Complexity of Enumerating Satisfying Assignments

Heribert Vollmer

Output Sensitivity of Enumeration Complexity

- Decision: Output accept/reject: 1 bit
- Counting: Output number of solutions: a binary number
- Enumeration: Output all solutions: exponentially long

Exponential running time (in size of input) unavoidable!

$\therefore$ measure efficiency w.r.t. input and output.

Enumeration Problems

Let $R \subseteq \Sigma^* \times \Sigma^*$ be polynomially bounded, i.e.,

$$(x, y) \in R \implies |y| \in |x|^{\mathcal{O}(1)}.$$

Enumeration problem corresponding to R:

ENUM-R (or E-R)	
Instance:	$x \in \Sigma^*$
Output:	$\mathrm{Sol}_R(x) := \{ y \in \Sigma^* \mid (x, y) \in R \}$.

No requirement on the complexity of the "check" problem,
i. e., to determine, given x, y, if $(x, y) \in R$.

Enumeration Problems

Let $R \subseteq \Sigma^* \times \Sigma^*$ be polynomially bounded, i.e.,

$$(x, y) \in R \implies |y| \in |x|^{\mathcal{O}(1)}.$$

Enumeration problem corresponding to R:

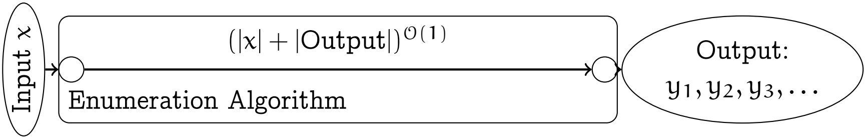
ENUM-R (or E-R)	
Instance:	$x \in \Sigma^*$
Output:	$\text{Sol}_R(x) := \{ y \in \Sigma^* \mid (x, y) \in R \}$.

No requirement on the complexity of the "check" problem, i. e., to determine, given x, y, if $(x, y) \in R$.

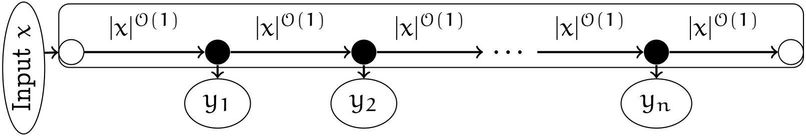
Enumeration algorithm $\mathcal{A}$ for ENUM-R:
RAM (polynomially bounded) with an output instruction, that outputs all $y \in \text{Sol}_R(x)$ without duplicates.

Measures for Enumeration Complexity

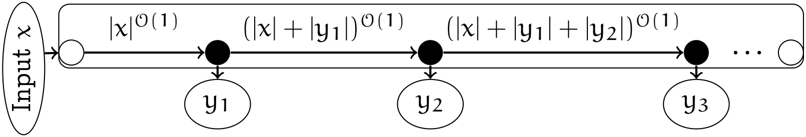
Output Polynomial Time (OutputP):



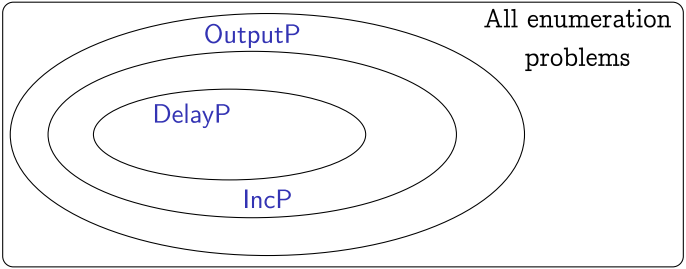
Polynomial Delay (DelayP):



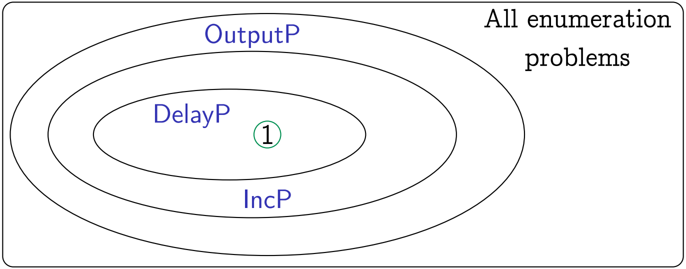
Incremental Polynomial Delay (IncP):



Tractable and Intractable Problems



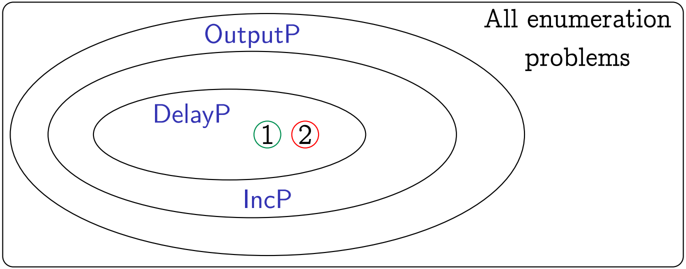
Tractable and Intractable Problems



① Maximal independent sets

[Johnson, Papadimitriou, Yannakakis, 1988]: First paper to study computational complexity of enumeration problems

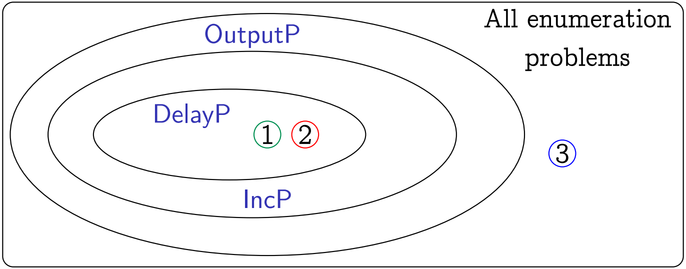
Tractable and Intractable Problems



1. Maximal independent sets
2. Satisfying assignments of Horn or Krom formulas

Flashlight search! Possible if satisfiability checkable in P

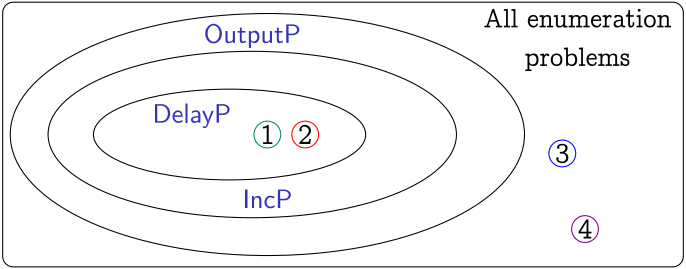
Tractable and Intractable Problems



1. Maximal independent sets
2. Satisfying assignments of Horn or Krom formulas
3. Satisfying assignments of propositional formulas

(Assuming $P \neq NP$.)

Tractable and Intractable Problems



① Maximal independent sets

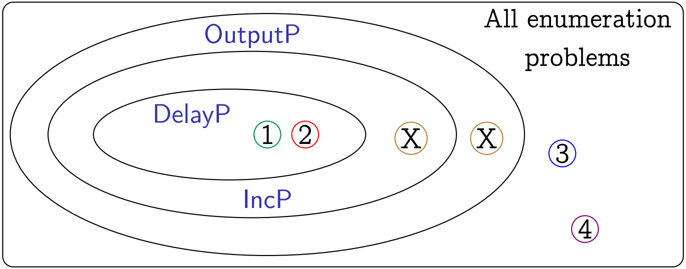
② Satisfying assignments of Horn or Krom formulas

③ Satisfying assignments of propositional formulas

④ All $\vec{y}$ s. t. $\psi = \exists x_1 \forall x_2 \dots Q_k x_k \phi(\vec{x}, \vec{y})$ is satisfiable ($\text{E-}\Sigma_k\text{SAT}$).

(Assuming $\mathsf{P} \neq \mathsf{NP}$.)

Tractable and Intractable Problems



1. Maximal independent sets
2. Satisfying assignments of Horn or Krom formulas
3. Satisfying assignments of propositional formulas
4. All $\vec{y}$ s. t. $\psi = \exists x_1 \forall x_2 \ldots Q_k x_k \phi(\vec{x}, \vec{y})$ is satisfiable ($\mathrm{E}\text{-}\Sigma_k\mathrm{SAT}$).

(Assuming $\mathsf{P} \neq \mathsf{NP}$.)

Satisfiability problems

ENUM-SAT

Input: A propositional formula/set of clauses Γ over a set of variables V

Output: an enumeration of all assignments over V that satisfy Γ

ENUM-Σ_kSAT: quantified Σ_k formula

ENUM-MONOTONE-SAT: positive (resp. negative) clauses

ENUM-IHS-SAT: monotone clauses plus implications

ENUM-XOR-SAT: clauses with xor disjunctions

ENUM-KROM-SAT: clauses of length at most 2

ENUM-HORN-SAT: clauses with at most one positive literal

Questions

I. Beyond DelayP:

Known: Enum-Sat $\notin$ DelayP unless P = NP.

Is there a "structural" result behind this?
What class corresponds to Enum-Sat?
What about quantified Boolean formulas?

II. Within DelayP:

How does the complexity of problems within DelayP compare?
Structure within DelayP?

I. Beyond DelayP

Joint work with Nadia Creignou, Markus Kröll,
Reinhard Pichler, Sebastian Skritek

(Discret. App. Math. 2019)

Enumeration with Oracles

We know that Enum-SAT $\notin$ OutputP (assuming $\mathsf{P} \neq \mathsf{NP}$).
But using an NP-decision oracle, we can enumerate satisfying assignments with a polynomial delay (flashlight search)!

New Enumeration Classes

Extend RAM model: oracle query built as concatenation of contents of a (possibly unbounded) sequence of special registers.

New Enumeration Classes

Extend RAM model: oracle query built as concatenation of contents of a (possibly unbounded) sequence of special registers.

Let $\mathcal{C}$ be a decision complexity class.

- ENUM-R $\in$ $\mathsf{Inc}\,\mathcal{C}$, if ENUM-R can be enumerated with incremental delay using a $\mathcal{C}$ oracle.

New Enumeration Classes

Extend RAM model: oracle query built as concatenation of contents of a (possibly unbounded) sequence of special registers.

Let $\mathcal{C}$ be a decision complexity class.

- Enum-R $\in \mathsf{Inc}\,\mathcal{C}$, if Enum-R can be enumerated with incremental delay using a $\mathcal{C}$ oracle.
- Enum-R $\in \mathsf{Del}^{+}\mathcal{C}$, if Enum-R can be enumerated with polynomial delay using a $\mathcal{C}$ oracle.

New Enumeration Classes

Extend RAM model: oracle query built as concatenation of contents of a (possibly unbounded) sequence of special registers.

Let $\mathcal{C}$ be a decision complexity class.

- $\textsc{Enum-R} \in \mathsf{Inc}\,\mathcal{C}$, if $\textsc{Enum-R}$ can be enumerated with incremental delay using a $\mathcal{C}$ oracle.
- $\textsc{Enum-R} \in \mathsf{Del}^{+}\mathcal{C}$, if $\textsc{Enum-R}$ can be enumerated with polynomial delay using a $\mathcal{C}$ oracle.
- $\textsc{Enum-R} \in \mathsf{Del}\,\mathcal{C}$, if $\textsc{Enum-R}$ can be enumerated with polynomial delay using a $\mathcal{C}$ oracle and the size of every oracle query is polynomially bounded in input length.

New Enumeration Classes

Extend RAM model: oracle query built as concatenation of contents of a (possibly unbounded) sequence of special registers.

Let $\mathcal{C}$ be a decision complexity class.

- $\textsc{Enum-R} \in \mathsf{Inc}\,\mathcal{C}$, if $\textsc{Enum-R}$ can be enumerated with incremental delay using a $\mathcal{C}$ oracle.
- $\textsc{Enum-R} \in \mathsf{Del}^{+}\mathcal{C}$, if $\textsc{Enum-R}$ can be enumerated with polynomial delay using a $\mathcal{C}$ oracle.
- $\textsc{Enum-R} \in \mathsf{Del}\,\mathcal{C}$, if $\textsc{Enum-R}$ can be enumerated with polynomial delay using a $\mathcal{C}$ oracle and the size of every oracle query is polynomially bounded in input length.

Oracle calls can be extended!

The Power of Growing Queries

Theorem

- For all $\mathcal{C}$, $\mathsf{Inc}^{+}\mathcal{C} = \mathsf{Inc}\,\mathcal{C}$.

The Power of Growing Queries

Theorem

- For all $\mathcal{C}$, $\mathsf{Inc}^{+}\mathcal{C} = \mathsf{Inc}\,\mathcal{C}$.
- Theorem: $\mathsf{Del}^{+}\mathsf{P} = \mathsf{IncP}$.

The Power of Growing Queries

Theorem

- For all $\mathcal{C}$, $\mathsf{Inc}^+\mathcal{C} = \mathsf{Inc}\,\mathcal{C}$.
- Theorem: $\mathsf{Del}^+\mathsf{P} = \mathsf{IncP}$.

"$\supseteq$": Let $\textsc{Enum-R} \in \mathsf{IncP}$ via $\mathcal{A}$ and fix some $x \in \{0,1\}^*$.
Let $<$ be the order on $\mathsf{Sol}_R(x)$ induced by $\mathcal{A}$.

The Power of Growing Queries

Theorem

- For all $\mathcal{C}$, $\mathsf{Inc}^+\mathcal{C} = \mathsf{Inc}\,\mathcal{C}$.
- Theorem: $\mathsf{Del}^+\mathsf{P} = \mathsf{IncP}$.

"$\supseteq$": Let $\textsc{Enum-}R \in \mathsf{IncP}$ via $\mathcal{A}$ and fix some $x \in \{0,1\}^*$.
Let $<$ be the order on $\mathsf{Sol}_R(x)$ induced by $\mathcal{A}$.

$\textsc{AnotherSolExt}_R^<$	
Instance:	$y_1, \ldots, y_n, y', x \in \Sigma^*$
Question:	Is y' a prefix of y_{n+1}, where y_{n+1} is the $(n+1)$-th element in $\mathsf{Sol}_R(x)$ w.r.t. $<$?

Since $\textsc{Enum-}R \in \mathsf{IncP}$, $\textsc{AnotherSolExt}_R^< \in \mathsf{P}$.

The Power of Growing Queries

Theorem

- For all $\mathcal{C}$, $\mathsf{Inc}^{+}\mathcal{C} = \mathsf{Inc}\,\mathcal{C}$.
- Theorem: $\mathsf{Del}^{+}\mathsf{P} = \mathsf{IncP}$.

"$\supseteq$": Let $\textsc{Enum-R} \in \mathsf{IncP}$ via $\mathcal{A}$ and fix some $x \in \{0,1\}^*$.
Let $<$ be the order on $\mathsf{Sol}_R(x)$ induced by $\mathcal{A}$.

$\textsc{AnotherSolExt}_R^{<}$	
Instance:	$y_1, \ldots, y_n, y', x \in \Sigma^*$
Question:	Is y' a prefix of y_{n+1}, where y_{n+1} is the $(n+1)$-th element in $\mathsf{Sol}_R(x)$ w.r.t. $<$?

Since $\textsc{Enum-R} \in \mathsf{IncP}$, $\textsc{AnotherSolExt}_R^{<} \in \mathsf{P}$.

But: $\textsc{Enum-R} \in \mathsf{DelayP}^{\textsc{AnotherSolExt}_R^{<}}$.

The Power of Growing Queries

Theorem

- For all $\mathcal{C}$, $\mathsf{Inc}^{+}\mathcal{C} = \mathsf{Inc}\,\mathcal{C}$.
- Theorem: $\mathsf{Del}^{+}\mathsf{P} = \mathsf{IncP}$.

"$\supseteq$": Let $\textsc{Enum-R} \in \mathsf{IncP}$ via $\mathcal{A}$ and fix some $x \in \{0,1\}^*$.
Let $<$ be the order on $\mathsf{Sol}_R(x)$ induced by $\mathcal{A}$.

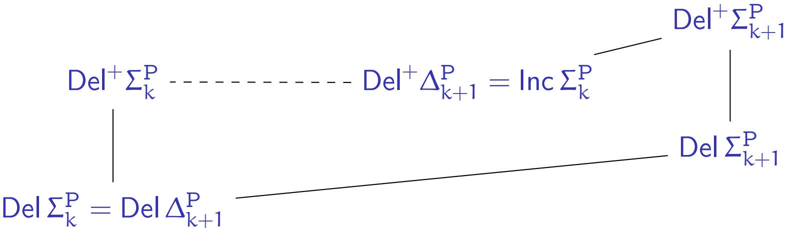
$\textsc{AnotherSolExt}_R^<$	
Instance:	$y_1, \ldots, y_n, y', x \in \Sigma^*$
Question:	Is y' a prefix of y_{n+1}, where y_{n+1} is the $(n+1)$-th element in $\mathsf{Sol}_R(x)$ w.r.t. $<$?

Since $\textsc{Enum-R} \in \mathsf{IncP}$, $\textsc{AnotherSolExt}_R^< \in \mathsf{P}$.

But: $\textsc{Enum-R} \in \mathsf{DelayP}^{\textsc{AnotherSolExt}_R^<}$.

In general: $\mathsf{Del}^{+}\Delta_{k+1}^{\mathsf{P}} = \mathsf{Inc}\,\Sigma_k^{\mathsf{P}}$ for all $k \geq 0$.

Hierarchy of Enumeration Complexity Classes



- Under the assumptions that the polynomial hierarchy does not collapse to Σ^P_{k+1} and $\mathsf{EXP} \subsetneq \Delta^{\mathrm{EXP}}_{k+1}$, all strict lines denote strict inclusions and there are no further inclusions.
- The dashed line denotes inclusion, not known to be strict.

A Conditional Separation

Theorem: Let $k \geq 1, \ell \geq 0$. Suppose $\mathsf{EXP} \subsetneq \Delta_{k+1}^{\mathsf{EXP}}$. Then,

$$\mathsf{Del}\,\Sigma_k^{\mathsf{P}} \subsetneq \mathsf{Del}^{+}\Sigma_k^{\mathsf{P}} \not\subseteq \mathsf{Del}\,\Sigma_\ell^{\mathsf{P}}.$$

A Conditional Separation

Theorem: Let $k \geq 1, \ell \geq 0$. Suppose $\mathsf{EXP} \subsetneq \Delta_{k+1}^{\mathsf{EXP}}$. Then,

$$\mathsf{Del}\,\Sigma_k^{\mathsf{P}} \subsetneq \mathsf{Del}^{+}\Sigma_k^{\mathsf{P}} \not\subseteq \mathsf{Del}\,\Sigma_\ell^{\mathsf{P}}.$$

Proof. Let $L \in \Delta_{k+1}^{\mathsf{EXP}} \setminus \mathsf{EXP}$ be decided in time $\mathcal{O}(2^{q(n)})$ using a Σ_k^{P}-oracle A.

ENUM-$D_0(L, q)$	
Instance:	$x \in \Sigma^*$
Output:	all $\{0, 1\}$-words of length $q(\lvert x\rvert)$, and 2 if $x \in L$

A Conditional Separation

Theorem: Let $k \geq 1, \ell \geq 0$. Suppose $\mathsf{EXP} \subsetneq \Delta_{k+1}^{\mathsf{EXP}}$. Then,

$$\mathsf{Del}\,\Sigma_k^{\mathsf{P}} \subsetneq \mathsf{Del}^{+}\Sigma_k^{\mathsf{P}} \not\subseteq \mathsf{Del}\,\Sigma_\ell^{\mathsf{P}}.$$

Proof. Let $L \in \Delta_{k+1}^{\mathsf{EXP}} \setminus \mathsf{EXP}$ be decided in time $\mathcal{O}(2^{q(n)})$ using a Σ_k^{P}-oracle A.

Enum-$D_0(L, q)$	
Instance:	$x \in \Sigma^*$
Output:	all $\{0, 1\}$-words of length $q(\|x\|)$, and 2 if $x \in L$

- Enum-$D_0(L, q) \in \mathsf{Del}^{+}\Sigma_k^{\mathsf{P}}$.
  Algorithm $\mathcal{A}$ enumerates all $2^{q(|x|)}$ words in $\{0, 1\}^{q(|x|)}$ in $\mathcal{O}(2^{q(|x|)})$ and in parallel decides $x \in L$ using oracle A.

A Conditional Separation

Theorem: Let $k \geq 1, \ell \geq 0$. Suppose $\mathsf{EXP} \subsetneq \Delta_{k+1}^{\mathsf{EXP}}$. Then,

$$\mathsf{Del}\,\Sigma_k^{\mathsf{P}} \subsetneq \mathsf{Del}^{+}\Sigma_k^{\mathsf{P}} \not\subseteq \mathsf{Del}\,\Sigma_\ell^{\mathsf{P}}.$$

Proof. Let $L \in \Delta_{k+1}^{\mathsf{EXP}} \setminus \mathsf{EXP}$ be decided in time $\mathcal{O}(2^{q(n)})$ using a Σ_k^{P}-oracle A.

ENUM-$D_0(L, q)$	
Instance:	$x \in \Sigma^*$
Output:	all $\{0,1\}$-words of length $q(\lvert x\rvert)$, and 2 if $x \in L$

- ENUM-$D_0(L, q) \in \mathsf{Del}^{+}\Sigma_k^{\mathsf{P}}$.
- ENUM-$D_0(L, q) \notin \mathsf{Del}\,\Sigma_\ell^{\mathsf{P}}$.
  Suppose contrary, then $(x, 2) \in D_0$ can be checked in EXPtime by enumerating all solutions, hence $L \in \mathsf{EXP}$, contradiction.

A Conditional Separation

Theorem: Let $k \geq 1, \ell \geq 0$. Suppose $\mathsf{EXP} \subsetneq \Delta_{k+1}^{\mathsf{EXP}}$. Then,

$$\mathsf{Del}\,\Sigma_k^{\mathsf{P}} \subsetneq \mathsf{Del}^+\Sigma_k^{\mathsf{P}} \not\subseteq \mathsf{Del}\,\Sigma_\ell^{\mathsf{P}}.$$

Proof. Let $L \in \Delta_{k+1}^{\mathsf{EXP}} \setminus \mathsf{EXP}$ be decided in time $\mathcal{O}(2^{q(n)})$ using a Σ_k^{P}-oracle A.

ENUM-$D_0(L, q)$	
Instance:	$x \in \Sigma^*$
Output:	all $\{0,1\}$-words of length $q(\lvert x \rvert)$, and 2 if $x \in L$

- ENUM-$D_0(L, q) \in \mathsf{Del}^+\Sigma_k^{\mathsf{P}}$.
- ENUM-$D_0(L, q) \notin \mathsf{Del}\,\Sigma_\ell^{\mathsf{P}}$.

□

Note: Generalizes $\mathsf{Del}\,\mathsf{P} \subsetneq \mathsf{Inc}\,\mathsf{P}$.

Self-Reducibility

Clearly, if $\textsc{Enum-R} \in \mathsf{Del}\,\Sigma_k^P$ then $\textsc{Exist-R} \in \Delta_k^P$.

Self-Reducibility

Clearly, if $\textsc{Enum-R} \in \mathsf{Del}\,\Sigma_k^P$ then $\textsc{Exist-R} \in \Delta_k^P$.

ExtSol-R	
Instance:	$(x, y) \in \Sigma^* \times \Sigma^*$
Output:	Is there $y' \in \Sigma^*$ such that $(x, yy') \in R$?

Note: $\textsc{Enum-R} \in \mathsf{DelayP}^{\textsc{ExtSol-R}}$.

Self-Reducibility

Clearly, if ENUM-R $\in \mathsf{Del}\,\Sigma_k^P$ then EXIST-R $\in \Delta_k^P$.

EXTSOL-R	
Instance:	$(x, y) \in \Sigma^* \times \Sigma^*$
Output:	Is there $y' \in \Sigma^*$ such that $(x, yy') \in R$?

Note: ENUM-R $\in \mathsf{DelayP}^{\text{EXTSOL-R}}$.

Definition: R is self-reducible if EXTSOL-R $\leq_T$ EXIST-R.

Theorem: Let R be self-reducible. Then

EXIST-R $\in \Delta_k^P$ if and only if ENUM-R $\in \mathsf{Del}\,\Sigma_k^P$.

Self-Reducibility

> ANOTHERSOL-R
> Instance: $x \in \Sigma^*, Y \subseteq R(x)$
> Output: $y \in R(x) \setminus Y$ or declare that no such y exists.

Definition: R is enumeration self-reducible if
ANOTHERSOL-R $\leq_T$ EXIST-ANOTHERSOL-R.

[Kimelfeld, Kolaitis, 2014]

Theorem:

- ANOTHERSOL-R can be solved in polynomial time with access to a Δ_k^P-oracle if and only if ENUM-R $\in \mathsf{Inc}\,\Sigma_k^P$.

  [cf. Capelli, Strozecki, 2018]

- Let R be enumeration self-reducible. Then EXIST-ANOTHERSOL-R $\in \Delta_k^P$ iff ENUM-R $\in \mathsf{Inc}\,\Sigma_k^P$.

  [cf. Kimelfeld, Kolaitis, 2014]

What Next?

- We have (likely strict) hierarchies of (hard) enumeration classes.
- We have methods to prove membership in these classes.

What Next?

- We have (likely strict) hierarchies of (hard) enumeration classes.

- We have methods to prove membership in these classes.

Any complete enumeration problems?

Reduction among Enumeration Problems

What do we want from a reduction?

- If Enum-$R_1 \leq$ Enum-R_2 and if we can enumerate Enum-R_2, then we can enumerate Enum-R_1.
- Relevant classes are closed under reduction.
- Reduction is transitive.
- Allows for (natural?) complete problems.

Enumeration Oracle Machines

Enumeration algorithm (RAM) with ENUM-R oracle,
i.e., enumeration oracle, not only language (yes-no) oracle

- oracle query built as concatenation of contents of a (possibly unbounded) sequence of special registers
- oracle call NOO ("next oracle output"): produces next element from $\mathsf{Sol}_R(x)$ or information that there is none, where x is oracle query
- oracle bounded: size of oracle query at most polynomial in size of input

Turing Style Reductions

Definition (Reductions $\leq_{\mathrm{D}}$, $\leq_{\mathrm{I}}$)

$\textsc{Enum-R}_1 \leq_x \textsc{Enum-R}_2$ (for $x \in \{\mathrm{D}, \mathrm{I}\}$) if there is a RAM $\mathcal{A}$ with oracle $\textsc{Enum-R}_2$ such that, independent of the order in which the $\textsc{Enum-R}_2$ oracle enumerates its answers,

- $\mathcal{A}$ enumerates $\textsc{Enum-R}_1$ in DelayP and is oracle-bounded, for $x = \mathrm{D}$.
- $\mathcal{A}$ enumerates $\textsc{Enum-R}_1$ in IncP, for $x = \mathrm{I}$.

Properties of $\leq_{\mathrm{D}}$ and $\leq_{\mathrm{I}}$

- $\leq_{\mathrm{D}}$ and $\leq_{\mathrm{I}}$ are transitive,
- classes $\mathsf{Del}\,\mathcal{C}$ are closed under $\leq_{\mathrm{D}}$,
- classes $\mathsf{Inc}\,\mathcal{C}$ are closed under $\leq_{\mathrm{I}}$

(for every class $\mathcal{C}$ in PH).

Completeness Theorem

Let $R \subseteq \Sigma^* \times \Sigma^*$ and $k \geq 1$ such that Exist-R is Σ_k^P-complete.

Then Enum-R is

- Del Σ_k^P-hard under $\leq_D$ reductions,
- Inc Σ_k^P-hard under $\leq_I$ reductions.

Completeness Theorem

Let $R \subseteq \Sigma^* \times \Sigma^*$ and $k \geq 1$ such that Exist-R is Σ_k^P-complete.

Then Enum-R is

- Del Σ_k^P-hard under $\leq_D$ reductions,
- Inc Σ_k^P-hard under $\leq_I$ reductions.

If additionally R is self-reducible, then Enum-R is

- Del Σ_k^P-complete under $\leq_D$ reductions,
- Inc Σ_k^P-complete under $\leq_I$ reductions.

Complete Problems

$\therefore$ E-Σ_kSAT is complete for $\mathsf{Del}\,\Sigma_k^P$ under $\leq_D$ reductions.

$\therefore$ E-Σ_kSAT is complete for $\mathsf{Inc}\,\Sigma_k^P$ under $\leq_I$ reductions.

In particular: ENUM-SAT is complete for $\mathsf{Del}\,\mathsf{NP}$.

Complete Problems

∴ E-Σ_kSAT is complete for Del Σ_k^P under $\leq_D$ reductions.

∴ E-Σ_kSAT is complete for Inc Σ_k^P under $\leq_I$ reductions.

In particular: ENUM-SAT is complete for Del NP.

∴ E-Π_{k-1}SAT is complete for Del Σ_k^P under $\leq_D$ reductions.

∴ E-Π_{k-1}SAT is complete for Inc Σ_k^P under $\leq_I$ reductions.

Boolean Constraint Satisfaction Problems

- Clear: If $\Gamma \in \{\text{Horn}, \text{dualHorn}, \text{bijunctive}, \text{affine}\}$, then Enum-SAT($\Gamma$) $\in$ DelayP.
- Otherwise:
  - Observe that not necessarily, SAT(Γ) is NP-hard (namely if Γ is 1-valid or 0-valid)!
  - But we can show: AnotherSol_SAT(Γ) is NP-complete.

$\therefore$ Enum-SAT(Γ) is Del NP-complete under $\leq_{\mathrm{D}}$.

Further Results

problem	member	$\leq_D$-hardness	$\leq_I$-hardness
E-Σ_kSAT	Del Σ_k^P	Del Σ_k^P	Inc Σ_k^P
E-Π_kSAT	Del Σ_{k+1}^P	Del Σ_{k+1}^P	Inc Σ_{k+1}^P
E-CIRCUMSCRIPTION	Del^+ NP	Del NP	Inc NP
E-CARDMINSAT	Del NP	Del NP	Inc NP
E-MODBASEDDIAGNOSIS	Del NP	Del NP	Inc NP
E-ABDUCTION	Del Σ_2^P	Del Σ_2^P	Inc Σ_2^P
E-REPAIR	Del Σ_2^P		Inc Σ_2^P

Example: Minimal Satisfiability

E-CardMinSAT

Instance: ϕ a Boolean formula

Output: All cardinality-minimal satisfying assignments of ϕ.

Theorem: E-CardMinSAT is Del NP-complete under $\leq_{\mathrm{D}}$.

Example: Minimal Satisfiability

E-CardMinSAT	
Instance:	ϕ a Boolean formula
Output:	All cardinality-minimal satisfying assignments of ϕ.

Theorem: E-CardMinSAT is DelNP-complete under $\leq_D$.

Hardness follows from Completeness Theorem.

Example: Minimal Satisfiability

E-CardMinSAT	
Instance:	ϕ a Boolean formula
Output:	All cardinality-minimal satisfying assignments of ϕ.

Theorem: E-CardMinSAT is Del NP-complete under $\leq_D$.

Hardness follows from Completeness Theorem.

Membership:

- Compute first the minimal cardinality of models in ptime with NP-oracle.
- Enumerate all cardinality-minimal models by the standard binary search tree with an NP-oracle.

□

Example: Abduction

E-ABDUCTION	
Instance:	Γ a set of formulæ (knowledge),
	H a set of literals (hypotheses),
	q a variable, $q \notin H$ (manifestation)
Output:	all sets $E \subseteq H$ such that $\Gamma \wedge E$ is satisfiable and
	$\Gamma \wedge E \models q$ (explantion).

Theorem: E-ABDUCTION is Del Σ_2^P-complete under $\leq_D$.

Example: Abduction

E-ABDUCTION	
Instance:	Γ a set of formulæ (knowledge),
	H a set of literals (hypotheses),
	q a variable, $q \notin H$ (manifestation)
Output:	all sets $E \subseteq H$ such that $\Gamma \wedge E$ is satisfiable and
	$\Gamma \wedge E \models q$ (explantion).

Theorem: E-ABDUCTION is Del Σ_2^P-complete under $\leq_D$.

Hardness follows from Completeness Theorem, since the decision problem is Σ_2^P-complete [Creignou, Zanuttini, 2006].

Example: Abduction

E-ABDUCTION

Instance: Γ a set of formulæ (knowledge),
H a set of literals (hypotheses),
q a variable, $q \notin H$ (manifestation)

Output: all sets $E \subseteq H$ such that $\Gamma \wedge E$ is satisfiable and $\Gamma \wedge E \models q$ (explantion).

Theorem: E-ABDUCTION is Del Σ_2^P-complete under $\leq_D$.

Hardness follows from Completeness Theorem, since the decision problem is Σ_2^P-complete [Creignou, Zanuttini, 2006].

Membership: Given (Γ, H, q) and $E \subseteq H$, E can be extended to an explanation iff $(\Gamma \wedge E, H, q) \in$ ABDUCTION.

$\therefore$ E-ABDUCTION is self-reducible

$\therefore$ E-ABDUCTION $\in$ Del Σ_2^P. $\square$

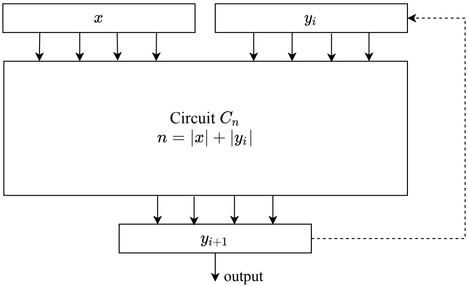
II. Within DelayP

Joint work with Nadia Creignou and Arnaud Durand

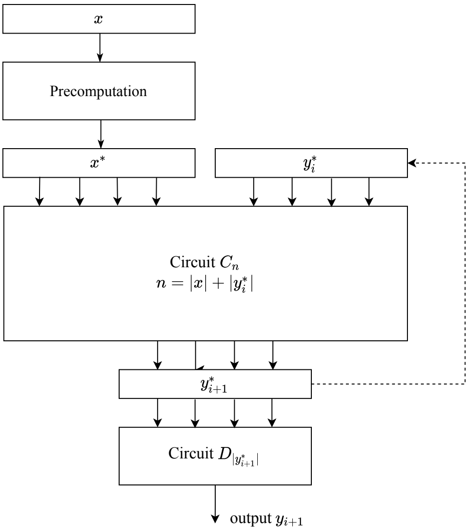
MFCS 2022

Circuit Enumeration – Simplistic View

- Starting point: a family of circuits (C_n) of a given kind.
- x is the input.
- Successive outputs serve as auxiliary input.



Circuit Enumeration with Precomputation and Memory



Formal Definitions

Let T be a complexity class and $\mathcal{K}$ a class of circuits.

Definition ($\mathcal{K}$-delay with T-precomputation)

For a predicate R, Enum-$R \in \mathrm{Del}_T \cdot \mathcal{K}$ if there exists an algorithm M working with resource T and a family of $\mathcal{K}$-circuits $\mathcal{C} = (C_n)_{n \in \mathbb{N}}$ such that, for all input x there is an enumeration $y_1, ..., y_k$ of $\mathrm{Sol}_R(x)$ and:

- M compute some value x^*, i.e., $M(x) = x^*$,
- $C_{|\cdot|}(x^*) = y_1 \in \mathrm{Sol}_R(x)$,
- for all $i < k$: $C_{|\cdot|}(x^*, y_i) = y_{i+1} \in \mathrm{Sol}_R(x)$,
- $C_{|\cdot|}(x^*, y_k) = y_k$.

Formal Definitions

Definition ($\mathcal{K}$-delay with T-precomputation and memory)

$\textsc{Enum}\text{-}R \in \mathsf{Del}^{*}_{T}\cdot\mathcal{K}$ if there exists an algorithm M working with resource T and two families of $\mathcal{K}$-circuits $\mathcal{C} = (C_n)_{n\in\mathbb{N}}$, $\mathcal{D} = (D_n)_{n\in\mathbb{N}}$ such that, for all input x there is an enumeration $y_1, ..., y_k$ of $\mathrm{Sol}_R(x)$ and:

- M computes some value x^*, i.e., $M(x) = x^*$,
- $C_{|\cdot|}(x^*) = y_1^*$ and $D_{|\cdot|}(y_1^*) = y_1 \in \mathrm{Sol}_R(x)$,
- $\forall i < k$: $C_{|\cdot|}(x^*, y_i^*) = y_{i+1}^*$ and $D_{|\cdot|}(y_{i+1}^*) = y_{i+1} \in \mathrm{Sol}_R(x)$,
- $C_{|\cdot|}(x^*, y_k^*) = y_k^*$.

$\mathsf{Del}^{c}_{T}\cdot\mathcal{K}$ (resp. $\mathsf{Del}^{p}_{T}\cdot\mathcal{K}$): constant (resp. polynomial) size memory.

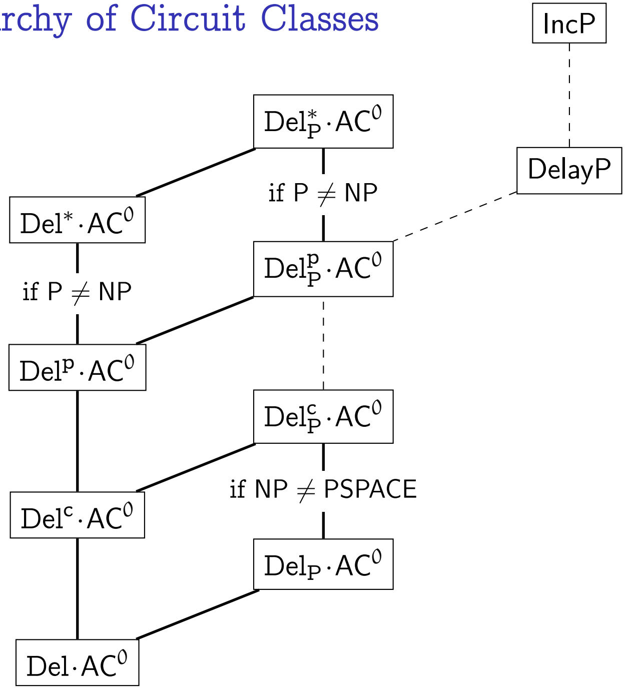
Classes under Study

Main circuit engines: AC^0 circuits/languages,
i.e., uniform families of Boolean circuits of polynomial size and constant depth with gates of unbounded fan-in.

- $\mathsf{Del}\cdot\mathsf{AC}^0$: no precomputation, no memory
- $\mathsf{Del}^c\cdot\mathsf{AC}^0$: no precomputation, constant size memory
- $\mathsf{Del}^p\cdot\mathsf{AC}^0$: no precomputation, polynomial size memory
- $\mathsf{Del}^*\cdot\mathsf{AC}^0$: no precomputation, unbounded memory

- $\mathsf{Del}_{\mathsf{P}}\cdot\mathsf{AC}^0$: polytime precomputation, no memory
- $\mathsf{Del}^c_{\mathsf{P}}\cdot\mathsf{AC}^0$: polytime precomputation, constant size memory
- $\mathsf{Del}^p_{\mathsf{P}}\cdot\mathsf{AC}^0$: polytime precomputation, polysize memory
- $\mathsf{Del}^*_{\mathsf{P}}\cdot\mathsf{AC}^0$: polytime precomputation, unbounded memory

A Hierarchy of Circuit Classes



Comparisons with Constant Delay

- CD∘lin: problems that can be enumerated on RAMs with constant delay after linear time preprocessing
- Classes $\mathrm{Del}\cdot\mathrm{AC}^0$ and CD∘lin are incomparable
  - Output the parity of the number of 1 is in CD∘lin (and not in $\mathrm{Del}\cdot\mathrm{AC}^0$)
  - Enumerate the 1 entries of $A \times B$ with A, B Boolean matrices is in $\mathrm{Del}\cdot\mathrm{AC}^0$
    (but, assuming the BMM hypothesis, not in CD∘lin)
- $\mathrm{CD}\circ\mathrm{lin} \subsetneq \mathrm{Del}^{\mathrm{p}}_{\mathrm{lin}}\cdot\mathrm{AC}^0$.

The Unreasonable Effectiveness of Unbounded Memory

Theorem: Let R be a polynomially balanced relation such that $\textsc{Enum}\text{-}R \in \mathsf{IncP}$ and R is AC^0-checkable.

Then $\textsc{Enum}\text{-}R \in \mathsf{Del}^*_{\mathsf{P}}{\cdot}\mathsf{AC}^0$.

The Unreasonable Effectiveness of Unbounded Memory

Theorem: Let R be a polynomially balanced relation such that
$\textsc{Enum-R} \in \mathsf{IncP}$ and R is AC^0-checkable.
Then $\textsc{Enum-R} \in \mathsf{Del}^*_{\mathsf{P}}{\cdot}\mathsf{AC}^0$.

Proof idea for $\textsc{Enum-Krom-SAT} \in \mathsf{Del}^*_{\mathsf{P}}{\cdot}\mathsf{AC}^0$:

- Use preprocessing to compute a polynomial number of solutions.
- While outputting these, use unbounded memory to build tree of all assignments.
- Mark satisfying assignments.
- Output these one after the other.

The Unreasonable Effectiveness of Unbounded Memory

Theorem: Let R be a polynomially balanced relation such that $\textsc{Enum-R} \in \mathsf{IncP}$ and R is AC^0-checkable.
Then $\textsc{Enum-R} \in \mathsf{Del}_{\mathsf{P}}^{*}{\cdot}\mathsf{AC}^0$.

Proof idea for $\textsc{Enum-Krom-SAT} \in \mathsf{Del}_{\mathsf{P}}^{*}{\cdot}\mathsf{AC}^0$:

- Use preprocessing to compute a polynomial number of solutions.
- While outputting these, use unbounded memory to build tree of all assignments.
- Mark satisfying assignments.
- Output these one after the other.

With a similar proof: If $\mathsf{P} \neq \mathsf{NP}$, then

- $\mathsf{Del}_{\mathsf{P}}^{\mathsf{p}}{\cdot}\mathsf{AC}^0 \subsetneq \mathsf{Del}_{\mathsf{P}}^{*}{\cdot}\mathsf{AC}^0$.
- $\mathsf{Del}^{\mathsf{p}}{\cdot}\mathsf{AC}^0 \subsetneq \mathsf{Del}^{*}{\cdot}\mathsf{AC}^0$.

Enumerating Satisfying Assignments Using Circuits

- ENUM-MONOTONE-SAT $\in \mathsf{Del}\cdot\mathsf{AC}^0$.

  Note: same as transversals for hypergraph.

Enumerating Satisfying Assignments Using Circuits

- Enum-Monotone-SAT $\in \mathsf{Del}\cdot\mathsf{AC}^0$.
- Enum-IHS-SAT $\in \mathsf{Del}_\mathsf{P}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.

Enumerating Satisfying Assignments Using Circuits

- ENUM-MONOTONE-SAT $\in \mathrm{Del}\cdot\mathrm{AC}^0$.
- ENUM-IHS-SAT $\in \mathrm{Del}_{\mathrm{P}}\cdot\mathrm{AC}^0 \setminus \mathrm{Del}^*\cdot\mathrm{AC}^0$.
- ENUM-KROM-SAT $\in \mathrm{Del}_{\mathrm{P}}\cdot\mathrm{AC}^0 \setminus \mathrm{Del}^*\cdot\mathrm{AC}^0$.

  Lower bound by reduction from ST-CONNECTIVITY.
  Upper bound: Precomputate reachability and topological order in implication graph; then use refinement of algorithm by [Aspvall, Plass, Tarjan, 1979]. No memory is ever needed.

Enumerating Satisfying Assignments Using Circuits

- Enum-Monotone-SAT $\in \mathsf{Del}\cdot\mathsf{AC}^0$.
- Enum-IHS-SAT $\in \mathsf{Del}_\mathsf{P}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.
- Enum-Krom-SAT $\in \mathsf{Del}_\mathsf{P}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.
- Enum-XOR-SAT $\in \mathsf{Del}_\mathsf{P}^\mathsf{c}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.

  Upper bound: Gaussian elimination + Gray code enumeration.
  Lower bound: Parity can be expressed.

Enumerating Satisfying Assignments Using Circuits

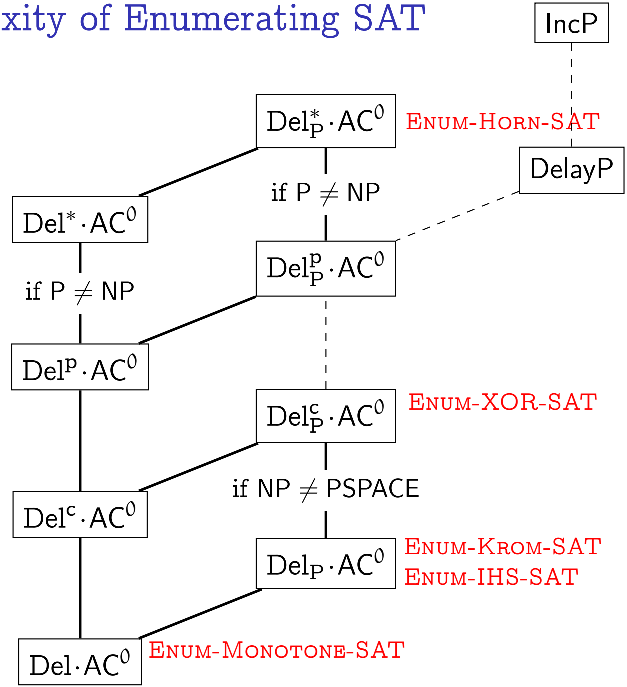
- Enum-Monotone-SAT $\in \mathrm{Del}\cdot\mathsf{AC}^0$.
- Enum-IHS-SAT $\in \mathrm{Del}_{\mathsf{P}}\cdot\mathsf{AC}^0 \setminus \mathrm{Del}^*\cdot\mathsf{AC}^0$.
- Enum-Krom-SAT $\in \mathrm{Del}_{\mathsf{P}}\cdot\mathsf{AC}^0 \setminus \mathrm{Del}^*\cdot\mathsf{AC}^0$.
- Enum-XOR-SAT $\in \mathrm{Del}^{\mathsf{c}}_{\mathsf{P}}\cdot\mathsf{AC}^0 \setminus \mathrm{Del}^*\cdot\mathsf{AC}^0$.
- Enum-Horn-SAT $\in \mathrm{Del}^{*}_{\mathsf{P}}\cdot\mathsf{AC}^0 \setminus \mathrm{Del}^*\cdot\mathsf{AC}^0$.

  Build tree of all possible assignment in memory.

Enumerating Satisfying Assignments Using Circuits

- Enum-Monotone-SAT $\in \mathsf{Del}\cdot\mathsf{AC}^0$.
- Enum-IHS-SAT $\in \mathsf{Del}_\mathsf{P}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.
- Enum-Krom-SAT $\in \mathsf{Del}_\mathsf{P}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.
- Enum-XOR-SAT $\in \mathsf{Del}^\mathsf{c}_\mathsf{P}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.
- Enum-Horn-SAT $\in \mathsf{Del}^*_\mathsf{P}\cdot\mathsf{AC}^0 \setminus \mathsf{Del}^*\cdot\mathsf{AC}^0$.

Complexity of Enumerating SAT



Main Open Question

- It is known that ENUM-HORN-SAT $\in \mathsf{DelayP} \cap \mathsf{Del}^*_{\mathsf{P}} \cdot \mathsf{AC}^0$, and it seems to be harder than ENUM-KROM-SAT.
  Open: Where is ENUM-HORN-SAT in this hierarchy of circuit classes?
- Completeness results? Reductions?
  Many classes here are promise classes and not known to possess complete problems.

Main Open Question

- It is known that ENUM-HORN-SAT $\in \mathsf{DelayP} \cap \mathsf{Del}^*_{\mathsf{P}} \cdot \mathsf{AC}^0$, and it seems to be harder than ENUM-KROM-SAT.
  Open: Where is ENUM-HORN-SAT in this hierarchy of circuit classes?
- Completeness results? Reductions?
  Many classes here are promise classes and not known to possess complete problems.

Thank you!

Contents