Efficient Enumeration Algorithms for Constraint Satisfaction Problems

Henning and Ilka Schnoor

Institut für Theoretische Informatik

Leibniz Universität Hannover

2.10.2006

Enumeration Problems

Task

Given a constraint-formula φ, enumerate its set of solutions.

Question

What is an “efficient” enumeration algorithm?

Definition

Efficient enumeration algorithm: Input φ, print solutions:

- Each solution is printed exactly once.
- Time between two solutions is polynomial.

Notation

- Γ is a constraint language
- A Γ-formula is of the form

$$\varphi = \bigwedge_{i=1}^{n} R_i(x_1^i, \ldots, x_{k_i}^i)$$

  for relations $R_1, \ldots, R_n \in \Gamma$.
- Similarly: R-formulas.

An algorithm for the Boolean case

- Let φ be a formula with variables $x_1, \ldots, x_n$.
- If $\varphi \land x_1 \in \text{SAT}$: enumerate all solutions of $\varphi[x_1/1]$
- If $\varphi \land \overline{x_1} \in \text{SAT}$: enumerate all solutions of $\varphi[x_1/0]$

Theorem (Creignou, Hébrard, 1997)

- *In the Boolean case, there is no other algorithm.*

A Simple Algorithm

- Algorithm needs decision for "Γ-formulas with literals."
- Works if $\mathrm{CSP}\left(\Gamma \cup \{\{(0)\}, \{(1)\}\}\right) \in \mathrm{P}$.
- Generalization to non-Boolean case:

$$\Gamma^{+} := \Gamma \cup \{\{(\alpha)\} \mid \alpha \in D\}.$$

- Γ^{+} is "Γ with literals"

Theorem (Cohen, 2004)

If CSP $(\Gamma^{+}) \in \mathrm{P}$*, then* Γ *has an efficient enumeration algorithm.*

- For the Boolean case, the converse holds as well (Creignou, Hébrard, 1997).
- What about arbitrary domains?

- A weaker version of the converse does hold.

Theorem

Equivalent:

1. CSP $(\Gamma^+) \in \mathrm{P}$.
2. *Solutions* Γ*-formuls can be enumerated in lexicographical ordering, with different order for each variable.*

- For x_1, we demand $0 < 2 < 4 < 3 < 1$
- For x_2, we demand $4 < 3 < 1 < 0 < 2$ ...

Proposition

- *If* $\mathrm{CSP}\,(\Gamma^{+}) \in \mathrm{P}$, *there is an efficient enumeration algorithm.*
- *If* $\mathrm{CSP}\,(\Gamma) \notin \mathrm{P}$, *there is no efficient enumeration algorithm.*

- Look at Γ where $\mathrm{CSP}\,(\Gamma) \in \mathrm{P}$, and $\mathrm{CSP}\,(\Gamma^{+}) \notin \mathrm{P}$.

Lemma

Let CSP $(\Gamma) \in \mathrm{P}$, CSP $(\Gamma^{+}) \notin \mathrm{P}$. *There is a unary* $f \in \mathrm{Pol}(\Gamma)$ *such that* f *is not injective and* $f \circ f = f$.

For the $|D| = 3$ this means this is a constant or of the form $f_{a \rightarrow b}$:

$$\begin{aligned} f_{a \rightarrow b}(a) &= b \\ f_{a \rightarrow b}(b) &= b \\ f_{a \rightarrow b}(c) &= c \end{aligned}$$

Example

Let $R := \{(2,0,0,2), (2,0,2,0), (2,2,0,0),$
$(1,0,0,1), (1,0,1,0), (1,1,0,0), (1,0,0,0)\}.$

R has an eff. enumeration algorithm, CSP (R^+) is NP-complete.

- $f_{2\to 1}$ is a polymorphism of R.

$S := R \cap \{0,1\}^4 = \{(1,0,0,1), (1,0,1,0), (1,1,0,0), (1,0,0,0)\}$.

- S is Schaefer (closed under conjunction).
- We can enumerate solutions for S-formulas.

- Enumerating S-formulas is the same as enumerating "Boolean solutions" for R-formulas.

$$\varphi = \bigwedge_{i=1}^{k} R(x_1^i, \ldots, x_n^i)$$

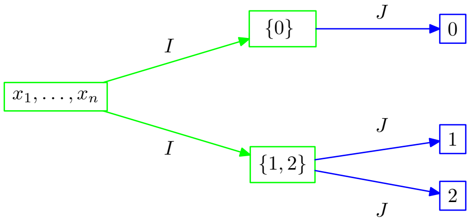
$$\downarrow$$

$$\psi = \bigwedge_{i=1}^{k} S(x_1^i, \ldots, x_n^i)$$

The formulas have the same "Boolean solutions."

An Example

- $f_{2\rightarrow 1} \in \mathrm{Pol}(R)$, hence there is a "Boolean" for each "real" solution.
- Boolean solution is real one with all 2s made to 1s.
- Given "Boolean Solution," generate all "fitting" solutions.
- Get "real" from "Boolean solutions:" change some 1s into 2s.
- The 1 is a "placeholder" for either 1 or 2.



Partial Solutions

- We get "Boolean solutions."
- How do we get the "refinements?"
- Set some variables from 1 to 2.
- This is a Boolean Constraint Satisfaction Problem.
- Consider a "Boolean solution" for a single clause:
- $R(w = \{1, 2\}, x = 0, y = 0, z = \{1, 2\})$
- Set both w and z to 2, or both to 1.
- Relation describing combinations:

$$R^I = \{(1, 0, 0, 1), (2, 0, 0, 2)\} \rightarrow \{(1, 1), (2, 2)\}$$

Partial Solutions

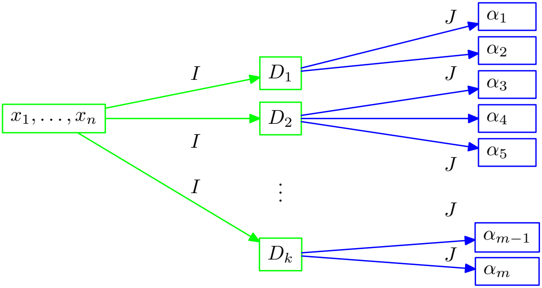
- Original: Enumeration problem over domain $\{0, 1, 2\}$
- Split up into two Boolean problems:

1. Enumerate the “Boolean solutions” of an R-formula
2. Find all “fitting” solutions to a “Boolean solution”

- $E := \{D_1, \ldots, D_k\}$ a partition of the domain D.
- $I \colon \mathrm{VAR}(\varphi) \to E$
- $J \colon \mathrm{VAR}(\varphi) \to D$ is compatible with I if $J(x) \in I(x)$ for all variables x of φ
- $$\begin{array}{lll} I(x) = \{0\} & I(y) = \{1,2\} & I(z) = \{1,2\} \\ J(x) = 0 & J(y) = 1 & J(z) = 2 \end{array}$$
- I is a partial solution for φ, if there is a solution $J \models \varphi$ and J is compatible with I.

Partial Solutions

- $E = \{D_1, \ldots, D_k\}$, I a partial E-solution, J compatible with I



- PROBLEM: How can we enumerate partial E-solutions?
- $E := \{D_1, \ldots, D_k\}$ a partition of the domain D.

Theorem

If CSP $(\Gamma \cup \{D_1, \ldots, D_k\}) \in \mathrm{P}$, *then* Γ *we can efficiently enumerate partial* E*-solutions for* Γ*-formulas.*

From Partial to "Real" Solutions

- PROBLEM: Given a partial solution I, how can we generate all compatible solutions J?

Definition

- $R \subseteq D^n$, E a partition on D.
- $\mathbf{v} \in E^n$.
- $\mathbf{v}^{E \to D} := \{\mathbf{t} \in R \mid \mathbf{t} \text{ compatible with } \mathbf{v}\}$
- $R_{\mathbf{v}}^{E \to D} := \mathbf{v}^{E \to D}$, without redundance
- $\Gamma_R^{E \to D} := \left\{ R_{\mathbf{v}}^{E \to D} \mid \mathbf{v} \in E^n \right\}$.

From Partial to "Real" Solutions

Theorem

$R \subseteq D^n$, E partition on D. Equivalent:

1. *We can get all compatible solutions to partial solutions for R-formulas efficiently*
2. *$\Gamma_R^{E \to D}$ is enumerable.*

Nested Algorithm

- Original: Enumeration problem over the domain D
- Consider partition E on D
- Split up into two problems:
  1. Enumerate the "partial E-solutions"
  2. Find all "compatible" solutions to a "partial E-solution"
- Two problems over a smaller domain.

Lemma

Let $D = \{a, b, c\}$, $R \subseteq D^n$, *such that:*

1. $f_{a \to b} \in \mathrm{Pol}(R)$
2. $f_{a \to b}(R)$ *does not have an efficient enumeration algorithm.*
3. *One of the following cases occurs::*
   - 3.1 *There is some* $\mathbf{v} \in \{a, b\}^n$ *such that* $\mathbf{v} \notin R$, $\mathbf{v}[a/c, b/b] \in R$, $\mathbf{v}[a/b, b/c] \in R$.
   - 3.2 $(b, \ldots, b) \notin R$
   - 3.3 $(a, \ldots, a), (c, \ldots, c) \notin R$, *and there is some* $\mathbf{v} \in \{a, b\}^n \setminus R$ *such that* $f_{a \to c}(\mathbf{v}) \in R$

R *not enumerable, if* $\mathrm{P} \neq \mathrm{NP}$.

The Algebraic Approach

- Let $\Gamma_1 \subseteq \langle \Gamma_2 \rangle$.
- Transform Γ_1-formulas into Γ_2-formulas, with new $\exists$-variables.
- New variables are a problem for enumeration.
- For Boolean case, enumeration follows the Galois connection.
- What about non-Boolean domains?

Theorem

There are Γ_1 *and* Γ_2 *with* $\Gamma_2 \subseteq \langle \Gamma_1 \rangle$ Γ_1 *has an efficient enumeration algorithm, and* Γ_2 *does not (unless* $\mathrm{P} = \mathrm{NP}$*).*

Example

$$\left(\begin{array}{l} \quad\Big(\{a\}\times\{a\}\times\{b\}\times\Big(\{a,b\}^3\setminus\{(b,b,b)\}\Big)\cup\{(b,b,b,b,b,b)\}\Big)\times\{b\} \\ \cup\Big(\{a\}\times\{b\}\times\{a\}\times\Big(\{a,b\}^3\setminus\{(b,b,b)\}\Big)\cup\{(b,b,b,b,b,b)\}\Big)\times\{c\} \\ \cup\Big(\{b,c\}^7\setminus\{(c,c,c,c,c,c,b),(c,c,c,c,c,c,c)\}\Big) \end{array}\right)\times\{c\}$$
$$\cup\{(a,a,a,a,a,a,a,c)\}.$$

$$\left(\begin{array}{l} \quad\Big(\{a\}\times\{a\}\times\{b\}\times\Big(\{a,b\}^3\setminus\{(b,b,b)\}\Big)\cup\{(b,b,b,b,b,b)\}\Big) \\ \cup\Big(\{a\}\times\{b\}\times\{a\}\times\Big(\{a,b\}^3\setminus\{(b,b,b)\}\Big)\cup\{(b,b,b,b,b,b)\}\Big) \\ \cup\Big(\{b,c\}^6\setminus\{(c,c,c,c,c,c)\}\Big) \end{array}\right)\times\{c\}$$
$$\cup\{(a,a,a,a,a,a,c)\}.$$

Theorem

$R \subseteq D^n$, $|D| = 3$, $E = \{\{a, b\}, \{c\}\}$ partition on D, such that all polymorphisms of R are "conservative" on E. Then our algorithms cover all cases.

Future Research

- We have shown new enumeration algorithms.
- Previously known algebraic methods are only of limited use.
- For more hardness results, we will need to study partial polymorphisms.

Thank You!

- Questions?