Privacy in Non-Private Environments

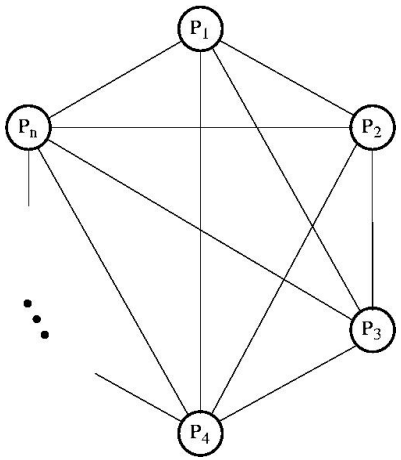
M. Bläser[1], A. Jakoby[2], M. Liśkiewicz[2], and B. Manthey[2]

[1]Institut für Theoretische Informatik
ETH Zürich, Switzerland

[2]Institut für Theoretische Informatik
Universität zu Lübeck, Germany

Private Computation - Motivating Example

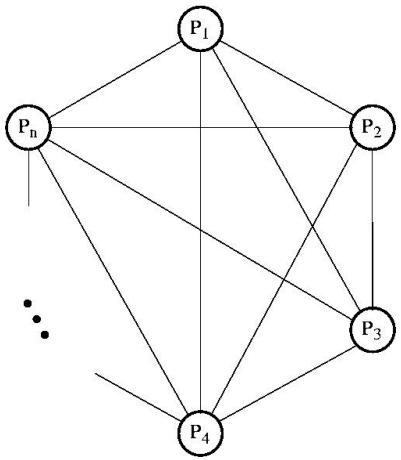
The secrete (*YES*/*NO*) voting:



- Problem: Decide whether the majority votes for *YES*.
- Constraint: After the voting **no** P_i gets any additional information.

Private Computation - Motivating Example

The secrete (*YES*/*NO*) voting:



- Problem: Decide whether the majority votes for *YES*.
- Constraint: After the voting **no** P_i gets any additional information.
- The players are assumed to be *honest but curious*.

Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$

Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$
2. parties can exchange data via the links of the network

Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$
2. parties can exchange data via the links of the network
3. every party P_i has an individual secret x_i

Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$
2. parties can exchange data via the links of the network
3. every party P_i has an individual secret x_i
4. parties have unlimited computational power

Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$
2. parties can exchange data via the links of the network
3. every party P_i has an individual secret x_i
4. parties have unlimited computational power
5. every party P_i can use a private random string R_i

Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$
2. parties can exchange data via the links of the network
3. every party P_i has an individual secret x_i
4. parties have unlimited computational power
5. every party P_i can use a private random string R_i

Compute a function $f(x_1, \ldots, x_n)$ on the network, such that no party gains some knowledge about the input of the other parties, that cannot be derived from the result of the function and its own input.

Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$
2. parties can exchange data via the links of the network
3. every party P_i has an individual secret x_i
4. parties have unlimited computational power
5. every party P_i can use a private random string R_i

Formally: For every pair $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ with $x_i = y_i$ and $f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$

$$\Pr[\ c \mid x, R_i] \quad = \quad \Pr[\ c \mid y, R_i]$$

for every communication sequence c seen by P_i.

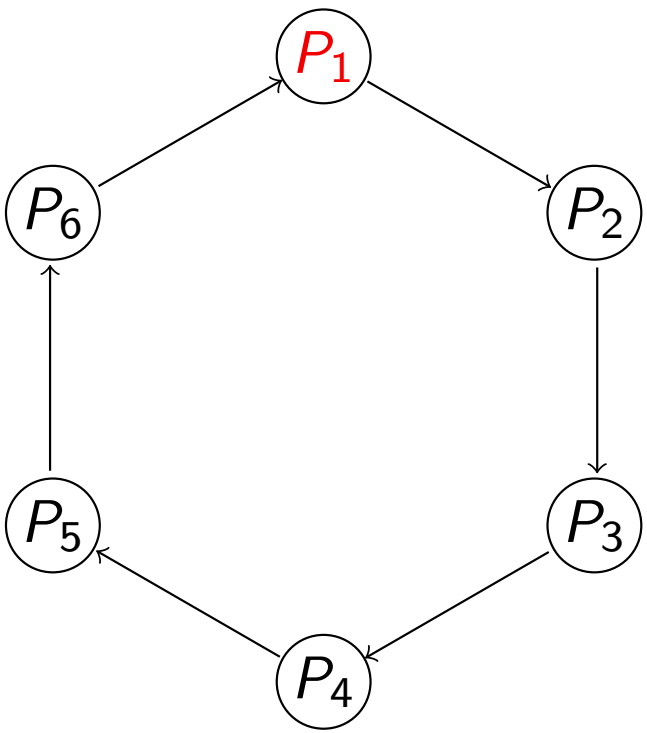
Private Computation in Information Theoretical Setting

1. given a network of n parties $P_1, \ldots, P_n$
2. parties can exchange data via the links of the network
3. every party P_i has an individual secret x_i
4. parties have unlimited computational power
5. every party P_i can use a private random string R_i

The Model:

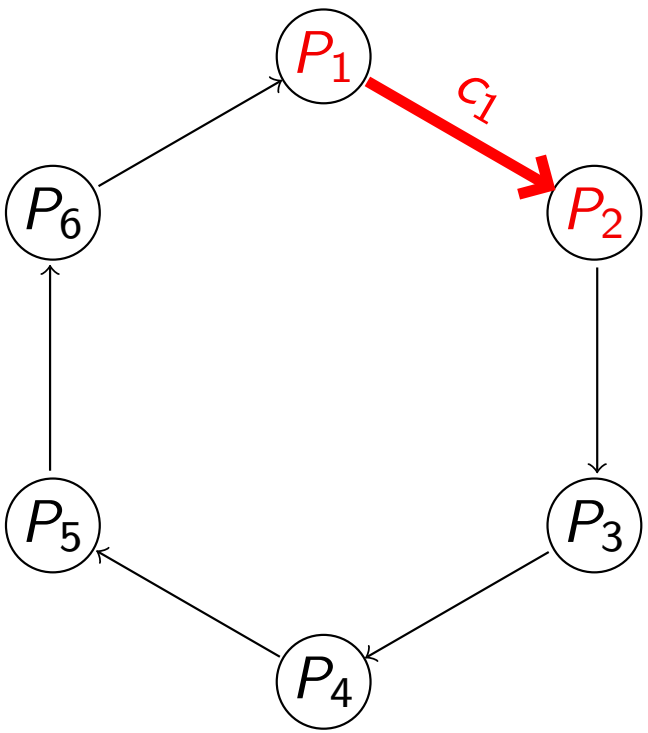
Ben-Or, Goldwasser, and Wigderson, STOC'88 and
Chaum, Crépau, and Damgard, STOC'88.

Example: Computing parity on a cycle



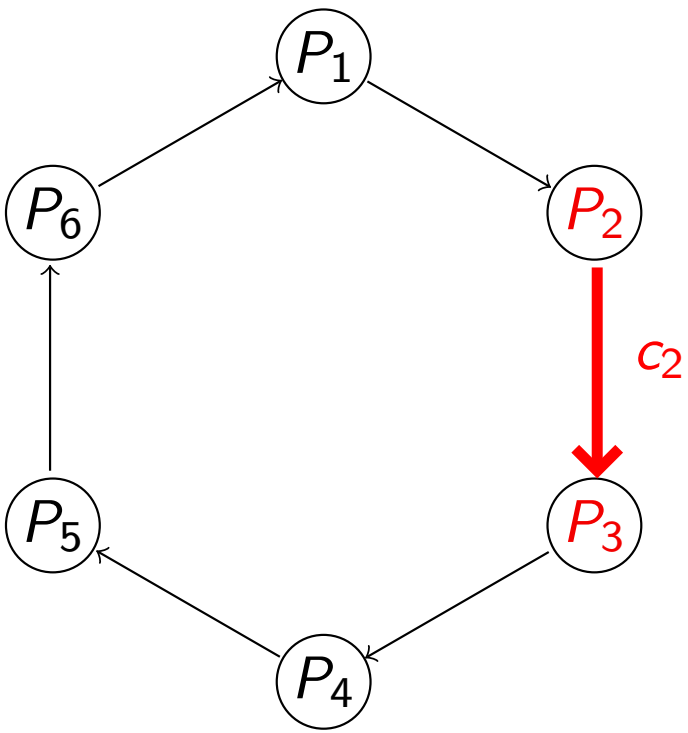
1. P_1 chooses $r \in_R \{0, 1\}$

Example: Computing parity on a cycle



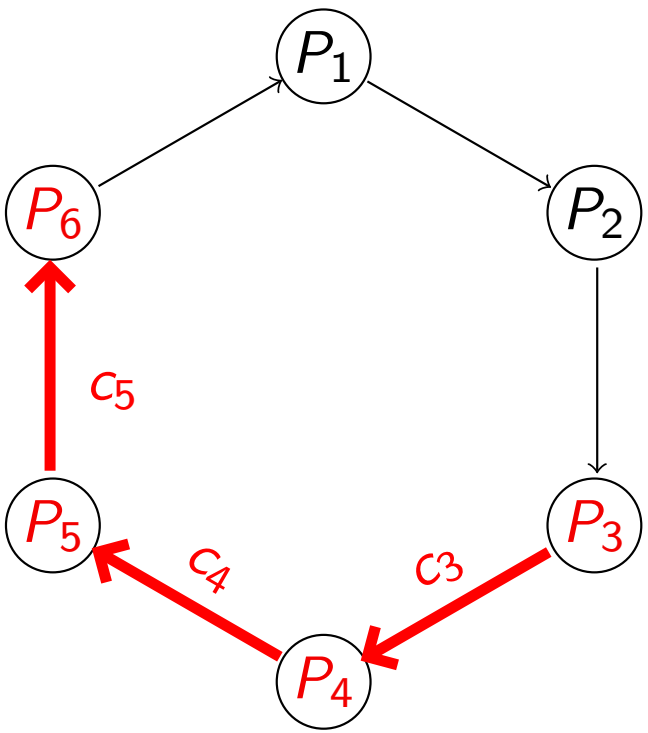
1. P_1 chooses $r \in_R \{0, 1\}$
2. P_1 sends $c_1 = r \oplus x_1$ to P_2

Example: Computing parity on a cycle



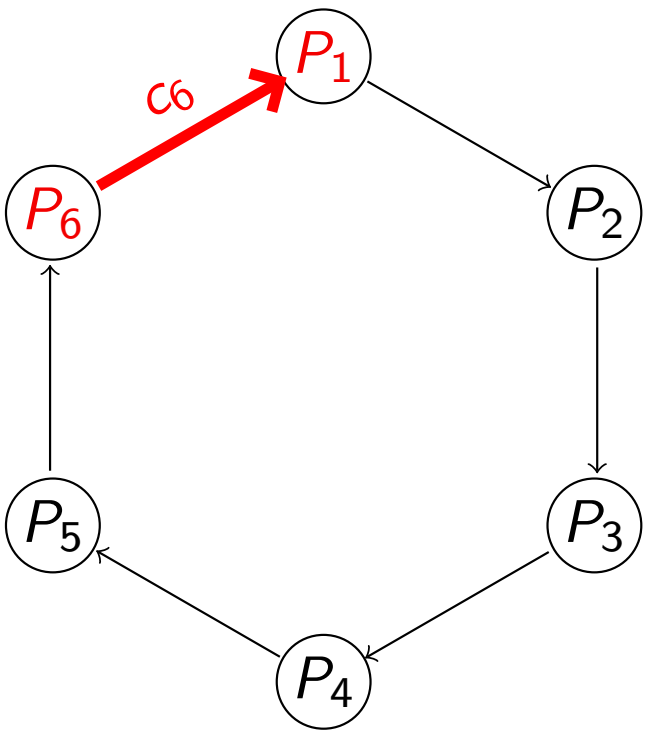
1. P_1 chooses $r \in_R \{0, 1\}$
2. P_1 sends $c_1 = r \oplus x_1$ to P_2
3. P_2 sends $c_2 = c_1 \oplus x_2$ to P_3

Example: Computing parity on a cycle



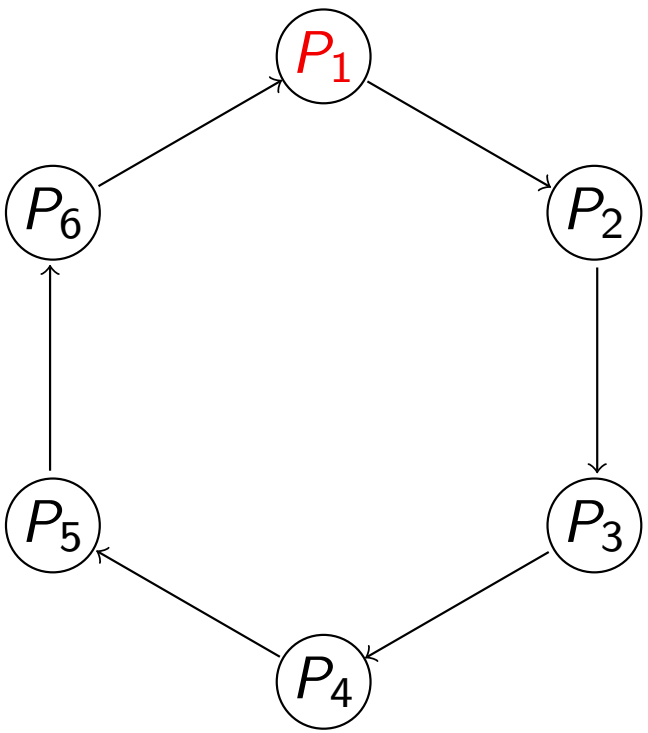
1. P_1 chooses $r \in_R \{0,1\}$
2. P_1 sends $c_1 = r \oplus x_1$ to P_2
3. P_2 sends $c_2 = c_1 \oplus x_2$ to P_3
4. and so on ...

Example: Computing parity on a cycle



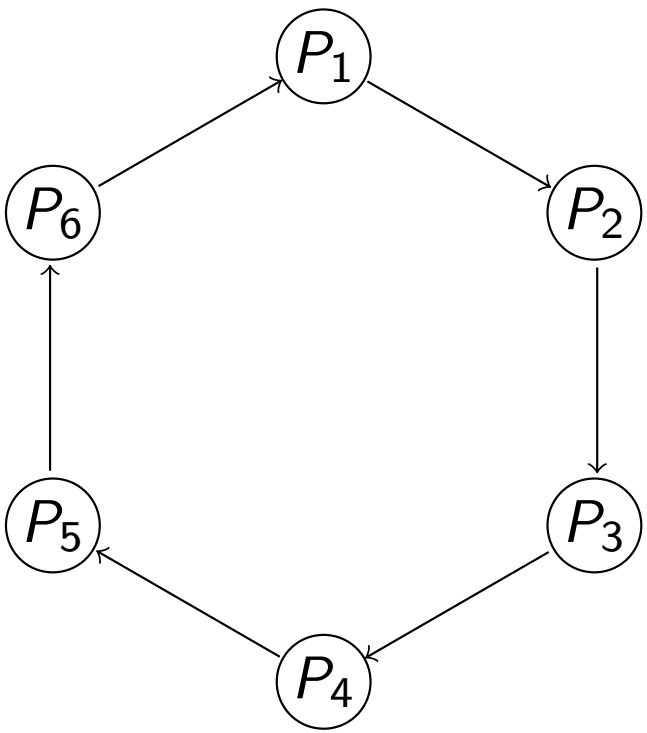
1. P_1 chooses $r \in_R \{0,1\}$
2. P_1 sends $c_1 = r \oplus x_1$ to P_2
3. P_2 sends $c_2 = c_1 \oplus x_2$ to P_3
4. and so on ...
5. P_6 sends $c_6 = c_5 \oplus x_6$ to P_1

Example: Computing parity on a cycle



1. P_1 chooses $r \in_R \{0,1\}$
2. P_1 sends $c_1 = r \oplus x_1$ to P_2
3. P_2 sends $c_2 = c_1 \oplus x_2$ to P_3
4. and so on ...
5. P_6 sends $c_6 = c_5 \oplus x_6$ to P_1
6. P_1 computes
   $c_6 \oplus r = x_1 \oplus \ldots \oplus x_6$

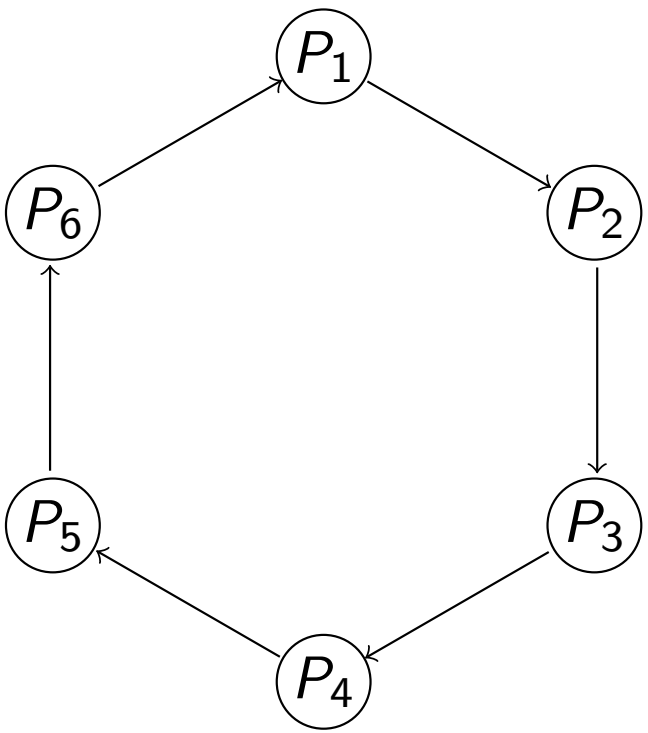
Example: Computing parity on a cycle



1. P_1 chooses $r \in_R \{0,1\}$
2. P_1 sends $c_1 = r \oplus x_1$ to P_2
3. P_2 sends $c_2 = c_1 \oplus x_2$ to P_3
4. and so on ...
5. P_6 sends $c_6 = c_5 \oplus x_6$ to P_1
6. P_1 computes
   $c_6 \oplus r = x_1 \oplus \ldots \oplus x_6$

Every P_i $(i \neq 1)$ receives 0 with prob. $\frac{1}{2}$ and 1 with prob. $\frac{1}{2}$.

Example: Computing parity on a cycle



1. P_1 chooses $r \in_R \{0,1\}$
2. P_1 sends $c_1 = r \oplus x_1$ to P_2
3. P_2 sends $c_2 = c_1 \oplus x_2$ to P_3
4. and so on ...
5. P_6 sends $c_6 = c_5 \oplus x_6$ to P_1
6. P_1 computes
   $c_6 \oplus r = x_1 \oplus \ldots \oplus x_6$

Every P_i $(i \neq 1)$ receives 0 with prob. $\frac{1}{2}$ and 1 with prob. $\frac{1}{2}$.
P_i learns nothing!

Non-Private Environments

- All Boolean functions can privately be computed on any 2-connected network.

Non-Private Environments

- All Boolean functions can privately be computed on any 2-connected network.
- Non 2-connected networks: a complete characterisation of functions which *cannot* be computed privately by
  - two players
    Kushilevitz SIAM J.Disc.Math.'92 and Beaver TR, Harvard'89;

Non-Private Environments

- All Boolean functions can privately be computed on any 2-connected network.
- Non 2-connected networks: a complete characterisation of functions which *cannot* be computed privately by
  - two players
    Kushilevitz SIAM J.Disc.Math.'92 and Beaver TR, Harvard'89;
  - arbitrary connected but not 2-connected networks
    Bläser, Jakoby, Liśkiewicz, and Manthey, CRYPTO'02:
    Examples: `Parity, OR, AND ...`
    Corollary: no non-degenerate function can privately be computed if the network consists of three or more blocks.

Non-Private Environments

- All Boolean functions can privately be computed on any 2-connected network.
- Non 2-connected networks: a complete characterisation of functions which *cannot* be computed privately by
  - two players
    Kushilevitz SIAM J.Disc.Math.'92 and Beaver TR, Harvard'89;
  - arbitrary connected but not 2-connected networks
    Bläser, Jakoby, Liśkiewicz, and Manthey, CRYPTO'02:
    Examples: `Parity, OR, AND ...`
    Corollary: no non-degenerate function can privately be computed if the network consists of three or more blocks.
- Problem:
  How functions that cannot privately be computed can still be computed while maintaining as much privacy as possible?

Previous Results

Bar-Yehuda, Chor, Kushilevitz, and Orlitsky, IEEE Trans.Inf.Th.'93:

- Two parties, each holding one n-bit input.

Previous Results

Bar-Yehuda, Chor, Kushilevitz, and Orlitsky, IEEE Trans.Inf.Th.'93:

- Two parties, each holding one n-bit input.
- Minimum leakage of information for functions that are *not* privately computable.

Previous Results

Bar-Yehuda, Chor, Kushilevitz, and Orlitsky, IEEE Trans.Inf.Th.'93:

- Two parties, each holding one n-bit input.
- Minimum leakage of information for functions that are *not* privately computable.
- For several functions: tight bounds on the minimum amount of information that must be learned.

Previous Results

Bar-Yehuda, Chor, Kushilevitz, and Orlitsky, IEEE Trans.Inf.Th.'93:

- Two parties, each holding one n-bit input.
- Minimum leakage of information for functions that are *not* privately computable.
- For several functions: tight bounds on the minimum amount of information that must be learned.
- Sacrificing some privacy can reduce the number of messages required during the computation.

Our Contribution - Information Source

- Let $\mathcal{A}$ be a protocol computing f on G. Let $c_1, c_2, c_3, \ldots$ be a fixed enumeration of all communication strings seen by any player during the execution of $\mathcal{A}$. Let for P_i

$$\mu_x(c_k) := \Pr[\ c_k \mid x, R_i],$$

where R_i is random string provided to P_i.

Our Contribution - Information Source

- Let $\mathcal{A}$ be a protocol computing f on G. Let $c_1, c_2, c_3, \ldots$ be a fixed enumeration of all communication strings seen by any player during the execution of $\mathcal{A}$. Let for P_i

$$\mu_x(c_k) := \Pr[\ c_k \mid x, R_i],$$

  where R_i is random string provided to P_i.
- The *information source* $\mathcal{S}_{\mathcal{A}}(i, a, b, R_i)$ is

$$\{\ (\mu_x(c_1), \mu_x(c_2), \ldots)\ \mid\ x \in \{0,1\}^n \wedge x_i = a \wedge f(x) = b\ \}$$

  and $\ell_{\mathcal{A}}(i, a, b) = \max_{R_i} \log |\mathcal{S}_{\mathcal{A}}(i, a, b, R_i)|$.

Our Contribution - Information Source

- Let $\mathcal{A}$ be a protocol computing f on G. Let $c_1, c_2, c_3, \ldots$ be a fixed enumeration of all communication strings seen by any player during the execution of $\mathcal{A}$. Let for P_i

$$\mu_x(c_k) := \Pr[\ c_k \mid x, R_i],$$

  where R_i is random string provided to P_i.
- The *information source* $\mathcal{S}_{\mathcal{A}}(i, a, b, R_i)$ is

$$\{\ (\mu_x(c_1), \mu_x(c_2), \ldots)\ \mid\ x \in \{0,1\}^n \wedge x_i = a \wedge f(x) = b\ \}$$

  and $\ell_{\mathcal{A}}(i, a, b) = \max_{R_i} \log |\mathcal{S}_{\mathcal{A}}(i, a, b, R_i)|$.
- If f is n-ary then for $G = (V, E)$ with $|V| = n$ define

$$\ell_G(i, a, b) := \min_{\mathcal{A}} \{\ell_{\mathcal{A}}(i, a, b) \mid \mathcal{A} \text{ is protocol for } f \text{ on } G\}.$$

Bridge Nodes

Theorem *Any protocol can be modified such that the loss to all internal players is zero, while the loss to any bridge player does not increase.*

Extracting Information from Probability Distribution

Let $\mathcal{A}$ be a protocol for f on G, P_i be a bridge and $a, b \in \{0,1\}$.
Define $X := \{x \in \{0,1\}^n \mid x_i = a \wedge f(x) = b\}$
and, for any communication string c,

$$\psi(c) := \{x \in X \mid \mu_x(c) > 0\}.$$

- If $\ell_{\mathcal{A}}(i,a,b) = \ell_G(i,a,b) = 0$, then $\psi(c) = X$ or $\psi(c) = \emptyset$.

Extracting Information from Probability Distribution

Let $\mathcal{A}$ be a protocol for f on G, P_i be a bridge and $a, b \in \{0,1\}$.
Define $X := \{x \in \{0,1\}^n \mid x_i = a \wedge f(x) = b\}$
and, for any communication string c,

$$\psi(c) := \{x \in X \mid \mu_x(c) > 0\}.$$

- If $\ell_{\mathcal{A}}(i,a,b) = \ell_G(i,a,b) = 0$, then $\psi(c) = X$ or $\psi(c) = \emptyset$.
- Theorem *If $\ell_G(i,a,b) > 0$, then for any $\mathcal{A}$ and every c that can be observed by P_i on $x \in X$,*

  1) $\psi(c)$ is a non-trivial subset of X and

  2) there exist at least $2^{\ell_G(i,a,b)}$ different such sets.

Extracting Information from Probability Distribution

Let $\mathcal{A}$ be a protocol for f on G, P_i be a bridge and $a, b \in \{0,1\}$.
Define $X := \{x \in \{0,1\}^n \mid x_i = a \wedge f(x) = b\}$
and, for any communication string c,

$$\psi(c) := \{x \in X \mid \mu_x(c) > 0\}.$$

Recall: $F(\mu, \mu') = \sum_c \sqrt{\mu(c) \cdot \mu'(c)}$.

Theorem *If $\mathcal{A}$ is optimal for P_i on a and b then for all R_i and all $\mu \neq \mu'$ in $\mathcal{S}_{\mathcal{A}}(i, a, b, R_i)$ we have $F(\mu, \mu') = 0$.*

Hence, in order to gain information, P_i can distinguish the distributions from the actual communication he observes.

Communication Complexity and Private Computation

Let G has two blocks and P_i be the bridge node.

Theorem (Two-Blocs Networks)
If f has communication complexity C then

$$\ell_G(i, a, b) \leq 2C \quad \text{for any } a, b.$$

If for computing f: $\ell_G(i, a, b) \leq \lambda$ for any a, b then the communication complexity of f is bounded by $6\lambda + O(1)$.

k-Phases Protocols

- Within a phase, a *bridge player* may exchange messages only once with each block he belongs to.

k-Phases Protocols

- Within a phase, a *bridge player* may exchange messages only once with each block he belongs to.
- 1-Phase Protocols on G
  - Let G consists of d blocks that all share bridge P.
  - In a 1-phase protocol: P communicates only once with each block he belongs to.
  - The loss of the protocol may depend on the order in which P communicates with the blocks.

k-Phases Protocols

- Within a phase, a *bridge player* may exchange messages only once with each block he belongs to.
- 1-Phase Protocols on G
  - Let G consists of d blocks that all share bridge P.
  - In a 1-phase protocol: P communicates only once with each block he belongs to.
  - The loss of the protocol may depend on the order in which P communicates with the blocks.

Corollary *For symmetric functions, optimum order is to sort the blocks by increasing size.*

k-Phases Protocols

- Within a phase, a *bridge player* may exchange messages only once with each block he belongs to.
- 1-Phase Protocols on G
  - Let G consists of d blocks that all share bridge P.
  - In a 1-phase protocol: P communicates only once with each block he belongs to.
  - The loss of the protocol may depend on the order in which P communicates with the blocks.

Corollary *For symmetric functions, optimum order is to sort the blocks by increasing size.*

Theorem *For any symmetric function f there exists a 1-phase protocol $\mathcal{A}$ s.t. for every 1-phase $\mathcal{A}'$*

$$\ell_{\mathcal{A}}(i,a,b) \leq \ell_{\mathcal{A}'}(i,a,b) \quad \text{for all } i,a,b.$$

A Phase Hierarchy

Theorem *For any k there is f such that every $(k-1)$-phase protocol for f has an information loss that is exponentially greater than that of the best k-phase protocol.*

Open Problems

- Optimal protocols for some concrete functions.
- *Does exist for any f a protocol $\mathcal{A}$ on G s.t.*

$$\ell_{\mathcal{A}}(i, a, b) = \ell_G(i, a, b) \quad \textit{for all } i, a, b?$$

(The answer is negative for 1-phase protocols.)