Secret Sharing

CS 6750 Lecture 7

October 29, 2009

Riccardo Pucella

The Treasure Map Problem

- Suppose you and a “friend” find a map that leads to a treasure
- You each want to go home and prepare
- Who keeps the map?
- What if you don’t trust each other?

A Real Life Solution

- Split the map in two
  - Such that you need both pieces to find the island
  - You and your friend each take a piece
- This is the basic idea of secret splitting
  - A special case of secret sharing

Secret Splitting

- **Definition**: given a secret S, we would like N parties to share the secret so that the following properties hold:

  1) All N parties can recover S

  2) Less than N parties cannot recover S

- In general, we split the secret into N pieces (shares) S_1, …, S_N and give one share to each party.

Does This Work?

- Without loss of generality, we consider the secret to be a bit string or an integer
  - We know everything can be encoded as such

- Concrete example: suppose you want to keep your salary secret, but share it between two parties. If your salary is $150,000, you could always split it as 150 and 000, and give each a piece.
  - Is this a good way to split such a secret?

Partial Information Disclosure

- In the above scheme, we are leaking partial information about the secret
  - E.g., the most significants digits of the salary
  - Problem for some applications (not always)
    - E.g., secret is a password
- In general, hard to characterize what kind of information should not be leaked, and which is okay to leak.
  - So we want to forbid any kind of partial information disclosure

Revised Definition

- **Revised definition**: given a secret S, we would like N parties to share the secret so that the following properties hold:

  1) All N parties can recover S

  2) Less than N parties cannot recover S or obtain any partial information about S

- This is surprisingly easy to achieve

A Two-Party Scheme

- Suppose S is a bitstring in $\{0,1\}^m$
  - Choose m bits at random (coin tosses)
  - Let S_1 be those m random bits
  - Let $S_2 = S \oplus S_1$
- Easy: Given S_1 and S_2, reconstruct $S = S_1 \oplus S_2$

No Partial Information Disclosure

- Given S_1 (or S_2), we do not get any partial information about S
  - How can we formalize that?
  - Show that given S_1, you do not restrict what S could have been. Information == restricted possibilities
  - Given S_1, for any T there exists S_T such that $S_1 \oplus S_T = T$
  - A share can be a share for any secret!

Generalization to N parties

- Suppose S is a bit string in $\{0,1\}^m$
  - Choose m bits at random (coin tosses)
  - Let S_1 be those m random bits
  - Do the same for $S_2, \ldots, S_{N-1}$ (all random)
  - Let $S_N = S \oplus S_1 \oplus \ldots \oplus S_{N-1}$

- Argument for no partial information disclosure similar to above

The Generals Problem

- You have been put in charge of designing a control mechanism for your country's nuclear arsenal. You choose a keyed secret code mechanism:
    - To launch missiles, you need the right secret code
- You don't want to give every general the code
    - A rogue general might just launch an attack!
    - You decide to split the code among the generals

- What's your new problem?

Availability

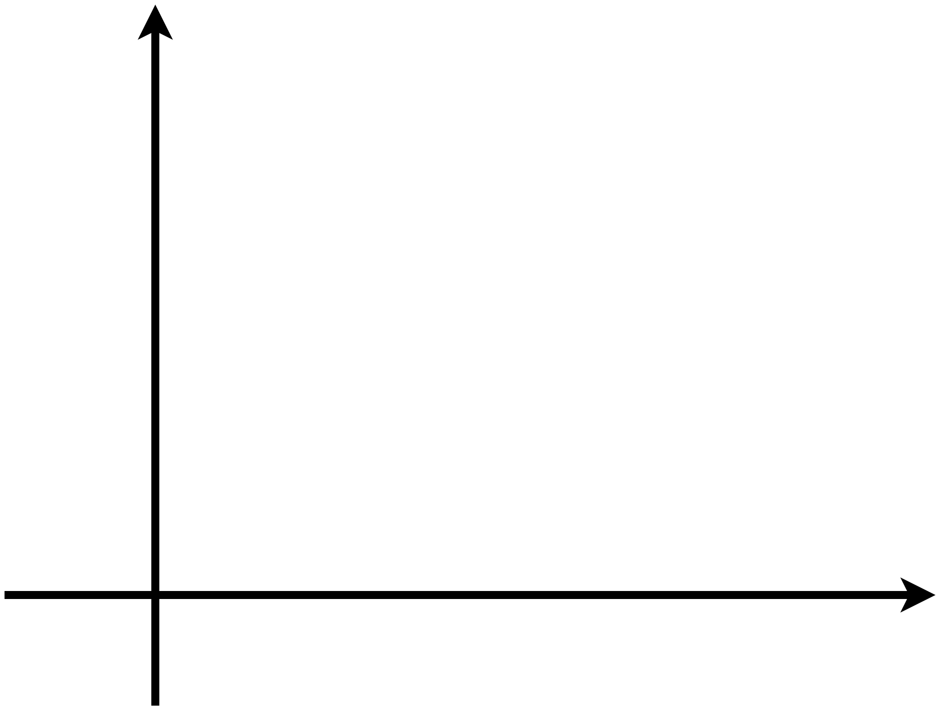
- Secret splitting ensures that the partial information about the secret is not recoverable unless you have all the shares
- But it does not guarantee availability, that you can recover the secret even if some of the shares are unavailable
  - E.g. 2 or more generals can launch missiles
  - but less than 2 generals cannot

(N,T) Secret Sharing

- **Definition**: Given a secret S, we would like N parties to share the secret so that the following properties hold:
  - Greater than or equal to T parties can recover S
  - Less than T parties cannot recover S or obtain any partial information about S
- Generals problem == (3,2) secret sharing
- Secret splitting == (N,N) secret sharing

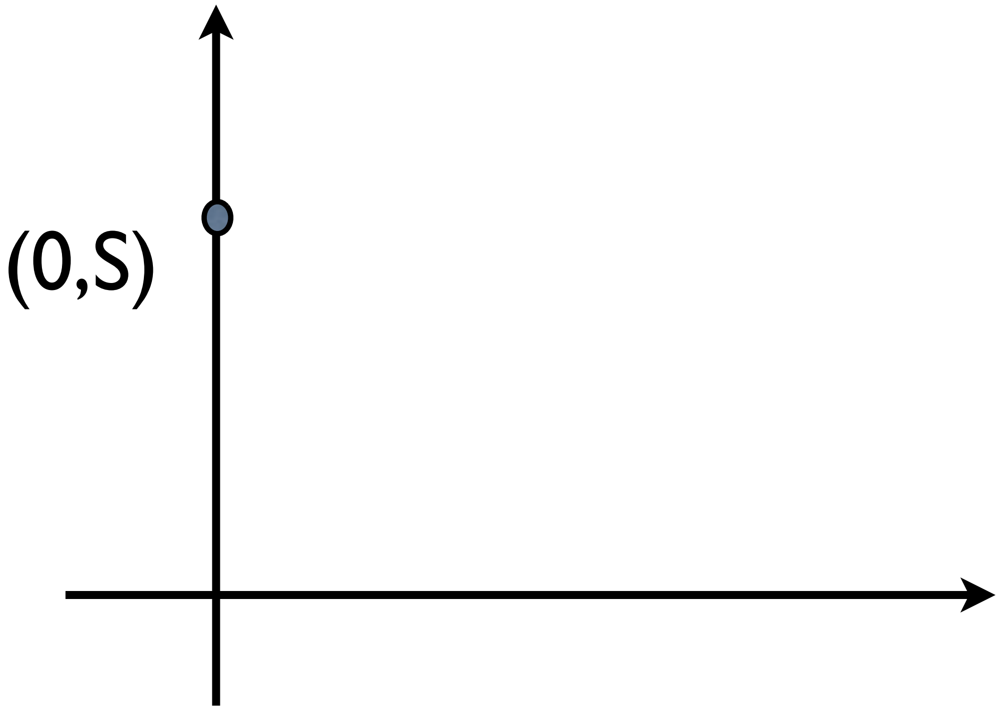
Shamir's Threshold Scheme

- To motivate the general solution, consider first an (N,2) secret sharing scheme
- Secret S is an integer



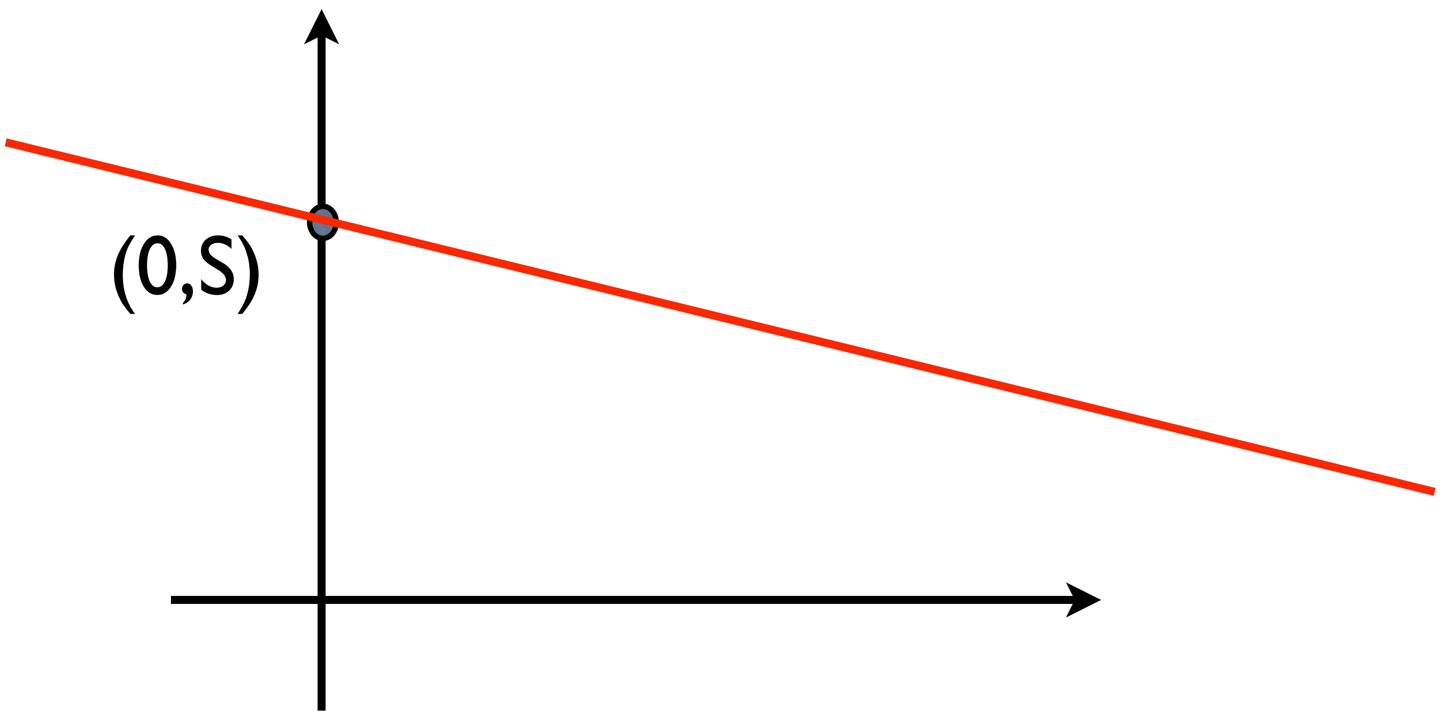
Shamir's Threshold Scheme

- To motivate the general solution, consider first an (N,2) secret sharing scheme
- Secret S is an integer



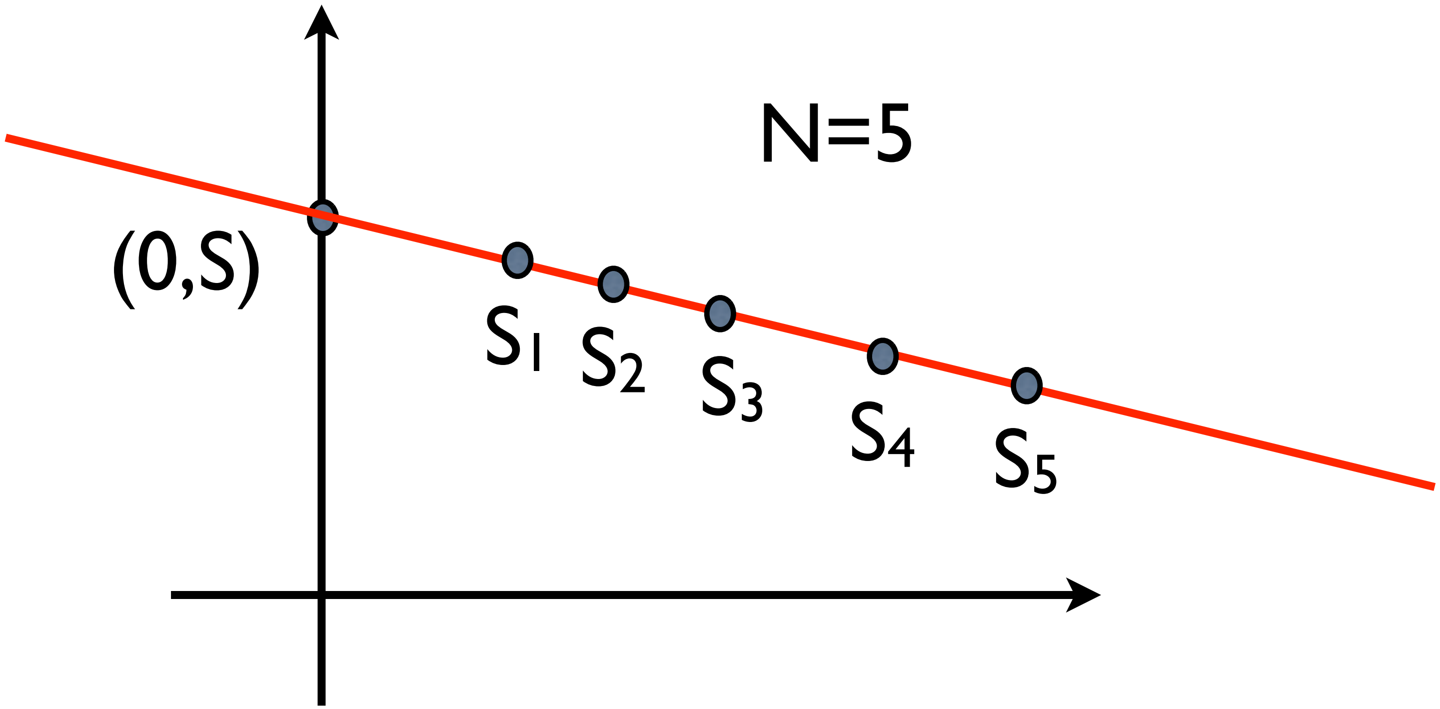
Shamir's Threshold Scheme

- To motivate the general solution, consider first an (N,2) secret sharing scheme
- Secret S is an integer



Shamir's Threshold Scheme

- To motivate the general solution, consider first an (N,2) secret sharing scheme
- Secret S is an integer

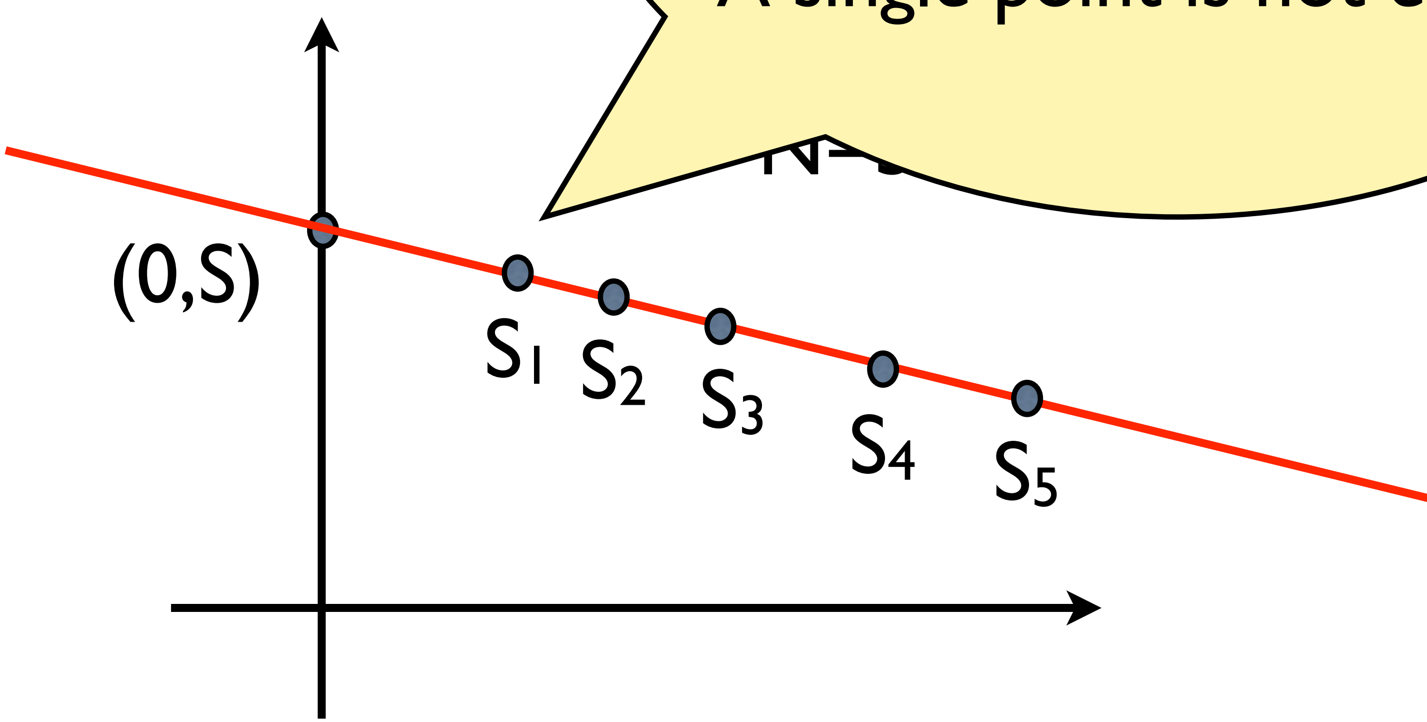


Shamir's Threshold Scheme

- To motivate the gene
  (N,2) secret sharing
- Secret S is an integ

Easy to check: any two points can be used to recover the line and hence (0,S)

A single point is not enough



Generalizing to (N,T)

- A line intersecting the y axis = degree 1 polynomial [$y = a_1x+a_0$]
  - Line uniquely characterized by two points
  - Once you know the line, you can compute where it crosses the y axis.
- Generalize to (N,T) threshold schemes
  - Use a degree T-1 polynomial [$y= a_{T-1}x^{T-1}+\ldots+a_1x+a_0$]
  - Curve uniquely characterized by T points
  - Once you know the curve, you can compute where it crosses the y axis

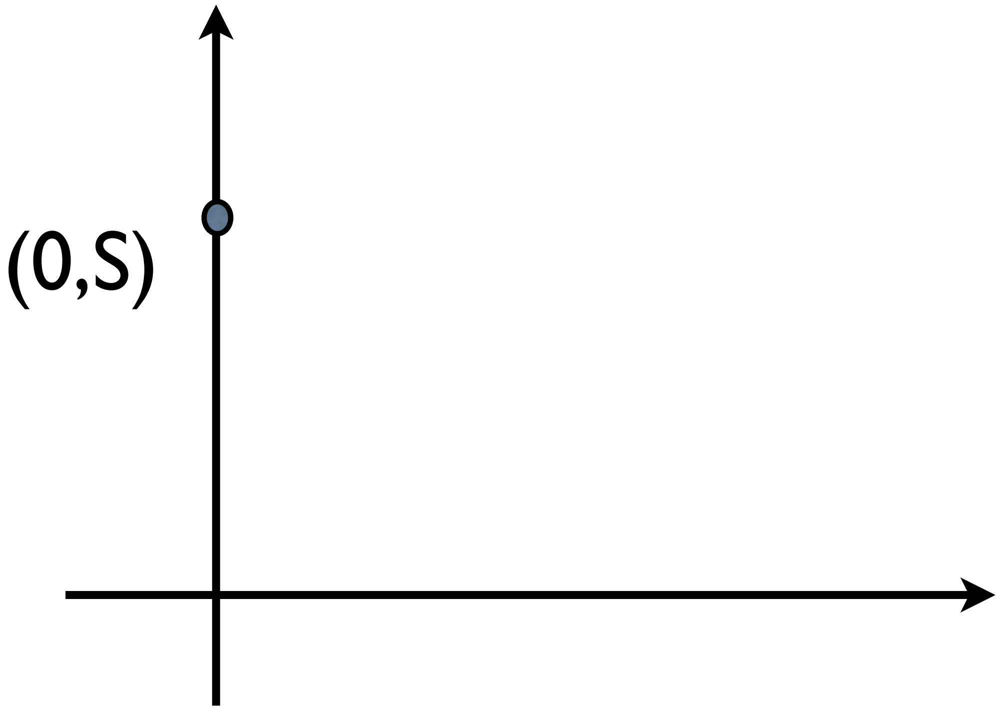
Resharing the Secret

- This can be useful when the secret needs to be kept for a long time
  - The longer a secret needs to be kept, the more likely the adversary is to get enough shares

- The Shamir threshold scheme admits resharing the secret **without** computing that secret

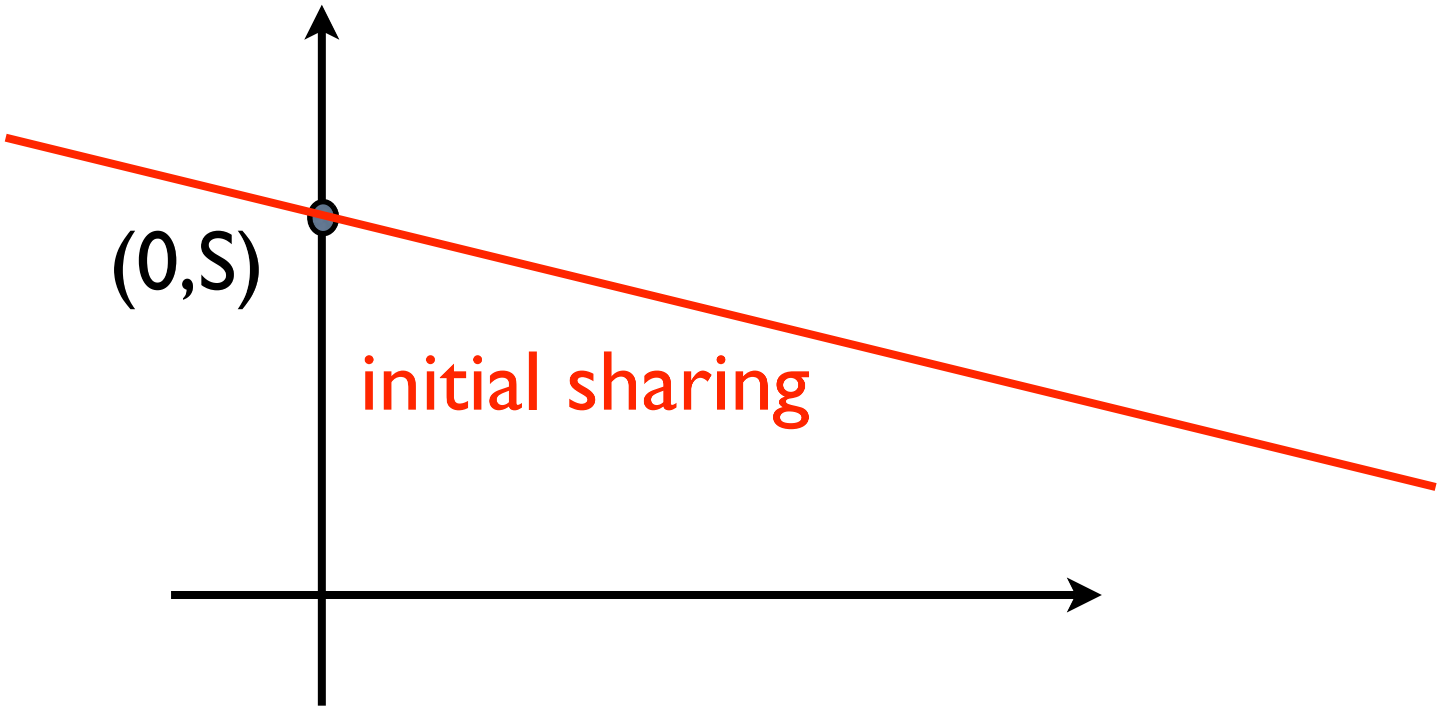
Generating New Shares

- Again, let's consider the (N,2) case
- Secret S is an integer



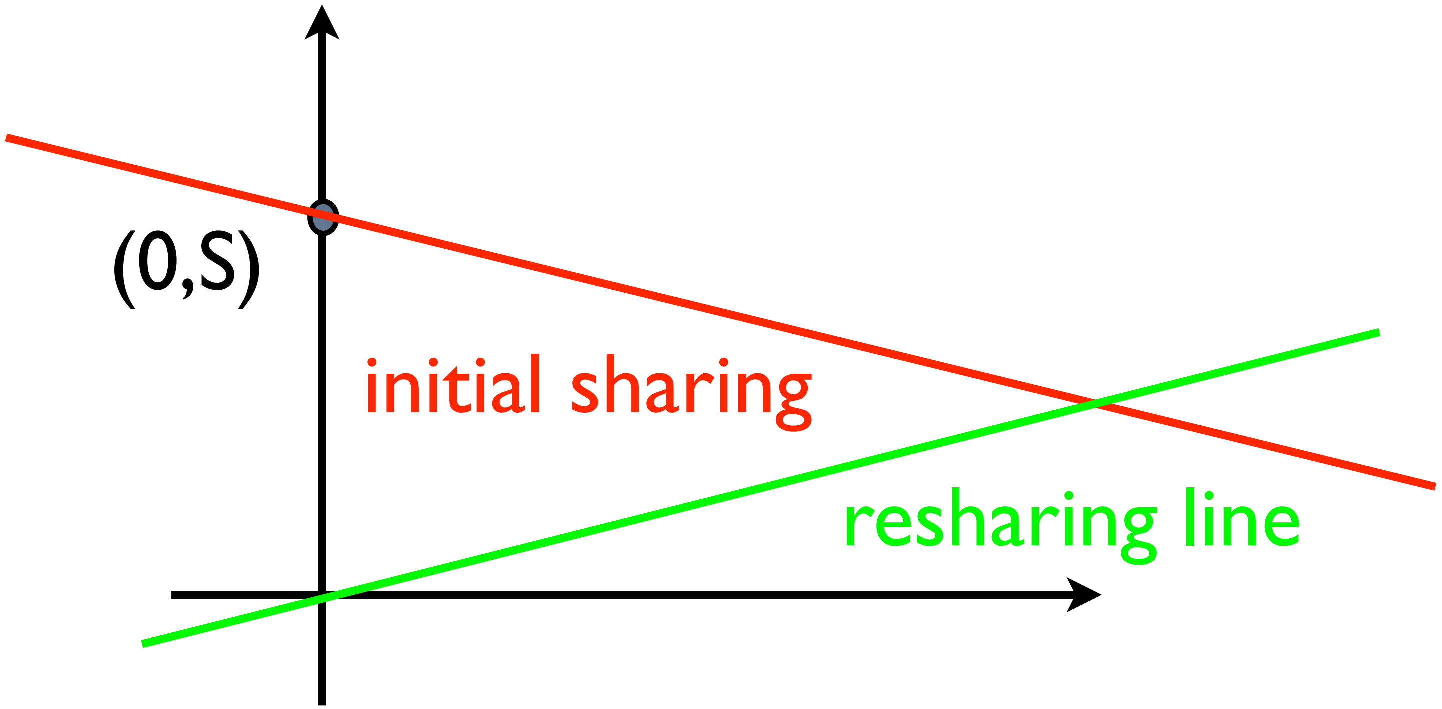
Generating New Shares

- Again, let's consider the (N,2) case
- Secret S is an integer



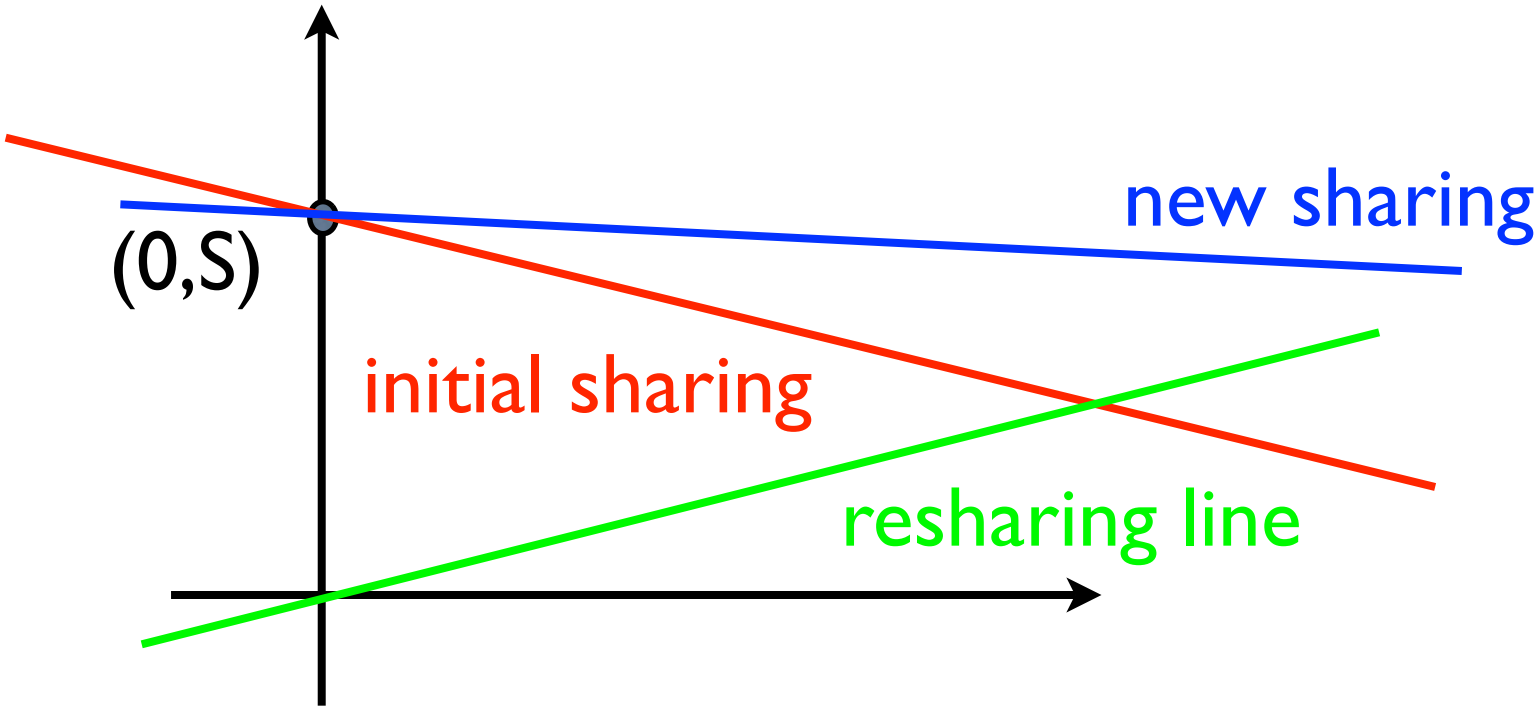
Generating New Shares

- Again, let's consider the (N,2) case
- Secret S is an integer



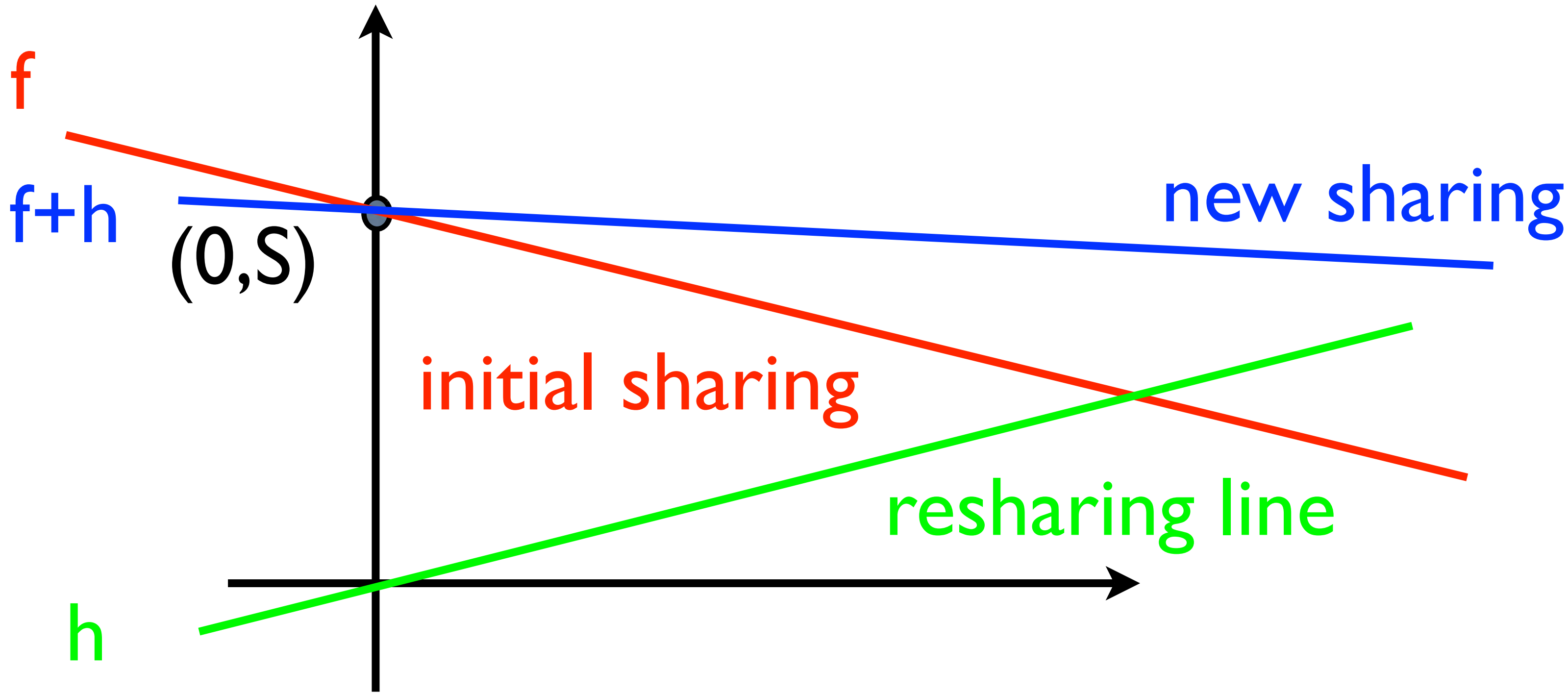
Generating New Shares

- Again, let's consider the (N,2) case
- Secret S is an integer



Generating New Shares

- Again, let's consider the (N,2) case
- Secret S is an integer

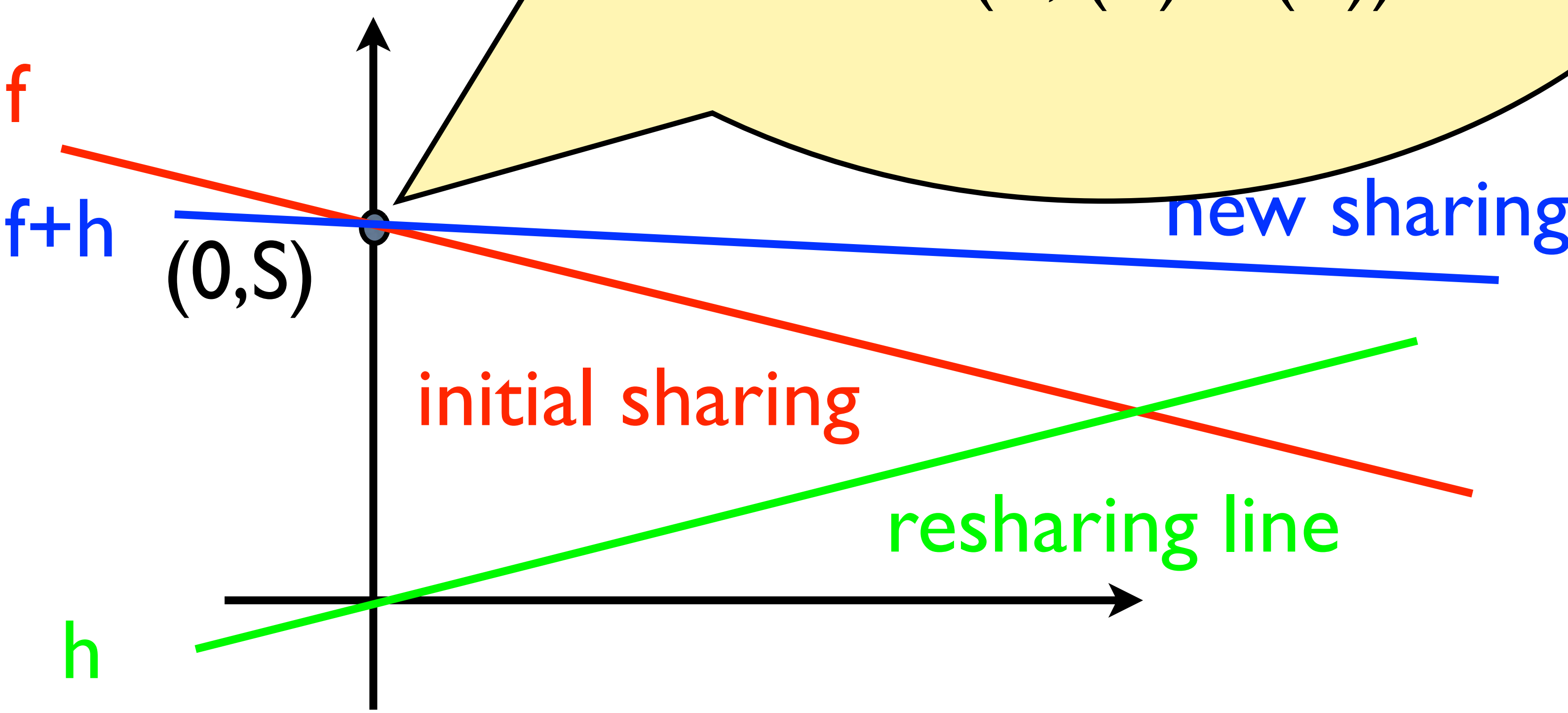


Generatin… es

- Again, let's con…
- Secret S is an…

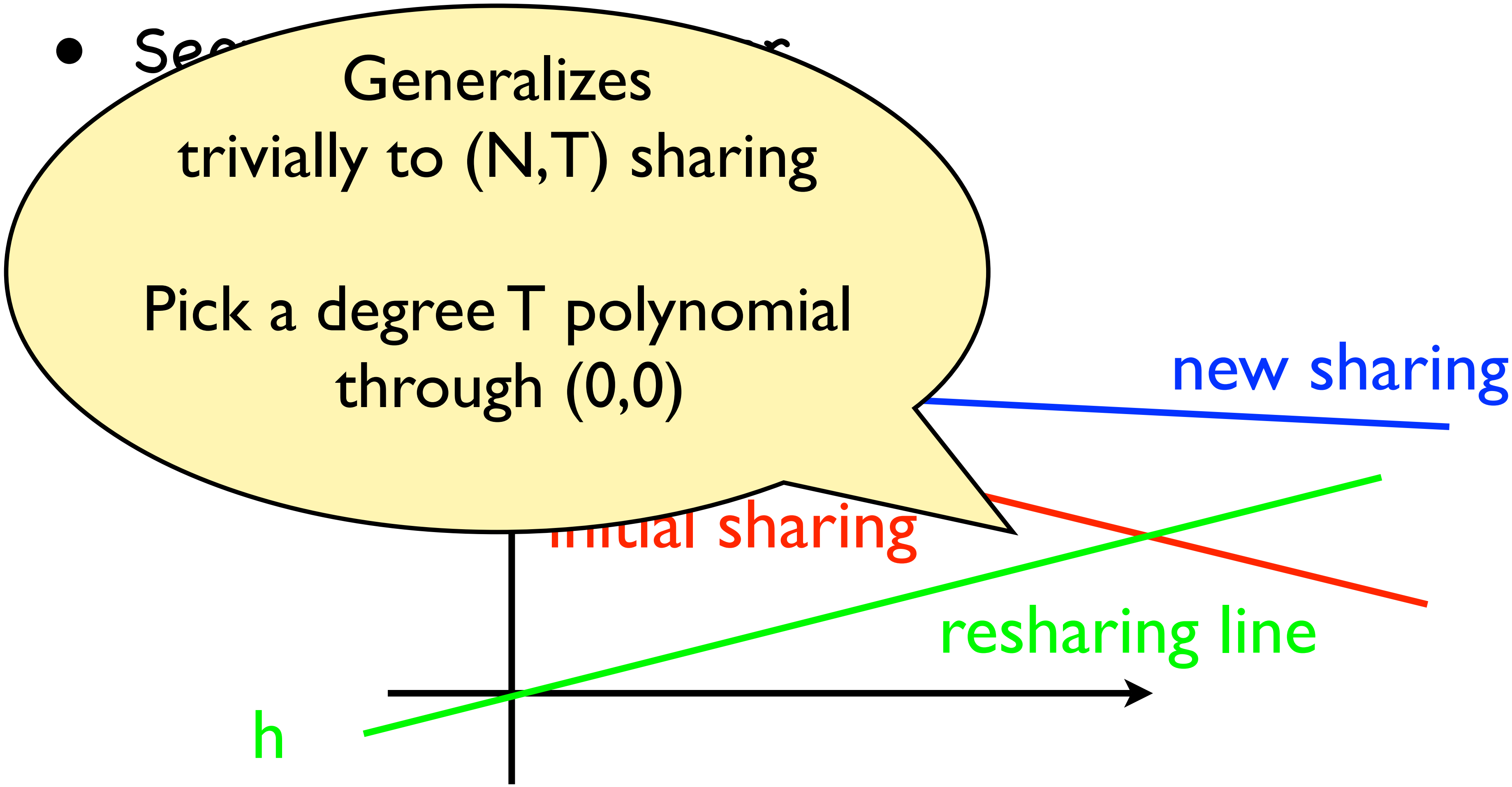
A central server wanting to reshare the secret would send $h(x_1)$ to party 1, ..., $h(x_n)$ to party n

Each party would compute their new share $(x_i, f(x_i)+h(x_i))$



Generating New Shares

- Again, let's consider the (N,2) case
- Se



General Secret Sharing

- Suppose you want an even more general way of sharing secrets
  - N parties, and you specify exactly what subsets of parties can get the secret
  - E.g. Bob and Alice can get together and reconstruct the secret, Bob and Charlie can get together and reconstruct the secrete, but no one else

Access Structure

- An access structure for a set P of parties is a set AS of subsets of P
- $B \in AS$ is called an authorized subset
- Access control structures are monotone:
  - If $B \in AS$ and $B \subseteq C \subseteq P$, then $C \in AS$
- We often only list the "minimal" elements: the sets $B \in AS$ such that there is no $C \in AS$ with $C \subset B$

Perfect Secret Sharing Scheme for AS

- **Definition:** A perfect secret sharing scheme realizing the access structure AS is a method of sharing a secret S among a set P of parties such that:

1) Any authorized subset of AS can recover S
2) No unauthorized subset can recover S or obtain any partial information about S

Threshold Access Structures

- Let P be a set of N parties
  - Take AS = { B $\subseteq$ P : |B| $\geq$ T}
  - This is called a threshold access structure
- A (N,T) secret sharing scheme == a perfect secret sharing scheme realizing a threshold access structure

Secret Sharing Scheme for AS

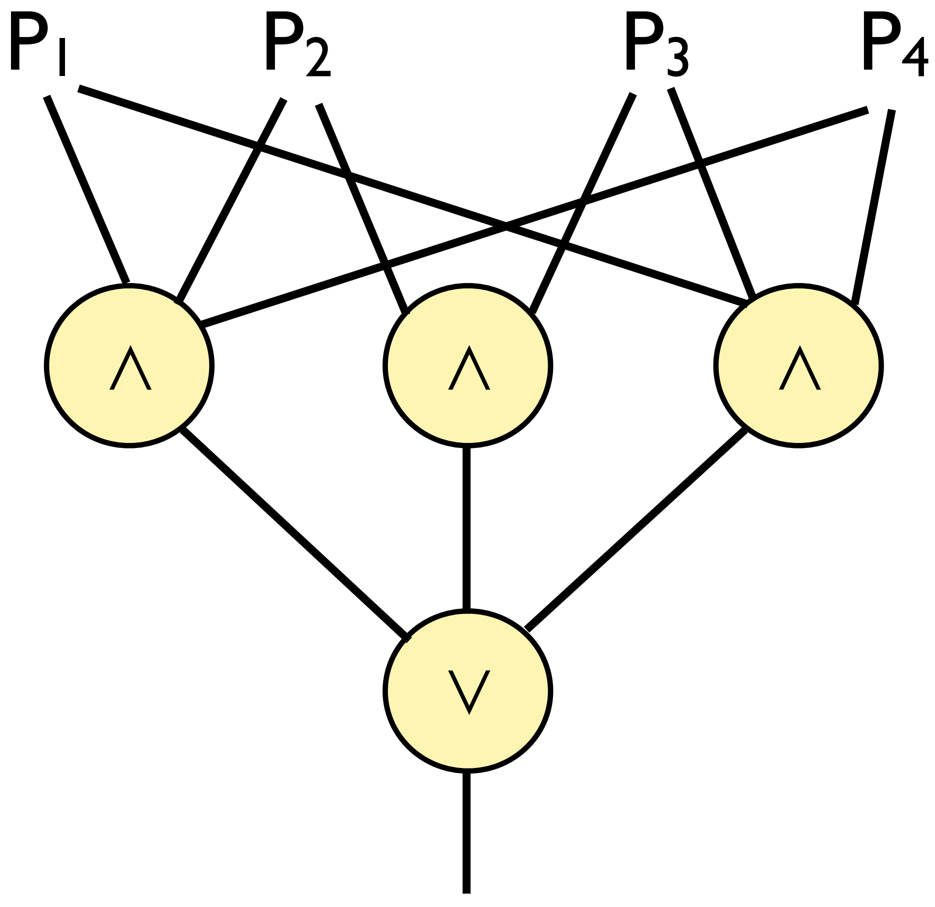
- Given an access structure AS, we want a perfect secret sharing scheme realizing AS
  - We use a Boolean circuit corresponding to AS
  - And a secret-splitting scheme
    - e.g., the $\oplus$-based scheme

Boolean Circuit for AS

- Inputs to the circuit:
  - a wire for every element of P
- Output of the circuit:
  - whether the set of elements that are given a 1 on input is a member of AS
- Can be constructed from the "minimal elements" of AS

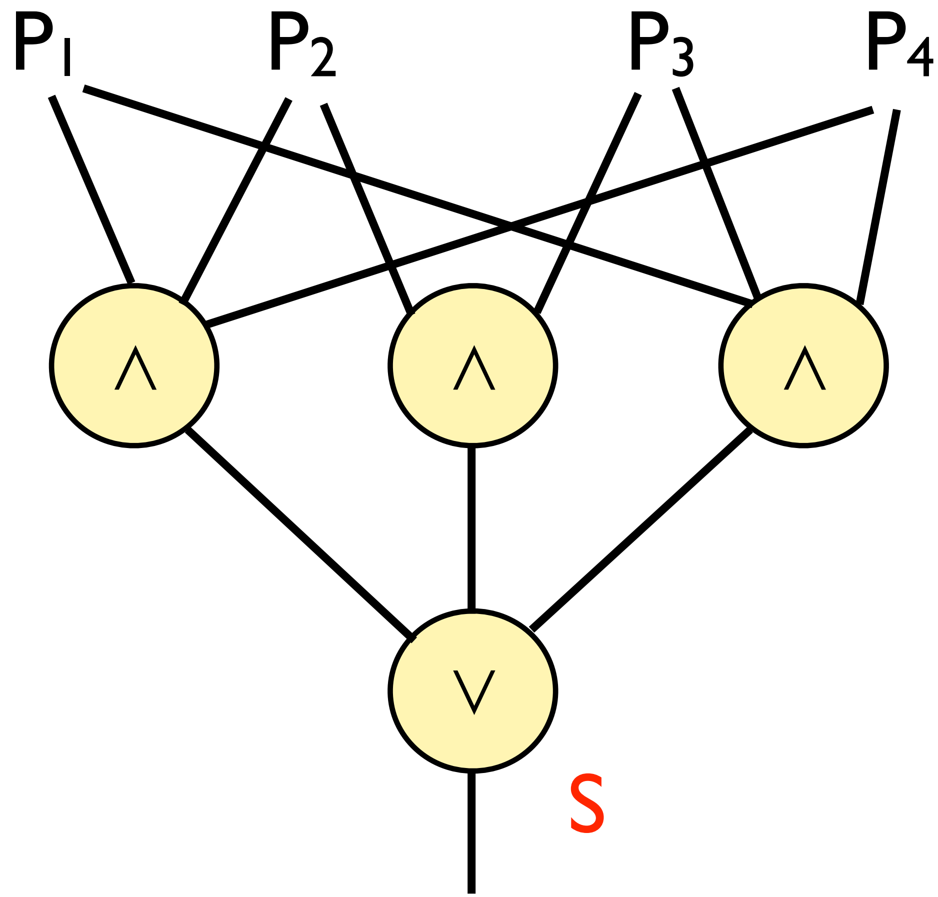
Example Circuit

- $P = \{P_1, P_2, P_3, P_4\}$
- AS with min elts { {P1,P2,P4}, {P1,P3,P4}, {P2,P3} }



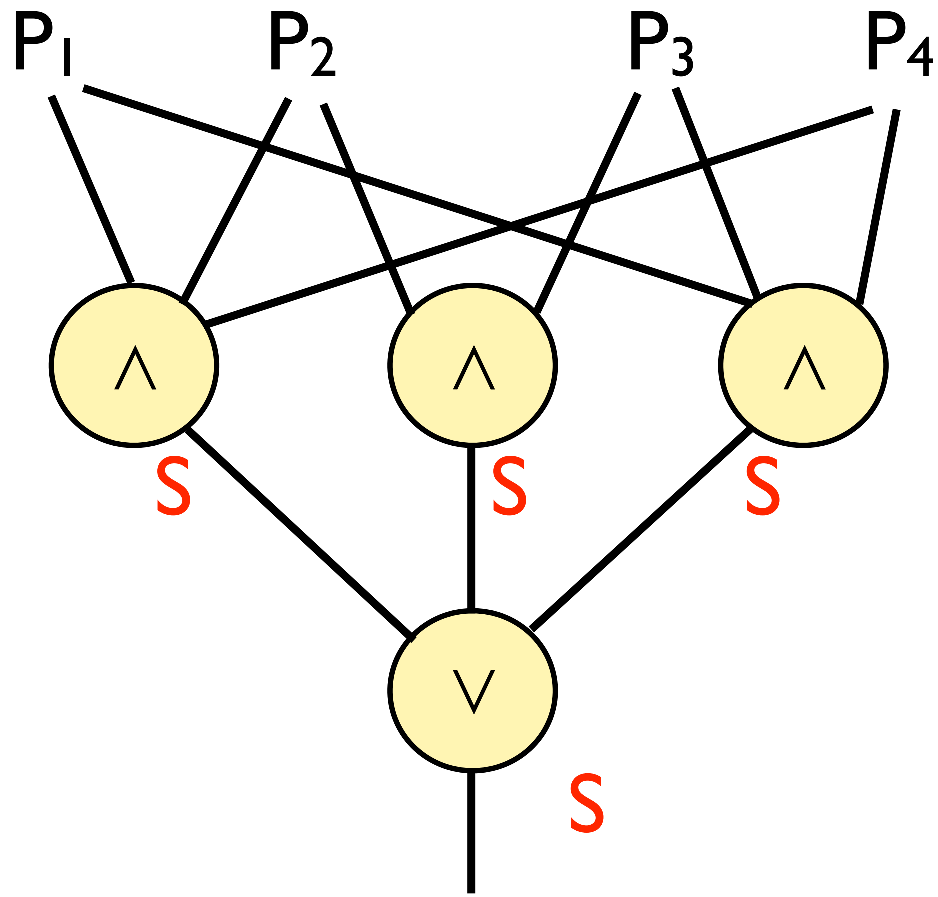
The Scheme

- Given a secret S as a bitstring in $\{0,1\}^m$
- First set output wire of circuit to be S



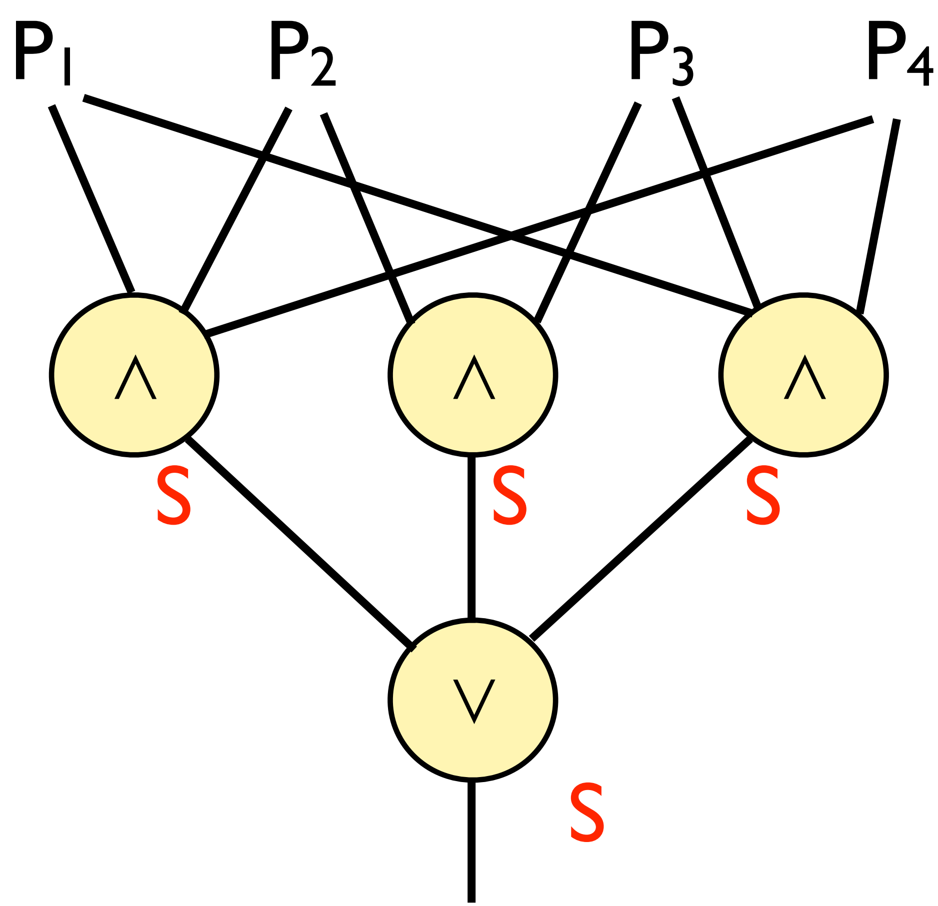
The Scheme

- Then duplicate secret back through a ∨ node



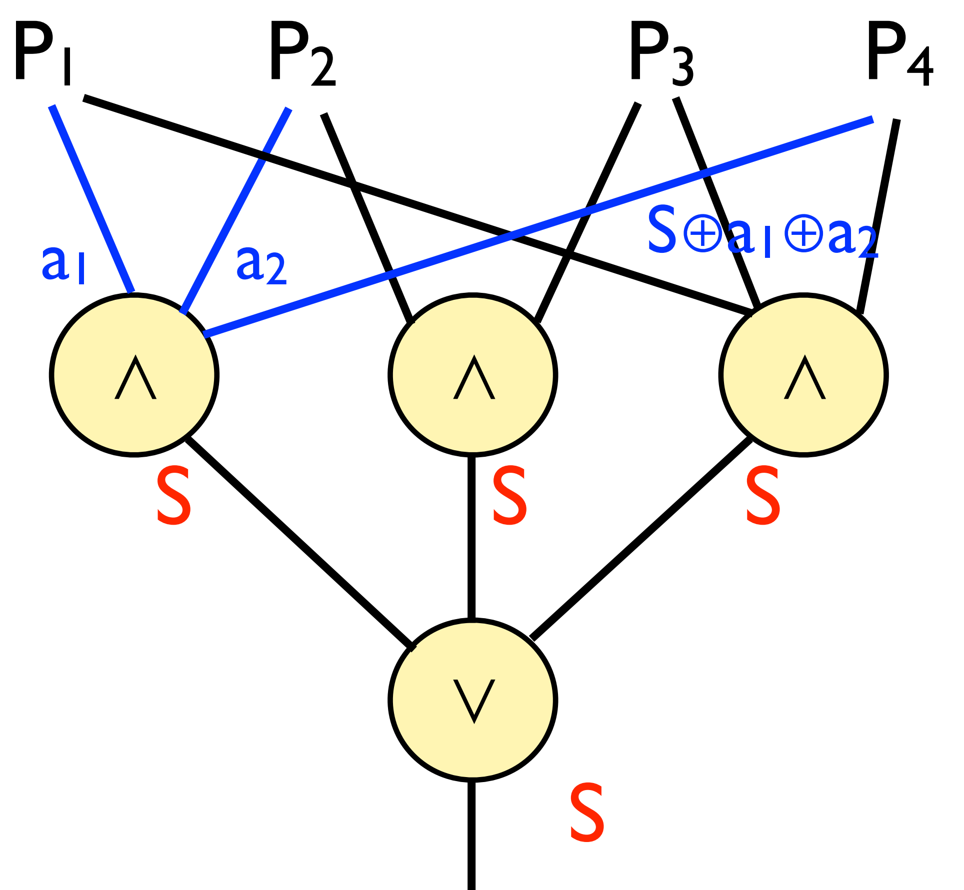
The Scheme

- For every $\wedge$ node, do a (T,T) secret-splitting of the output of the node among the inputs of the node



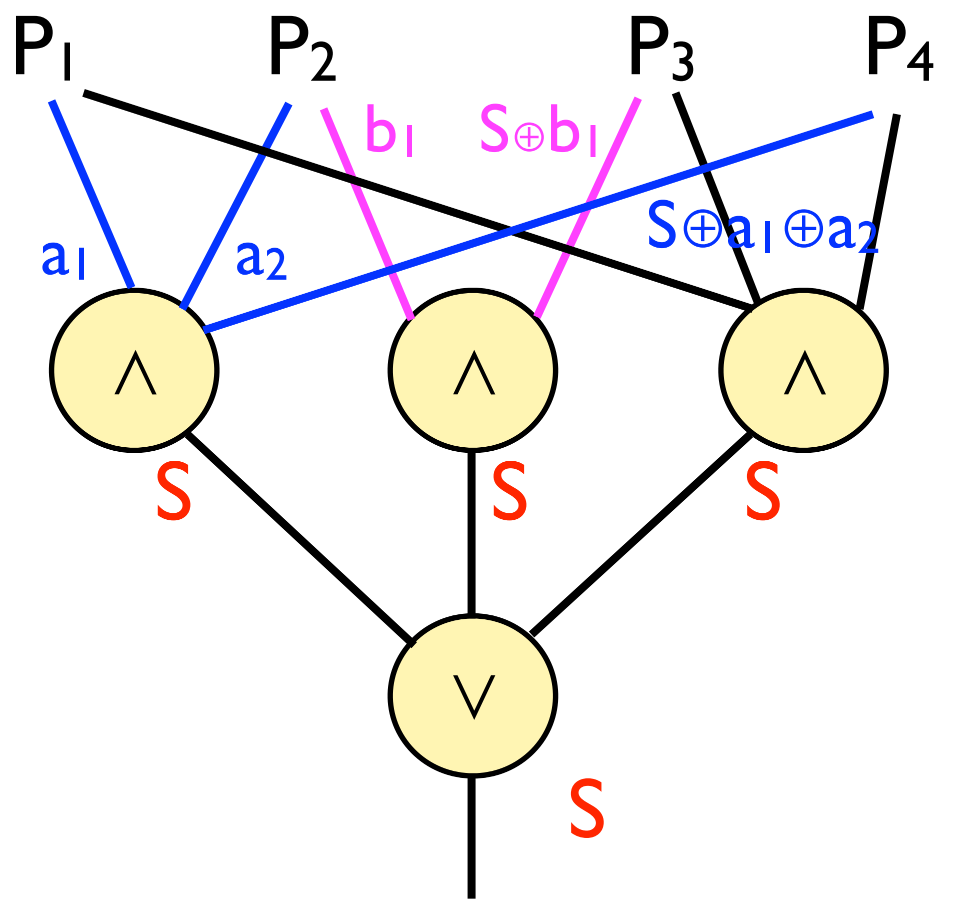
The Scheme

- For every $\wedge$ node, do a (T,T) secret-splitting of the output of the node among the inputs of the node



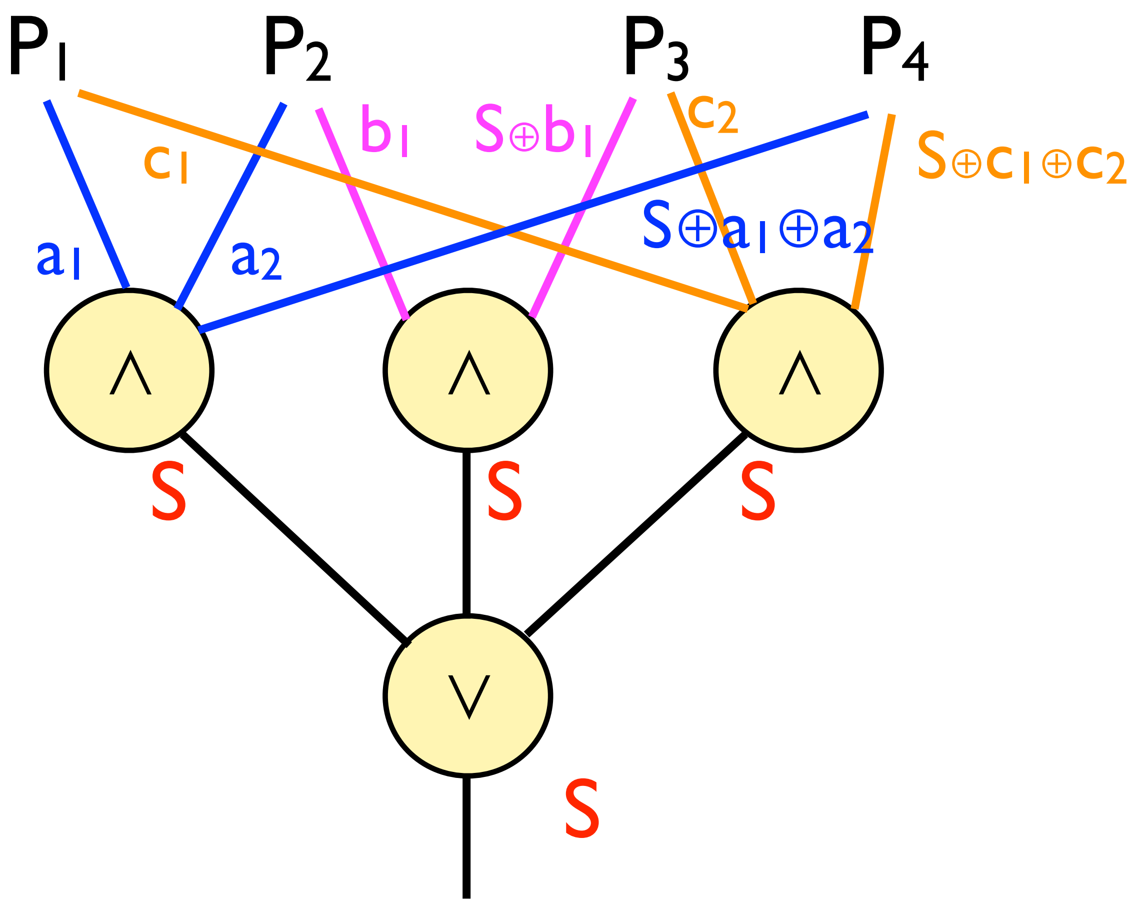
The Scheme

- For every ∧ node, do a (T,T) secret-splitting of the output of the node among the inputs of the node



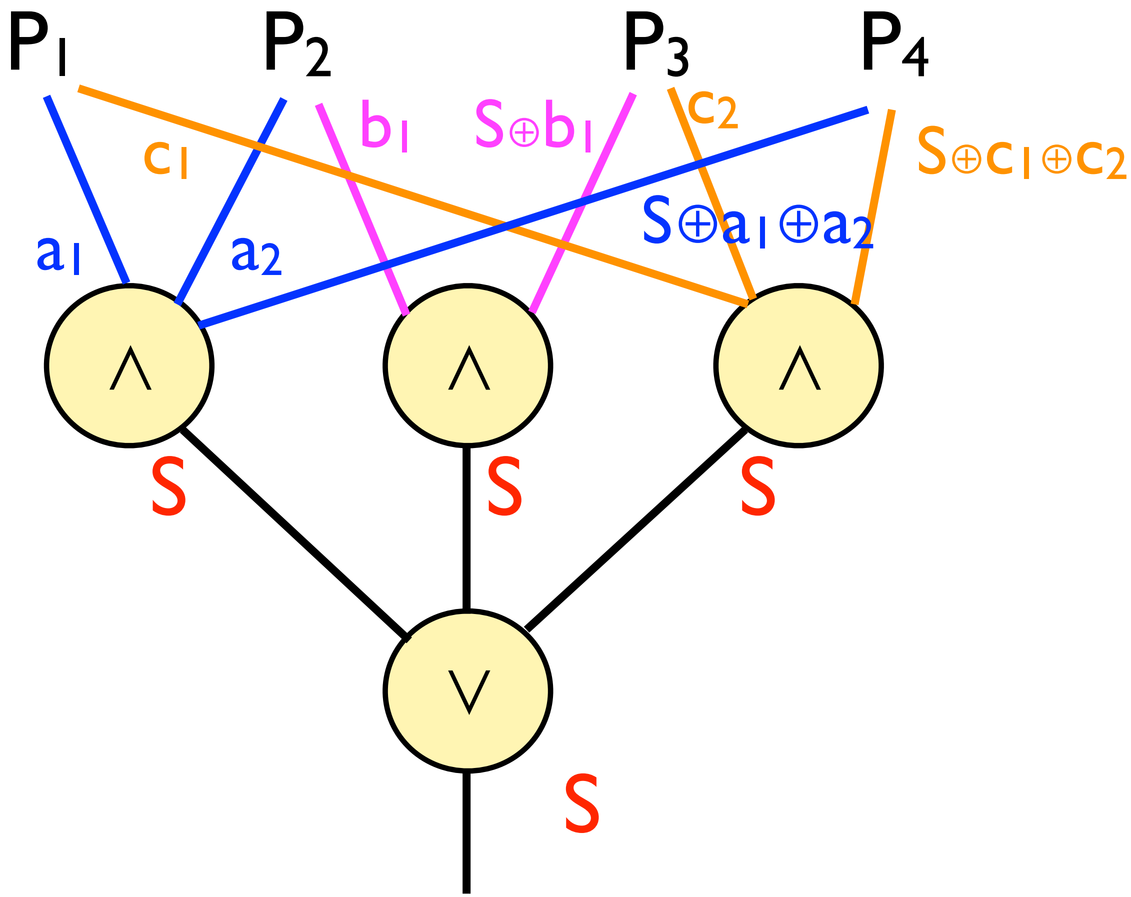
The Scheme

- For every $\wedge$ node, do a (T,T) secret-splitting of the output of the node among the inputs of the node



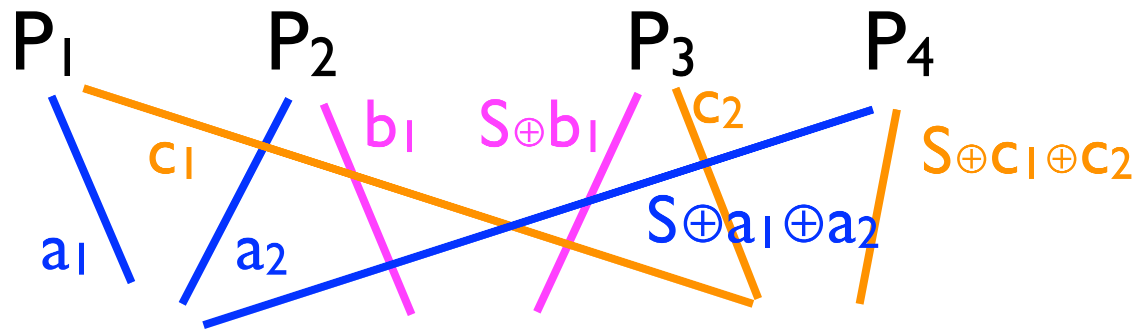
The Scheme

- Give the appropriate shares to each party by looking at the wires out of that party



The Scheme

- Give the appropriate shares to each party by looking at the wires out of that party

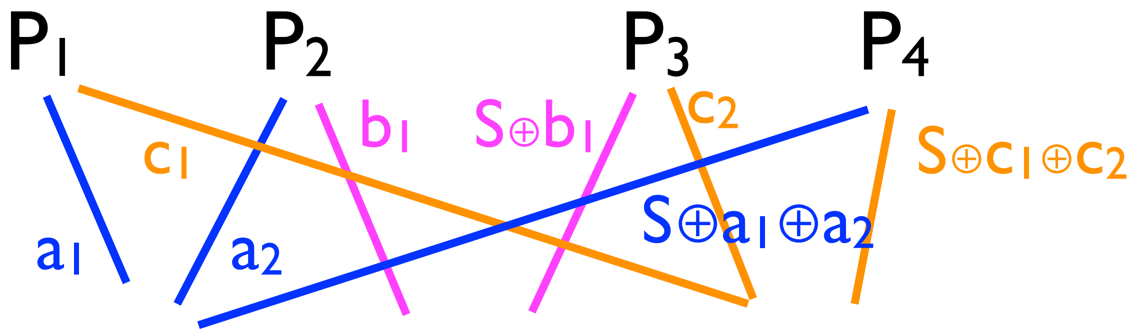


P_1 gets $\{ a_1, c_1 \}$ P_2 gets $\{ a_2, b_1 \}$

P_3 gets $\{ S \oplus b_1, c_2 \}$ P_4 gets $\{ S \oplus a_1 \oplus a_2, S \oplus c_1 \oplus c_2 \}$

The Scheme

- Give the appropriate shares to each party by looking at the wires out of that party



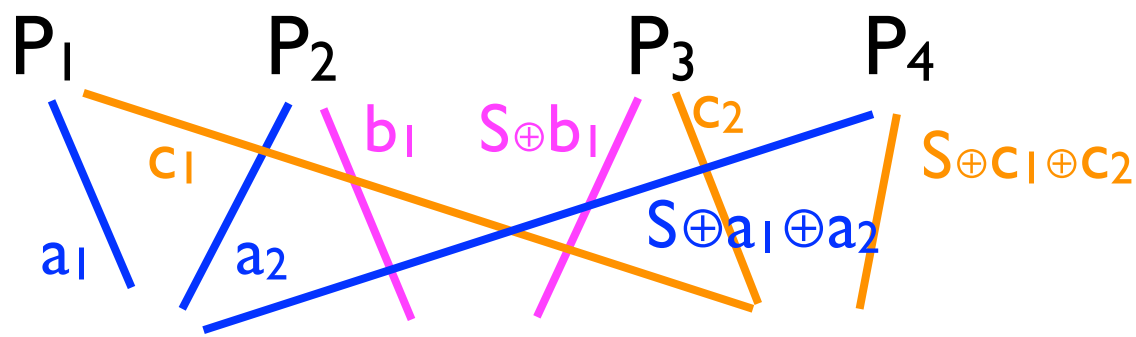
P_1 gets { a_1, c_1 } P_2 gets { a_2, b_1 }

P_3 gets { $S \oplus b_1$, c_2 } P_4 gets { $S \oplus a_1 \oplus a_2$, $S \oplus c_1 \oplus c_2$ }

CHECK: This is a perfect secret sharing scheme

The Scheme

- Give the appropriate shares to each party by looking at the wires out of that party



P_1 gets { a_1, c_1 }

P_2 gets { a_2, b_1 }

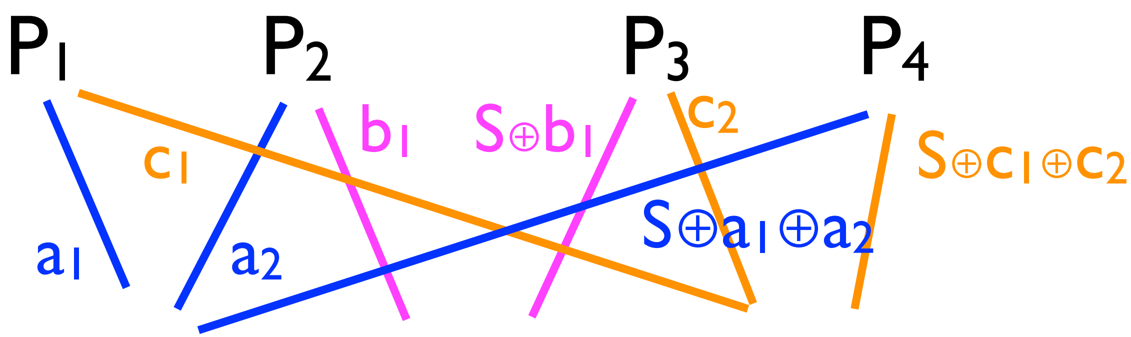
P_3 gets { $S \oplus b_1$, c_2 }

P_4 gets { $S \oplus a_1 \oplus a_2$, $S \oplus c_1 \oplus c_2$ }

CHECK: This is a perfect secret sharing scheme

The Scheme

- Give the appropriate shares to each party by looking at the wires out of that party



P_1 gets { a_1, c_1 } P_2 gets { a_2, b_1 }

P_3 gets { $S \oplus b_1$, c_2 } P_4 gets { $S \oplus a_1 \oplus a_2$, $S \oplus c_1 \oplus c_2$ }

CHECK: This is a perfect secret sharing scheme