Theory and Practice of Natural Computing 2013
December 5, Cáceres, Spain

Securely Computing the Three-Input Majority Function with Eight Cards

Takuya Nishida, Takaaki Mizuki, Hideaki Sone
Tohoku University

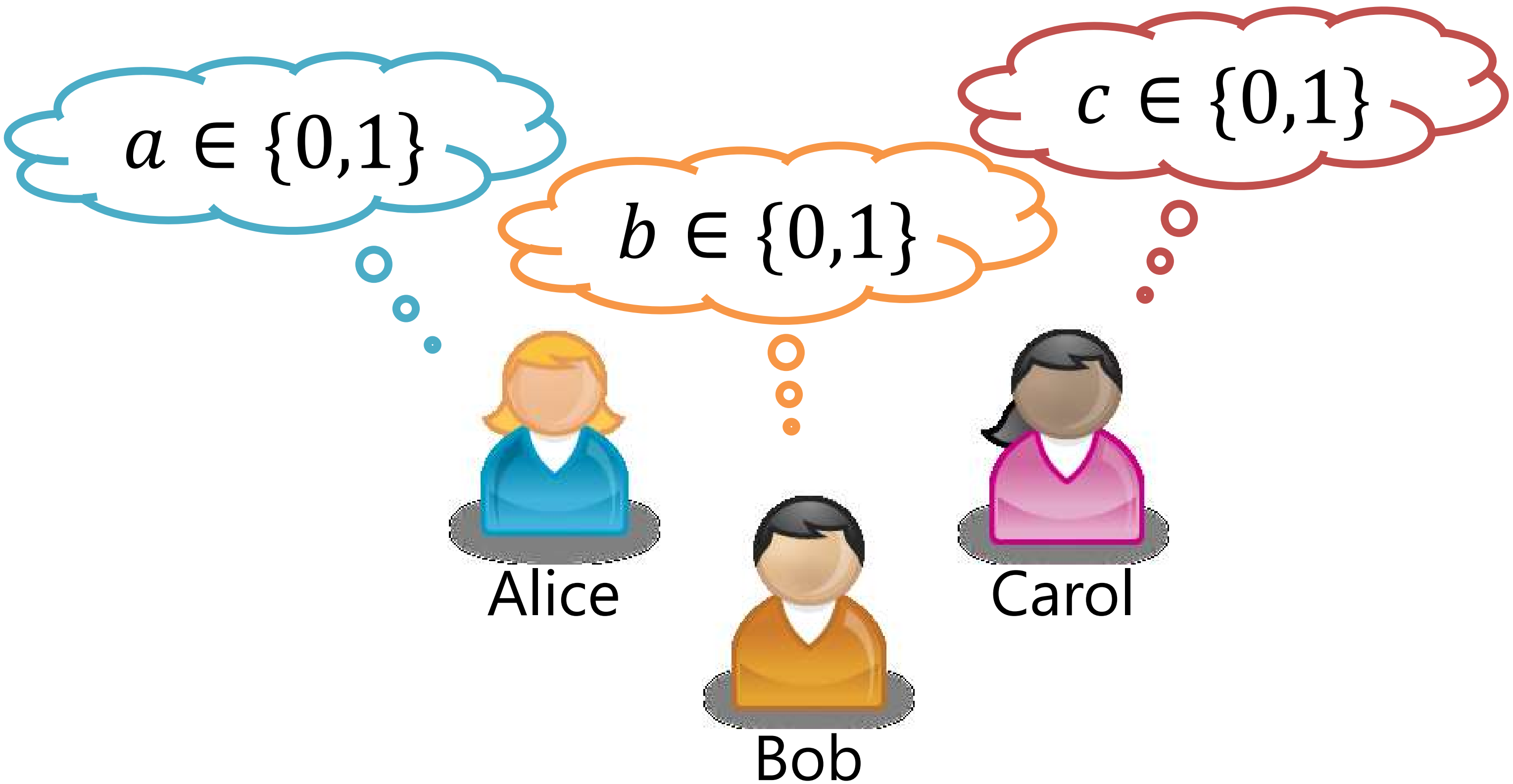
Contents

Contents

1. **Introduction**
   - Card-Based Cryptographic Protocols
   - Our Results
2. K
3. Straightforward
   Secure Majority Computations
4. An Improved Secure Majority Protocol
5. Conclusion

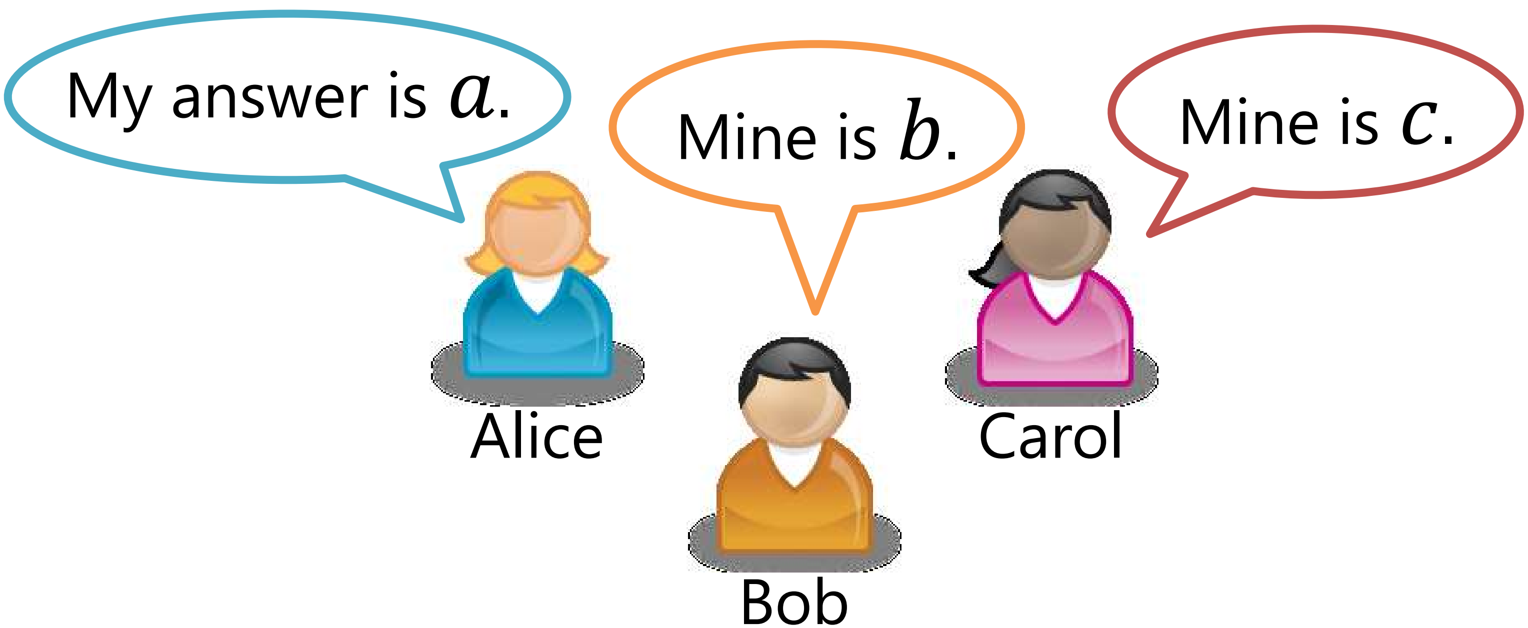
Introduction

- Assume that there are 3 players, Alice, Bob and Carol, who **privately** hold 1-bit inputs, respectively.



Introduction

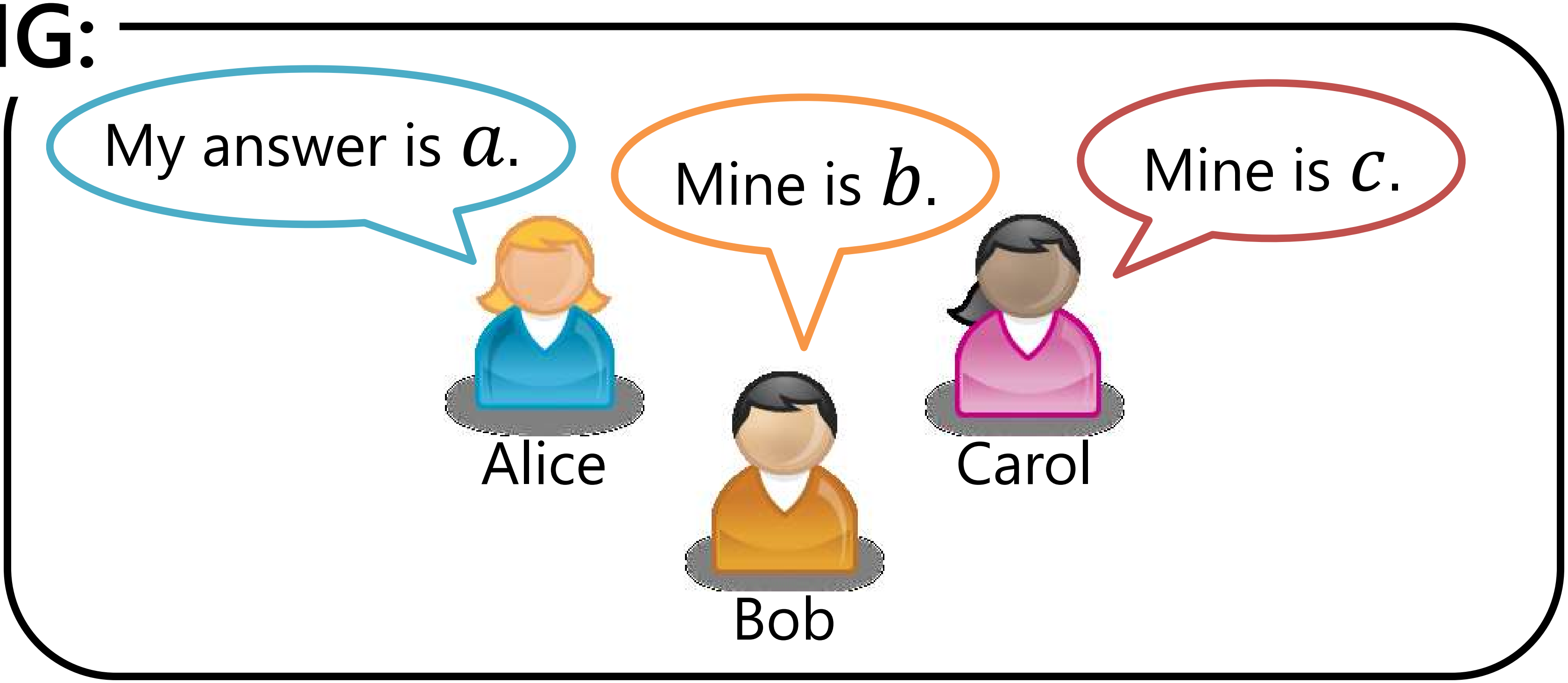
- Majority voting is often conducted to find the majority of their answers **by revealing all of the inputs**.



Introduction

- Each of them does **not** want to **let** the other 2 players **know** his/her answer.

NG:



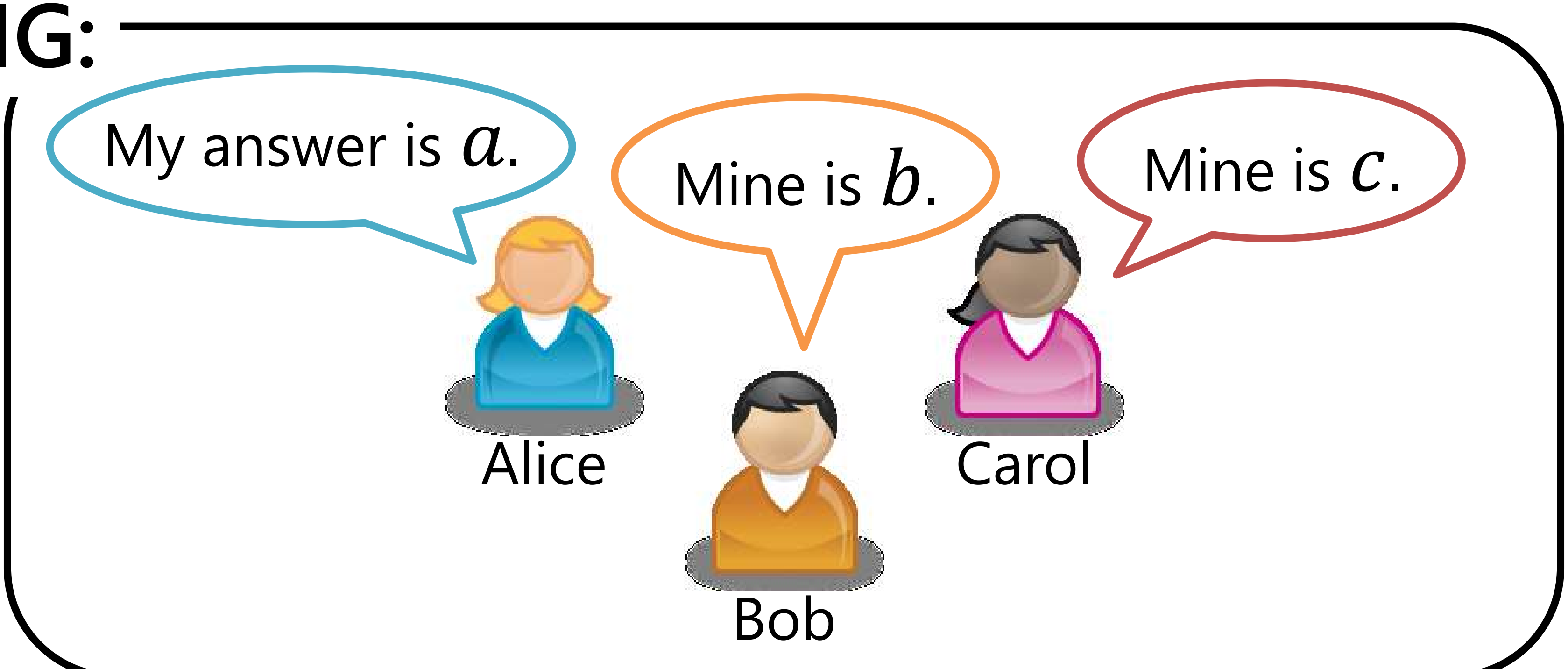
Introduction

- They all want to learn the value of the **majority function**

$$\mathrm{maj}(a, b, c)$$

without revealing more of their own secret inputs than is necessary.

NG:



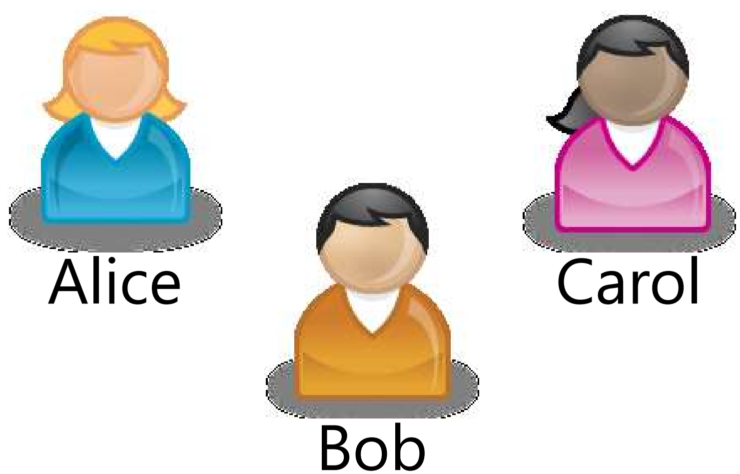
Introduction

- That is, they wish to know **only** the value of

$$\mathrm{maj}(a,b,c) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } a+b+c \geq 2 \\ 0 & \text{if } a+b+c \leq 1 \end{cases}$$

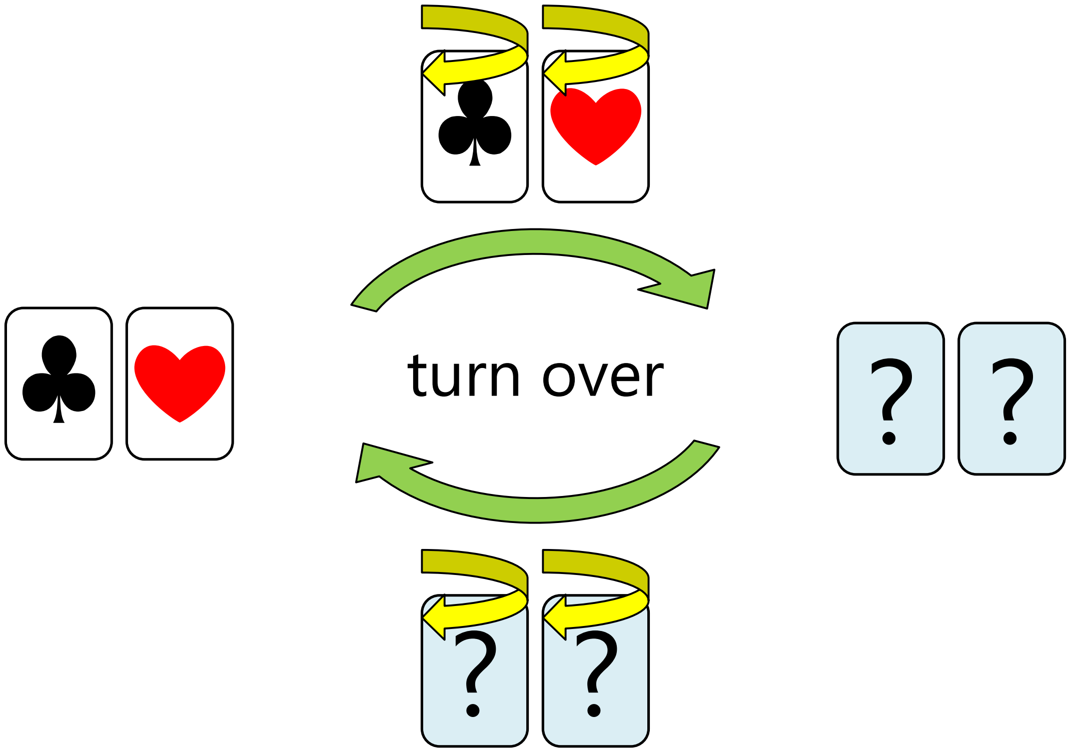
- Note that

$$\mathrm{maj}(a,b,c) = (a \wedge b) \vee (b \wedge c) \vee (c \wedge a)$$



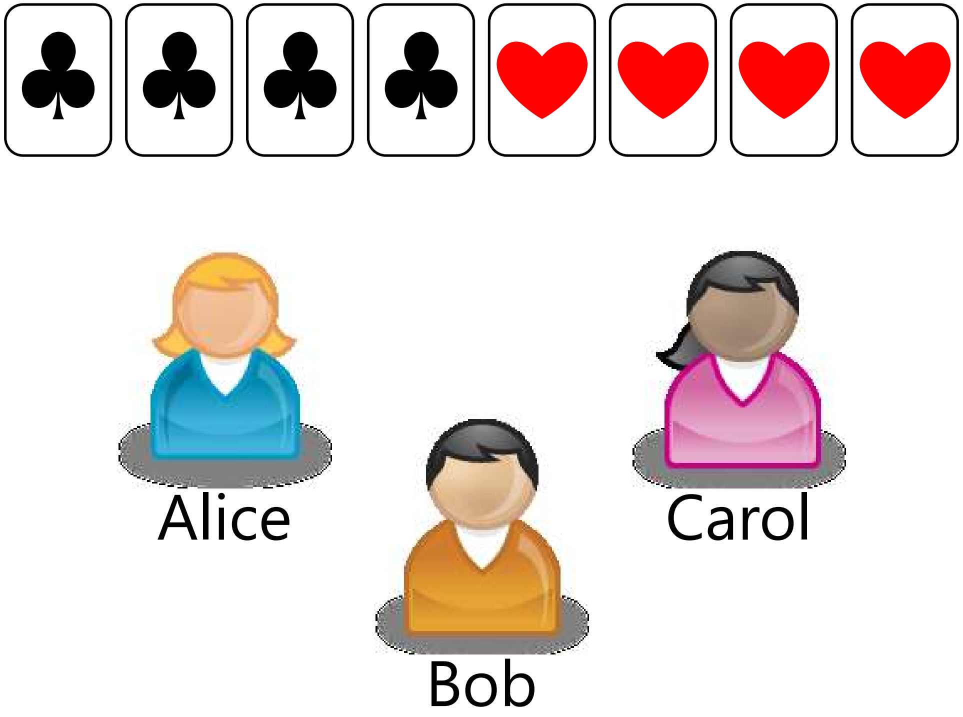
Introduction

- Such a **secure majority computation** can be conducted using a deck of **real** cards.



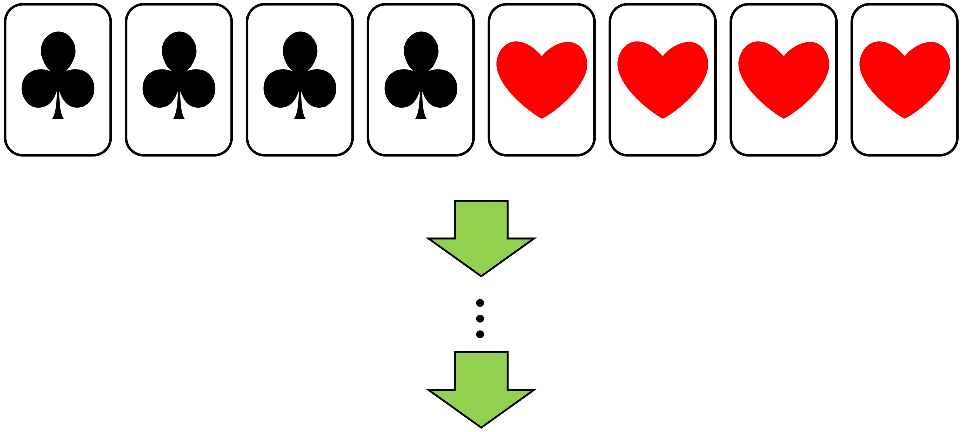
Introduction

- We show that the 3 players can learn only $\mathrm{maj}(a, b, c)$ using **8 physical cards**.



Introduction

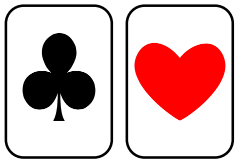
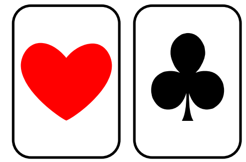
- Our secure majority computation is a kind of **card-based cryptographic protocol**.



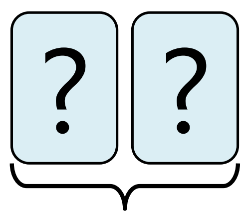
$$\mathrm{maj}(a, b, c)$$

Card-Based Cryptographic Protocols

- To deal with Boolean values, we use the following encoding:

 $= 0$  $= 1$

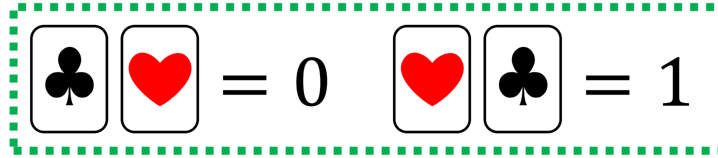
- Given a bit $x \in \{0,1\}$, a pair of face-down cards whose value is equal to x is called a **commitment** to x and is expressed as

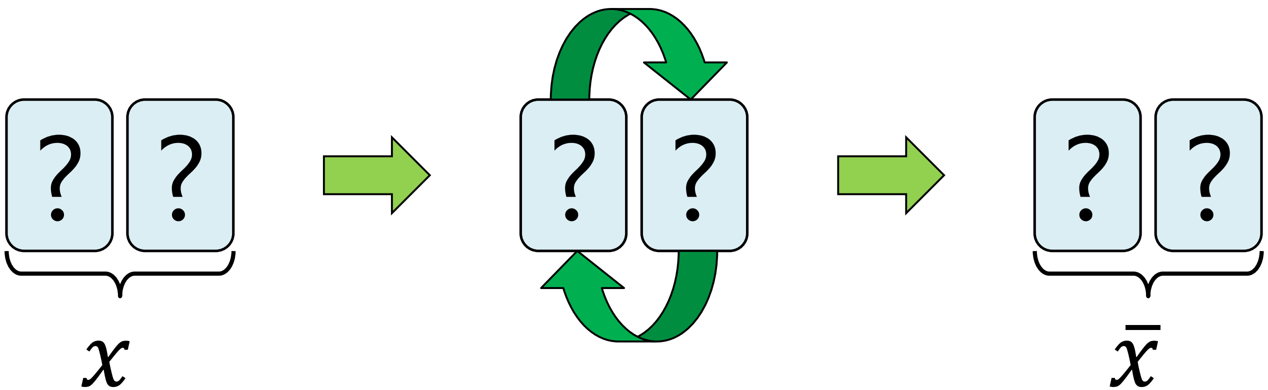


x

Card-Based Cryptographic Protocols

- Swapping the 2 cards constituting a commitment to a bit x results in a commitment to negation $\bar{x}$:

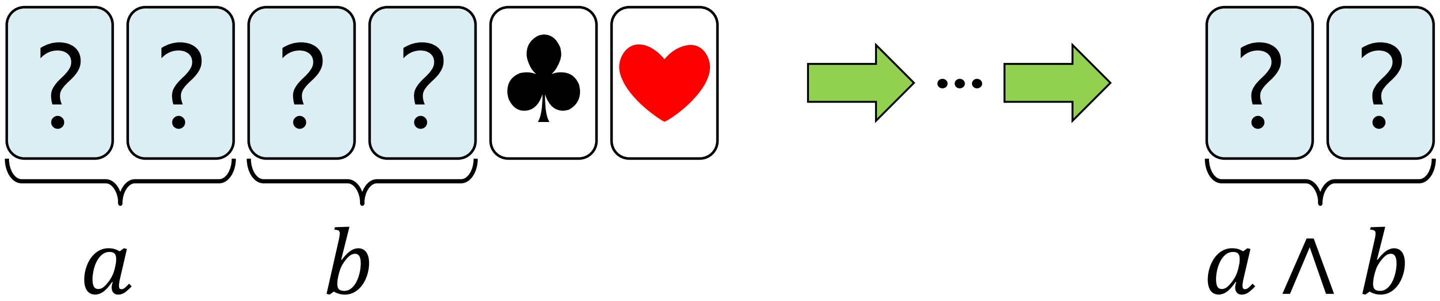




- Thus, a secure **NOT** operation is trivial.

Card-Based Cryptographic Protocols

- There are some secure **AND** protocols.

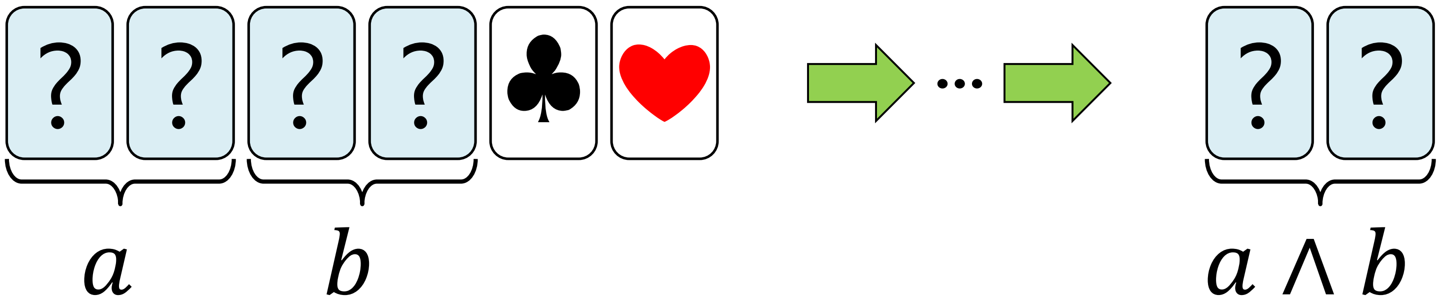


○ History of Secure AND protocols

	# of colors	# of cards	Avg. # of trials
Crépeau-Kilian [CRYPTO '93]	4	10	6
Niemi-Renvall [TCS 1998]	2	12	2.5
Stiglic [TCS 2001]	2	8	2
Mizuki-Sone [FAW 2009]	2	6	1

Card-Based Cryptographic Protocols

- There are some secure **AND** protocols.



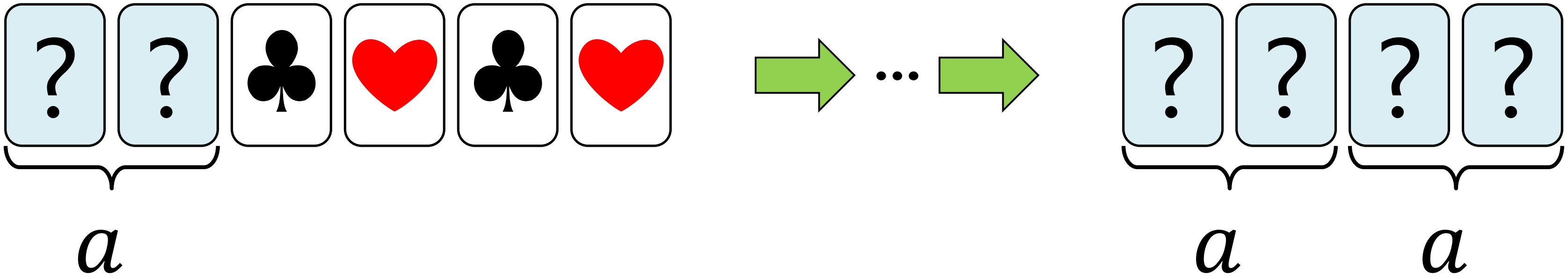
○ History of Secure AND protocols

	# of colors	# of cards	Avg. # of trials
Crépeau-Kilian [CRYPTO '93]	4	10	6
[illegible] [TCS 1998]	2	12	2.5
Stiglic [TCS 2001]	2	8	2
Mizuki-Sone [FAW 2009]	2	6	1

Will be introduced

Card-Based Cryptographic Protocols

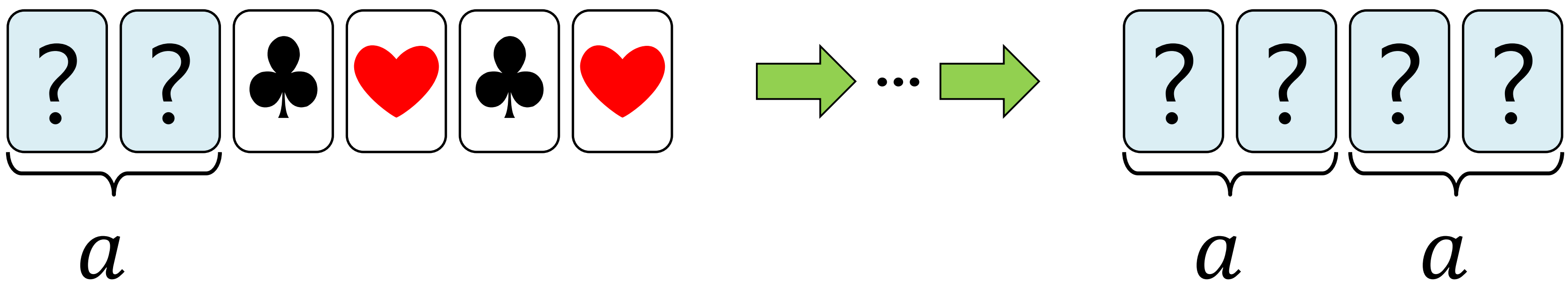
- There are also **copy** protocols.



Card-Based Cryptographic Protocols

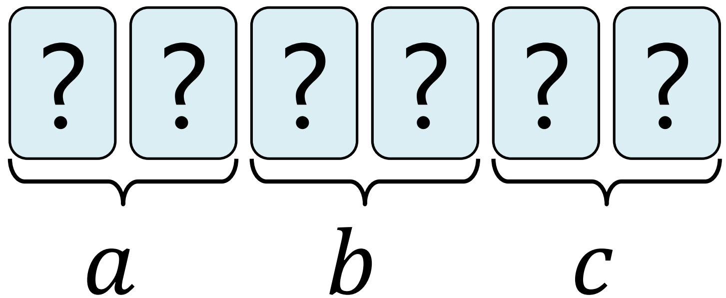
- There are also **copy** protocols.

One of which is introduced

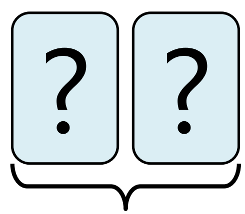


Our Results

- Recall our **goal**: given 3 commitments,



we desire to obtain a commitment to

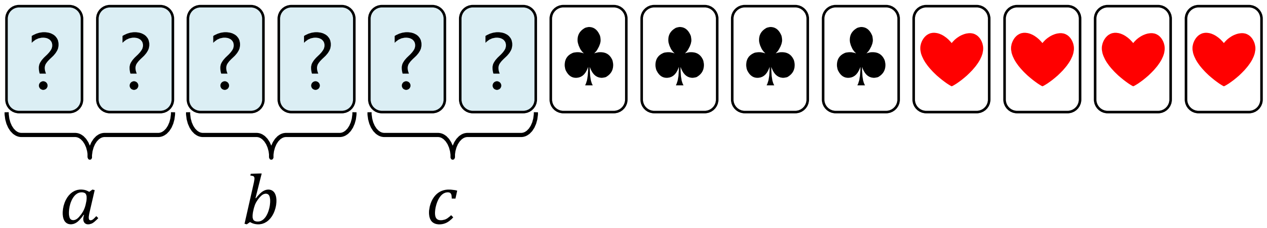


$$\mathrm{maj}(a, b, c)$$

Our Results

Section 2

- Using the existing protocols, we can construct a **trivial** protocol for $\mathrm{maj}(a, b, c)$ by using 14 cards.

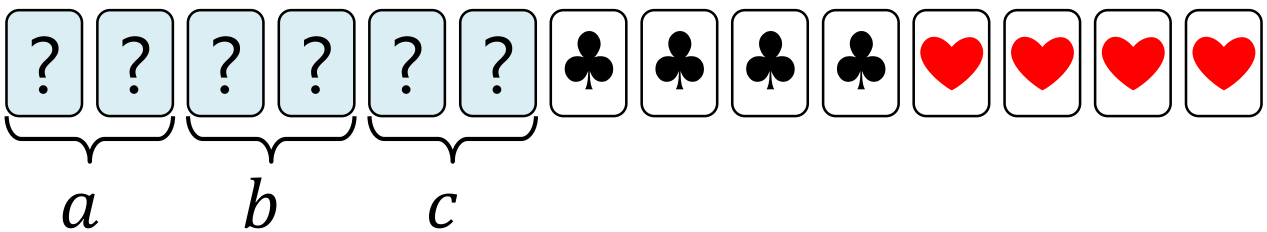


Our Results

Section 2

- Using the existing protocols, we can construct a **trivial** protocol for $\mathrm{maj}(a,b,c)$ by using 14 cards.

Section 3

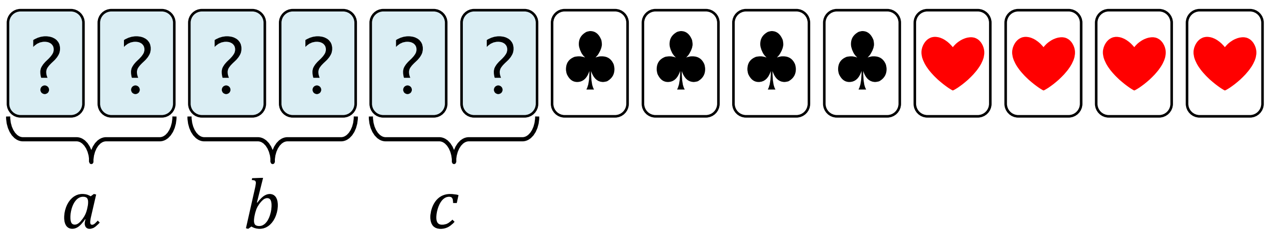


Our Results

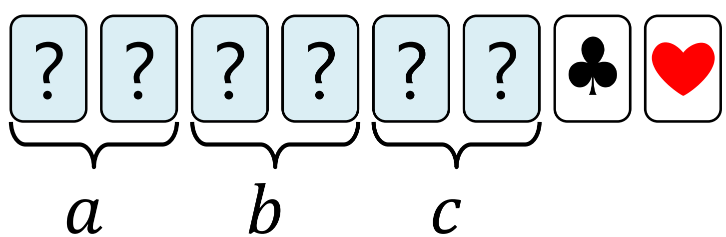
Section 2

- Using the existing protocols, we can construct a **trivial** protocol for $\mathrm{maj}(a, b, c)$ by using 14 cards.

Section 3



- We give a **non-trivial** protocol using **only 8 cards**.

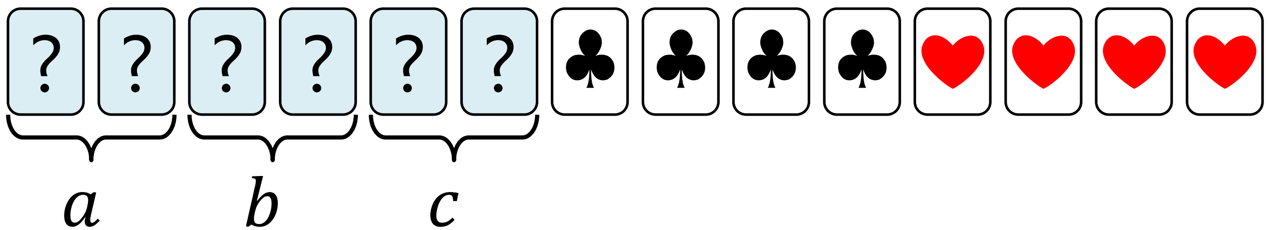


Our Results

Section 2

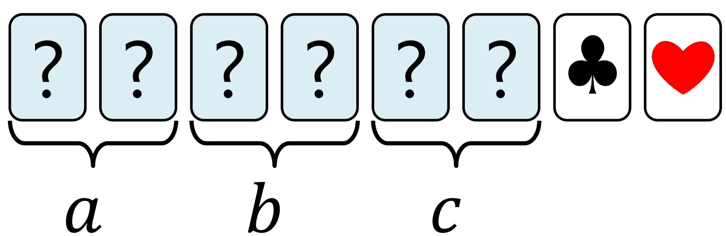
- Using the existing protocols, we can construct a **trivial** protocol for $\mathrm{maj}(a, b, c)$ by using 14 cards.

Section 3



- We give a **non-trivial** protocol using **only 8 cards**.

Section 4



Contents

Contents

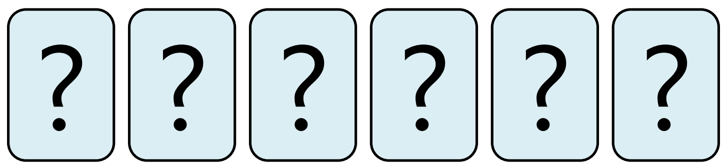
1. Introduction
2. **Known Protocols**
3. St…

   Se…
4. A… otocol
5. Conclusion

- Random Bisection Cuts
- The Six-Card AND Protocol
- The Copy Protocol
  with a Random Bisection Cut

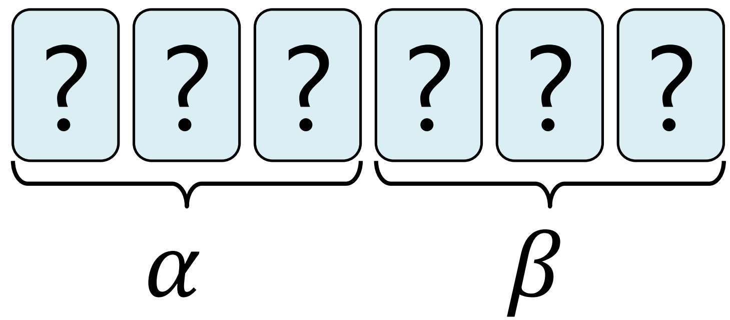
Random Bisection Cuts

- Assume that there are 6 cards as follows:



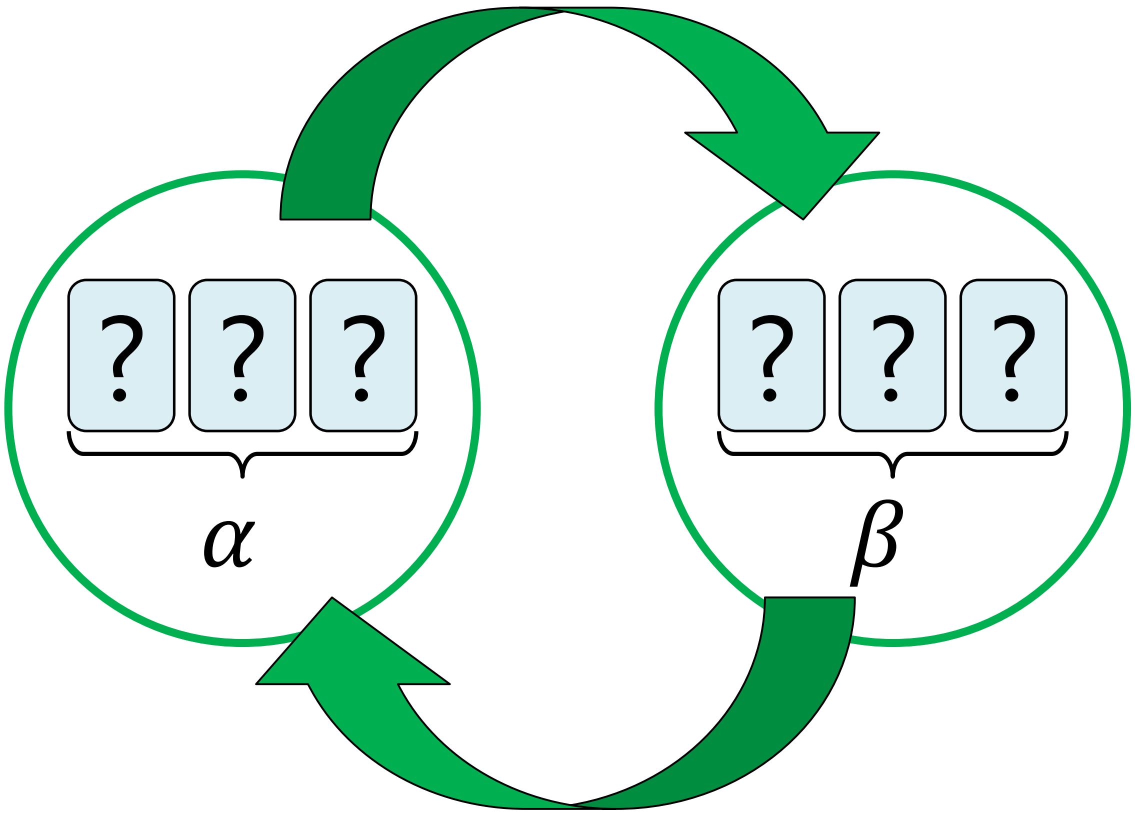
Random Bisection Cuts

- Bisect the deck of cards, and let the 2 sections be α and β :

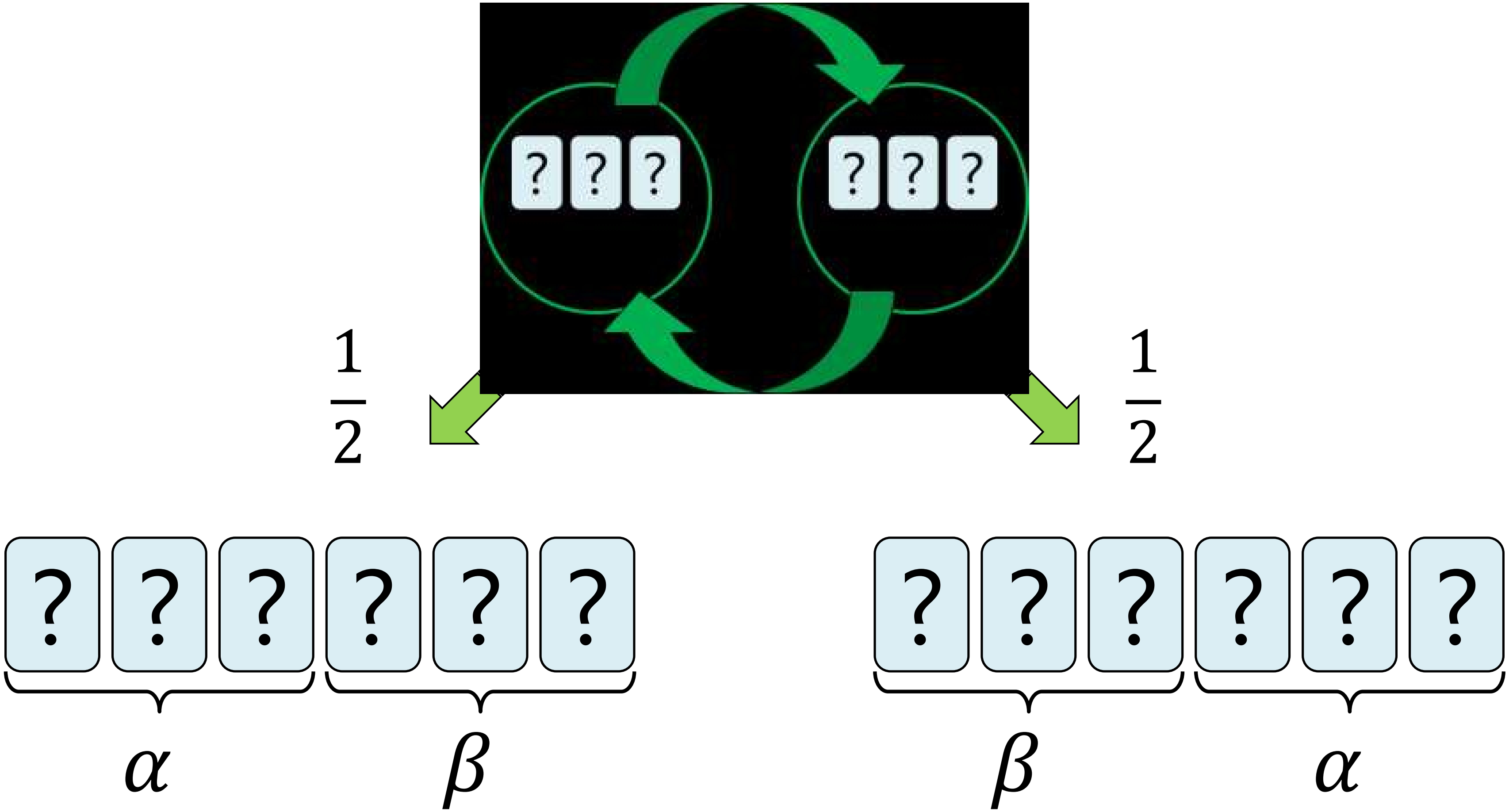


Random Bisection Cuts

- Shift α and β **randomly**:

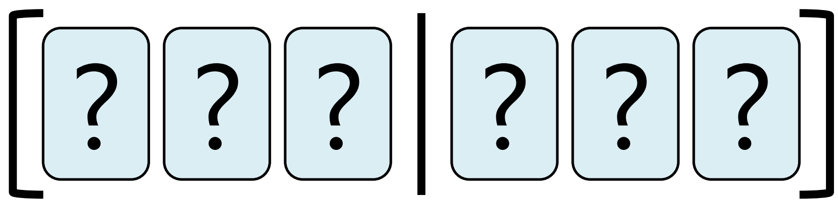


Random Bisection Cuts

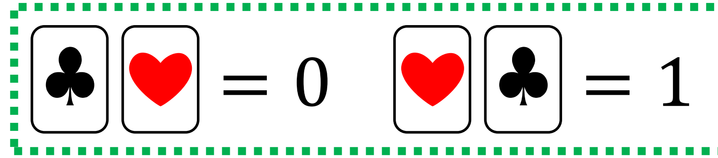


Random Bisection Cuts

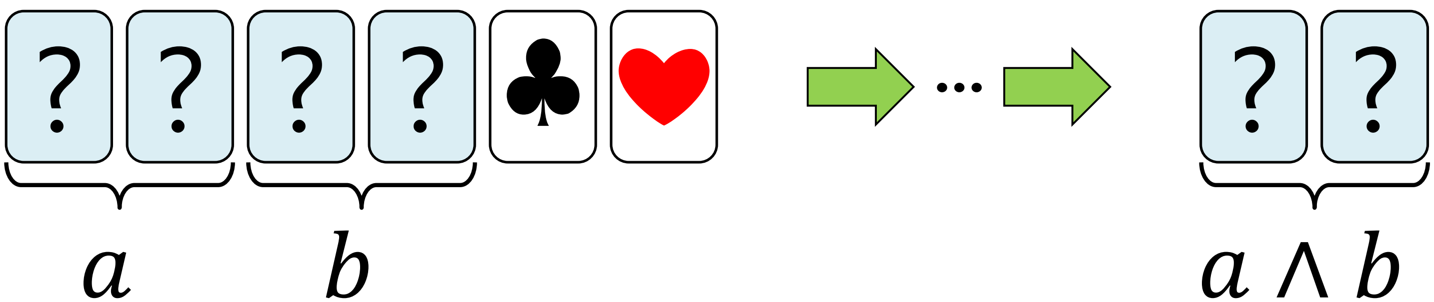
- The expression of a random bisection cut for 6 cards:



The Six-Card AND Protocol



- Using the random bisection cut, we can construct a 6-card AND protocol [8].



[8] Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: FAW 2009, LNCS 5598, pp. 358–369. (2009)

The Six-Card AND Protocol

- Before going into the details, we define 2 operations, **get** and **shift**. Given a pair of bits (x, y),

$$\mathrm{get}^0(x, y) = x$$

$$\mathrm{get}^1(x, y) = y$$

$$\mathrm{shift}^0(x, y) = (x, y)$$

$$\mathrm{shift}^1(x, y) = (y, x)$$

The Six-Card AND Protocol

returns the first bit

$$\mathrm{get}^0(x, y) = x$$

$$\mathrm{get}^1(x, y) = y$$

returns the second bit

The Six-Card AND Protocol

returns the 2 bits identically

$$\mathrm{shift}^{0}(x, y) = (\underline{x, y})$$

$$\mathrm{shift}^{1}(x, y) = (\underline{y, x})$$

swaps the 2 bits

The Six-Card AND Protocol

- Using the **get** and **shift**, the AND function can be written as

$$a \wedge b = \mathrm{get}^{a}(0, b)$$

$$\because a \wedge b = \begin{cases} 0 & \text{if } a = 0 \\ b & \text{if } a = 1 \end{cases}$$

The Six-Card AND Protocol

- Furthermore

$$a \land b = \mathrm{get}^{a}(0, b) = \quad \mathrm{get}^{a}(\mathrm{shift}^{0}(0, b))$$

$$= \mathrm{get}^{a \oplus 1}\left(\mathrm{shift}^{1}(0, b)\right)$$

The Six-Card AND Protocol

- Furthermore

$$a \wedge b = \mathrm{get}^{a}(0, b) = \mathrm{get}^{a}(\mathrm{shift}^{0}(0, b))$$

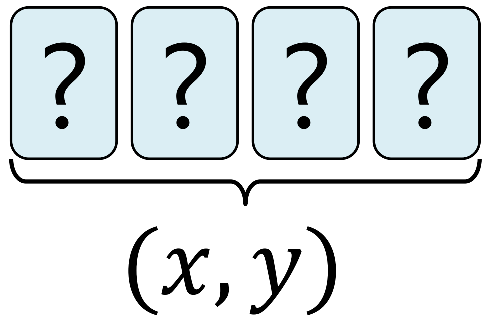
$$= \mathrm{get}^{a \oplus 1}\left(\mathrm{shift}^{1}(0, b)\right)$$

$$\therefore a \wedge b = \mathrm{get}^{a \oplus r}(\mathrm{shift}^{r}(0, b))$$

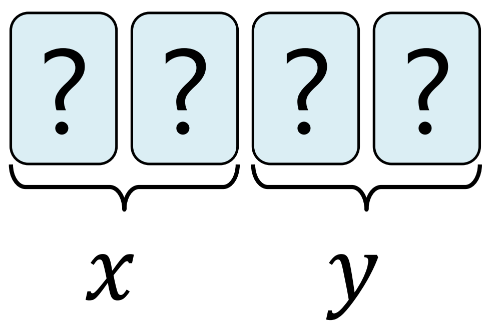
for a random bit $r \in \{0,1\}$.

The Six-Card AND Protocol

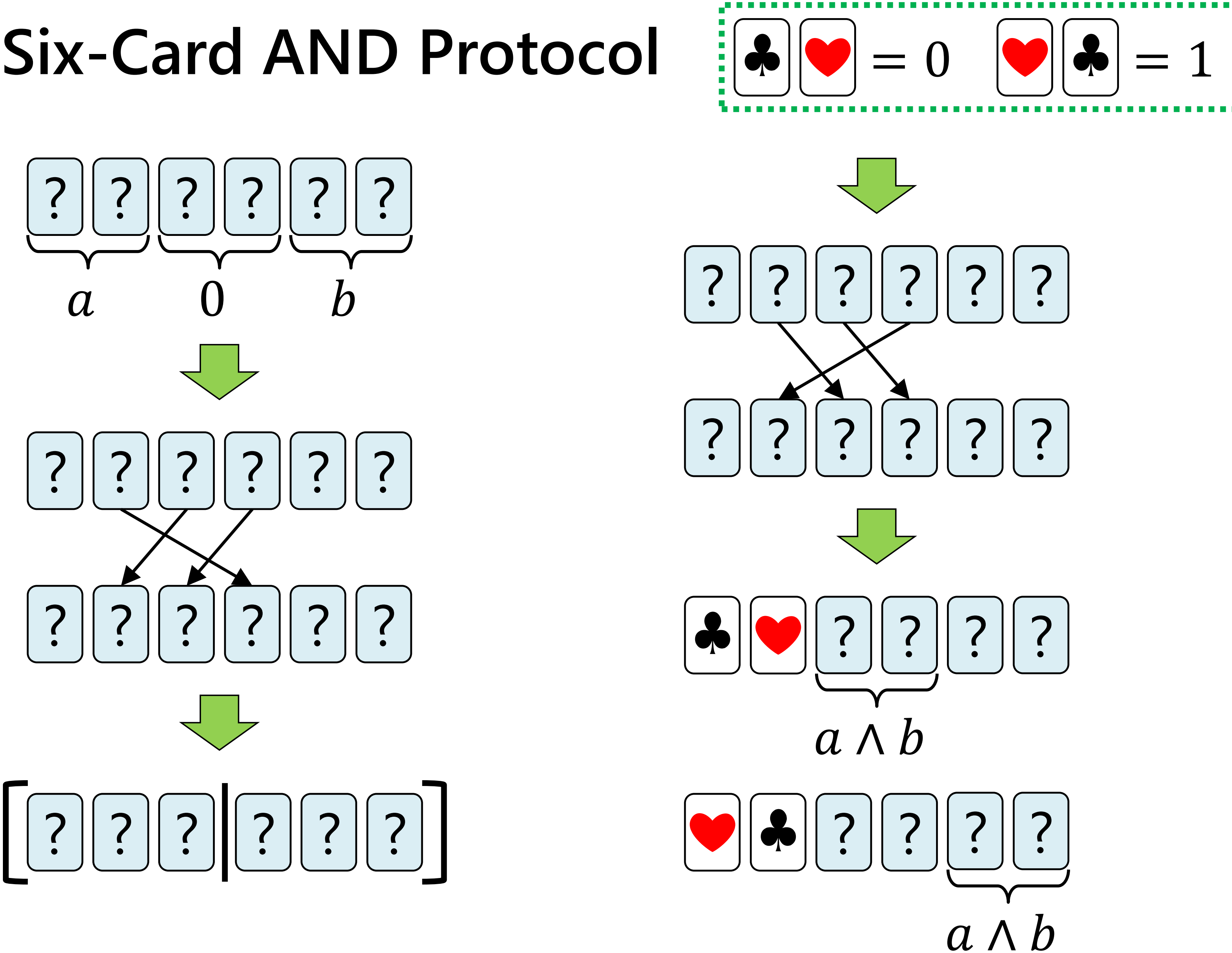
- Hereafter, for 2 bits x and y, the notation



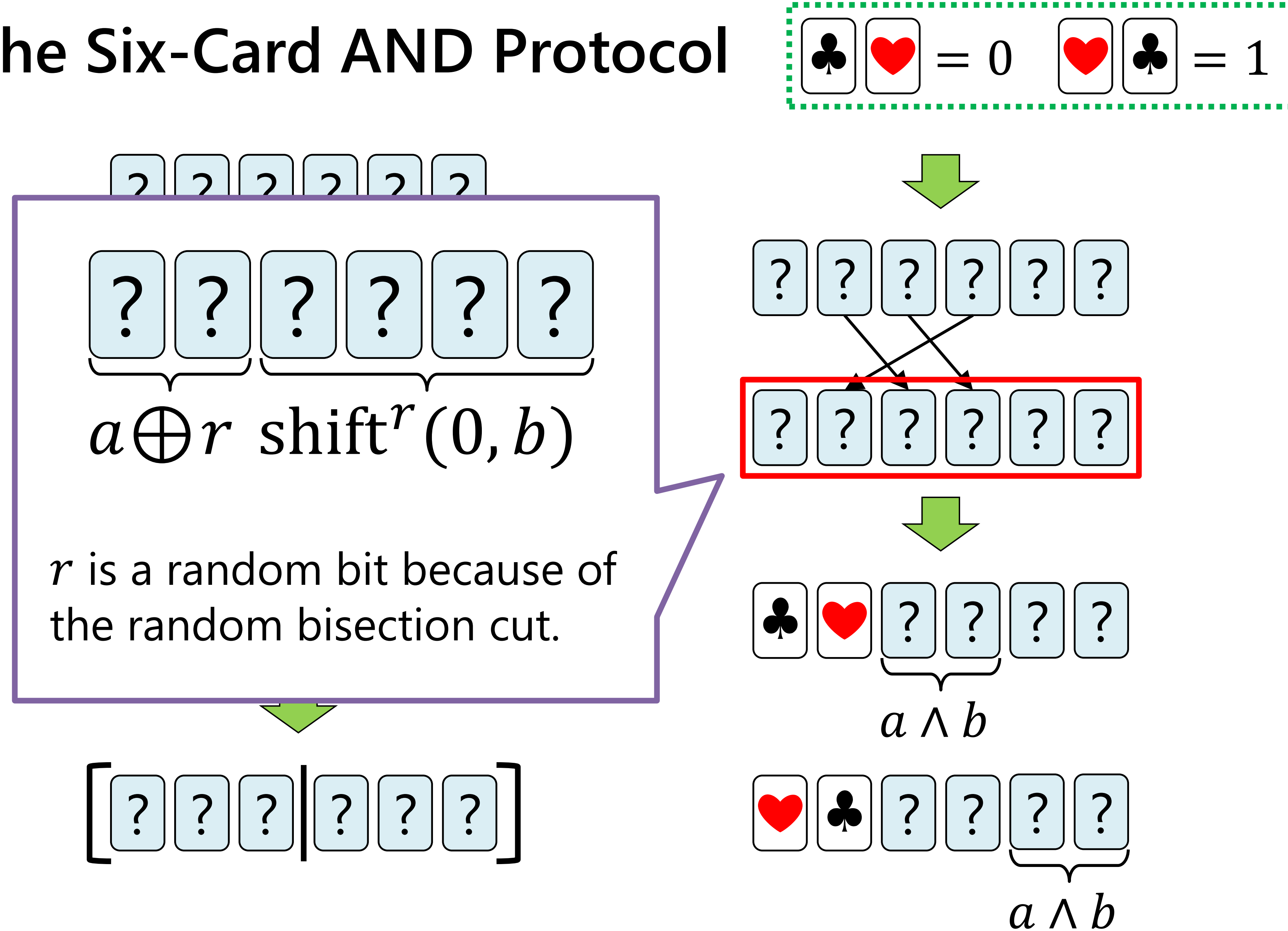
means



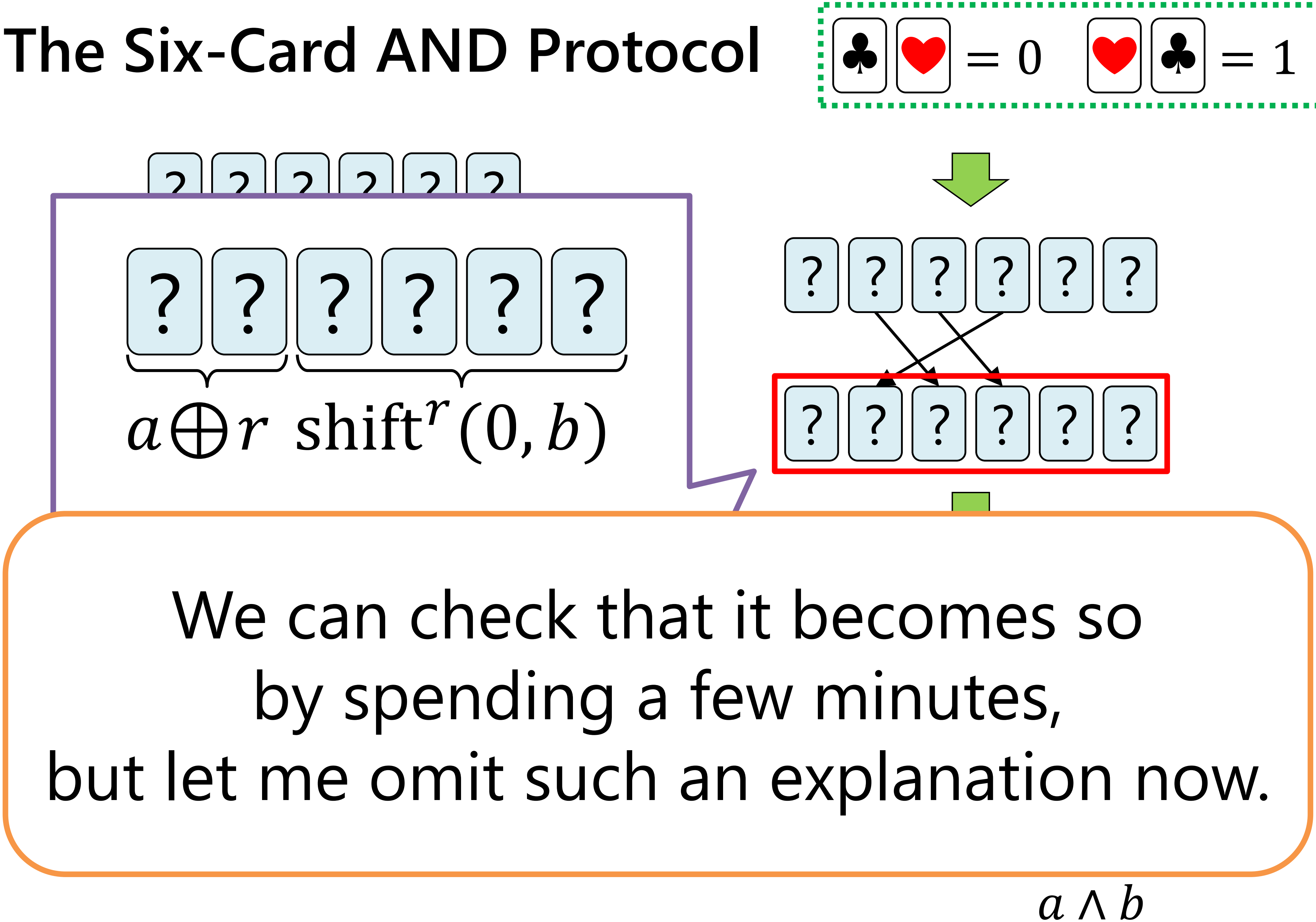
The Six-Card AND Protocol



The Six-Card AND Protocol

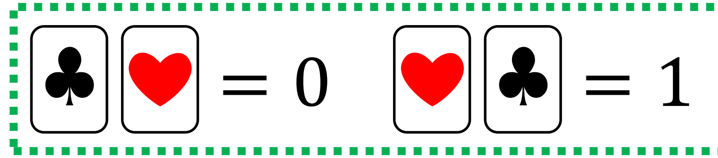


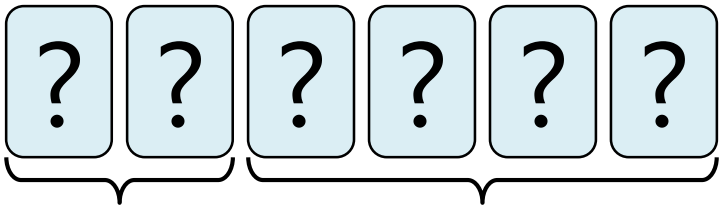
The Six-Card AND Protocol



We can check that it becomes so
by spending a few minutes,
but let me omit such an explanation now.

The Six-Card AND Protocol

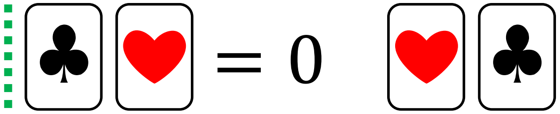




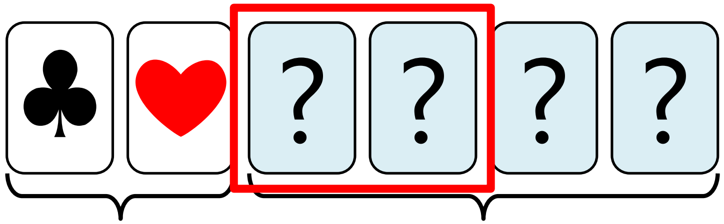
$a \oplus r$ $\underline{\mathrm{shift}^r(0, b)}$

$$a \wedge b = \mathrm{get}^{a \oplus r}(\underline{\mathrm{shift}^r(0, b)})$$

The Six-Card AND Protocol



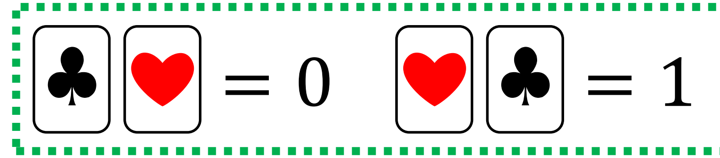
$= 0$ $= 1$

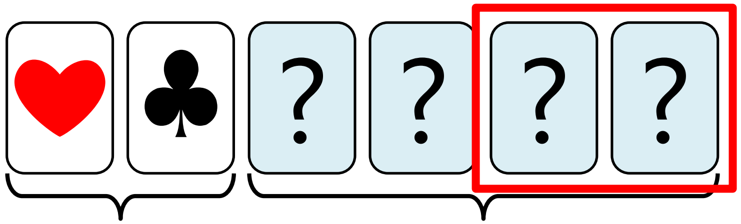


$a \oplus r$ $\mathrm{shift}^r(0, b)$

$$a \wedge b = \begin{cases} \mathrm{get}^0\left(\mathrm{shift}^r(0, b)\right) & \text{if } a \oplus r = 0 \\ \mathrm{get}^1\left(\mathrm{shift}^r(0, b)\right) & \text{if } a \oplus r = 1 \end{cases}$$

The Six-Card AND Protocol

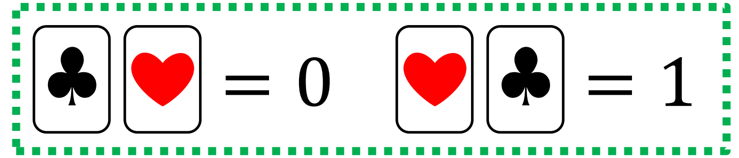


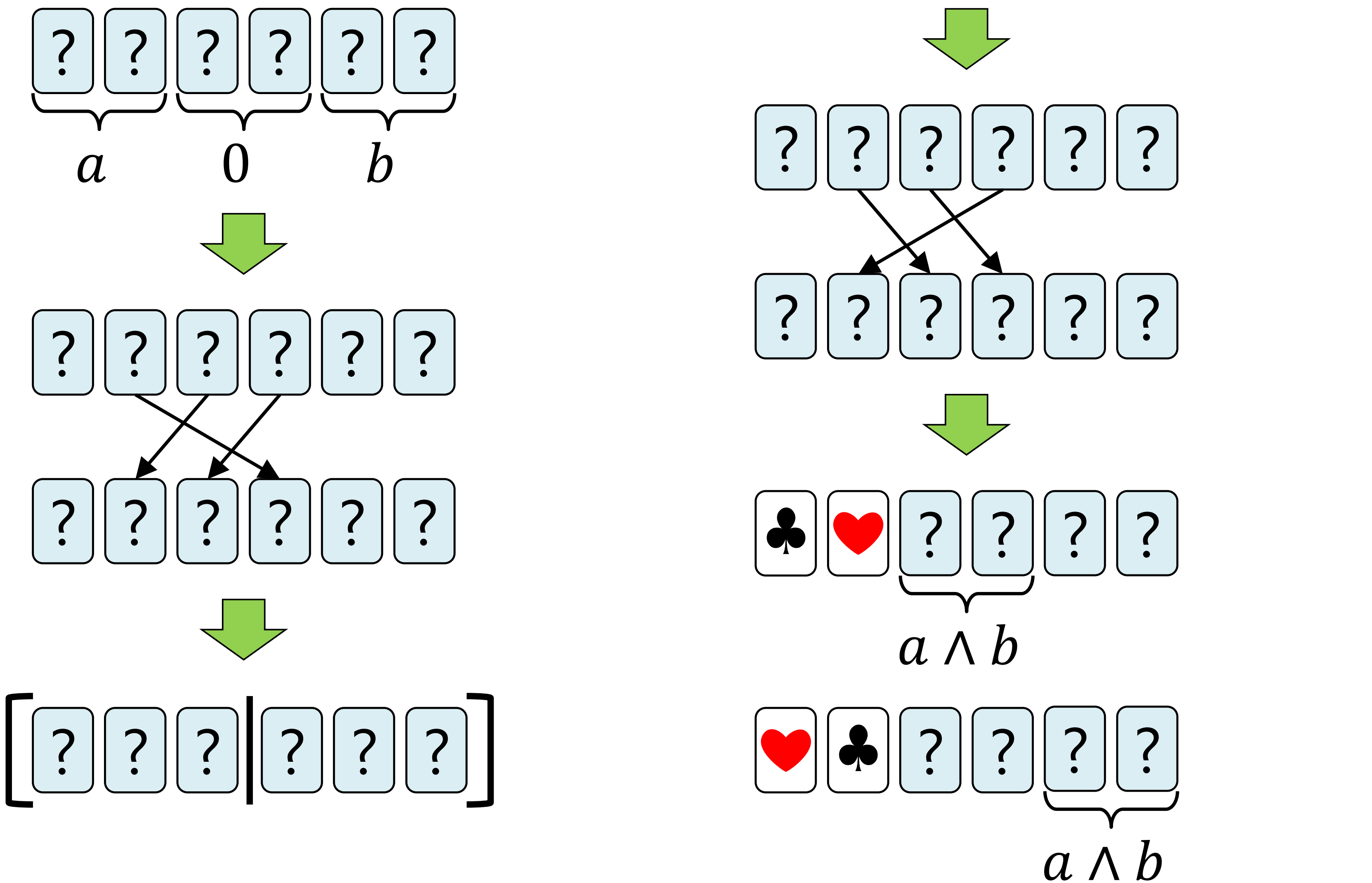


$a \oplus r$ $\text{shift}^r(0, b)$

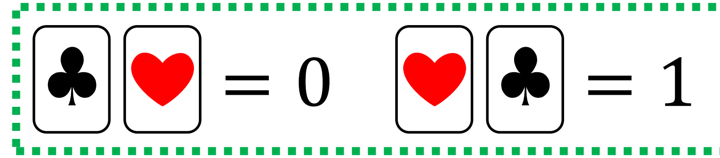
$$a \land b = \begin{cases} \text{get}^0\left(\text{shift}^r(0, b)\right) & \text{if } a \oplus r = 0 \\ \text{get}^1\left(\text{shift}^r(0, b)\right) & \text{if } a \oplus r = 1 \end{cases}$$

The Six-Card AND Protocol



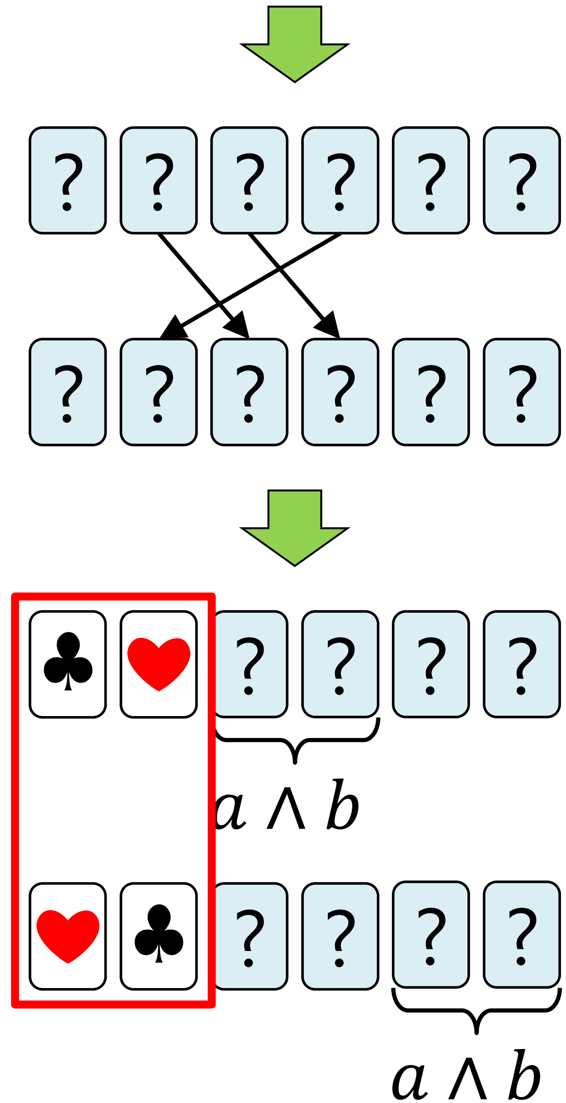


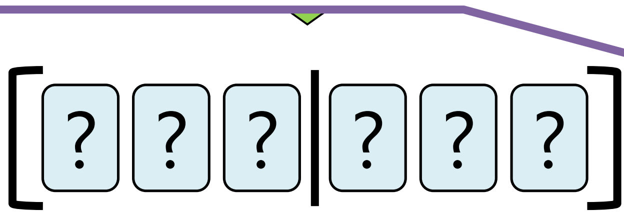
The Six-Card AND Protocol



Revealing $a \oplus r$ does **not leak any information** about a because r is **random**.

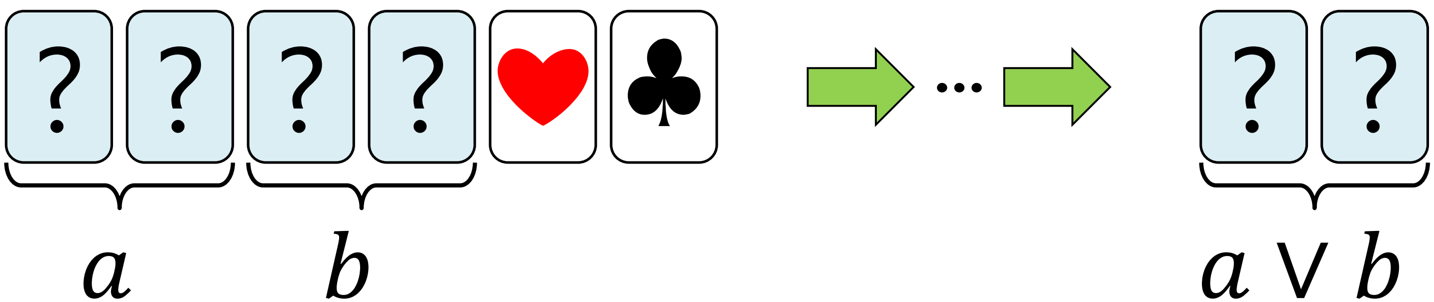
The 2 face-up cards are available for another computation.





The Six-Card AND Protocol

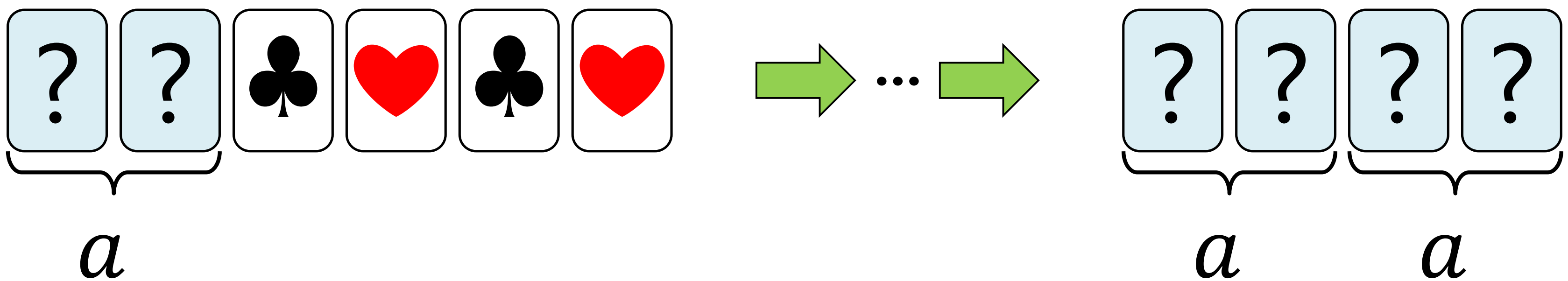
- A 6-card **OR** protocol can be easily constructed in a similar manner [6].



[6] Mizuki, T., Asiedu, I. K., Sone, H.: Voting with a logarithmic number of cards. In: UCNC 2013, LNCS 7956, pp. 162–173. (2013)

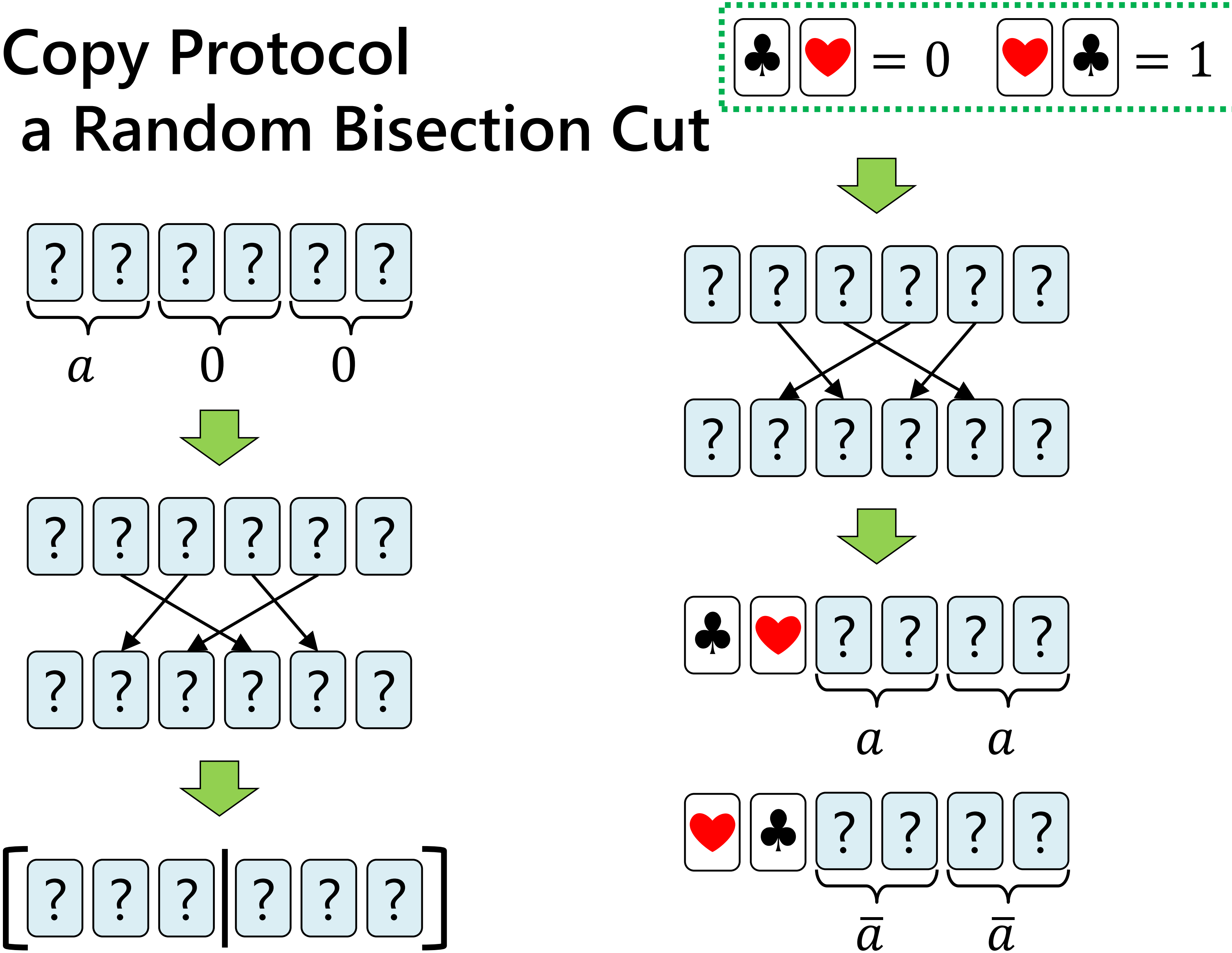
The Copy Protocol with a Random Bisection Cut

- Given a commitment to a bit a, 4 additional cards are sufficient to make 2 copies of the commitment [8].

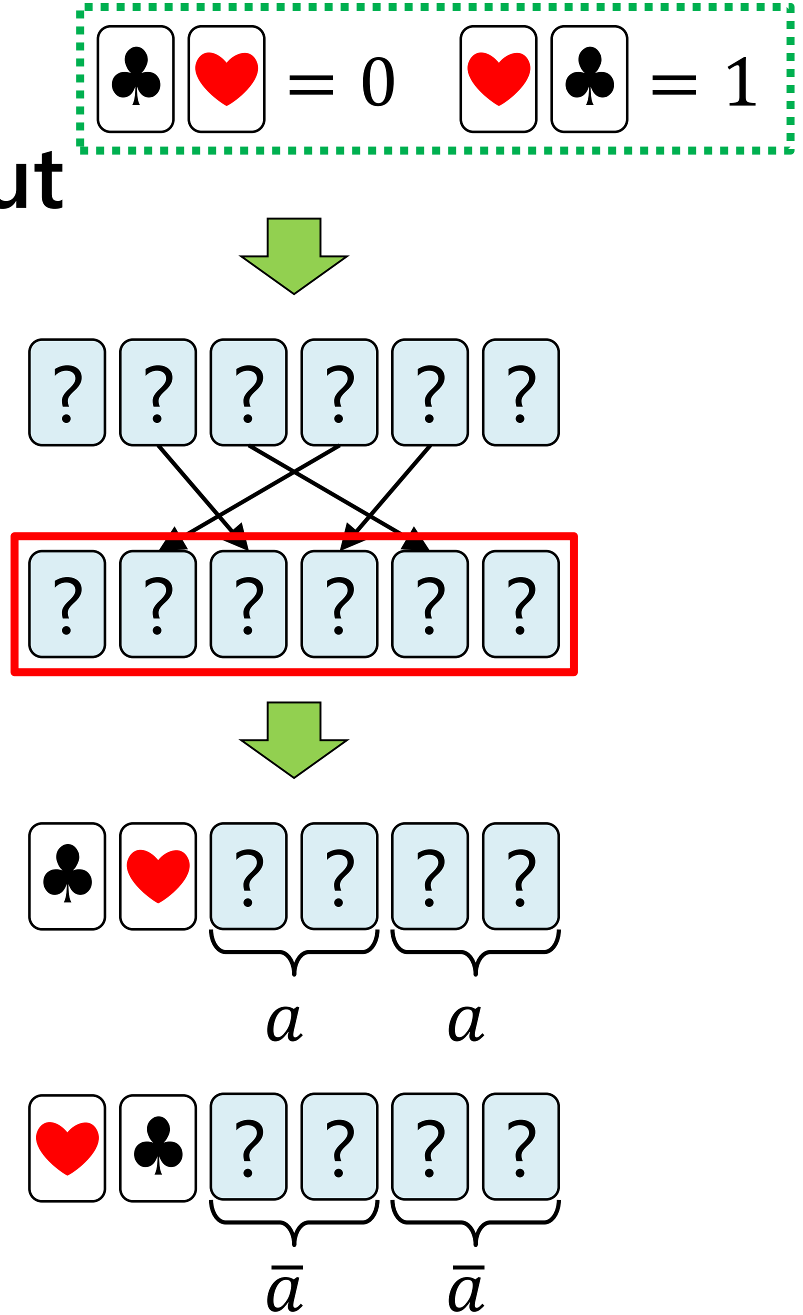


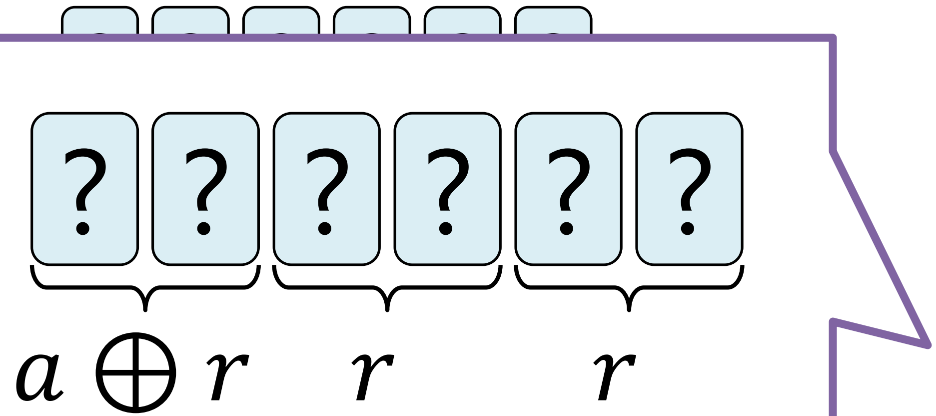
[8] Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: FAW 2009, LNCS 5598, pp. 358–369. (2009)

The Copy Protocol with a Random Bisection Cut

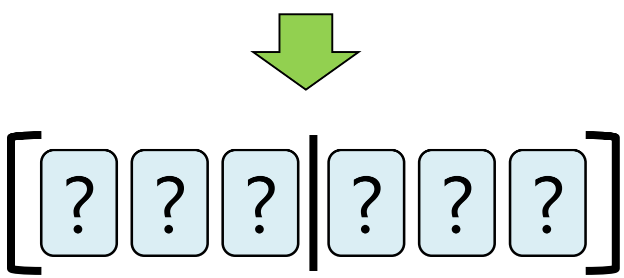


The Copy Protocol with a Random Bisection Cut

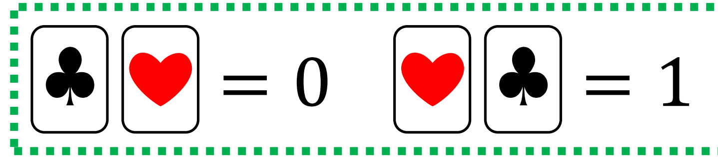


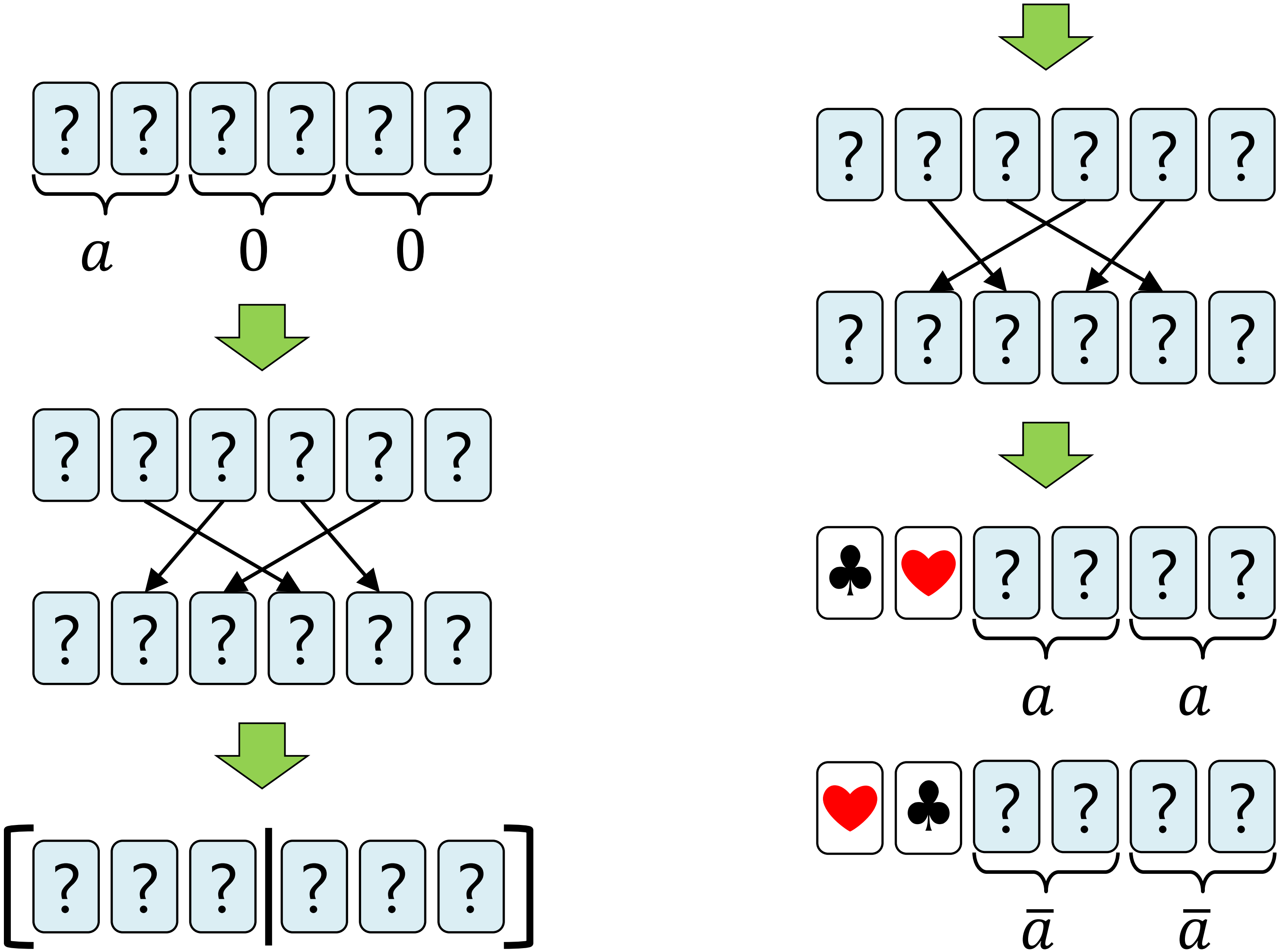


r is a random bit because of the random bisection cut.



The Copy Protocol with a Random Bisection Cut

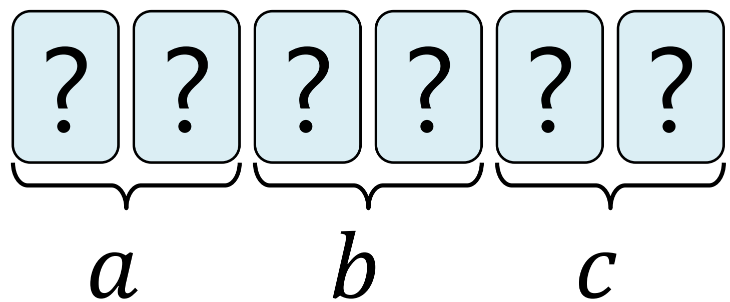




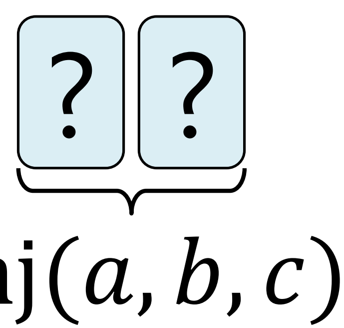
Contents

Straightforward Secure Majority Computations

- By applying the existing protocols, $\mathrm{maj}(a, b, c)$ can be **naively** computed with 14 cards. That is, given



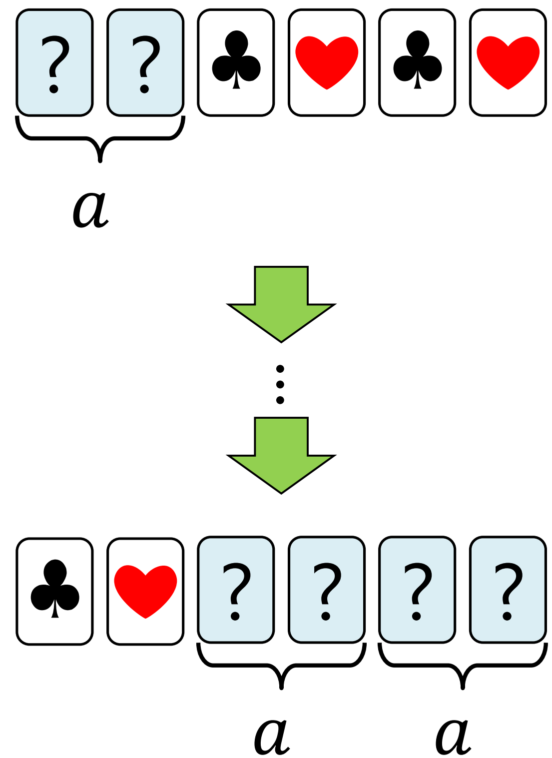
together with 8 additional cards, we can obtain



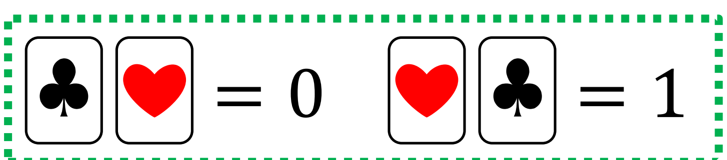
Straightforward Secure Majority Computations

Recall

- Applying the copy protocol to a commitment along with 4 additional cards

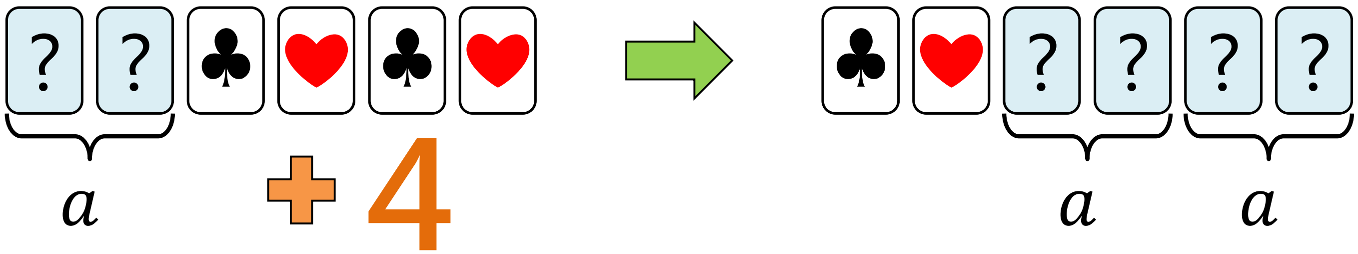


results in 2 copied commitments as well as 2 available cards.



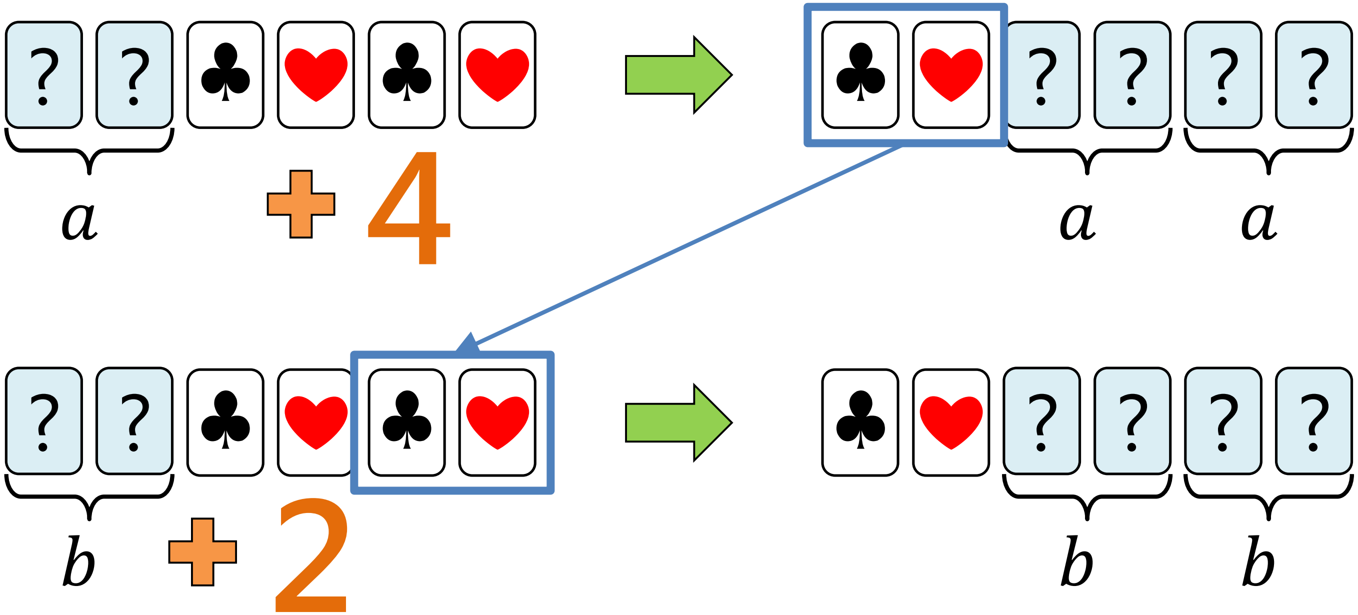
Straightforward Secure Majority Computations

- Applying the copy protocol 3 times.



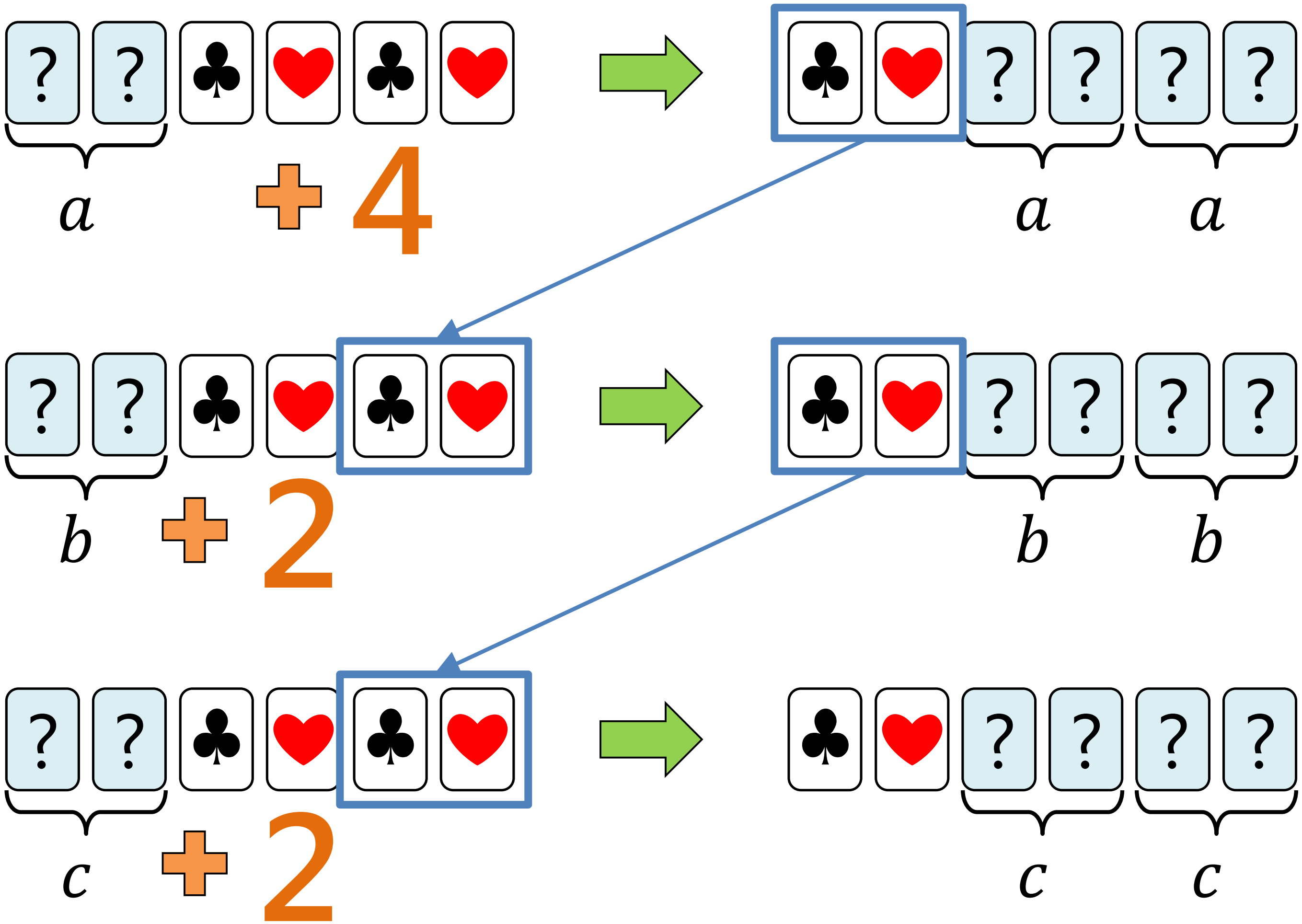
Straightforward Secure Majority Computations

- Applying the copy protocol 3 times.



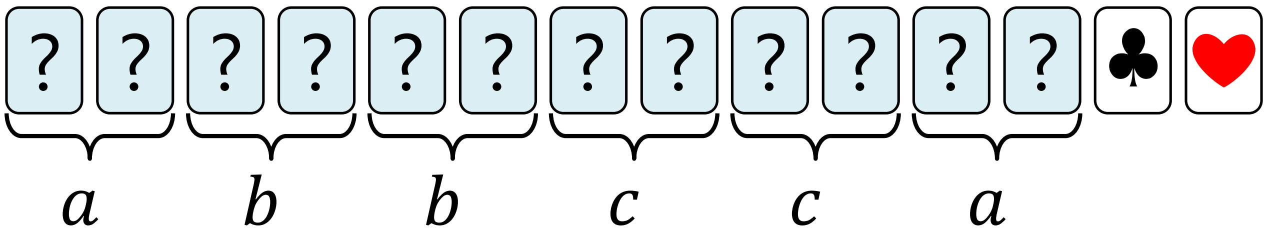
Straightforward Secure Majority Computations

- Applying the copy protocol 3 times.



Straightforward Secure Majority Computations

- We have



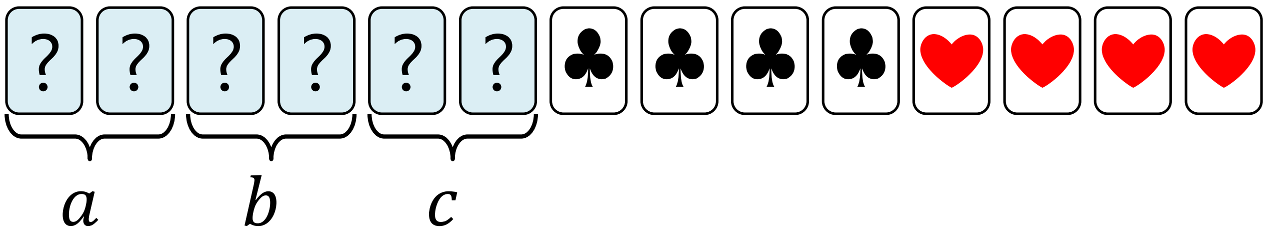
- Since

$$\mathrm{maj}(a,b,c) = (a \wedge b) \vee (b \wedge c) \vee (c \wedge a)$$

we can easily obtain a commitment to $\mathrm{maj}(a,b,c)$ by applying the AND/OR protocol mentioned before.

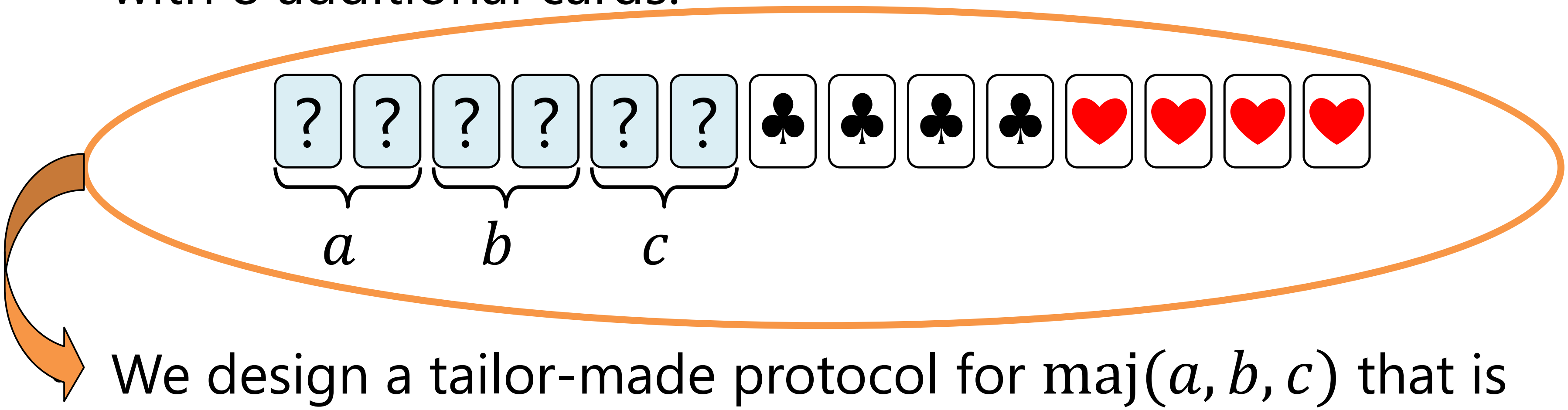
Straightforward Secure Majority Computations

- Thus, $\mathrm{maj}(a, b, c)$ can be straightforwardly conducted with 8 additional cards.



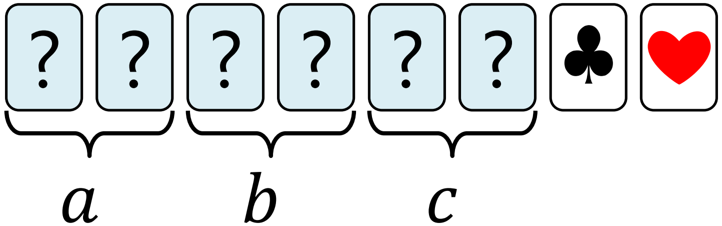
Straightforward Secure Majority Computations

- Thus, $\mathrm{maj}(a, b, c)$ can be straightforwardly conducted with 8 additional cards.



We design a tailor-made protocol for $\mathrm{maj}(a, b, c)$ that is simple and needs only 2 additional cards.

Next Section!



Contents

Contents

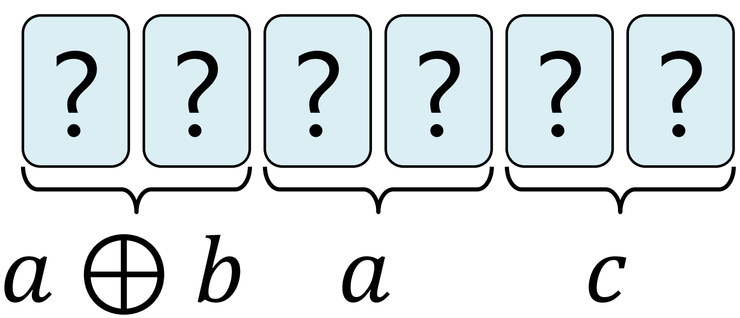
1. Introduction
2. Known Protocols
3.
   - The Idea
   - An Eight-Card Secure Majority Protocol
4. **An Improved Secure Majority Protocol**
5. Conclusion

The Idea

$$\mathrm{maj}(a,b,c) = \begin{cases} a & \text{if } a = b \\ c & \text{if } a \neq b \end{cases}$$

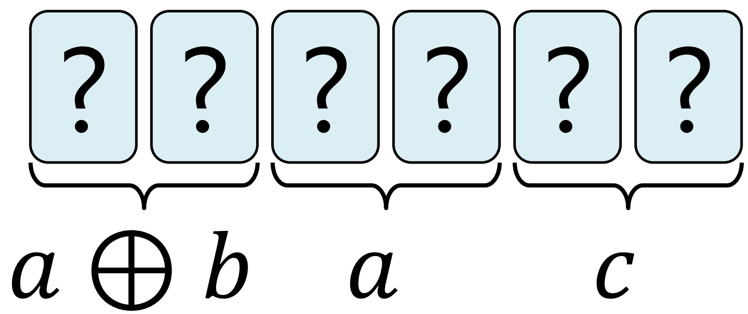
$$\therefore \mathrm{maj}(a,b,c) = \mathrm{get}^{a \oplus b}(a,c)$$

- Hence, our protocol first makes the following sequence:



The Idea

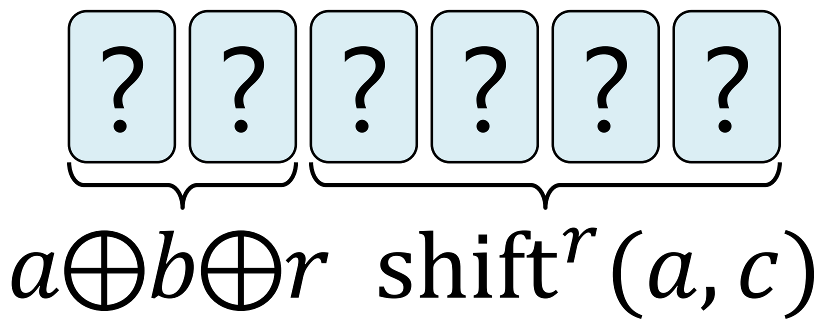
- If we turned over the leftmost 2 cards, then we could determine the position of the desired commitment to $\mathrm{get}^{a\oplus b}(a,c)$, but the value of $a \oplus b$ would also be leaked.



$$\mathrm{maj}(a,b,c) = \mathrm{get}^{a\oplus b}(a,c)$$

The Idea

- Therefore, our protocol next adds randomization to hide the value of $a \oplus b$: it produces a sequence



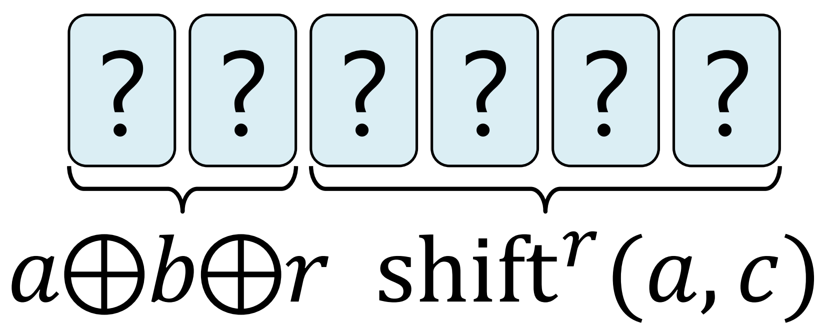
where r is a random bit.

The Idea

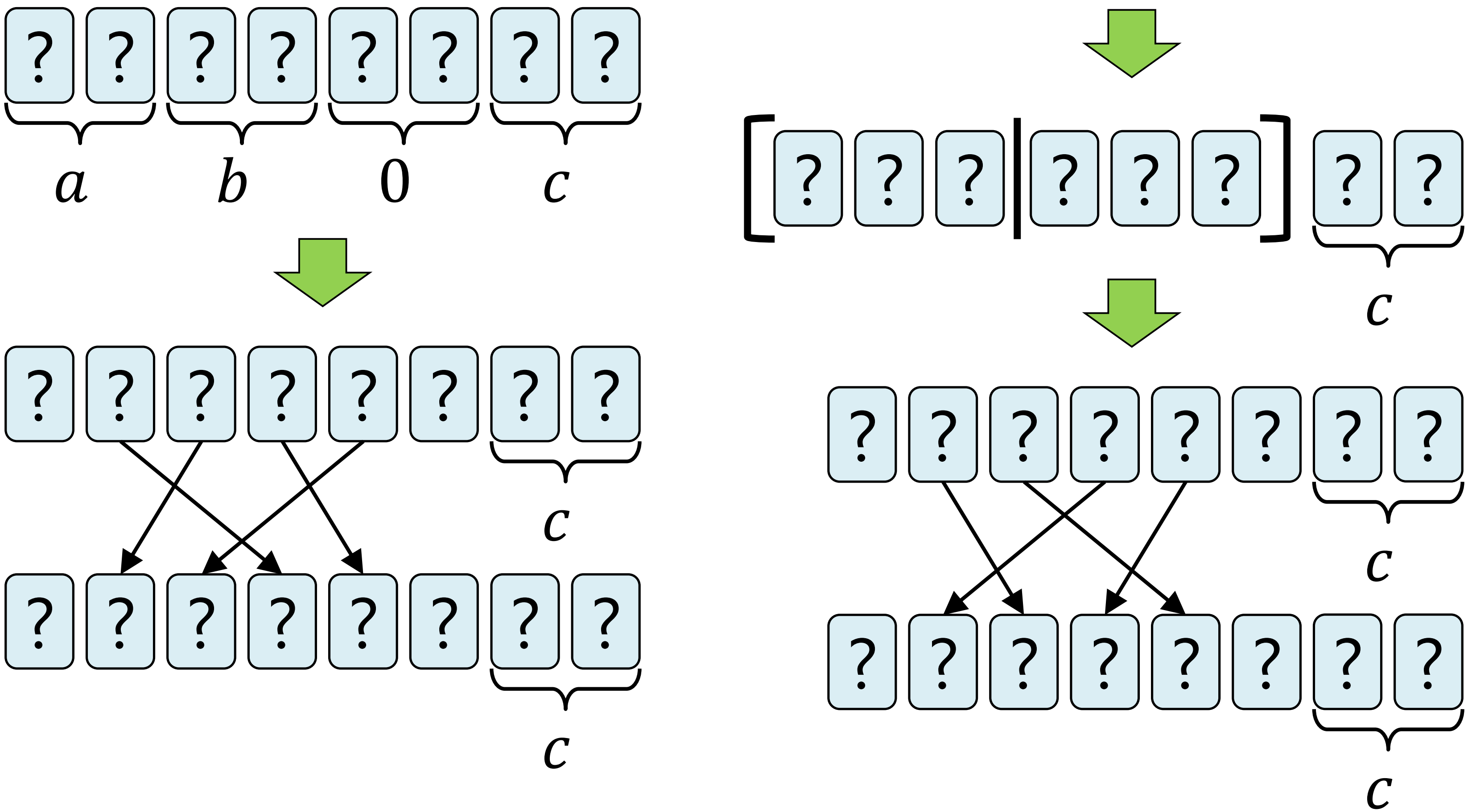
- Based on the equality

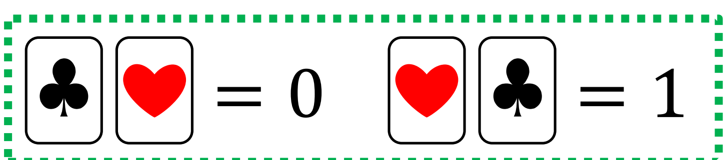
$$\mathrm{maj}(a,b,c) = \mathrm{get}^{a\oplus b\oplus r}(\mathrm{shift}^{r}(a,c))$$

we can obtain a commitment to $\mathrm{maj}(a,b,c)$ by revealing the leftmost 2 cards.

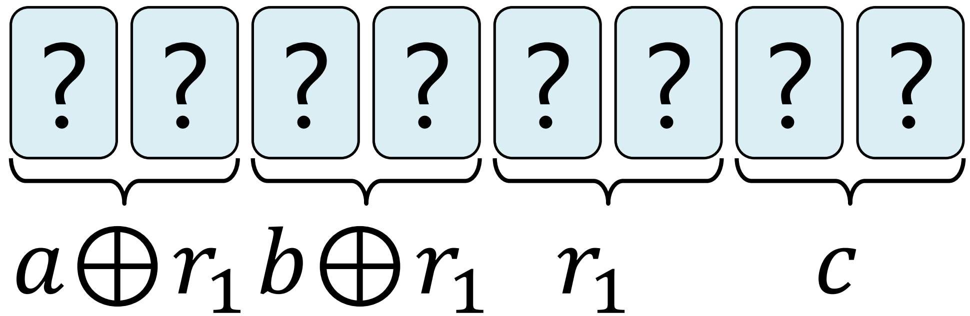


An Eight-Card Secure Majority Protocol

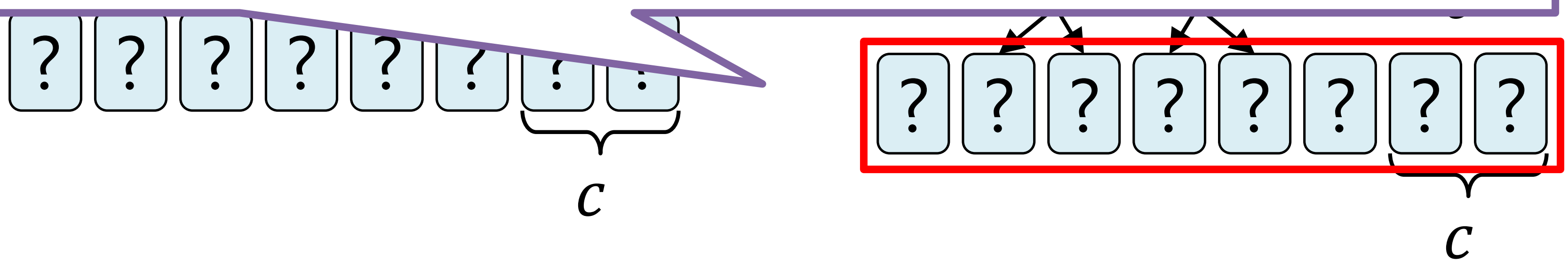


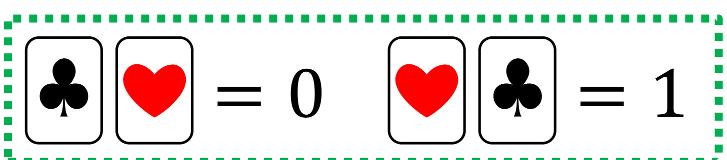


An Eight-Card Secure Majority Protocol



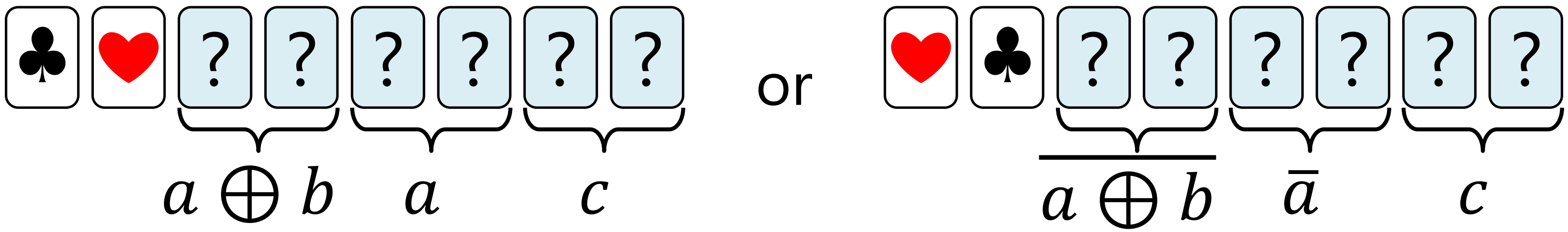
r_1 is a random bit because of the random bisection cut.



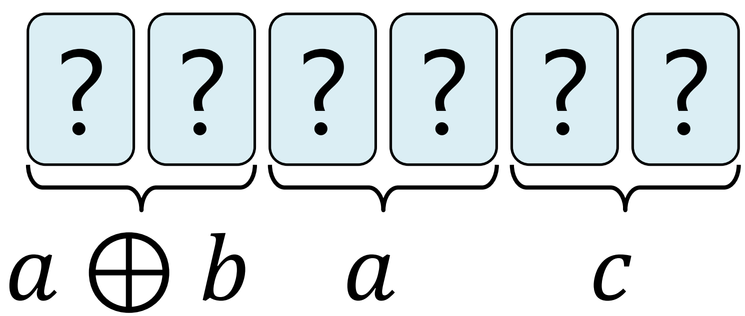


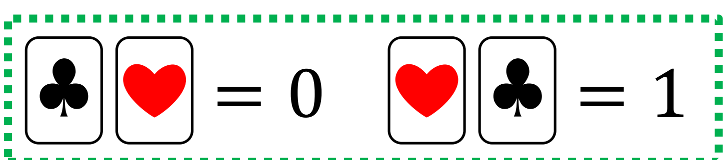
An Eight-Card Secure Majority Protocol

- Reveal the leftmost two cards. Then, we have either

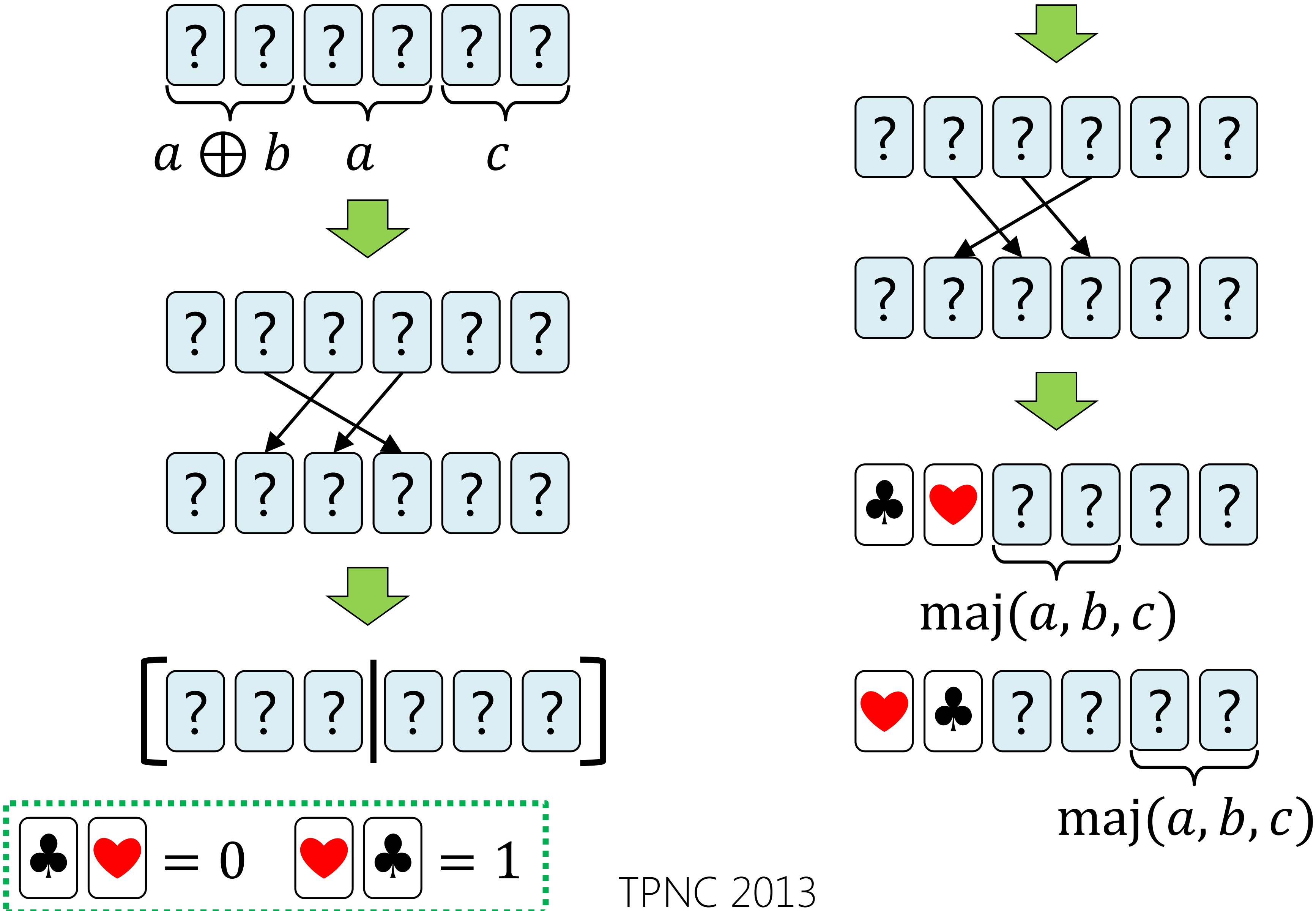


- In the latter case, apply the secure NOT operation to $\overline{a \oplus b}$ and $\bar{a}$. Hence, in either case, we have 3 commitments

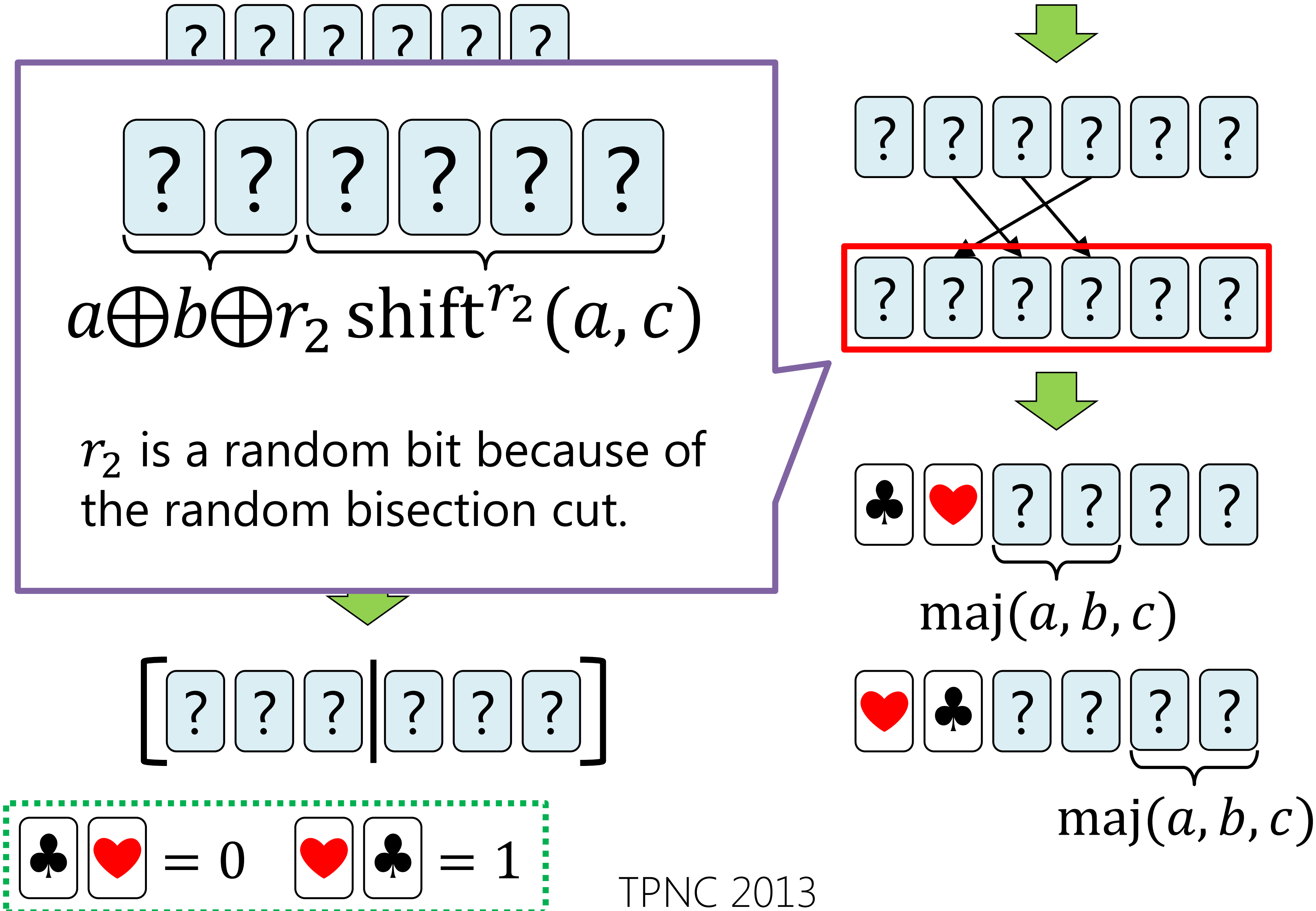




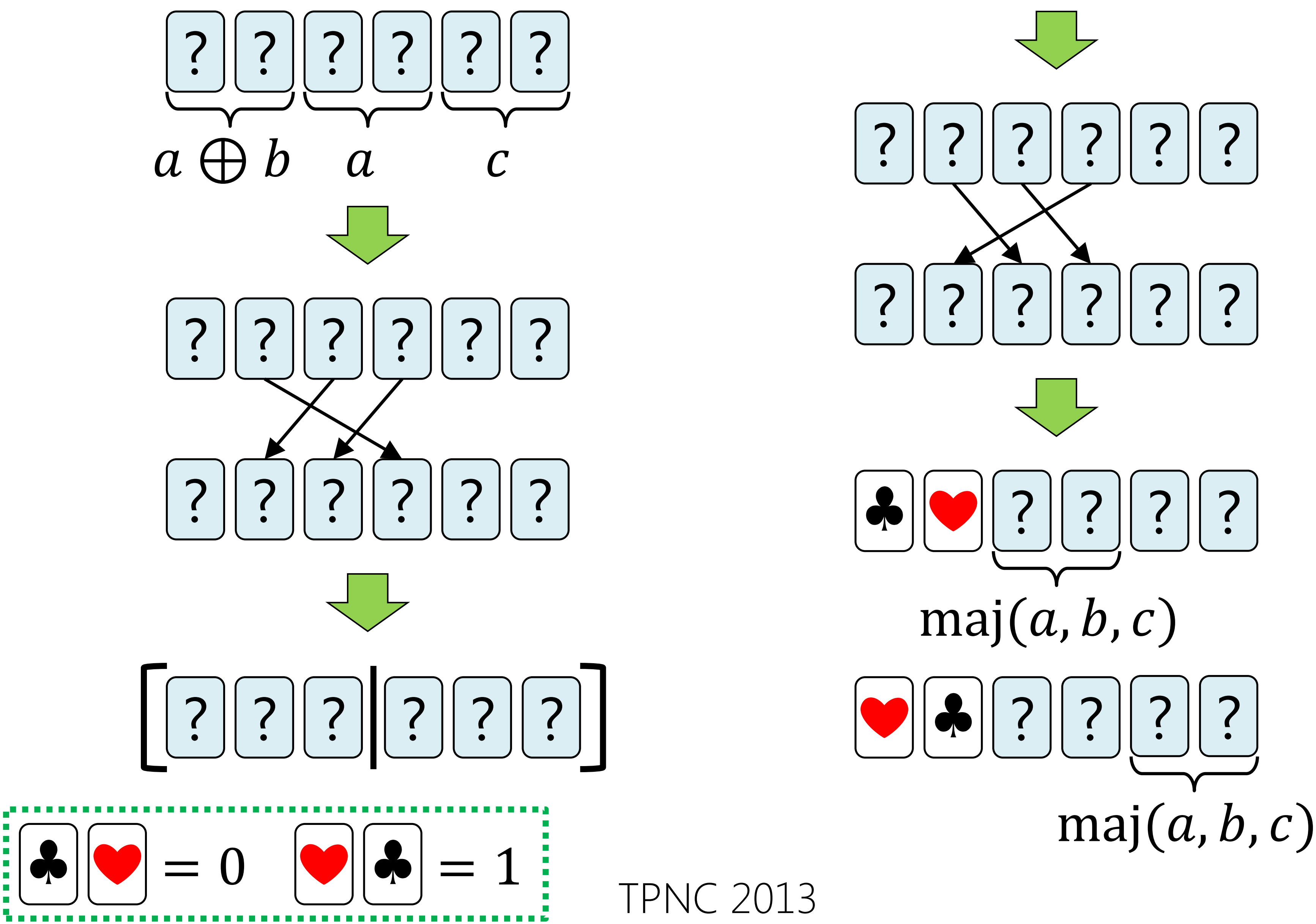
An Eight-Card Secure Majority Protocol



An Eight-Card Secure Majority Protocol



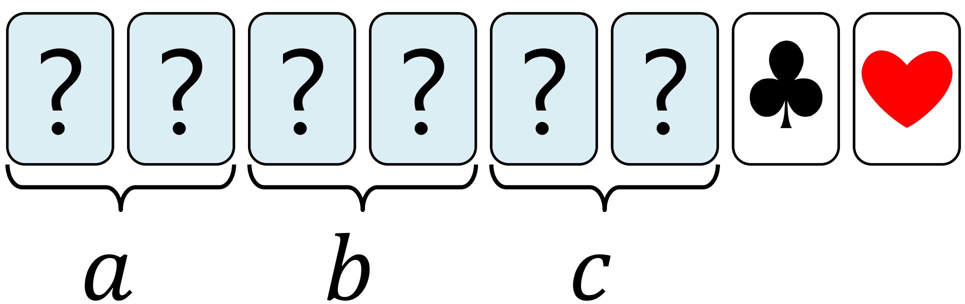
An Eight-Card Secure Majority Protocol



Contents

Conclusion

- We designed an 8-card 3-input secure majority protocol.
- Since the naive implementation of $\mathrm{maj}(a, b, c)$ requires 14 cards, we have reduced the # of required cards **by 6**.
- We can also easily prove that **any 3-variable symmetric function** can be securely computed with **8 cards** or less.



Thank you!