Randomness and Intractability in Kolmogorov Complexity

Igor Carboni Oliveira

University of Oxford

ICALP 2019
Background and motivation
Structure versus Randomness

Given a string \( x \in \{0, 1\}^n \), is it “structured” or “random”?

Question of relevance to several fields, including:

**LEARNING:** Detecting pattern/structure in data.

**CRYPTO:** Encrypted strings must look random.
Different ways of measuring the complexity of $x$.

If provably secure cryptography exists, algorithms shouldn't be able to estimate the "complexity" of strings.

This talk: Interested in hardness of estimating complexity.
Complexity of strings

- Different ways of measuring the complexity of \( x \).

010011000111000001111000000111110000000111111000000001111111

- **This talk**: Interested in **hardness** of estimating complexity.

If provably secure cryptography exists, algorithms shouldn’t be able to estimate the “complexity” of strings.
Circuit Complexity:

- View $x$ as a boolean function $f : \{0, 1\}^\ell \to \{0, 1\}$.
- complexity($x$) = minimum size of a circuit for $f$.
- Deciding complexity is just the MCSP. Showing this is hard implies $P \neq NP$.

Kolmogorov Complexity:

- complexity($x$) = minimum length of TM that prints $x$.
- Estimating complexity of $x$ is undecidable.
Circuit complexity and Kolmogorov complexity

**Circuit Complexity:**
- View $x$ as a boolean function $f : \{0, 1\}^\ell \to \{0, 1\}$.
- complexity($x$) = minimum size of a circuit for $f$.
- Deciding complexity is just the MCSP. Showing this is hard implies $P \neq NP$.

**Kolmogorov Complexity:**
- complexity($x$) = minimum length of TM that prints $x$.
- Estimating complexity of $x$ is undecidable.
Circuit complexity and Kolmogorov complexity

**Circuit Complexity:**
- View $x$ as a boolean function $f : \{0, 1\}^\ell \to \{0, 1\}$.
- complexity($x$) = minimum size of a circuit for $f$.
- Deciding complexity is just the MCSP. Showing this is hard implies $P \neq NP$.

**Kolmogorov Complexity:**
- complexity($x$) = minimum length of TM that prints $x$.
- Estimating complexity of $x$ is undecidable.

“Extremal” . . . Is there an intermediate notion that is useful?
Time-bounded Kolmogorov complexity

- Takes into account **description length** and **running time** of TM.

\[
K_t(x) \overset{\text{def}}{=} \min_{\text{TM } M, \text{ time } t} |M| + \log t
\]

\(
M \text{ prints } x \text{ in time } t
\)

Takes into account description length and running time of TM.

\[ K_t(x) \overset{\text{def}}{=} \min_{\text{TM } M, \text{ time } t} |M| + \log t \]

TM \( M \), time \( t \)
\( M \) prints \( x \) in time \( t \)

\( K_t(x) \) can be computed in exponential time (brute-force).
Time-bounded Kolmogorov complexity

[Introduced by L. Levin in 1984.]

[Takes into account description length and running time of TM.]

\[ K_t(x) \stackrel{\text{def}}{=} \min_{\text{TM } M, \text{time } t} |M| + \log t \]

\[ \text{M prints } x \text{ in time } t \]

[Kt(x) can be computed in exponential time (brute-force).]

Circuit Complexity

Levin’s (Time-Bounded) Kt

Kolmogorov Complexity

NP

EXP

undecidable
Why is $K_t$ an interesting measure?

$\log t$ gives the “right” measure: connection to optimal search.

Example: Deterministic generation of $n$-bit prime numbers. Fastest known algorithm runs in time $2^{n/2}$ [Lagarias-Odlyzko, 1987].
Why is $K_t$ an interesting measure?

- $\log t$ gives the “right” measure: connection to **optimal search**.

**Example:** Deterministic generation of $n$-bit prime numbers. Fastest known algorithm runs in time $2^{n/2}$ [Lagarias-Odlyzko, 1987].

- Is there a sequence $\{p_n\}$ of $n$-bit primes such that $K_t(p_n) = o(n)$?
Why is $K_t$ an interesting measure?

$\triangleright \log t$ gives the “right” measure: connection to optimal search.

**Example:** Deterministic generation of $n$-bit prime numbers.
Fastest known algorithm runs in time $2^{n/2}$ [Lagarias-Odlyzko, 1987].

$\triangleright$ Is there a sequence $\{p_n\}$ of $n$-bit primes such that $K_t(p_n) = o(n)$?

True $\iff$ there is deterministic prime generation in time $2^{o(n)}$
Can we compute $K_t(x)$ in polynomial time?

- Explicitly posed in [ABK$^+$06]. We already know that $P \neq \text{EXP} \ldots$

- Question strongly connected to power of learning algorithms.

- If provably secure cryptography exists, the answer should be **negative**.
Main Result
We introduce a randomized analogue of Levin’s $K_t$ complexity.

**Main Result:** Randomized $K_t$ complexity cannot be estimated in BPP. (The problem can be solved in randomized exponential time.)

This is an unconditional lower bound for a natural problem.
Randomized Kt Complexity

- Adaptation of Levin’s definition to Randomized Computation.

- For \( x \in \{0, 1\}^n \), we consider algorithms that generate \( x \) w.h.p.:

\[
rKt(x) \overset{\text{def}}{=} \min_{\text{randomized TM } M, \text{ time } t} \left| M \right| + \log t \quad \Pr_M[M \text{ prints } x \text{ in time } t] \geq 2/3
\]

**Intuition:** String probabilistically decompressed from short representation.
Remarks about $K_t$ Complexity

$$rK_t(x) \overset{\text{def}}{=} \min_{\text{randomized TM } M, \text{ time } t} \Pr_M[M \text{ prints } x \text{ in time } t] \geq \frac{2}{3} |M| + \log t$$

Definition is robust.
Remarks about Kt Complexity

\[ r_{Kt}(x) \overset{\text{def}}{=} \min_{\text{randomized TM } M, \text{ time } t} |M| + \log t \]

\[ \Pr_M[M \text{ prints } x \text{ in time } t] \geq 2/3 \]

- Definition is **robust**.

- Connected to **pseudodeterministic algorithms**. In particular, it follows from a recent joint work with R. Santhanam that
  - There is an infinite sequence \( \{p_m\}_m \) of \( m \)-bit primes such that \( r_{Kt}(p_m) \leq m^{o(1)} \).
Remarks about Kt Complexity

\[
\text{rKt}(x) \triangleq \min_{\text{randomized TM } M, \text{ time } t} |M| + \log t \quad \Pr_M[ M \text{ prints } x \text{ in time } t ] \geq 2/3
\]

- Definition is robust.

- Connected to pseudodeterministic algorithms. In particular, it follows from a recent joint work with R. Santhanam that
  - There is an infinite sequence \( \{p_m\}_m \) of \( m \)-bit primes such that \( \text{rKt}(p_m) \leq m^{o(1)} \).

- Under standard derandomization assumptions, \( \text{Kt}(x) = \Theta(\text{rKt}(x)) \).
How difficult is to compute the complexity of a string?

Can we compute $K_t(x)$ in polynomial time?
MKtP – Minimum Kt Problem

Can we compute $rK_t(x)$ in randomized polynomial time?
MrKtP – Minimum rKt Problem
“rKt cannot be approximated in quasi-polynomial time.”

**Theorem 1.** For every $\varepsilon > 0$, there is no randomized algorithm running in time $n^{\text{poly} (\log n)}$ that distinguishes between $rKt(x) \leq n^\varepsilon$ versus $rKt(x) \geq .99n$, where $n$ is the length of the input string $x$.

**Remark.** This problem can be solved in randomized exponential time.
Techniques
Preliminaries

**Gap-MrKtP**[$n^\varepsilon, .99n$]:

\[
\mathcal{YES}_n \overset{\text{def}}{=} \{x \in \{0, 1\}^n \mid rKt(x) \leq n^\varepsilon\}
\]
\[
\mathcal{NO}_n \overset{\text{def}}{=} \{x \in \{0, 1\}^n \mid rKt(x) > .99n\}
\]

▷ Algorithm for Gap-MrKtP[$n^\varepsilon, .99n$] distinguishes two cases.
Preliminaries

Gap-MrKtP\( [n^\varepsilon, .99n] \):

\[ \forall \mathcal{S}_n \overset{\text{def}}{=} \{ x \in \{0, 1\}^n \mid rKt(x) \leq n^\varepsilon \} \]

\[ \forall O_n \overset{\text{def}}{=} \{ x \in \{0, 1\}^n \mid rKt(x) > .99n \} \]

- Algorithm for Gap-MrKtP\( [n^\varepsilon, .99n] \) distinguishes two cases.

- Approach: *indirect diagonalization* using techniques from theory of pseudorandomness.
Main Lemmas

**Lemma 1.** For every $\varepsilon > 0$, $\text{BPE} \leq_{\text{P}/\text{poly}} \text{Gap-MrKtP}[n^\varepsilon, .99n]$.

▷ Very strong **non-uniform inclusion**.
Lemma 1. For every $\varepsilon > 0$, $\text{BPE} \leq_{P/poly} \text{Gap-MrKtP}[n^\varepsilon, .99n]$.

▷ Very strong non-uniform inclusion.

Lemma 2. For every $\varepsilon > 0$, $\text{PSPACE} \subseteq \text{BPP}^{\text{Gap-MrKtP}[n^\varepsilon, .99n]}$.

▷ Strong uniform inclusion.
Main Lemmas

**Lemma 1.** For every $\varepsilon > 0$, $\text{BPE} \leq_{P/\text{poly}} \text{Gap-MrKtP}[n^\varepsilon, .99n]$.

- Very strong **non-uniform inclusion**.

**Lemma 2.** For every $\varepsilon > 0$, $\text{PSPACE} \subseteq \text{BPP}^{\text{Gap-MrKtP}[n^\varepsilon, .99n]}$.

- Strong **uniform inclusion**.

**Lemma 3.** If $n \leq s(n) \leq 2^{o(n)}$ then $\text{DSPACE}[s^3] \not\subseteq \text{Circuit}[s]$.

- Nexus between **uniform** and **non-uniform** inclusions.
Main Result from Lemmas 1, 2, and 3

Proof by contradiction. Sketch of weaker result:

Assume Gap-MrKtP[\(n^\varepsilon, .99n\)] \(\in\) BPP. This also gives inclusion in P/poly.

L1. BPE \(\leq_{P/poly}\) Gap-MrKtP[\(n^\varepsilon, .99n\)]. This implies BPE \(\subseteq\) Circuit[poly].

L2. PSPACE \(\subseteq\) BPP^Gap-MrKtP[\(n^\varepsilon,.99n\)]. This implies PSPACE \(\subseteq\) BPP.

Translation gives DSPACE[\(n^{\text{poly} \log n}\)] \(\subseteq\) BPTIME[\(n^{\text{poly} \log n}\)] \(\subseteq\) BPE \(\subseteq\) Circuit[poly].

This inclusion contradicts L3. DSPACE[\(s^3\)] \(\nsubseteq\) Circuit[\(s\)].
Hardness versus Randomness paradigm:

From “hard” \( f : \{0, 1\}^m \rightarrow \{0, 1\} \), one designs a “pseudorandom generator”

\[
G^f : \{0, 1\}^\ell \rightarrow \{0, 1\}^n.
\]
Hardness versus Randomness paradigm:

From “hard” \( f : \{0, 1\}^m \rightarrow \{0, 1\} \), one designs a “pseudorandom generator”

\[
G^f : \{0, 1\}^\ell \rightarrow \{0, 1\}^n.
\]

Proof often shows: Algorithm “breaking” \( G^f \) can be used to “compute” \( f \).
> **Hardness versus Randomness paradigm:**

From “hard” \( f : \{0, 1\}^m \rightarrow \{0, 1\} \), one designs a “pseudorandom generator”

\[ G^f : \{0, 1\}^\ell \rightarrow \{0, 1\}^n. \]

**Proof often shows:** Algorithm “breaking” \( G^f \) can be used to “compute” \( f \).

**Crucial:** We can upper bound \( rKt \) complexity of output strings of \( G^f \).

Algorithm solving \( \text{Gap-MrKtP}[n^\varepsilon, .99n] \) acts as a **distinguisher**!
L1. \( \text{BPE} \leq_{P/poly} \text{Gap-MrKtP}[n^\epsilon, .99n] \). Relies on PRG construction of [BFNW93].

L2. \( \text{PSPACE} \subseteq \text{BPP}^{\text{Gap-MrKtP}[n^\epsilon, .99n]} \). Relies on PRG construction of [TV07].
**L1.** BPE $\leq_{P/poly} \operatorname{Gap-MrKtP}[n^\epsilon, .99n]$. Relies on PRG construction of [BFNW93].

**L2.** $\text{PSPACE} \subseteq \operatorname{BPP}^{\operatorname{Gap-MrKtP}[n^\epsilon,.99n]}$. Relies on PRG construction of [TV07].

- **L1** and variants: require notions of string complexity such as $rKt$ and $Kt$.

- **Randomness is used** in the proof of **L2**: bottleneck to Levin’s $Kt$. 
Further Results

(uniform versus non-uniform lower bounds)
Circuit lower bounds

- Lower bound presented before holds against **uniform** algorithms.

- Boolean circuits capture **non-uniform** computation.

**Major Challenge:** Show for an explicit problem that any circuit solving the problem requires several AND, OR, NOT gates.
State-of-the-art circuit lower bounds

After 50+ years of intensive investigation:

- Existing circuit lower bounds are of the form $c \cdot n$ for constant $c$.

- Boolean formulas (weaker model): lower bounds of the form $n^{3-o(1)}$. 
After 50+ years of intensive investigation:

- Existing circuit lower bounds are of the form $c \cdot n$ for constant $c$.
- Boolean formulas (weaker model): lower bounds of the form $n^{3-o(1)}$.

We know that Gap-MrKtP[$n^\epsilon, .99n$] is hard. Can we use it to prove better circuit and formula lower bounds?
Emerging theory showing that \textit{weak} lower bounds can be "magnified" to \textit{strong} lower bounds.
Emerging theory showing that weak lower bounds can be “magnified” to strong lower bounds.

By adapting recent joint work with J. Pich and R. Santhanam:

**Theorem 2.** If for every $\varepsilon > 0$,

\[
\text{Gap-MrKtP}[n^\varepsilon, .99n] \not\subseteq \text{Circuit}[n^{1.01}], \text{ then } \text{BPEXP} \not\subseteq \text{Circuit}[\text{poly}].
\]

\[
\text{Gap-MrKtP}[n^\varepsilon, .99n] \not\subseteq \text{Formula}[n^{3.01}], \text{ then } \text{BPEXP} \not\subseteq \text{Formula}[\text{poly}].
\]
Open Problems
Can we prove that computing Levin’s $K_t$ complexity cannot be done in deterministic polynomial time?
This work: natural problem that cannot be solved in randomized quasi-polynomial time.

Can we reduce approximating $rKt$ to a problem in $\text{NEXP}$?

Even a randomized reduction would show $\text{NEXP} \neq \text{BPP}$. 
References and related work

Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger.
*Power from random strings.*

Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy.
*The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory.*

Eric Allender.
*The complexity of complexity.*

László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson.
*BPP has subexponential time simulations unless EXPTIME has publishable proofs.*

Eran Gat and Shafi Goldwasser.
*Probabilistic search algorithms with unique answers and their cryptographic applications.*

Leonid A. Levin.
*Randomness conservation inequalities; information and independence in mathematical theories.*

Ming Li and Paul M. B. Vitányi.
*An Introduction to Kolmogorov Complexity and Its Applications.*

Igor Carboni Oliveira, Ján Pich, and Rahul Santhanam.
*Hardness magnification near state-of-the-art lower bounds.*

Igor Carboni Oliveira and Rahul Santhanam.
*Pseudodeterministic constructions in subexponential time.*

Luca Trevisan and Salil P. Vadhan.
*Pseudorandomness and average-case complexity via uniform reductions.*