NP-Hardness of Learning Programs and Partial MCSP

Shuichi Hirahara

National Institute of Informatics, Japan

@Oxford-Warwick complexity meeting

August 25, 2022

Outline

1. History of MCSP

2. MINLT and Learning Programs

3. Proof Techniques

The Cook-Levin Theorem

The Cook-Levin Theorem [Cook **1971**, Levin **1973**]

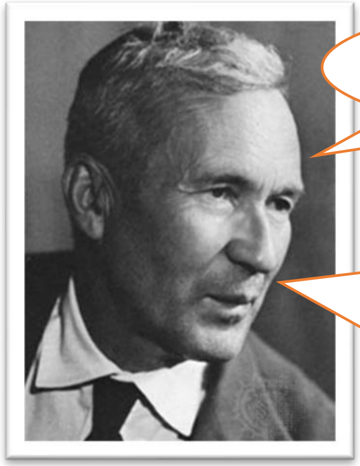
SAT is NP-complete.

- One of the most fundamental theorems in complexity theory
- Independently proved by Cook (in the Western Bloc) and by Levin (in Soviet Union) during the cold war.

Stephen Cook

Leonid Levin

In the early 1970s...



Andrey Kolmogorov

Very interesting!

Leonid,
PUBLISH IT!

Leonid Levin

Some problems
are
NP-complete!

But these
problems are of
narrow interest.

The results would not be worth publishing
unless **more popular problems** can be
shown to be NP-complete.

For example:

1. Graph isomorphism
2. Factoring

Unlikely to be
NP-complete.

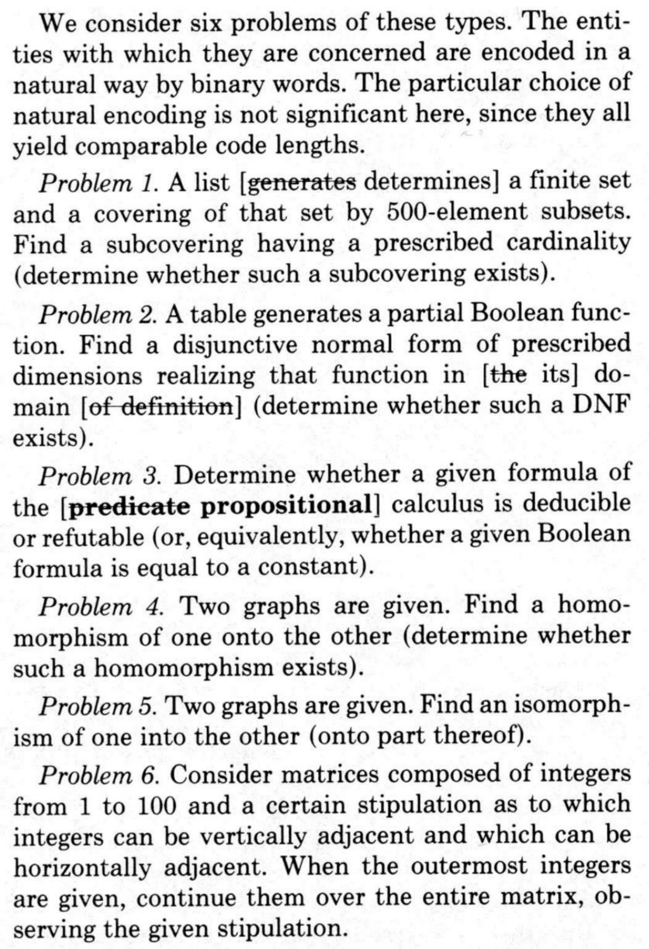
3. The Minimum Circuit Size Problem

   (MCSP, named by [Kabanets & Cai'00])

It is still open to prove NP-completeness!

Reference: https://www.cs.bu.edu/fac/lnd/research/hard.htm

Levin’s 1973 paper (submitted in 1972)



We consider six problems of these types. The entities with which they are concerned are encoded in a natural way by binary words. The particular choice of natural encoding is not significant here, since they all yield comparable code lengths.

Problem 1. A list [~~generates~~ determines] a finite set and a covering of that set by 500-element subsets. Find a subcovering having a prescribed cardinality (determine whether such a subcovering exists).

Problem 2. A table generates a partial Boolean function. Find a disjunctive normal form of prescribed dimensions realizing that function in [~~the~~ its] domain [~~of definition~~] (determine whether such a DNF exists).

Problem 3. Determine whether a given formula of the [~~**predicate**~~ **propositional**] calculus is deducible or refutable (or, equivalently, whether a given Boolean formula is equal to a constant).

Problem 4. Two graphs are given. Find a homomorphism of one onto the other (determine whether such a homomorphism exists).

Problem 5. Two graphs are given. Find an isomorphism of one into the other (onto part thereof).

Problem 6. Consider matrices composed of integers from 1 to 100 and a certain stipulation as to which integers can be vertically adjacent and which can be horizontally adjacent. When the outermost integers are given, continue them over the entire matrix, observing the given stipulation.

- ➢ Levin presented six NP-complete problems.

Problem 1: the Set Cover problem

Problem 2: DNF-MCSP*

(The partial variant of DNF-MCSP)

Problem 3: SAT

Problem 5: The Subgraph Isomorphism Problem

This work: NP-completeness of MCSP*

Has been open over the last 50 years!

The English translation from [Trakhtetenbrot’84].

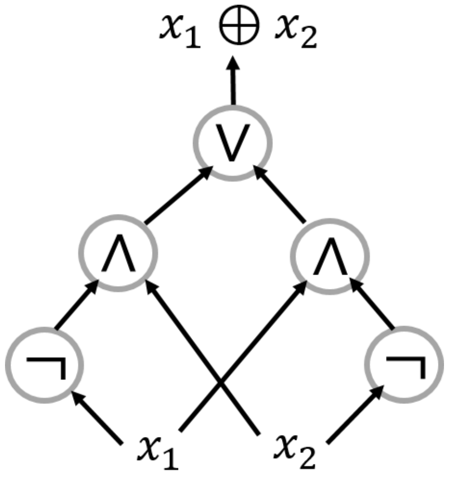
MCSP (The Minimum Circuit Size Problem [Kabanets & Cai'00])

Input

- The truth table of a function
  $$f: \{0,1\}^n \to \{0,1\}$$
  (encoded as a string of length 2^n)
- A size parameter $s \in \mathbb{N}$

Output

Is there a circuit of size $\leq s$ that computes f?



Example $\mathrm{truthtable}(\oplus_2) = 0110$

x_1	x_2	$x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

$\mathrm{size}(\oplus_2) = 3$

Here, we count the number of ∨ and ∧ gates.

(We may consider other measures of "circuit size".)

Fact: $\mathrm{MCSP} \in \mathrm{NP}$

Open: NP-hardness of MCSP

MCSP^* (Partial MCSP)

Input

- The truth table of a partial function
  $f\colon \{0,1\}^n \to \{0,1,*\}$
  ($f(x) = *$ indicates "I don't care")
- A size parameter $s \in \mathbb{N}$

Output

Is there a circuit of size $\le s$ that outputs $f(x)$ on input $x \in f^{-1}(\{0,1\})$?

Example $\mathrm{truthtable}(f) = 0 * 1 *$

x_1	x_2	$f(x_1, x_2)$
0	0	0
0	1	$*$
1	0	1
1	1	$*$

$\mathrm{size}(f) = 0$

A circuit $C(x_1, x_2) := x_1$ computes $f(x_1, x_2)$ on input $(0,0)$ and $(1,0)$.

Fact: $\mathrm{MCSP}^* \in \mathrm{NP}$

Main Theorem 1

MCSP^* is NP-hard under randomized poly-time reductions.

Minimum DNF Size Problem (DNF-MCSP)

Input

- The truth table of a Boolean function $f: \{0,1\}^n \to \{0,1\}$ (encoded as a string of length 2^n)
- A size parameter $s \in \mathbb{N}$

Output

Is there a DNF formula of size $\leq s$ that computes f?

$$x_1 \oplus x_2 = (x_1 \land \neg x_2) \lor (\neg x_1 \land x_2)$$

Example $\mathrm{truthtable}(\oplus_2) = 0110$

$$\mathrm{DNFsize}(\oplus_2) = 4$$

Theorem [Masek'79]: DNF-MCSP is NP-complete.

Theorem [H.-Oliveira-Santhanam'18]: $(\mathrm{DNF} \circ \mathrm{XOR})$-MCSP is NP-complete.

Theorem [Ilango'20]: AC^0 formula-MCSP is NP-complete.
MCSP^* is hard under Exponential Time Hypothesis.

Theorem [Ilango'21]: Formula-MCSP is hard under Exponential Time Hypothesis.

MCSP versus MCSP*

[Allender, Hellerstein, McCabe, Pitassi, Saks'08]

$\mathrm{NP} \quad \le_m^p \quad \text{DNF-MCSP}^* \quad \le_m^p \quad \text{DNF-MCSP}$

[H.-Oliveira-Santhanam'18]

$\mathrm{NP} \quad \le_m^p \quad \text{DNF} \circ \text{XOR-MCSP}^* \quad \le_m^p \quad \text{DNF} \circ \text{XOR-MCSP}$

[Ilango'20] [Ilango'21]

$\mathrm{SAT} \quad \le_m^{exp} \quad \text{Formula-MCSP}^* \quad \le_T^{exp} \quad \text{Formula-MCSP}$

This work

$\mathrm{NP} \quad \le_m^{\mathrm{BPP}} \quad \mathrm{MCSP}^* \quad \overset{?}{\le} \quad \mathrm{MCSP}$

(open)

The Main Theorem

- In fact, initially I didn't try to prove NP-hardness of MCSP^*.
- Our starting point was NP-hardness of MINLT [Ko'91], which asks the Kolmogorov complexity of a partial function (succinctly encoded).

Main Theorem

MINLT, MINKT^*, MKTP^*, MCSP^*, $\mathrm{NC}^1\text{-}\mathrm{MCSP}^*$ are all NP-hard via a single reduction!

(listed in the order of the difficulty of the proofs)

- Previously, no non-trivial reduction among them was known. (even $\mathrm{NC}^1\text{-}\mathrm{MCSP}^* \leq_m^{\mathrm{BPP}} \mathrm{MCSP}^*$ was unknown.)
- NP-hardness of MCSP^* has nothing to do with Kolmogorov complexity, but it plays an important role in the proof.

MINKT^* (The partial variant of MINKT [Ko'91])

Input

- A partial string $x \in \{0,1,*\}^n$
- A size parameter $s \in \mathbb{N}$
- A time parameter $t \in \mathbb{N}$ (in unary)

Output

Is there a t-time program of size s that prints $y \in \{0,1\}^n$ consistent with x?

Example: $0 * 11 *$ is consistent with 00111
but not consistent with 10110

In terms of Kolmogorov complexity:

- Kolmogorov complexity $\mathrm{K}(y) := \min\{\, |M| : M \text{ prints } y \,\}$.
- t-time-bounded Kolmogorov complexity $\mathrm{K}^t(y) := \min\{\, |M| : M \text{ prints } y \text{ in } t \text{ steps} \,\}$.
- For $x \in \{0,1,*\}^n$, $\mathrm{K}^{*,t}(x) := \min\{\, \mathrm{K}^t(y) : y \in \{0,1\}^n \text{ is consistent with } x \,\}$.

Informally, MINKT^* is the problem of computing $\mathrm{K}^{*,t}(-)$.

Outline

1. History of MCSP

2. MINLT and Learning Programs

3. Proof Techniques

PAC Learning and Occam Learning

- The task of learning is parameterized by
  - a concept class $\mathcal{C}$ and
  - a hypothesis class $\mathcal{H}$.

E.g., $\mathcal{C} = \{\text{linear-size circuits}\}, \mathcal{H} = \{\text{poly-size circuits}\}$.

- Occam learning of $\mathcal{C}$ by $\mathcal{H}$ [Blumer, Ehrenfeucht, Haussler, Warmuth'87]

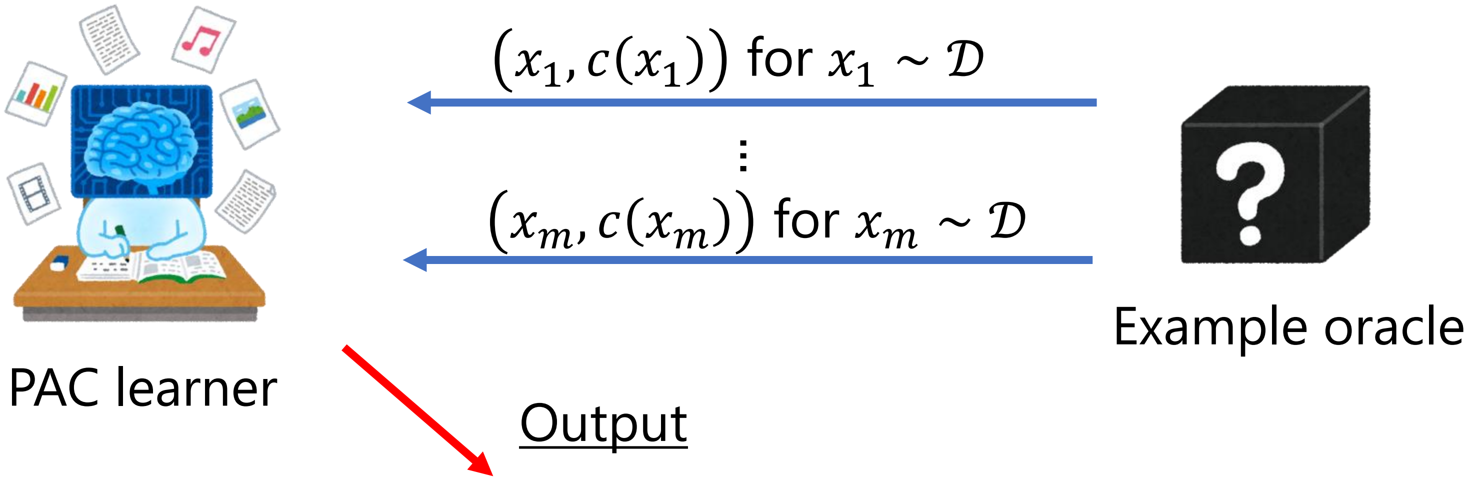
  $\in \mathrm{FNP}$ (an NP search problem)

  Given $(x_1, c(x_1)), \ldots, (x_m, c(x_m))$ as input for some unknown concept $c \in \mathcal{C}$, output a hypothesis $h \in \mathcal{H}$ such that $h(x_i) = c(x_i)$ for every i.

- Occam learning of $\mathcal{C}$ by $\mathcal{H}$ is equivalent to PAC learning of $\mathcal{C}$ by $\mathcal{H}$ (for a sufficiently large $\mathcal{H}$).

  [Blumer, Ehrenfeucht, Haussler, Warmuth'87] [Board & Pitt'92] [Schapire'90]

PAC Learning of $\mathcal{C}$ by $\mathcal{H}$



An unknown concept $c \in \mathcal{C}$
An arbitrary distribution $\mathcal{D}$

Input $\delta, \epsilon > 0$

$(x_1, c(x_1))$ for $x_1 \sim \mathcal{D}$

$\vdots$

$(x_m, c(x_m))$ for $x_m \sim \mathcal{D}$

Example oracle

PAC learner

Output

$h \in \mathcal{H}$ such that $\Pr_{x \sim \mathcal{D}}[c(x) = h(x)] \geq 1 - \epsilon$
(with probability $1 - \delta$)

Is improper learning "NP-complete"?

Big Open Problem: Is PAC learning of linear-size circuits by poly-size circuits as hard as NP?

Known Results

[Pitt & Valiant ‘88]

PAC learning of k-term DNFs by k-term DNFs is NP-hard.

[Alekhnovich, Braverman, Feldman, Klivans, Pitassi ‘08]

PAC learning of linear-size DNFs by poly-size $\mathrm{OR} \circ \{\mathrm{halfspaces}\}$ is NP-hard.

Open: NP-hardness of linear-size NC^1 by poly-size NC^1 circuits

➢ In general, as a hypothesis class $\mathcal{H}$ becomes larger,
it becomes more difficult to prove NP-hardness.

Ko’s Question: $\mathcal{H} = \{\text{programs}\}$

Ker-I Ko
(1950-2018)

[Ko’91]

- Consider the “largest” hypothesis class

  $\mathcal{H}$ = {efficient programs}.

E.g., a circuit can be simulated by a program.

More generally, a program can represent a function most succinctly.
(by the fundamental principle of Kolmogorov complexity)

Ko’s Question: Can we prove NP-hardness of Occam learning of $\mathcal{H}$ by $\mathcal{H}$?

MINLT [Ko'91]

- The decision version of Occam learning for efficient programs

Input

- Samples $(x_1, b_1), \ldots, (x_m, b_m) \in \{0,1\}^n \times \{0,1\}$
- A time parameter 1^t
- A size parameter 1^s

Output

Is there a t-time program M of size s such that $M(x_i) = b_i$ for every i?

"The complexity of MINLT appears very difficult to classify precisely." [Ko'91]

Theorem [Ko'91]: No relativizing proof for NP-hardness of MINLT exists.

NP-hardness of MINLT

- We overcome Ko's relativization barrier!

Theorem 1 (NP-hardness of the decision version of PAC learning for programs)

It is NP-hard to solve the following promise problem:

Input: a distribution $\mathcal{D}$, a size parameter $s \in \mathbb{N}$

Yes: there exists a poly-time program M of size s such that

$$\Pr_{(x,b)\sim\mathcal{D}}[M(x) = b] = 1$$

No: for any time-unbounded program M of size $s \cdot n^{1/\log^{O(1)} \log n}$,

$$\Pr_{(x,b)\sim\mathcal{D}}[M(x) = b] \leq \frac{1}{2} + 2^{-n^{0.99}}$$

- In particular, MINLT is also NP-hard.

NP-hardness of MINLT

- We overcome Ko's relativization barrier!

Theorem 1 (NP-hardness of the decision version of PAC learning for programs)

It is NP-hard to solve the following promise pro

Input: a distribution $\mathcal{D}$, a size parameter $s \in \mathbb{N}$

Yes: there exists a poly-time program M of size s such that

$$\Pr_{(x,b)\sim\mathcal{D}}[M(x) = b] = 1$$

In fact, M computes a linear function.

No: for any time-unbounded program M of size $s \cdot n^{1/\log^{O(1)}\log n}$,

$$\Pr_{(x,b)\sim\mathcal{D}}[M(x) = b] \leq \frac{1}{2} + 2^{-n^{0.99}}$$

If this is improved to $1.01n$, then Heuristica doesn't exist! [H. & Nanashima'21]

- In particular, MINLT is also NP-hard.

NP-hardness of MCSP^*

- By optimizing the reduction of Theorem 1, we get:

Theorem 2

It is NP-hard to solve the following promise problem:

Input: a partial function $f\colon \{0,1\}^n \to \{0,1,*\}$, a size parameter $s \in \mathbb{N}$

Yes: ∃ a program M of size s and ∃ an NC^1 circuit C of size $\frac{s}{\log s}$ such that

$$\Pr_{x\sim D}[M(x) = f(x)] = 1 \quad \& \Pr_{x\sim D}[C(x) = f(x)] = 1$$

No: ∀ program M of size $s \cdot n^{0.01}$ and ∀ circuit C of size $\frac{s}{\log s} \cdot n^{0.01}$,

$$\Pr_{x\sim D}[M(x) = f(x)] \leq \frac{1}{2} + n^{-0.01} \quad \& \Pr_{x\sim D}[C(x) = f(x)] \leq \frac{1}{2} + n^{0.01}$$

$D \coloneqq$ the uniform distribution over $f^{-1}(\{0,1\})$

NP-hardness of MCSP*

- By optimizing the reduction of Theorem 1, we get:

Theorem 2

$s = 2^{\Theta(n)}$.

It is NP-hard to solve the following promise problem:

Input: a partial function $f\colon \{0,1\}^n \to \{0,1,*\}$, a size parameter $s \in \mathbb{N}$

Yes: ∃ f size s and ∃ an NC^1 circuit C of size $\frac{s}{\log s}$ such th

$f(x)] = 1 \quad \& \Pr_{x\sim D}[C(x) = f(x)] = 1$

Follows from this program l.b.

Exponential circuit lower bounds!

No: ∀ program M of size $s \cdot n^{0.01}$ and ∀ circuit C of size $\frac{s}{\log s} \cdot n^{0.01}$,

$$\Pr_{x\sim D}[M(x) = f(x)] \le \frac{1}{2} + n^{-0.01} \quad \& \Pr_{x\sim D}[C(x) = f(x)] \le \frac{1}{2} + n^{0.01}$$

$D :=$ the uniform distribution over $f^{-1}(\{0,1\})$

Program lower bound $\Longrightarrow$ Circuit lower bound

- $f: \{0,1\}^n \to \{0,1\}$, a function

- Suppose that there is no program of size s that can compute f.

Claim: There is no circuit of size $\Omega\left(\frac{s}{\log s}\right)$ that computes f.

Proof:

- Assume that there is a circuit C of size s' that computes f.
- Since C can be simulated by a program,
  we may construct a program of size $O(s' \log s')$ that computes f.
- Therefore, $s \leq O(s' \log s')$. ■

Outline

1. History of MCSP

2. MINLT and Learning Programs

3. Proof Techniques

Proof Techniques

Pseudorandomness, PCP theorems, Cryptography.

Both results use Kolmogorov complexity in a fundamental way.

- NP-hardness of MINLT
  - A reduction from Minimum Monotone Satisfying Assignment
  - Secret sharing scheme
  - Use a pseudorandom generator construction as a one-time encryption scheme.
- NP-hardness of MCSP^*
  - PCP theorem (for Sliding Scale Conjecture)
  - Nisan-Wigderson pseudorandom generator construction
  - Impagliazzo-Wigderson derandomized XOR lemma
  - Uhlig's theorem

Minimum Monotone Satisfying Assignment Problem (MMSA)

Input

- A monotone formula φ
- A threshold parameter $\theta \in \mathbb{N}$

Output

Is there a satisfying assignment $\alpha \in \{0,1\}^n$ for φ with Hamming weight θ?

Example: $\varphi = (x_1 \vee x_2) \wedge (x_1 \vee x_3)$

$\alpha =$ "100" $\in \{0,1\}^3$ satisfies φ

➢ NP-hard to approximate within a factor of $n^{1/\log^{O(1)} \log n}$

[Dinur & Safra'04] [Dinur, Harsha & Kindler'15]

Secret Sharing Scheme [Shamir’79, Blakley’79]

- Any monotone formula φ admits a secret sharing scheme.

  [Ito, Saito, and Nishizeki’93] [Benaloh and Leichter’88]

  - A set $T \subseteq [n]$ of parties is authorized if $\varphi(\chi_T) = 1$.

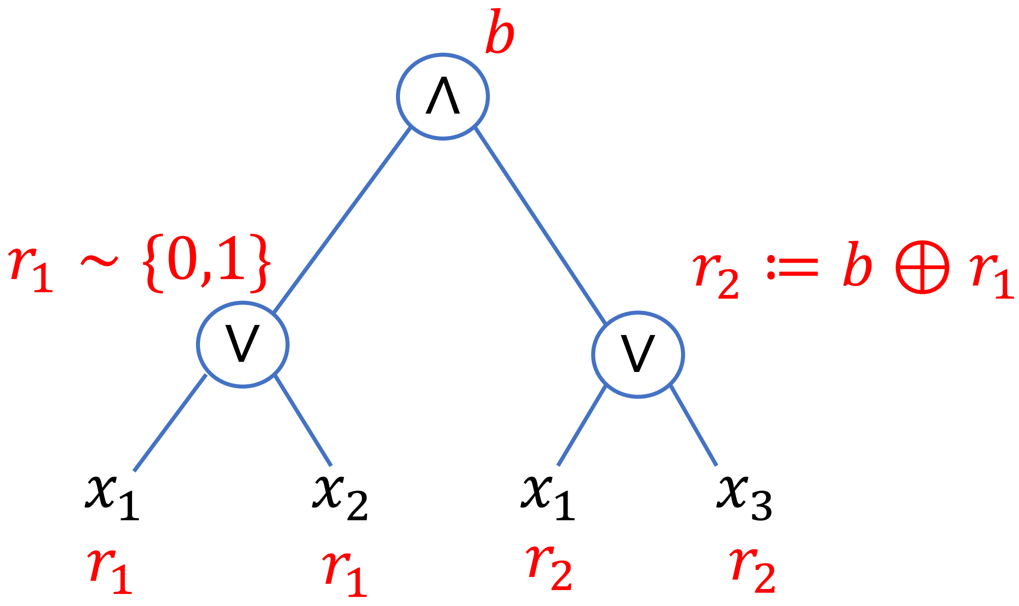
  - A secret sharing scheme shares a secret $b \sim \{0,1\}$ among n parties so that

    (Correctness) any authorized set of parties can reconstruct b, but

    (Privacy) no unauthorized set has no information about b.

Secret Sharing Scheme (continued)

Example: $\varphi = (x_1 \vee x_2) \wedge (x_1 \vee x_3)$



$x_1's$ share $s_1 := (r_1, r_2)$
$x_2's$ share $s_2 := r_1$
$x_3's$ share $s_3 := r_2$

- 3 parties: x_1, x_2, x_3
- A secret $b \sim \{0,1\}$
- Authorized sets $\{x_1\}, \{x_2, x_3\}$ can reconstruct b.
- $\{x_2\}$ has no information about b ($s_2 = r_1$ is independent of b).

Any monotone formula φ admits a secret sharing scheme.

- A set $T \subseteq [n]$ of parties is authorized if $\varphi(\chi_T) = 1$.
- A secret sharing scheme shares a secret $b \sim \{0,1\}$ among n parties so that
  (Correctness) any authorized set can reconstruct b, but
  (Privacy) no unauthorized set has no information about b.

The reduction from MMSA to MINLT

- Let φ be a (depth-3) monotone formula on n variables.
- Choose $f_1, \ldots, f_n \sim \{0,1\}^\lambda$ (using the randomness of a randomized reduction).
- Define a distribution $\mathcal{E} = \mathcal{E}(f_1, .., f_n)$ as follows.
  - Choose a secret $b \sim \{0,1\}$.
  - Share b among n parties.
    Let $s_1, \ldots, s_n$ be the shares given to the i-th party.
  - Define $x := (z_1, G(f_1; z_1) \oplus s_1, \ldots, z_n, G(f_n; z_n) \oplus s_n)$ for $z_i \sim \{0,1\}^*$
  - Output (x, b).

Hide the share s_i in x so that only a program that knows f_i can read s_i.

$G(f; z^1, \ldots, z^k) := \langle z^1, f \rangle \ldots \langle z^k, f \rangle \in \{0,1\}^k$, a pseudorandom generator construction

Completeness of the reduction

- Suppose φ is satisfiable by assignment α with Hamming weight $\leq \theta$.

- Then, the set $T \coloneqq \{i | \alpha_i = 1\}$ is authorized.

- Consider the following program M:
  - Hard-wired input: $\{f_i | i \in T\}$.
  - Input: $x = (z_1, \xi_1, \dots, z_n, \xi_n)$
  - Let $s_i \coloneqq \xi_i \oplus G(f_i; z_i)$ for each $i \in T$.
  - Reconstruct b by using $\{s_i | i \in T\}$.
  - Output $b \in \{0,1\}$.

- The size of M is $\sum_{i \in T} |f_i| = |T| \cdot \lambda \leq \theta\lambda$ and $\Pr_{(x,b)\sim\mathcal{E}}[M(x) = b] = 1$.

Distribution $\mathcal{E} = \mathcal{E}(f_1, .., f_n)$

- Choose a secret $b \sim \{0,1\}$.
- Share b among n parties.
  Let $s_1, \dots, s_n$ be the shares given to the i-th party.
- Define $x \coloneqq (z_1, G(f_1; z_1) \oplus s_1, \dots, z_n, G(f_n; z_n) \oplus s_n)$
- Output (x, b).

Soundness of the reduction (1/3)

Claim: If no assignment α of weight 2θ can satisfy φ, then for every program M of size $\theta\lambda$,

$$\Pr_{(x,b)\sim\mathcal{E}}[M(x) = b] \leq \frac{1}{2} + o(1).$$

- $\mathrm{DP}_k(f;z) := \big(z, G(f;z)\big) = (z, \langle z_1, f\rangle \dots \langle z_k, f\rangle)$ is known to be pseudorandom against any algorithm M such that $\mathrm{K}(f\,|M) \gg k$. [H.'20]

$\mathrm{DP}_k\colon \{0,1\}^\lambda \times \big(\{0,1\}^\lambda\big)^k \to \{0,1\}^{\lambda k+k}$, a k-wise direct product generator

If $\mathrm{K}(f\,|M) > k + O(\log n)$, then

$$\left|\Pr_z\Big[M\big(z, \boldsymbol{G(f;z)}\big) = 1\Big] - \Pr_{\substack{z,\\ w\sim\{0,1\}^k}}[M(z, \boldsymbol{w}) = 1]\right| \leq o(1).$$

Soundness of the reduction (2/3)

Claim: If no assignment α of weight 2θ can satisfy φ, then for every program M of size $\theta\lambda$, $\Pr_{(x,b)\sim\mathcal{E}}[M(x) = b] \le \frac{1}{2} + o(1)$.

- Idea: We want to formalize that M "knows" f_i (Is f_i hard-wired in M?).
- M *knows* $f_i \overset{\text{def}}{\Longleftrightarrow} \mathrm{K}(f_i|M) \ll |f_i| = \lambda$.

  (Equivalently, the mutual information $\mathrm{I}(f_i : M) = \mathrm{K}(f_i) - \mathrm{K}(f_i|M)$ is large.)
- Let $B \coloneqq \{i | M \text{ knows } f_i\}$.

Key Lemma: "Algorithmic Information Extraction Lemma"

$$|B| \le \left(1 + o(1)\right) \cdot \frac{|M|}{\lambda} \text{ and } \Pr_z[M(\mathrm{DP}_k(f_i; z)) = 1] \approx \Pr_w[M(w) = 1] \text{ for every } i \notin B.$$

- $|B| \le 1.1 \cdot \frac{|M|}{\lambda} \le 1.1 \cdot \theta < 2\theta$.
- B is not authorized, so the secret cannot be reconstructed from $\{s_i | i \in B\}$.

Soundness of the reduction (3/3)

Example: Assume $B = \{2\}$ and $n = 2$.

M does not know f_1, but M knows f_2.

$$\Pr_{(x,b)\sim\mathcal{E}}[M(x) = b] = \Pr[M(\mathrm{DP}_k(f_1; z_1) \oplus s_1, \mathrm{DP}_k(f_2; z_2) \oplus s_2) = b]$$

$$\approx \Pr_{w_1\sim\{0,1\}^*}[M(\qquad w_1 \qquad, \mathrm{DP}_k(f_2; z_2) \oplus s_2) = b]$$

Because M does not know f_1

$$= \frac{1}{2}$$

Because B is not authorized.
(s_2 does not reveal any information about b) ■

Extension to MCSP^*

- A distribution $\mathcal{E}$ defines a partial function $f\colon\{0,1\}^n \to \{0,1,*\}$, where
$$f(x) \coloneqq \begin{cases} b, & \text{if } (x,b) \in \mathrm{supp}(\mathcal{E}) \\ *, & \text{otherwise} \end{cases}$$

- The reduction to MINLT produces $f\colon\{0,1\}^{n^{O(1)}} \to \{0,1,*\}$, given φ of size n.

- The Ideas for NP-hardness of $\mathrm{MINKT}^*, \mathrm{MKTP}^*, \mathrm{MCSP}^*$:

Reduce the input length of f to $O(\log n)$.

(Then, the truth table of $f\colon\{0,1\}^{O(\log n)} \to \{0,1,*\}$ is of polynomial length.)

NP-hardness of MINKT^*

The locality of PCP theorems + Nisan-Wigderson generator

- We start from a PCP system of [Dinur, Fischer, Kindler, Raz, Safra'11] (to obtain a large inapproximability factor).

- Instead of MMSA, we consider a *collection* of monotone formulas
$$\{\varphi_1, \dots, \varphi_n\}$$
over $[n]$ variables, where each φ_j "checks" the consistency of proofs.

- Locality of PCP: $\Delta \coloneqq |\varphi_j| \ll \log n$.

- The distribution $\mathcal{E} = \mathcal{E}(f_1, \dots, f_n)$ (that outputs (x, b)) is defined as follows:
  - Let $j \sim [n]$. Share a secret $b \sim \{0,1\}$ among Δ parties using φ_j.
  - $x \coloneqq \left(j, z, \mathrm{NW}\left(\mathrm{Enc}(f_1); z_{S_1}\right) \oplus s_1, \dots, \mathrm{NW}\left(\mathrm{Enc}(f_\Delta); z_{S_\Delta}\right) \oplus s_\Delta\right)$

NW: Nisan-Wigderson generator, Enc: error-correcting code.

NP-hardness of MKTP^*

Impagliazzo-Wigderson’s derandomized hardness amplification theorem

- A program has $\{f_i | i \in T\}$ as hard-wired input for an authorized set T.
- In order to compute $\mathrm{NW}(\mathrm{Enc}(f_i); z_{S_i})$ from f_i, one needs to read almost all bits of $f_i \in \{0,1\}^\lambda$.
- The time complexity is $\gg \lambda$, whereas MKTP^* asks sublinear-time-bounded Kolmogorov complexity.
- We use the hardness amplification theorem of [Impagliazzo-Wigderson’97], which provides a locally-encodable list-decodable error-correcting code.

NP-hardness of MCSP^*

Uhlig's theorem

- We must hard-wire $\{f_i | i \in T\}$ in a circuit using at most $O\left(\frac{\lambda}{\log \lambda} \cdot |T|\right)$ gates.
- Moreover, the circuit must be able to compute $\mathrm{NW}\left(\mathrm{Enc}(f_i); z_{S_i}\right)$.
- Observation: $\mathrm{Enc}(f)$ is computable by a f-oracle nonadaptive circuit.

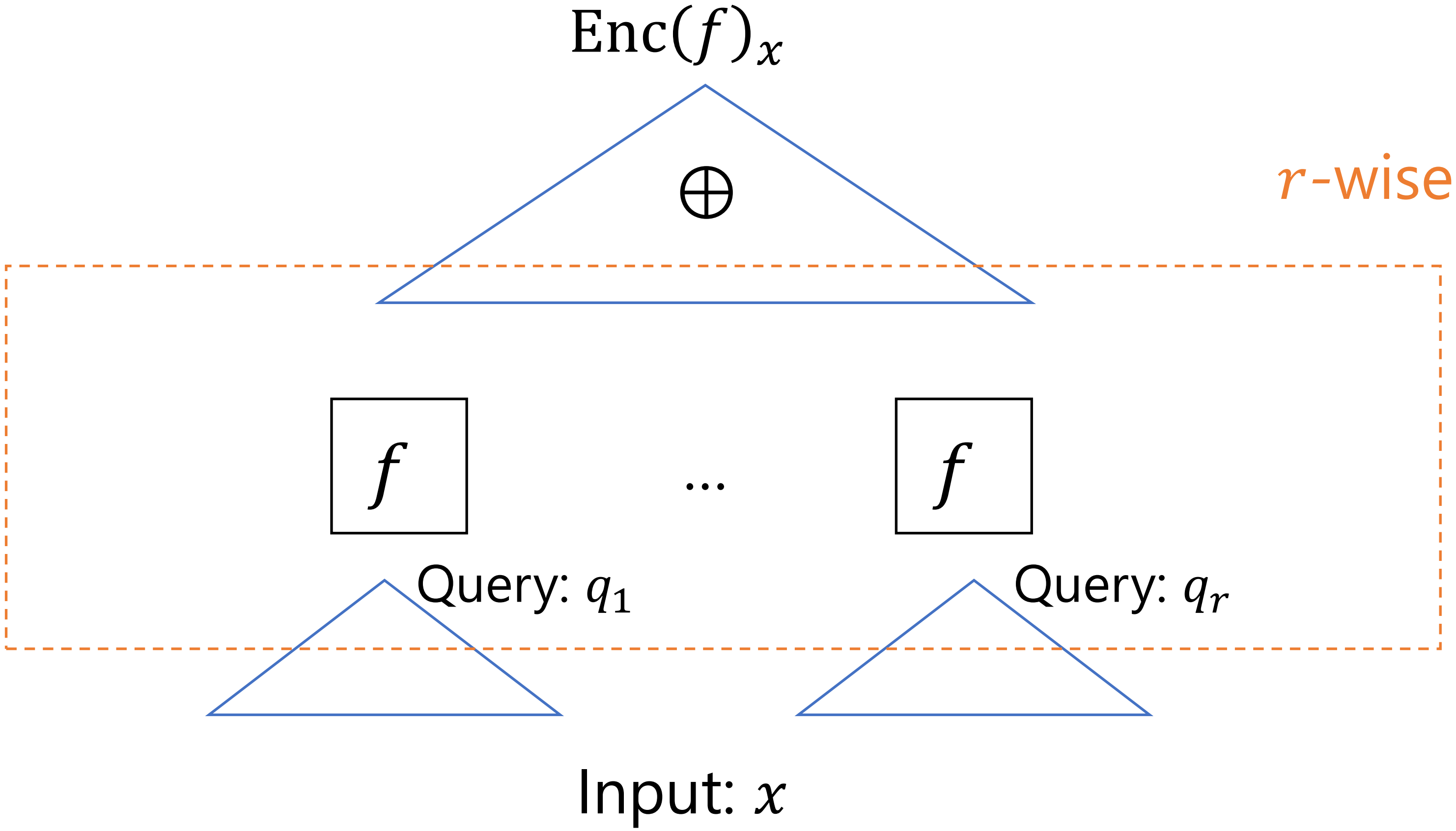
Theorem [Uhlig'74, Uhlig'92]

For any $f : \{0,1\}^{\log \lambda} \to \{0,1\}$, the r-wise direct product

$$f^r(x_1, \dots, x_r) = \left(f(x_1), \dots, f(x_r)\right)$$

can be computed by a circuit of size $O\left(\frac{\lambda}{\log \lambda}\right)$ if $r = \lambda^{o(1/\log\log\lambda)}$.

A circuit computing $\mathrm{Enc}(f)$



r-wise direct product of f

Circuit size: $O\left(\frac{\lambda}{\log \lambda}\right)$

Open Problems

- ➢ Can we prove $\mathrm{MCSP}^* \leq_m^{\mathrm{BPP}} \mathrm{MCSP}$?
  - A simple idea: Replace $f(x) = *$ with $f(x) \sim \{0,1\}$.
  - This proves NP-hardness of $\mathrm{AveMCSP}$ (the average-case version of MCSP).

- ➢ Can we exclude Heuristica? ($\mathrm{P} = \mathrm{NP}$ iff NP is easy on average?)
  - Requires *non-black-box* and *non-relativizing* proof techniques. [Bogdanov-Trevisan'06], [Impagalizzo'11], [H.-Nanashima'21]
  - $\mathrm{GapMINKT} \in \mathrm{P}$ if $\mathrm{DistNP} \subseteq \mathrm{AvgP}$ [H.'18] non-black-box, relativizing
  - $\mathrm{NP} \leq_m^{\mathrm{BPP}} \mathrm{GapMINKT}^*$ [This Work] black-box, non-relativizing
  - We need to simultaneously overcome the two barriers.
  - It suffices to prove $\mathrm{GapMINKT}^* \leq_m^{\mathrm{BPP}} \mathrm{GapMINKT}$.