Partial Minimum Branching Program Size Problem is ETH-Hard

Ludmila Glinskih

Based on joint work with Artur Riazanov (EPFL)

Meta-Complexity Reunion
Simons Institute for the Theory of Computing
April 15, 2024

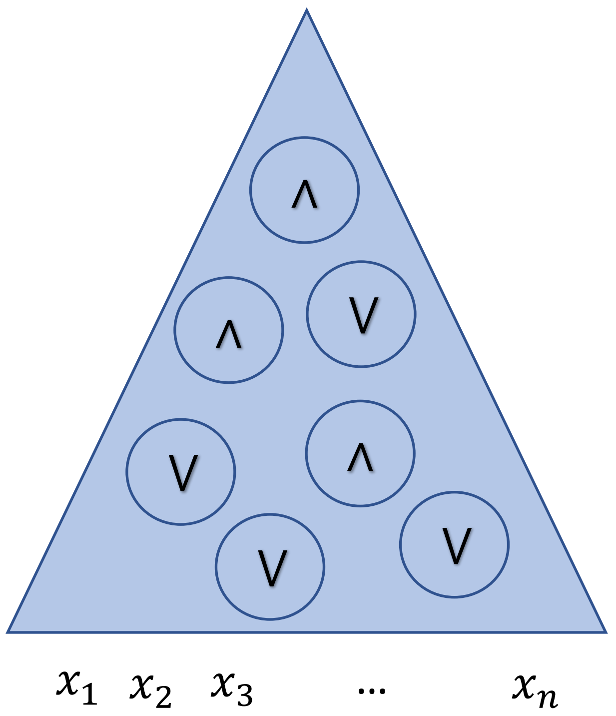
Minimum Circuit Size Problem

Input:

- truth table of a Boolean function $f: \{0,1\}^n \to \{0,1\}$
- size parameter s

1	0	0	1	0	1	1	0	…	1

Truth table of f of length $N = 2^n$



Output:

yes, if f can be computed by a circuit of size at most s

no, otherwise

Minimum *C*-**Circuit** Size Problem

What other circuit types we may consider?

1	0	0	1	0	1	1	0	…	1

Truth table of f of length $N = 2^n$

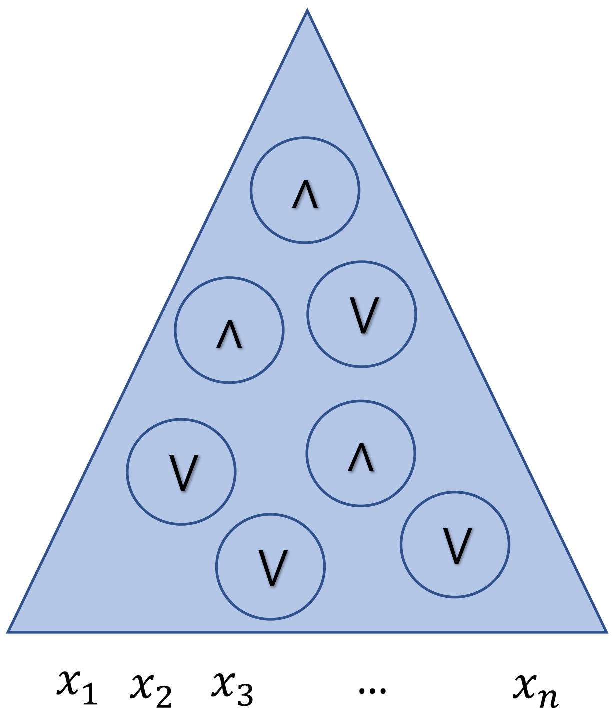
Input:

- truth table of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$
- size parameter s

Output:

yes, if f can be computed by a ***C*-circuit** of size at most s

no, otherwise



Hardness of *C*-MCSP for various circuit classes

NP-hardness is known for *C*=

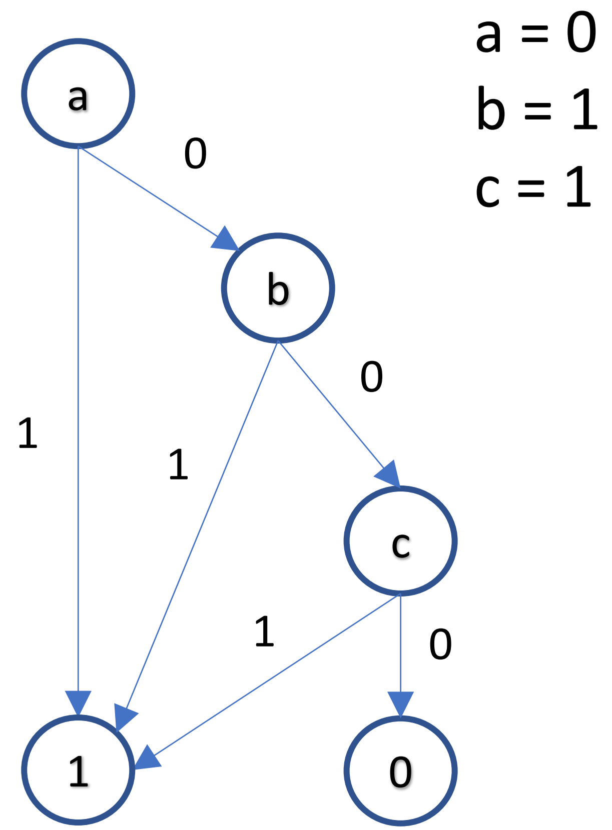
- DNF [Masek'80]
- DNF $\circ$ XOR [Hirahara, Oliveira, Santhanam'19]
- AC0 formula [Ilango'20]

ETH-hardness is known for DeMorgan Boolean formulas [Ilango'20]

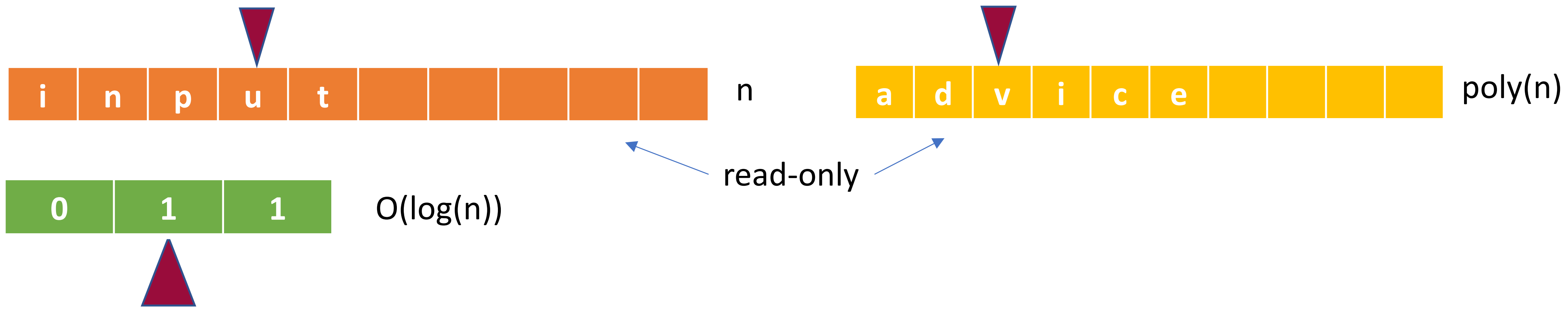
This work: what happens if we consider *C* as **Branching Programs**?

Branching program

- Branching Programs (BPs) represent Boolean functions:
  - directed graph without cycles
  - one source
  - two sinks: labeled with 0 and 1
  - all other vertices labeled with variables
  - values of variables on edges
- Size of a BP is the number of vertices
- k-BP: on every path every variable occurs no more than k time
- Oblivious 1-BP (OBDD): an ordered version of 1-BP



Complexity class with logarithmic space



BP(f)=poly $\Leftrightarrow$ f is in L/poly

LBs on the size of BP representation imply space-complexity LBs

Branching program minimization

Minimization of OBDD and 1-BP is **NP**-hard

- Given f as an OBDD find an equivalent OBDD of size s [Bollig, Wegener'96]
- Given f as a 1-BP find a 1-BP of size s [Sieling'02]
- Given a set of pairs $(x_1, f(x_1)), \dots, (x_t, f(x_t))$
  - Find an OBDD of size s consistent with f given an order of variables [Takenaga, Yajima'00]
  - Approximate the min size of a 1-BP consistent with f [Sieling'02]

There exists a $O(3^n poly(n))$-time algorithm for OBDD-MCSP [Friedman, Supowit'88]

input length $N = 2^n$

Hardness of branching program minimization

Theorem: assuming Exponential Time Hypothesis every Turing machine computing Partial Minimum Branching Program Size Problem requires time $N^{\Omega(\log\log N)}$

holds also for minimizing
1-BP, k-BP, OBDD

Partial minimization problems

Minimizing the size of

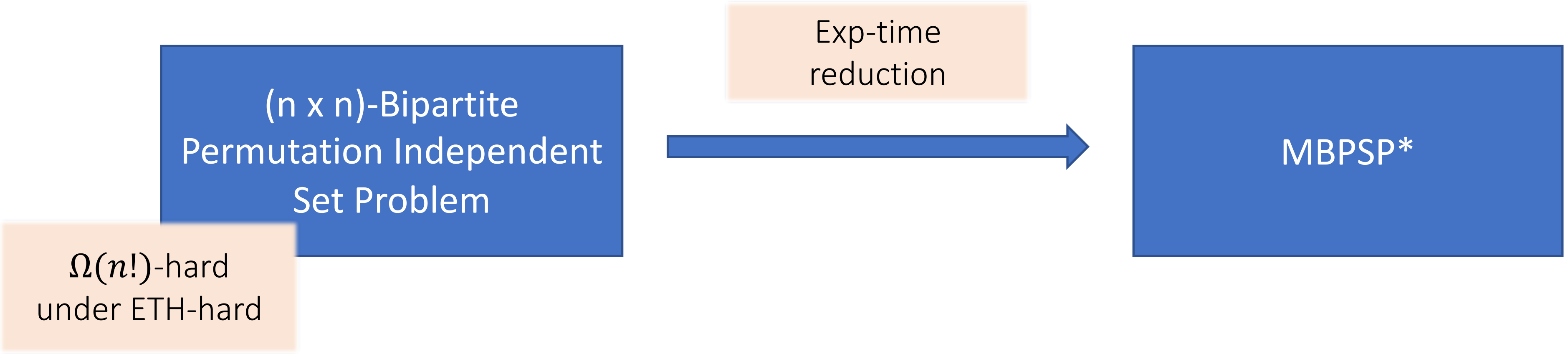
- DNF is NP-hard [Macek'79]
- DeMorgan Formula is ETH-hard [Ilango'21]
- Partial MBPSP is ETH-hard **[this work]**
- Partial MCSP is ETH-hard [Ilango'20]
- Partial MCSP is NP-hard under randomized reductions [Hirahara'22]

First shown for partial

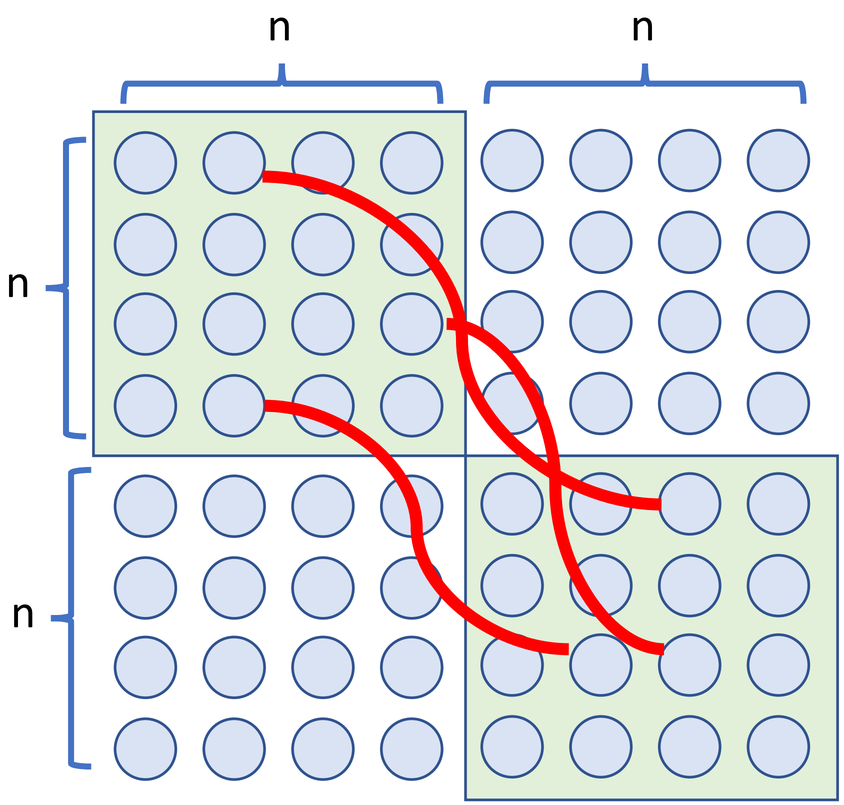
Proof idea of hardness of MBPSP*

Theorem: assuming Exponential Time Hypothesis every Turing machine computing MBPSP* requires time $N^{\Omega(\log\log N)}$

We use the same proof structure introduced by Ilango for showing ETH-hardness of MCSP*



(n x n)-Bipartite Permutation Independent Set



- Graph with 2n x 2n vertices,
- Edges exist only between vertices from two quadrants
- Determine whether there exists a set with one vertex from every row, and one vertex from every column, such that
  - These vertices are from the two quadrants
  - These vertices form independent set

The hardness reduction

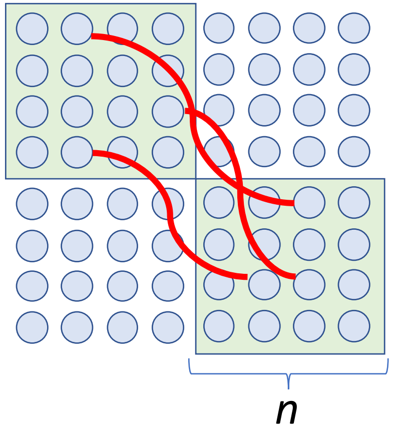
(n x n)-Bipartite Permutation Independent Set Problem

ETH-hard

Exp-time reduction

MBPSP*

Graph G



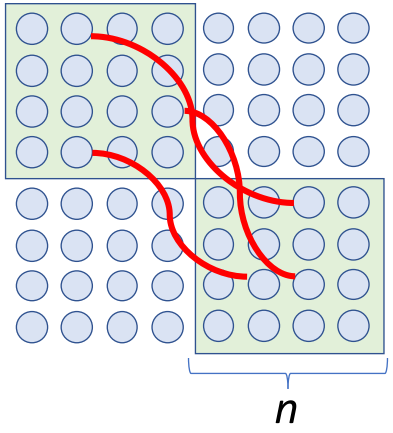
1	*	0	*	*	1	1	0	…	1

Truth table of a partial function γ_G that depends on $\mathbf{6n}$ variables

Key lemma: there exists a total function consistent with γ_G which can be computed by a branching program of size $\mathbf{6n}$ ⇔ G is a yes-instance of (n x n)-BPIS

The hardness reduction

Graph G



1	*	0	*	*	1	1	0	…	1

Truth table of a partial function γ_G
that depends on $\mathbf{6}\boldsymbol{n}$ variables

Key lemma: there exists a total function consistent with γ_G which can be computed by a branching program of size $\mathbf{6}\boldsymbol{n}$ ⇔ G is a yes-instance of (n x n)-BPIS

Idea of Ilango's proof for circuit and formulas:

γ_G depends on $6n$ variables $x_1, \dots, x_{2n}, y_1, \dots, y_{2n}, z_1, \dots, z_{2n}$

a good permutation in G exists iff we can compute γ_G as $\bigvee((y_i \wedge x_k) \vee z_i)$

computable by a read-once monotone formula

The hardness reduction

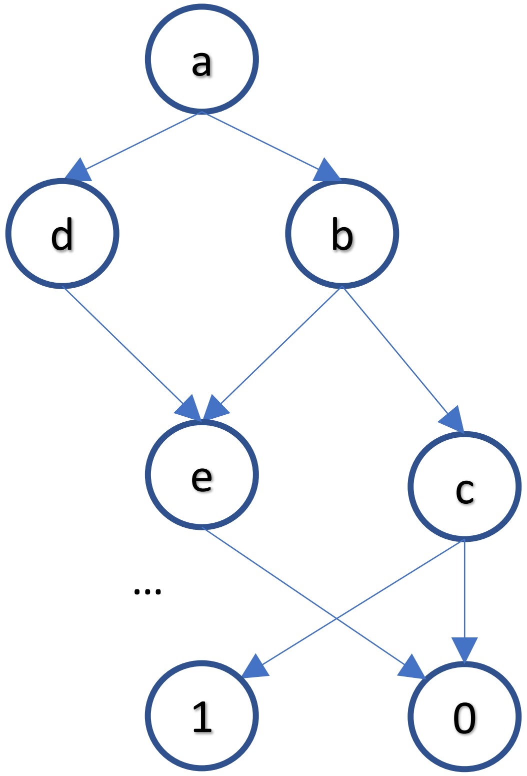
Key lemma: there exists a total Boolean function consistent with γ_G which can be computed by a branching program of size $6n$ ⇔ G is a yes-instance of (n x n)-BPIS

Proof idea:

γ_G depends on $6n$ variables

If G has a bipartite permutation independent set, then

- γ_G can be computed by a **very restricted BP** in which every variable occurs at most once
- As γ_G is sensitive in all variables that MBPSP*(γ_G)= $6n$



We call such BPs
once-appearance BPs

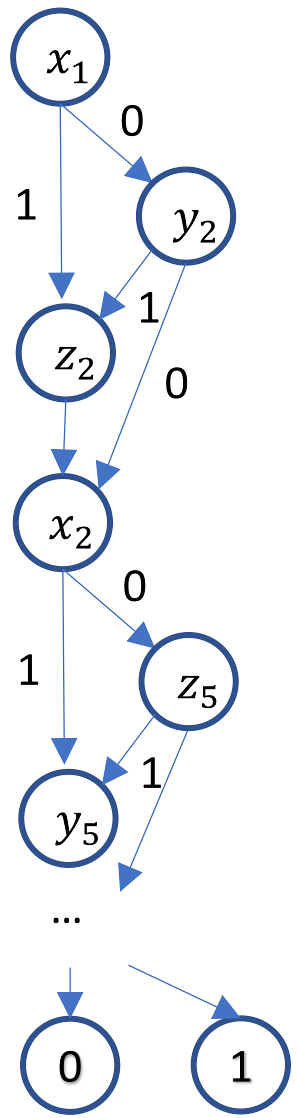
The hardness reduction

If there exists a once-appearance BP computing γ_G => we can extract a permutation on $[2n]$ corresponding to an independent set in G from such BP

γ_G depends on $6n$ variables $x_1, \ldots, x_{2n}, y_1, \ldots, y_{2n}, z_1, \ldots, z_{2n}$

A once-appearance BP for γ_G has a very specific shape:

- Topological sort of the nodes forms groups of xyz-triplets
- In each such triplet y and z have the same index
- If for every triplet x_k, y_i, z_i we map $k \rightarrow i$, we get a bipartite permutation set in G



Corollaries: ETH-hardness of k-BP minimization

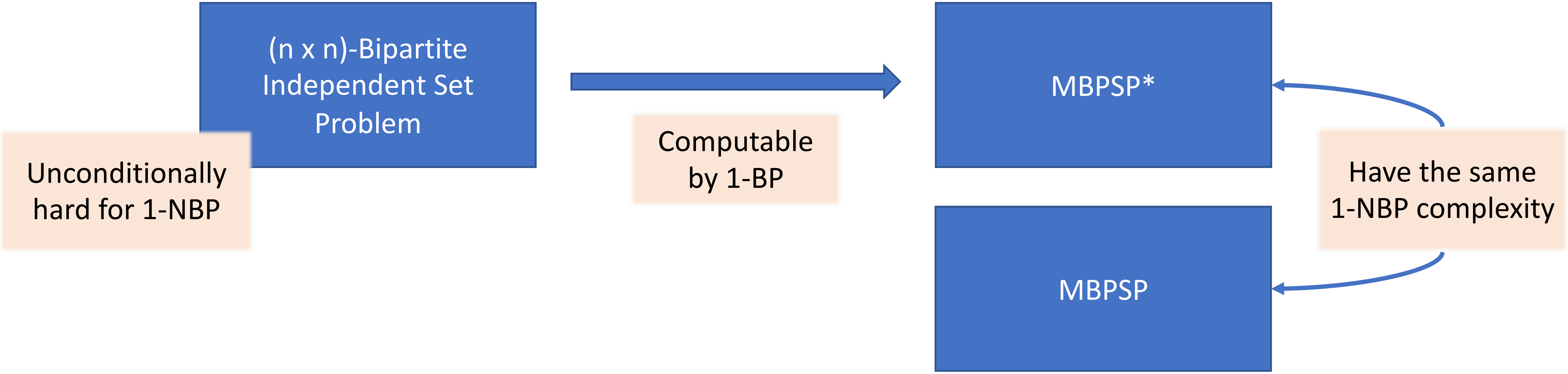
We showed that it is hard to distinguish whether γ_G can be represented by a once-appearance BP or not

Corollary 1: assuming Exponential Time Hypothesis for every k every Turing machine computing Partial Minimum k-BP Size Problem requires time $N^{\Omega(\log\log N)}$

or OBDD

Corollaries: 1-NBP complexity

Corollary 2: size of 1-NBP computing MBPSP is $N^{\Omega(\log\log N)}$



Corollaries: NP and coNP-hardness

Corollary 3: the problem of compressing an input partial BP to a specific size is NP and coNP-hard

NP-hardness

1	*	0	*	*	1	1	0	…	1

Truth table of a partial function γ_G that depends on $\boldsymbol{6n}$ variables

This partial function can be easily computed by a 2-BP over {0,1,*} of polynomial size and can be compressed to a BP of linear size iff G is in BPIS

coNP-hardness

$\phi \in UNSAT_4$

This total function can be easily computed by a 4-BP over {0,1} of size 0 if the formula ϕ is unsatisfiable

Next steps: hardness of total MBPSP

Extend this result to total MBPSP

- Already shown for DeMorgan Formulas [Ilango’21], DNFs [Masek’79]

Can we use ideas like what Ilango used for Formulas?

- Not sure, as BPs are very good in re-using some of its states (similarly to circuits)

But possibly showing the reduction from partial 1-BP minimization to total 1-BP minimization

Next steps: connections to other results

MBPSP, similarly to MCSP, is a sparse language

- Existence of OWF is equivalent to hardness-on-average of a sparse language
- Better-than-linear LB for some sparse language
  => NP $\not\subset$ SIZE$[n^k]$ for all k

[Liu, Pass'23]

[Chen, Jin, Williams'19]

Can we show any connections specific to MBPSP?

Next steps: oaBPs

In our proof we use a very weak class of BPs: **once-appearance BPs**
How powerful is this class of BPs?

- Every read-once formulas over basis without XOR (and its negation) can be converted to a once-appearance BP of the same or smaller size
- $(y \wedge x) \vee (z \wedge \neg x)$ can be computed by a once-appearance BP, but cannot be computed by read-once DeMorgan formulas

Any other connections?

Recap

- Partial MBPSP is ETH hard
  - Holds for various restricted versions of BPs such as OBDDs, 1-BPs, k-BPs
- Unconditional 1-NBP hardness of BP minimization for general and restricted BPs
- NP- and coNP-hardness of compressing branching programs