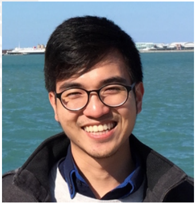
A Parameterized Framework for Hardness of Approximation

Karthik C. S.
(Weizmann Institute of Science)

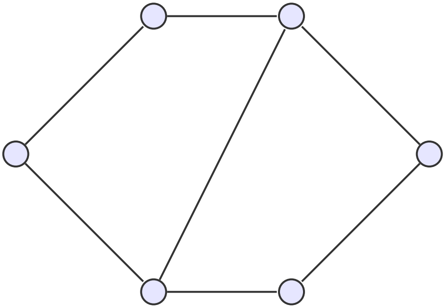
Joint work with



Bundit Laekhanukit
(Shanghai University of Finance and Economics)

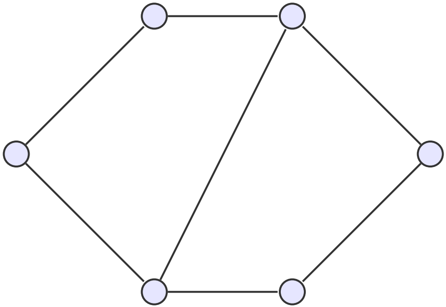
Pasin Manurangsi
(UC Berkeley)

Dominating Set Problem



$G(V, E)$

Dominating Set Problem

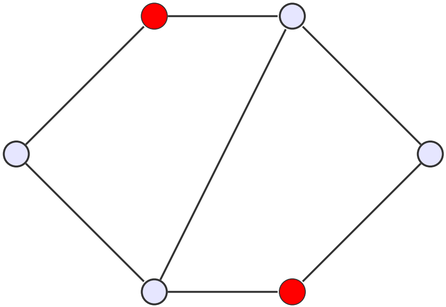


$G(V, E)$

$S \subseteq V$ is a Dominating Set of G if $\forall u \in V$:

- $u \in S$, or
- $\exists\, v \in S$ such that $(u, v) \in E$

Dominating Set Problem

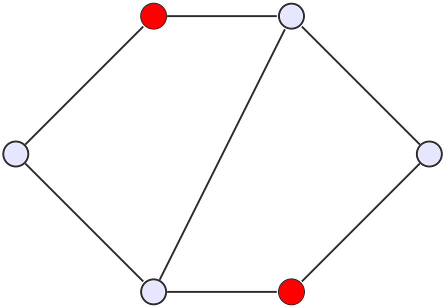


$G(V, E)$

$S \subseteq V$ is a Dominating Set of G if $\forall u \in V$:

- $u \in S$, or
- $\exists\, v \in S$ such that $(u, v) \in E$

Dominating Set Problem



$G(V, E)$

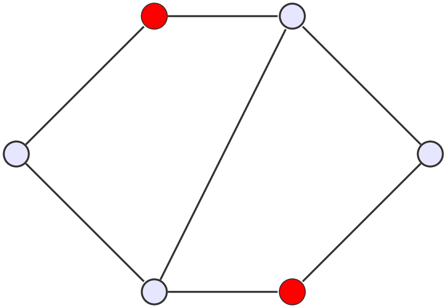
$S \subseteq V$ is a Dominating Set of G if $\forall u \in V$:

- $u \in S$, or
- $\exists\, v \in S$ such that $(u, v) \in E$

Computational Problem: Given G and $k \in \mathbb{N}$, determine if $\exists\, S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

Dominating Set Problem



$G(V, E)$

$S \subseteq V$ is a Dominating Set of G if $\forall u \in V$:

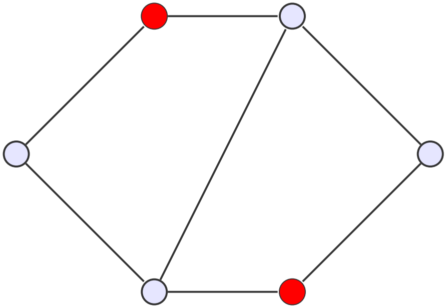
- $u \in S$, or
- $\exists\, v \in S$ such that $(u, v) \in E$

Computational Problem: Given G and $k \in \mathbb{N}$, determine if $\exists\, S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

$\longrightarrow$ NP-Complete [Karp'72]

Dominating Set Problem



$G(V, E)$

$\longrightarrow$ $\ln |V|$ approximation is in P [Slavík'96]

$S \subseteq V$ is a Dominating Set of G if $\forall u \in V$:

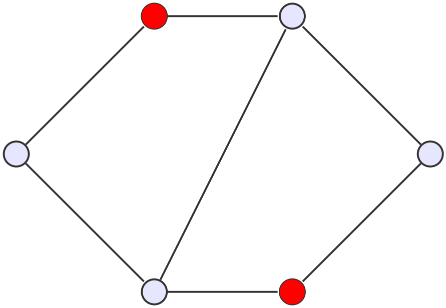
- $u \in S$, or
- $\exists\, v \in S$ such that $(u, v) \in E$

Computational Problem: Given G and $k \in \mathbb{N}$, determine if $\exists\, S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

$\longrightarrow$ NP-Complete [Karp'72]

Dominating Set Problem



$G(V, E)$

$S \subseteq V$ is a Dominating Set of G if $\forall u \in V$:

- $u \in S$, or
- $\exists\, v \in S$ such that $(u, v) \in E$

Computational Problem: Given G and $k \in \mathbb{N}$, determine if $\exists\, S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

$\longrightarrow$ NP-Complete [Karp'72]

$\longrightarrow$ $\ln|V|$ approximation is in P [Slavík'96]

$\longrightarrow$ $(1-\varepsilon)\ln|V|$ approximation is NP-Complete [DS'14]

Parameterized Dominating Set Problem

Computational Problem: Given G and parameter $k \in \mathbb{N}$, determine if $\exists\, S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

Parameterized Dominating Set Problem

Computational Problem: Given G and parameter $k \in \mathbb{N}$, determine if $\exists\, S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

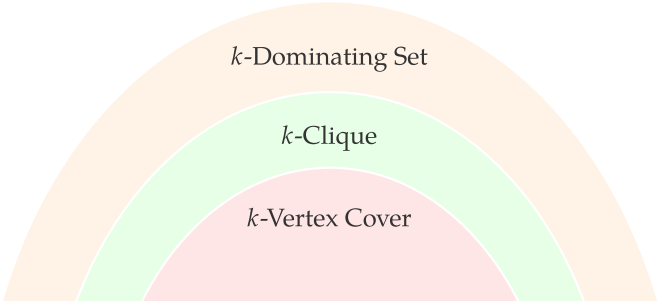
Fixed Parameter Tractability (FPT): The problem can be decided in $F(k) \cdot \text{poly}(|V|)$ time, for some computable function F.

Parameterized Dominating Set Problem

Computational Problem: Given G and parameter $k \in \mathbb{N}$, determine if $\exists\ S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

Fixed Parameter Tractability (FPT): The problem can be decided in $F(k) \cdot \text{poly}(|V|)$ time, for some computable function F.

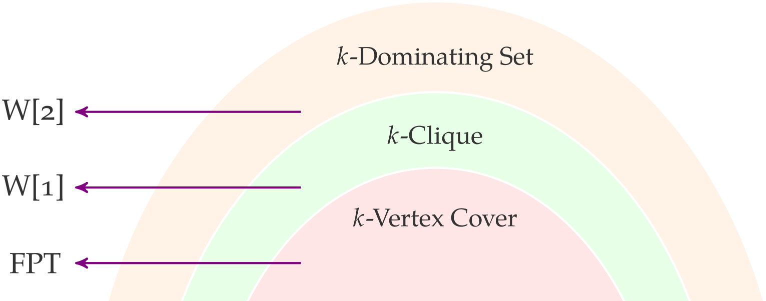


Parameterized Dominating Set Problem

Computational Problem: Given G and parameter $k \in \mathbb{N}$, determine if $\exists\, S \subseteq V$:

- S is a Dominating Set of G
- $|S| \leq k$

Fixed Parameter Tractability (FPT): The problem can be decided in $F(k) \cdot \mathrm{poly}(|V|)$ time, for some computable function F.



Parameterized Complexity of Dominating Set Problem

Given graph on N vertices and parameter k:

- W[2] complete [DF'95]

Parameterized Complexity of Dominating Set Problem

Given graph on N vertices and parameter k:

- W[2] complete [DF'95]
- Trivial Algorithm: $O(N^{k+1})$ time

Parameterized Complexity of Dominating Set Problem

Given graph on N vertices and parameter k:

- W[2] complete [DF'95]
- Trivial Algorithm: $O(N^{k+1})$ time
- State of the Art: $N^{k+o(1)}$ time [EG'04, PW'10]

Parameterized Complexity of Dominating Set Problem

Given graph on N vertices and parameter k:

- W[2] complete [DF'95]
- Trivial Algorithm: $O(N^{k+1})$ time
- State of the Art: $N^{k+o(1)}$ time [EG'04, PW'10]
- No $N^{o(k)}$ time algorithm assuming ETH [CHKX'06]

Parameterized Complexity of Dominating Set Problem

Given graph on N vertices and parameter k:

- W[2] complete [DF'95]
- Trivial Algorithm: $O(N^{k+1})$ time
- State of the Art: $N^{k+o(1)}$ time [EG'04, PW'10]
- No $N^{o(k)}$ time algorithm assuming ETH [CHKX'06]

There exists $\delta > 0$ such that no algorithm can solve 3-CNF-SAT in $O(2^{\delta n})$ time where n is the number of variables.

Parameterized Complexity of Dominating Set Problem

Given graph on N vertices and parameter k:

- W[2] complete [DF'95]
- Trivial Algorithm: $O(N^{k+1})$ time
- State of the Art: $N^{k+o(1)}$ time [EG'04, PW'10]
- No $N^{o(k)}$ time algorithm assuming ETH [CHKX'06]
- No $O(N^{k-\varepsilon})$ algorithm assuming SETH [PW'10]

Parameterized Complexity of Dominating Set Problem

Given graph on N vertices and parameter k:

- W[2] complete [DF'95]
- Trivial Algorithm: $O(N^{k+1})$ time
- State of the Art: $N^{k+o(1)}$ time [EG'04, PW'10]
- No $N^{o(k)}$ time algorithm assuming ETH [CHKX'06]
- No $O(N^{k-\varepsilon})$ algorithm assuming SETH [PW'10]

For every $\varepsilon > 0$, there exists $\ell(\varepsilon) \in \mathbb{N}$ such that no algorithm can solve ℓ-SAT in $O(2^{(1-\varepsilon)n})$ time where n is the number of variables.

FPT Approximability of Dominating Set Problem

FPT Approximability: The problem has a $T(k)$ approximation algorithm running in time $F(k) \cdot \text{poly}(N)$ time.

FPT Approximability of Dominating Set Problem

FPT Approximability: The problem has a $T(k)$ approximation algorithm running in time $F(k) \cdot \text{poly}(N)$ time.

Approximate Parameterized Dominating Set Problem: Given a graph G and parameter k distinguish between:

- $\exists$ a dominating set of size at most k
- There is no dominating set of size $T(k) \cdot k$

FPT Approximability of Dominating Set Problem

FPT Approximability: The problem has a $T(k)$ approximation algorithm running in time $F(k) \cdot \text{poly}(N)$ time.

Approximate Parameterized Dominating Set Problem: Given a graph G and parameter k distinguish between:

- $\exists$ a dominating set of size at most k
- There is no dominating set of size $T(k) \cdot k$

Is there some computable function T for which the above problem is in FPT?

Two decades later:

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]
- No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]
- No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming Gap-ETH [CCKLMNT'17]

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]
- No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming Gap-ETH [CCKLMNT'17]

There exists a constant $\delta > 0$ such that any algorithm that, on input a 3-SAT formula φ on n variables and $O(n)$ clauses, can distinguish between $\mathsf{SAT}(\varphi) = 1$ and $\mathsf{SAT}(\varphi) < 0.9$, must run in time at least $2^{\delta n}$.

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]
- No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming Gap-ETH [CCKLMNT'17]

★ Can we show every $T(k)$ approximation is W[1]-hard?

Previous Works

Two decades later:

- ◎ Any constant approximation is W[1]-hard [CL'16]
- ◎ No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]
- ◎ No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming Gap-ETH [CCKLMNT'17]

- ★ Can we show every $T(k)$ approximation is W[1]-hard?
- ★ Can we show no $T(k)$ approximation algorithm exists running in time $N^{o(k)}$, assuming ETH?

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]
- No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming Gap-ETH [CCKLMNT'17]

- ⋆ Can we show every $T(k)$ approximation is W[1]-hard?
- ⋆ Can we show no $T(k)$ approximation algorithm exists running in time $N^{o(k)}$, assuming ETH?
- ⋆ Can we show no $T(k)$ approximation algorithm exists running in time $N^{k-\varepsilon}$, assuming SETH?

Previous Works

Two decades later:

- Any constant approximation is [redacted] [CL'16]
- No $(\log k)^{1/4}$ approxima[redacted] in $N^{o(\sqrt{k})}$ time, assuming ETH [redacted]
- No $T(k)$ [redacted] algorithm in $N^{o(k)}$ time, assuming Gap-E[redacted]LMNT'17]

No PCP Machinery

- ★ Can we show every $T(k)$ approximation is W[1]-hard?
- ★ Can we show no $T(k)$ approximation algorithm exists running in time $N^{o(k)}$, assuming ETH?
- ★ Can we show no $T(k)$ approximation algorithm exists running in time $N^{k-\varepsilon}$, assuming SETH?

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]
- No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming Gap-ETH [CCKLMNT'17]

★ Can we show every $T(k)$ approximation is W[1]-hard?

★ Can we show no $T(k)$ approximation algorithm exists running in time $N^{o(k)}$, assuming ETH?

★ Can we show no $T(k)$ approximation algorithm exists running in time $N^{k-\varepsilon}$, assuming SETH?

Previous Works

Two decades later:

- Any constant approximation is W[1]-hard [CL'16]
- No $(\log k)^{1/4}$ approximation algorithm in $N^{o(\sqrt{k})}$ time, assuming ETH [CL'16]
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming Gap-ETH [CCKLMNT'17]

- ★ Can we show every $T(k)$ approximation is W[1]-hard?
- ★ Can we show no $T(k)$ approximation algorithm exists running in time $N^{o(k)}$, assuming ETH?
- ★ Can we show no $T(k)$ approximation algorithm exists running in time $N^{k-\varepsilon}$, assuming SETH?

Yes!

Our Results

- Any $T(k)$ approximation is W[1]-hard
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming ETH
- No $T(k)$ approximation algorithm in $N^{k-\varepsilon}$ time, assuming SETH
- No $T(k)$ approximation algorithm in $N^{\lceil k/2 \rceil - \varepsilon}$ time, assuming k-SUM Hypothesis

Our Results

- Any $T(k)$ approximation is W[1]-hard
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming ETH
- No $T(k)$ approximation algorithm in $N^{k-\varepsilon}$ time, assuming SETH
- No $T(k)$ approximation algorithm in $N^{\lceil k/2 \rceil - \varepsilon}$ time, assuming k-SUM Hypothesis

k-SUM Problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

Our Results

- Any $T(k)$ approximation is W[1]-hard
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming ETH
- No $T(k)$ approximation algorithm in $N^{k-\varepsilon}$ time, assuming SETH
- No $T(k)$ approximation algorithm in $N^{\lceil k/2 \rceil - \varepsilon}$ time, assuming k-SUM Hypothesis

k-SUM Problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

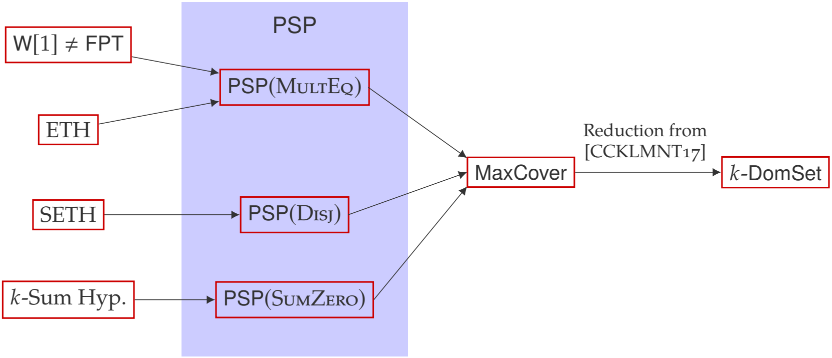
k-SUM Hypothesis: For every integer $k \geq 3$ and every $\varepsilon > 0$, no $O(N^{\lceil k/2 \rceil - \varepsilon})$ time algorithm can solve the k-SUM problem.

Our Results

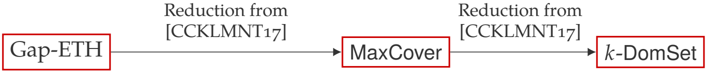
- Any $T(k)$ approximation is W[1]-hard
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming ETH
- No $T(k)$ approximation algorithm in $N^{k-\varepsilon}$ time, assuming SETH
- No $T(k)$ approximation algorithm in $N^{\lceil k/2 \rceil - \varepsilon}$ time, assuming k-SUM Hypothesis

All results obtained in an Unified Proof Framework!

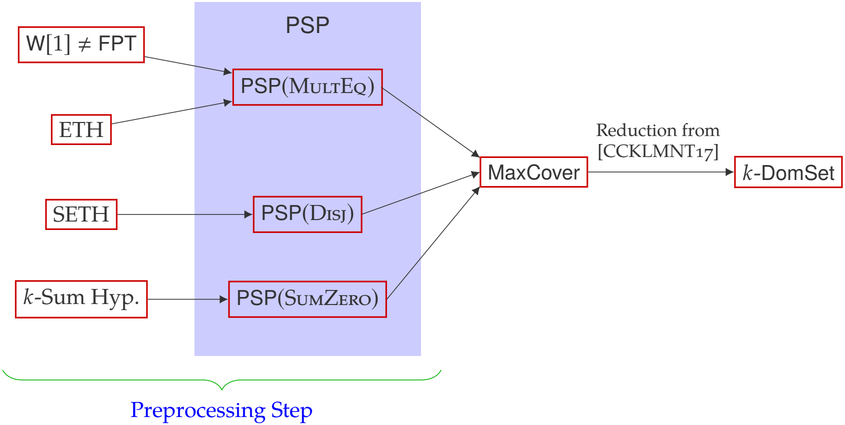
The Framework



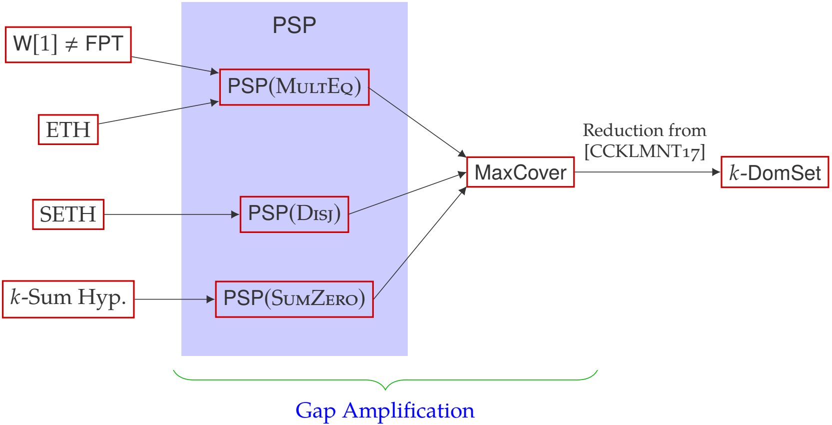
The Framework



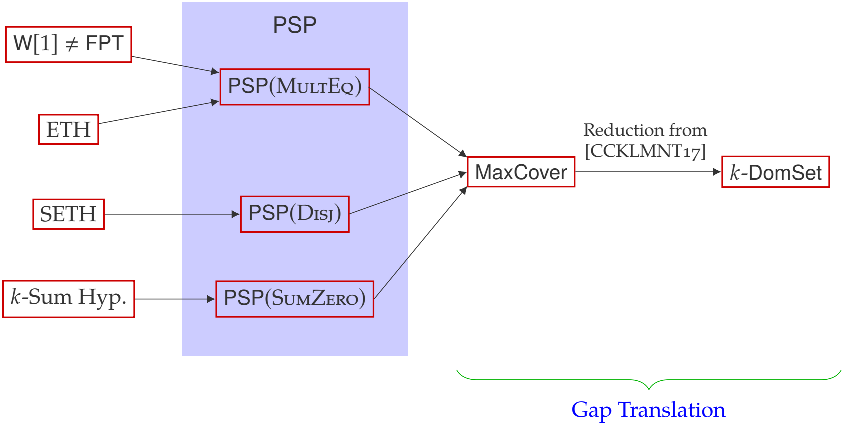
The Framework



The Framework



The Framework

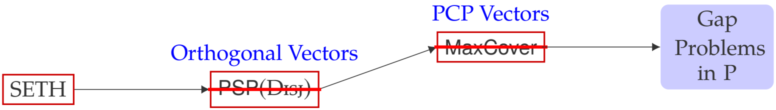


The Framework

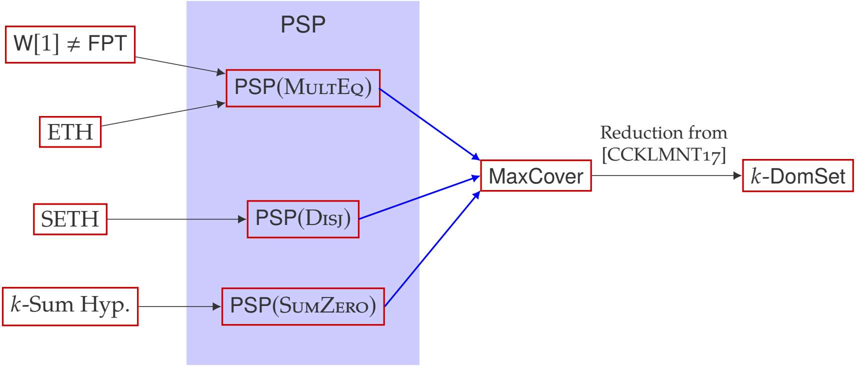
Generalization of Distributed PCP Framework [ARW'17]

$$\begin{tikzcd}
 & & \boxed{\text{MaxCover}} \\
\boxed{\text{SETH}} \arrow[r] & \boxed{\text{PSP}(\textsc{Disj})} \arrow[ur] &
\end{tikzcd}$$

Generalization of Distributed PCP Framework [ARW'17]

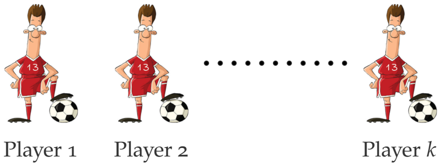


The Framework Revisited

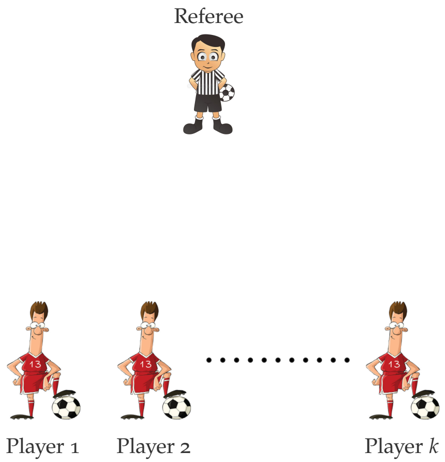


Simultaneous Message Passing (SMP) Model

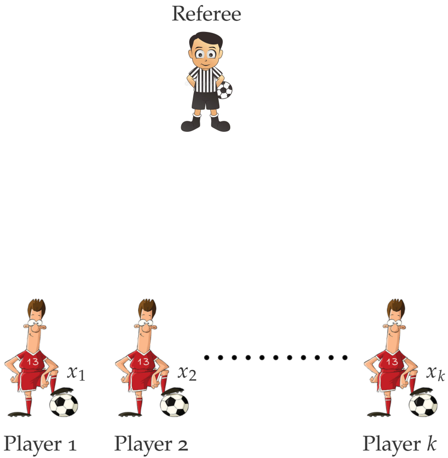
Simultaneous Message Passing (SMP) Model



Simultaneous Message Passing (SMP) Model



Simultaneous Message Passing (SMP) Model



Simultaneous Message Passing (SMP) Model

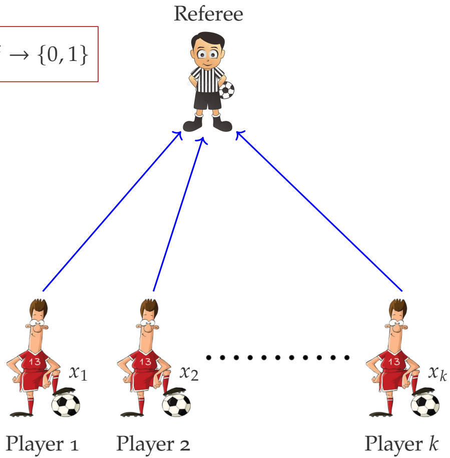
$$\boxed{f : \{0,1\}^{m \times k} \to \{0,1\}}$$

Referee

Player 1 Player 2 Player k

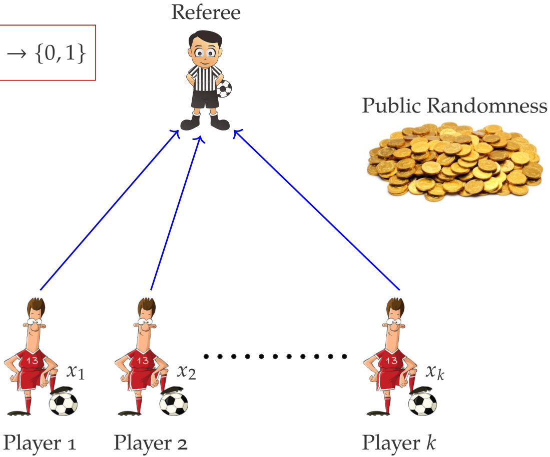
Simultaneous Message Passing (SMP) Model

$$f : \{0,1\}^{m \times k} \to \{0,1\}$$



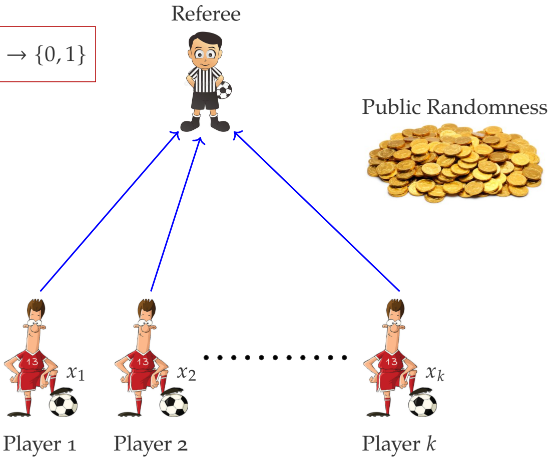
Simultaneous Message Passing (SMP) Model

$$\boxed{f : \{0,1\}^{m \times k} \to \{0,1\}}$$



Simultaneous Message Passing (SMP) Model

$$\boxed{f : \{0,1\}^{m\times k} \to \{0,1\}}$$



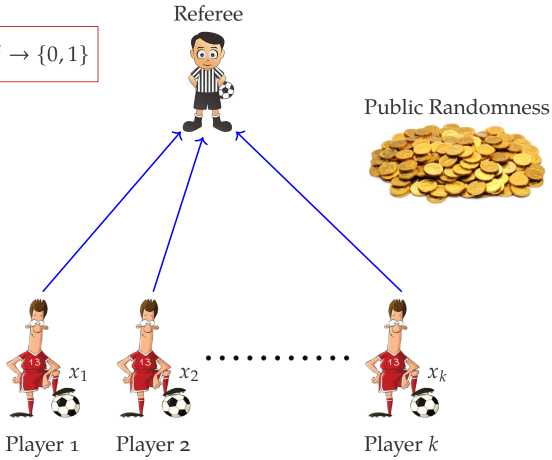
Randomized Protocols:

Completeness: If $f(x_1, \ldots, x_k) = 1$ then the referee always accepts

Soundness: If $f(x_1, \ldots, x_k) = 0$ then the referee accepts with probability $\leq s$

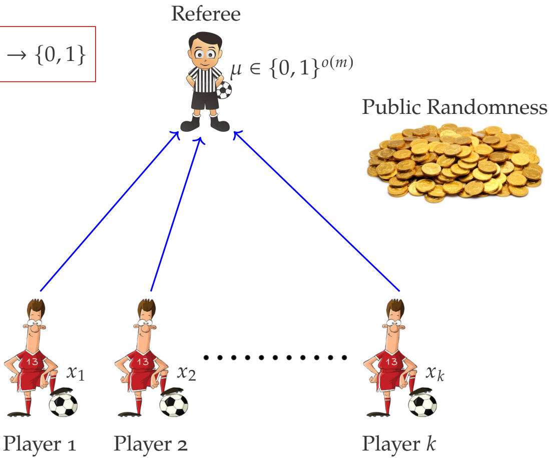
Simultaneous Message Passing (SMP) Model

$$\boxed{f : \{0,1\}^{m \times k} \rightarrow \{0,1\}}$$



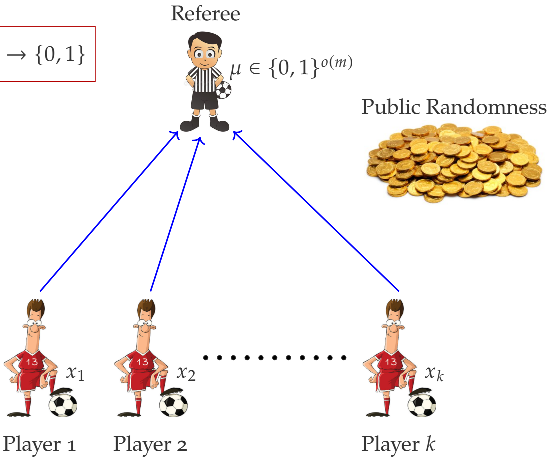
Simultaneous Message Passing (SMP) Model

$\boxed{f : \{0,1\}^{m \times k} \to \{0,1\}}$



Simultaneous Message Passing (SMP) Model

$\boxed{f : \{0,1\}^{m \times k} \to \{0,1\}}$

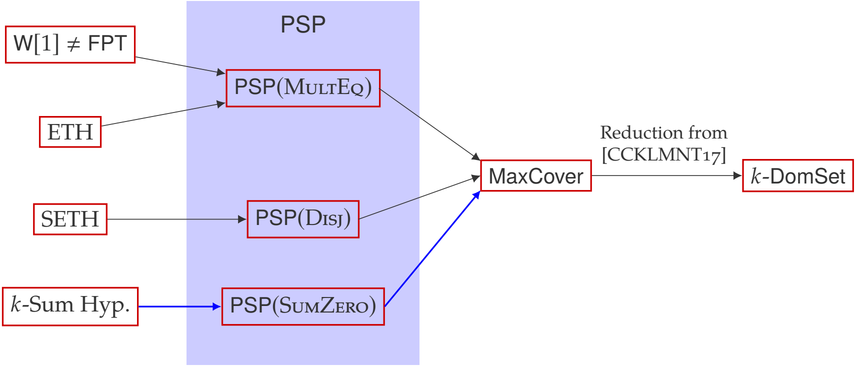


MA Protocols:

Completeness: If $f(x_1, \ldots, x_k) = 1$ then there exists μ for which referee always accepts

Soundness: If $f(x_1, \ldots, x_k) = 0$ then for all μ, the referee accepts with probability $\leq s$

The Framework Revisited



k-sum to Maxcover: Proof Sketch

k-SUM problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

k-sum to Maxcover: Proof Sketch

k-SUM problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

SumZero problem: Player i is given $x_i \in [-N^{2k}, N^{2k}]$ as input. Referee wants to determine whether $\sum_{i \in [k]} x_i = 0$.

k-sum to Maxcover: Proof Sketch

k-SUM problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

SumZero problem: Player i is given $x_i \in [-N^{2k}, N^{2k}]$ as input. Referee wants to determine whether $\sum_{i \in [k]} x_i = 0$.

Consider the following randomized protocol for SumZero [Nisan'94]:

1. The players and referee jointly draw a prime p^* in $\{p_1, \ldots, p_r\}$ (log r random bits)
2. Player i sends x_i mod p^* to the referee (log p^* bits)
3. The referee accepts if the sum of all the numbers he receives is zero

k-sum to Maxcover: Proof Sketch

k-SUM problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

SumZero problem: Player i is given $x_i \in [-N^{2k}, N^{2k}]$ as input. Referee wants to determine whether $\sum_{i \in [k]} x_i = 0$.

Consider the following randomized protocol for SumZero [Nisan'94]:

1. The players and referee jointly draw a prime p^* in $\{p_1, \ldots, p_r\}$ (log r random bits)
2. Player i sends x_i mod p^* to the referee (log p^* bits)
3. The referee accepts if the sum of all the numbers he receives is zero

Completeness: If $\sum_{i \in [k]} x_i = 0$ then $\sum_{i \in [k]} x_i \bmod p^* = 0$

k-sum to Maxcover: Proof Sketch

k-SUM problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

SumZero problem: Player i is given $x_i \in [-N^{2k}, N^{2k}]$ as input. Referee wants to determine whether $\sum_{i \in [k]} x_i = 0$.

Consider the following randomized protocol for SumZero [Nisan'94]:

1. The players and referee jointly draw a prime p^* in $\{p_1, \ldots, p_r\}$ ($\log r$ random bits)
2. Player i sends $x_i \bmod p^*$ to the referee ($\log p^*$ bits)
3. The referee accepts if the sum of all the numbers he receives is zero

Completeness: If $\sum_{i \in [k]} x_i = 0$ then $\sum_{i \in [k]} x_i \bmod p^* = 0$

Soundness: If $\sum_{i \in [k]} x_i \neq 0$ then the number of prime factors of $\sum_{i \in [k]} x_i$ is at most $r^* = 2k \log N + \log k$.

k-sum to Maxcover: Proof Sketch

k-SUM problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

SumZero problem: Player i is given $x_i \in [-N^{2k}, N^{2k}]$ as input. Referee wants to determine whether $\sum_{i \in [k]} x_i = 0$.

Consider the following randomized protocol for SumZero [Nisan'94]:

1. The players and referee jointly draw a prime p^* in $\{p_1, \ldots, p_r\}$ (log r random bits)
2. Player i sends $x_i \bmod p^*$ to the referee (log p^* bits)
3. The referee accepts if the sum of all the numbers he receives is zero

Completeness: If $\sum_{i \in [k]} x_i = 0$ then $\sum_{i \in [k]} x_i \bmod p^* = 0$

Soundness: If $\sum_{i \in [k]} x_i \neq 0$ then the number of prime factors of $\sum_{i \in [k]} x_i$ is at most $r^* = 2k \log N + \log k$. Therefore if $r \geq 2r^*$ then the referee rejects with probability $\geq 1/2$

k-sum to Maxcover: Proof Sketch

k-SUM problem: Given $A_1, \ldots, A_k \subseteq [-N^{2k}, N^{2k}]$ where $N = \sum_{i \in [k]} |A_i|$, determine whether there exist $x_i \in A_i, \forall i \in [k]$ such that $\sum_{i \in [k]} x_i = 0$.

SumZero problem: Player i is given $x_i \in [-N^{2k}, N^{2k}]$ as input. Referee wants to determine whether $\sum_{i \in [k]} x_i = 0$.

Consider the following randomized protocol for SumZero [Nisan'94]:

1. The players and referee jointly draw a prime p^* in $\{p_1, \ldots, p_r\}$ ($\log r$ random bits)
2. Player i sends $x_i \bmod p^*$ to the referee ($\log p^*$ bits)
3. The referee accepts if the sum of all the numbers he receives is zero

Completeness: If $\sum_{i \in [k]} x_i = 0$ then $\sum_{i \in [k]} x_i \bmod p^* = 0$

Soundness: If $\sum_{i \in [k]} x_i \neq 0$ then the number of prime factors of $\sum_{i \in [k]} x_i$ is at most $r^* = 2k \log N + \log k$. Therefore if $r \geq 2r^*$ then the referee rejects with probability $\geq 1/2$

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: $1/2$

k-sum to Maxcover: Proof Sketch (Continued)

Parameters of the SumZero protocol [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: $1/2$

k-sum to Maxcover: Proof Sketch (Continued)

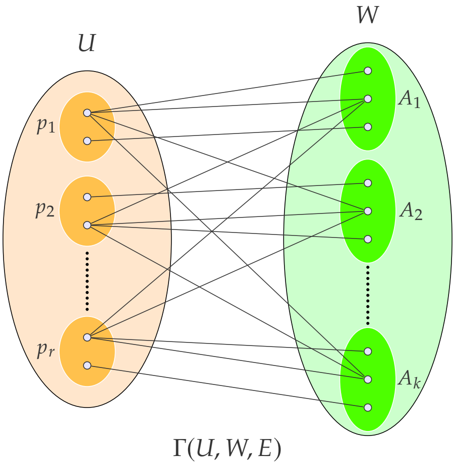
Parameters of the SumZero protocol [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: $1/2$



k-sum to Maxcover: Proof Sketch (Continued)

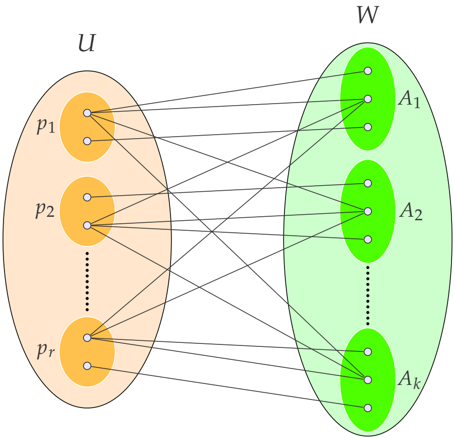
Parameters of the SumZero protocol [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: $1/2$



$\Gamma(U, W, E)$

Nodes in p_i are all $(z_1, \ldots, z_k) \in \mathbb{Z}_p^k$ such that $\sum_{j \in [k]} z_j = 0 \bmod p_i$

k-sum to Maxcover: Proof Sketch (Continued)

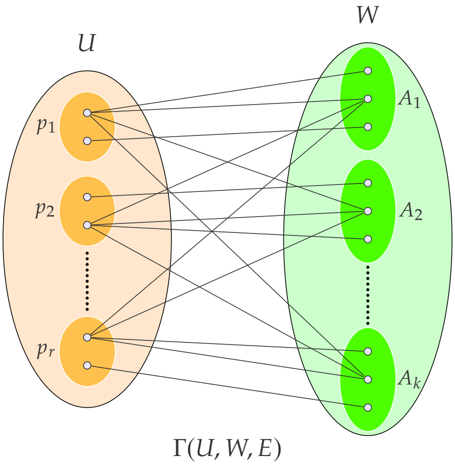
Parameters of the SumZero protocol [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: $1/2$



Nodes in p_i are all $(z_1, \ldots, z_k) \in \mathbb{Z}_p^k$ such that $\sum_{j \in [k]} z_j = 0 \bmod p_i$

For every $x \in A_j$ and $z = (z_1, \ldots, z_k) \in p_i$, $(x, z) \in E \iff z_j = x \bmod p_i$

k-sum to Maxcover: Proof Sketch (Continued)

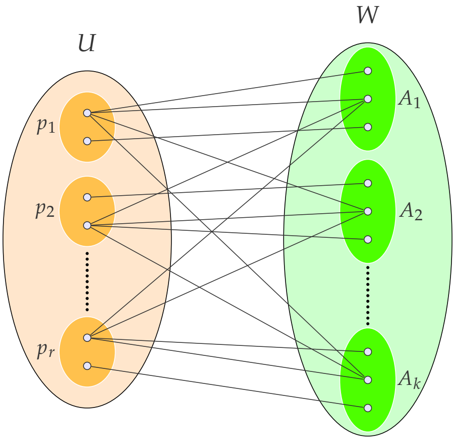
Parameters of the SumZero protocol [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: 1/2



$\Gamma(U, W, E)$

Nodes in p_i are all $(z_1, \ldots, z_k) \in \mathbb{Z}_p^k$ such that $\sum_{j \in [k]} z_j = 0 \bmod p_i$

For every $x \in A_j$ and $z = (z_1, \ldots, z_k) \in p_i$, $(x, z) \in E \iff z_j = x \bmod p_i$

A labeling $(x_1, \ldots, x_k)$ covers p_i

$\Updownarrow$

The referee accepts on random prime p_i

k-sum to Maxcover: Proof Sketch (Continued)

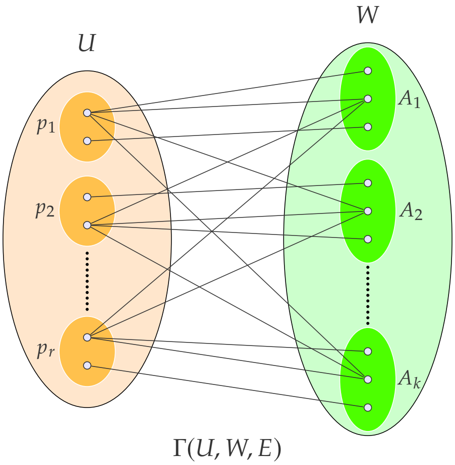
Parameters of the SumZero protocol [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: 1/2



Nodes in p_i are all $(z_1, \ldots, z_k) \in \mathbb{Z}_p^k$ such that $\sum_{j \in [k]} z_j = 0 \bmod p_i$

For every $x \in A_j$ and $z = (z_1, \ldots, z_k) \in p_i$, $(x, z) \in E \iff z_j = x \bmod p_i$

A labeling $(x_1, \ldots, x_k)$ covers p_i

$\Updownarrow$

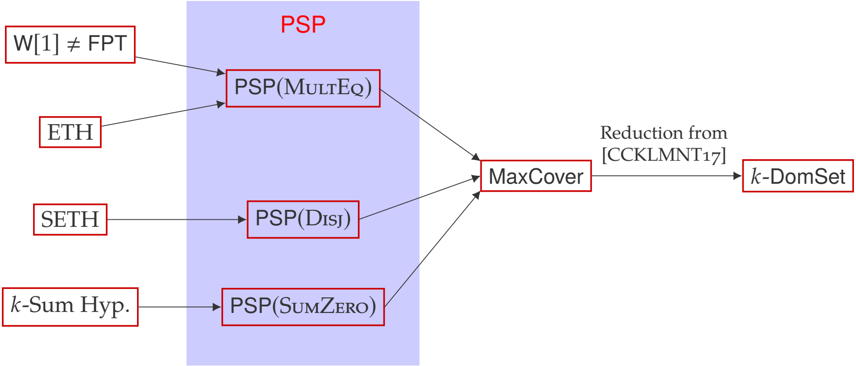
The referee accepts on random prime p_i

Soundness of SumZero protocol

$\Updownarrow$

Soundness of MaxCover

The Framework Revisited



Product Space Problems

Let $f : \{0,1\}^{m \times k} \to \{0,1\}$

Problem: $\mathsf{PSP}(f)$

Input: $A_1, \ldots A_k \subseteq \{0,1\}^m$ where $|A_i| \leq N$

Output: Determine if $\exists\, a_i \in A_i, \forall i \in [k]$, such that $f(a_1, \ldots, a_k) = 1$

Product Space Problems

Let $f : \{0,1\}^{m \times k} \to \{0,1\}$

Problem: PSP(f)

Input: $A_1, \ldots A_k \subseteq \{0,1\}^m$ where $|A_i| \leq N$

Output: Determine if $\exists\, a_i \in A_i, \forall i \in [k]$, such that $f(a_1, \ldots, a_k) = 1$

Product Space Problem (PSP)

Let $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be any function and $\mathcal{F}$ be a family of Boolean functions indexed by N and k as follows: $\mathcal{F} := \{f_{N,k} : \{0,1\}^{m(N,k) \times k} \to \{0,1\}\}_{N,k \in \mathbb{N}}$.

Product Space Problems

Let $f : \{0,1\}^{m \times k} \to \{0,1\}$

Problem: $\mathsf{PSP}(f)$

Input: $A_1, \ldots A_k \subseteq \{0,1\}^m$ where $|A_i| \leq N$

Output: Determine if $\exists\, a_i \in A_i, \forall i \in [k]$, such that $f(a_1, \ldots, a_k) = 1$

Product Space Problem (PSP)

Let $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be any function and $\mathcal{F}$ be a family of Boolean functions indexed by N and k as follows: $\mathcal{F} := \{f_{N,k} : \{0,1\}^{m(N,k) \times k} \to \{0,1\}\}_{N,k \in \mathbb{N}}$.

For each $k \in \mathbb{N}$, the *product space problem* $\mathsf{PSP}(k, \mathcal{F})$ of order N is defined as follows: given k subsets $A_1, \ldots, A_k$ of $\{0,1\}^{m(N,k)}$ each of cardinality at most N as input, determine if there exists $(a_1, \ldots, a_k) \in A_1 \times \cdots \times A_k$ such that $f_{N,k}(a_1, \ldots, a_k) = 1$.

Product Space Problems

Let $f : \{0,1\}^{m\times k} \to \{0,1\}$

Problem: $\mathsf{PSP}(f)$

Input: $A_1, \ldots A_k \subseteq \{0,1\}^m$ where $|A_i| \le N$

Output: Determine if $\exists\, a_i \in A_i, \forall i \in [k]$, such that $f(a_1, \ldots, a_k) = 1$

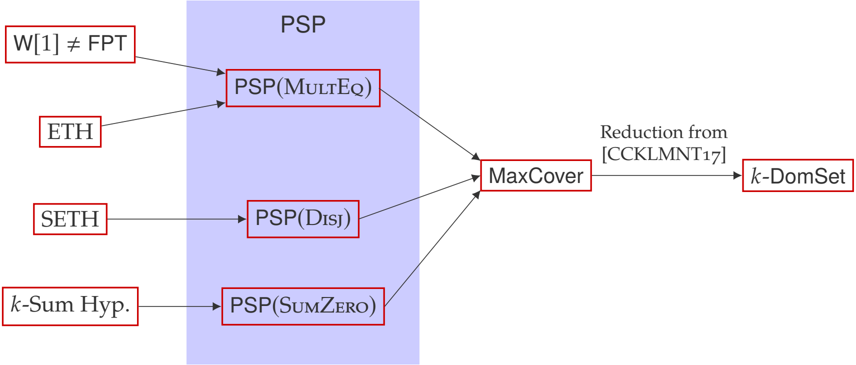
Product Space Problem (PSP)

Let $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be any function and $\mathcal{F}$ be a family of Boolean functions indexed by N and k as follows: $\mathcal{F} := \{f_{N,k} : \{0,1\}^{m(N,k)\times k} \to \{0,1\}\}_{N,k\in\mathbb{N}}$.

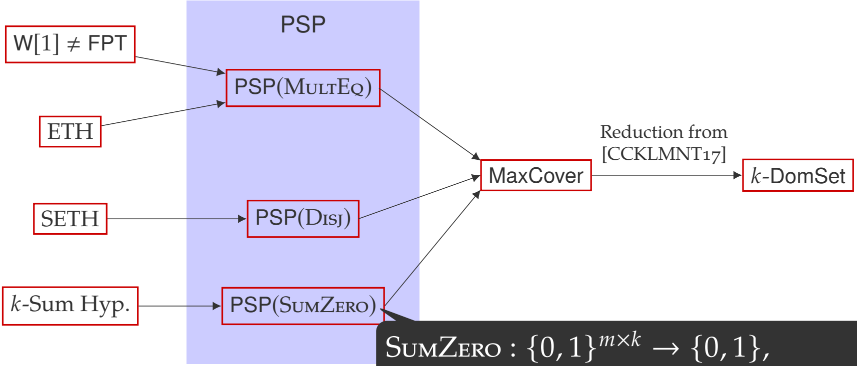
For each $k \in \mathbb{N}$, the *product space problem* $\mathsf{PSP}(k, \mathcal{F})$ of order N is defined as follows: given k subsets $A_1, \ldots, A_k$ of $\{0,1\}^{m(N,k)}$ each of cardinality at most N as input, determine if there exists $(a_1, \ldots, a_k) \in A_1 \times \cdots \times A_k$ such that $f_{N,k}(a_1, \ldots, a_k) = 1$.

For the rest of the talk, $m(N,k) = \mathrm{poly}(k) \cdot \log N$.

The Framework Revisited



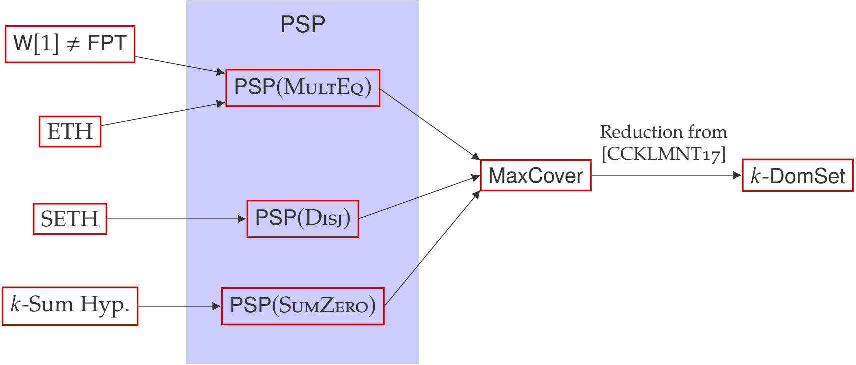
The Framework Revisited



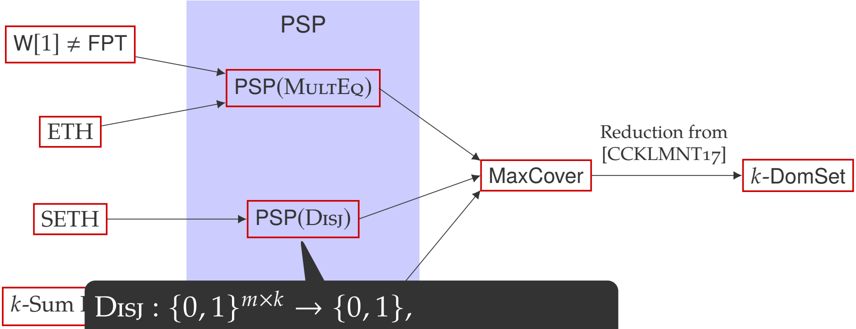
$$\textsc{SumZero} : \{0,1\}^{m\times k} \to \{0,1\},$$

$$\textsc{SumZero}(x_1,\ldots,x_k) = \begin{cases} 1 & \text{if } \sum_{i\in[k]} x_i = 0, \\ 0 & \text{otherwise.} \end{cases}$$

The Framework Revisited



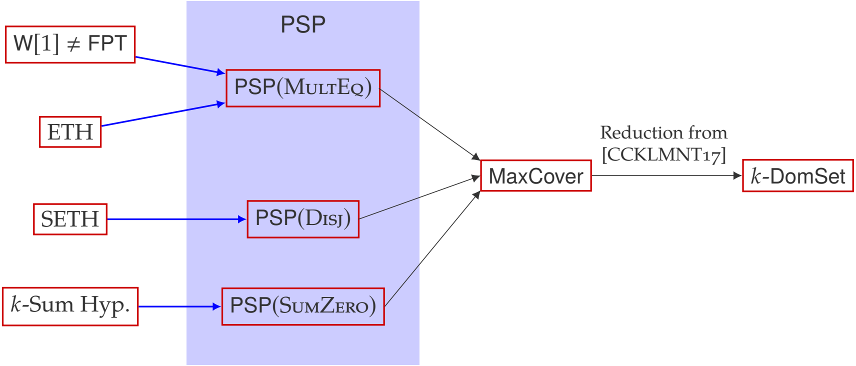
The Framework Revisited



$\textsc{Disj} : \{0,1\}^{m \times k} \to \{0,1\},$

$$\textsc{Disj}(x_1, \ldots, x_k) = \neg \left(\bigvee_{i \in [m]} \left(\bigwedge_{j \in [k]} (x_j)_i \right) \right).$$

The Framework Revisited



Popular Hypotheses to PSP

SETH $\implies$ PSP(Disj)

Popular Hypotheses to PSP

SETH $\implies$ PSP(Disj)

Let $X = X_1 \dot\cup \cdots \dot\cup X_k$

For every partial assignment σ to X_i, we build $a_\sigma \in A_i \subseteq \{0,1\}^m$ as follows:

$$a_\sigma(j) = \begin{cases} 0 & \text{if } \sigma \text{ satisfies } j^{\text{th}} \text{ clause} \\ 1 & \text{otherwise} \end{cases}$$

Popular Hypotheses to PSP

SETH $\implies$ PSP(Disj)

Let $X = X_1 \dot{\cup} \cdots \dot{\cup} X_k$

For every partial assignment σ to X_i, we build $a_\sigma \in A_i \subseteq \{0,1\}^m$ as follows:

$$a_\sigma(j) = \begin{cases} 0 & \text{if } \sigma \text{ satisfies } j^{\text{th}} \text{ clause} \\ 1 & \text{otherwise} \end{cases}$$

Note from above that ETH $\implies$ PSP(Disj). We will skip ETH $\implies$ PSP(MultEq)

Popular Hypotheses to PSP

SETH $\Longrightarrow$ PSP(Disj)

Let $X = X_1 \dot\cup \cdots \dot\cup X_k$

For every partial assignment σ to X_i, we build $a_\sigma \in A_i \subseteq \{0,1\}^m$ as follows:

$$a_\sigma(j) = \begin{cases} 0 & \text{if } \sigma \text{ satisfies } j^{\text{th}} \text{ clause} \\ 1 & \text{otherwise} \end{cases}$$

Note from above that ETH $\Longrightarrow$ PSP(Disj). We will skip ETH $\Longrightarrow$ PSP(MultEq)

W[1]$\neq$FPT $\Longrightarrow$ PSP(MultEq)

Popular Hypotheses to PSP

SETH $\implies$ PSP(Disj)

Let $X = X_1 \dot\cup \cdots \dot\cup X_k$

For every partial assignment σ to X_i, we build $a_\sigma \in A_i \subseteq \{0,1\}^m$ as follows:

$$a_\sigma(j) = \begin{cases} 0 & \text{if } \sigma \text{ satisfies } j^{\text{th}} \text{ clause} \\ 1 & \text{otherwise} \end{cases}$$

Note from above that ETH $\implies$ PSP(Disj). We will skip ETH $\implies$ PSP(MultEq)

W[1]≠FPT $\implies$ PSP(MultEq)

Starting point: ℓ-clique problem on graph $G(V, E)$

Popular Hypotheses to PSP

SETH $\Longrightarrow$ PSP(Disj)

Let $X = X_1 \dot\cup \cdots \dot\cup X_k$

For every partial assignment σ to X_i, we build $a_\sigma \in A_i \subseteq \{0,1\}^m$ as follows:

$$a_\sigma(j) = \begin{cases} 0 & \text{if } \sigma \text{ satisfies } j^{\text{th}} \text{ clause} \\ 1 & \text{otherwise} \end{cases}$$

Note from above that ETH $\Longrightarrow$ PSP(Disj). We will skip ETH $\Longrightarrow$ PSP(MultEq)

W[1]≠FPT $\Longrightarrow$ PSP(MultEq)

Starting point: ℓ-clique problem on graph $G(V,E)$

Let $k = \binom{\ell}{2}$ and set $A_i = E$, i.e., each edge $\in \left(\{0,1\}^{\log|V|} \times \{\perp, \top\}\right)^{\ell}$

Popular Hypotheses to PSP

SETH $\Longrightarrow$ PSP(Disj)

Let $X = X_1 \dot\cup \cdots \dot\cup X_k$

For every partial assignment σ to X_i, we build $a_\sigma \in A_i \subseteq \{0,1\}^m$ as follows:

$$a_\sigma(j) = \begin{cases} 0 & \text{if } \sigma \text{ satisfies } j^{\text{th}} \text{ clause} \\ 1 & \text{otherwise} \end{cases}$$

Note from above that ETH $\Longrightarrow$ PSP(Disj). We will skip ETH $\Longrightarrow$ PSP(MultEq)

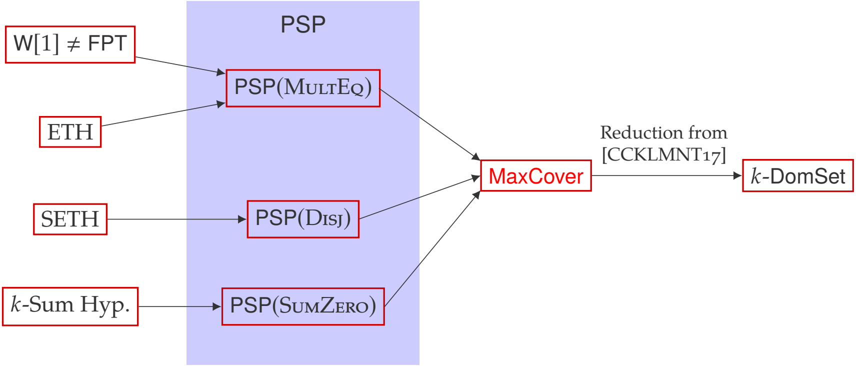
W[1]≠FPT $\Longrightarrow$ PSP(MultEq)

Starting point: ℓ-clique problem on graph $G(V, E)$

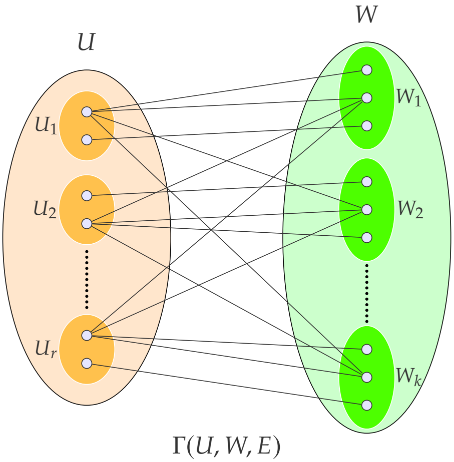
Let $k = \binom{\ell}{2}$ and set $A_i = E$, i.e., each edge $\in \left(\{0,1\}^{\log|V|} \times \{\perp, \top\}\right)^\ell$

Check for each vertex that the ℓ incident edges have assigned the same vertex (equality checking)

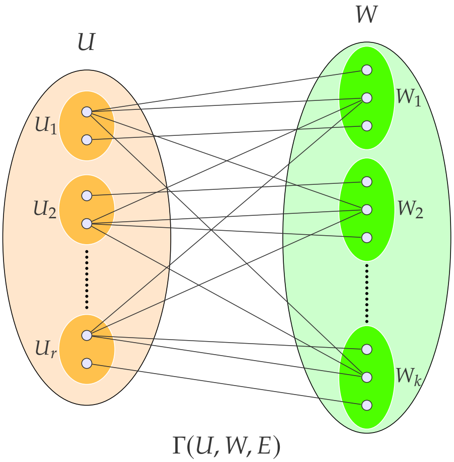
The Framework Revisited



Maxcover [CCKLMNT'17]



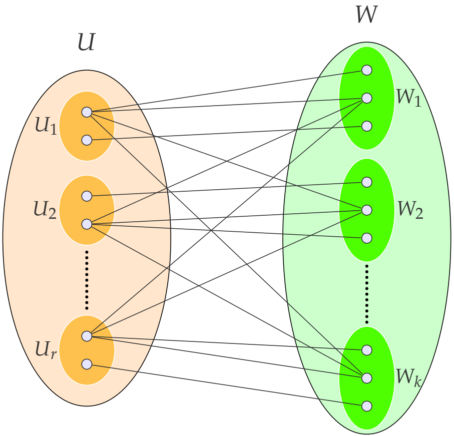
Maxcover [CCKLMNT'17]



Each W_i is a Right Super Node

Each U_i is a Left Super Node

Maxcover [CCKLMNT'17]



$\Gamma(U, W, E)$

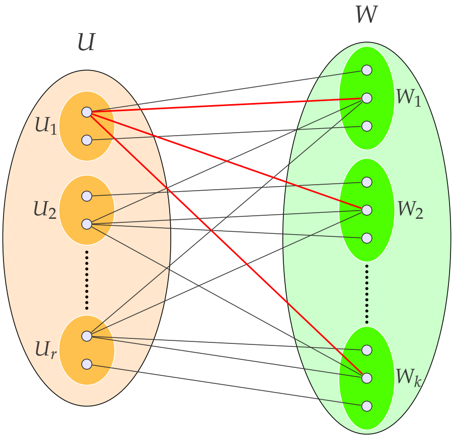
Each W_i is a Right Super Node

Each U_i is a Left Super Node

$S \subseteq W$ is a labeling of W if

$$\forall i \in [k], |S \cap W_i| = 1$$

Maxcover [CCKLMNT'17]



$\Gamma(U, W, E)$

Each W_i is a Right Super Node

Each U_i is a Left Super Node

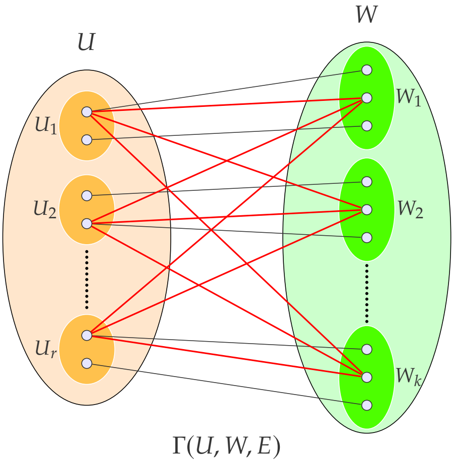
$S \subseteq W$ is a labeling of W if

$$\forall i \in [k], |S \cap W_i| = 1$$

S covers U_i if

$$\exists u \in U_i,\ \forall v \in S, (u, v) \in E$$

Maxcover [CCKLMNT'17]



Each W_i is a Right Super Node

Each U_i is a Left Super Node

$S \subseteq W$ is a labeling of W if

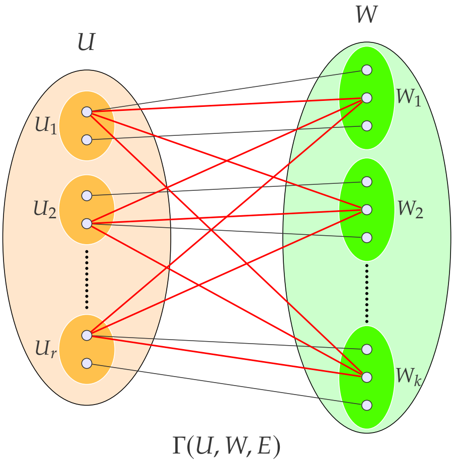
$$\forall i \in [k], |S \cap W_i| = 1$$

S covers U_i if

$$\exists u \in U_i,\ \forall v \in S, (u, v) \in E$$

MaxCover(Γ, S) = Fraction of U_i's covered by S

Maxcover [CCKLMNT'17]



Each W_i is a Right Super Node
Each U_i is a Left Super Node

$S \subseteq W$ is a labeling of W if

$$\forall i \in [k], |S \cap W_i| = 1$$

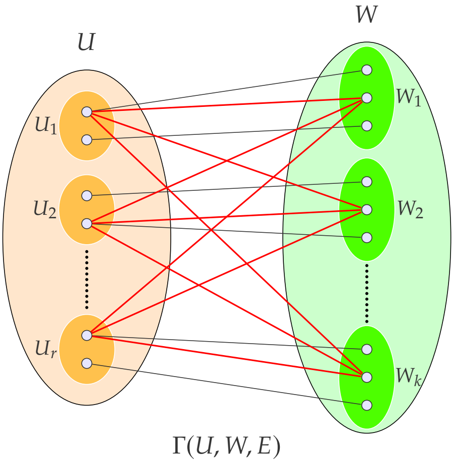
S covers U_i if

$$\exists u \in U_i,\ \forall v \in S, (u, v) \in E$$

MaxCover(Γ, S) = Fraction of U_i's covered by S

$$\text{MaxCover}(\Gamma) = \max_S \text{MaxCover}(\Gamma, S)$$

Maxcover [CCKLMNT'17]



Each W_i is a Right Super Node

Each U_i is a Left Super Node

$S \subseteq W$ is a labeling of W if

$$\forall i \in [k], |S \cap W_i| = 1$$

S covers U_i if

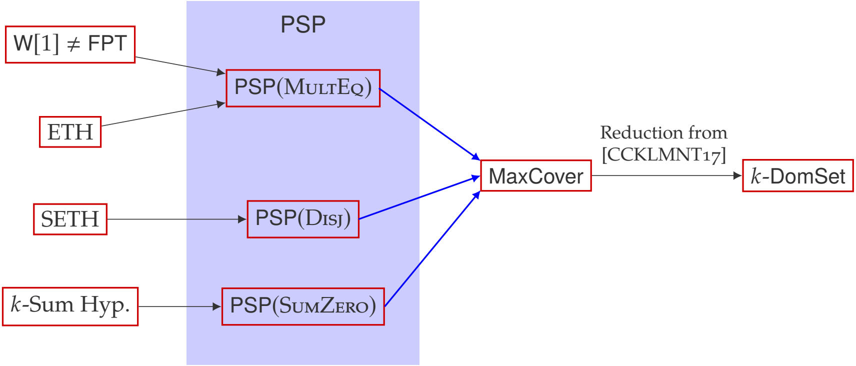
$$\exists u \in U_i, \ \forall v \in S, (u, v) \in E$$

MaxCover(Γ, S) = Fraction of U_i's covered by S

$$\text{MaxCover}(\Gamma) = \max_S \text{MaxCover}(\Gamma, S)$$

Determine if MaxCover$(\Gamma) = 1$ or MaxCover$(\Gamma) \leq s$

The Framework Revisited



PSP to Maxcover

Parameters of SMP protocol Π for $f : \{0,1\}^{m \times k} \to \{0,1\}$:

Advice: γ bits

Randomness: R bits

Message Length: L bits

Soundness: s

PSP to Maxcover

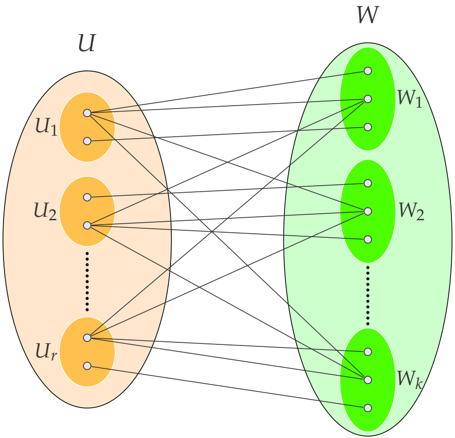
Parameters of SMP protocol Π for $f : \{0,1\}^{m \times k} \to \{0,1\}$:

Advice: γ bits

Randomness: R bits

Message Length: L bits

Soundness: s



$\Gamma(U, W, E)$

PSP to Maxcover

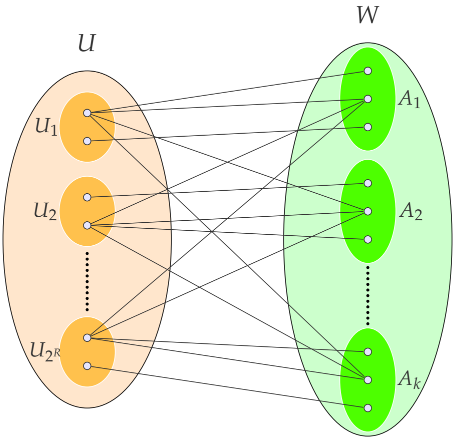
Parameters of SMP protocol Π for $f : \{0,1\}^{m\times k} \to \{0,1\}$:

Advice: γ bits

Randomness: R bits

Message Length: L bits

Soundness: s



$\Gamma(U, W, E)$

PSP to Maxcover

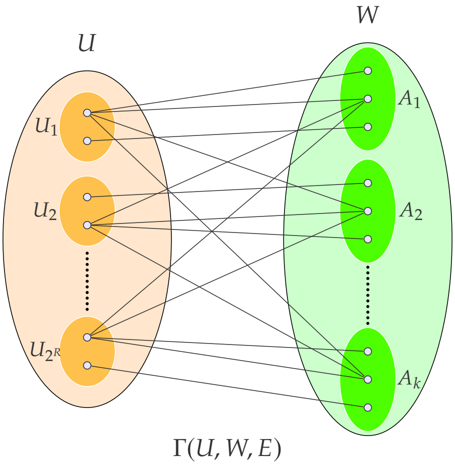
Parameters of SMP protocol Π for $f : \{0,1\}^{m\times k} \rightarrow \{0,1\}$:

Advice: γ bits

Randomness: R bits

Message Length: L bits

Soundness: s



2^{γ} instances of MaxCover

PSP to Maxcover

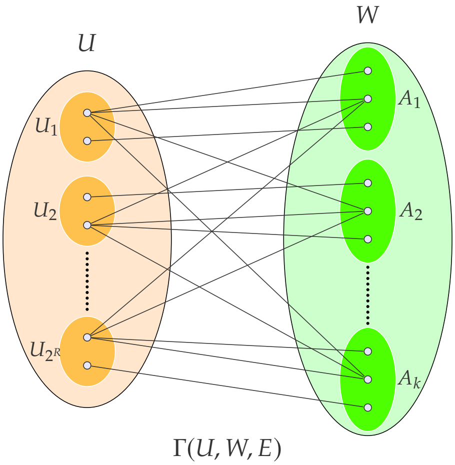
Parameters of SMP protocol Π for $f : \{0,1\}^{m \times k} \to \{0,1\}$:

Advice: γ bits

Randomness: R bits

Message Length: L bits

Soundness: s



2^γ instances of MaxCover

Nodes in U_i are all k-tuples of messages that referee accepts on randomness i and advice $\mu \in \{0,1\}^\gamma$

PSP to Maxcover

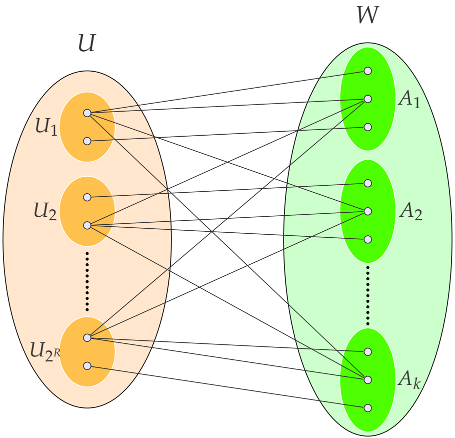
Parameters of SMP protocol Π for $f : \{0,1\}^{m \times k} \to \{0,1\}$:

Advice: γ bits

Randomness: R bits

Message Length: L bits

Soundness: s



$\Gamma(U, W, E)$

2^γ instances of MaxCover

Nodes in U_i are all k-tuples of messages that referee accepts on randomness i and advice $\mu \in \{0,1\}^\gamma$

For every $x \in A_j$ and $z = (z_1, \ldots, z_k) \in U_i$, $(x, z) \in E \iff z_j$ is message of player j on input x and randomness i

PSP to Maxcover

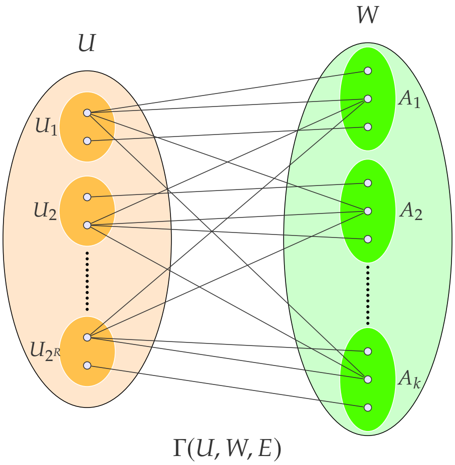
Parameters of SMP protocol Π for $f : \{0,1\}^{m \times k} \to \{0,1\}$:

Advice: γ bits

Randomness: R bits

Message Length: L bits

Soundness: s



2^γ instances of MaxCover

Nodes in U_i are all k-tuples of messages that referee accepts on randomness i and advice $\mu \in \{0,1\}^\gamma$

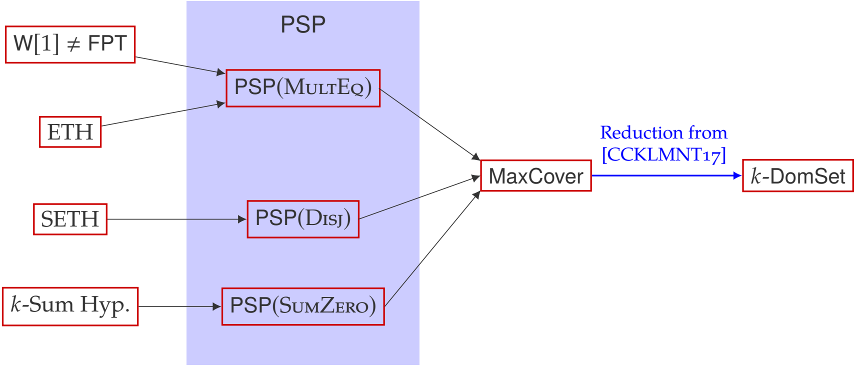
For every $x \in A_j$ and $z = (z_1, \ldots, z_k) \in U_i$, $(x, z) \in E \iff z_j$ is message of player j on input x and randomness i

Soundness of Π

$\Updownarrow$

Soundness of MaxCover

The Framework Revisited



Maxcover to Parameterized Dominating Set

Reduction from MaxCover to k-DomSet [CCKLMNT17]

There is a reduction from MaxCover instance $\Gamma = \left(U = \bigcup_{j=1}^{r} U_j, W = \bigcup_{j=1}^{k} W_i, E\right)$ to a k-DomSet instance G such that

Maxcover to Parameterized Dominating Set

Reduction from MaxCover to k-DomSet [CCKLMNT17]

There is a reduction from MaxCover instance $\Gamma = \left(U = \bigcup_{j=1}^{r} U_j, W = \bigcup_{j=1}^{k} W_i, E\right)$ to a k-DomSet instance G such that

- If $\mathsf{MaxCover}(\Gamma) = 1$, then $\mathsf{DomSet}(G) = k$
- If $\mathsf{MaxCover}(\Gamma) \leq \varepsilon$, then $\mathsf{DomSet}(G) \geq (1/\varepsilon)^{1/k} \cdot k$

Maxcover to Parameterized Dominating Set

Reduction from MaxCover to k-DomSet [CCKLMNT17]

There is a reduction from MaxCover instance $\Gamma = \left(U = \bigcup_{j=1}^{r} U_j, W = \bigcup_{j=1}^{k} W_i, E\right)$ to a k-DomSet instance G such that

- If $\text{MaxCover}(\Gamma) = 1$, then $\text{DomSet}(G) = k$
- If $\text{MaxCover}(\Gamma) \le \varepsilon$, then $\text{DomSet}(G) \ge (1/\varepsilon)^{1/k} \cdot k$
- $|V(G)| = |W| + \sum_{j \in [r]} k^{|U_j|}$

Maxcover to Parameterized Dominating Set

Reduction from MaxCover to k-DomSet [CCKLMNT17]

There is a reduction from MaxCover instance $\Gamma = \left(U = \bigcup_{j=1}^{r} U_j, W = \bigcup_{j=1}^{k} W_i, E\right)$ to a k-DomSet instance G such that

- If MaxCover$(\Gamma) = 1$, then DomSet$(G) = k$
- If MaxCover$(\Gamma) \leq \varepsilon$, then DomSet$(G) \geq (1/\varepsilon)^{1/k} \cdot k$
- $|V(G)| = |W| + \sum_{j\in[r]} k^{|U_j|}$
- The reduction runs in time $O\left(|W|\left(\sum_{j\in[r]} k^{|U_j|}\right)\right)$.

Maxcover to Parameterized Dominating Set

Reduction from MaxCover to k-DomSet [CCKLMNT17]

There is a reduction from MaxCover instance $\Gamma = \left(U = \bigcup_{j=1}^{r} U_j, W = \bigcup_{j=1}^{k} W_i, E\right)$ to a k-DomSet instance G such that

- If $\text{MaxCover}(\Gamma) = 1$, then $\text{DomSet}(G) = k$
- If $\text{MaxCover}(\Gamma) \leq \varepsilon$, then $\text{DomSet}(G) \geq (1/\varepsilon)^{1/k} \cdot k$
- $|V(G)| = |W| + \sum_{j \in [r]} k^{|U_j|}$
- The reduction runs in time $O\left(|W| \left(\sum_{j \in [r]} k^{|U_j|}\right)\right)$.

We want $1/\varepsilon = \omega(1)$ and $|U_j| = o(m)$

Required Parameters of SMP Protocols

Greedily we want SMP protocols:

Input: m bits

Randomness: polylog(m) bits

Message Length: $O_k(1)$ bits

Soundness: $1/2$

SMP Protocol for k-sumZero

SMP Protocol of Nisan [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: $1/2$

SMP Protocol for k-sumZero

SMP Protocol of Nisan [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: $1/2$

SMP Protocol of Viola [Viola'15]:

Input: m bits

Randomness: $O(m)$ bits

Message Length: $O_k(1)$ bits

Soundness: $1/2$

SMP Protocol for k-sumZero

SMP Protocol of Nisan [Nisan'94]:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(\log m)$ bits

Soundness: 1/2

SMP Protocol of Viola [Viola'15]:

Input: m bits

Randomness: $O(m)$ bits

Message Length: $O_k(1)$ bits

Soundness: 1/2

New SMP Protocol:

Input: m bits

Randomness: $O_k(\log m)$ bits

Message Length: $O_k(1)$ bits

Soundness: 1/2

SMP Protocol for k-multiequality

Idea: Use any binary code of constant rate and distance δ

SMP Protocol for k-multiequality

Idea: Use any binary code of constant rate and distance δ

SMP Protocol Parameters:

Input: m bits

Randomness: $O(\log m)$ bits

Message Length: $O(1)$ bits

Soundness: $1 - \delta$

SMP Protocol for k-disjointness

A straightforward extension of Rubinstein's two-party protocol [R'18,ARW'17,AW'09]

SMP Protocol for k-disjointness

A straightforward extension of Rubinstein's two-party protocol [R'18,ARW'17,AW'09]

Good Pointwise Product (GPP) Codes

Let q be a prime power and $k \in \mathbb{N}$. A code C over $\mathbb{F}_q$ is said to be a q-GHP code if there exists a constant $\delta(k) > 0$ such that the following holds.

- C is systematic and can be encoded efficiently.
- Let C^k be the set of all k-pointwise product of codewords of C. Then, there exists a linear good code $\widetilde{C}$ such that $C^k \subseteq \widetilde{C}$, i.e., $\widetilde{C}$ has relative distance and rate greater than δ.

SMP Protocol for k-disjointness

A straightforward extension of Rubinstein's two-party protocol [R'18,ARW'17,AW'09]

Good Pointwise Product (GPP) Codes

Let q be a prime power and $k \in \mathbb{N}$. A code C over $\mathbb{F}_q$ is said to be a q-GHP code if there exists a constant $\delta(k) > 0$ such that the following holds.

- C is systematic and can be encoded efficiently.
- Let C^k be the set of all k-pointwise product of codewords of C. Then, there exists a linear good code $\widetilde{C}$ such that $C^k \subseteq \widetilde{C}$, i.e., $\widetilde{C}$ has relative distance and rate greater than δ.

- Player i divides his input x_i into T parts $x_i^1, \ldots x_i^T$

SMP Protocol for k-disjointness

A straightforward extension of Rubinstein's two-party protocol [R'18,ARW'17,AW'09]

Good Pointwise Product (GPP) Codes

Let q be a prime power and $k \in \mathbb{N}$. A code C over $\mathbb{F}_q$ is said to be a q-GHP code if there exists a constant $\delta(k) > 0$ such that the following holds.

- C is systematic and can be encoded efficiently.
- Let C^k be the set of all k-pointwise product of codewords of C. Then, there exists a linear good code $\widetilde{C}$ such that $C^k \subseteq \widetilde{C}$, i.e., $\widetilde{C}$ has relative distance and rate greater than δ.

- Player i divides his input x_i into T parts $x_i^1, \ldots x_i^T$
- The advice μ of the referee is $\sum_{j \in [T]} \prod_{\ell \in [k]} C(x_\ell^j)$ – a codeword of $\widetilde{C}$!

SMP Protocol for k-disjointness

A straightforward extension of Rubinstein's two-party protocol [R'18,ARW'17,AW'09]

Good Pointwise Product (GPP) Codes

Let q be a prime power and $k \in \mathbb{N}$. A code C over $\mathbb{F}_q$ is said to be a q-GHP code if there exists a constant $\delta(k) > 0$ such that the following holds.

- C is systematic and can be encoded efficiently.
- Let C^k be the set of all k-pointwise product of codewords of C. Then, there exists a linear good code $\widetilde{C}$ such that $C^k \subseteq \widetilde{C}$, i.e., $\widetilde{C}$ has relative distance and rate greater than δ.

- Player i divides his input x_i into T parts $x_i^1, \ldots x_i^T$
- The advice μ of the referee is $\sum_{j \in [T]} \prod_{\ell \in [k]} C(x_\ell^j)$ – a codeword of $\widetilde{C}$!
- Referee checks that μ is zero in the systematic part and on a random coordinate

SMP Protocol for k-disjointness

A straightforward extension of Rubinstein's two-party protocol [R'18,ARW'17,AW'09]

Good Pointwise Product (GPP) Codes

Let q be a prime power and $k \in \mathbb{N}$. A code C over $\mathbb{F}_q$ is said to be a q-GHP code if there exists a constant $\delta(k) > 0$ such that the following holds.

- C is systematic and can be encoded efficiently.
- Let C^k be the set of all k-pointwise product of codewords of C. Then, there exists a linear good code $\widetilde{C}$ such that $C^k \subseteq \widetilde{C}$, i.e., $\widetilde{C}$ has relative distance and rate greater than δ.

- Player i divides his input x_i into T parts $x_i^1, \ldots x_i^T$
- The advice μ of the referee is $\sum_{j \in [T]} \prod_{\ell \in [k]} C(x_\ell^j)$ – a codeword of $\widetilde{C}$!
- Referee checks that μ is zero in the systematic part and on a random coordinate

Advice: $O_k(m/T \log q)$ bits

Randomness: $O_k(\log m)$ bits

Message Length: $T \log q$ bits

Soundness: $1 - \delta$

SMP Protocol for k-disjointness (continued)

SMP Protocol Parameters:

Advice: $O_k(m/T \log q)$ bits

Message Length: $T \log q$ bits

Randomness: $O_k(\log m)$ bits

Soundness: $1 - \delta$

SMP Protocol for k-disjointness (continued)

SMP Protocol Parameters:

Advice: $O_k(m/T \log q)$ bits

Message Length: $T \log q$ bits

Randomness: $O_k(\log m)$ bits

Soundness: $1 - \delta$

Reed Solomon Codes

Let $\ell \in \mathbb{N}$ and q be a prime number in $[4\ell, 8\ell)$. Then, there exists a q-GPP code of message length ℓ.

SMP Protocol for k-disjointness (continued)

SMP Protocol Parameters:

Advice: $O_k(m/T \log q)$ bits

Message Length: $T \log q$ bits

Randomness: $O_k(\log m)$ bits

Soundness: $1 - \delta$

Reed Solomon Codes

Let $\ell \in \mathbb{N}$ and q be a prime number in $[4\ell, 8\ell)$. Then, there exists a q-GPP code of message length ℓ.

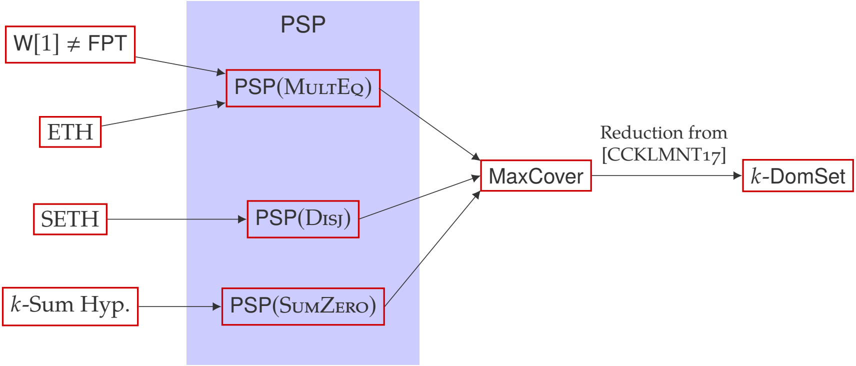
Algebraic Geometric Codes [GS'96, SAKSD'01]

There exists a constant $c \in \mathbb{N}$ such that for any prime number q greater than c there is a q^2-GPP code for every message length $\ell \in \mathbb{N}$.

Recap of the Results

- Any $T(k)$ approximation is W[1]-hard
- No $T(k)$ approximation algorithm in $N^{o(k)}$ time, assuming ETH
- No $T(k)$ approximation algorithm in $N^{k-\varepsilon}$ time, assuming SETH
- No $T(k)$ approximation algorithm in $N^{\lceil k/2 \rceil - \varepsilon}$ time, assuming k-SUM Hypothesis

Summary of the Framework



Important Open Questions

- Parameterized Dominating Set is W[2]-complete. Can we show every $T(k)$ approximation is also W[2]-hard?

Important Open Questions

- Parameterized Dominating Set is W[2]-complete. Can we show every $T(k)$ approximation is also W[2]-hard?
- Parameterized Clique is W[1]-complete. Can we show every $T(k)$ approximation is also W[1]-hard?

Important Open Questions

- Parameterized Dominating Set is W[2]-complete. Can we show every $T(k)$ approximation is also W[2]-hard?
- Parameterized Clique is W[1]-complete. Can we show every $T(k)$ approximation is also W[1]-hard? Can we show 1.01 approximation is W[1]-hard?

Open Questions

- Are there natural problems in PSP which do not have efficient MA protocols?

Open Questions

- Are there natural problems in PSP which do not have efficient MA protocols?
- Conceptually/Philosophically can we say something about the various time hypotheses?

THANK
YOU!

The Framework