TECHNISCHE UNIVERSITÄT WIEN

On hardness of some Boolean counting CSPs

Mike Behrisch[×] Miki Hermann[†]

[×]Institute of Discrete Mathematics and Geometry, Algebra Group, TU Wien, Austria

[†]LIX, École Polytechnique, Palaiseau, France

4th June 2021 • Novi Sad

Counting problems

Formally: counting problem $\mathbb{P}$

$\equiv$ a relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$, such that
$\exists p\colon \mathbb{N} \to \mathbb{N}$ polynomial: $\forall (x,y) \in R\colon \quad |y| \leq p(|x|)$

Informally

a binary relation R between instances x and their solutions y;
size of solutions polynomially bounded in size of instance

Aim: count solutions

For $x \in \{0,1\}^*$:

$$\mathrm{S}_{\mathbb{P}}(x) \equiv \mathrm{S}_R(x) := \{y \in \{0,1\}^* \mid (x,y) \in R\}$$
$$|\mathrm{S}_{\mathbb{P}}(x)| \equiv |\mathrm{S}_R(x)| = ? \in \mathbb{N}$$

Complexity of $\mathbb{P}$ understood via Turing machines T computing $|\mathrm{S}_R|\colon \{0,1\}^* \to \mathbb{N}$.

Example: $\#\mathrm{CSP}(Q)$

For a finite set of relations on a finite set A: $Q \subseteq \mathcal{R}_A$

$\#\mathrm{CSP}(Q)$

Input formula $\varphi \equiv \bigwedge_{i=1}^{\ell} \varrho_i(v_{i,1}, \dots, v_{i,m_i})$
$\varrho_i \in Q^{(m_i)}$, $v_{i,j} \in \{x_1, \dots, x_n\}$ for $1 \leq i \leq \ell$, $1 \leq j \leq m_i$

Goal number of satisfying assignments (solutions)
$|\{ s \colon \{x_1, \dots, x_n\} \to A \quad | \quad s \models \varphi \}|$

Compare:

Decision problem # solutions: $= 0$ vs. > 0?

Counting problem # solutions: $= ? \in \mathbb{N}$

In this talk:

$A = \{0, 1\}$ Boolean relations . . . . Post's lattice!

Our Motivation

Goal

Understanding of #ETH

. . . . **E**xponential **T**ime **H**ypothesis for Counting Problems

Our Motivation

Goal

Understanding of #ETH
.... **E**xponential **T**ime **H**ypothesis for Counting Problems

What is #ETH?

analogue of ETH for counting problems

Our Motivation

Goal

Understanding of #ETH
.... Exponential Time Hypothesis for Counting Problems

What is #ETH?

analogue of ETH for counting problems

What is ETH? (Impagliazzo, Paturi, Zane, 2001)

There is $c \in \mathbb{R}_{>0}$ such that...
... no deterministic algorithm solves 3-SAT in time $O(2^{cn})$

i.e., 3-SAT is not solvable in subexponential time.

#ETH

What is #ETH? (Dell, Husfeldt, Marx, Taslaman, Wahlén, 2014)

There is $c \in \mathbb{R}_{>0}$ such that...

... no deterministic algorithm solves #3-SAT in time $O(2^{cn})$

¬#ETH?

$\forall \varepsilon > 0 \, \exists$ deterministic $O((2^{\varepsilon})^n)$-algorithm A: A solves #3-SAT

Lower bound on bases of runtime

$b := \inf\{ c \in \mathbb{R}_{\geq 0} \mid \exists$ deterministic $O(2^{cn})$-algorithm A for #3-SAT$\}$

¬#ETH $\iff$ $b = 0$

algos with faster and faster runtimes

#ETH $\iff$ $b > 0$

no algo better than $O\left(\left(2^b\right)^n\right)$

A characterisation of #ETH, or 'What is n?'

#ETH (Dell et al., 2014)

There is $c \in \mathbb{R}_{>0}$ such that...
... no deterministic algorithm solves #3-SAT in time $O(2^{cn})$
where n number of variables in the solution.

$\forall\, k \geq 3$: #ETH $\iff$ (Dell et al., 2014)

There is $c \in \mathbb{R}_{>0}$ such that...
... no deterministic algorithm solves #k-SAT in time $O(2^{cN})$
where N size of the formula (number of clauses/atoms).

A characterisation of #ETH, or 'What is n?'

#ETH (Dell et al., 2014)

There is $c \in \mathbb{R}_{>0}$ such that...
... no deterministic algorithm solves #3-SAT in time $O(2^{cn})$
where n number of variables in the solution.

$\forall\, k \geq 3$: #ETH $\iff$ (Dell et al., 2014)

There is $c \in \mathbb{R}_{>0}$ such that...
... no deterministic algorithm solves #k-SAT in time $O(2^{cN})$
where N size of the formula (number of clauses/atoms).

Remark

- analogous to ETH and k-SAT by Impagliazzo, Paturi, Zane
- importance of complexity parameter already noted there

Reductions

Standard reductions: $\mathbb{P} \leq \mathbb{Q}$

$\exists$ deterministic polynomial-time $\mathbb{Q}$-oracle algorithm counting $\mathbb{P}$

Reductions with linear parameter growth: $(\mathbb{P}, n) \leq_{\text{lin}} (\mathbb{Q}, N)$

$\exists\, a, b \in \mathbb{N}$: $\exists$ deterministic $\mathbb{Q}$-oracle algorithm A with oracle B:
$\forall$ input x of $\mathbb{P}$ of measure n:

- A counts $\mathrm{S}_{\mathbb{P}}(x)$
- $\forall \varepsilon > 0$: A runs in at most $O(2^{\varepsilon n})$ time-steps (subexponential)
- for each oracle call:
  A calls B on an input of measure $N \leq an + b$

Example

$(\#\mathrm{CSP}(Q), \text{atoms}) \leq_{\text{lin}} (\#\mathrm{CSP}(Q), \text{variables})$ $\quad a = \max_{\varrho \in Q} \mathrm{ar}(\varrho)$

Subexponentiality

$(\mathbb{P}, n)$ is subexponential

$\forall \varepsilon > 0\, \exists$ deterministic $O((2^{\varepsilon})^n)$-algorithm A: A counts $(\mathbb{P}, n)$

$\neg\#\mathrm{ETH} \iff \#3\text{-}\mathrm{SAT}$ is subexponential (wrt. variables)

$\iff \forall k \geq 3$: $\#k\text{-}\mathrm{SAT}$ is subexponential (wrt. variables/atoms)

Lemma:

If $(\mathbb{P}, n) \leq_{\mathrm{lin}} (\mathbb{Q}, N)$, then:

$(\mathbb{Q}, N)$ subexponential $\implies$ $(\mathbb{P}, n)$ subexponential

Preserving subexponentiality

Lemma: For $(\mathbb{P}, n) \leq_{\text{lin}} (\mathbb{Q}, N)$:

$$(\mathbb{Q}, N) \text{ subexponential } \implies (\mathbb{P}, n) \text{ subexponential}$$

- Consider any given $\delta > 0$; define $\varepsilon := \frac{\delta}{a+2}$
- Consider input x for $\mathbb{P}$ of measure $n \geq b$.
- Use the $\mathbb{Q}$-oracle algorithm A with an $O(2^{\varepsilon N})$-oracle B on x
- Each oracle call to B takes $O(2^{\varepsilon N})$ time, i.e. $\leq C_1 \cdot 2^{\varepsilon N} \leq C_1 \cdot 2^{\varepsilon(an+b)} \leq C_1 \cdot 2^{\varepsilon(an+n)} = C_1 \cdot 2^{\varepsilon(a+1)n}$
- Altogether $O(2^{\varepsilon n})$ steps, i.e. $\leq C_2 \cdot 2^{\varepsilon n}$ oracle calls
- Total time for x: $\leq C_3 \cdot 2^{\varepsilon n} \cdot 2^{\varepsilon(a+1)n} = C_3 \cdot 2^{\varepsilon(a+2)n} = C_3 \cdot 2^{\delta n}$

Consequence:

$\neg\#\text{ETH} \iff \#3\text{-SAT}$ is subexp. (wrt. variables)

$\implies \#3\text{-SAT}$ is subexp. (wrt. atoms)

Sparsification

#3-SAT subexp. wrt. atoms $\implies$ #3-SAT subexp. wrt. variables

…needs a different construction:

Problem:

with N variables $\rightsquigarrow$ populate $n = N^\ell$ ℓ-ary atoms (constraints)

Sparsification (Impagliazzo, Paturi, Zane)

$\forall\, \varepsilon > 0 : \exists C \geq 0$: split up any big instance of measure N

- into $\leq 2^{\varepsilon N}$ small subproblems
- each subproblem is sparse $n \leq CN$
- the whole algorithm of splitting and combining runs in $O(2^{\varepsilon N})$

Counting CSPs and reductions

Importance of the complexity measure

- size of instance (# atoms) vs. solution size (# variables)
- $(\#\mathrm{CSP}(P), \mathsf{param}_1) \leq_{\mathsf{lin}} (\#\mathrm{CSP}(Q), \mathsf{param}_2) \implies$
  $(\#\mathrm{CSP}(Q), \mathsf{param}_2)$ subexponential
  $\implies (\#\mathrm{CSP}(P), \mathsf{param}_1)$ subexponential
- for free:
  $(\#\mathrm{CSP}(Q), \mathsf{variables})$ subexponential
  $\implies (\#\mathrm{CSP}(Q), \mathsf{size})$ subexponential
- needs work (e.g. sparsification):
  $(\#\mathrm{CSP}(Q), \mathsf{size})$ subexponential
  $\implies (\#\mathrm{CSP}(Q), \mathsf{variables})$ subexponential

Counting CSPs and reductions

Importance of the complexity measure

- size of instance (# atoms) vs. solution size (# variables)
- $(\#\mathrm{CSP}(P), \mathsf{param}_1) \leq_{\mathsf{lin}} (\#\mathrm{CSP}(Q), \mathsf{param}_2) \implies$
  $(\#\mathrm{CSP}(Q), \mathsf{param}_2)$ subexponential
  $\implies (\#\mathrm{CSP}(P), \mathsf{param}_1)$ subexponential
- for free:
  $(\#\mathrm{CSP}(Q), \mathsf{variables})$ subexponential
  $\implies (\#\mathrm{CSP}(Q), \mathsf{size})$ subexponential
- needs work (e.g. sparsification):
  $(\#\mathrm{CSP}(Q), \mathsf{size})$ subexponential
  $\implies (\#\mathrm{CSP}(Q), \mathsf{variables})$ subexponential

Universal algebra

helps constructing $\leq_{\mathsf{lin}}$-reductions

A Galois connection

Partial polymorphisms

$\forall W \subseteq \mathcal{R}_A \qquad \mathrm{pPol}(W) := \{f \in \mathcal{P}_A \mid \forall \varrho \in W \colon f \rhd \varrho\}$

Invariant relations

$\forall C \subseteq \mathcal{P}_A \qquad \mathrm{Inv}(C) := \{\varrho \in \mathcal{R}_A \mid \forall f \in C \colon f \rhd \varrho\}$

Theorem: for finite A (Romov 1981)

- $\{\mathrm{pPol}(\mathrm{Inv}(W)) \mid W \subseteq \mathcal{R}_A\}$ .... all strong partial clones
  (closure under projections, composition, domain restriction)
- $\{\mathrm{Inv}(\mathrm{pPol}(C)) \mid C \subseteq \mathcal{P}_A\}$ ...... all weak systems with equality
  (closure under conjunctive definitions incl. $=$)
  $[W]_{\wedge,=} = \mathrm{pPol}(\mathrm{Inv}(W))$

Theorem on intervals of strong partial clones

Given a clone $F = \operatorname{Pol} Q$ with relational clone $Q = \operatorname{Inv} F$

Partial clones with total part F

$$\begin{aligned}\mathcal{I}(F) &= \{\, C \leq \mathcal{P}_A \mid C \cap \mathcal{O}_A = F\} \\ &= \{\, C \leq \mathcal{P}_A \mid C \cap \mathcal{O}_A = \operatorname{Pol} Q\} \\ &= \{\, \operatorname{pPol}(W) \mid W \subseteq \mathcal{R}_A \wedge \operatorname{Pol} W = \operatorname{pPol}(W) \cap \mathcal{O}_A = \operatorname{Pol} Q\}\end{aligned}$$

contains largest element $F_\top$

Weak systems with equality generating Q

$$\begin{aligned}\mathcal{I}(Q) &= \{\, \operatorname{Inv} \operatorname{pPol}(W) \mid W \subseteq \mathcal{R}_A \wedge \operatorname{Pol} W = \operatorname{Pol} Q\} \\ &= \left\{ S = [S]_{\wedge,=} \subseteq Q \;\middle|\; \operatorname{Pol} S = \operatorname{Pol} Q \right\} \\ &= \left\{ S = [S]_{\wedge,=} \subseteq Q \;\middle|\; [S]_{\exists,\wedge,=} = Q \right\}\end{aligned}$$

has a least element $S_\perp$

Weak bases

Given a relational clone Q, $F := \mathrm{Pol}\, Q$

Weak basis of Q (Schnoor&Schnoor, 2008)

$S_\perp$ be least weak system incl. $=$ with $[S_\perp]_{\exists,\wedge,=} = Q$

W weak base of Q: any finite $W \subseteq S_\perp$ with $S_\perp = [W]_{\wedge,=}$

(i.e. finite weak generating sets of $S_\perp$)

Properties of weak bases W, W' of $Q = [Q_0]_{\exists,\wedge,=}$

- $[W]_{\exists,\wedge,=} = Q$
- $W \subseteq [Q_0]_{\wedge,=}$

$\Longrightarrow$ $(\#\mathrm{CSP}(W), \text{var.}) \leq_{\text{lin}} (\#\mathrm{CSP}(Q_0), \text{var.})$

- $[W]_{\wedge,=} = [W']_{\wedge,=}$

V. Lagerkvist (2014) determined weak bases for Boolean rel. clones

Result

Theorem

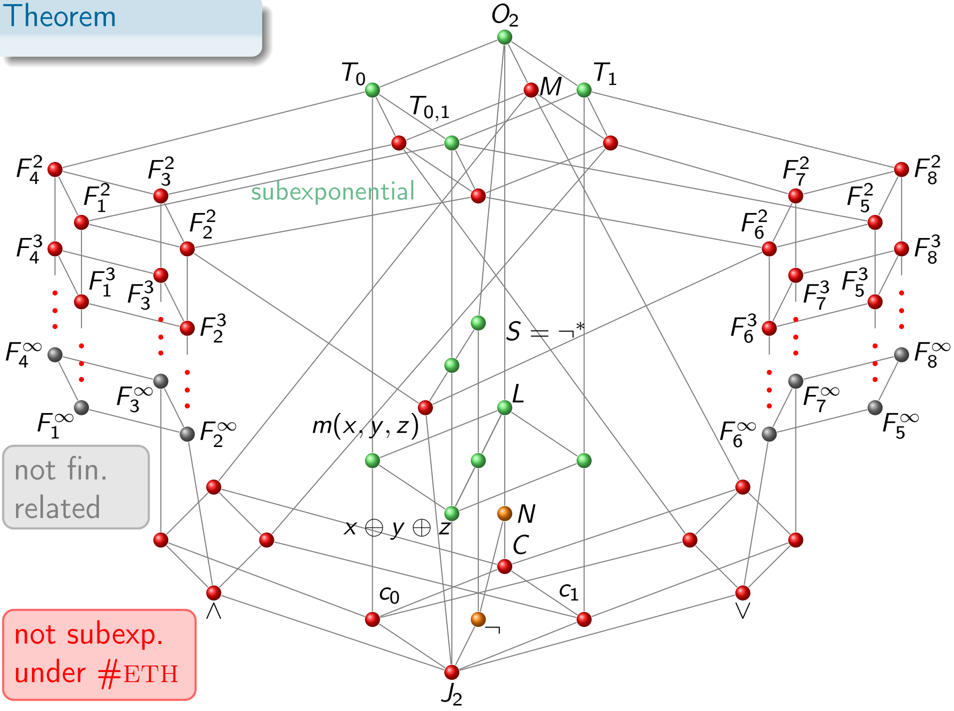
$\forall H \in \mathcal{H}: \quad \neg\#\mathrm{ETH} \iff \#\mathrm{CSP}(H)$ is subexponential wrt. var.

i.e. many Boolean counting CSPs do not have subexponential algorithms under $\#\mathrm{ETH}$.

What is $\mathcal{H}$?

$\mathcal{H} = \{\, H \subseteq_{\mathrm{fin}} \mathcal{R}_2 \mid \mathrm{Pol}\, H \subseteq M \vee \mathrm{Pol}\, H \subseteq F_4^2 \vee \mathrm{Pol}\, H \subseteq F_8^2 \}$

Theorem



O2
T0
T1
M
T0,1
subexponential
F4^2
F3^2
F1^2
F2^2
F4^3
F1^3
F3^3
F2^3
F4^∞
F3^∞
F1^∞
F2^∞
F8^2
F7^2
F5^2
F6^2
F8^3
F5^3
F7^3
F6^3
F8^∞
F7^∞
F6^∞
F5^∞
S = ¬*
L
m(x, y, z)
x ⊕ y ⊕ z
N
C
c0
c1
∧
∨
¬
J2
not fin. related
not subexp. under #ETH

An important problem in social relations

The T-counting problem

input a string

goal count the number of occurrences of the letter T

An important problem in social relations

The T-counting problem

input a string

goal count the number of occurrences of the letter T

Example

input Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.
Thank you for your attention. Thank you for your attention. Thank you for your attention.

answer 42