CIRCUITS, LOWER BOUNDS, AND CIRCUIT ANALYSIS ALGORITHMS

Dimitrios Myrisiotis

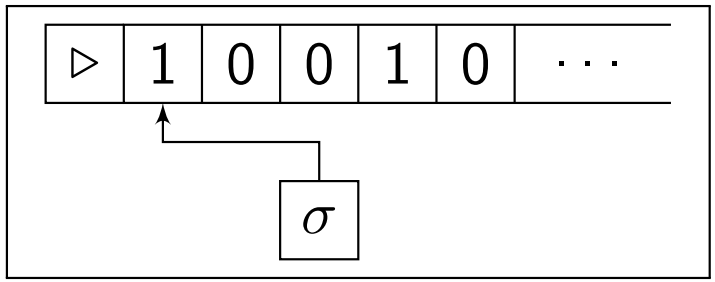
Dept. of Computing • Imperial College London (ICL)

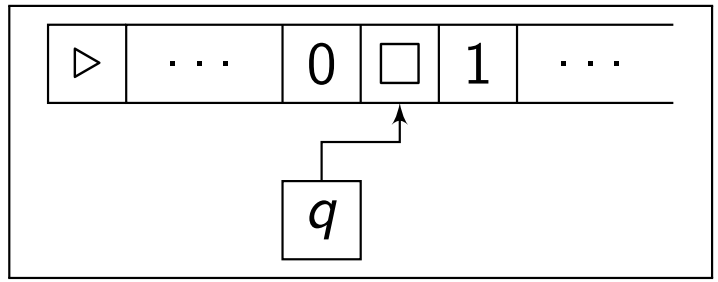
Joint work (in progress) with M. Cheraghchi (ICL), Z. Lu (SFU), and N. Rajgopal (Oxford).

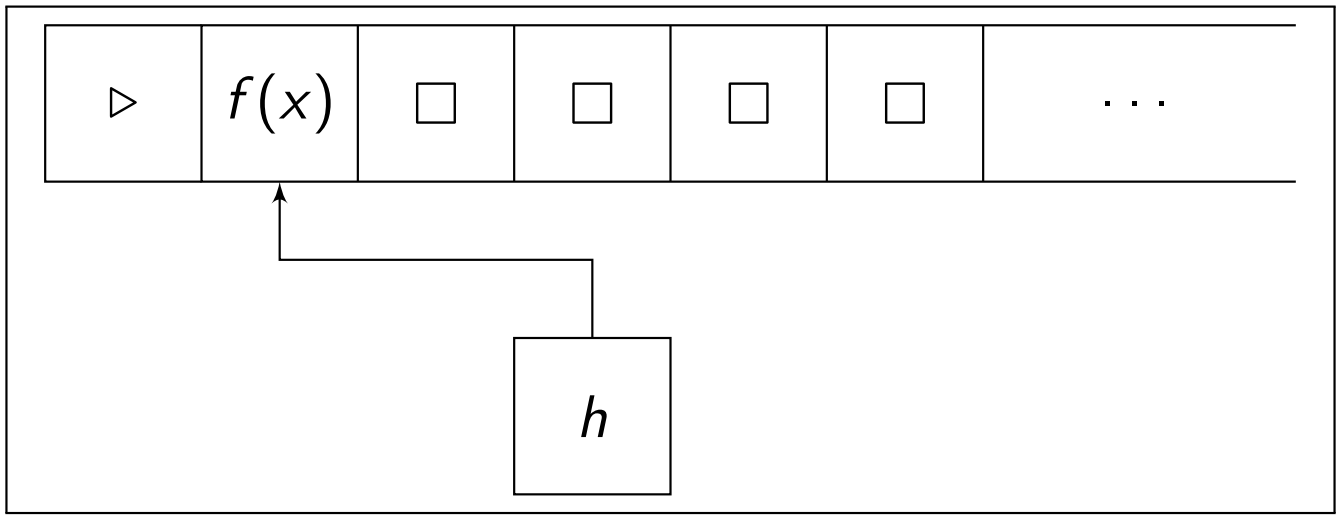
Table of Contents

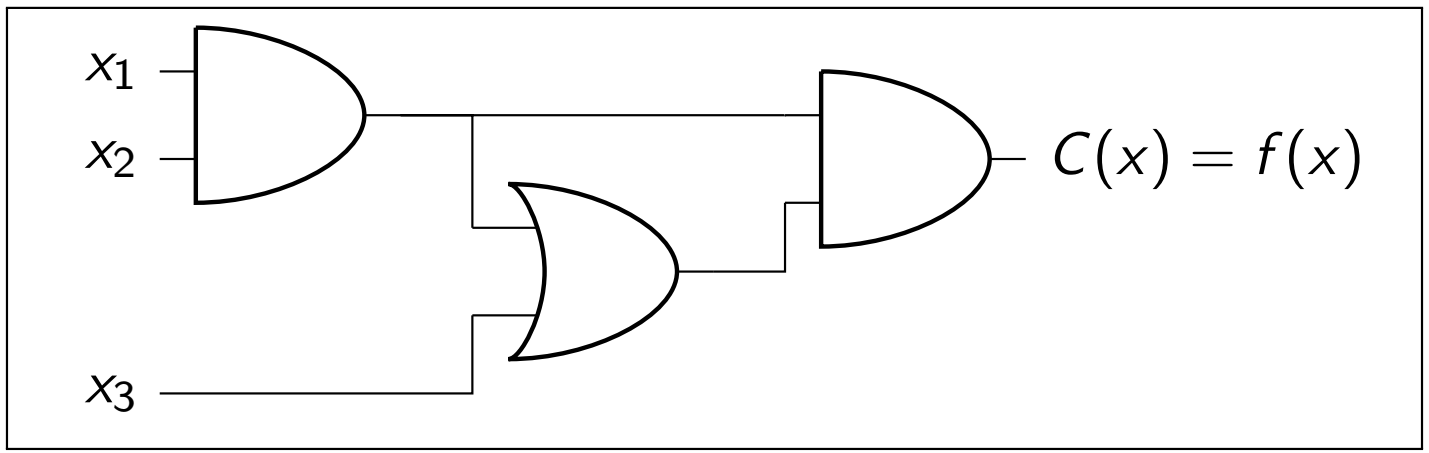
Introduction

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


▷
1
0
0
1
0
· · ·
σ


▷
· · ·
0
□
1
· · ·
q


▷
f(x)
□
□
□
□
· · ·
h


x_1
x_2
x_3
$C(x) = f(x)$

Definition

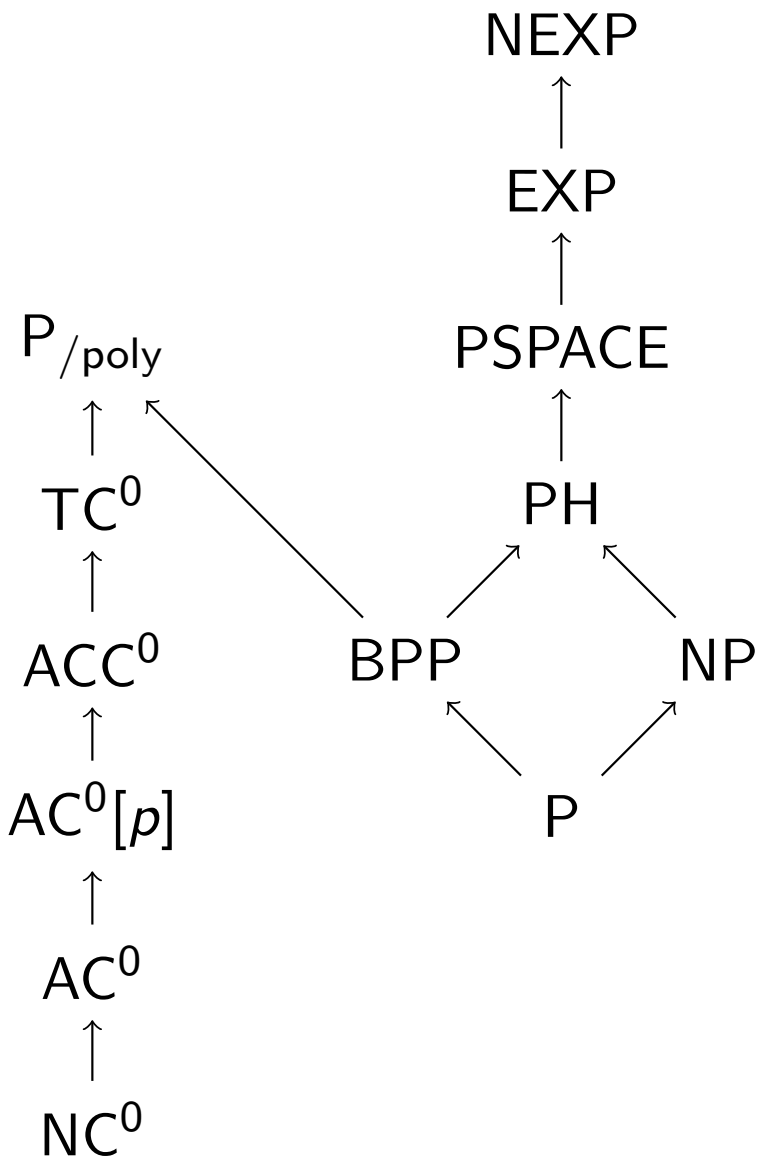
$\mathsf{P} = \bigcup_{k \in \mathbb{N}} \mathsf{TIME}(n^k)$.

Definition

$\mathsf{P}_{/\mathsf{poly}} = \bigcup_{k \in \mathbb{N}} \mathsf{SIZE}(n^k)$.

Lemma

$\mathsf{P} \subsetneq \mathsf{P}_{/\mathsf{poly}}$.


NEXP
EXP
PSPACE
PH
NP
P
BPP
$P_{/poly}$
TC^0
ACC^0
$AC^0[p]$
AC^0
NC^0

Circuit Lower Bounds

Question (OPEN)

$\mathsf{NP} \overset{?}{\subseteq} \mathsf{P}$.

Question (OPEN)

$\mathsf{NP} \overset{?}{\subseteq} \mathsf{P}_{/\mathsf{poly}}$.

Question

$\mathsf{NP} \stackrel{?}{\subseteq} \mathsf{NC}^0$.

Lemma

$\mathsf{NP} \not\subseteq \mathsf{NC}^0$.

Question

$\mathsf{NP} \stackrel{?}{\subseteq} \mathsf{AC}^0$.

Theorem ([FSS81, Ajt83, Yao85, Hås86])

$\oplus \notin \mathsf{AC}^0$.

Corollary

$\mathsf{NP} \not\subseteq \mathsf{AC}^0$.

Question

$\mathsf{NP} \stackrel{?}{\subseteq} \mathsf{AC}^0[p]$.

Theorem ([Raz87, Smo87])

$\mathtt{MOD}_q \notin \mathsf{AC}^0[p]$.

Corollary

$\mathsf{NP} \not\subseteq \mathsf{AC}^0[p]$.

Question (OPEN)

$\mathsf{NP} \stackrel{?}{\subseteq} \mathsf{ACC}^0$.

Theorem ([MW18])

$\mathsf{NQuasiP} \not\subseteq \mathsf{ACC}^0$.

Circuit Analysis Notions

Definition (Satisfiability)

Input : A Boolean circuit $C : \{0,1\}^n \to \{0,1\}$ from a circuit class $\mathfrak{C}$.

Question : Is there any $x \in \{0,1\}^n$ such that $C(x) = 1$?

Definition (Minimum Circuit Size Problem [KC00])

Input : The truth table $\mathtt{tt}(f) \in \{0,1\}^{2^n}$ of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ and an integer $1 \leq s \leq 2^n$.

Question : Is there any Boolean circuit $C : \{0,1\}^n \to \{0,1\}$ of size $|C| \leq s$ that computes f?

Definition (Learning [Val84, KV94])

Let $0 < \varepsilon, \delta < 1$.

Input : An oracle for some Boolean function $f : \{0,1\}^n \to \{0,1\}$ from a circuit class $\mathfrak{C}$.

Output : A circuit $h : \{0,1\}^n \to \{0,1\}$, such that

$$\Pr_{x \sim \{0,1\}^n}[h(x) \neq f(x)] \leq \varepsilon,$$

with probability at least $1 - \delta$.

Definition (Compression [CKK+14])

Input : The truth table $\mathtt{tt}(f) \in \{0,1\}^{2^n}$ of some Boolean function $f : \{0,1\}^n \to \{0,1\}$ from a circuit class $\mathfrak{C}$.

Output : A circuit C of size $o(2^n/n)$ that computes f.

Definition (Natural Properties [RR94])

Let

- $\mathcal{F}_n$ denote the set of all Boolean functions on n variables,
- D be some complexity class, and
- $s : \mathbb{N} \to \mathbb{N}$ is a size function.

A D-natural property useful against $\mathfrak{C}[s]$ is an algorithm $A : \{0,1\}^{2^n} \to \{0,1\}$ such that

1. A is computable in D,
2. if $f \in \mathfrak{C}[s]$, then $A(\mathtt{tt}(f)) = 0$, and
3. $\Pr_{g \sim \mathcal{F}_n}[A(\mathtt{tt}(g)) = 1] \geq \frac{1}{2}$.

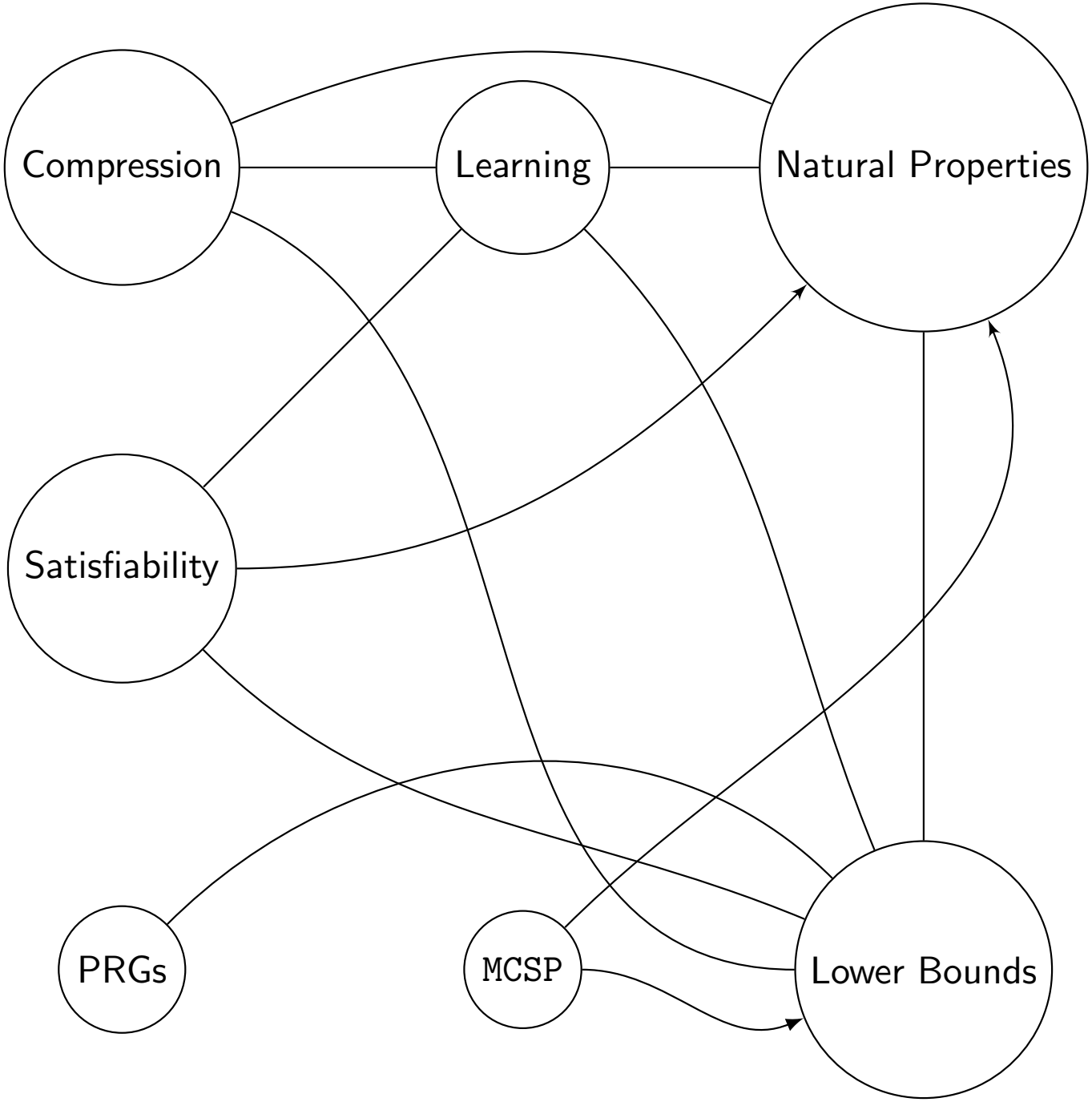
Definition (PRGs)

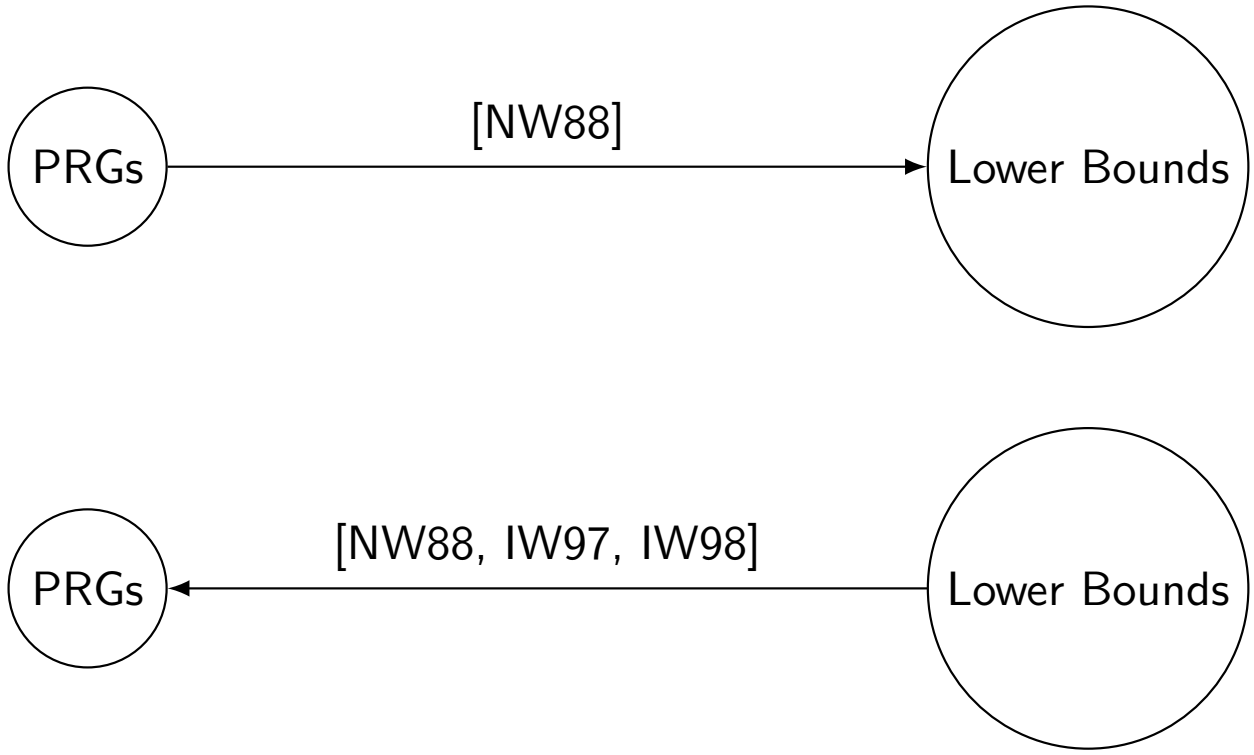
Let $\mathcal{F}$ be a class of functions on n variables and $\varepsilon > 0$. We say that $G : \{0,1\}^s \to \{0,1\}^n$ is a PRG that ε-fools $\mathcal{F}$ if

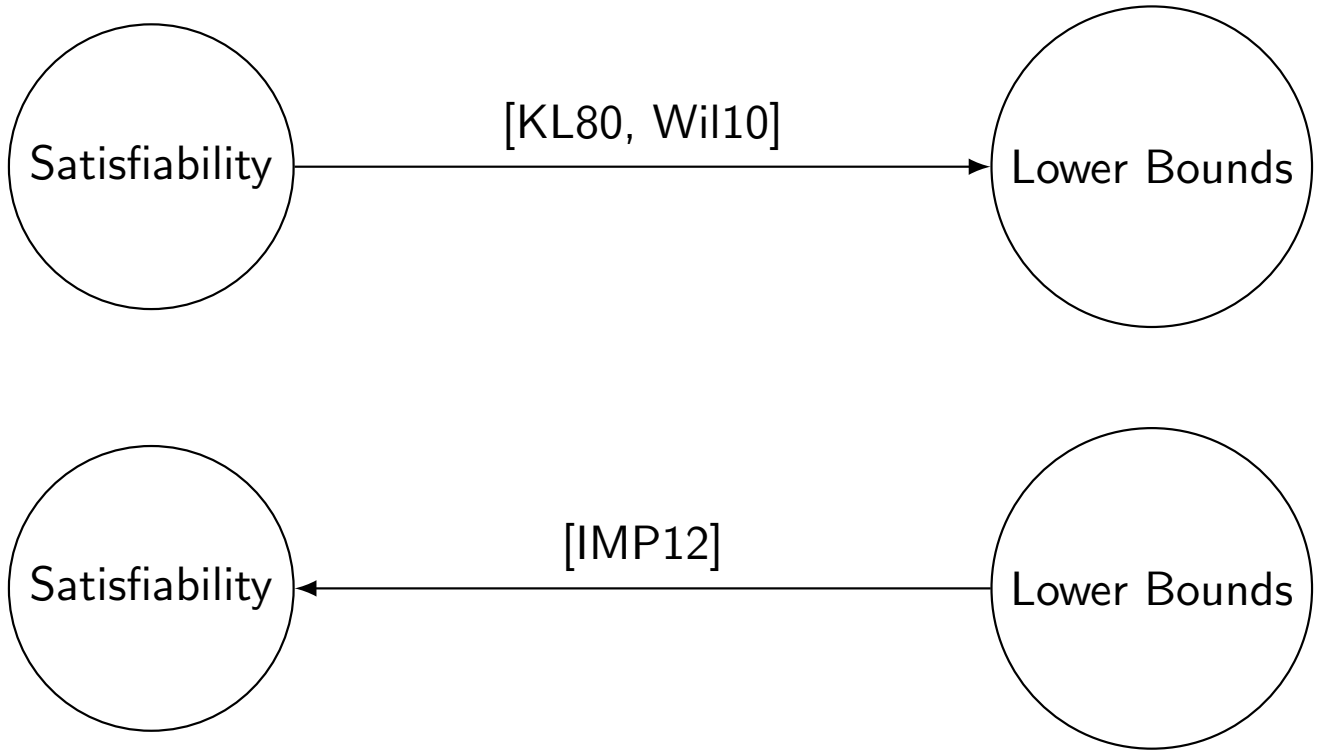
$$\left| \Pr_{y \sim \{0,1\}^s}[f(G(y)) = 1] - \Pr_{x \sim \{0,1\}^n}[f(x) = 1] \right| \leq \varepsilon,$$

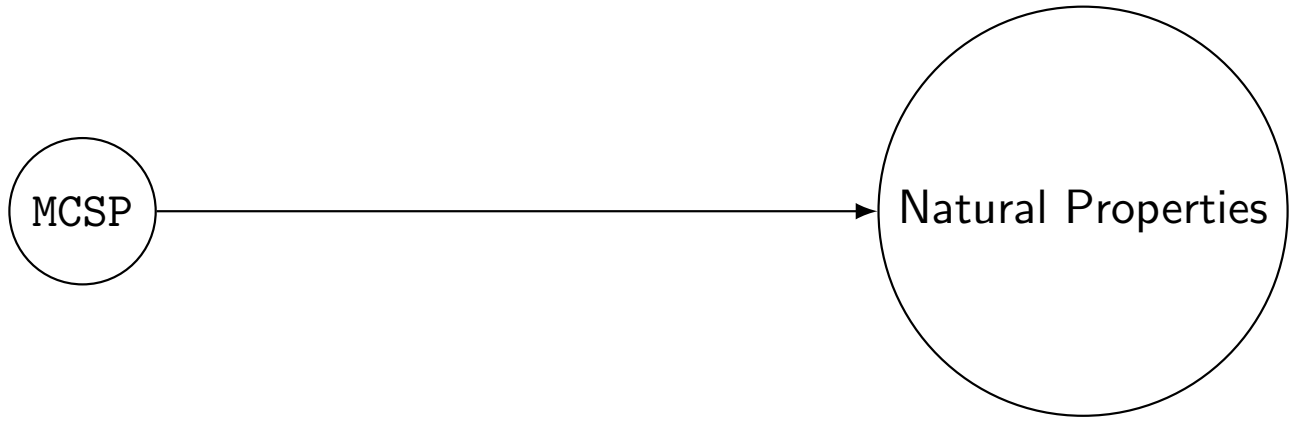
for all $f \in \mathcal{F}$.

Known Connections

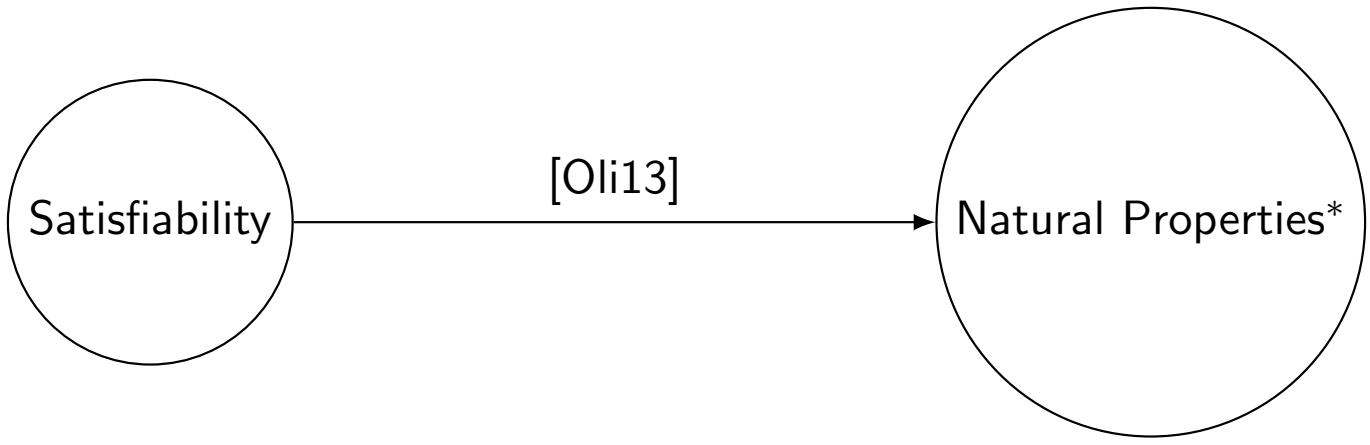

Compression
Learning
Natural Properties
Satisfiability
PRGs
MCSP
Lower Bounds

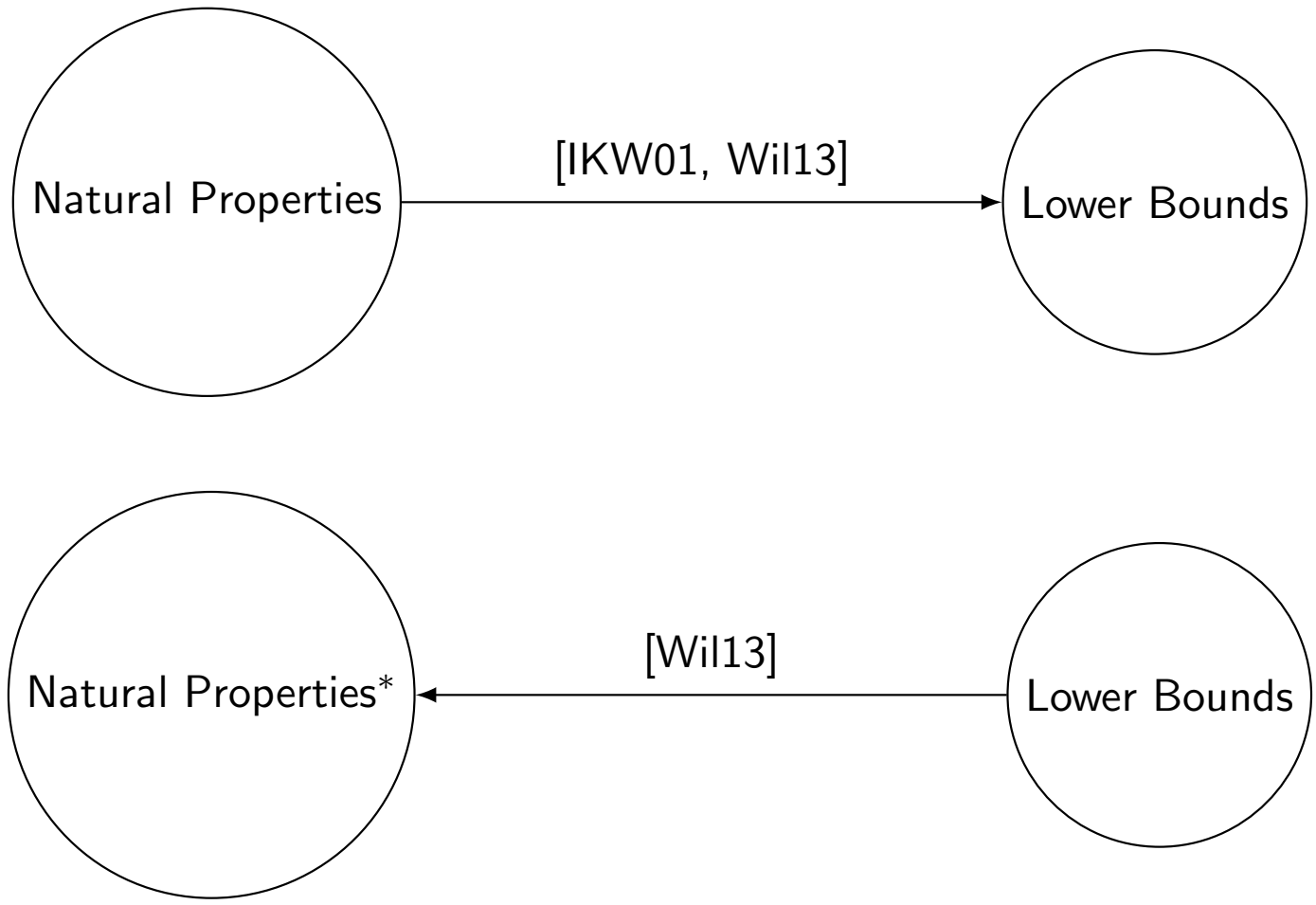

PRGs
[NW88]
Lower Bounds
PRGs
[NW88, IW97, IW98]
Lower Bounds

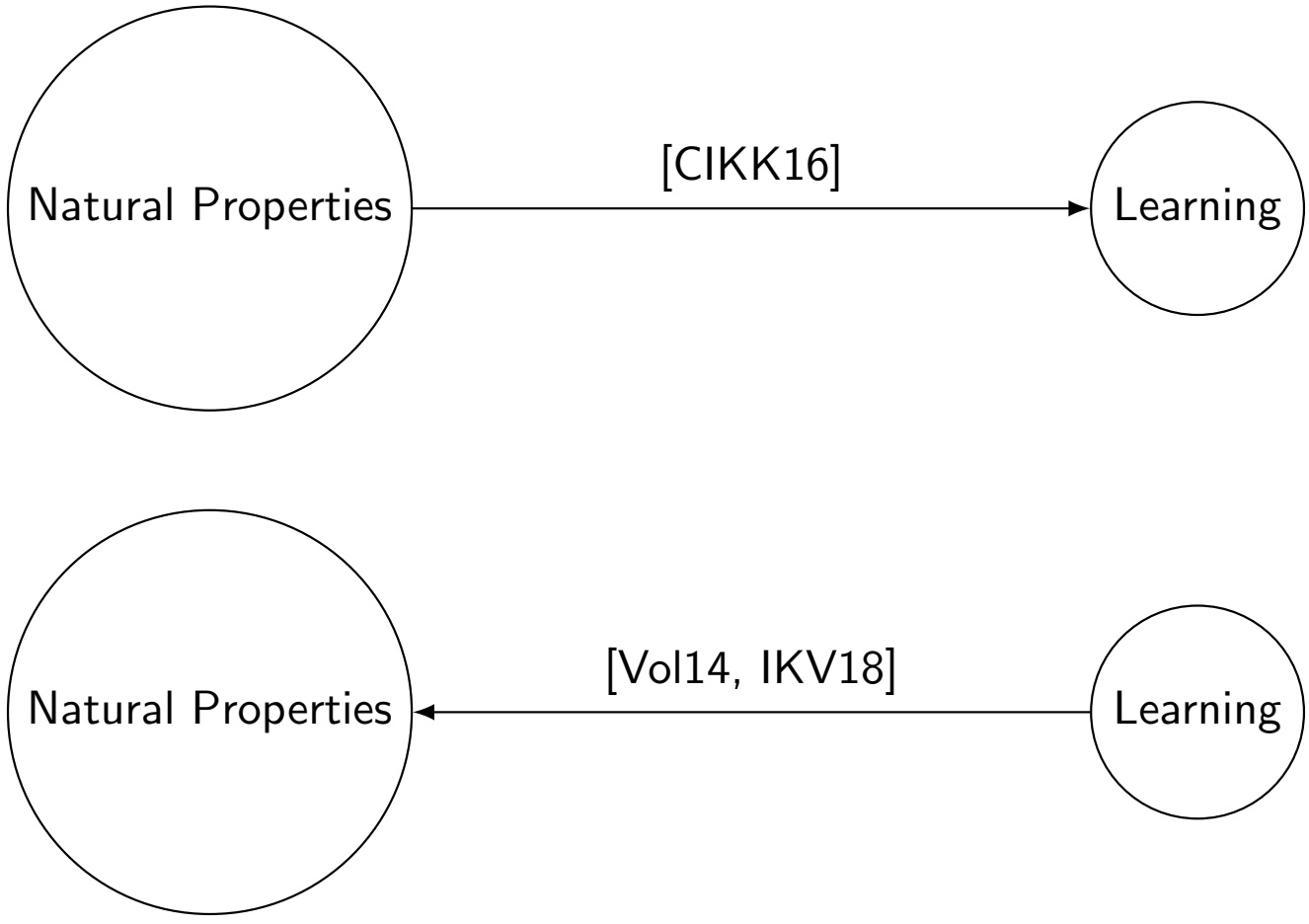

Satisfiability
[KL80, Wil10]
Lower Bounds
Satisfiability
[IMP12]
Lower Bounds

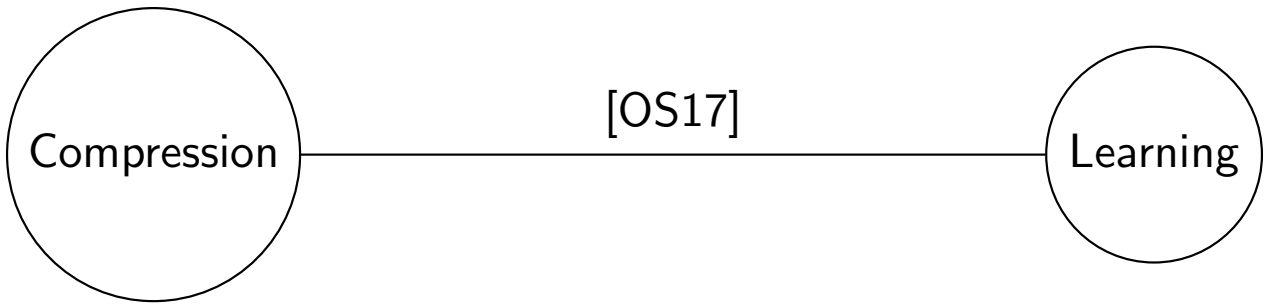

MCSP
Natural Properties

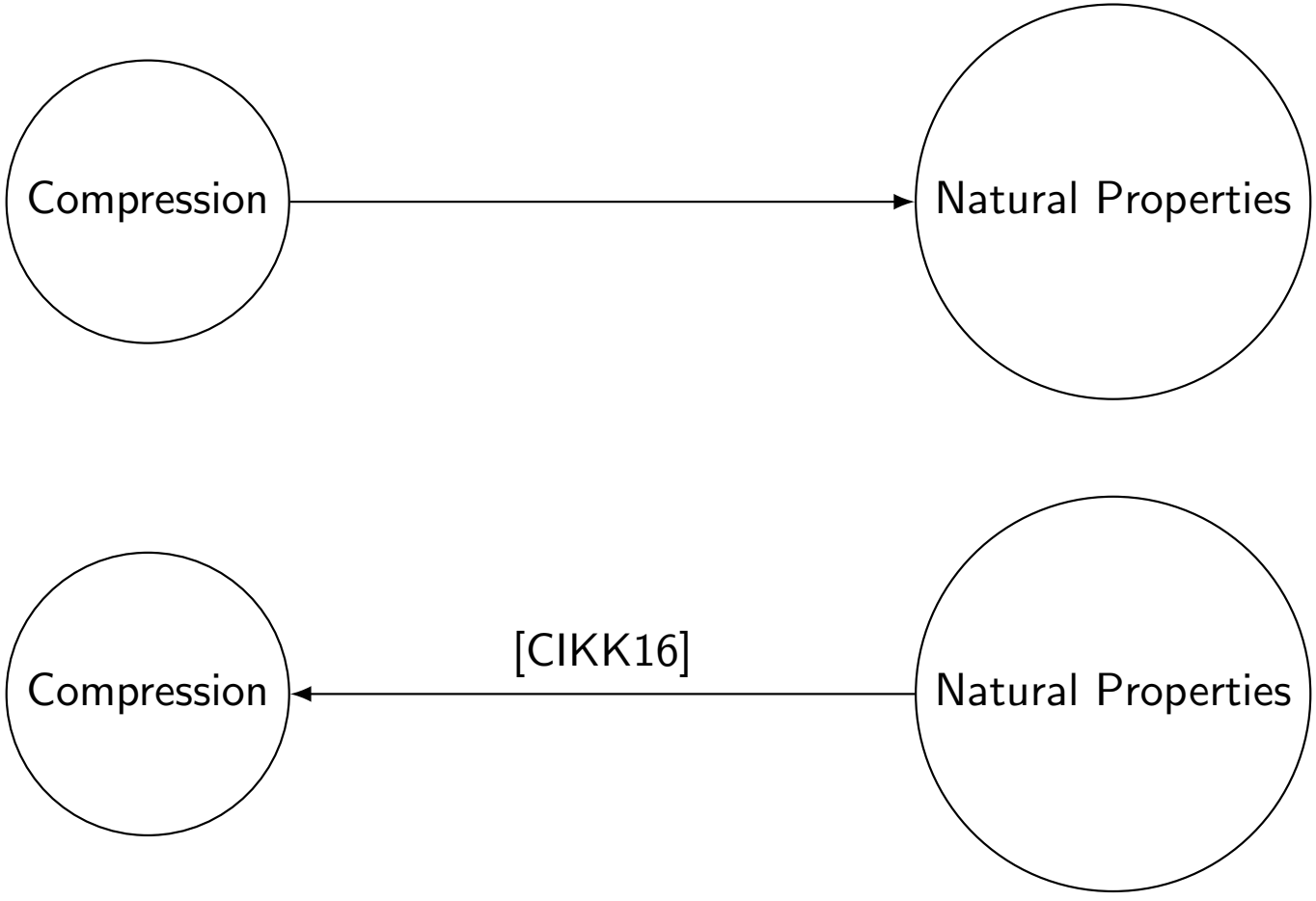

Natural Properties
Natural Properties*

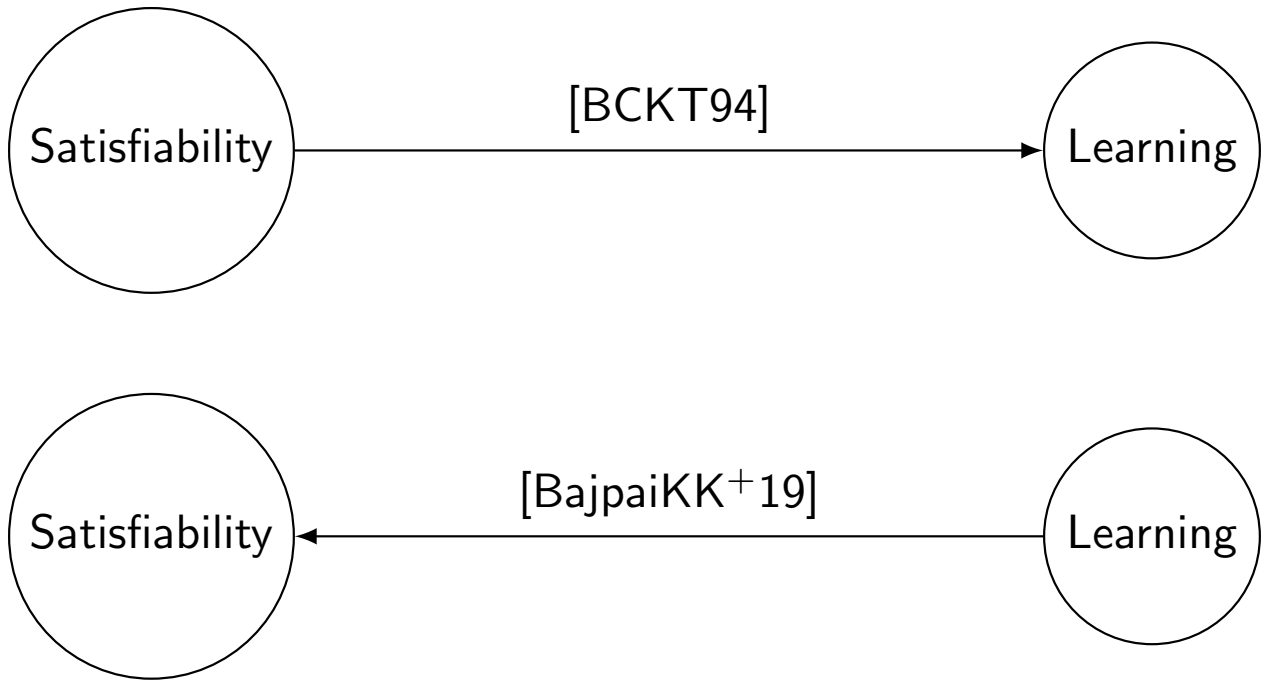

Satisfiability
[Oli13]
Natural Properties*

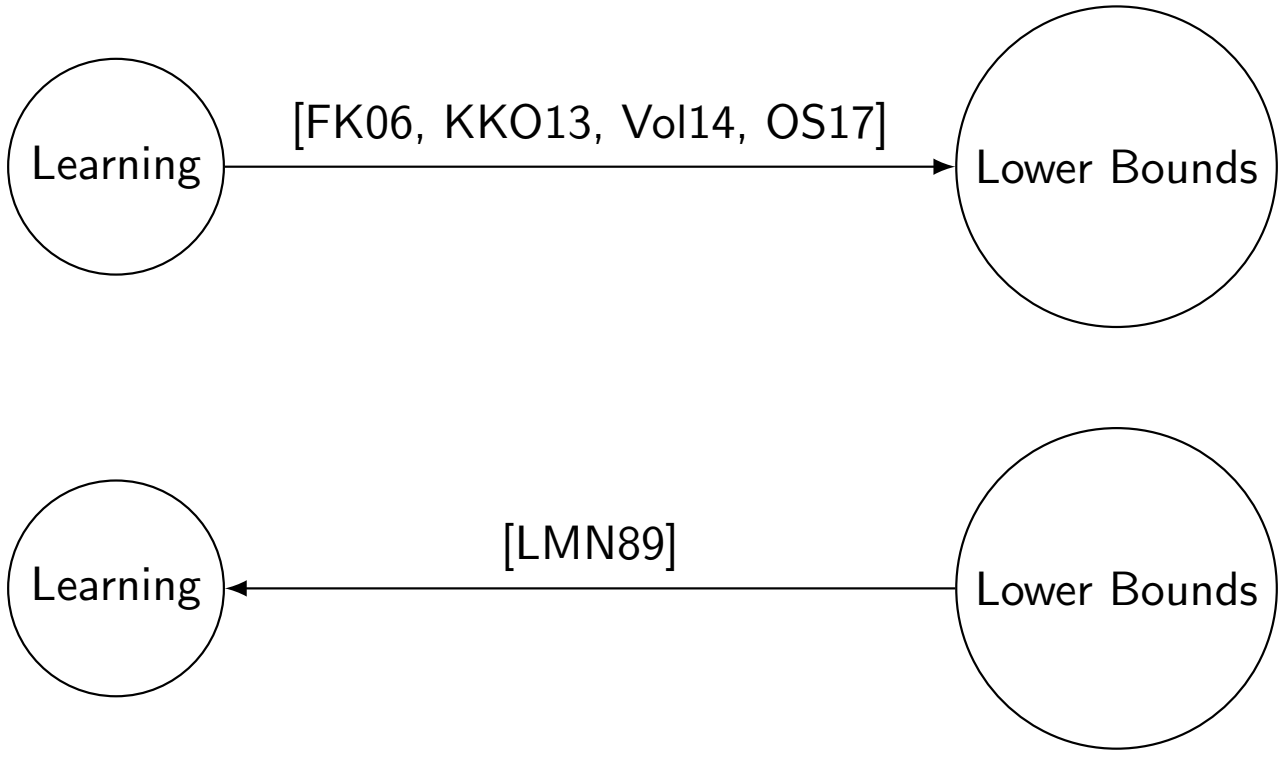

Natural Properties
[IKW01, Wil13]
Lower Bounds
Natural Properties*
[Wil13]
Lower Bounds

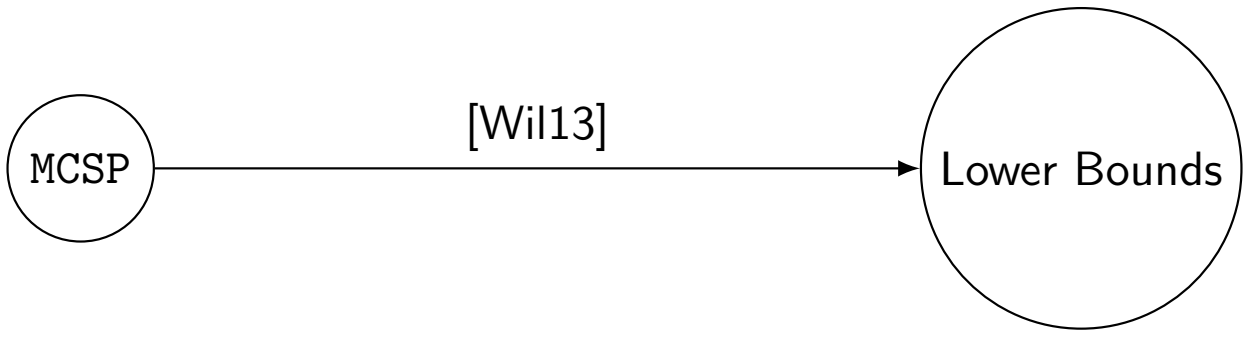

Natural Properties
[CIKK16]
Learning
Natural Properties
[Vol14, IKV18]
Learning

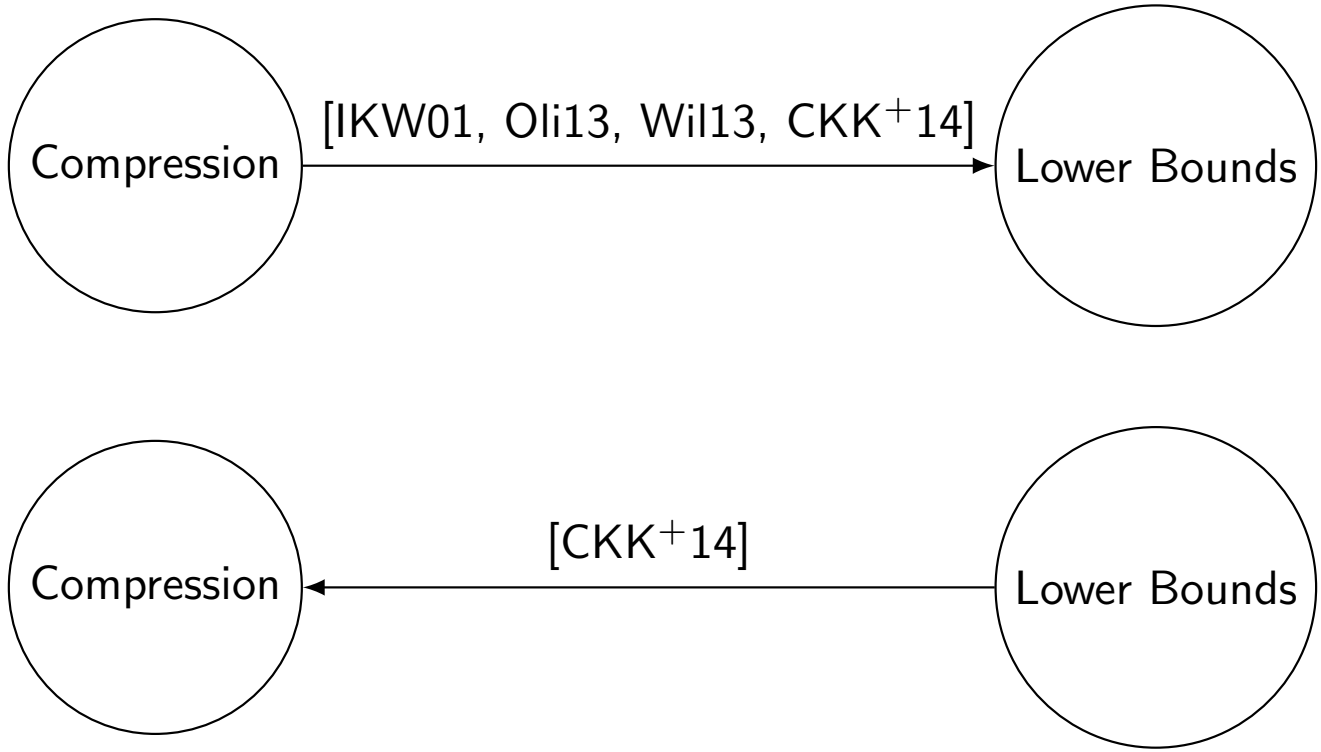

Compression
[OS17]
Learning


Compression
Natural Properties
Compression
[CIKK16]
Natural Properties

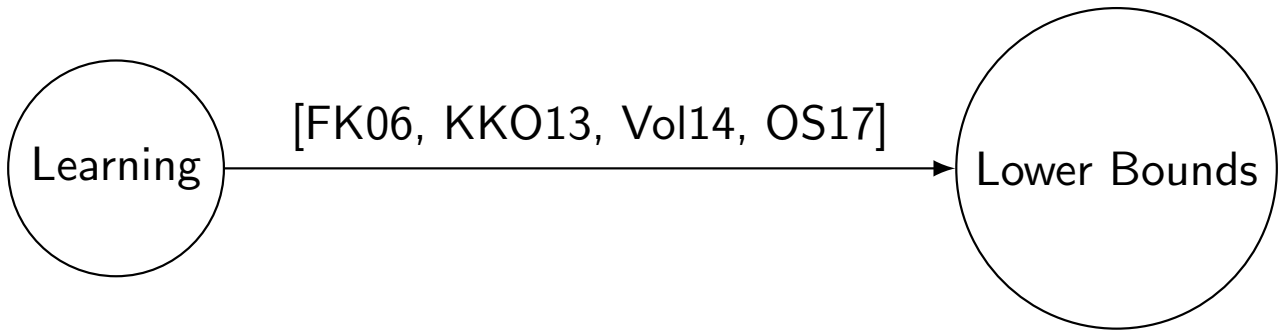

Satisfiability
[BCKT94]
Learning
Satisfiability
[BajpaiKK+19]
Learning

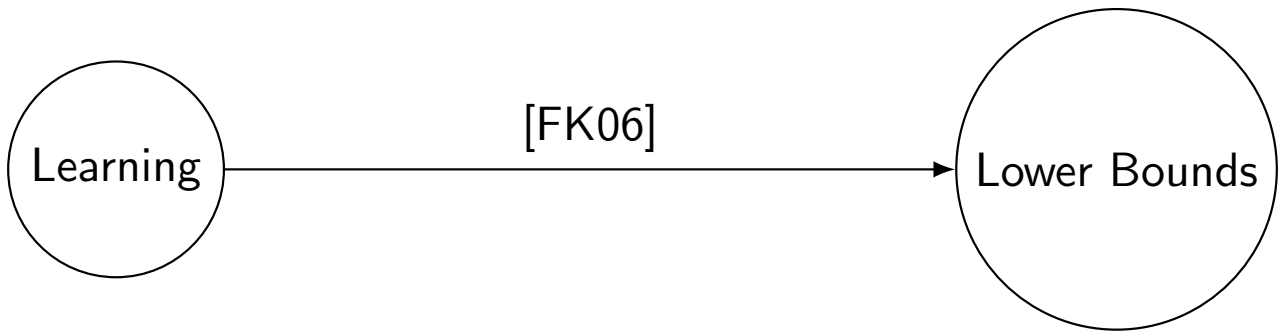

Learning
[FK06, KKO13, Vol14, OS17]
Lower Bounds
Learning
[LMN89]
Lower Bounds


MCSP
[Wil13]
Lower Bounds


Compression
[IKW01, Oli13, Wil13, CKK+14]
Lower Bounds
Compression
[CKK+14]
Lower Bounds

Example: Learning to Lower Bounds


Learning
[FK06, KKO13, Vol14, OS17]
Lower Bounds


Learning
[FK06]
Lower Bounds

Theorem ([FK06])

Let $s : \mathbb{N} \to \mathbb{N}$ *be a (polynomially-bounded) size function.*

Let $\mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$ *be a circuit class exactly learnable in time* $t := 2^{s^{o(1)}}$ *from membership and equivalence queries.*

Then, $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathfrak{C}[s]$.

Note: The learning algorithm runs in subexponential time.

Definition

Let $f \in \mathfrak{C}[s]$ with $f : \{0,1\}^n \to \{0,1\}$.

$$\forall x \in \{0,1\}^n : \mathtt{MQ}(x) = f(x).$$

Definition

Let $f \in \mathfrak{C}[s]$ with $f : \{0,1\}^n \to \{0,1\}$.

For all hypotheses (circuits) $h : \{0,1\}^n \to \{0,1\}$ we have that

$$\mathtt{EQ}(h) = \begin{cases} 1, & \text{if } h(x) = f(x) \text{ for all } x \in \{0,1\}^n, \\ y : h(y) \neq f(y), & \text{otherwise.} \end{cases}$$

Theorem ([FK06])

Let $s : \mathbb{N} \to \mathbb{N}$ *be a (polynomially-bounded) size function.*

Let $\mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$ *be a circuit class exactly learnable in time* $t := 2^{s^{o(1)}}$ *from membership and equivalence queries.*

Then, $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathfrak{C}[s]$.

Note: The learning algorithm runs in subexponential time.

Proof.

1. Assume that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$. $(*)$
2. This implies $\mathsf{EXP}^{\mathsf{NP}} = \mathsf{P}^{\#\mathsf{P}}$.
3. Thus $\mathtt{PERM}$ is complete for $\mathsf{EXP}^{\mathsf{NP}}$.
4. Now use the (time-t) learning algorithm to show

$$\mathtt{PERM} \in \mathsf{SUBEXP}^{\mathsf{NP}} := \mathsf{TIME}\left(2^{n^{o(1)}}\right)^{\mathsf{NP}}.$$

5. This yields $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{SUBEXP}^{\mathsf{NP}}$. A contradiction!

Proof.

1. Assume that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$. $(*)$
2. **This implies** $\mathsf{EXP}^{\mathsf{NP}} = \mathsf{P}^{\#\mathsf{P}}$**.**
3. Thus $\mathtt{PERM}$ is complete for $\mathsf{EXP}^{\mathsf{NP}}$.
4. Now use the (time-t) learning algorithm to show

$$\mathtt{PERM} \in \mathsf{SUBEXP}^{\mathsf{NP}} := \mathsf{TIME}\left(2^{n^{o(1)}}\right)^{\mathsf{NP}}.$$

5. This yields $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{SUBEXP}^{\mathsf{NP}}$. A contradiction!

Proof (continued).

Assume that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$.

This implies $\mathsf{EXP}^{\mathsf{NP}} = \mathsf{P}^{\#\mathsf{P}}$, as

$$\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}_{/\mathsf{poly}} \overset{[\mathsf{BH92}]}{\Longrightarrow} \mathsf{EXP}^{\mathsf{NP}} = \mathsf{EXP},$$

$$\mathsf{EXP} \subseteq \mathsf{P}_{/\mathsf{poly}} \overset{[\mathsf{BN91}]}{\Longrightarrow} \mathsf{EXP} = \mathsf{MA},$$

and

$$\mathsf{MA} \subseteq \mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}}.$$

Proof (continued).

1. Assume that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$. $(*)$
2. This implies $\mathsf{EXP}^{\mathsf{NP}} = \mathsf{P}^{\#\mathsf{P}}$.
3. Thus $\mathtt{PERM}$ is complete for $\mathsf{EXP}^{\mathsf{NP}}$.
4. Now use the (time-t) learning algorithm to show

$$\mathtt{PERM} \in \mathsf{SUBEXP}^{\mathsf{NP}} := \mathsf{TIME}\left(2^{n^{o(1)}}\right)^{\mathsf{NP}}.$$

5. This yields $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{SUBEXP}^{\mathsf{NP}}$. A contradiction!

Proof (continued).

1. Assume that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$. $(*)$
2. This implies $\mathsf{EXP}^{\mathsf{NP}} = \mathsf{P}^{\#\mathsf{P}}$.
3. Thus $\mathtt{PERM}$ is complete for $\mathsf{EXP}^{\mathsf{NP}}$.
4. **Now use the (time-t) learning algorithm to show**

$$\mathtt{PERM} \in \mathsf{SUBEXP}^{\mathsf{NP}} := \mathsf{TIME}\left(2^{n^{o(1)}}\right)^{\mathsf{NP}}.$$

5. This yields $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{SUBEXP}^{\mathsf{NP}}$. A contradiction!

Proof (continued).

We will use the (time-t) learning algorithm to show

$$\texttt{PERM} \in \mathsf{SUBEXP}^{\mathsf{NP}},$$

by induction on i.

Here, i refers to the input matrices size, namely $i \times i$.

Proof (continued).

Input: x; a $n \times n$ Boolean matrix.

For $i \leftarrow 1$ to n:

a. If $i = 1$, then output the trivial circuit for $\texttt{PERM}$ on 1×1 matrices; else

b. Run the (time-t) exact learning algorithm to find C_i, the circuit that computes $\texttt{PERM}$ on $i \times i$ matrices.

c. Simulate $\texttt{MQ}$ and $\texttt{EQ}$ using C_{i-1} and an NP oracle.

Output: $C_n(x) = \texttt{PERM}(x)$.

Proof (continued).

Input: x; a $n \times n$ Boolean matrix.

For $i \leftarrow 1$ to n:

a. If $i = 1$, then output the trivial circuit for $\texttt{PERM}$ on 1×1 matrices; else

b. Run the (time-t) exact learning algorithm to find C_i, the circuit that computes $\texttt{PERM}$ on $i \times i$ matrices.

c. **Simulate $\texttt{MQ}$ and $\texttt{EQ}$ using C_{i-1} and an NP oracle.**

Output: $C_n(x) = \texttt{PERM}(x)$.

Proof (continued).

Simulate $\texttt{MQ}(y) = \texttt{PERM}(y)$, with $|y| = i$, using C_{i-1}.

Self-reduction: $\texttt{PERM}(y)$ can be computed by i calls to C_{i-1}.

This runs in polynomial (in n) time as $i \leq n$ and C_{i-1} is of polynomial (in n) size.

Proof (continued).

Simulate $\texttt{EQ}(h)$, with $h : \{0,1\}^i \to \{0,1\}$, using C_{i-1} and an NP oracle:

i. Ask the NP oracle:

> *"Does there exist a* $z \in \{0,1\}^i$ *such that* $h(z) \neq \texttt{PERM}(z)$*?"*

ii. If NO, then $h(x) = \texttt{PERM}(x)$ for all $x \in \{0,1\}^i$.

iii. If YES, then find z. Ask the NP oracle:

> *"Does there exist a* $z \in 0\{0,1\}^{i-1}$ *such that* $h(z) \neq \texttt{PERM}(z)$*?"*

Repeat until all i bits of z are computed.

This runs in polynomial (in n) time as $i \leq n$ and h and C_{i-1} are of polynomial (in n) size.

Proof (continued).

Input: x; a $n \times n$ Boolean matrix.

For $i \leftarrow 1$ to n:

a. If $i = 1$, then output the trivial circuit for $\texttt{PERM}$ on 1×1 matrices; else

b. Run the (time-t) exact learning algorithm to find C_i, the circuit that computes $\texttt{PERM}$ on $i \times i$ matrices.

c. Simulate $\texttt{MQ}$ and $\texttt{EQ}$ using C_{i-1} and an NP oracle.

Output: $C_n(x) = \texttt{PERM}(x)$.

Proof (continued).

What is the running time?

The running time of the previous procedure is

$$\mathrm{poly}(n, t) = \mathrm{poly}\left(n, 2^{s^{o(1)}}\right).$$

Note that $s = \mathrm{poly}(n)$; this yields $\texttt{PERM} \in \mathrm{SUBEXP}^{\mathrm{NP}}$.

Proof.

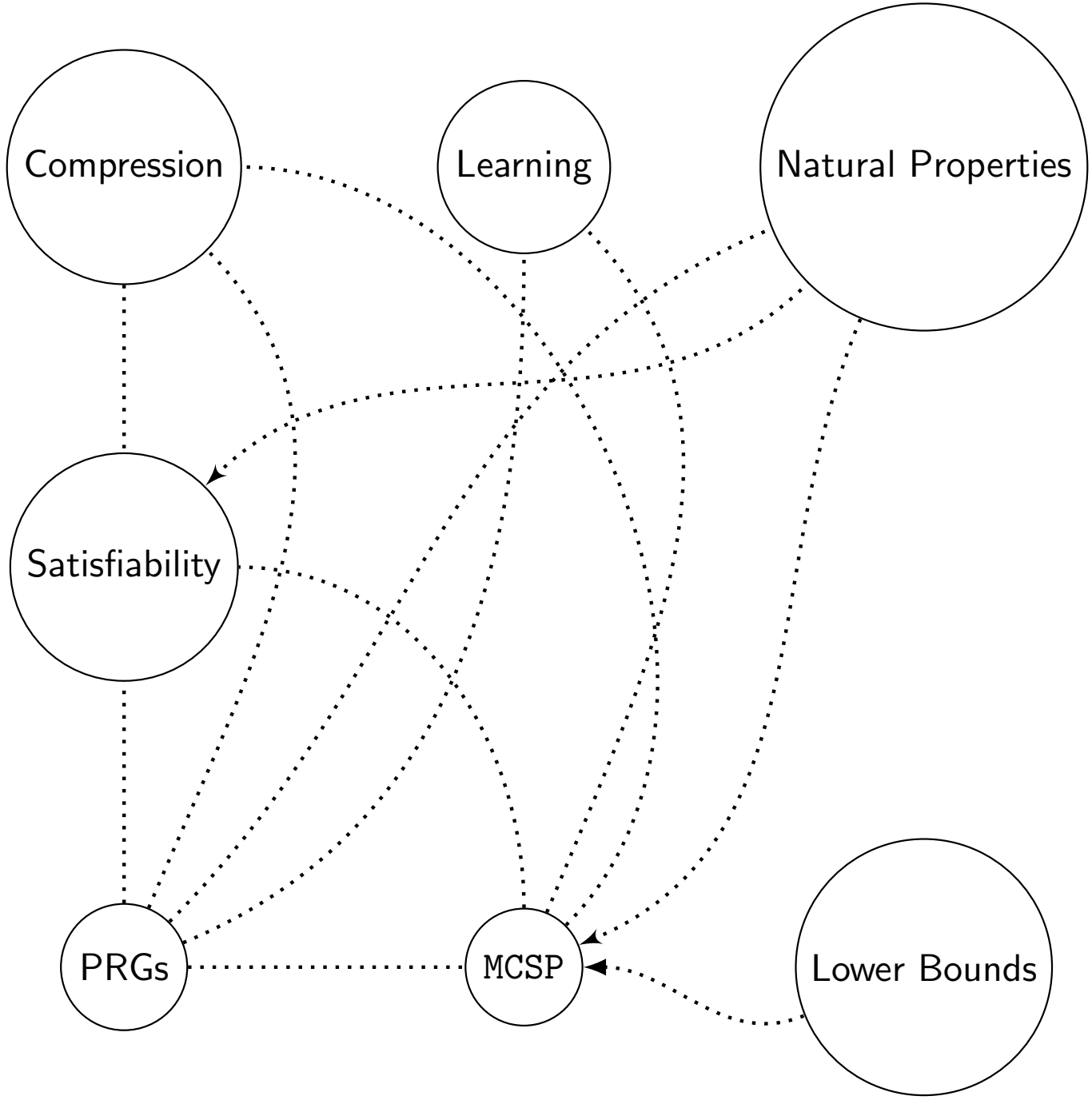
1. Assume that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathfrak{C}[s] \subseteq \mathsf{P}_{/\mathsf{poly}}$. $(*)$
2. This implies $\mathsf{EXP}^{\mathsf{NP}} = \mathsf{P}^{\#\mathsf{P}}$.
3. Thus $\mathtt{PERM}$ is complete for $\mathsf{EXP}^{\mathsf{NP}}$.
4. Now use the (time-t) learning algorithm to show

$$\mathtt{PERM} \in \mathsf{SUBEXP}^{\mathsf{NP}} := \mathsf{TIME}\left(2^{n^{o(1)}}\right)^{\mathsf{NP}}.$$

5. This yields $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{SUBEXP}^{\mathsf{NP}}$. A contradiction!

The proof is complete. □

Discussion: OPEN Problems


Compression
Learning
Natural Properties
Satisfiability
PRGs
MCSP
Lower Bounds

Acknowledgements I

Many thanks to Igor Oliveira (Oxford) and Ryan Williams (MIT).

Acknowledgements II

Many thanks to Valentine Kabanets (SFU), Rahul Santhanam (Oxford), and Igor Oliveira (Oxford).

Thank you!

References

[Ajt83] M. Ajtai.
Σ_1^1-Formulae on Finite Structures.
Annals of Pure and Applied Logic, 24(1):1–48, 1983.

[BCKT94] Nader H. Bshouty, Richard Cleve, Sampath Kannan, and Christino Tamon.
Oracles and queries that are sufficient for exact learning (extended abstract).
In *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory, COLT 1994, New Brunswick, NJ, USA, July 12-15, 1994.*, pages 130–139, 1994.

References

[BH92] Harry Buhrman and Steven Homer.
Superpolynomial Circuits, Almost Sparse Oracles and the Exponential Hierarchy.
In *FSTTCS*, volume 652 of *Lecture Notes in Computer Science*, pages 116–127. Springer, 1992.

[BN91] László Babai and Noam Nisan.
BPP has Subexponential Time Simulation unless EXPTIME has Pubishable Proofs.
In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*, pages 213–219, 1991.

References

[CIKK16] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova.
Learning Algorithms from Natural Proofs.
In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 10:1–10:24, 2016.

[CKK+14] Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman.
Mining Circuit Lower Bound Proofs for Meta-algorithms.

In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 262–273, 2014.

References

[FK06] Lance Fortnow and Adam R. Klivans.
Efficient Learning Algorithms Yield Circuit Lower Bounds.
In *Learning Theory, 19th Annual Conference on Learning Theory, COLT 2006, Pittsburgh, PA, USA, June 22-25, 2006, Proceedings*, pages 350–363, 2006.

[FSS81] Merrick L. Furst, James B. Saxe, and Michael Sipser.
Parity, Circuits, and the Polynomial-Time Hierarchy.
In *22nd Annual Symposium on Foundations of Computer Science, Nashville, Tennessee, USA, 28-30 October 1981*, pages 260–270, 1981.

References

[Hås86] J. Håstad.
Almost Optimal Lower Bounds for Small Depth Circuits.
In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 6–20, New York, NY, USA, 1986. ACM.

[IKV18] Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich.
The Power of Natural Properties as Oracles.
In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 7:1–7:20, 2018.

References

[IKW01] Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson.
In Search of an Easy Witness: Exponential Time vs. Probabilistic Polynomial Time.
In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity, Chicago, Illinois, USA, June 18-21, 2001*, pages 2–12, 2001.

[IMP12] Russell Impagliazzo, William Matthews, and Ramamohan Paturi.
A Satisfiability Algorithm for AC^0.
In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 961–972, 2012.

References

[IW97] Russell Impagliazzo and Avi Wigderson.
P = BPP if E Requires Exponential Circuits: Derandomizing the XOR Lemma.
In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 220–229, 1997.

[IW98] Russell Impagliazzo and Avi Wigderson.
Randomness vs. Time: De-Randomization under a Uniform Assumption.
In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 734–743, 1998.

References

[KC00] Valentine Kabanets and Jin-Yi Cai.
Circuit Minimization Problem.
In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 73–79, 2000.

[KKO13] Adam R. Klivans, Pravesh Kothari, and Igor Carboni Oliveira.
Constructing Hard Functions Using Learning Algorithms.

In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 86–97, 2013.

References

[KL80] Richard M. Karp and Richard J. Lipton.
Some Connections Between Nonuniform and Uniform Complexity Classes.
In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing, April 28-30, 1980, Los Angeles, California, USA*, pages 302–309, 1980.

[KV94] Michael J. Kearns and Umesh V. Vazirani.
An Introduction to Computational Learning Theory.
MIT Press, 1994.

References

[LMN89] Nathan Linial, Yishay Mansour, and Noam Nisan.
Constant Depth Circuits, Fourier Transform, and Learnability.
In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 574–579, 1989.

[MW18] Cody Murray and R. Ryan Williams.
Circuit Lower Bounds for Nondeterministic Quasi-Polytime: An Easy Witness Lemma for NP and NQP.
In *STOC*, pages 890–901. ACM, 2018.

References

[NW88] Noam Nisan and Avi Wigderson.
Hardness vs. Randomness (Extended Abstract).
In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 2–11, 1988.

[Oli13] Igor Carboni Oliveira.
Algorithms versus Circuit Lower Bounds.
CoRR, abs/1309.0249, 2013.

[OS17] Igor Carboni Oliveira and Rahul Santhanam.
Conspiracies Between Learning Algorithms, Circuit Lower Bounds, and Pseudorandomness.

References

In *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, pages 18:1–18:49, 2017.

[Raz87] Alexander Razborov.
Lower Bounds on the Size of Bounded Depth Circuits Over a Complete Basis with Logical Addition.
Mat. Zametki, 41(4):598–607, 1987.
English translation in Mathematical Notes of the Academy of Sci. of the USSR, 41(4):333-338, 1987.

[RR94] Alexander A. Razborov and Steven Rudich.
Natural Proofs.
In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 204–213, New York, NY, USA, 1994. ACM.

References

[Smo87] Roman Smolensky.
Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity.
In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 77–82, 1987.

[Val84] Leslie G. Valiant.
A Theory of the Learnable.
In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 436–445, 1984.

References

[Vol14] Ilya Volkovich.
On Learning, Lower Bounds and (un)Keeping Promises.
In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 1027–1038, 2014.

[Wil10] Ryan Williams.
Improving Exhaustive Search Implies Superpolynomial Lower Bounds.
In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 231–240, 2010.

References

[Wil13] Ryan Williams.
Natural Proofs versus Derandomization.
In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 21–30, 2013.

[Yao85] Andrew Chi-Chih Yao.
Separating the Polynomial-Time Hierarchy by Oracles (Preliminary Version).
In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 1–10, 1985.