On Optimal XOR-Circuits: The One True Shape

...

Ngu (Nathan) Dang

BU Theory Seminar - 04/29/2024

Joint work with Marco Carmosino (IBM) and Tim Jackman

Primer

This talk is mainly based on the following paper:

Minimal XOR Circuits: The One True Shape is a Binary Tree

Submitted to CCC'24 & Draft of the Journal Version available here:

https://mcsp.work/opt-xor-shape.pdf

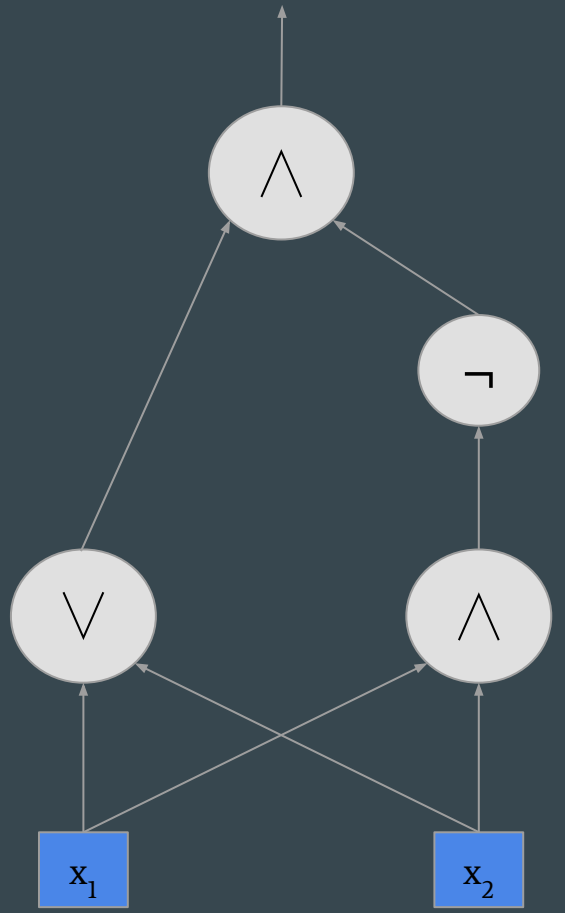
Background & Motivation

- What are Circuits?
- Measuring Circuit Complexity
- Research Questions
- Our Motivation & Result
- Connection with Prior Work
- Does counting NOT-gates matter?

What are Circuits?

Circuits model the computation of Boolean functions on fixed input length by acyclic wires between “logical gates”, i.e. Directed Acyclic Graphs (DAGs).

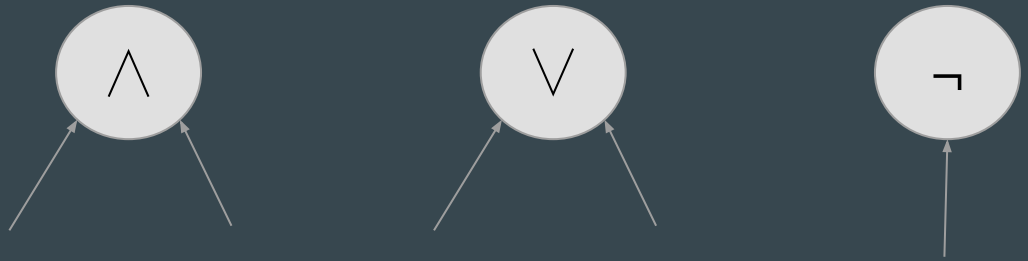
- There is 1 or multiple sources (variables) and exactly 1 sink (output),
- Each internal node including the sink are labeled with logical operations.



What are Circuits?

basis = a finite set of logical gates.

- In our work, we consider the **DeMorgan basis** $= \{\wedge, \vee, \neg\}$



- $\wedge$-gates and $\vee$-gates have fan-in 2, and $\neg$-gates have fan-in 1.
- We allow all types of gates to have **unbounded** fan-out.

Measuring Circuit Complexity

- Measurements of **Circuit Complexity** of a Boolean function f, given a basis $\mathcal{B}$
  - count the number of gates or wires required to construct a circuit for f,
  - compute the circuit-depth, i.e. the longest path from a variable to the output.
- For our work, $\mathcal{B} = \{\wedge, \vee, \neg\}$
  - The Circuit Complexity (CC) is the number of $\wedge$-gates and $\vee$-gates, we refer them as the "**costly**" gates.
  - **¬-gates do not count towards the CC.**
- For the rest of the talk, we consider **normalized circuits** (i.e. no double ¬).

Measuring Circuit Complexity

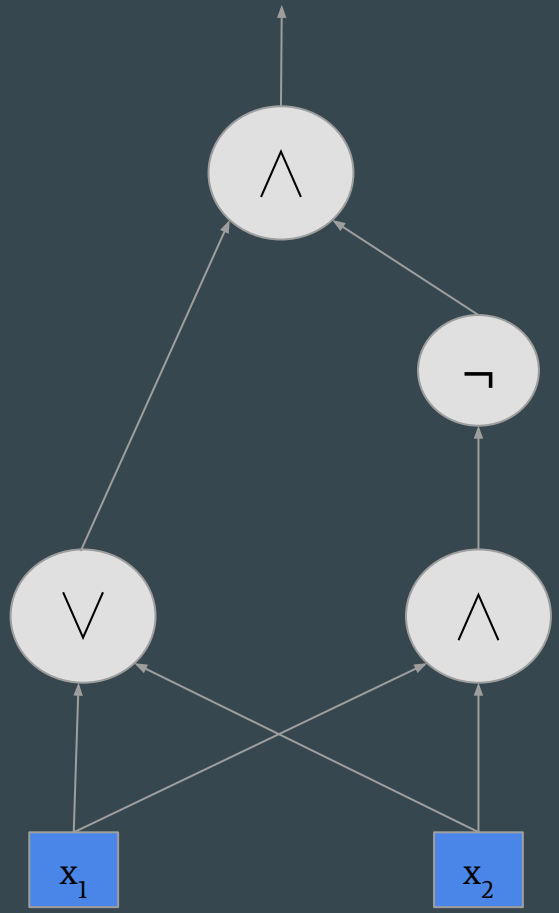
- A warm-up example:

  $XOR_2(x_1, x_2) = (x_1 \vee x_2) \wedge \neg(x_1 \wedge x_2)$

  $CC(XOR_2) = 3$

This circuit is **optimal** under our measurement!

use the **least** number of costly gates



Measuring Circuit Complexity

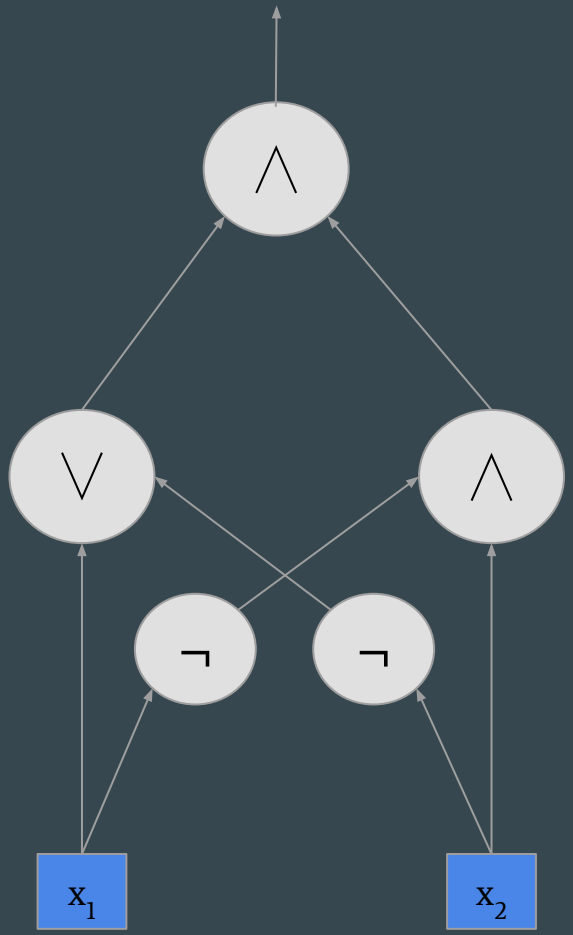
- A warm-up example:

  $XOR_2(x_1, x_2) = (x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$

  $CC(XOR_2) = 3$

This circuit is also **optimal** under our complexity measurement!

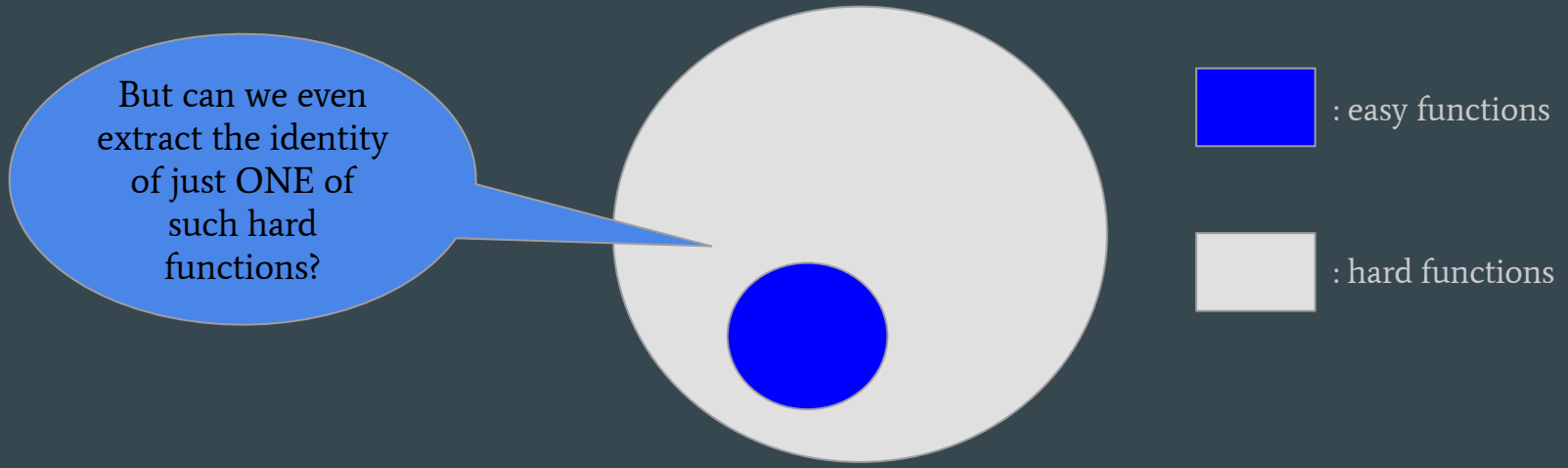
But when ¬-gates count toward the complexity, this circuit is no longer optimal!



Research Questions

The existential question: ***Do functions that require "large circuits" exist?***

Solved by Shannon who showed that almost all Boolean functions require circuits of size $\Omega(2^n/n)$ [Sha 49]



Research Questions

The algorithmic question: ***The Minimum Circuit Size Problem (MCSP)***

- Input: an n-input Boolean function f as a 2^n-bit truth table, an integer s
- Question: does there exist a DeMorgan circuit with at most s gates that computes f ?

What do we know so far about **MCSP**?

- **MCSP** is in **NP**.
- That's pretty much it... 
- Whether it is **NP**-complete is a major open problem!
- Many variants of MCSP are NP-complete! [Mas 79, HOS 18, ILO 20, Ila 20]

Research Questions

The algorithmic question: ***The Minimum Circuit Size Problem (MCSP)***

- Input: an n-input Boolean function f as a 2^n-bit truth table, an integer s
- Question: does there exist a DeMorgan circuit with at most s gates that computes f ?

MCSP is **NP**-complete implies major complexity breakthrough!

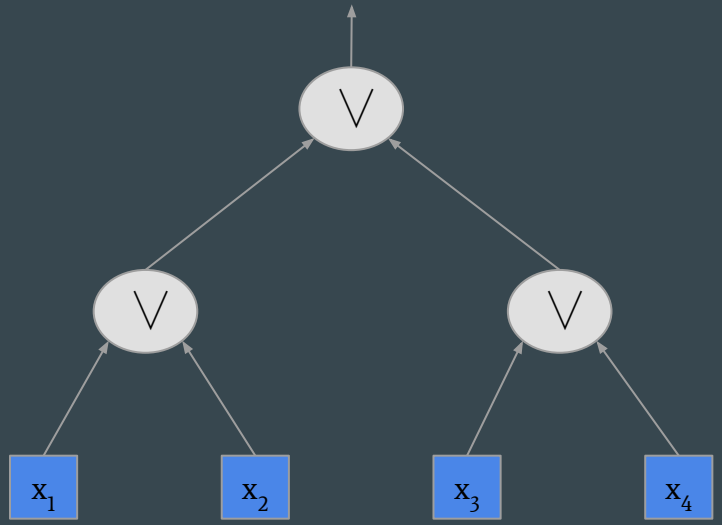
e.g. the class EXP does not have small DeMorgan circuits! [KC 00, SS 20]

Research Questions

This work – The design question: given a Boolean function f, can we characterize every optimal circuit computing f?

For some functions, this is easy to answer! e.g. minimal circuits for n-bit OR/AND are simply trees of (n – 1) costly gates.

Ex: an optimal OR_4 circuit



Research Questions

This work – The design question: given a Boolean function f , can we characterize every optimal circuit computing f?

What about some slightly more complicated functions, e.g. XOR, the parity function?

- If parity-gates (e.g. $\oplus$) are allowed, then the optimal circuits computing XOR is similar to that of OR, a tree of (n - 1) $\oplus$-gates.

- So, given the DeMorgan Basis, would the “shape” of such circuits change significantly?

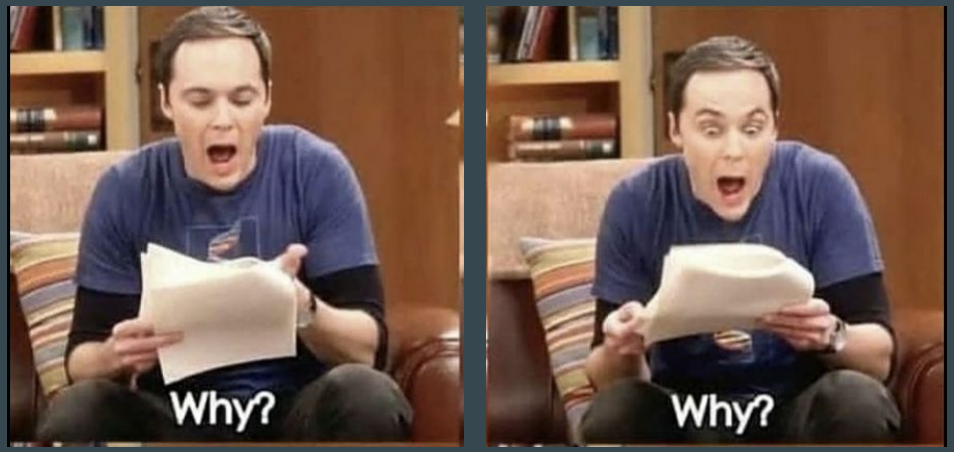
Research Questions

This work – The design question: given a Boolean function f , can we characterize every optimal circuit computing f?

What about some slightly more complicated functions, e.g. XOR, the parity function?

- But most of all, **why bother?**

?? 😀 ??



(source: imgflip)

Our Motivation – Hardness of Partial MCSP

Under the Exponential Time Hypothesis (ETH), Ilango showed hardness of Partial MCSP (a variant of MCSP for partial function) [Ila 20].

Partial function $f: \{0, 1\}^n \rightarrow \{0, 1, *\}$, defined on a subset of inputs, i.e. return 0 or 1 given strings in the subset, and return * on everything else.

The Partial Minimum Circuit Size Problem (MCSP*)

- Input: an n-input **Partial** Boolean function f as a 2^n-bit truth table, an integer s
- Question: does there exist a DeMorgan circuit with **at most** s gates that agrees with f on defined inputs ?

Our Motivation – Hardness of Partial MCSP

The Exponential Time Hypothesis (ETH): 3SAT cannot be solved in DTIME($2^{o(n)}$)

Theorem [Ila 20]: Assuming ETH, MCSP* $\notin$ **P**

We will inspect this in a bit!

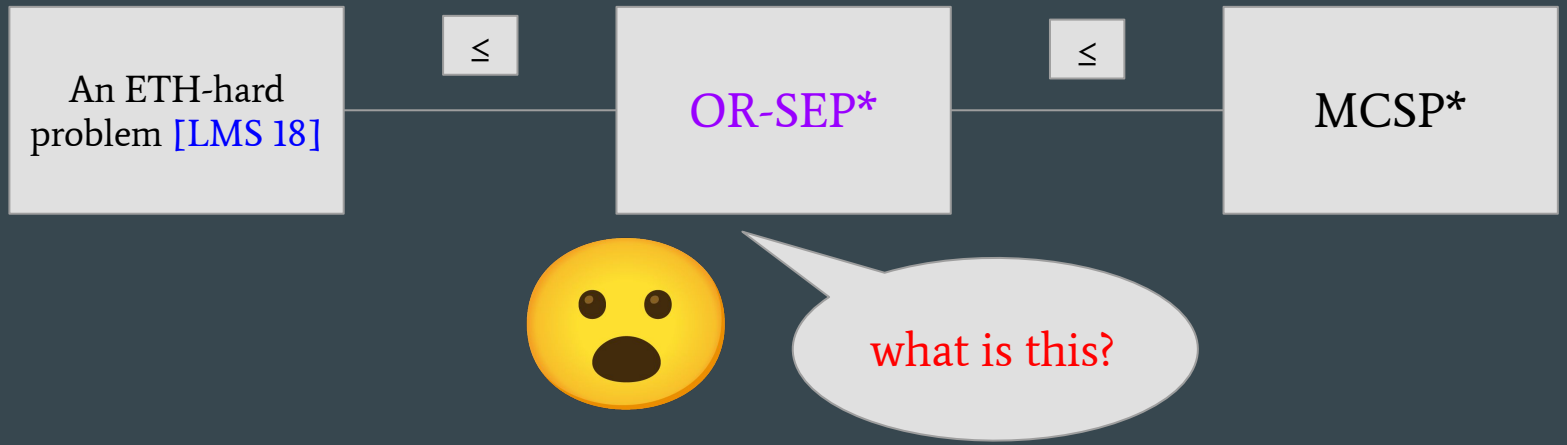
Open Problem: Assuming ETH, MCSP $\notin$ **P**

Our Motivation – Hardness of Partial MCSP

The Exponential Time Hypothesis (ETH): 3SAT cannot be solved in DTIME($2^{o(n)}$)

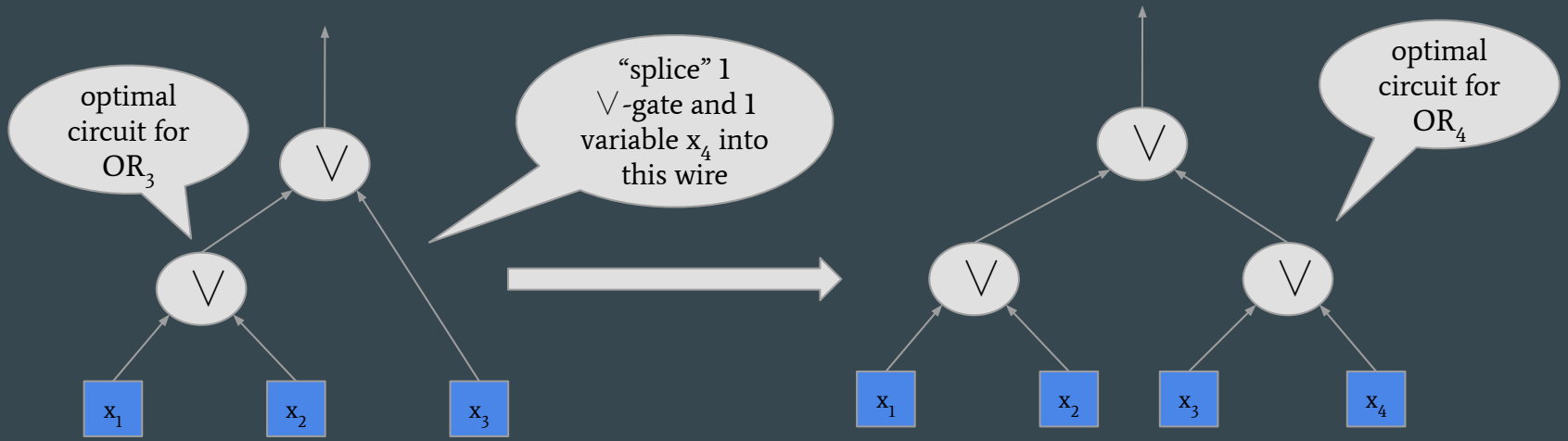
Theorem [Ila 20]: Assuming ETH, MCSP* $\notin$ **P**

Summary of Ilango's Reduction:



Our Motivation – Hardness of Partial MCSP

Simple Extension: let $f: \{0,1\}^n \to \{0,1\}$ be a Boolean function, then $g: \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ is said to be a Simple Extension of f if optimal circuits computing g can be obtained by adding **exactly** m extra costly gates to optimal circuits computing f.



$f(x) = (x_1 \vee x_2) \vee x_3$

Simple Extension

$g(x) = (x_1 \vee x_2) \vee (x_3 \vee x_4)$

Our Motivation – Hardness of Partial MCSP

Simple Extension: let $f: \{0,1\}^n \to \{0,1\}$ be a Boolean function, then $g: \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ is said to be a Simple Extension of f if optimal circuits computing g can be obtained by adding **exactly** m extra gates to optimal circuits computing f.

The f-Simple Extension Problem (f-SEP):

- Input: a Boolean function $g: \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ as a 2^{n+m}-bit truth-table
- Question: is g a Simple Extension of $f: \{0,1\}^n \to \{0,1\}$?

Fact: f-SEP ≤ MCSP

extend to partial function

$g: \{0,1\}^n \times \{0,1\}^m \to \{0,1,*\}$

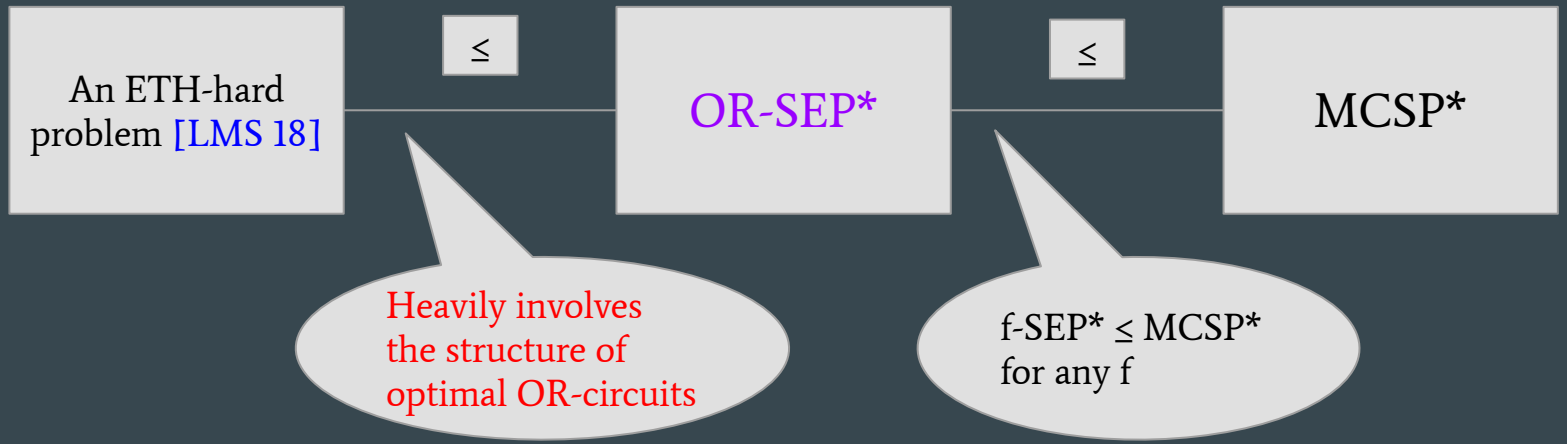
Fact: f-SEP* ≤ MCSP*

Our Motivation – Hardness of Partial MCSP

The Exponential Time Hypothesis (ETH): 3SAT cannot be solved in DTIME($2^{o(n)}$)

Theorem [Ila 20]: Assuming ETH, MCSP* $\notin$ **P**

Summary of Ilango's Reduction:



Our Motivation - Does The Framework Still Work for Total MCSP?

The same framework **does not hold** for proving hardness of (total) MCSP !

> "the set of optimal OR-circuits is so well structured that OR-SEP $\in$ P"
> *Rahul Ilango, SIAM J. of Computing, 2022*

Facts:

- Optimal OR-circuits are Read-once Formulae.
- Simple Extension of a Read-once Formula is also a Read-once Formula.
- There exists poly-time algorithms that learn the exact identification of a given the truth-table of a read-once formula [AHK 93, BHH 95].

Our Motivation

Earlier, we asked Why care about characterizing optimal circuits?

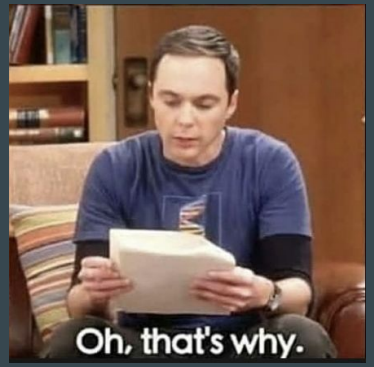
Now, we know Ilango's framework does not hold for proving hardness of (total) MCSP !

Thus...

(source: imgflip)

"… the missing component in extending our results to MCSP is finding some function f whose optimal circuits we can characterize but are also sufficiently complex."

Rahul Ilango, SIAM J. of Computing, 2022



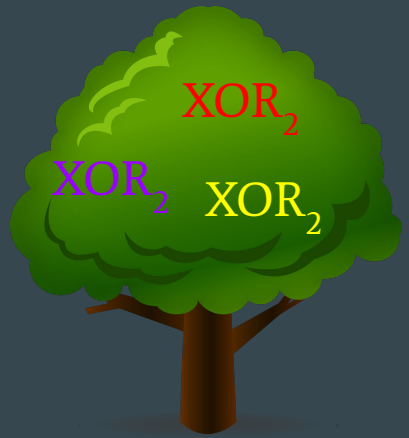
Our Result – Statement

Our work characterizes optimal circuits for the function f = XOR

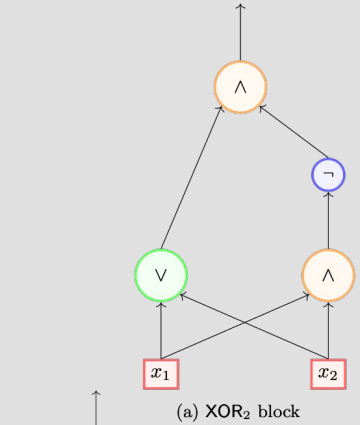
Theorem: Optimal (¬)XOR_n circuits over the DeMorgan basis partition into trees of (n - 1) (¬)XOR_2 sub-circuits when ¬-gates are free.

Visualization...

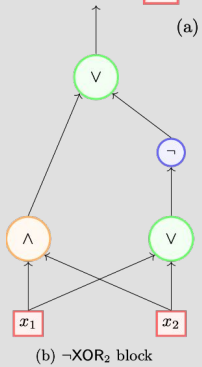
Opt XOR-Ckt ≡



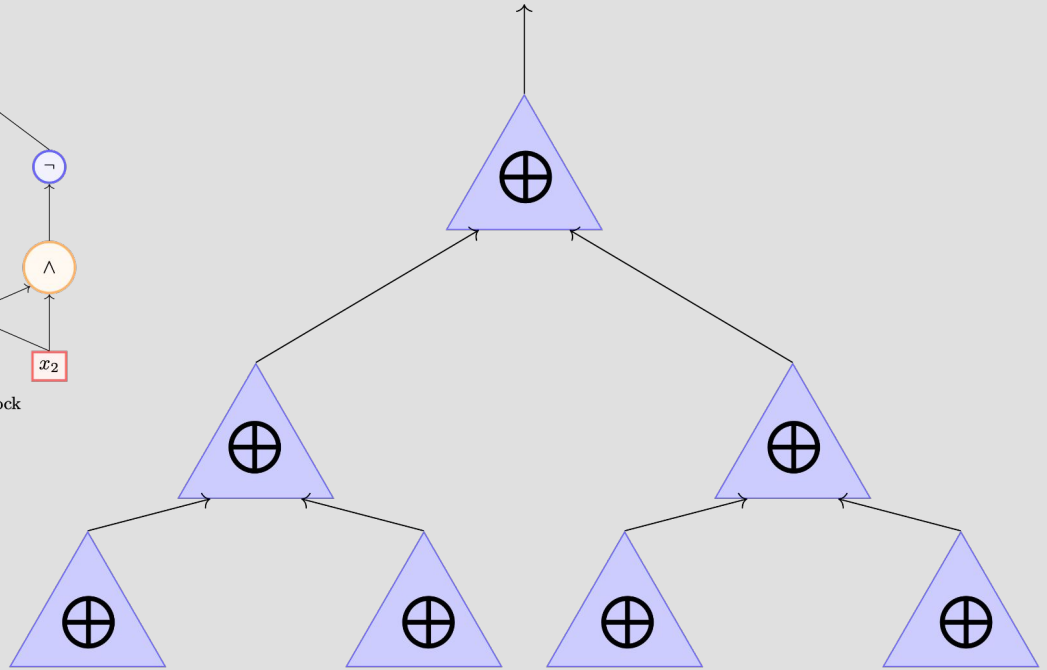
Our Result – Visualization



(a) XOR_2 block



(b) $\neg\mathsf{XOR}_2$ block



(c) $(\neg)\mathsf{XOR}_8$ optimal circuit shape which consists of $8-1=7$ non-intersecting blocks where each is either Figure 1a or 1b

Our Result – Connection to Prior Work

This structure for optimals circuits for XOR_n was already established by Kombarov.

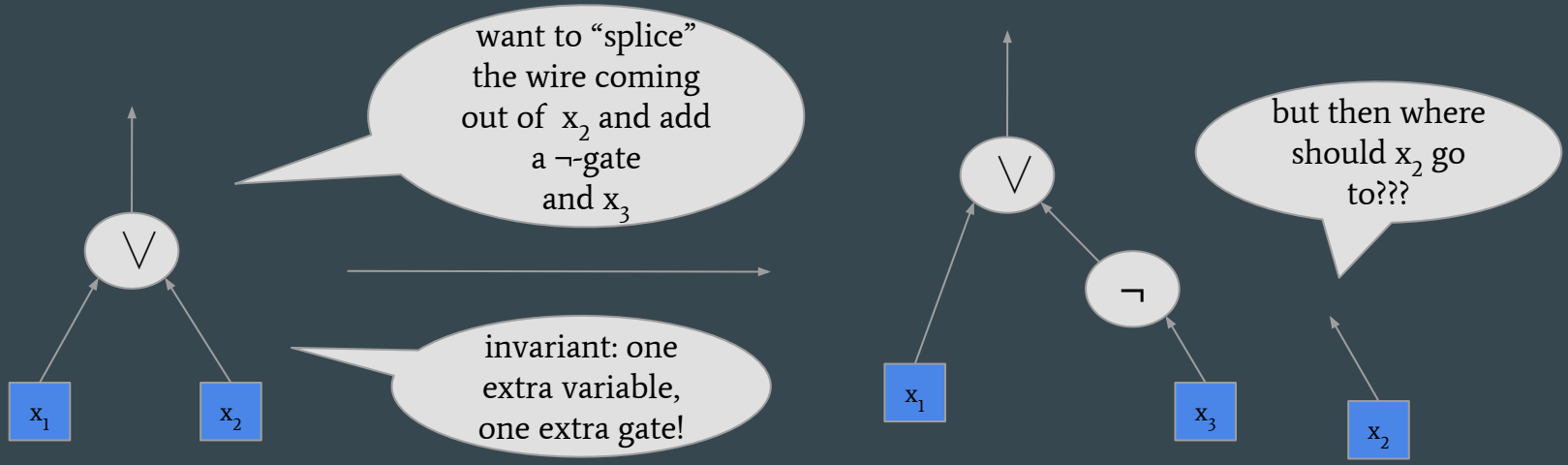
But we can't directly use this result to address Ilango's question!

The reason is there are two issues:

- Kombarov's result counts ¬-gates towards the circuit complexity! [Kom 11]
- It seems plausible that the same result might not hold since ¬-gates **are free to use.**

Does NOT-gate fit in with Simple Extension?

Observation: the definition of Simple Extension is not compatible with counting ¬-gates towards the CC



Basic Tools

- XOR-function
- Gate Elimination Proof Framework
- Warm-up: Schnorr's Lower-bound

The XOR-Function

For an n-bit string $x \in \{0, 1\}^n$, $n \geq 2$, we define $XOR_n(x)$ as follows

- $XOR_n(x) = 1$ if an odd number of bits of x are 1, and
- $XOR_n(x) = 0$ otherwise

Fact 1 (XOR is Downward Self-Reducible (DSR)): For any input $x \in \{0, 1\}^n$, we have
$XOR_n(x_1 \ldots x_n) = XOR_{n-1}(x_1 \ldots x_{n-1}) \oplus x_n$

Fact 2 (XOR is Read-twice): let C be an optimal circuit for XOR_n, then every variable in C has fan-out 2.

The XOR-Function

For an n-bit string $x \in \{0, 1\}^n$, $n \geq 2$, we define $XOR_n(x)$ as follows

- $XOR_n(x) = 1$ if an odd number of bits of x are 1, and
- $XOR_n(x) = 0$ otherwise

Fact 3 (XOR is Fully Sensitive): for all $i \in [n]$ and **for all** assignments $x = b$, we have $XOR_n(b) \neq XOR_n(b \oplus e_i)$

b with the i-th bit flipped

Gate Elimination (GE) – Definition

Let C be a circuit computing $f: \{0,1\}^n \to \{0,1\}$

For $i \in [n]$ and $b \in \{0,1\}$, we consider the 1-bit restriction $x_i \leftarrow b$ which sets the i-th variable x_i to the constant b.

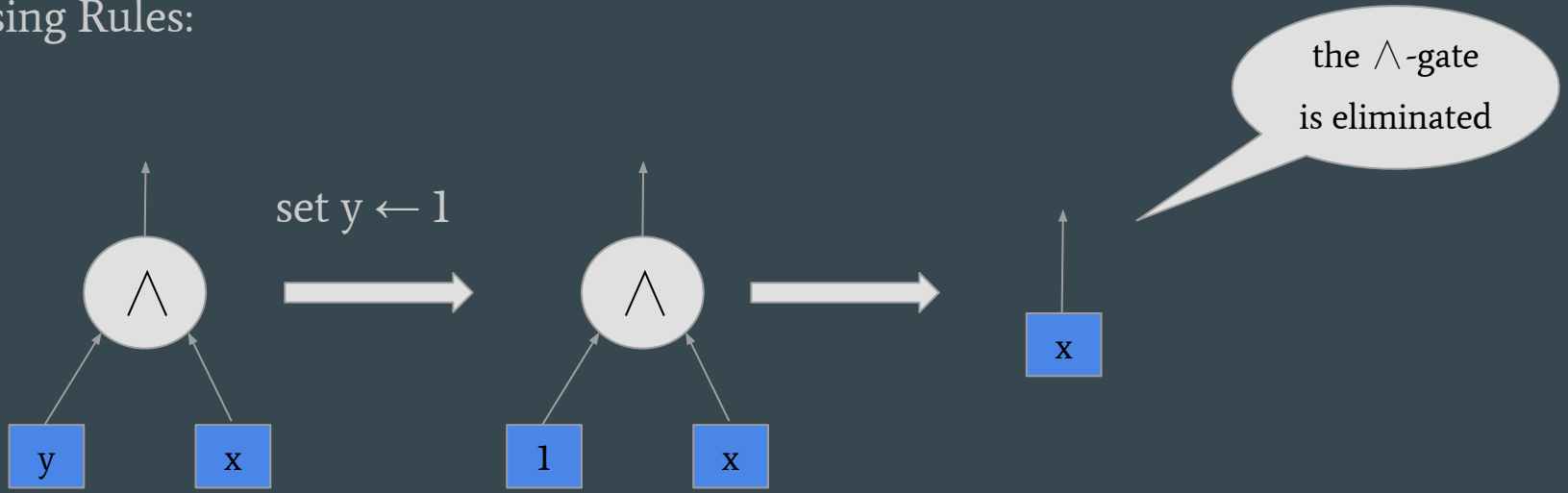
$C' = C|_{x_i \leftarrow b}$ is the $\leq (n-1)$-ary circuit obtained from C as follows.

- substitute $x_i \leftarrow b$ for all inputs labeled by x_i
- perform the following constant **simplification rules** on sub-circuits of C whenever possible

two types:
passing and
fixing rules

Gate Elimination (GE) – Rules

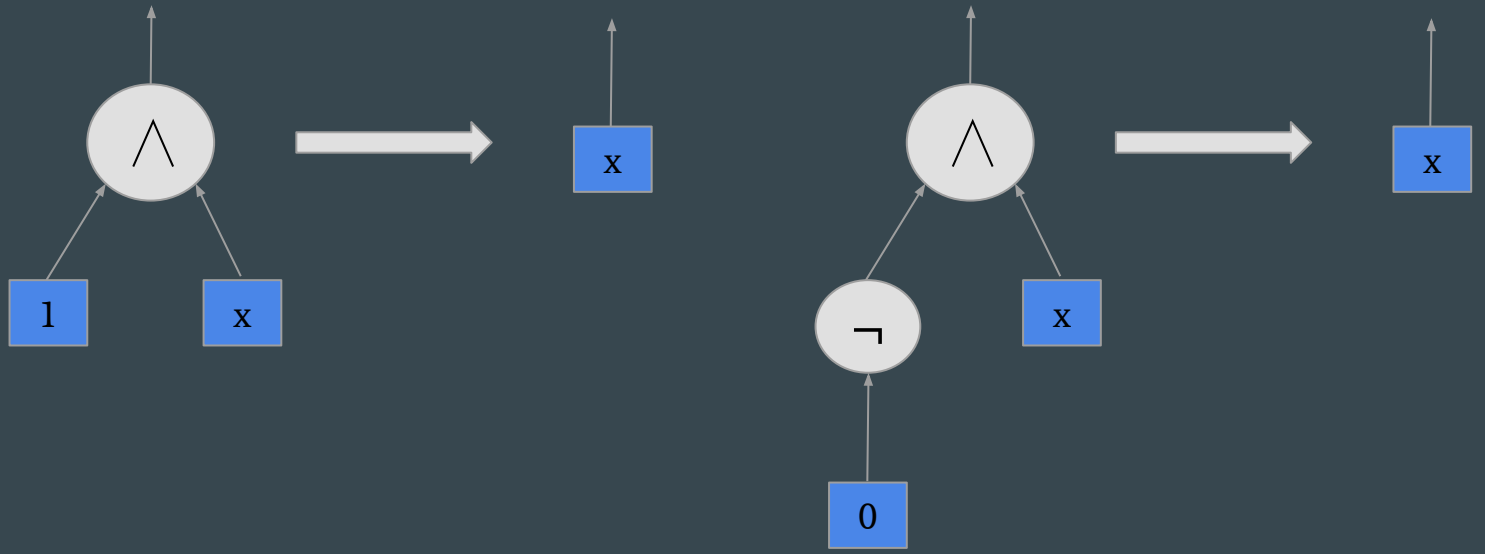
Passing Rules:



Gate Elimination (GE) – Rules

Passing Rules:

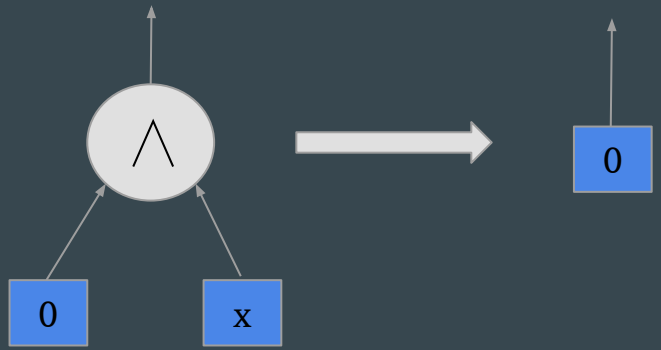
Key Takeaway : the variable x is still relevant to the rest of the circuit!

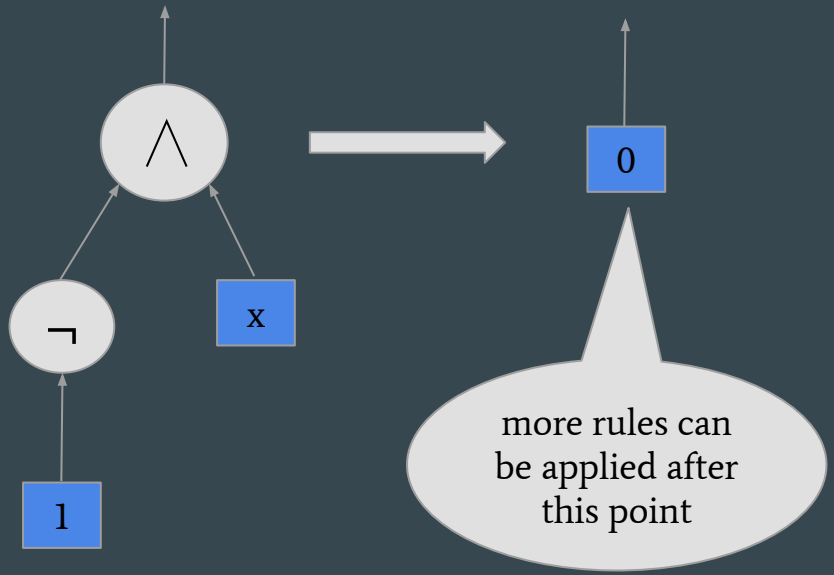


Gate Elimination (GE) – Rules

Fixing Rules:

Key Takeaway : the variable x **may or may not** be disconnected completely from the rest of the circuit!





Gate Elimination (GE) – Proof Framework

Task: given a circuit C, show C has some property $\mathcal{P}$

Framework: Repeats the following proof by contradiction argument

1. Assume C has $\neg\mathcal{P}$
2. Select a variable x_i and set $x_i \leftarrow b \in \{0, 1\}$
3. Apply the rules to simplify C to obtain a constant-free circuit C'
4. Argue that C' has a critical property $\mathcal{Q}$ that implies a contradiction.

Warm-up: Schnorr 3n - o(1) lower bound for XOR

Theorem **[Sch 73]** : The optimal circuit size computing XOR_n under DeMorgan's Basis, free ¬-gates, is at least 3(n - 1).

when ¬-gates count,
$CC(XOR_n) \geq 4(n-1)$
[Red'kin 71]

Proof Sketch:

- The key step is to use a Gate Elimination argument to show that some one-bit restriction eliminates at least 3 costly gates.
- Then, apply Fact 1 (XOR is DSR) to inductively argue for the desired lower-bound.

Warm-up: Schnorr 3n - o(1) lower bound for XOR

Lemma: Let C be a circuit computing XOR_n, for $n \geq 2$, then there exists one-bit restriction that eliminates at least 3 costly gates.

Proof: Let C be a circuit computing XOR_n and let h be a bottom level **costly gate** fed by two distinct variables x_i and x_j.

We want to prove the following two properties for C:

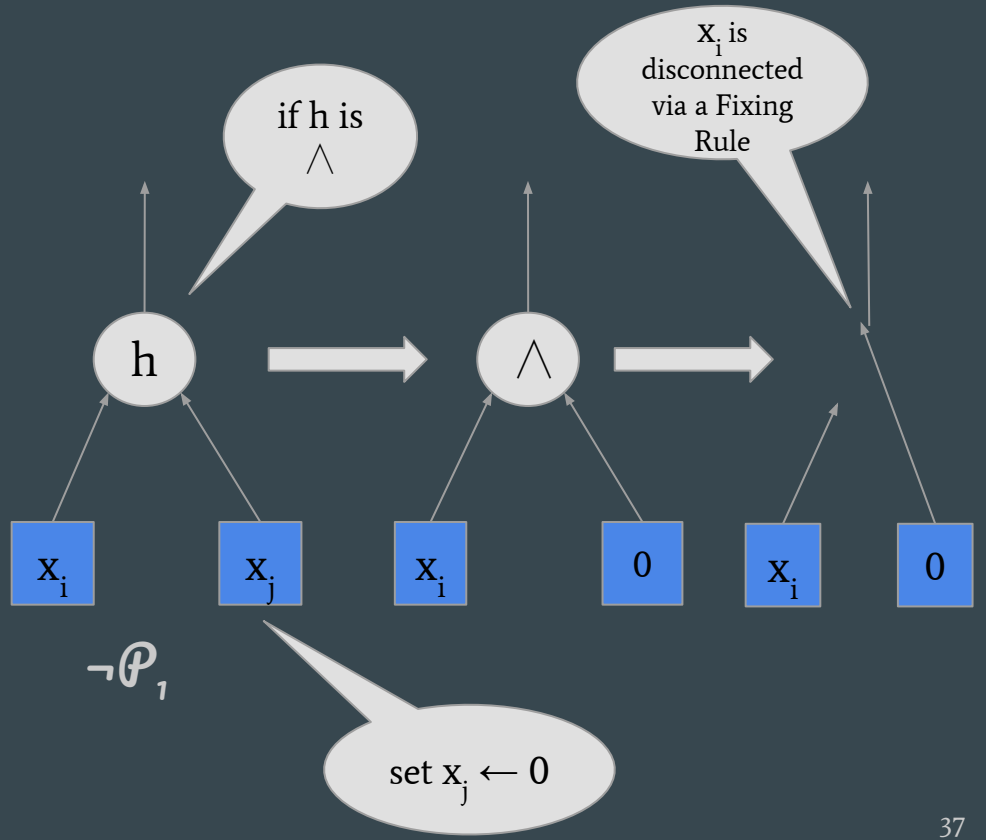
- $\mathcal{P}_1$ = "x_i is fed into another gate $h'_i \neq h$"
- $\mathcal{P}_2$ = "h'_i is not the output gate"

Note: for simplicity, we will omit ¬-gates in diagrams

Warm-up: Schnorr 3n - o(1) lower bound for XOR

Claim 1: C has property $\mathcal{P}_1$ = "x_i is fed into another gate $h'_i \neq h$"

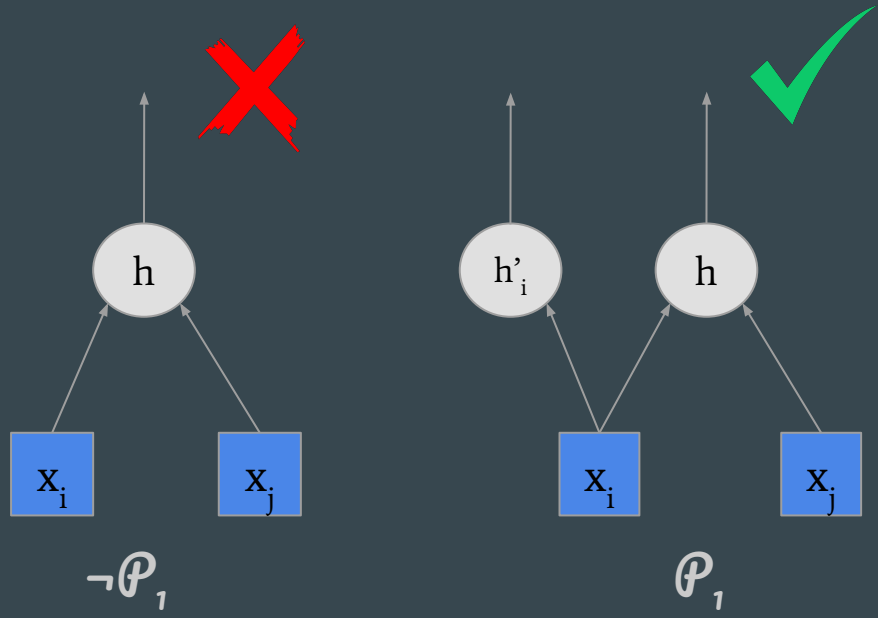
Proof: Assume C has $\neg\mathcal{P}_1$, then there exists an assignment $x_j \leftarrow b \in \{0, 1\}$ such that x_i becomes degenerate which contradicts Fact 3 (XOR is Non-degenerate).



Warm-up: Schnorr 3n - o(1) lower bound for XOR

Claim 1: C has property $\mathcal{P}_1$ = "x_i is fed into another gate $h'_i \neq h$"

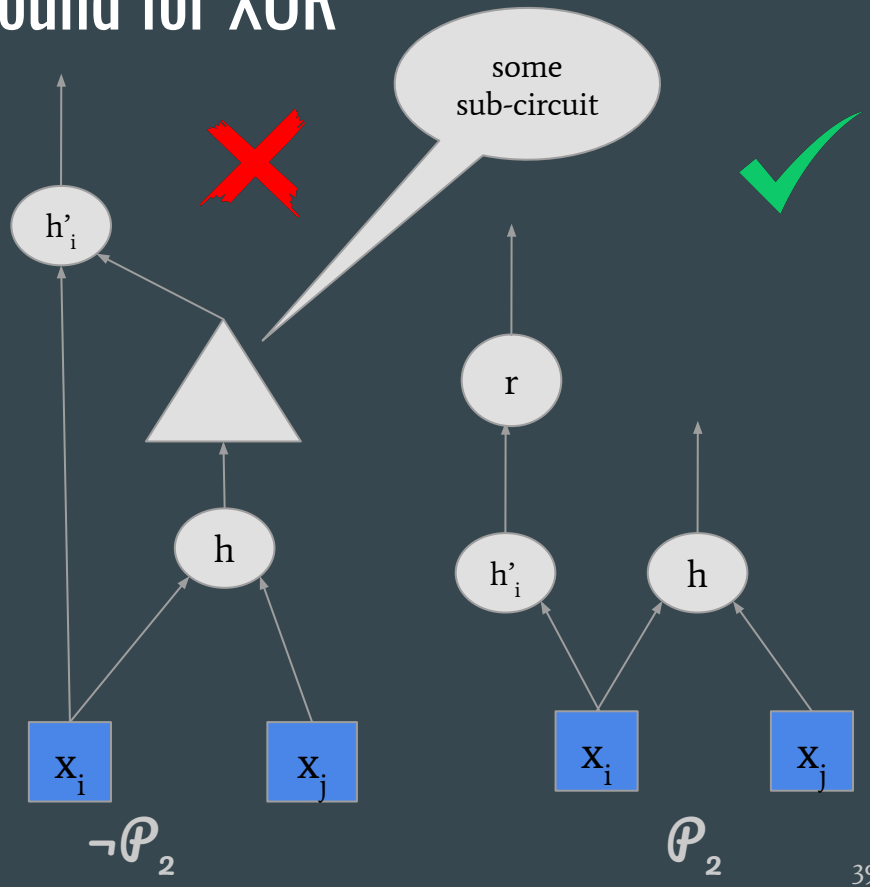
Proof: Assume C has $\neg\mathcal{P}_1$, then there exists an assignment $x_j \leftarrow b \in \{0, 1\}$ such that x_i becomes degenerate which contradicts Fact 3 (XOR is Fully Sensitive).



Warm-up: Schnorr 3n - o(1) lower bound for XOR

Claim 2: C has property $\mathcal{P}_2$ = "h'_i is not the output gate"

Proof: Assume C has $\neg\mathcal{P}_2$, then there exists an assignment $x_i \leftarrow b \in \{0, 1\}$ that makes C constant (via a Fixing Rule) which contradicts "C computes XOR_n". This implies h'_i is "below" another costly gate r.



Warm-up: Schnorr 3n - o(1) lower bound for XOR

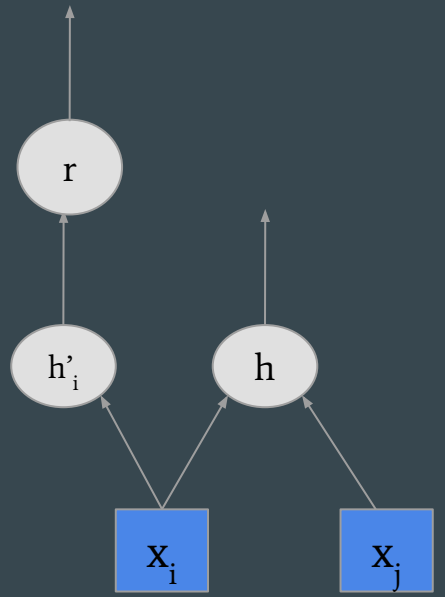
Lemma: Let C be a circuit computing XOR_n, for $n \geq 2$, then there exists one-bit restriction that eliminates at least 3 costly gates.

Set $x_i \leftarrow b \in \{0, 1\}$ so that h'_i is eliminated via a fixing rule.

$\Rightarrow$ r and h will also be eliminated by any type of rules

$\Rightarrow$ at least 3 gates are eliminated (q.e.d.)

To complete the Theorem: Use Fact 1 (XOR is DSR) to inductively achieve the desired lower-bound.



Warm-up: Schnorr 3n - o(1) lower bound for XOR

Observation: The 3(n - 1) lower-bound also has a matching upper-bound, obtained by replacing each of of (n - 1) $\oplus$-gates with a 3-gate block computing XOR_2.

Corollary: The optimal circuit size computing XOR_n under DeMorgan's Basis, free $\neg$-gates, is **exactly** 3(n - 1).

Fact 4: (Elimination Rate Limit) : Let C be an optimal circuit computing XOR_n, then each 1-bit restriction eliminates at most 3 costly gates.

Main Result

- Proof Sketch of Theorem
  - Main Ideas
  - Schnorr's Proof Revisit
  - Fleshing out the Local Structure

Our Technique – Proof Ideas

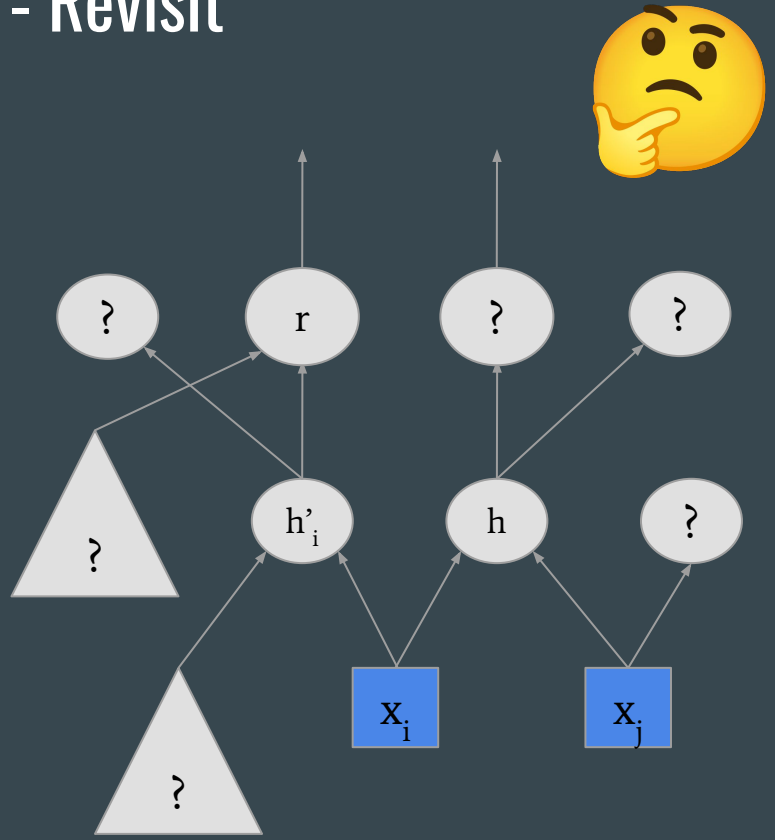
Theorem : Optimal (¬)XOR_n circuits over the DeMorgan basis partition into trees of n - 1 (¬)XOR_2 sub-circuits when ¬-gates are free.

Proof Ideas:

- Extract more information from the local structure given by the proof of Schnorr's lower bound for XOR-circuit.
- Carry out case analysis of restricting and eliminating gates from optimal XOR circuits to characterize the "templates" using the Gate Elimination proof framework.

Schnorr 3n - o(1) lower bound for XOR - Revisit

Lemma: Let C be a circuit computing XOR_n, for $n \geq 2$, then there exists one-bit restriction that eliminates at least 3 costly gates.



Major Unknowns:

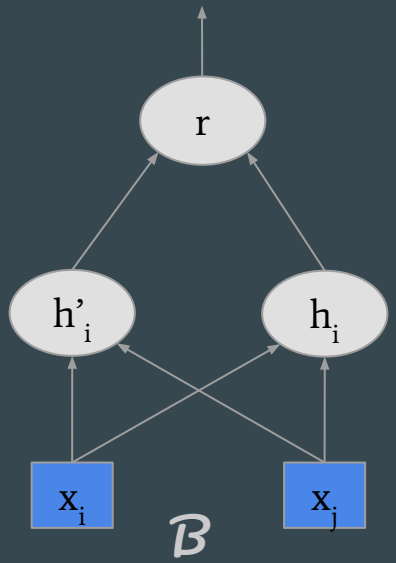
- Second input of r and h'_i ?
- The costly gate that x_j is fed to ?
- The costly gate that h is fed to ?
- Fan-out > 1 for h and h'_i ?

Our Technique – Proof Sketch

Theorem: Optimal (¬)XOR_n circuits over the DeMorgan basis partition into trees of n - 1 (¬)XOR_2 sub-circuits when ¬-gates are free.

Proof Sketch: we adapt the strategy from Schnorr

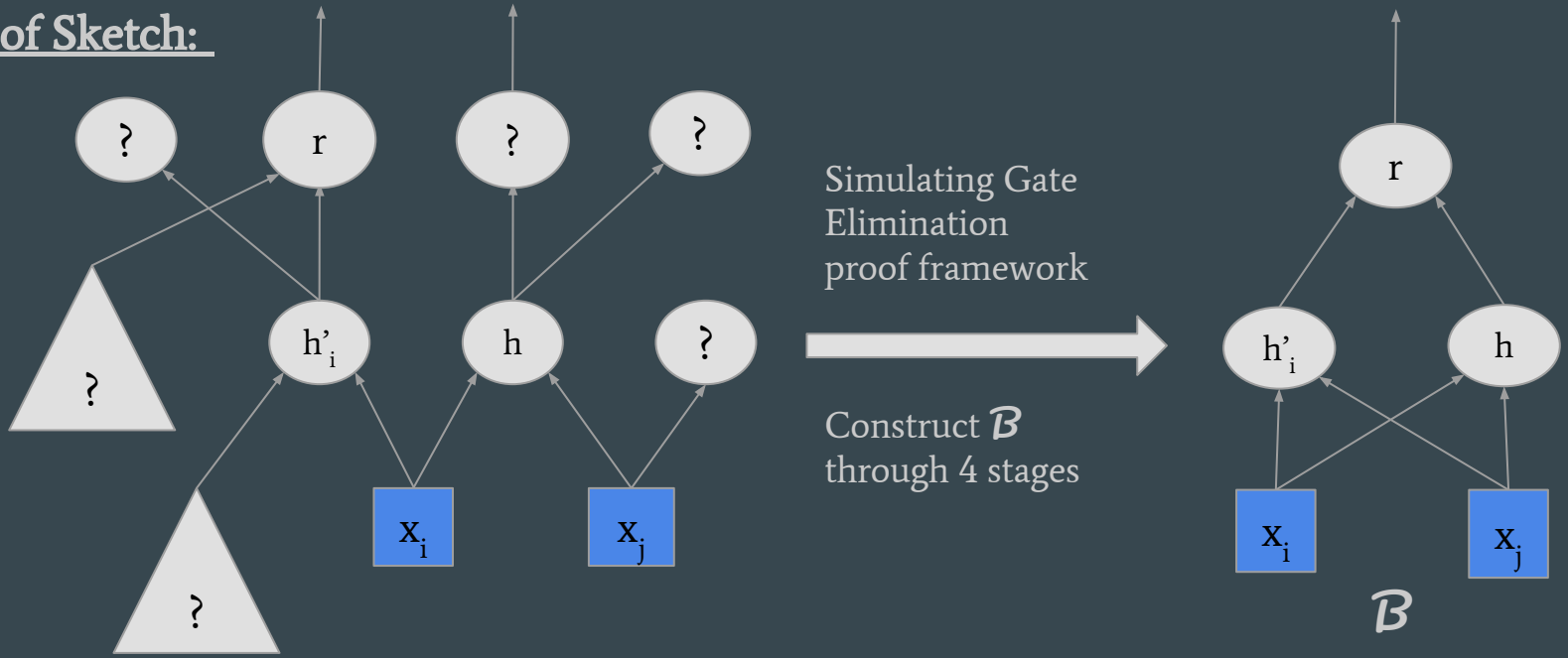
- We will show that there exists two distinct inputs x_i, x_j that are fed into a block $\mathcal{B}$
- Argue $\mathcal{B}$ must compute XOR_2
- Then we apply an inductive argument over the number of variables to obtained the desired optimal circuit template.



Fleshing out the circuit template

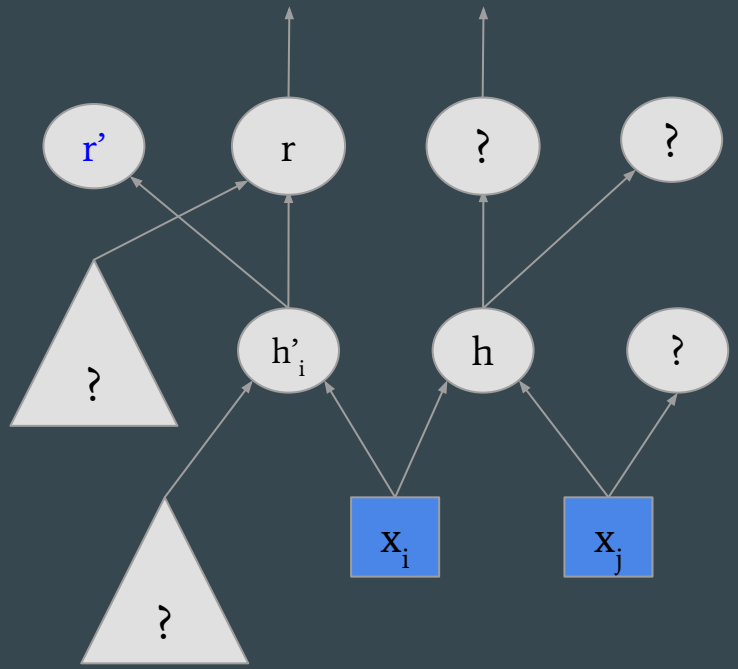
Lemma: there exists two distinct inputs x_i, x_j that are fed into a block $\mathcal{B}$

Proof Sketch:



Fleshing out the circuit template

Stage 1: show $\mathcal{P}_1$ = "r is the only costly successor of h'_i"

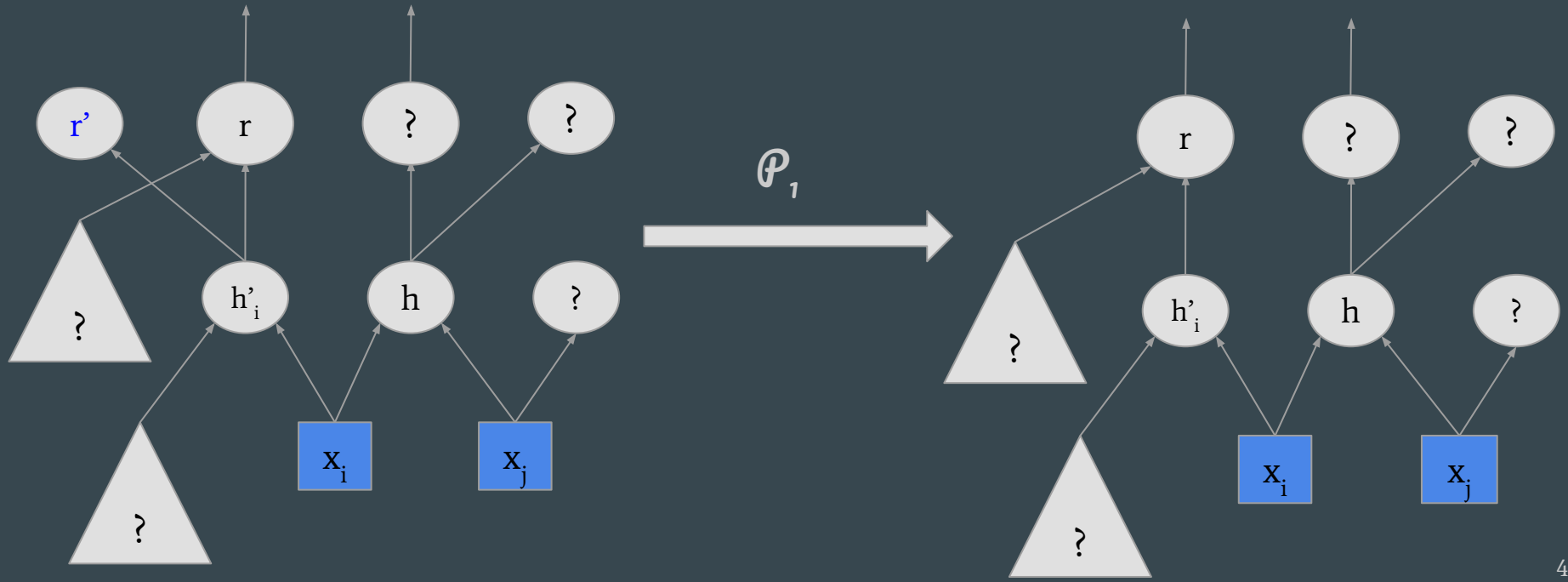


Proof: Suppose $\neg\mathcal{P}_1$ = "h'_i is connected to $r' \neq r$" Then there exists an assignment $x_i \leftarrow b \in \{0, 1\}$ such that h'_i, h, r, and r' are eliminated.

A contradiction to Fact 4 (Elimination Rate Limit) which says any 1-bit restriction can only eliminate 3 gates.

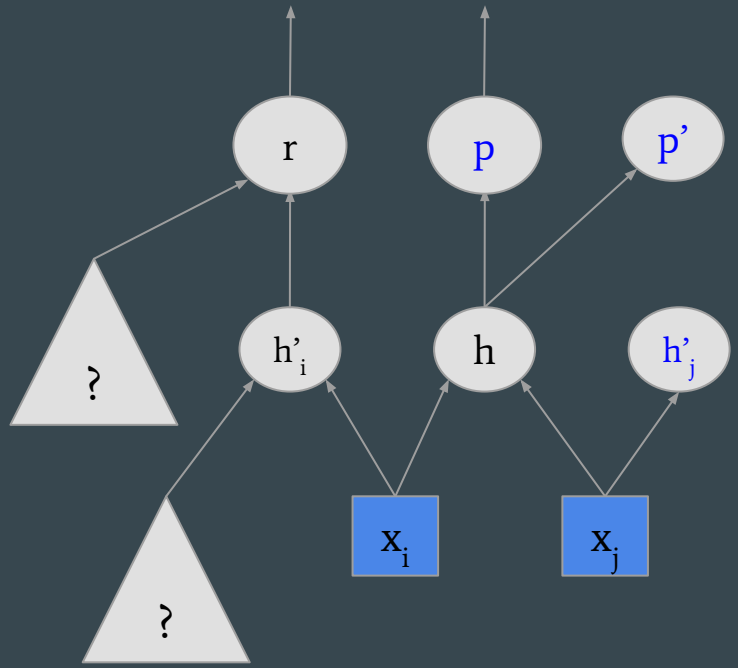
Fleshing out the circuit template

Stage 1: show $\mathcal{P}_1$ = "r is the only costly successor of h'_i"



Fleshing out the circuit template

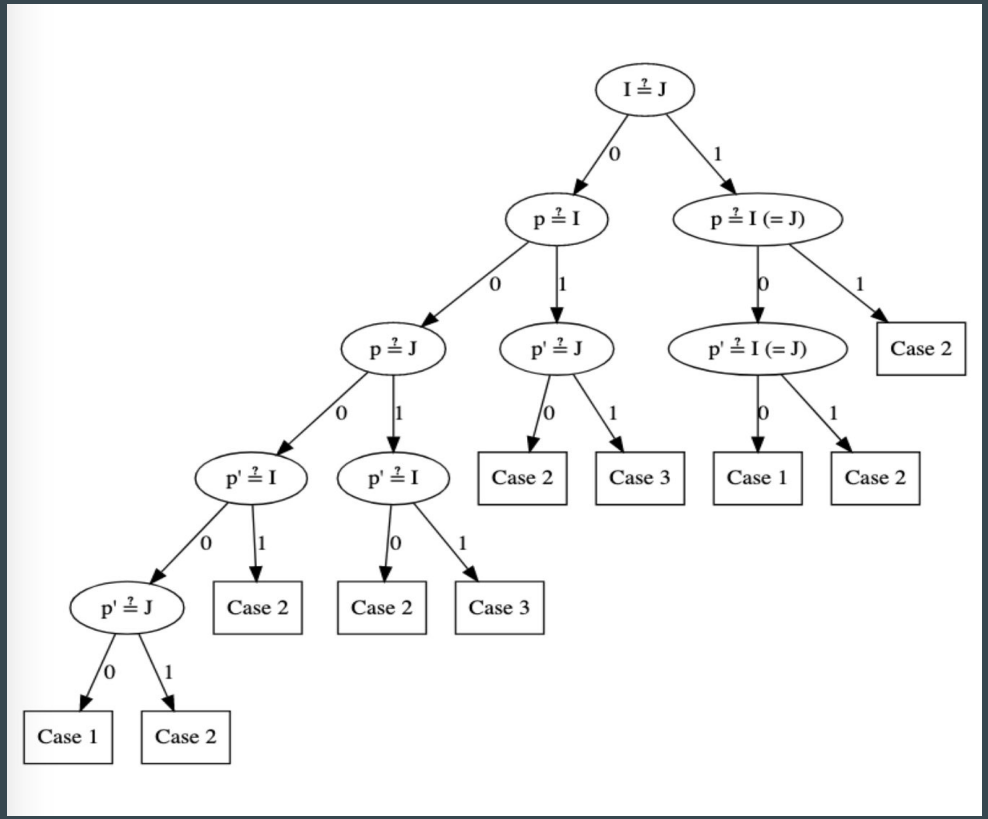
Stage 2: assume h is connected to p, show $\mathcal{P}_2$ = "p is the only costly successor of h"



Proof sketch: Suppose $\neg\mathcal{P}_2$ = "h is connected to p' ≠ p" and x_j is connected to $h'_j \neq h$
Perform case analysis on the identities of p, p', h'_j

1. $p =? h'_j$ (i.e. is p fed to h'_i ?)
2. $p' =? h'_i$ (i.e. is p' fed to h'_i ?)
3. $p' =? h'_j$ (i.e. is x_j fed to p' ?)
4. $h'_j =? h'_i$ (i.e. is x_j fed to h'_i ?)

Fleshing out the circuit template – Pf Sketch of Stage 2 (cont)

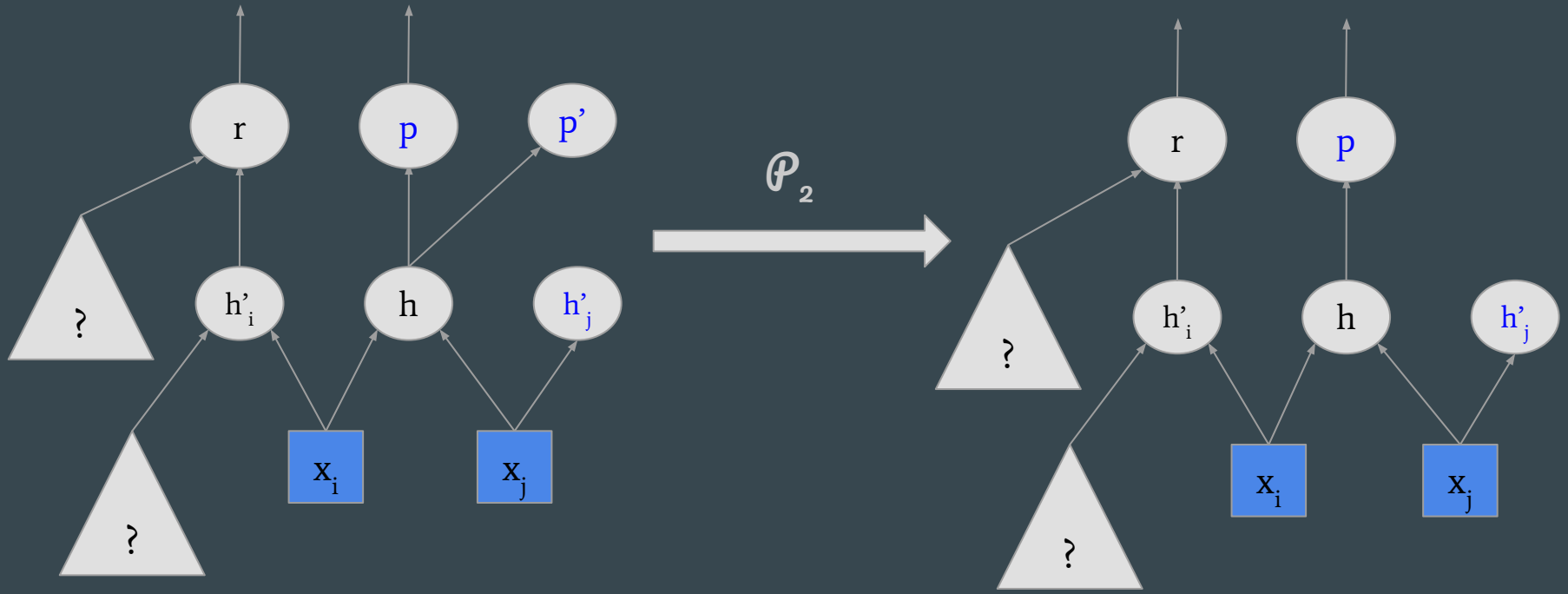


Visualizing all cases via a Decision Tree

- Each branch represents a distinct possibility for the identity of p, p', h'_j
- Each branch promises the existence of an assignment s.t. **4 gates are eliminated** .

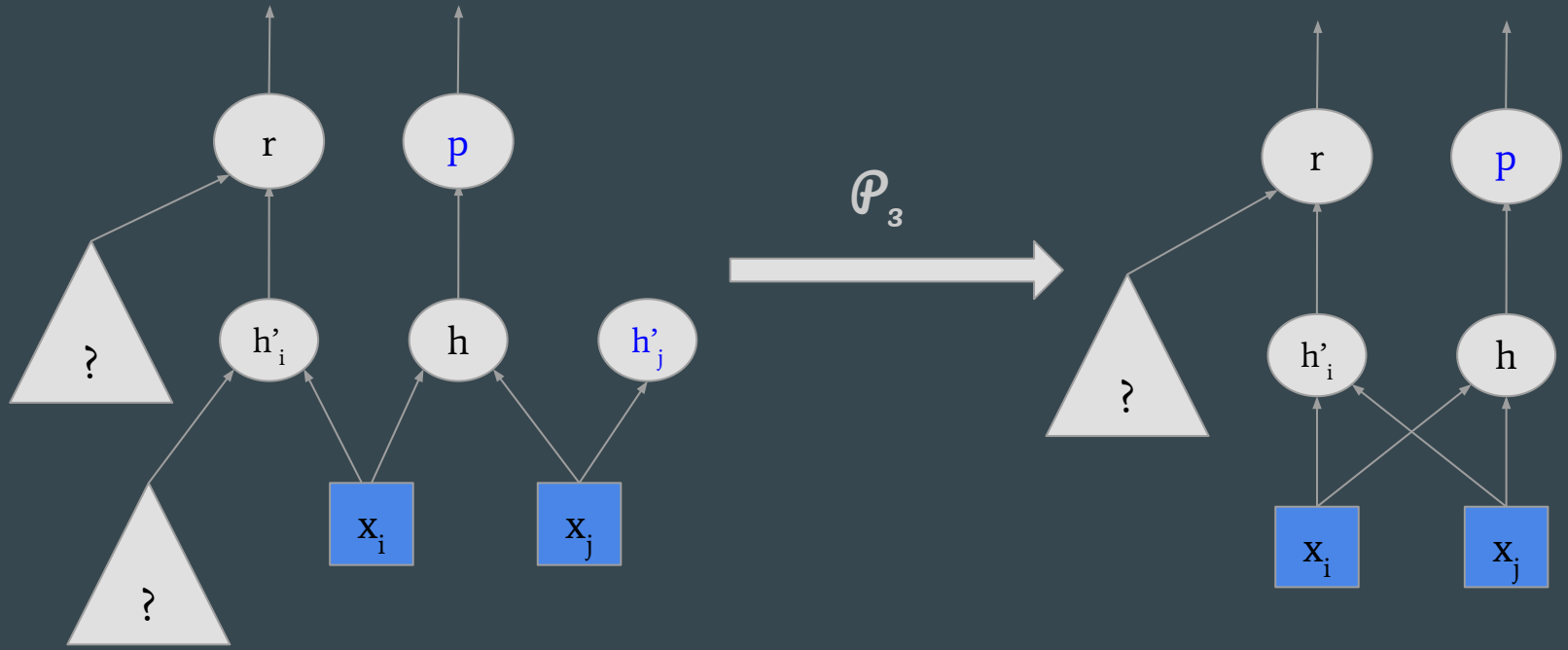
Fleshing out the circuit template

Stage 2: assume h is connected to p, show $\mathcal{P}_2$ = “p is the only costly successor of h”



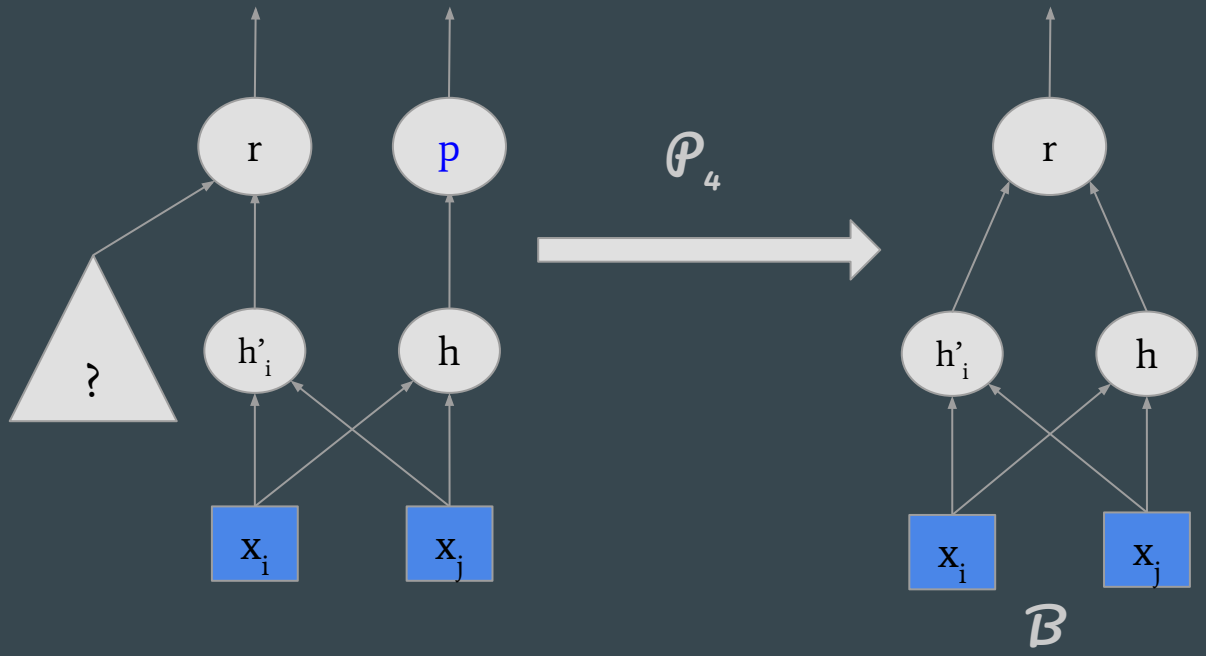
Fleshing out the circuit template

Stage 3: show $\mathcal{P}_3$ = "h'_j is the same as h'_i"



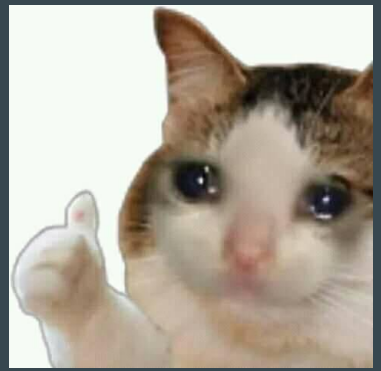
Fleshing out the circuit template

Stage 4: show $\mathcal{P}_4$ = "p is the same as r"



What remains:

- via GE, show $\mathcal{B}$ computes XOR_2
- use an inductive argument to complete the proof



Open Problems

- 2 opposite directions
- Identify non-optimal XOR-circuits?

Open Problems

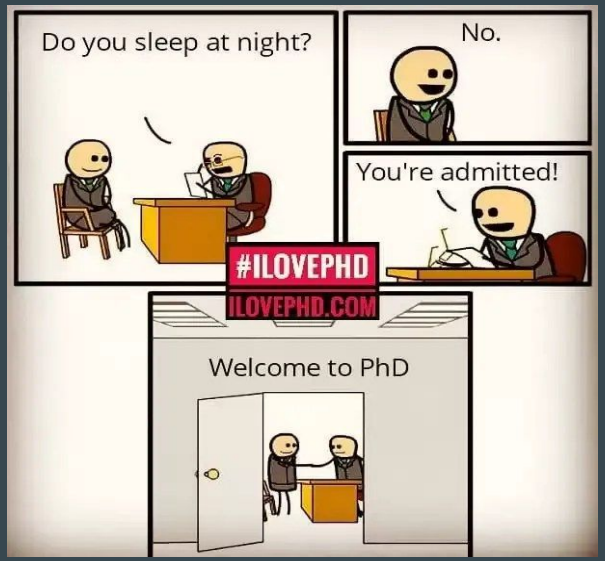
Our Main Theorem opens **two opposite directions** :

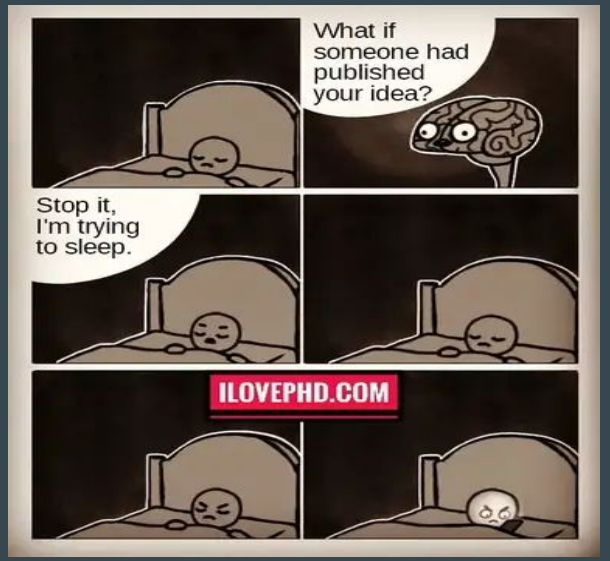
- **Optimistic approach:** one can attempt to adapt Ilango's Proof Framework for proving Hardness of MCSP* to extend the hardness result to MCSP.
- **Pessimistic approach:** one can try to develop a polytime solution solving the Simple Extension Problem for XOR which rules out the possibility of using Ilango's technique to show hardness of MCSP.

Can we characterize non-optimal circuits computing XOR?

Thank You!!!

Questions?





(source of memes in pics)

Bonus: Application

Optimal XOR-circuit Identity Testing

Optimal XOR-circuit Identity Testing

- Input: a *normalized* circuit C computing a Boolean function $f: \{0,1\}^n \to \{0,1\}$
- Question: is C an optimal circuit computing XOR_n?

Brute-force Solution: on input C

1. Count the number of costly gates in C. If not 3n - 3, **reject**.
2. For each $x \in \{0,1\}^n$
   a. Evaluate C(x)
   b. If $C(x) \neq XOR_n(x)$, **reject**
3. **Accept**

Running Time: $O(2^n)$

Optimal XOR-circuit Identity Testing

- Input: a *normalized* circuit C computing a Boolean function $f: \{0, 1\}^n \to \{0, 1\}$
- Question: is C an optimal circuit computing XOR_n?

Improved Solution: on input C

1. Count the number of costly gates in C. If not 3n - 3, **reject**.
2. Partition the sorted gates into n - 1 blocks as described in our Main Result
3. For each block B:
   a. Check if B computes $(\neg)XOR_2$, **reject** if not.
4. If $C(0^n) = 0$, **accept**. Otherwise, **reject**.

Optimal XOR-circuit Identity Testing

Improved Solution: on input C

1. Count the number of costly gates in C. If not 3n - 3, **reject**.
2. Partition the sorted gates into n - 1 blocks as described in our Main Result
3. For each block B:
   a. Check if B computes (¬)XOR_2, **reject** if not.
4. If $C(0^n) = 0$, **accept**. Otherwise, **reject**.

Running Time: $O(n)$ since C is normalized

Correctness:

- Step 1-3 follows from

<u>Claim:</u> let C be a circuit of size 3n - 3 and can be partition into n - 1 (¬)XOR_2-blocks, then C computes (¬)XOR_n

Proof idea: use strong induction

- Step 4 follows from the definition of XOR, i.e. $XOR_n(0^n) = 0$