} - s) k_{ij} \Big) \boldsymbol{y} \quad \text{where } r_{00} = s,\ k_{00} = 1,\ r_{ij} = r_{ji},\ \text{and } k_{ij} = k_{ji}.$$

になっているため、最初から $\hat{F}(X,Y)$ を、$\hat{F}(0,0) = \hat{s}$ かつ $\hat{F}(X,\alpha_j) = f_j(X)$ $(P_j \in \mathcal{A}_{t,n})$ の条件のもと二変数対称多項式としてランダムに選んだものと同分布である。
よって、統計的にも $\mathcal{A}_{t,n}$ が新たに得られる情報はない。 □

Reed-Solomon 符号

$(n, t+1)$-線型符号とは、n 次元ベクトル空間の $t+1$ 次元部分空間である。$K = \mathbb{F}_q$（元の個数が q 個の有限体）とする。$\alpha_1, \ldots, \alpha_n \in K$ $(\alpha_i \neq \alpha_j \text{ for } i \neq j)$ とする。

Reed-Solomon (RS) 符号

$f \in K[X]$, $\deg(f) \leq t$ となる多項式から作られる次の符号語

$$(f(\alpha_1), \ldots, f(\alpha_n))$$

の全体からなる集合を、$(n, t+1)$-Reed-Solomon 符号と言う。

$$\text{i.e., } C^{\mathrm{RS}} = \{(f(\alpha_1), \ldots, f(\alpha_n)) \mid f \in K[X] \text{ s.t. } \deg(f) \leq t\}.$$

$(n, t+1)$-RS 符号は $(n, t+1)$-線型符号

線型符号

線型符号のこと

- 線型符号 C は、ベクトル空間の部分ベクトル空間であるので、

$$\boldsymbol{c}, \boldsymbol{c}' \in C, \quad a, b \in K \qquad \Longrightarrow \qquad a\boldsymbol{c} + b\boldsymbol{c}' \in C.$$

  つまり、符号語の線形和は符号語。
- 任意の相異なる符号語 $\boldsymbol{c}$, $\boldsymbol{c}'$ と（ハミング）重み τ 以下の任意の（エラー）ベクトル $\boldsymbol{e}$, $\boldsymbol{e}'$ に対して、必ず

$$\boldsymbol{c} + \boldsymbol{e} \neq \boldsymbol{c}' + \boldsymbol{e}'$$

  なら、τ 個誤り訂正可能 (τ-error correcting) という。
- 線型符号の最小距離 d は、非零の最小の符号語のハミング重みに等しい
- 線型符号誤り訂正可能（最大）数は、$\tau < \frac{d}{2}$ の最大値.

- $\boldsymbol{v} \in K^n$ のハミング重み: $w_H(\boldsymbol{v}) \triangleq \#$ of $\{v_i \neq 0$ for $\boldsymbol{v} = (v_1, \ldots, v_n)\}$.
- $\boldsymbol{v}$ と $\boldsymbol{w}$ 間のハミング距離: $d_H(\boldsymbol{v}, \boldsymbol{w}) \triangleq w_H(\boldsymbol{v} - \boldsymbol{w})$.
- 符号の最小距離: 二つの符号間のハミング距離の最小値.
  $d = \min\{d_H(\boldsymbol{c}, \boldsymbol{c}') \,|\, \forall \boldsymbol{c}, \boldsymbol{c}' \in C \text{ s.t. } \boldsymbol{c} \neq \boldsymbol{c}'\}$

RS-符号

Theorem 7

$(n, t+1)$-*Reed-Solomon* 符号の最小距離は $n-t$ （$=(n, t+1)$-線型符号の最小距離の上界)。ここから誤り訂正可能最大数は $\tau < \frac{n-t}{2}$ なる τ の最大値。

Proof.

RS 符号語が $\boldsymbol{f} = (f(\alpha_1), \ldots, f(\alpha_n))$ であることを頭に入れ、

$$\text{非零の符号 } \boldsymbol{f} \in C^{\mathrm{RS}} \iff f(x) = 0 \text{ の零点が } t \text{ 個以下}$$

何故ならば、$\deg(f) \leq t$ より $t+1$ 点以上の零点があると $f \equiv 0$. よって、

$$f(x) = 0 \text{ の零点が } t \text{ 個以下} \iff w_H(\boldsymbol{f}) \geq n - t$$

これから、最小距離 $d = n - t$ が導かれる。さらに、$w_H(\boldsymbol{e}) < n - t$ なる $\boldsymbol{e}$ が、$\boldsymbol{e} \notin C^{\mathrm{RS}}$ ということは、

$$\forall \boldsymbol{c}, \boldsymbol{c'} \in C^{\mathrm{RS}},\ \forall \boldsymbol{e}, \boldsymbol{e'} (w_H(\boldsymbol{e}), w_H(\boldsymbol{e'}) < n - t) \implies \boldsymbol{c} + \boldsymbol{e} \neq \boldsymbol{c'} + \boldsymbol{e'}.$$

□

RS-符号の復号（その１）

Shamir SS の不正者が 1/3 未満の場合の Reconstruction の具体的手法に相当。

- 符号語 $\boldsymbol{f} = (f(\alpha_1), \ldots, f(\alpha_n))$ where $\deg(f) \leq t$ and $n \geq 3t+1$.
- エラー付き $\hat{\boldsymbol{f}} = (y_1, \ldots, y_n) = \boldsymbol{f} + \boldsymbol{e}$ where $W_H(\boldsymbol{e}) \leq t$
- Goal: $\hat{\boldsymbol{f}}$ から $\boldsymbol{f}$ に復号する

Welch/Berlekamp Algorithm

- $Q(X, Y) \triangleq f_0(X) - f_1(X)Y$
- $Q(\alpha_i, y_i) = 0$ for $i = 1, \ldots, n$
- $\deg(f_0) \leq 2t$ and $\deg(f_1) \leq t$
- $f_1(0) = 1$

となる $f_0(X)$, $f_1(X)$ を求め、$f(X) = \frac{f_0(X)}{f_1(X)}$ として出力。

$W_H(\boldsymbol{e}) \leq t$ かつ、$n \geq 3t+1$ ならこのような f_0, f_1 は存在し、さらに $f(X) = \frac{f_0(X)}{f_1(X)}$ を満たす。

RS-符号の復号（その２）

f_0, f_1 の存在証明.

$y_i = f(\alpha_i) + e_i$ で、$e_i \neq 0$ は高々 t 個なので、f_0, f_1 は次のように存在する。
$B = \{i \mid e_i \neq 0\}$ とおく。 定義より、$\#B \leq t$.

$$k(X) = \frac{\prod_{i \in B}(X - \alpha_i)}{\prod_{i \in B}(-\alpha_i)}$$

とおく。

- $f_0(X) = k(X)f(X)$,
- $f_1(X) = k(X)$

とおくと、$f_1(0) = 1$, $\deg(f_0) \leq 2t$, $\deg(f_1) \leq t$. さらに、$i = 1, \ldots, n$ に対して、

$$Q(\alpha_i, y_i) = f_0(\alpha_i) - f_1(\alpha_i)y_i = k(\alpha_i)\Big(f(\alpha_i) - y_i\Big) = 0.$$

よって、f_0, f_1 は存在。

RS-符号の復号（その３）

- $f_0(X) = a_0 + a_1 X + \ldots\ldots + a_{2t} X^{2t}$
- $f_1(X) = 1 + b_1 X + \ldots + b_t X^t$

$Q(\alpha_i, y_i) = f_0(\alpha_i) - f_1(\alpha_i) y_i = 0$ for $i = 1, \ldots, n$ から、以下の未知変数 $3t + 1$ の連立一次方程式を解く。

$$\begin{pmatrix} 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & \alpha_1 & \cdots & \cdots & \alpha_1^{2t} & -y_1 & -y_1\alpha_1 & \cdots & -y_1\alpha_1^t \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha_n & \cdots & \cdots & \alpha_n^{2t} & -y_1 & -y_n\alpha_n & \cdots & -y_n\alpha_n^t \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_{2t} \\ 1 \\ b_1 \\ \vdots \\ b_t \end{pmatrix}$$

RS-符号の復号（その４）

- $\hat{Q}(X) = Q(X, f(X))$ と定義する。すると $\deg(\hat{Q}) \leq 2t$.
- $W_H(\boldsymbol{e}) \leq t$ より、$y_i = f(\alpha_i)$ となる点が少なくとも $n - t$ 個ある。
- よって、$n - t$ 個の (α_i) で、$\hat{Q}(\alpha_i) = 0$.
- $n \geq 3t + 1$ より、$\deg(\hat{Q}) \leq 2t < n - t$. よって、

$$\deg(\hat{Q}) \equiv 0.$$

- よって、

$$f_0(X) - f_1(X) f(X) \equiv 0.$$

- よって、

$$f(X) = \frac{f_0(X)}{f_1(X)}.$$