Voting with a Logarithmic Number of Cards

Takaaki Mizuki, Isaac Kobina Asiedu, Hideaki Sone
Tohoku University
Abstract
Abstract

- There are 2 candidates and $n$ voters.
There are 2 candidates and $n$ voters. Usually, $n$ ballot papers are required.
Abstract

- There are 2 candidates and \( n \) voters.
- Usually, \( n \) ballot papers are required.
- We show \( O(\log n) \) cards conduct an election.
1. Introduction
2. Known Protocols
3. Voting with a Logarithmic Number of Cards
4. New Adder Protocols
5. Conclusion
1. Introduction

2. Known Protocols
   1.1 Computation Using a Deck of Cards
   1.2 History of Card-Based Protocols
   1.3 Our Results

3. Voting with a Logarithmic Number of Cards

4. New Adder Protocols

5. Conclusion
In this paper, we use a deck of cards.
face-up

turn over

face-down
How to implement voting?
The simplest way is as follows.
1. Distribute two cards of different suits to each voter.

\[ n \text{ voters} \]
1. Distribute two cards of different suits to each voter.

$n$ voters
1. Distribute two cards of different suits to each voter.
2. Each voter privately commits his/her ballot according to the encoding.
1. Distribute two cards of different suits to each voter.
2. Each voter privately commits his/her ballot.
3. Shuffle all left cards and reveal them.
1. Distribute two cards of different suits to each voter.
2. Each voter privately commits his/her.
3. Shuffle all left cards and reveal them.
1. Distribute two cards of different suits to each voter.
2. Each voter privately commits his/her.
3. Shuffle all left cards and reveal them.
1. Distribute two cards of different suits to each voter.
2. Each voter privately commits his/her.
3. Shuffle all left cards and reveal them.

3 votes
2 votes
Voting can be naively done using $2n$ cards.

$n$ voters

2n cards

✓ Voting can be naively done using $2n$ cards.
Voting can be naively done using $2n$ cards.

This paper shows that, by applying card-based cryptographic protocols, $O(\log n)$ cards can also conduct voting.
Notations and the history

Card-based protocols provide secure computation.
Notations and the history

Card-based protocols provide secure computation.

To deal with Boolean values, this encoding is used:

\[
\begin{align*}
\text{♣️❤️} &= 0 \\
\text{❤️♣️} &= 1
\end{align*}
\]
To deal with Boolean values, this encoding is used:

\[
\begin{align*}
\text{♠️❤️} &= 0 \\
\text{❤️♠️} &= 1
\end{align*}
\]

A **commitment** to a bit \( x \in \{0,1\} \) is a pair of two face-down cards holding the value of \( x \).
To deal with Boolean values, this encoding is used:

\[
\begin{align*}
\text{♠️❤️} & = 0 \\
❤️♣️ & = 1
\end{align*}
\]

A commitment to a bit \( x \in \{0,1\} \) is a pair of two face-down cards holding the value of \( x \).
Notations and the history

Example of secure computation

\[\begin{align*}
\spadesuit \heartsuit &= 0 \\
\heartsuit \clubsuit &= 1
\end{align*}\]
Notations and the history

Example

reverse the order

\[ \spadesuit \heartsuit = 0 \]

\[ \heartsuit \clubsuit = 1 \]
Notations and the history

Example

reverse the order

\[ \begin{array}{c}
\clubsuit \heartsuit = 0 \\
\heartsuit \clubsuit = 1
\end{array} \]
With keeping the value of $\chi$ secret, we can get a commitment to the negation $\overline{\chi}$ of $\chi$. 

\[ \clubsuit \heartsuit = 0 \]
\[ \heartsuit \clubsuit = 1 \]
Notations and the history

Example

reverse the order

With keeping the value of $\chi$ secret, we can get a commitment to the negation $\overline{\chi}$ of $\chi$.

- Secure **NOT** operation is trivial.
With keeping the value of $x$ secret, we can get a commitment to the negation $\overline{x}$ of $x$.

- Secure NOT operation is trivial.
- How about secure AND operation?
Notations and the history

How about secure **AND** operation?

With keeping the values of $a$ and $b$ secret, we want to get a commitment to $a \land b$. 

<table>
<thead>
<tr>
<th>Card</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>♣️</td>
<td>0</td>
</tr>
<tr>
<td>♥️</td>
<td>1</td>
</tr>
</tbody>
</table>
How about secure AND operation?

With keeping the values of \(a\) and \(b\) secret, we want to get a commitment to \(a \land b\).

There have been such four protocols in the literatures.
## History of Secure AND protocols

<table>
<thead>
<tr>
<th>AND</th>
<th>required cards</th>
<th>avg. # of trials</th>
</tr>
</thead>
</table>
| Crepeau-Kilian [CRYPTO ’93] | 10  
  
  ♠️ ♠️ ♠️ ♠️ ♥️ ♥️ ♥️ ♥️ ♥️ ♥️ ♥️ ♦️ ♦️ | 6 |
| Niemi-Renvall [TCS, 1998] | 12  
  
  ♠️ ♠️ ♠️ ♠️ ♠️ ♠️ ♥️ ♥️ ♥️ ♥️ ♥️ ♥️ ♥️ ♥️ | 2.5 |
| Stiglic [TCS, 2001] | 8   
  
  ♠️ ♠️ ♠️ ♠️ ✔️ ✔️ ✔️ ✔️ ✔️ ✔️ ✔️ ✔️ ✔️ | 2 |
| Mizuki-Sone [FAW 2009] | 6   
  
  ♠️ ♠️ ♠️ ♥️ ♥️ ♥️ ♥️ | 1 |
History of Secure AND protocols

<table>
<thead>
<tr>
<th>AND</th>
<th>required cards</th>
<th>avg. # of trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crepeau-Kilian</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>[CRYPTO ’93]</td>
<td>⚖️♠️♣️♣️♠️❤️❤️❤️❤️❤️♣️♣️♠️</td>
<td></td>
</tr>
<tr>
<td>Niemi-Renvall</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>[TCS, 1998]</td>
<td>⚖️♠️♣️♣️♠️❤️❤️❤️❤️❤️♣️♣️♠️</td>
<td></td>
</tr>
<tr>
<td>Stiglic</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>[TCS, 2001]</td>
<td>⚖️♠️♣️♣️♠️❤️❤️❤️❤️❤️♣️♣️♠️</td>
<td></td>
</tr>
<tr>
<td>Mizuki-Sone</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>[FAW 2009]</td>
<td>⚖️♠️♣️♣️♠️❤️❤️❤️❤️❤️♣️♣️♠️</td>
<td></td>
</tr>
</tbody>
</table>

Will be introduced in Section 2.2
### History of Secure XOR protocols

![Diagram showing cards and an XOR operation]

<table>
<thead>
<tr>
<th>XOR</th>
<th># of required cards</th>
<th># of types</th>
<th>avg. # of trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crepeau-Kilian [CRYPTO '93]</td>
<td>14</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Mizuki, et. al [AJoC, 2006]</td>
<td>10</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Mizuki-Sone [FAW 2009]</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
History of Secure XOR protocols

<table>
<thead>
<tr>
<th>XOR</th>
<th># of required cards</th>
<th>avg. # of trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crepeau-Kilian</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>[CRYPTO ’93]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mizuki, et. al</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>[AJoC, 2006]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mizuki-Sone</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>[FAW 2009]</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Will be introduced in Section 2.3
Existing COPY protocols

Make identical copies of a commitment.
Existing COPY protocols

Make identical copies of a commitment.

I’ll introduce the best existing COPY protocol in Section 2.4
Outline of our results
Outline of our results

Half adder

\[ s = a \oplus b \]

\[ c = a \land b \]

Voting
Outline of our results

Using existing AND/XOR/COPY protocols

<table>
<thead>
<tr>
<th>Half adder</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting</td>
<td></td>
</tr>
</tbody>
</table>

[# of cards]
### Outline of our results

Using existing AND/XOR/COPY protocols

<table>
<thead>
<tr>
<th>Half adder</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting</td>
<td>$2^{\lceil \log n \rceil} + 8$</td>
</tr>
</tbody>
</table>

Applying a half adder

[# of cards]
Outline of our results

<table>
<thead>
<tr>
<th></th>
<th>Using existing AND/XOR/COPY protocols</th>
<th>Devising a tailor-made half adder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Half adder</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>Voting</td>
<td>$2\lceil \log n \rceil + 8$</td>
<td></td>
</tr>
</tbody>
</table>

[Number of cards]
# Outline of our results

Using existing AND/XOR/COPY protocols

Devising a tailor-made half adder

<table>
<thead>
<tr>
<th>Half adder</th>
<th>10</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting</td>
<td>$2\lceil\log n\rceil + 8$</td>
<td>$2\lceil\log n\rceil + 6$</td>
</tr>
</tbody>
</table>

[\# of cards]
Outline of our results

Using existing AND/XOR/COPY protocols

Devising a tailor-made half adder

<table>
<thead>
<tr>
<th>Half adder</th>
<th>10</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting</td>
<td>$2 \lceil \log n \rceil + 8$</td>
<td>$2 \lceil \log n \rceil + 6$</td>
</tr>
</tbody>
</table>

[&# of cards]
2.1 Random Bisection Cuts

a random bisection cut

Bisect a given deck of cards, and then randomly switch the resulting two portions:

\[
\begin{array}{c|c}
? & ? \\
? & ? \\
? & ? \\
\end{array}
\]

prob. 1/2
not switched

\[
\begin{array}{c|c}
? & ? \\
? & ? \\
? & ? \\
\end{array}
\]

prob. 1/2
switched
a random bisection cut

Bisect a given deck of cards, and then randomly switch the resulting two portions:

```
```

- prob. 1/2
- not switched

- prob. 1/2
- switched

easy-to-implement card shuffling operation
Secure AND can be done with 6 cards [6].

2.2 Six-Card AND Protocol

\[ a \land b \]

Arrange 2 commitments and 2 additional cards:
Turn over the rightmost two cards:

They become a commitment to 0.
Rearrange the positions:

\[ a \land b \]

\[ \spadesuit \heartsuit = 0 \]
\[ \heartsuit \spadesuit = 1 \]
Apply a random bisection cut:

\[
\begin{bmatrix}
\end{bmatrix}
\]

- prob. of 1/2
- prob. of 1/2

\[
\begin{bmatrix}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\end{bmatrix}
\]

\( a \land b \)
Rearrange the positions:

(a)  
(b)  

\( a \land b \)
(a)  
\[ a \quad b \quad 0 \]

(b)  
\[ \bar{a} \quad 0 \quad b \]
where $r \in \{0,1\}$ is a random bit.
where \( r \in \{0, 1\} \) is a random bit.
reveal

\[ a \oplus r \quad \overline{r} \land b \quad r \land b \]

\[ \begin{align*}
& a \lor r \\
& \overline{r} \land b \\
& r \land b \\
& [? ? ? ??]
\end{align*} \]

\[ \begin{align*}
& \clubsuit \heartsuit \\
& \heartsuit \clubsuit \\
& \begin{aligned}
& a \land b \\
& a \land b
\end{aligned}
\]
$a \oplus r = 0$, i.e., $a = r$

\[a \land b\]
reveal

\[ a \oplus r = 0, \text{ i.e.}, a = r \]

\[ a \oplus r = 0 \]

\[ r \land b = a \land b \]

\[ a \land b = 0 \]

\[ a \land b = 1 \]
No information about \( a \) leaks because \( r \) is random.
reveal

\[ a \oplus r = \overline{r} \land b \land r \land b \]

\[ a \oplus r = 1, \text{ i.e., } a = \overline{r} \]
reveal

\[ a \oplus r \quad \bar{r} \land b \]

\[ = a \land b \]

\[ a \oplus r = 1, \text{ i.e., } a = \bar{r} \]
Works!
2.3 Four-Card XOR Protocol

Secure XOR can be done with 4 cards [6].

2.4 Copy Protocol with a Random Bisection Cut

Making a copy can be done with 4 additional cards [6].


\[
\begin{align*}
\text{\(a\)} & \downarrow \\
\text{\(0\)} & \downarrow \\
\text{\(a\)} & \downarrow \\
\end{align*}
\]

\[
\begin{align*}
\text{\(= 0\)} & \\
\text{\(= 1\)} & \\
\end{align*}
\]
Outline of our results

Using existing AND/XOR/COPY protocols

Devising a tailor-made half adder

<table>
<thead>
<tr>
<th>Half adder</th>
<th>10</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting</td>
<td>$2\lceil\log n\rceil + 8$</td>
<td>$2\lceil\log n\rceil + 6$</td>
</tr>
</tbody>
</table>
Half adder

\[ s = a \oplus b \]
\[ c = a \land b \]
Half adder

\[ s = a \oplus b \]
\[ c = a \land b \]
Remember that

✓ AND can be done with 6 cards;
✓ XOR can be done with 4 cards;
✓ COPY can be done with 4 additional cards.
- AND can be done with 6 cards;
- XOR can be done with 4 cards;
- COPY can be done with 4 additional cards.

\[\text{\begin{array}{c}
\text{\lor} = \text{AND}
\end{array}}\]

\[\text{\begin{array}{c}
\text{\xor} = \text{XOR}
\end{array}}\]

\[\text{\begin{array}{c}
\text{\copy} = \text{COPY}
\end{array}}\]
✓ AND can be done with 6 cards;
✓ XOR can be done with 4 cards;
✓ COPY can be done with 4 additional cards.
✓ AND can be done with 6 cards;
✓ XOR can be done with 4 cards;
✓ COPY can be done with 4 additional cards.
✓ AND can be done with 6 cards;
✓ XOR can be done with 4 cards;
✓ COPY can be done with 4 additional cards.
✓ AND can be done with 6 cards;
✓ XOR can be done with 4 cards;
✓ COPY can be done with 4 additional cards.
✓ AND can be done with 6 cards;
✓ XOR can be done with 4 cards;
✓ COPY can be done with 4 additional cards.
# Outline of our results

Using existing AND/XOR/COPY protocols

Devising a tailor-made half adder

<table>
<thead>
<tr>
<th>Half adder</th>
<th>Voting</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Half adder circuit" /></td>
<td><img src="image" alt="Voting diagram" /></td>
</tr>
<tr>
<td>10</td>
<td>2[\log n] + 8</td>
</tr>
<tr>
<td>8</td>
<td>2[\log n] + 6</td>
</tr>
</tbody>
</table>

[\# of cards]
## Outline of our results

- Using existing AND/XOR/COPY protocols
- Devising a tailor-made half adder

<table>
<thead>
<tr>
<th></th>
<th>10</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Half adder</strong></td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td><img src="image1.png" alt="Half adder" /></td>
<td><img src="image2.png" alt="Image2" /></td>
<td></td>
</tr>
<tr>
<td><strong>Voting</strong></td>
<td>$2\left\lceil \log n \right\rceil + 8$</td>
<td>$2\left\lceil \log n \right\rceil + 6$</td>
</tr>
<tr>
<td><img src="image3.png" alt="Voting" /></td>
<td><img src="image4.png" alt="Image4" /></td>
<td></td>
</tr>
</tbody>
</table>

[# of cards]
$n$ voters

$x_1 \in \{0,1\}$

$x_2$

$x_3$

$x_4$

$\ldots$

$2\lceil \log n \rceil + 8$ cards

encoding for candidates

$= 0$

$= 1$
$x_1 \in \{0, 1\}$
$x_2 \in \{0, 1\}$
Apply a half adder
binary representation of $x_1 + x_2$
binary representation of $x_1 + x_2$
binary representation of $x_1 + x_2$

Apply a half adder (and XOR)
binary representation
of $x_1 + x_2 + x_3$
For example, if $x_1 = x_2 = x_3 = 1$, then

binary representation of $x_1 + x_2 + x_3$
For example, if \( x_1 = 1 \) and \( x_2 = x_3 = 0 \), then

The binary representation of \( x_1 + x_2 + x_3 \) is

\[
\begin{array}{ccccc}
\clubsuit & \heartsuit & \text{？} & \text{？} & \text{？} \\
\text{？} & \text{？} & \text{？} & \text{？} & \text{？} \\
\text{？} & \text{？} & \text{？} & \text{？} & \text{？} \\
\end{array}
\]

0 1

binary representation of \( x_1 + x_2 + x_3 \)
binary representation of $x_1 + \cdots + x_{n-1}$

$2^\lceil \log n \rceil$ cards
$x_n$ is the binary representation of $x_1 + \cdots + x_{n-1}$ with $2^{\lceil \log n \rceil}$ cards.
\[ 2 \lceil \log n \rceil + 2 \] (or \( 2 \lceil \log n \rceil \)) cards

binary representation of \( x_1 + \cdots + x_n \)
Outline of our results

Using existing AND/XOR/COPY protocols

Devising a tailor-made half adder

<table>
<thead>
<tr>
<th>Half adder</th>
<th>10</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting</td>
<td>$2\lceil \log n \rceil + 8$</td>
<td>$2\lceil \log n \rceil + 6$</td>
</tr>
</tbody>
</table>

[# of cards]
1. Introduction
2. Known Protocols
3. Voting with a Logarithmic Number of Cards
4. New Adder Protocols
5. Conclusion
### Outline of our results

<table>
<thead>
<tr>
<th>Half adder</th>
<th>Using existing AND/XOR/COPY protocols</th>
<th>Devising a tailor-made half adder</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Half adder diagram" /></td>
<td><img src="image" alt="Diagram of 10 cards" /></td>
<td><img src="image" alt="Diagram of 8 cards" /></td>
</tr>
<tr>
<td><strong>Voting</strong></td>
<td><strong>2\lceil\log n\rceil + 8</strong></td>
<td><strong>2\lceil\log n\rceil + 6</strong></td>
</tr>
<tr>
<td><img src="image" alt="Voting diagram" /></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[\# of cards]
COPPY

\[ a \wedge b = 0 \]

\[ a \oplus b = a \wedge b \]
\[ a \oplus r \quad a \oplus r \quad \overline{r} \land b \quad r \land b \]

\[ \overline{a \oplus b} \quad a \land b \]
\[ a \land b = 0 \lor a \lor b = 1 \]
Outline of our results

Using existing AND/XOR/COPY protocols
Devising a tailor-made half adder

<table>
<thead>
<tr>
<th>Half adder</th>
<th>10</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting</td>
<td>$2\lceil\log n\rceil + 8$</td>
<td>$2\lceil\log n\rceil + 6$</td>
</tr>
</tbody>
</table>

[# of cards]
Outline of our results

Using existing AND/XOR/COPY protocols
Devising a tailor-made half adder

<table>
<thead>
<tr>
<th></th>
<th>10</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Half adder</strong></td>
<td><img src="image1" alt="Diagram" /></td>
<td><img src="image2" alt="Diagram" /></td>
</tr>
<tr>
<td><strong>Voting</strong></td>
<td>$2 \lceil \log n \rceil + 8$</td>
<td>$2 \lceil \log n \rceil + 6$</td>
</tr>
</tbody>
</table>

[# of cards]
### Outline of our results

#### Using existing AND/XOR/COPY protocols

#### Devising a tailor-made half adder

<table>
<thead>
<tr>
<th></th>
<th>Voting</th>
<th>Half adder</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>$2 \lceil \log n \rceil + 8$</td>
<td>$2 \lceil \log n \rceil + 6$</td>
</tr>
</tbody>
</table>

[Image of half adder circuit]

[Image of voting process]

[Image of number of cards]
Contents

1. Introduction
2. Known Protocols
3. Voting with a Logarithmic Number of Cards
4. New Adder Protocols
5. Conclusion
We gave a 8-card secure half adder protocol.

\[ \spadesuit \heartsuit = 0 \] \[ \spadesuit \clubsuit = 1 \]

It enables us to conduct voting with \( 2 \lceil \log n \rceil + 6 \) cards.

\[ a \oplus b \] \[ a \land b \]
I hope card-based protocols would help you with
• intuitive explanation of crypto. to non-specialists
• education in classroom.

That’s all.
Thank you for your attention.

A (real) deck of cards available to the first several people; please contact the speaker.