Isomorphism Problems and Minimum Circuit Size

Eric Allender
Joshua Grochow
Cris Moore
Dieter van Melkebeek
Andrew Morgan

WACT Paris, March 6, 2018

Outline

Characters:

(1) Graph Isomorphism and generalizations

(2) Minimum Circuit Size and related problems

Outline

Characters:

(1) Graph Isomorphism and generalizations
(2) Minimum Circuit Size and related problems

Status: NP-intermediate

Outline

Characters:

(1) Graph Isomorphism and generalizations
(2) Minimum Circuit Size and related problems

Status: NP-intermediate

Result: Reductions from (1) to (2)

Isomorphism Problem

Isomorphism Problem

Parameterized by family of actions (H_x, Ω_x) of a finite group H_x on a universe Ω_x (depending on the input x)

Isomorphism Problem

Parameterized by family of actions (H_x, Ω_x) of a finite group H_x on a universe Ω_x (depending on the input x)

Given: $\omega_0, \omega_1 \in \Omega$
Question: $(\exists h \in H)\, h(\omega_0) = \omega_1$?

Isomorphism Problem

Parameterized by family of actions (H_x, Ω_x) of a finite group H_x on a universe Ω_x (depending on the input x)

Given: $\omega_0, \omega_1 \in \Omega$
Question: $(\exists h \in H)\, h(\omega_0) = \omega_1$?

Instantiations:

Isomorphism Problem	H	Ω
Discrete Log	$\mathbb{F}_p$	$\mathbb{F}_p$
Graph Isomorphism	S_n	graphs with n vertices
Linear Code Equivalence	S_n	subspaces of $\mathbb{F}_q^n$
Permutation Group Conjugacy	S_n	subgroups of S_n
Matrix Subspace Conjugacy	$\mathrm{GL}_n(\mathbb{F}_q)$	subspaces of $\mathbb{F}_q^{n \times n}$

Minimum Circuit Size and Related Problems

Minimum Circuit Size and Related Problems

$\mathrm{CC}(x)$ is the smallest size of circuit with truth-table x.

Minimum Circuit Size and Related Problems

$\mathrm{CC}(x)$ is the smallest size of circuit with truth-table x.

$\mathrm{MCSP} = \{(x, \theta) : \mathrm{CC}(x) \leq \theta\}$

Minimum Circuit Size and Related Problems

$\mathrm{CC}(x)$ is the smallest size of circuit with truth-table x.

$\mathrm{MCSP} = \{(x, \theta) : \mathrm{CC}(x) \leq \theta\}$

Variants based on various notions of CC.

Minimum Circuit Size and Related Problems

$\mathrm{CC}(x)$ is the smallest size of circuit with truth-table x.

$\mathrm{MCSP} = \{(x, \theta) : \mathrm{CC}(x) \leq \theta\}$

Variants based on various notions of CC.

Fix a universal Turing Machine U.

Given x, consider programs p that, on input i, output x_i, *i.e.*, $U(p, i) = x_i$.

Minimum Circuit Size and Related Problems

$\mathrm{CC}(x)$ is the smallest size of circuit with truth-table x.

$\mathrm{MCSP} = \{(x, \theta) : \mathrm{CC}(x) \leq \theta\}$

Variants based on various notions of CC.

Fix a universal Turing Machine U.

Given x, consider programs p that, on input i, output x_i, *i.e.*, $U(p, i) = x_i$.

$\mathrm{KT}(x)$ is the smallest value of $|p| + T$, where p runs in time T.

Minimum Circuit Size and Related Problems

$\mathrm{CC}(x)$ is the smallest size of circuit with truth-table x.

$\mathrm{MCSP} = \{(x, \theta) : \mathrm{CC}(x) \leq \theta\}$

Variants based on various notions of CC.

Fix a universal Turing Machine U.

Given x, consider programs p that, on input i, output x_i, *i.e.*, $U(p, i) = x_i$.

$\mathrm{KT}(x)$ is the smallest value of $|p| + T$, where p runs in time T.

$\mathrm{KT}(x)$ and $\mathrm{CC}(x)$ are polynomially related.

Minimum Circuit Size and Related Problems

$\mathrm{CC}(x)$ is the smallest size of circuit with truth-table x.

$\mathrm{MCSP} = \{(x, \theta) : \mathrm{CC}(x) \leq \theta\}$

Variants based on various notions of CC.

Fix a universal Turing Machine U.

Given x, consider programs p that, on input i, output x_i, *i.e.*, $U(p, i) = x_i$.

$\mathrm{KT}(x)$ is the smallest value of $|p| + T$, where p runs in time T.

$\mathrm{KT}(x)$ and $\mathrm{CC}(x)$ are polynomially related.

$\mathrm{MKTP} = \{(x, \theta) : \mathrm{KT}(x) \leq \theta\}$

Breaking Pseudorandom Generators

Breaking Pseudorandom Generators

A pseudorandom generator is an efficient deterministic algorithm PRG such that PRG(U_ℓ) looks like U_r for $\ell \ll r$.

Breaking Pseudorandom Generators

A pseudorandom generator is an efficient deterministic algorithm PRG such that $\mathrm{PRG}(U_\ell)$ looks like U_r for $\ell \ll r$.

$\mathrm{MCSP}/\mathrm{MKTP}$ breaks PRGs that fool circuits/efficient programs.

Breaking Pseudorandom Generators

A pseudorandom generator is an efficient deterministic algorithm PRG such that PRG(U_ℓ) looks like U_r for $\ell \ll r$.

$\mathrm{MCSP}/\mathrm{MKTP}$ breaks PRGs that fool circuits/efficient programs.

"Hard" problems underlying PRG constructions that fool circuits/efficient programs reduce to $\mathrm{MCSP}/\mathrm{MKTP}$.

Inverting Functions

Inverting Functions

PRG construction of [Hastad Impagliazzo Levin Luby] based on one-way functions.

Inverting Functions

PRG construction of [Hastad Impagliazzo Levin Luby] based on one-way functions.

Lemma: There exists a polynomial-time probabilistic Turing machine M using oracle access to $\mathrm{MCSP}/\mathrm{MKTP}$ so that the following holds. For any circuit C of size n, if $\sigma \sim \{0,1\}^n$ and then

$$\Pr[C(\tau) = C(\sigma)] \geq 1/\mathrm{poly}(n) \text{ with } \tau \doteq M(C, C(\sigma)).$$

Inverting Functions

PRG construction of [Hastad Impagliazzo Levin Luby] based on one-way functions.

Lemma: There exists a polynomial-time probabilistic Turing machine M using oracle access to $\mathrm{MCSP}/\mathrm{MKTP}$ so that the following holds. For any circuit C of size n, if $\sigma \sim \{0,1\}^n$ and then

$$\Pr[C(\tau) = C(\sigma)] \geq 1/\mathrm{poly}(n) \text{ with } \tau \doteq M(C, C(\sigma)).$$

Apply to $C(\sigma) = \sigma(G_1)$ where $\sigma \sim S_n$.

Inverting Functions

PRG construction of [Hastad Impagliazzo Levin Luby] based on one-way functions.

Lemma: There exists a polynomial-time probabilistic Turing machine M using oracle access to $\mathrm{MCSP}/\mathrm{MKTP}$ so that the following holds. For any circuit C of size n, if $\sigma \sim \{0,1\}^n$ and then

$$\Pr[C(\tau) = C(\sigma)] \geq 1/\mathrm{poly}(n) \text{ with } \tau \doteq M(C, C(\sigma)).$$

Apply to $C(\sigma) = \sigma(G_1)$ where $\sigma \sim S_n$.

If $G_0 \equiv G_1$ then $\sigma(G_1) \equiv \sigma(G_0)$, so

$$\Pr[C(\tau) = \sigma(G_0)] \geq 1/\mathrm{poly}(n) \text{ with } \tau \doteq M(C, \sigma(G_0)).$$

Inverting Functions

PRG construction of [Hastad Impagliazzo Levin Luby] based on one-way functions.

Lemma: There exists a polynomial-time probabilistic Turing machine M using oracle access to $\mathrm{MCSP}/\mathrm{MKTP}$ so that the following holds. For any circuit C of size n, if $\sigma \sim \{0,1\}^n$ and then

$$\Pr[C(\tau) = C(\sigma)] \geq 1/\mathrm{poly}(n) \text{ with } \tau \doteq M(C, C(\sigma)).$$

Apply to $C(\sigma) = \sigma(G_1)$ where $\sigma \sim S_n$.

If $G_0 \equiv G_1$ then $\sigma(G_1) \equiv \sigma(G_0)$, so

$$\Pr[C(\tau) = \sigma(G_0)] \geq 1/\mathrm{poly}(n) \text{ with } \tau \doteq M(C, \sigma(G_0)).$$

$C(\tau) = \sigma(G_0)$ means $\tau^{-1} \circ \sigma$ is an isomorphism from G_0 to G_1.

Randomized Reductions

Yieds randomized reduction from Graph Isomorphism to $\mathrm{MCSP}/\mathrm{MKTP}$ without false positives [Allender Das].

Randomized Reductions

Yieds randomized reduction from Graph Isomorphism to $\mathrm{MCSP}/\mathrm{MKTP}$ without false positives [Allender Das].

Generalizes to generic Isomorphism Problem under mild conditions.

Randomized Reductions

Yieds randomized reduction from Graph Isomorphism to $\mathrm{MCSP}/\mathrm{MKTP}$ without false positives [Allender Das].

Generalizes to generic Isomorphism Problem under mild conditions.

Question: How to eliminate errors?

Randomized Reductions

Yieds randomized reduction from Graph Isomorphism to $\mathrm{MCSP}/\mathrm{MKTP}$ without false positives [Allender Das].

Generalizes to generic Isomorphism Problem under mild conditions.

Question: How to eliminate errors?

Suffices to obtain reductions without false negatives.

Randomized Reductions

Yieds randomized reduction from Graph Isomorphism to $\mathrm{MCSP}/\mathrm{MKTP}$ without false positives [Allender Das].

Generalizes to generic Isomorphism Problem under mild conditions.

Question: How to eliminate errors?

Suffices to obtain reductions without false negatives.

Approach: Use $\mathrm{MCSP}/\mathrm{MKTP}$ to *verify* witnesses of nonisomorphism.

Witnessing Rigid Graph Nonisomorphism

Let G_0, G_1 be *rigid* graphs, *i.e.*, no non-trivial automorphisms.

Witnessing Rigid Graph Nonisomorphism

Let G_0, G_1 be *rigid* graphs, *i.e.*, no non-trivial automorphisms.

Key fact: if $G_0 \equiv G_1$, there are only $n!$ distinct graphs among permutations of G_0 and G_1; if $G_0 \not\equiv G_1$, there are $2(n!)$.

Witnessing Rigid Graph Nonisomorphism

Let G_0, G_1 be *rigid* graphs, *i.e.*, no non-trivial automorphisms.

Key fact: if $G_0 \equiv G_1$, there are only $n!$ distinct graphs among permutations of G_0 and G_1; if $G_0 \not\equiv G_1$, there are $2(n!)$.

Consider sampling $r \sim \{0,1\}$ and $\pi \sim S_n$ uniformly, and outputting the adjacency matrix of $\pi(G_r)$.

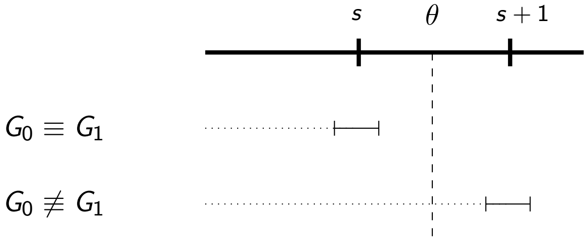
- If $G_0 \equiv G_1$, this has entropy $s \doteq \log n!$
- If $G_0 \not\equiv G_1$, this has entropy $s + 1$

Main idea: Use KT-complexity of a random sample as a proxy for entropy.

Witnessing Rigid Graph Nonisomorphism

Let $y = \pi(G_r)$, $\pi \sim S_n$, $r \sim \{0, 1\}$.

Hope: $\mathrm{KT}(y)$ is typically near the entropy, never much larger:



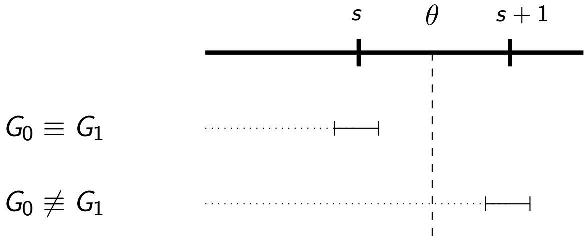
where $s = \log(n!)$

Witness nonisomorphism by checking $\mathrm{KT}(y) > \theta$.

Witnessing Rigid Graph Nonisomorphism

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$, $\pi_i \sim S_n$, $r_i \sim \{0,1\}$.

Reality: $\mathrm{KT}(y)/y$ is typically near the entropy, never much larger:



where $s = \log(n!)$

Witness nonisomorphism by checking $\mathrm{KT}(y)/t > \theta$.

Isomorphic Case

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2}) \cdots \pi_t(G_{r_t})$.

Isomorphic Case

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

Since $G_0 \equiv G_1$ can can describe y by:

- Adjacency matrix of G_0: n^2 bits.
- For each $i \in [t]$, a permutation $\tau_t \in S_n$ such that $\tau_i(G_0) = \pi_i(G_{r_i})$.

Isomorphic Case

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

Since $G_0 \equiv G_1$ can can describe y by:

- Adjacency matrix of G_0: n^2 bits.
- For each $i \in [t]$, a permutation $\tau_t \in S_n$ such that $\tau_i(G_0) = \pi_i(G_{r_i})$.

Lehmer Code. There is an indexing of S_n by the numbers $1, \ldots, n!$ so that the i-th permutation can be decoded from the binary representation of i in time poly(n).

Isomorphic Case

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

Since $G_0 \equiv G_1$ can can describe y by:

- Adjacency matrix of G_0: n^2 bits.
- For each $i \in [t]$, a permutation $\tau_t \in S_n$ such that $\tau_i(G_0) = \pi_i(G_{r_i})$.

Lehmer Code. There is an indexing of S_n by the numbers $1, \ldots, n!$ so that the i-th permutation can be decoded from the binary representation of i in time poly(n).

Yields $\mathrm{KT}(y) \leq n^2 + t \cdot \lceil s \rceil + \text{poly}(n, \log t)$.

Isomorphic Case

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

Since $G_0 \equiv G_1$ can can describe y by:

- Adjacency matrix of G_0: n^2 bits.
- For each $i \in [t]$, a permutation $\tau_t \in S_n$ such that $\tau_i(G_0) = \pi_i(G_{r_i})$.

Lehmer Code. There is an indexing of S_n by the numbers $1, \ldots, n!$ so that the i-th permutation can be decoded from the binary representation of i in time poly(n).

Yields $\mathrm{KT}(y) \leq n^2 + t \cdot \lceil s \rceil + \mathrm{poly}(n, \log t)$.

Lemma: $\mathrm{KT}(y) \leq ts + t^{1-\alpha}\mathrm{poly}(n)$ for some constant $\alpha > 0$.

Isomorphic Case

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

Since $G_0 \equiv G_1$ can can describe y by:

- Adjacency matrix of G_0: n^2 bits.
- For each $i \in [t]$, a permutation $\tau_t \in S_n$ such that $\tau_i(G_0) = \pi_i(G_{r_i})$.

Lehmer Code. There is an indexing of S_n by the numbers $1, \ldots, n!$ so that the i-th permutation can be decoded from the binary representation of i in time poly(n).

Yields $\mathrm{KT}(y) \leq n^2 + t \cdot \lceil s \rceil + \mathrm{poly}(n, \log t)$.

Lemma: $\mathrm{KT}(y) \leq ts + t^{1-\alpha}\mathrm{poly}(n)$ for some constant $\alpha > 0$.

Implies that $\mathrm{KT}(y)/t \leq s + o(1)$ for sufficiently large $t = \mathrm{poly}(n)$.

Entropy Estimator

Entropy Estimator

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

Entropy Estimator

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

If $G_0 \equiv G_1$ then $\mathrm{KT}(y)/t \leq s + o(1)$ always holds for sufficiently large $t = \mathrm{poly}(n)$.

Entropy Estimator

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2}) \cdots \pi_t(G_{r_t})$.

If $G_0 \equiv G_1$ then $\mathrm{KT}(y)/t \leq s + o(1)$ always holds for sufficiently large $t = \text{poly}(n)$.

As y consists of t independent samples from a flat distribution of entropy s, $\mathrm{KT}(y)/t \geq s - o(1)$ holds with high probability.

Entropy Estimator

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$.

If $G_0 \equiv G_1$ then $\mathrm{KT}(y)/t \leq s + o(1)$ always holds for sufficiently large $t = \text{poly}(n)$.

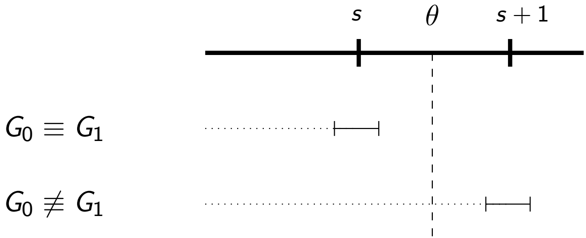
As y consists of t independent samples from a flat distribution of entropy s, $\mathrm{KT}(y)/t \geq s - o(1)$ holds with high probability.

Corollary: $KT(y)/t$ is a *probably approximately correct (PAC) underestimator* for the entropy.

Witnessing Graph Nonisomorphism

Let $y = \pi_1(G_{r_1})\pi_2(G_{r_2})\cdots\pi_t(G_{r_t})$, $\pi_i \sim S_n$, $r_i \sim \{0,1\}$.

Reality: $\mathrm{KT}(y)/y$ is typically near the entropy, never much larger:



where $s = \log(n!/|\mathrm{Aut}(G_i)|)$ (assumed the same for $i \in \{0,1\}$ for ease of exposition).

Witness nonisomorphism by checking $\mathrm{KT}(y)/t > \theta$.

Witnessing Graph Nonisomorphism

Witnessing Graph Nonisomorphism

Two complications for non-rigid graphs:

- Encoding distinct permutations of G_0 as numbers $1, \ldots, n!$ is too expensive.

Witnessing Graph Nonisomorphism

Two complications for non-rigid graphs:

- Encoding distinct permutations of G_0 as numbers $1, \ldots, n!$ is too expensive.
- Knowing threshold $\theta = s + 1/2$ requires knowing $|\mathrm{Aut}(G_i)|$.

Witnessing Graph Nonisomorphism

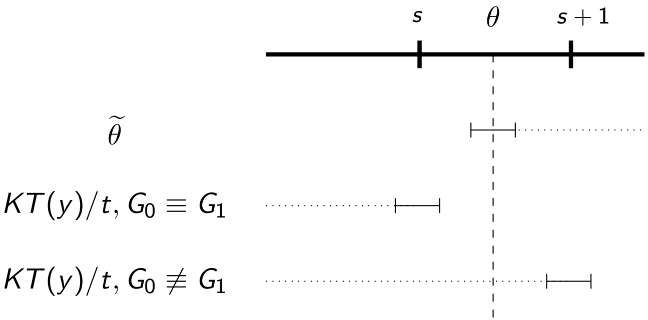
Two complications for non-rigid graphs:

- Encoding distinct permutations of G_0 as numbers $1, \ldots, n!$ is too expensive.
- Knowing threshold $\theta = s + 1/2$ requires knowing $|\mathrm{Aut}(G_i)|$.

Using entropy estimator yields randomized reduction with two-sided error (from which false positives can be eliminated using known search-to-decision for Graph Isomorphism).

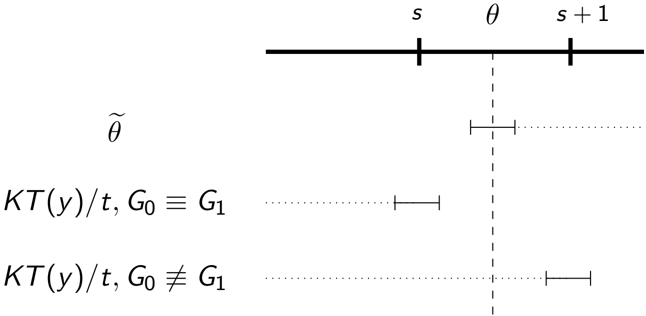
Witnessing Graph Nonisomorphism

It suffices to give a probably approximately correct (PAC) *over*estimator for θ:



Witnessing Graph Nonisomorphism

It suffices to give a probably approximately correct (PAC) *over*estimator for θ:



Equivalently, it suffices to give a PAC *under*estimator for $\log|\mathrm{Aut}(G_i)|$, since $\theta = \log(n!) - \log|\mathrm{Aut}(G_i)|$.

PAC Underestimating $\log|\mathrm{Aut}(G)|$

Claim. There is an efficient randomized algorithm using MKTP to PAC underestimate $\log|\mathrm{Aut}(G)|$ when given G.

PAC Underestimating $\log|\mathrm{Aut}(G)|$

Claim. There is an efficient randomized algorithm using MKTP to PAC underestimate $\log|\mathrm{Aut}(G)|$ when given G.

Proof. Recall that there is an efficient deterministic algorithm using an oracle for Graph Isomorphism that computes generators for $\mathrm{Aut}(G)$.

Plug in an existing $\mathsf{RP}^{\mathrm{MKTP}}$ algorithm for the oracle. This gives us generators for a subgroup $A \leq \mathrm{Aut}(G)$ such that $A = \mathrm{Aut}(G)$ with high probability.

$|A|$ can be computed efficiently from its generators. Output $\log|A|$.

PAC Underestimating $\log|\mathrm{Aut}(G)|$

Claim. There is an efficient randomized algorithm using MKTP to PAC underestimate $\log|\mathrm{Aut}(G)|$ when given G.

Proof. Recall that there is an efficient deterministic algorithm using an oracle for Graph Isomorphism that computes generators for $\mathrm{Aut}(G)$.

Plug in an existing $\mathrm{RP}^{\mathrm{MKTP}}$ algorithm for the oracle. This gives us generators for a subgroup $A \leq \mathrm{Aut}(G)$ such that $A = \mathrm{Aut}(G)$ with high probability.

$|A|$ can be computed efficiently from its generators. Output $\log|A|$.

Theorem: Graph Isomorphism is in $\mathrm{ZPP}^{\mathrm{MKTP}}$.

Generalization

Generalization

Isomorphism Problem with underlying action (H, Ω):

Given: $\omega_0, \omega_1 \in \Omega$
Question: $(\exists h \in H)\ h(\omega_0) = \omega_1$?

Generalization

Isomorphism Problem with underlying action (H, Ω):

Given: $\omega_0, \omega_1 \in \Omega$
Question: $(\exists h \in H)\ h(\omega_0) = \omega_1$?

$h(\omega)$ for $h \sim H$ yields a flat distribution of entropy
$s = \log(|H|/|\mathrm{Aut}(\omega)|)$ where $\mathrm{Aut}(\omega) \doteq \{h \in H : \ h(\omega) = \omega)\}$.

Generalization

Isomorphism Problem with underlying action (H, Ω):

Given: $\omega_0, \omega_1 \in \Omega$
Question: $(\exists h \in H)\, h(\omega_0) = \omega_1$?

$h(\omega)$ for $h \sim H$ yields a flat distribution of entropy
$s = \log(|H|/|\mathrm{Aut}(\omega)|)$ where $\mathrm{Aut}(\omega) \doteq \{h \in H :\ h(\omega) = \omega)\}$.

Use $y = h_1(\omega_{r_1}) \cdots h_t(G_{r_t})$ where $h_i \sim H$ and $r_i \sim \{0, 1\}$.

Generalization

Isomorphism Problem with underlying action (H, Ω):

Given: $\omega_0, \omega_1 \in \Omega$
Question: $(\exists h \in H)\ h(\omega_0) = \omega_1$?

$h(\omega)$ for $h \sim H$ yields a flat distribution of entropy
$s = \log(|H|/|\mathrm{Aut}(\omega)|)$ where $\mathrm{Aut}(\omega) \doteq \{h \in H : h(\omega) = \omega)\}$.

Use $y = h_1(\omega_{r_1}) \cdots h_t(G_{r_t})$ where $h_i \sim H$ and $r_i \sim \{0, 1\}$.

Need:

- Encoding of the cosets of $\mathrm{Aut}(\omega)$ that achieves KT-complexity close to the entropy s.
- Efficiently compute $|H|$.
- PAC underestimator for $|\mathrm{Aut}(\omega)|$ that is efficiently computable with access to MKTP.

Encoding Flat Samplable Distributions

Encoding Flat Samplable Distributions

Encoding Lemma: Let C be a circuit sampling a distribution of max-entropy s. There is a circuit D of size poly($|C|$) and, for each outcome z, a string i_z of length $s + \log s + O(1)$ s. t. $D(i_z) = z$.

Encoding Flat Samplable Distributions

Encoding Lemma: Let C be a circuit sampling a distribution of max-entropy s. There is a circuit D of size poly$(|C|)$ and, for each outcome z, a string i_z of length $s + \log s + O(1)$ s. t. $D(i_z) = z$.

Proof based on hashing with linear-algebraic family.

Encoding Flat Samplable Distributions

Encoding Lemma: Let C be a circuit sampling a distribution of max-entropy s. There is a circuit D of size poly($|C|$) and, for each outcome z, a string i_z of length $s + \log s + O(1)$ s. t. $D(i_z) = z$.

Proof based on hashing with linear-algebraic family.

Example: C computes a random permutation of a graph G on n vertices. Each permutation of G can be decoded from a string of length $s + \log s + O(1)$, where $s = \log(n!/|\mathrm{Aut}(G)|)$.

Encoding Flat Samplable Distributions

Encoding Lemma: Let C be a circuit sampling a distribution of max-entropy s. There is a circuit D of size $\text{poly}(|C|)$ and, for each outcome z, a string i_z of length $s + \log s + O(1)$ s. t. $D(i_z) = z$.

Proof based on hashing with linear-algebraic family.

Example: C computes a random permutation of a graph G on n vertices. Each permutation of G can be decoded from a string of length $s + \log s + O(1)$, where $s = \log(n!/|\text{Aut}(G)|)$.

Worse than $\lceil s \rceil$ but suffices to show $\mathrm{KT}(y)/t \leq s + o(1)$ for sufficiently large polynomial t.

Encoding Flat Samplable Distributions

Encoding Lemma: Let C be a circuit sampling a distribution of max-entropy s. There is a circuit D of size poly($|C|$) and, for each outcome z, a string i_z of length $s + \log s + O(1)$ s. t. $D(i_z) = z$.

Proof based on hashing with linear-algebraic family.

Example: C computes a random permutation of a graph G on n vertices. Each permutation of G can be decoded from a string of length $s + \log s + O(1)$, where $s = \log(n!/|\mathrm{Aut}(G)|)$.

Worse than $\lceil s \rceil$ but suffices to show $\mathrm{KT}(y)/t \leq s + o(1)$ for sufficiently large polynomial t.

Corollary: If y is the concatenation of t independent samples from a samplable distribution p then for any sufficiently large polynomial t, $\mathrm{KT}(y)/t$ is a PAC underestimator for the entropy of p with deviation $\Delta + o(1)$ where $\Delta =$ max-entropy - min-entropy of p.

PAC Underestimating $\log|\mathrm{Aut}(\omega)|$

PAC Underestimating $\log|\mathrm{Aut}(\omega)|$

Approach: Efficient procedures with access to MKTP to compute:

(a) A list L of elements of H that generates a subgroup $\langle L\rangle$ of $\mathrm{Aut}(\omega)$ such that $\langle L\rangle = \mathrm{Aut}(\omega)$ with high probability.

PAC Underestimating $\log |\mathrm{Aut}(\omega)|$

Approach: Efficient procedures with access to MKTP to compute:

(a) A list L of elements of H that generates a subgroup $\langle L \rangle$ of $\mathrm{Aut}(\omega)$ such that $\langle L \rangle = \mathrm{Aut}(\omega)$ with high probability.

(b) A PAC underestimator for $\log |\langle L \rangle|$.

Finding Generators for Aut(ω)

Finding Generators for Aut(ω)

Use the power of MKTP to invert circuits on average.

Finding Generators for $\mathrm{Aut}(\omega)$

Use the power of MKTP to invert circuits on average.

Let C be a circuit that samples a random h and outputs $h(\omega)$.

Finding Generators for $\mathsf{Aut}(\omega)$

Use the power of MKTP to invert circuits on average.

Let C be a circuit that samples a random h and outputs $h(\omega)$.

Consider sampling h' uniformly from H, and running $M(C, h'(\omega))$

On success, we get h so that $h(\omega) = h'(\omega)$, *i.e.*, $h^{-1}h' \in \mathsf{Aut}(\omega)$

In fact, conditioned on M's success, $h^{-1}h'$ is *uniform* on $\mathsf{Aut}(\omega)$

Only polynomially-many trials are needed to obtain a full set of generators with high probability.

PAC Undestimating $\log |\langle L \rangle|$

PAC Undestimating $\log |\langle L \rangle|$

By the Entropy Estimator Corollary, it suffices to show how to sample almost uniformly from $\langle L \rangle$ using a small circuit *without access to* MKTP.

PAC Undestimating $\log |\langle L \rangle|$

By the Entropy Estimator Corollary, it suffices to show how to sample almost uniformly from $\langle L \rangle$ using a small circuit *without access to* MKTP.

A sequence $h_1, h_2, \ldots, h_k$ of elements of a finite group Γ is said to be *Erdős–Rényi* if the random subproduct

$$h_1^{r_1} h_2^{r_2} \cdots h_k^{r_k}, \quad r_i \sim \{0, 1\}$$

is distributed uniformly on Γ.

PAC Undestimating $\log |\langle L \rangle|$

By the Entropy Estimator Corollary, it suffices to show how to sample almost uniformly from $\langle L \rangle$ using a small circuit *without access to* MKTP.

A sequence $h_1, h_2, \ldots, h_k$ of elements of a finite group Γ is said to be *Erdős–Rényi* if the random subproduct

$$h_1^{r_1} h_2^{r_2} \cdots h_k^{r_k}, \quad r_i \sim \{0, 1\}$$

is distributed uniformly on Γ.

[Erdős Rényi] showed that every finite group has such a generating set of size $\mathrm{poly}(\log |\Gamma|)$, and [Babai] gave an efficient randomize algorithm to find one with overwhelming probability. This gives the required circuit.

Generic Result

Generic Result

Theorem. Let Iso be an Isomorphism Problem satisfying the following conditions:

- The action $(h, \omega) \mapsto h(\omega)$, products and inverses in H, and $|H|$ can be computed efficiently.
- The uniform distribution on H can be sampled efficiently.
- There is an efficiently computable complete invariant for H.
- There is an efficiently computable complete invariant for Ω.

Then $\mathrm{Iso} \in \mathsf{ZPP}^{\mathrm{MKTP}}$.

Summary

Summary

Under mild conditions any Isomorphism problems reduces to MKTP under randomized reductions with zero-sided error.

Summary

Under mild conditions any Isomorphism problems reduces to MKTP under randomized reductions with zero-sided error.

The conditions are met for Discrete Log, Graph Isomorphism, Linear Code Equivalence, Permutation Group Conjugacy, and Matrix Subspace Conjugacy.

Summary

Under mild conditions any Isomorphism problems reduces to MKTP under randomized reductions with zero-sided error.

The conditions are met for Discrete Log, Graph Isomorphism, Linear Code Equivalence, Permutation Group Conjugacy, and Matrix Subspace Conjugacy.

Open problems:

- Replace MKTP by MCSP.
- Replace Isomorphism Problem by Statistical Zero Knowledge (SZK).

Summary

Under mild conditions any Isomorphism problems reduces to MKTP under randomized reductions with zero-sided error.

The conditions are met for Discrete Log, Graph Isomorphism, Linear Code Equivalence, Permutation Group Conjugacy, and Matrix Subspace Conjugacy.

Open problems:

- Replace MKTP by MCSP.
- Replace Isomorphism Problem by Statistical Zero Knowledge (SZK).

Questions?

Summary

Under mild conditions any Isomorphism problems reduces to MKTP under randomized reductions with zero-sided error.

The conditions are met for Discrete Log, Graph Isomorphism, Linear Code Equivalence, Permutation Group Conjugacy, and Matrix Subspace Conjugacy.

Open problems:

- Replace MKTP by MCSP.
- Replace Isomorphism Problem by Statistical Zero Knowledge (SZK).

Questions?

Thanks!