Threshold Secret Sharing

Gilad Asharov
Bar-Ilan University

Center for Research in Applied Cryptography and Cyber Security

Agenda

- Secret Sharing Schemes
  - Shamir's Secret Sharing

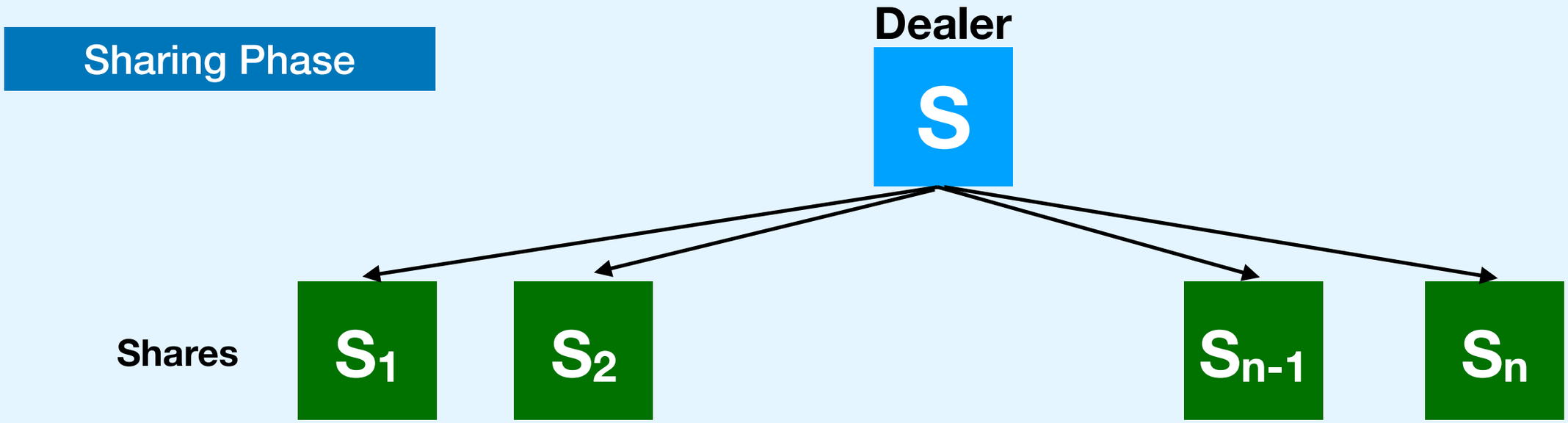
- Dealing with corrupted parties in reconstruction
  - Error correcting codes
  - Noisy decoding

- Verifiable Secret Sharing
  - Dealing with a corrupted dealer

Secret Sharing

- Suppose that we have sensitive information
  - Missile launch codes
  - A secret key for my crypto-wallet
  - A sensitive database
- We do not want to put all our eggs in one basket
  - A single point of failure
- We want to split the trust!

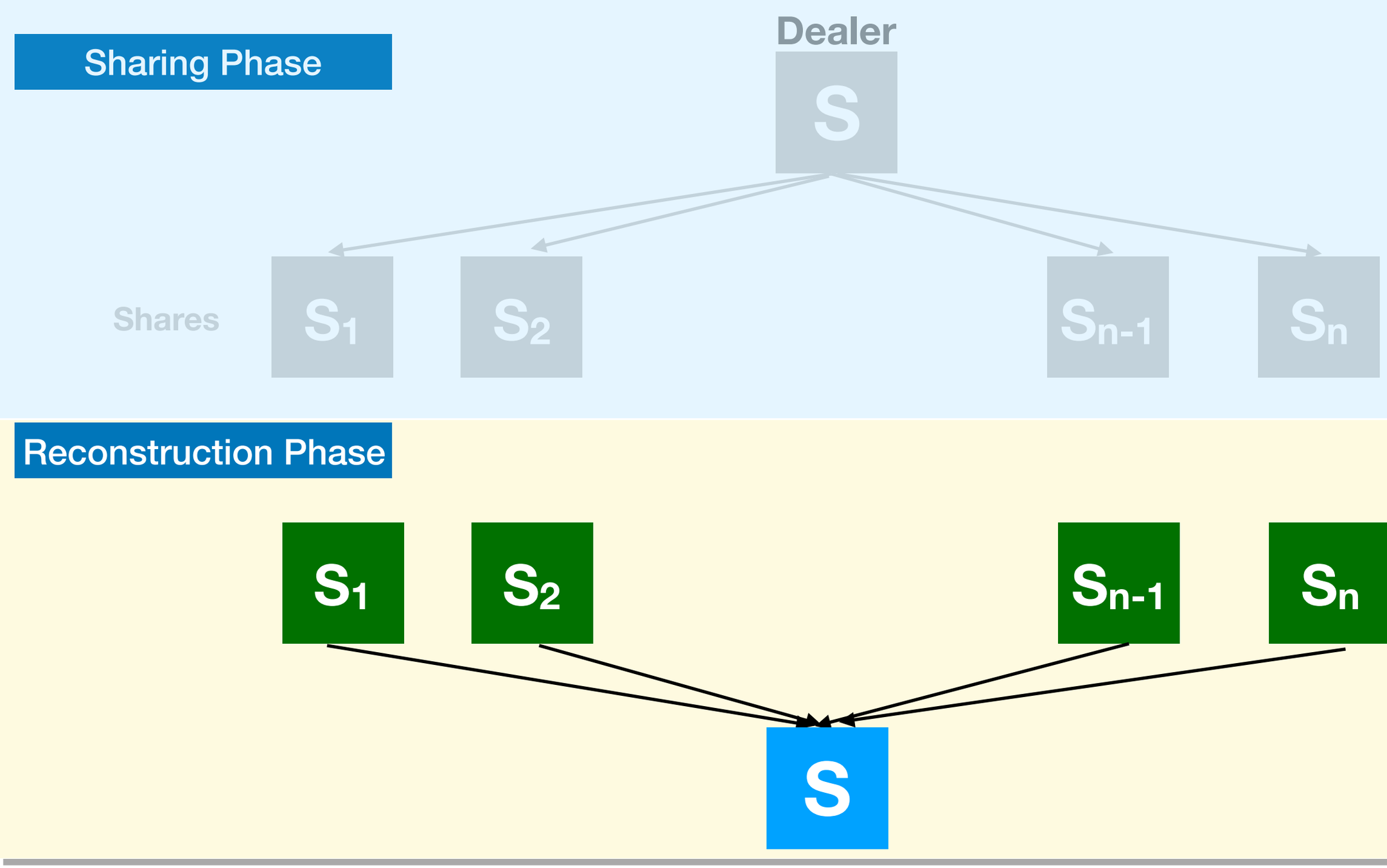
Secret Sharing



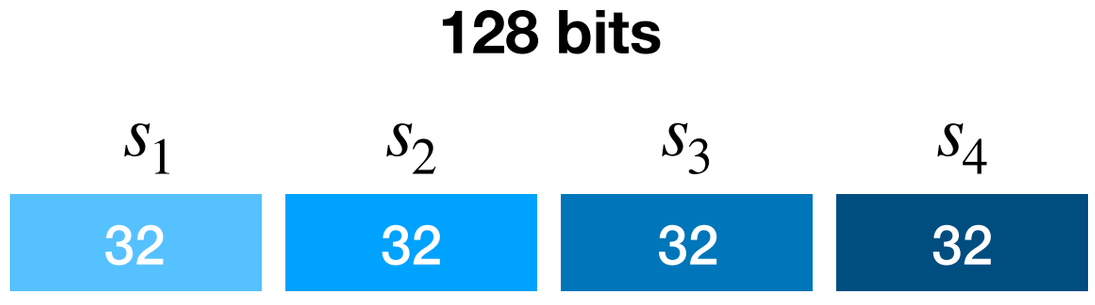
Privacy: Each share does not provide any information about the secret **s**

(Even **subset** of shares do not provide any information about **s**)

Secret Sharing



Secret Sharing



- To share a secret s among n parties?
  - Say s is 128-bit secret key, $n = 4$
  - Give each party 32 bits
  - What does each party know about the key?
  - What do 3 parties know about the secret?

A Better Idea

- Sharing(s, n): (with $n = 4$, $s \in \{0,1\}^{128}$)
  - Choose: $s_1, s_2, s_3 \leftarrow \{0,1\}^{128}$
  - Set: $s_4 := s - (s_1 + s_2 + s_3) \leftarrow \mod 2^{128}$
  - Party P_i receives share s_i

- Reconstruction(s_1, s_2, s_3, s_4):
  - Output: $\sum_{i=1}^{4} s_i \mod 2^{128} = s$

- What do 3 parties know about the secret?

Notations

- S : the domain of the secret (e.g., $\{0,1\}^{128}$)
- n : the number of parties
- t: threshold
  - $t+1$ parties can reconstruct the secret
  - t parties cannot learn anything about the secret
  - This is "$t+1$-out-of-n" secret sharing

- A secret sharing scheme consists of a pair of functions (computable in poly$(n, \log|\mathsf{S}|)$:
  - Dealing function: Sharing
  - Recovery function: Reconstruction

Syntax and Requirements

- Sharing:

$$\text{Sharing}(s; r) \rightarrow (s_1, \ldots, s_n),$$

where $s \in \mathsf{S}$ is the secret, r is the randomness,
and s_i is the share of the i-th party

- $\text{Reconstruction}(s_{i_1}, \ldots, s_{i_{t+1}}) \rightarrow s'$

- **Correctness**:
  For every set of parties $A \subseteq [n]$ with $|A| \geq t + 1$:

$$\text{Reconstruction}(\{\text{Sharing}(s)\}_A) \;=\; s$$

- t**-privacy**: The distribution of any t shares is independent of the secret s
  - ***Formally***: for any pair of secrets $s, s' \in \mathsf{S}$, and for any $I \subseteq [n]$ with $|I| \leq t$

$$\left\{\{s_i\}_{i\in I} \,|\, (s_1, \ldots, s_n) \leftarrow \text{Sharing}(s)\right\} \equiv \left\{\{s'_i\}_{i\in I} \,|\, (s'_1, \ldots, s'_n) \leftarrow \text{Sharing}(s')\right\}$$

n-out-of-n Secret Sharing

- We already saw a construction of n-out-of-n secret sharing:
- Assume that S is a group with operation +
  - For example, $S = \mathbb{Z}_m$
- Sharing(s):
  - Choose $s_1, \ldots, s_{n-1}$ uniformly and independently at random from S
  - Set $s_n := s - (s_1 + \ldots + s_{n-1})$
  - Output $(s_1, \ldots, s_n)$
- Reconstruction($s_1, \ldots, s_n$):
  - Output: $s_1 + \ldots + s_n = s$
- Efficiency? $\| s_i \| = \| s \|$ for every i

What about $t < n$?

- **Sharing**:
  For every authorized subset $A \subseteq [n]$ of size $t+1$, use the $t+1$-out-of-$t+1$ secret sharing scheme solution to share s
- **Reconstruction**:
  An authorized set of parties can reconstruct s
- **Efficiency**:
  Each party receives $\binom{n}{t+1}$ shares of size $\| s \|$ each
- **Exponential** when, e.g., $t = n/2$
- We can do better…

Shamir's Secret Sharing

- This time we need a richer algebraic structure: **a field**
  - E.g., $\mathbb{Z}_p$ is a field when p is a prime
    - $(\mathbb{Z}_p, +_p)$ is a commutative (additive) group,
      - $(\mathbb{Z}_p \backslash \{0\}, \cdot_p)$ is a commutative (multiplicative) group
- In general, we will denote the field as $\mathbb{F}$
- Let $S = \mathbb{F}$ and assume that $|\mathbb{F}| > n$
- Let $\alpha_1, \ldots, \alpha_n$ be distinct non-zero elements in $\mathbb{F}$

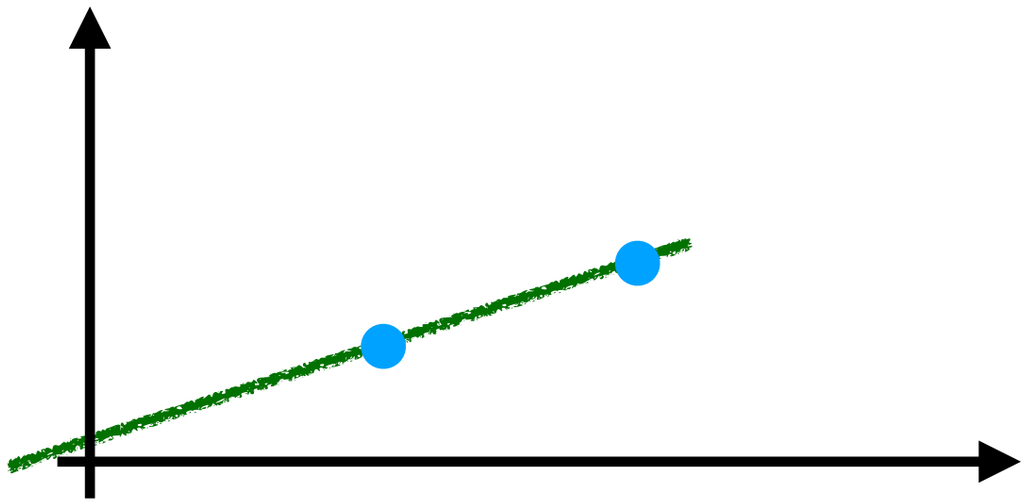
Shamir's Secret Sharing

- Sharing$_{t+1,n}(s)$:
  - Choose a random degree t polynomial with s as its constant term
  - $p(x) = s + p_1 x + \ldots, p_t x^t$
  - Party P_i receives $(\alpha_i, p(\alpha_i))$

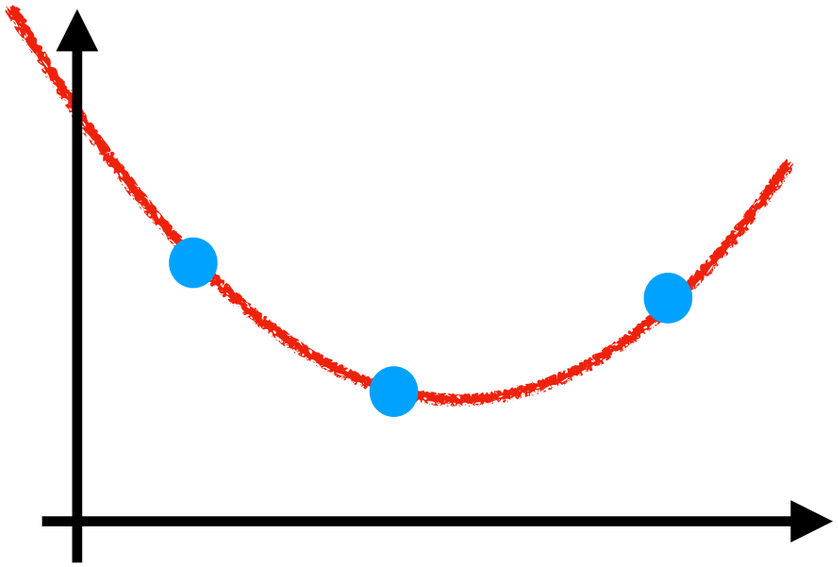
- **Properties**:
  - Every set of $t+1$ participants can **recover** the secret
  - Every set of t shares **does not reveal** any information about s
    - **Even stronger**: every t shares are uniformly distributed in $\mathbb{F}$
  - Size of each share: $\|\mathbb{F}\|$

How to Reconstruct?

2 (distinct) points determines exactly one degree-1 polynomial



3 (distinct) points determines exactly one degree-2 polynomial



In general:

$t+1$ (distinct) points determines exactly one degree t polynomial

Reconstruction - Lagrange Interpolation

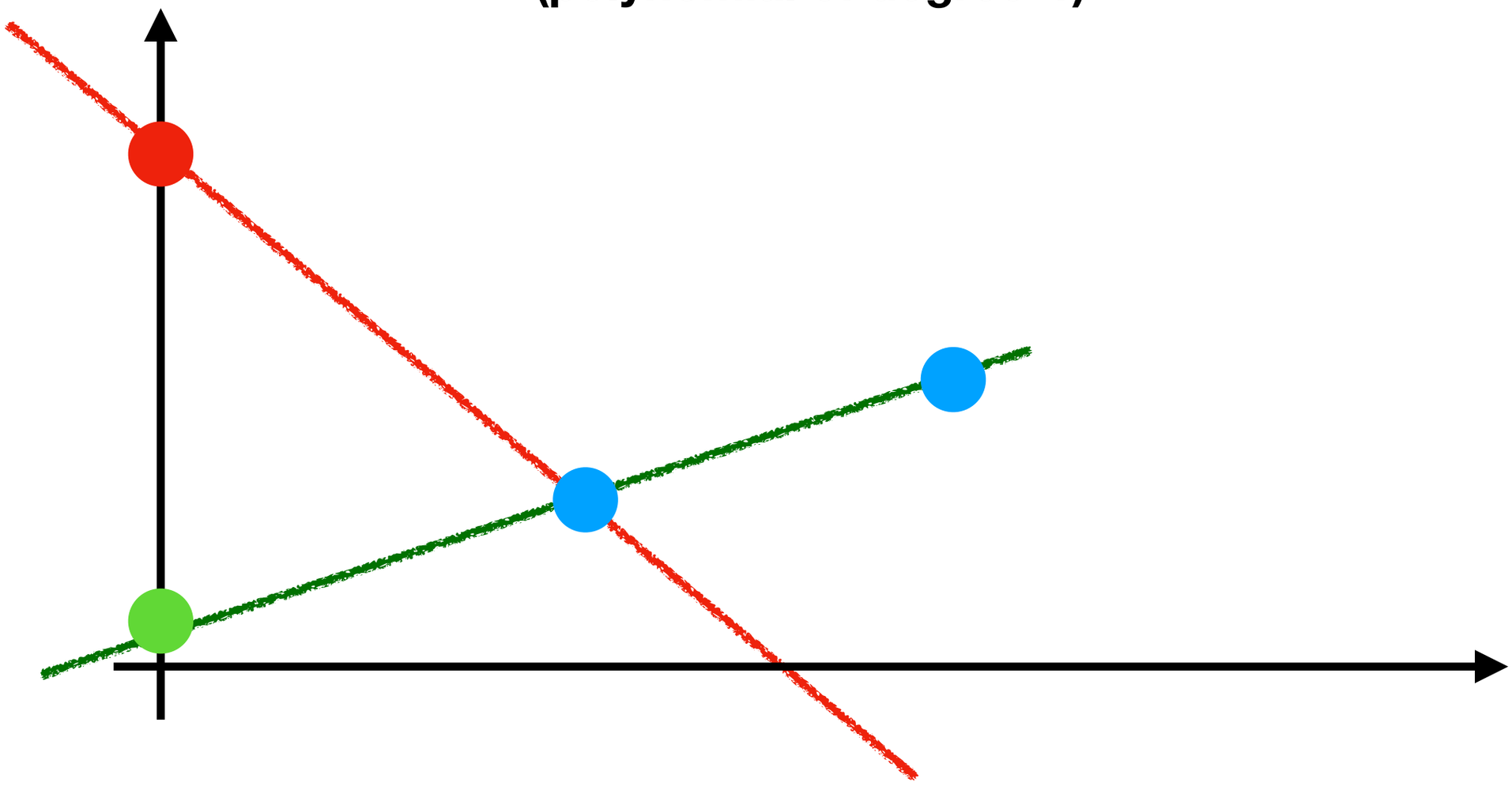
- **Input**: $(\alpha_1, y_1), \ldots, (\alpha_{t+1}, y_{t+1}) = ((\alpha_1, p(\alpha_1)), \ldots, (\alpha_{t+1}, p(\alpha_{t+1})))$

$$f_1(x) := y_1 \cdot \frac{x - \alpha_2}{\alpha_1 - \alpha_2} \cdot \frac{x - \alpha_3}{\alpha_1 - \alpha_3} \cdot \ldots \cdot \frac{x - \alpha_{t+1}}{\alpha_1 - \alpha_{t+1}}$$

- **Q**: What is the degree of $f_1(x)$?
- **Q**: What is $f_1(\alpha_1)$? What are $f_1(\alpha_2), \ldots, f_1(\alpha_{t+1})$?
  - **A**: $\deg(f_1(x)) = t, f_1(\alpha_1) = y_1, f_1(\alpha_2) = \ldots = f_1(\alpha_{t+1}) = 0$
- We can define $f_2(x), \ldots, f_{t+1}(x)$ analogously, and set $f(x) := f_1(x) + \ldots + f_{t+1}(x)$
- **Exercise**: prove that $f(x) = p(x)$
  - $f(x)$ is a degree t polynomial, $f(\alpha_1) = y_1, \ldots, f(\alpha_{t+1}) = y_{t+1}$
  - The polynomial $f(x) - p(x)$ if a non-zero degree t polynomial with $t + 1$ distinct roots - impossible

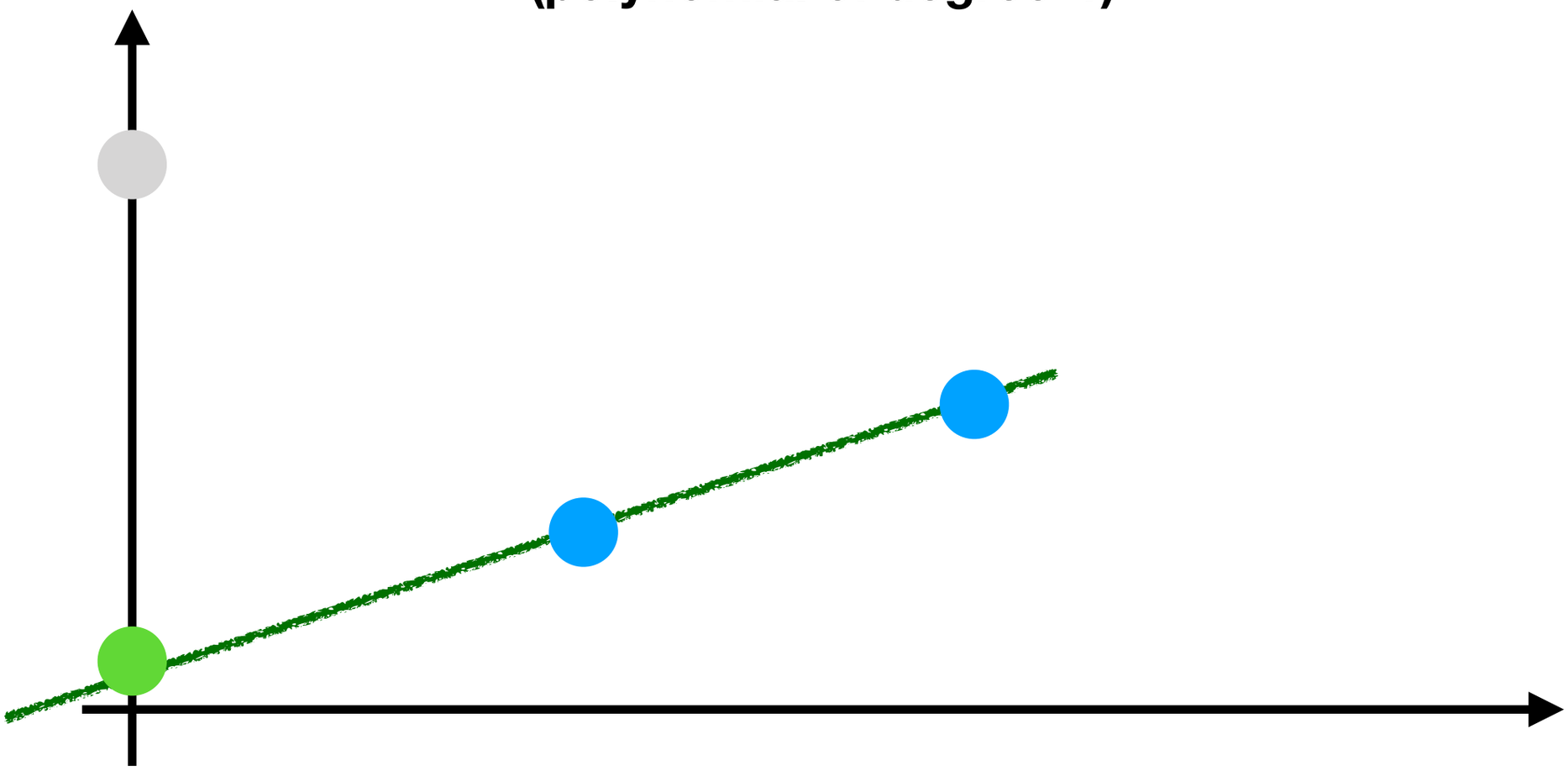
Security - Intuition

2-out-of-2 secret sharing
(polynomial of degree-1)



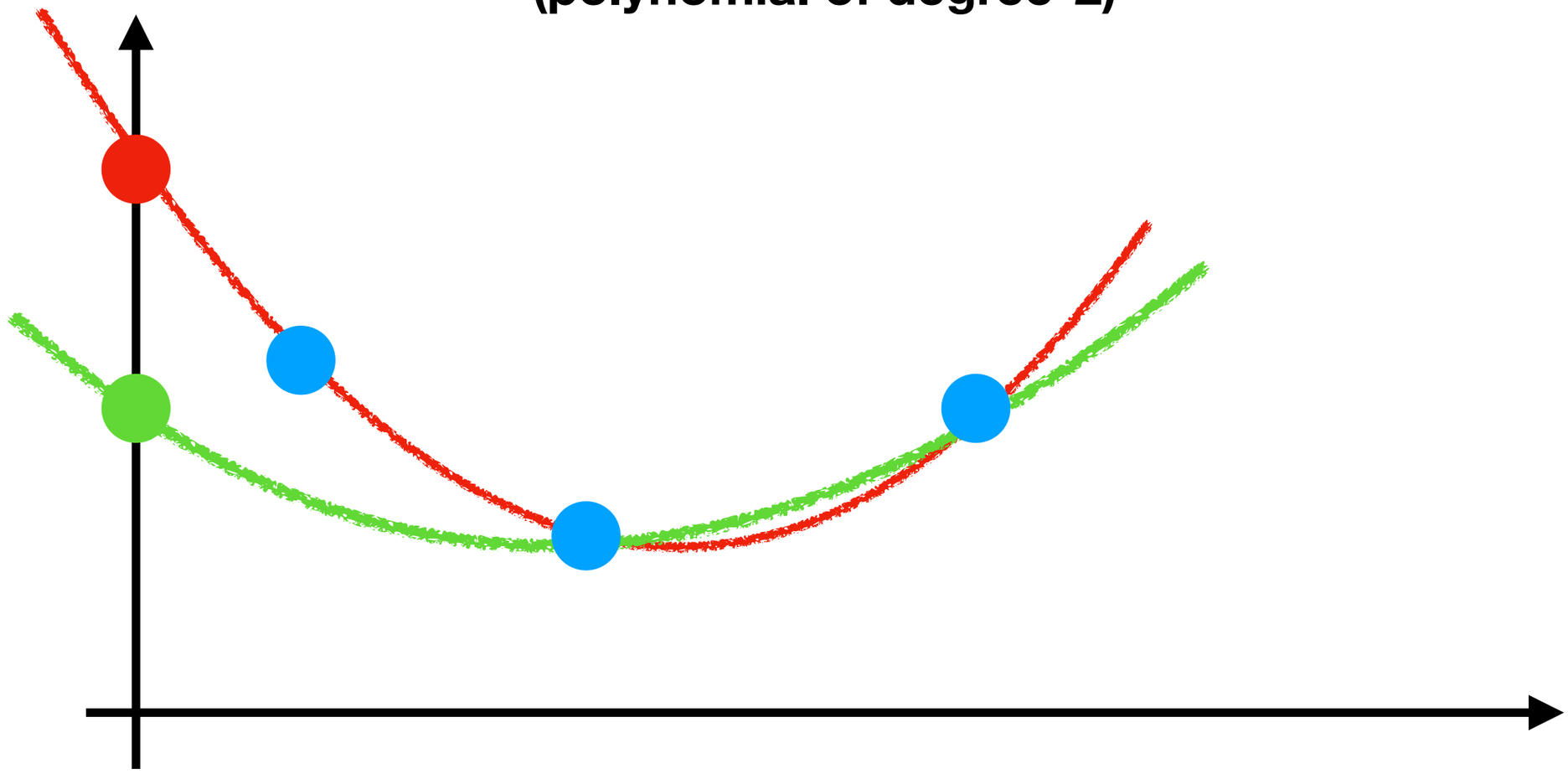
Security - Intuition

2-out-of-2 secret sharing
(polynomial of degree-1)



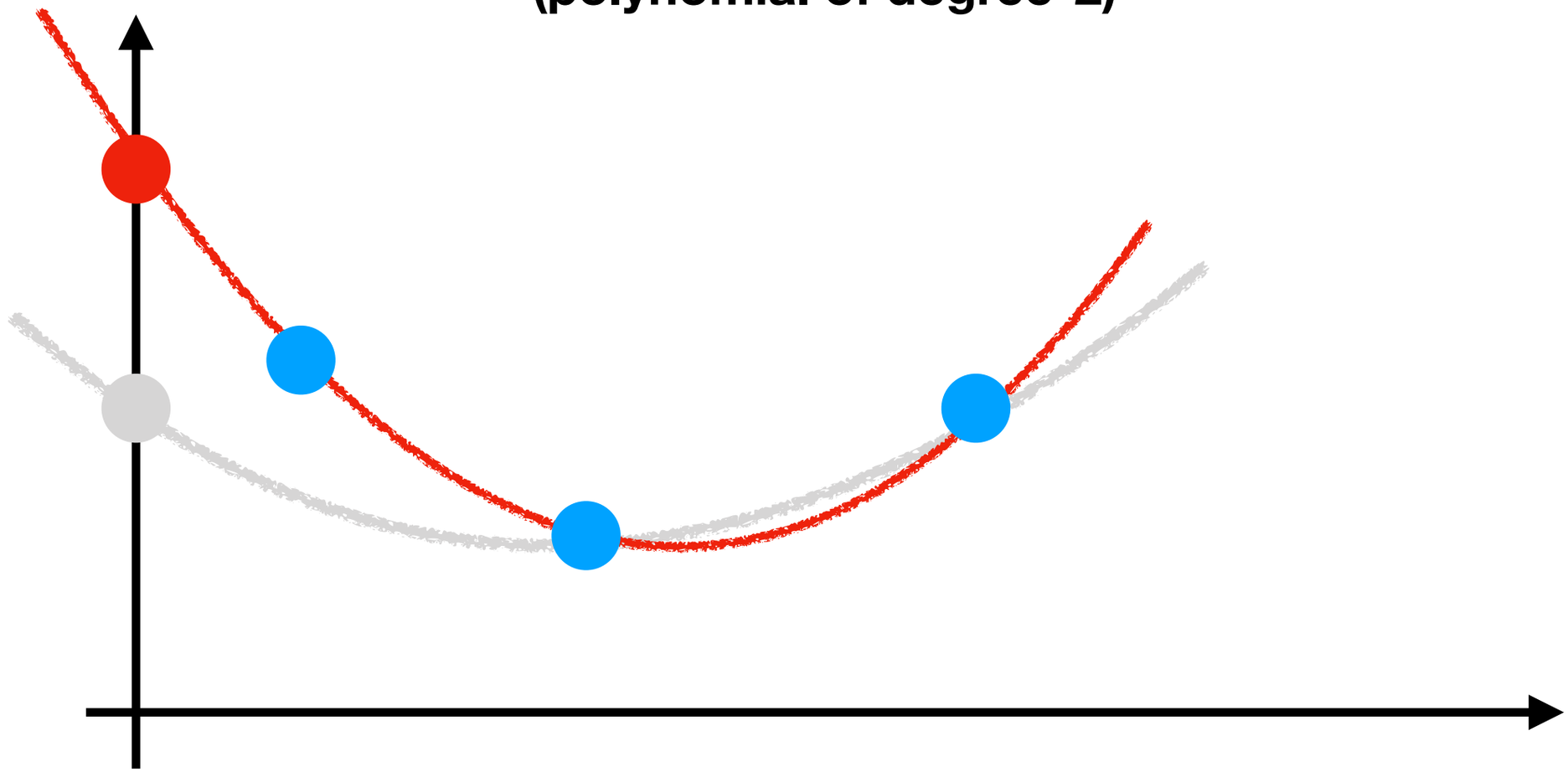
Another Example

3-out-of-4 secret sharing
(polynomial of degree-2)



Another Example

3-out-of-4 secret sharing
(polynomial of degree-2)



Security - Formally

- $P^{s,t}$: the set of all polynomials over $\mathbb{F}$ with degree t and constant term s
- $|P^{s,t}| = |\mathbb{F}|^t$

- Sharing$(s, n, t+1)$:
  Choose a random polynomial $p(x) \leftarrow P^{s,t}$ and output $\big(p(\alpha_1), \ldots, p(\alpha_n)\big)$

Security - Formally

- **Claim**: Every t shares are distributed uniformly at random in $\mathbb{F}$
- Fix any t values $(y_1, \ldots, y_t) \in \mathbb{F}$, subset $I \subset [n], |I| = t$
  - There is exactly one polynomial $p(x) \in P^{s,t}$ that satisfies $p(0) = s$ and $p(\alpha_i) = y_i$ for every $i \in I$
    - The probability that $\mathsf{Sharing}(s, n, t+1)$ chose $p(x)$ is $1/|\mathbb{F}|^t$
  - There is exactly one polynomial $q(x) \in P^{s',t}$ that satisfies $q(0) = s'$ and $q(\alpha_i) = y_i$ for every $i \in I$
    - The probability that $\mathsf{Sharing}(s', n, t+1)$ chose $q(x)$ is $1/|\mathbb{F}|^t$

$$\Pr_{p(x) \leftarrow P^{s,t}}[p(\alpha_1) = y_1, \ldots, p(\alpha_t) = y_t] = \Pr_{q(x) \leftarrow P^{s',t}}\left[q(\alpha_1) = y_1, \ldots, q(\alpha_t) = y_t\right] = \frac{1}{|\mathbb{F}|^t}$$

Application: Key Recovery

- It is hard to remember a secret key k
- So remember a password p and store $H(p) \oplus k$
  - What if I forget the password?
- 3-out-of-5 **secret sharing**:
  - Share k to $s_1, \ldots, s_5$
  - The user selects 5 personal questions in which he knows answers $a_1, \ldots, a_5$
  - Store $H(a_1) \oplus s_1, \ldots, H(a_5) \oplus s_5$
  - The user can recover if he remembers at least 3 answers

Application: Robust Combiner

- We have 3 encryption schemes Π_1, Π_2, Π_3, each $\Pi_i = (\mathsf{KeyGen}_i, \mathsf{Enc}_i, \mathsf{Dec}_i)$
  - We do not know which one is actually secure
- Can we construct a new scheme that is secure iff at least one of Π_1, Π_2, Π_3 is secure?
- **Yes**:
  - $\mathsf{Enc}(k, m)$:
    - Secret share m into m_1, m_2, m_3 (3-out-of-3)
    - Encrypt m_i using Π_i
  - The message is protected if at least one of Π_1, Π_2, Π_3 is secure!

Linear Secret Sharing

- The secret is an element in the field
- The randomness of Sharing is a vector of random elements in the field
- The share of each party is some vector
  - Each one of its coordinates is some fixed linear combination of the secret and the randomness

- Is Shamir's scheme a linear secret sharing scheme?

Shamir Secret Sharing: Matrix View

$$\begin{pmatrix} 1 & \alpha_1 & \alpha_1^2 & \dots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \alpha_2^2 & \dots & \alpha_2^{n-1} \\ \vdots & & \ddots & & \vdots \\ 1 & \alpha_{n-1} & \alpha_{n-1}^2 & \dots & \alpha_{n-1}^{n-1} \\ 1 & \alpha_n & \alpha_n^2 & \dots & \alpha_n^{n-1} \end{pmatrix} \begin{pmatrix} s \\ p_1 \\ \vdots \\ p_t \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} p(\alpha_1) \\ p(\alpha_2) \\ \vdots \\ \\ \\ \\ \vdots \\ p(\alpha_n) \end{pmatrix}$$

$$V_{\overrightarrow{\alpha}} \cdot \overrightarrow{p} = \overrightarrow{p(\alpha)}$$

$$V_{\vec{\alpha}}[i,j] = \alpha_i^{j-1} \qquad \det(V_\alpha) = \prod_{1 \le i \le j \le n} (\alpha_j - \alpha_i)$$

The Vandermonde $V_{\overrightarrow{\alpha}}$ has an inverse iff $\alpha_1, \dots, \alpha_n$ are distinct elements in $\mathbb{F}$

Shamir Secret Sharing: Matrix View

$$V_{\overrightarrow{\alpha}} \cdot \overrightarrow{p} = \overrightarrow{p(\alpha)}$$

$$\overrightarrow{p} = V^{-1}_{\overrightarrow{\alpha}} \cdot \overrightarrow{p(\alpha)}$$

$$\begin{pmatrix} s \\ p_1 \\ \vdots \\ p_t \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} \lambda_1 & \lambda_2 & \dots & \lambda_n \\ \ddots & & & \ddots \\ \dots & & & \dots \end{pmatrix} \begin{pmatrix} p(\alpha_1) \\ p(\alpha_2) \\ \vdots \\ \\ \\ \vdots \\ p(\alpha_n) \end{pmatrix}$$

s is a linear combination of the points $(p(\alpha_1), \dots, p(\alpha_n))$

$$s = \lambda_1 \cdot p(\alpha_1) + \dots + \lambda_n \cdot p(\alpha_n)$$

Linear Secret Sharing

- **Shamir is a linear secret sharing scheme!**
- Equivalent to **Monotone Span Programs**:
  - Linear algebraic model of computation
  - Introduced by Karchmer and Wigderson (1993)

- **Packed Secret Sharing:**
  - Can the shares hide more than one secret?

General Access Structure

- Who is authorized to reconstruct the secret?
  - Shamir is a "**threshold scheme**"
    - **Authorized sets**: all subsets of cardinality $t+1$
    - E.g. $n=4$, all subsets of 2 parties are authorized:
      $\Gamma = \{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$
      - What about $(1,2,3)$? $(2,3,4)$?
- What about general access structure?
  - E.g.: $\Gamma = \{(1,2,4),(1,3,4),(2,3)\}$?
- **Monotone access structure:**
  - If $A \in \Gamma$ and $A \subseteq B$ then $B \in \Gamma$

General Access Structure

[ItoSaitoNishizaki91]

$$\Gamma = \{(1,2,4), (1,3,4), (2,3)\}$$

Represent it as a CNF

$$\underbrace{(P_1 \wedge P_2 \wedge P_4)}_{s} \vee \underbrace{(P_1 \wedge P_3 \wedge P_4)}_{s} \vee \underbrace{(P_2 \wedge P_3)}_{s}$$

$$a_1 \oplus a_2 \oplus a_4 \qquad b_1 \oplus b_3 \oplus b_4 \qquad c_2 \oplus c_3$$

P_1	P_2	P_3	P_4
a_1 b_1	a_2 c_2	b_3 c_3	a_4 b_4

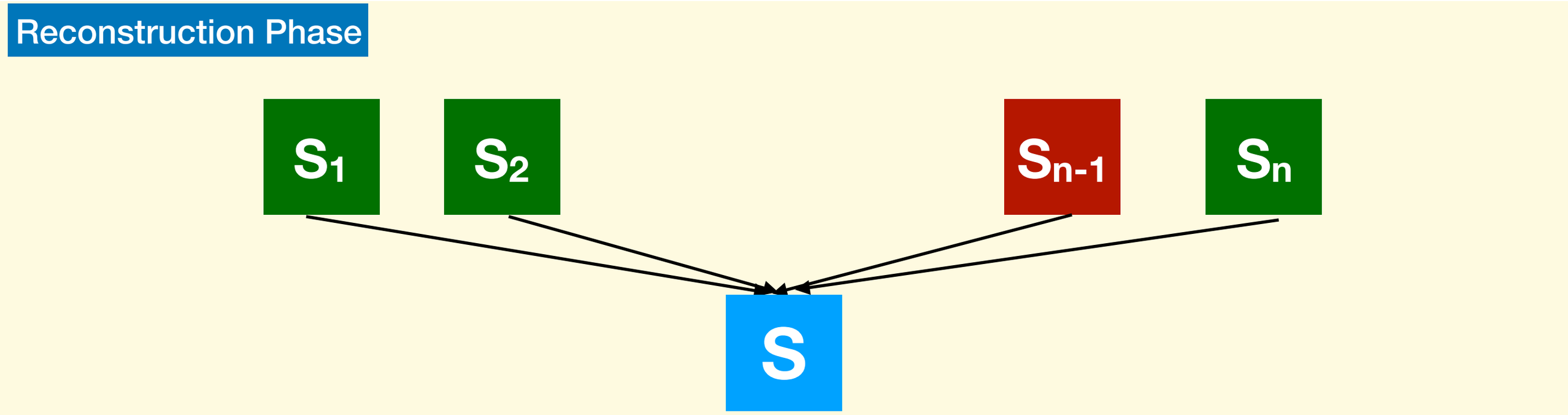
Additional Topics

- Consider threshold access structure with $t+1$
  - **Q:** What is the size of the shares if we are using Shamir's scheme?
  - **Q:** What is the size of the shares if we are using ISN?
  - **Fundamental question:** What is the optimal share size for a given access structure?

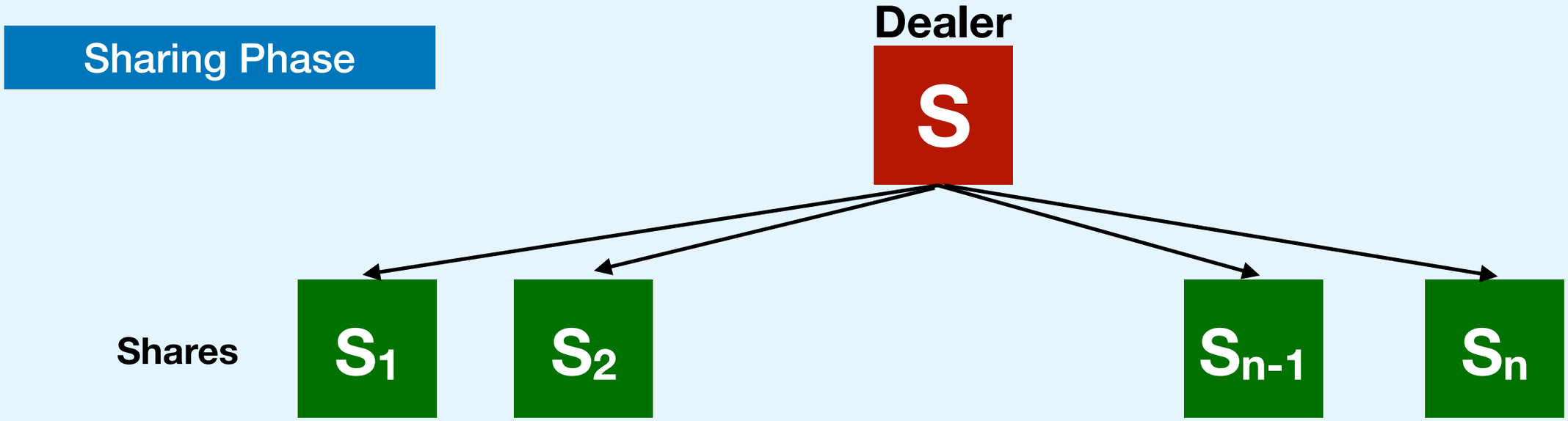
- **Exercise[BenalohLeichter]:** Construct a scheme where the access structure is represented as a circuit (AND/OR gates)
  - **C**(S)=1 iff the subset S is an authorized set
  - The fan-out is 1 (every wire is an input to one gate only)
  - CNF is a depth 2 circuit

What's Next?

What if some party is malicious and sends an incorrect share?
Can the honest party still reconstruct the secret?
How many wrong shares can they tolerate?



What's Next?



What if **the dealer** is malicious and sends shares that do not define points on a polynomial?
Can the honest parties detect it?

How to Deal with Wrong Shares in Reconstruction?

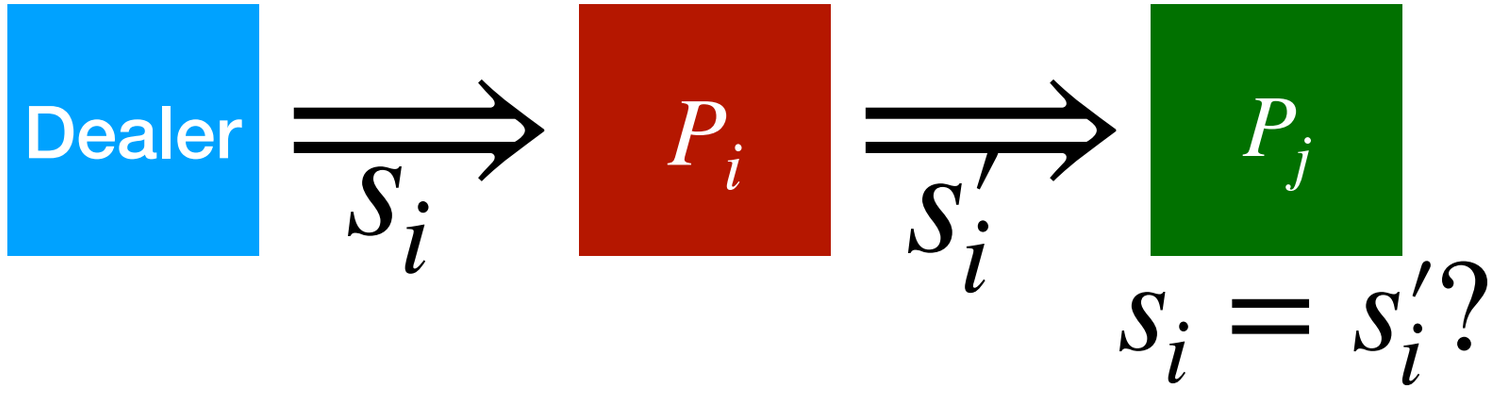
Robust Secret Sharing

BIU
Center for Research in Applied
Cryptography and Cyber Security

Warmup: Computational Setting

- Consider $t+1$-out-of-n secret sharing
  - At most t parties are corrupted
  - We assume that the dealer is **honest**
  - **The difficulty**: corrupted parties might send in the reconstruction phase different shares than those received by the dealer
- How can we do it in the Computational Setting?
  - The dealer *signs* on each share
  - In **reconstruction**, parties can verify whether the share is correct with the signature

Do We Really Need Signatures?



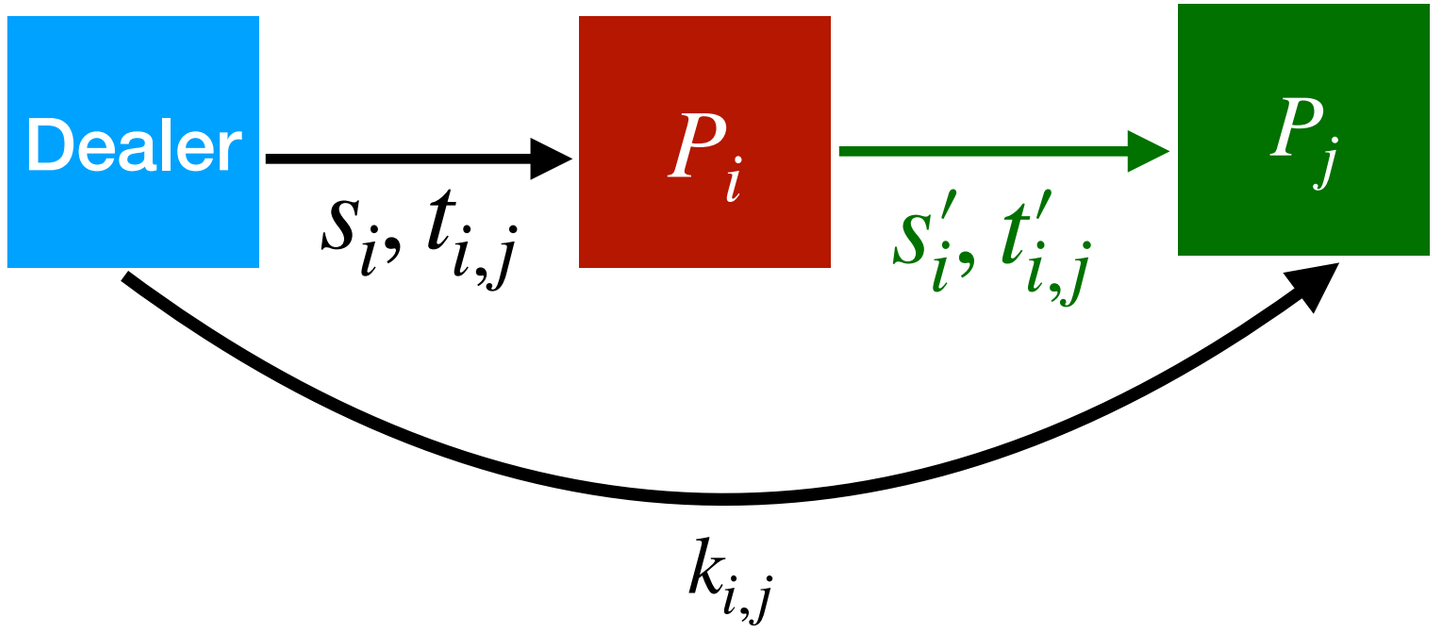
- We just want to "**authenticate**" the information provided by the dealer
- 1-time-MAC: $\Pi = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$
- One-time message authentication experiment $\mathsf{Mac\text{-}forge}^{\text{1-time}}_{A,\Pi}$:
  - $k \leftarrow \mathsf{Gen}(1^n)$
  - A outputs a message m' and is given in return a tag $t' \leftarrow \mathsf{Mac}_k(m')$
  - A outputs (m, t) and wins if $m \neq m'$ and $\mathsf{Vrfy}_k(m, t) = 1$

How to Construct IT 1-time-MAC?

- Gen(1^n) : (the security parameter relates to the size of the field $\mathbb{F}$)
  - Choose random elements $a, b \in \mathbb{F}$, set $k = (a, b)$
- $\text{Mac}_k(m)$:
  - Output $t := a \cdot m + b$
- $\text{Vrfy}_k(m, t)$: Output 1 iff $\text{Mac}_k(m) = t$
- **Security**:
  - The adversary sees a pair of $(m', t') = (m', am' + b)$ for m' of its choice
  - Can it find another point on this line?
  - **Exercise**: formalize it
  - Probability to win is $1/|\mathbb{F}|$

Robust Secret sharing

[RabinBenOr89]



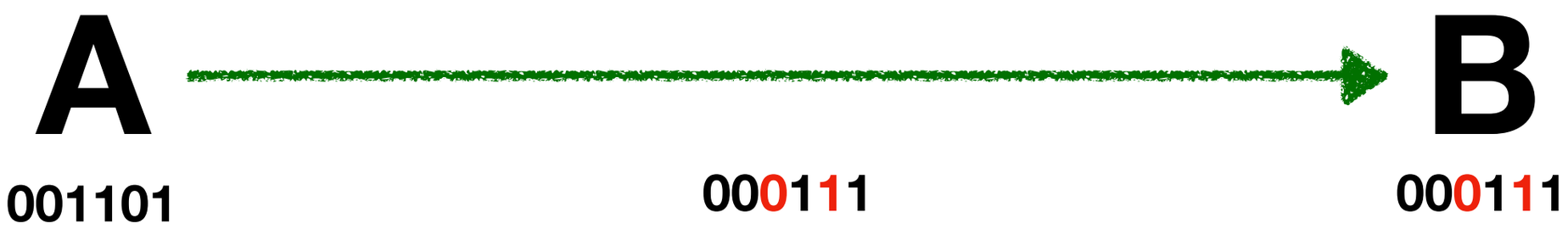
- P_j accepts s'_i iff $\mathsf{Vrfy}_{k_{i,j}}(s'_i, t'_{i,j}) = 1$

So What Do We Have So Far?

- **Computational setting:**
  - The dealer can provide extra information to aid verification
- **Information theoretic setting:**
  - The dealer can provide extra information to aid verification
  - Assuming large field, negligible failure probability

- **Can we do better?**
  - Can we recognize the wrong shares **without** extra information from the dealer?
  - Can we correct errors with 0-probability of failure?
  - **Yes**! Using error correcting codes

Noisy Channels



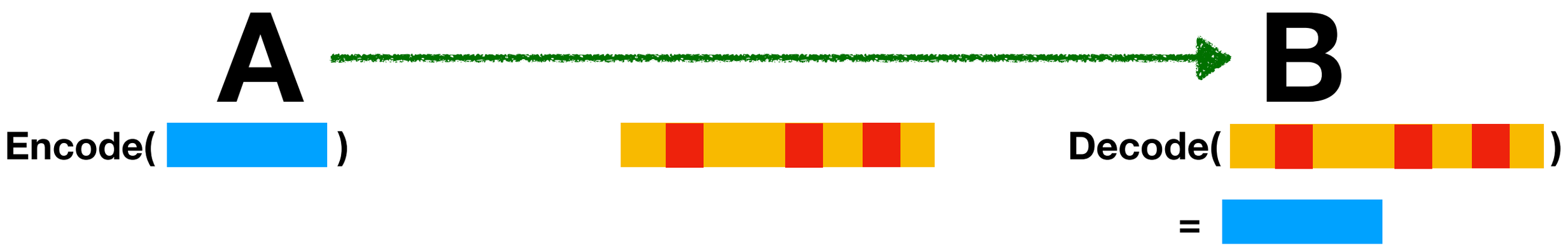
Flips a bit with some probability p

Goal: Reliable communication over a “noisy” channel

Error Correcting Codes

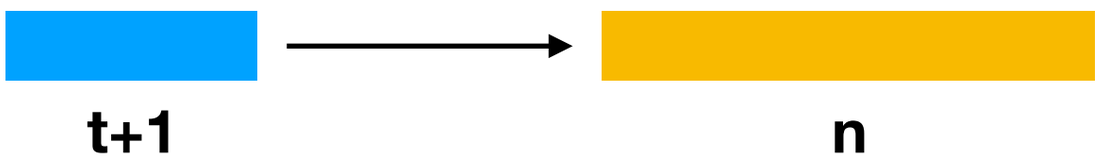
Correctness: Decode(Encode(m))=m

Error Correcting Codes

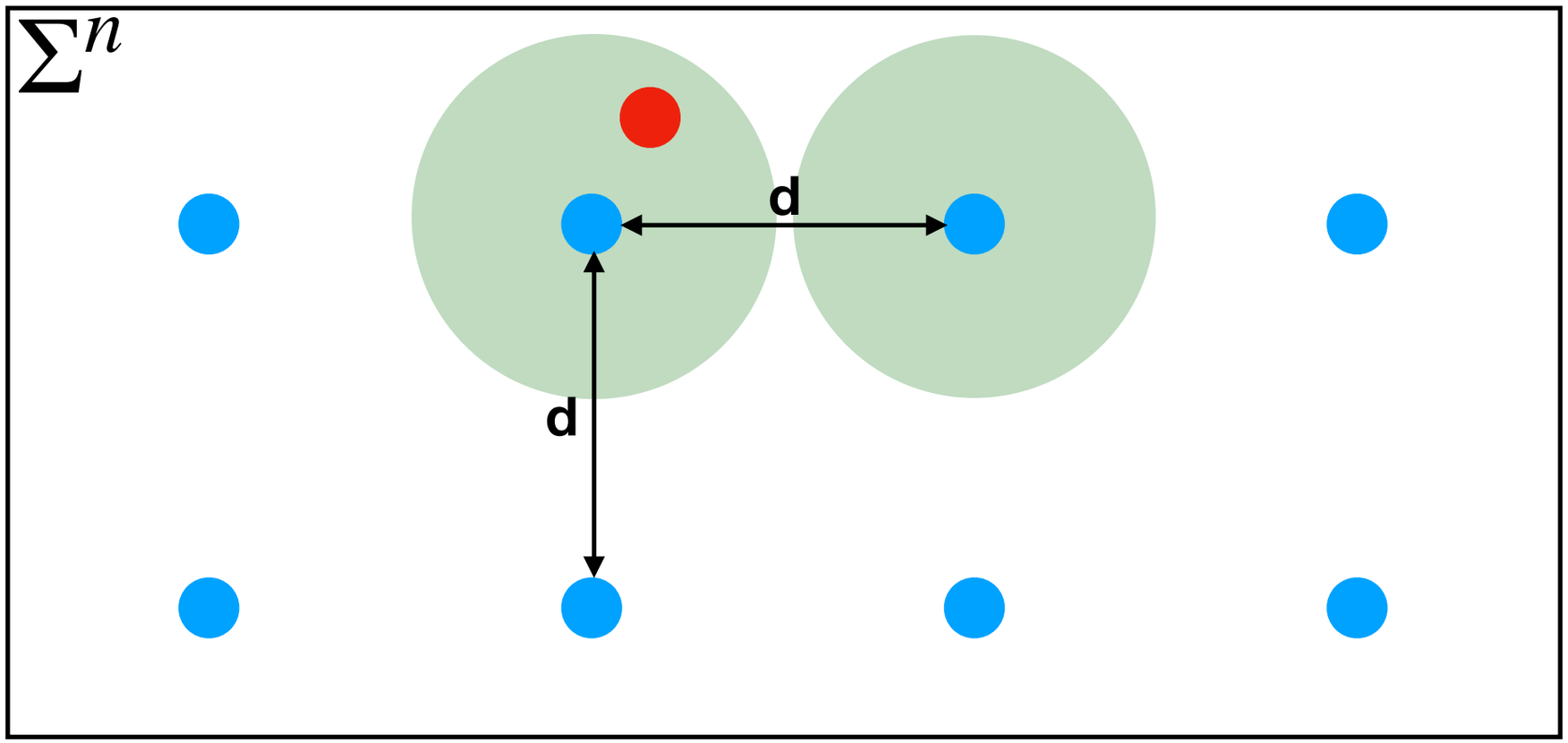


Correctness: Decode(Encode(m))=m

Even stronger: Decode(Encode(m)+”noise”)=m



(n,t+1,d)-Code



Encode : $\Sigma^{t+1} \rightarrow \Sigma^n$

$C = \{c \,|\, \exists m \in \Sigma^{t+1} \text{ s.t. Encode}(m) = c\} \subset \Sigma^n$

$d(C) = \min\{d(c_1, c_2) \,|\, c_1, c_2 \in C, c_1 \neq c_2\}$

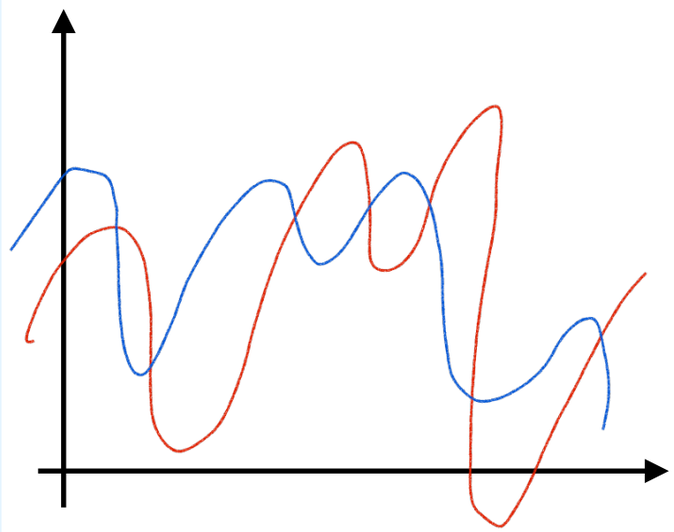
The decoding function will decode to the closest codeword

(theoretically, it is possible to support $(d-1)/2$ errors)

Decode : $\Sigma^n \rightarrow \Sigma^{t+1} \cup \{ \perp \}$

Reed-Solomon Code

- $\Sigma = \mathbb{F}$, fix some $\alpha_1, \ldots, \alpha_n$, distinct elements in $\mathbb{F}$
- Encode(m) : $m \in \mathbb{F}^{t+1}$
  - Given a message $m = (m_0, \ldots, m_t) \in \mathbb{F}^{t+1}$ define $p(x) := m_0 + m_1 x + \ldots + m_t x^t$
  - Output $(p(\alpha_1), \ldots, p(\alpha_n))$



- What is the minimum distance of this code?
  - Two **distinct** polynomials of degree t: $p(x), q(x)$
  - How many points might they agree on?
  - At most t
    - Consider the polynomial $h(x) := p(x) - q(x)$; it has at most t roots
  - Reed Solomon is $(n, t+1, n-t)$ code. Might correct $(n-t-1)/2$ errors

Reed Solomon Code and Secret Sharing

- Set n≥3t+1, we get (3t+1,t+1,2t+1)
  - **t** parties do not learn anything about the secret
  - We can tolerate $(d-1)/2 = t$ wrong shares during reconstruction
    - The ≥2t+1 honest parties can reconstruct the secret!
    - We assume here an honest dealer

How to Efficiently Reconstruct the Secret?

- Given n shares that were received, how can we recognize the wrong shares?
- One option:
  - We need only t+1 shares for reconstruction
  - Try all $\binom{n}{t+1}$ options and take the majority
    - $t \approx n/3$, and so this is not particularly efficient…

Noisy Decoding: Berlekamp-Welch Algorithm

- **Input**: $(\alpha_1, y_1), \ldots, (\alpha_n, y_n)$. At least $2t + 1$ of the y_is lie on the same degree-t polynomial $p(x)$
- **Goal**: Find that polynomial $p(x)$
- **The idea:**
  - Find a polynomial $E(x)$ of degree #errors, such that for every i:
    - $y_i \cdot E(\alpha_i) = p(\alpha_i) \cdot E(\alpha_i)$
    - Moreover:
      - $E(\alpha_i) = 0$ if $p(\alpha_i) \neq y_i$
      - $E(\alpha_i) \neq 0$ if $p(\alpha_i) = y_i$
- The algorithm will find a polynomial $Q(x)$ of degree-2t, and a polynomial $E(x)$ of degree at most t as above such that
  - $y_i \cdot E(\alpha_i) = Q(\alpha_i)$ for every i
- Output $P(x) = Q(x)/E(x)$

How to Find Q and E?

- We have $3t + 1$ equations of the form (for every α_i):

$$y_i(e_0 + e_1 \cdot \alpha_i \ldots + e_{t-1}\alpha_i^{t-1} + \alpha_i^t) - (q_0 + q_1\alpha_i + \ldots + q_{2t}\alpha_i^{2t}) = 0$$

- We have $(2t + 1) + t = 3t + 1$ unknowns $(e_0, e_1, \ldots, e_{t-1}), (q_0, \ldots, q_{2t})$

- Simply use linear algebra

Exercise / Example

- Reed Solomon with n=7, t=3, $\mathbb{F} = \mathbb{Z}_{929}$
- You are given
  $c + e = \{(000,001), (001,006), (002,123), (003,456), (004,057), (005,086), (006,121)\}$
- Find the message m
- Use: $y_i(e_0 + e_1\alpha_i) - (q_0 + q_1\alpha_i + q_2\alpha_i^2 + q_3\alpha_i^3 + q_4\alpha_i^4) = -y_i\alpha_i^2$

$$\begin{bmatrix} 001 & 000 & 928 & 000 & 000 & 000 & 000 \\ 006 & 006 & 928 & 928 & 928 & 928 & 928 \\ 123 & 246 & 928 & 927 & 925 & 921 & 913 \\ 456 & 439 & 928 & 926 & 920 & 902 & 848 \\ 057 & 228 & 928 & 925 & 913 & 865 & 673 \\ 086 & 430 & 928 & 924 & 904 & 804 & 304 \\ 121 & 726 & 928 & 923 & 893 & 713 & 562 \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ q_0 \\ q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} 000 \\ 923 \\ 437 \\ 541 \\ 017 \\ 637 \\ 289 \end{bmatrix} \rightarrow \begin{bmatrix} 006 \\ 924 \\ 006 \\ 007 \\ 009 \\ 916 \\ 003 \end{bmatrix}$$

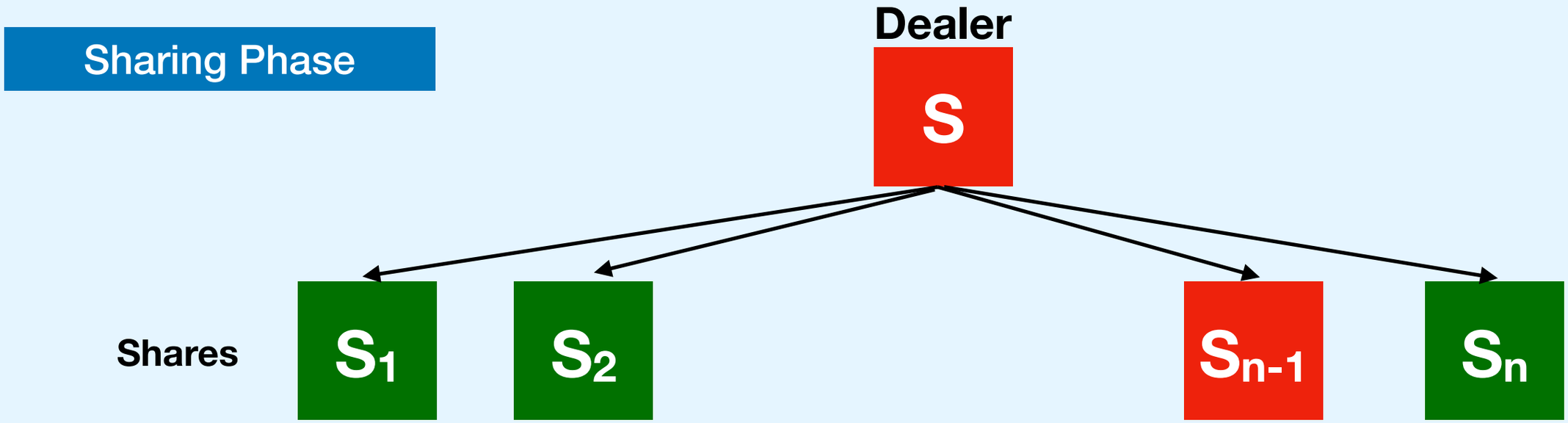
$$E(x) = 006 + 924x + x^2 \qquad Q(x) = 006 + 007x + 009x^2 + 916x^3 + 003x^4$$

$$Q(x)/E(x) = 001 + 002x + 003x^2$$

Dealing with a Corrupted Dealer: Verifiable Secret Sharing

Center for Research in Applied
Cryptography and Cyber Security

What's Next?



How can we validate that all shares of the honest parties lie on the same polynomial of degree t?

Security Properties

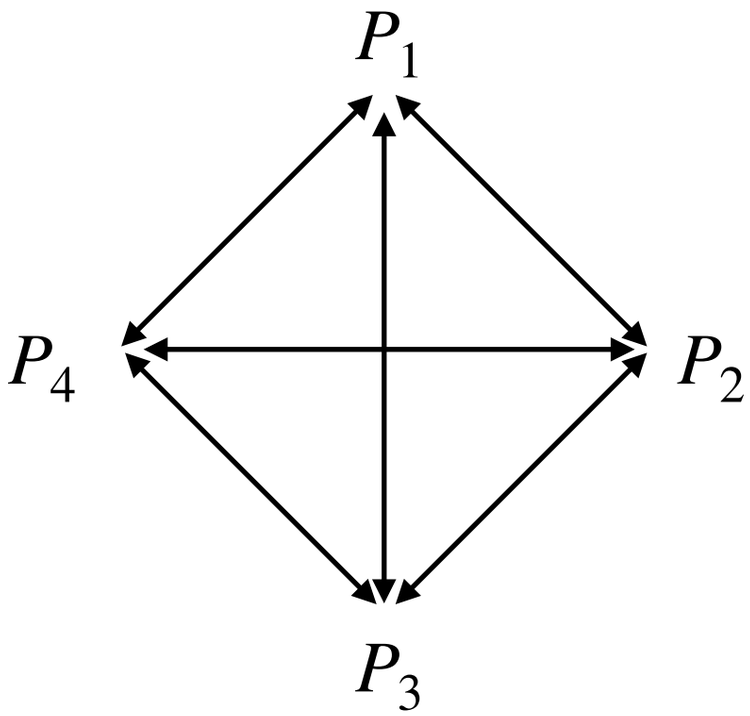
- This time, Sharing is an interactive protocol!
  - The input of the dealer is s
  - All other parties have no inputs

- **Privacy:**
  For an honest dealer, the adversary learns nothing about s

- **Consistency:**
  The outputs of the honest party are consistent with some s^* even if the adversary is corrupted (agreement)

- **Correctness:**
  For an honest dealer, consistency holds with $s^* = s$

Communication Model



Pairwise private channel

Authenticated broadcast channel

Warmup:
The Computational Setting

- $(\mathbb{G}, q, g)$ a group of order q with generator g in which the discrete logarithm problem is hard
- H is a random oracle
- Sharing(s, n, t):
  - **Round I: The dealer:**
    - Choose a random polynomial $B(x) = b_0 + b_1 x + \ldots + b_t x^t$ of degree t
    - Give to each party P_i the share $s_i := B(\alpha_i)$
    - **Broadcast** the values $B_0 = g^{b_0}, \ldots, B_t = g^{b_t}$
    - **Broadcast** the masked secret $c := H(b_0) \oplus s$

Are the Shares Consistent?

$$B_0 = g^{b_0}, \quad \ldots, \quad B_t = g^{b_t}, \qquad c := H(b_0) \oplus s$$

$B(\alpha_1)$ $B(\alpha_2)$ $B(\alpha_{n-1})$ $B(\alpha_n)$

Is my share consistent with whatever was broadcasted?

$$(g^{B(\alpha_i)} =)\Pi_{k=0}^{t}(B_k)^{\alpha_i^k} = \Pi_{k=0}^{t} g^{b_k \cdot \alpha_i^k}?$$

No? This dealer is cheating!

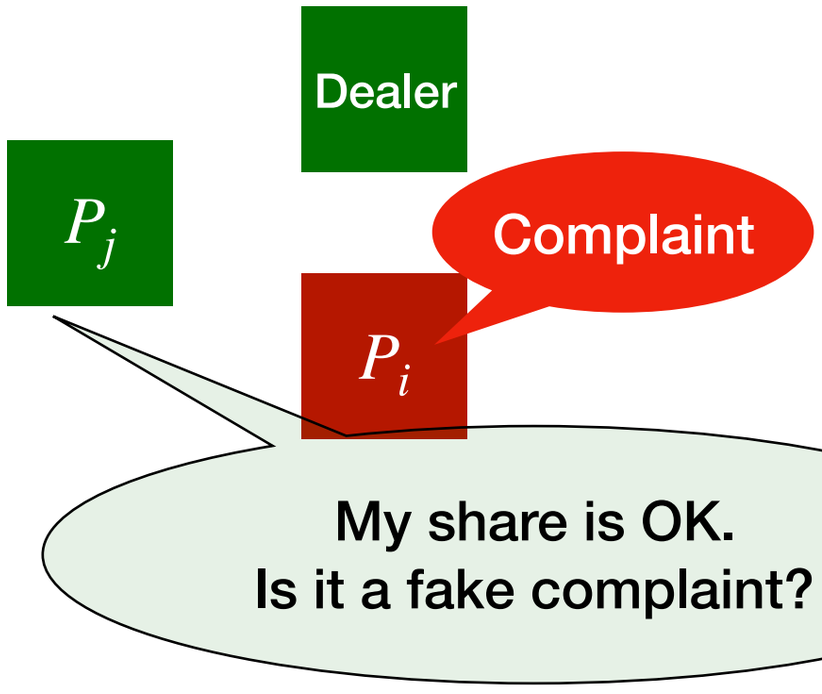
How can I convince the others?

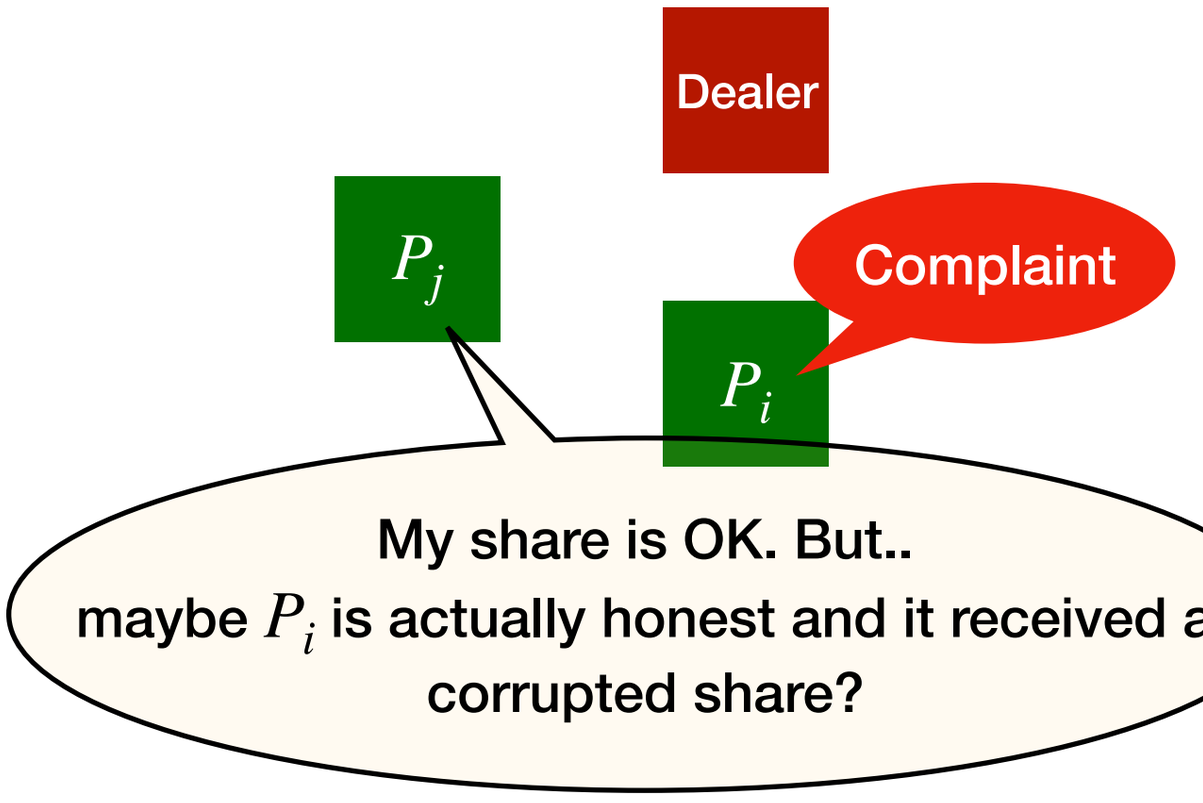
Good or Bad Idea?

- How can I convince others that my share is incorrect?
  - A. I send each party in *a private channel* the share s_i that I received
  - B. I broadcast a "**complaint!**" and I hope that everyone will believe me
  - C. I broadcast a "**complaint!**". If there are more than **t** complaints, then the dealer is corrupted and we can abort the protocol
  - D. I broadcast a "**complaint!**". If there are $\leq t$ complains, then the dealer is honest, and we can finish

Are the Shares Consistent?

Step II: Each party that has a wrong share, broadcasts a complaint
Step III (a): If there are more than **t** complaints, abort
Otherwise…



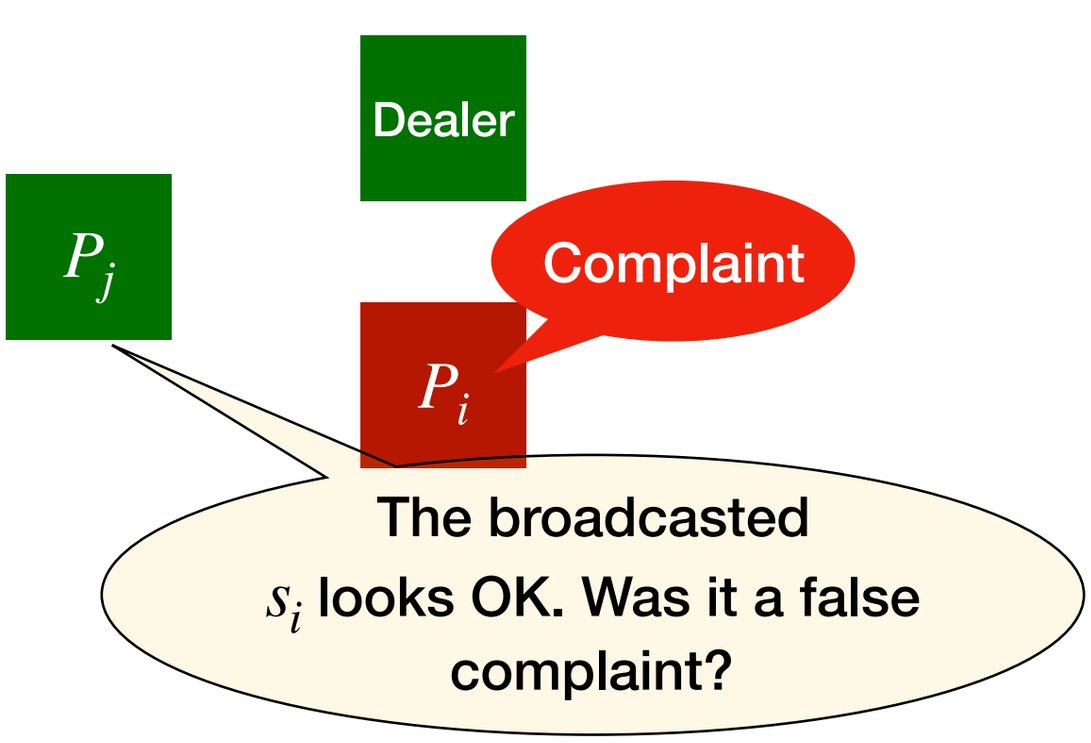


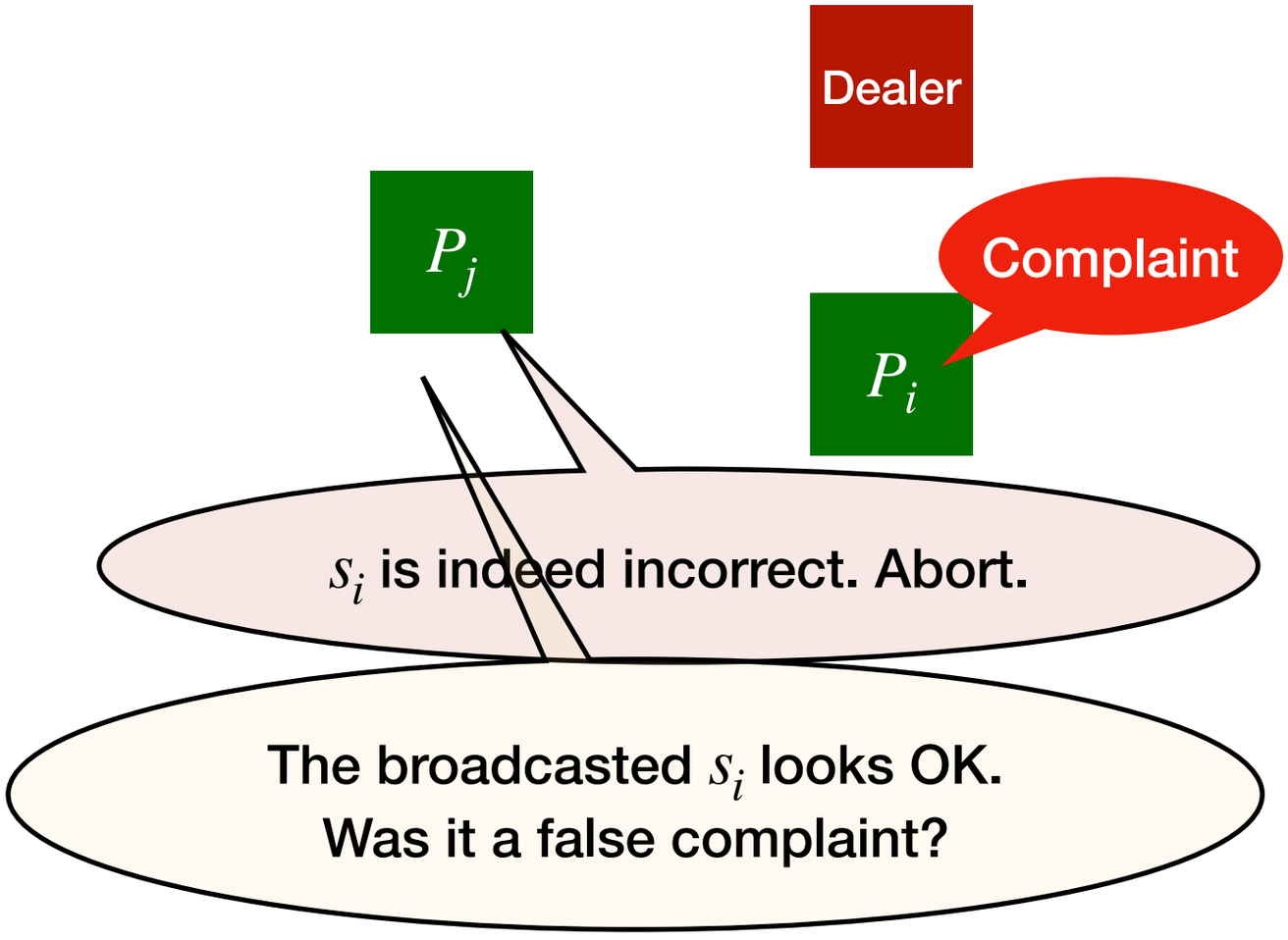
Step III (b): The dealer broadcasts the share of each party that complains

Why is it OK to reveal the share of a party?

Maybe the adversary learns new information?

Three Options To Consider





P_i updates its share to the broadcasted information

If all complaints were resolved $\Longrightarrow$
All honest parties hold shares on the same polynomial

The Entire Protocol

(Think: how to get rid of 🚨 ?)

- **Round I:**
  - The dealer computes shares $s_1, \ldots, s_n$ and sends s_i to P_i privately
  - The dealer publishes authentication information
- **Round II:** Each party checks its share. If wrong - broadcasts “complaint”
- **Round III:**
  - If #complaints $> t$, abort
  - Otherwise, the dealer broadcasts the share of each party that complained
- **Round IV:**
  - All parties verify the broadcasted share (and parties might need to update their shares)
  - If there is some contradiction then abort
  - Otherwise, output your share

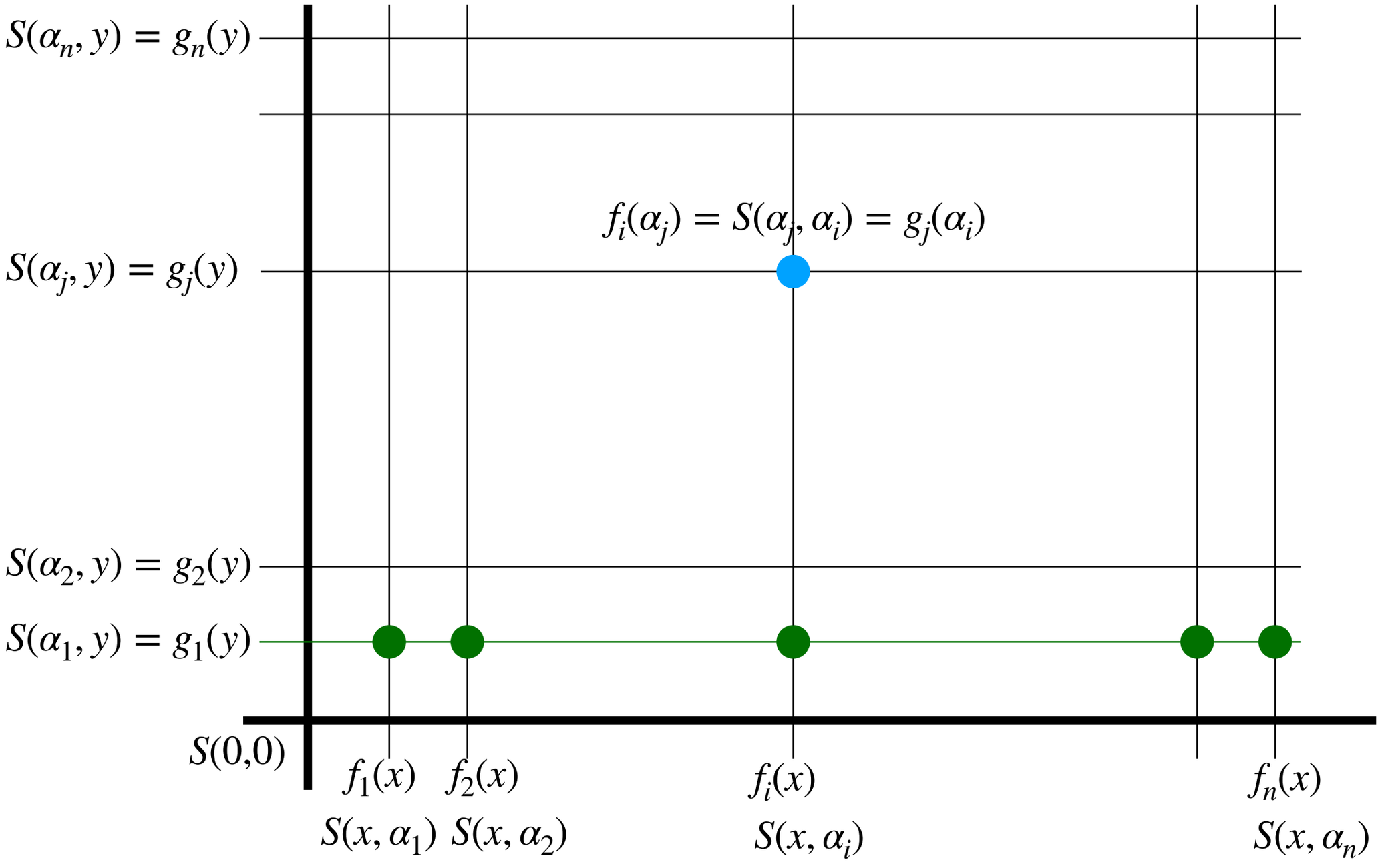
Verifiable Secret Sharing in the Information Theoretic Setting

Center for Research in Applied
Cryptography and Cyber Security

Authentication in the Information Theoretic Setting?

- Bivariate polynomials of degree t that hides the secret s:
- $$S(x,y) = \sum_{i=0}^{t}\sum_{j=0}^{t} a_{i,j} \cdot x^i \cdot y^j \qquad a_{i,j} \in \mathbb{F}, \quad a_{0,0} = s$$
  - Properties:
    - Define $f_i(x) := S(x, \alpha_i)$, $g_i(y) := S(\alpha_i, y)$
      - Both are univariate polynomials of degree-t
      - $f_i(x)$ will be the share of party P_i
      - $g_i(y)$ will be the authentication information of P_i
        - It holds that $g_i(\alpha_j) = S(\alpha_i, \alpha_j) = f_j(\alpha_i)$

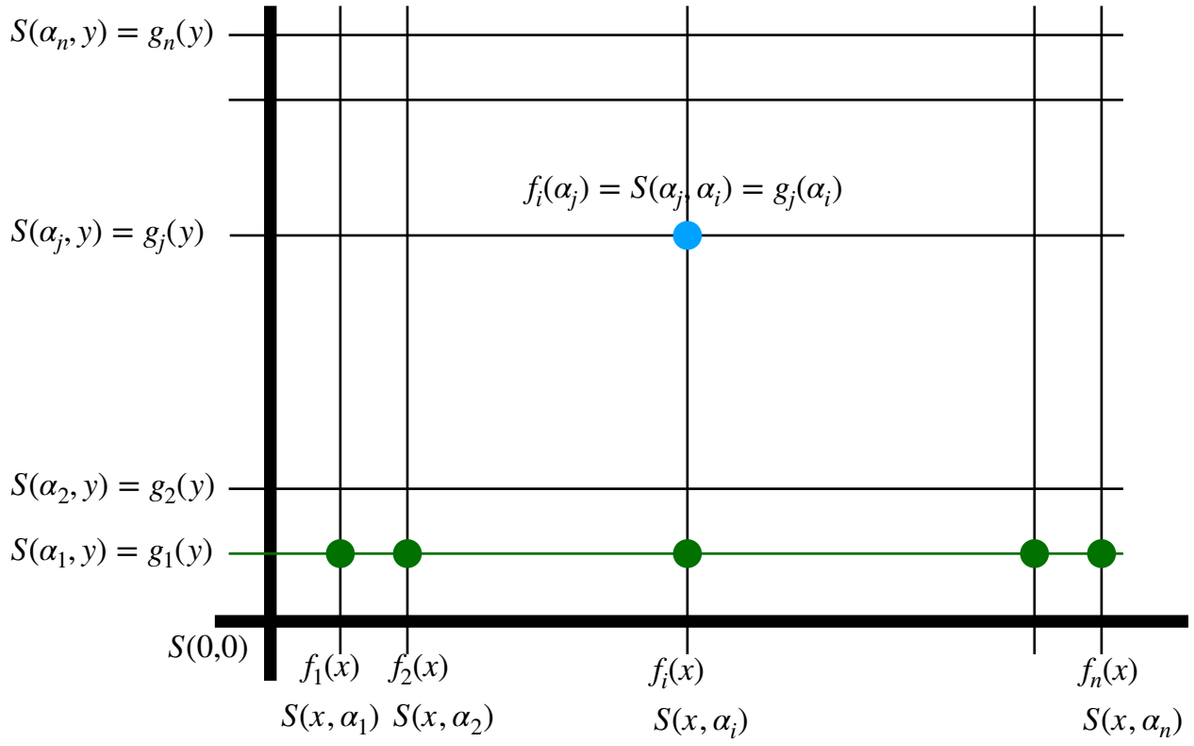
Bivariate Polynomial



Bivariate Polynomial

Security:

- The t polynomials $\{f_i(x)\}_{i\in I}$ for $|I| \leq t$ look random
- A set of $2t$ polynomials $\{f_i(x), g_i(y)\}_{i\in I}$ for $|I| \leq t$ such that $f_i(\alpha_j) = g_j(\alpha_i)$ for every $i, j \in I$ do not reveal any information about s



Interpolation:

- $\{f_i(x), g_i(y)\}_{i\in I} + s$ completely determines $S(x, y)$
- $t + 1$ polynomials $\{f_j(x)\}_{j\in J}$ for $|J| \geq t + 1$ $S(x, y)$

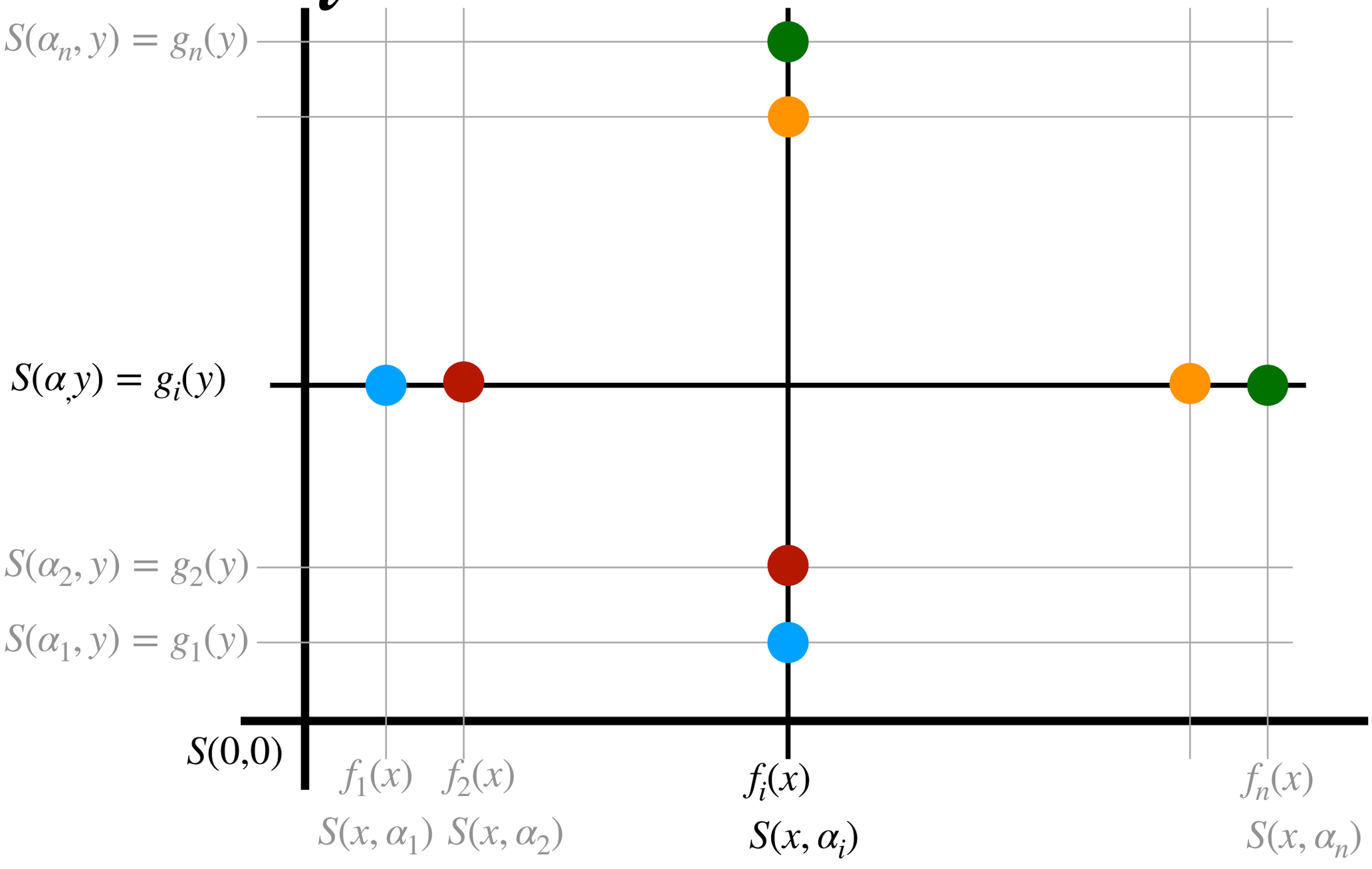
VSS

[BGW88,Feldman88]

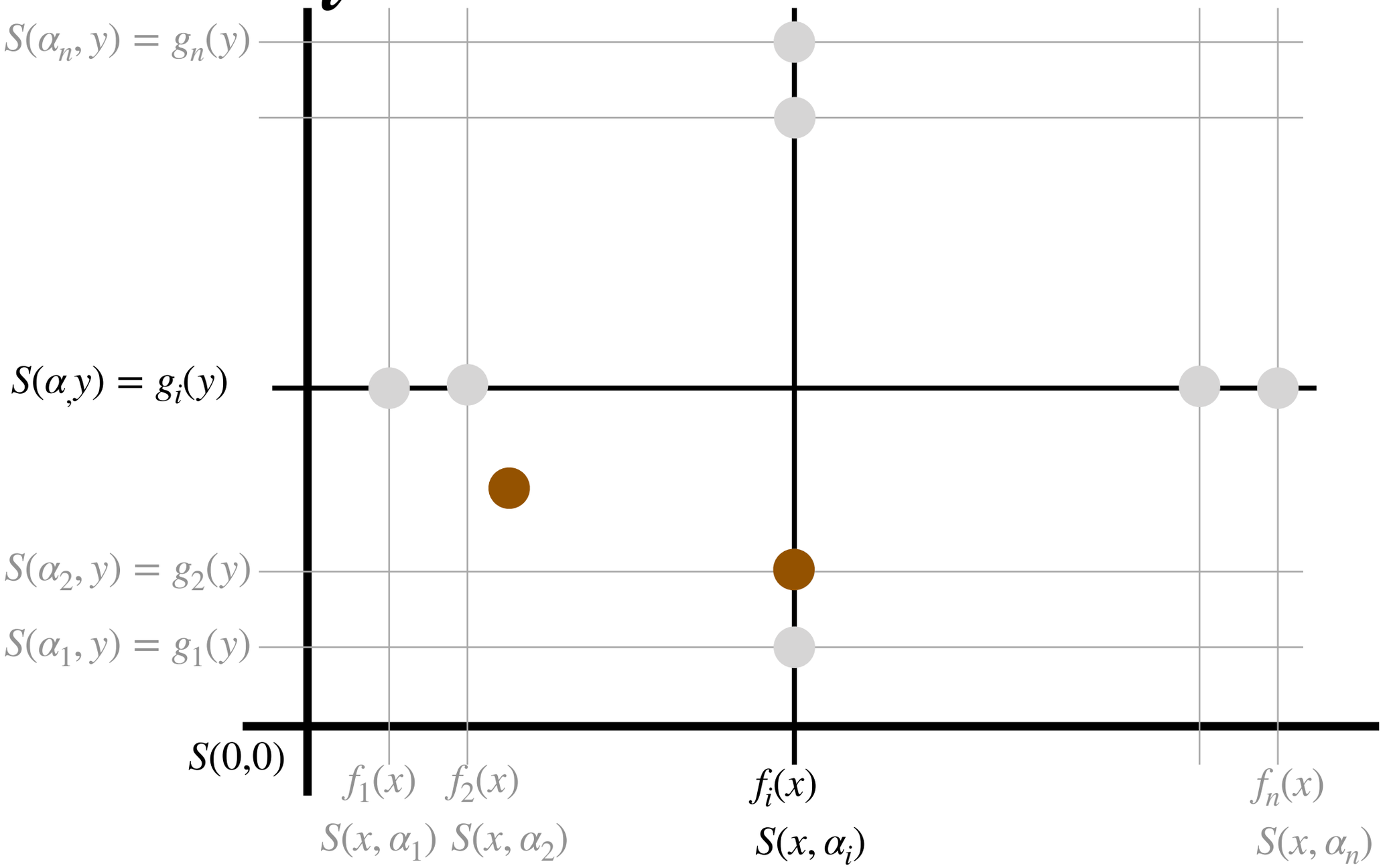
- **Round I - :** The dealer chooses a random bivariate polynomial with $S(x,y) = \sum_{i=0}^{t}\sum_{j=0}^{t} a_{i,j} \cdot x^i \cdot y^j$, $a_{i,j} \in \mathbb{F}$, $S(0,0) = s$
  - Send to party P_i , $f_i(x) = S(x, \alpha_i)$ and $g_i(y) = S($
- **Round II - exchange sub-shares:**
  - P_i sends $f_i(\alpha_j) = S(\alpha_j, \alpha_i)$ and $g_i(\alpha_j) = S(\alpha_i,$
- **Round III - Check and complain:**
  - P_i checks the two values (u_j, v_j) it received from P_j
  - If something is wrong, **complaint**$(i, j, f_i(\alpha_j), g_i(\alpha_j))$

Attention!
P_i complains with the points it received from the dealer, not P_j

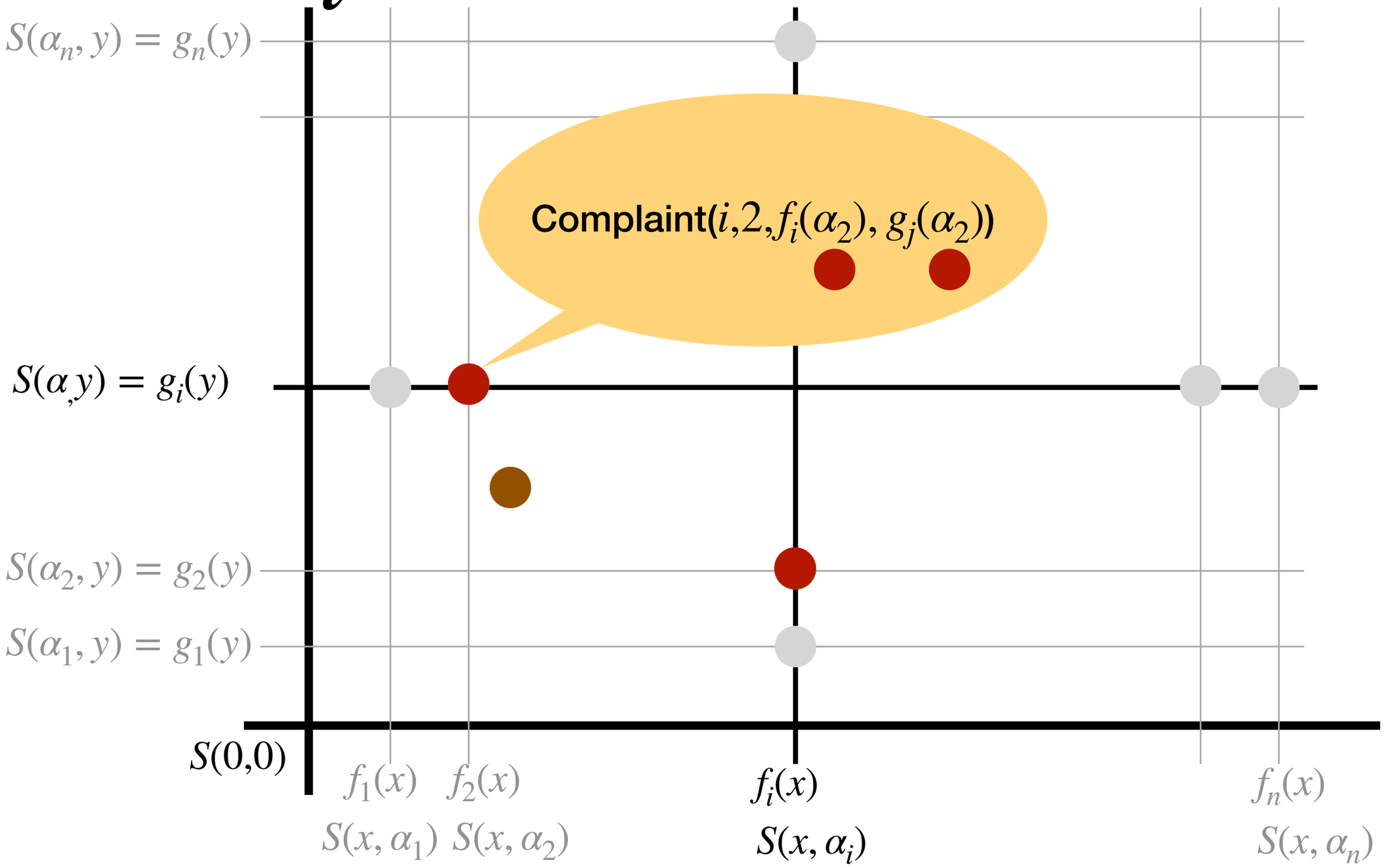
P_i's Point of View



P_i's Point of View



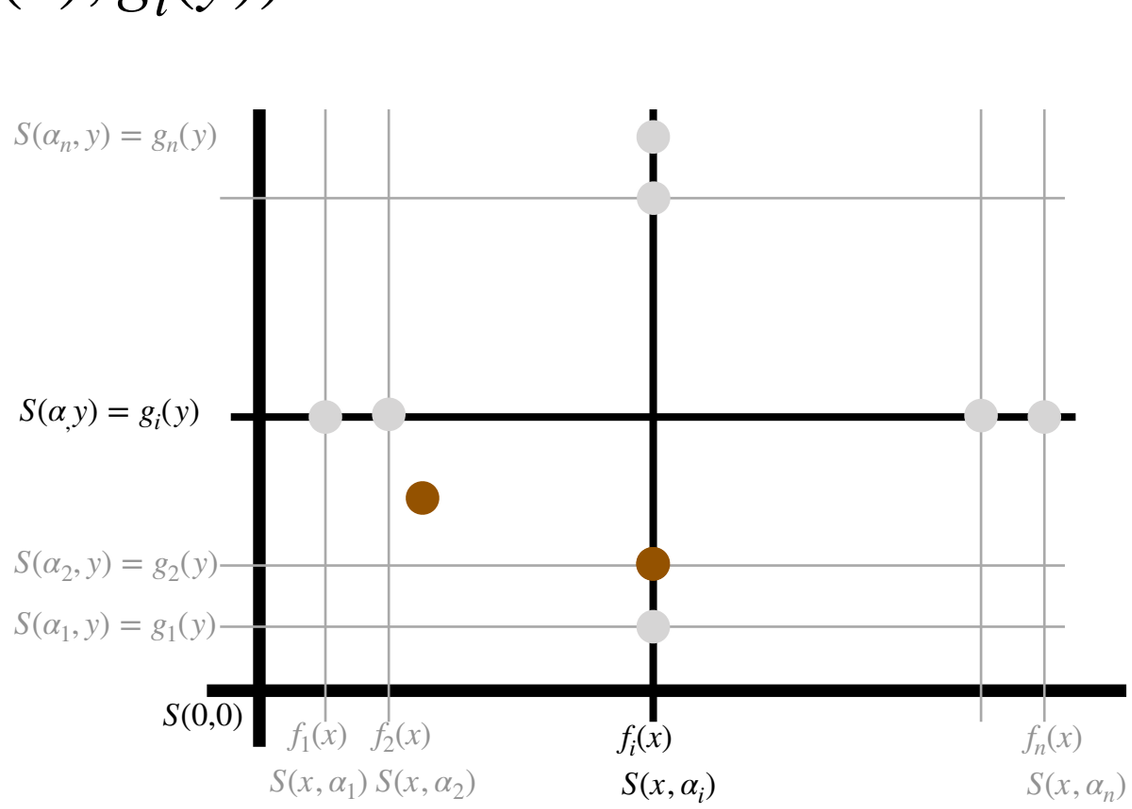
P_i's Point of View



VSS - Round IV

- **Round IV: Check complaints - dealer:**
  - For each **complaint**(i, j, u, v) check: $u = S(\alpha_j, \alpha_i)$ and $v = S(\alpha_i, \alpha_j)$.
  - If holds - do nothing
  - Otherwise, broadcast **reveal**$(i, f_i(x), g_i(y))$

The dealer reveals the entire share of P_i



What Happens if the Dealer is Honest?

- **Q:** Is it possible that an honest party receive wrong sub-shares from another honest party?

- **Q:** Will an honest party ever broadcast a **complaint** if the dealer is honest?

- **Q:** Will the dealer broadcast the shares of honest parties?

- **Q:** May the dealer reveal a share of someone?

- **Q:** If some polynomial is broadcasted, will it contradict a polynomial that is held by some honest party?

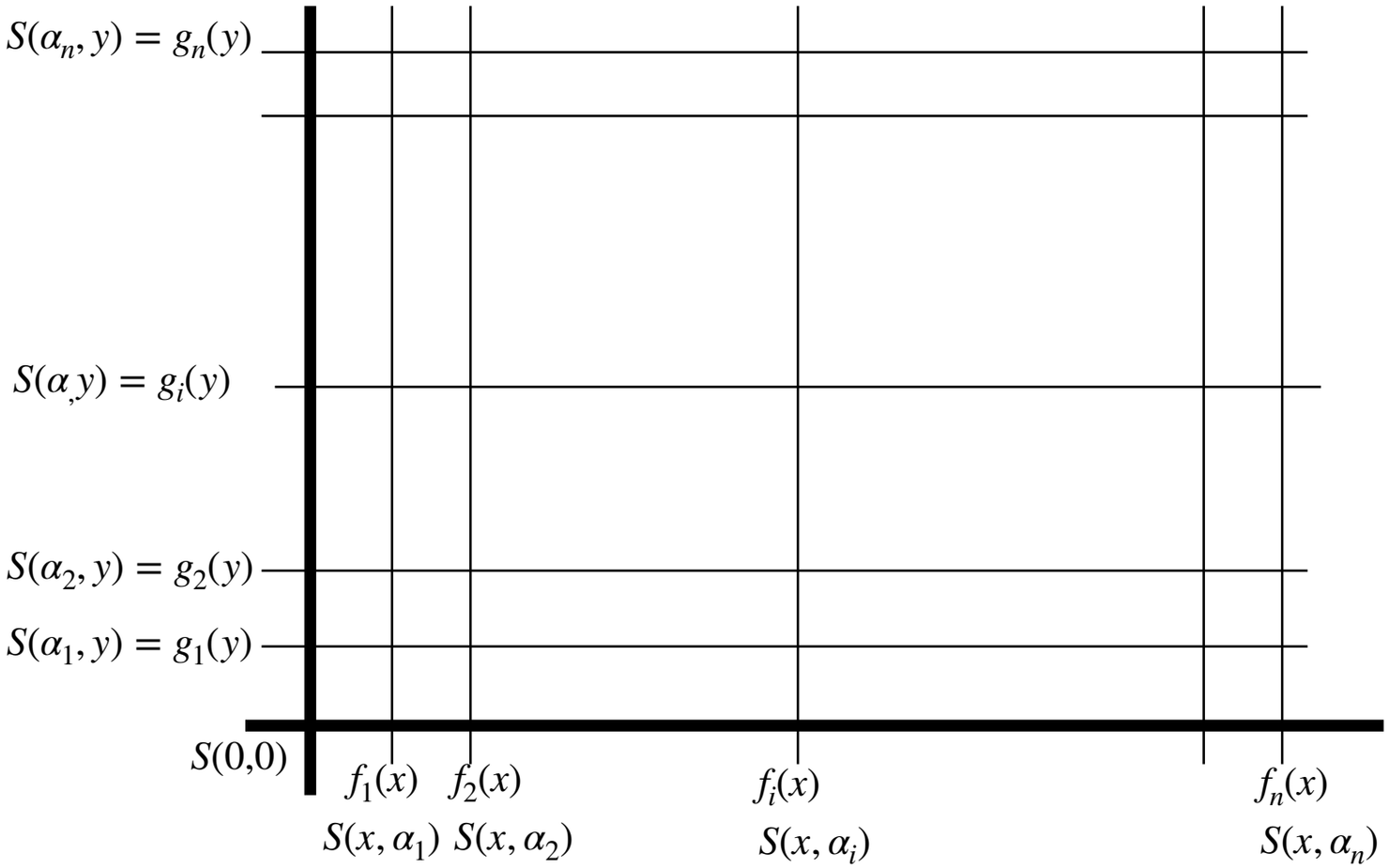
Round V: A Vote

- Each party P_i:
  - If there are joint complaints — **complaint**(k, j, u, v) and **complaint**(j, k, u', v') then the dealer must response to one of them
    - No response from the dealer $\Longrightarrow$ reject
      - For each message **reveal**$(j, f_j(x), g_j(y))$ broadcasted by the dealer
        - Check that $f_i(\alpha_j) = g_j(\alpha_i)$ and $g_i(\alpha_j) = f_j(\alpha_i)$
        - If $j = i$ then P_i updates its share
- Vote: if whatever was broadcasted is consistent, and my share was not updated, then broadcast **consistent**
- If at least $n - t$ broadcasted **consistent** then output $f_i(x)$

Questions: Corrupted Dealer

- **Q:** Is it possible that an honest party receive wrong points from another honest party?
  - Yes; The dealer might send contradicting shares to $P_i,\ P_j$
- **Q:** $n - t$ parties voted **consistent**. How many polynomials of honest parties were replaced?
  - At most t!
  - More importantly, at least $t + 1$ honest parties received consistent shares already in Round I
  - The bivariate polynomial and the secret are well defined!
- **Q:** If only $t + 1$ honest parties broadcasted **consistent**, does it guarantee that all honest parties hold shares on the same bivariate polynomial?
  - Yes! An honest party broadcasts **consistent** only if all conflicts were resolved and its share was not replaced
  - All information that the dealer broadcasted is consistent

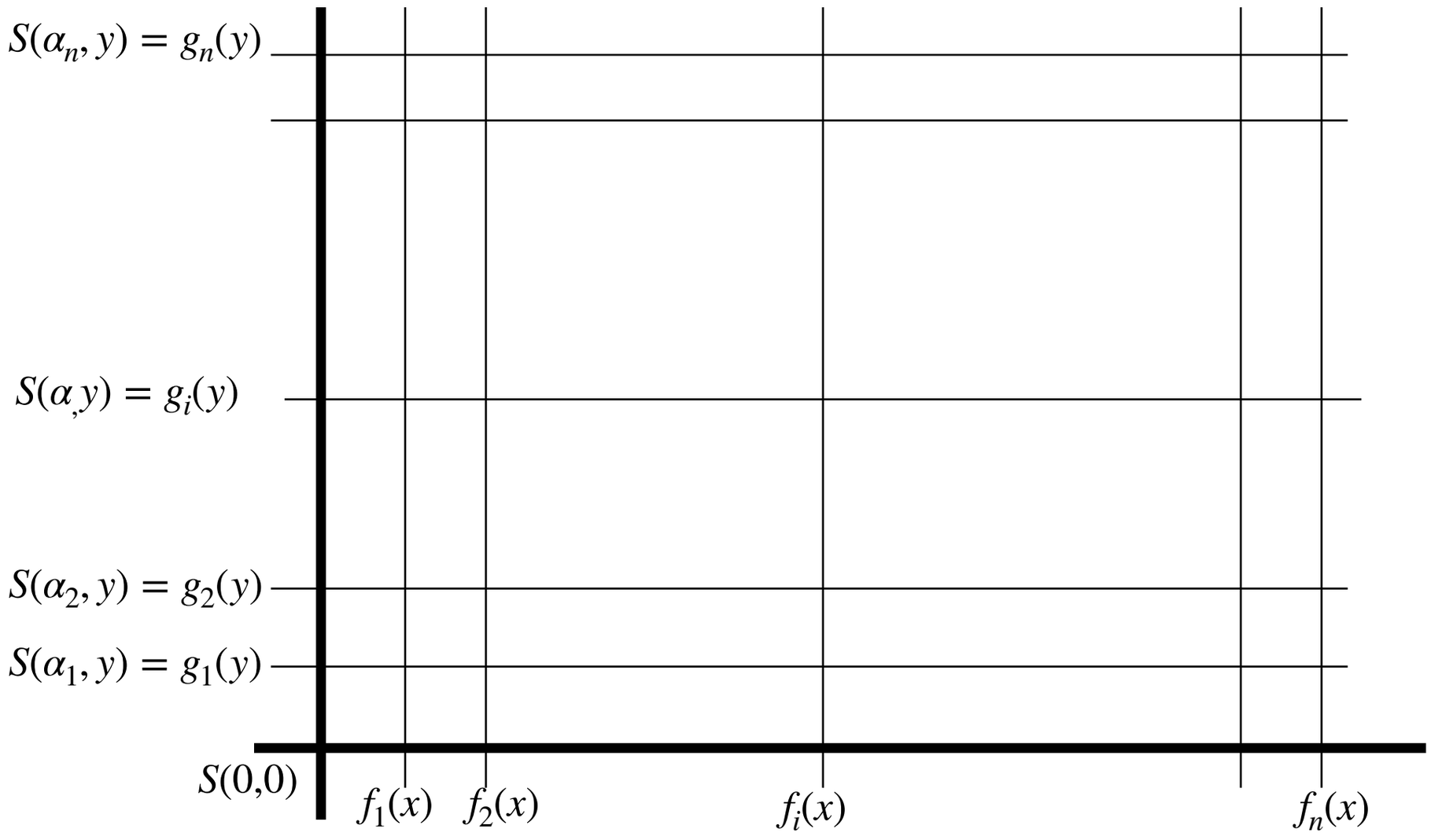
Bivariate Shares - Reconstruction



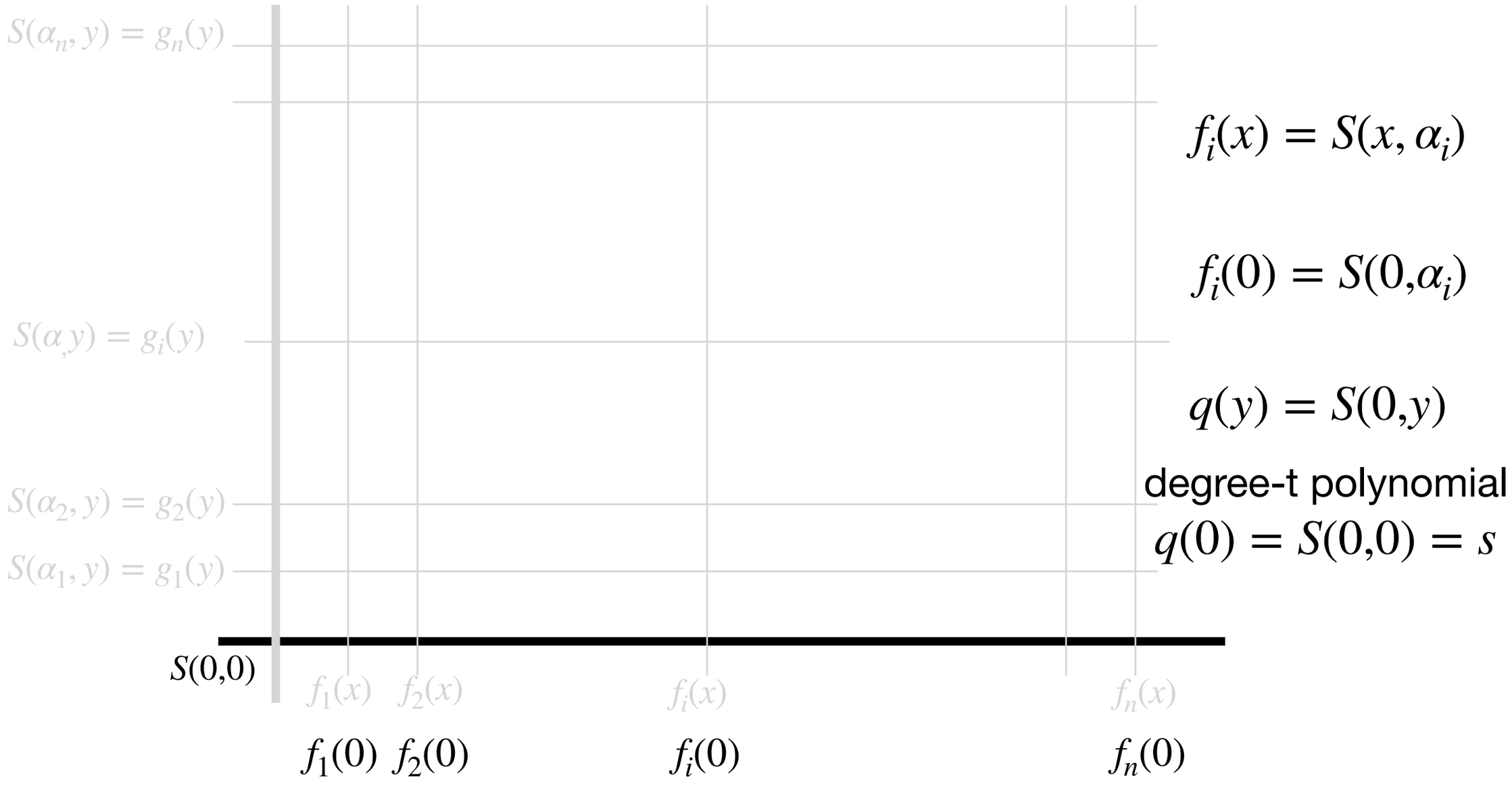
- Each party P_i:
  - Publish $f_i(x), g_i(y)$
- Initialize $K \leftarrow \varnothing$
- For $j = 1, \ldots, n$:
  - Does f_j consistent with at least $2t + 1$ g's?
  - Yes - add j to K
- Reconstruct $S(x, y)$ from the set of polynomials in K

Reconstruction requires just private channels (no broadcast!)

However, in Many Protocols…



However, in Many Protocols…



Conclusion

- Let $t < n/3$. There exists a perfectly secure VSS protocol in the presence of a malicious adversary
- Moreover:
  - Let $t < n/2$. There exists a *statistically* secure VSS protocol in the presence of a malicious adversary [RabinBenOr89]
    - (assuming broadcast)

Thank You!