Enumeration algorithms

*M2 ORCO – Grenoble*

Aurélie Lagoutte

November 9, 2022
Enumeration : principle

Some problems need as an answer a **list** of solutions, instead of a **single** solution. For example:
Enumeration: principle

Some problems need as an answer a **list** of solutions, instead of a **single** solution. For example:

- **Answer to a database query**

  ```
  $ select appellation, vignoble, type from AOC
  Côtes-Rôties     | Vallée du Rhône     | Rouge
  Saint-Emilion    | Bordeaux            | Rouge
  Saint-Nicolas-de-Bourgueil | Val de Loire | Rouge
  ```
Enumeration: principle

Some problems need as an answer a **list** of solutions, instead of a **single** solution. For example:

- **Answer to a database query**
  ```
  $ select appellation, vignoble, type from AOC 
  Côtes-Rôties    | Vallée du Rhône   | Rouge
  Saint-Emilion   | Bordeaux          | Rouge
  Saint-Nicolas-de-Bourgueil | Val de Loire    | Rouge
  ```

- **Truth table: list all input combinations**

<table>
<thead>
<tr>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$S$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Enumeration: a typical example

**Input:** Graph $G$

**Output:** The list of all *inclusion-wise maximal* stable sets of $G$

$\{1, 3, 5\}, \{1, 4\}, \{2, 5\}, \{3, 6\}$
Focus on easy problems

**Input:** Graph $G$

**Output:** one *inclusion-wise maximal* stable set. $\in \mathbb{P}$ (greedy)

Not to be confused with:

**Input:** Graph $G$

**Output:** a stable set of *maximum* size

NP-complete
Enumerating in graphs: useful cases

- Graph databases: answer to a query
- Graph model is not exact: some solutions are best based on qualitative criteria, we have to examine them one by one
- Identify all problematic (or interesting!) patterns in a network

Application fields: bioinformatics (phylogenetic trees), chemistry (molecule structure), complex system modeling, databases...

Diagram of a stereoisomer

---

1Comparison and Enumeration of Chemical Graphs, T. Akutsu, H. Nagamochi, Comp. and Struct. Biotechnology Journal
Complexity for enumeration problems

In most cases: exponential number of solutions to output
(ex: $3^{n/3}$ max. stable sets)

⇒ Good complexity measure?
Complexity for enumeration problems

In most cases: exponential number of solutions to output
(ex: $3^{n/3}$ max. stable sets)

⇒ Good complexity measure?

1. Output-polynomial

Input of size $n$, $N$ solutions to output.

\[ poly(n + N) \]
Complexity for enumeration problems

In most cases: exponential number of solutions to output
(ex: $3^{n/3}$ max. stable sets)
⇒ *Good complexity measure?*

1. Output-polynomial
2. Incremental polynomial

Input of size $n$, $N$ solutions to output.
Complexity for enumeration problems

In most cases: exponential number of solutions to output
(ex: $3^{n/3}$ max. stable sets)

⇒ Good complexity measure?

1. Output-polynomial
2. Incremental polynomial
3. Polynomial delay

Input of size $n$, $N$ solutions to output.
Enumeration?

Enumerate... but what?

Methods

Complexity for enumeration problems

In most cases: exponential number of solutions to output
(ex: $3^{n/3}$ max. stable sets)
⇒ Good complexity measure?

1. Output-polynomial
2. Incremental polynomial
3. Polynomial delay

poly space vs. exponential space

Input of size $n$, $N$ solutions to output.
Interesting objects to enumerate

1. Inclusion-wise minimal transversal of a hypergraph
Interesting objects to enumerate

1. Inclusion-wise minimal transversal of a hypergraph
2. Inclusion-wise minimal dominating sets
Interesting objects to enumerate

1. Inclusion-wise minimal transversal of a hypergraph
2. Inclusion-wise minimal dominating sets
3. Spanning trees
Interesting objects to enumerate

1. Inclusion-wise minimal transversal of a hypergraph
2. Inclusion-wise minimal dominating sets
3. Spanning trees
4. "Structured patterns" : inclusion-wise max. stable sets or cliques ...
Interesting objects to enumerate

1. Inclusion-wise minimal transversal of a hypergraph
2. Inclusion-wise minimal dominating sets
3. Spanning trees
4. "Structured patterns" : inclusion-wise max. stable sets or cliques ...
5. Inclusion-wise minimal "Π-fixings" of a graph
   → Completions, deletion, induced subgraphs of a graph ...
   ... satisfying a given property Π
Minimal fixings

3 variants

We want to satisfy a given property $\Pi$

Example: $\Pi = C_4$-free

(contains no induced $C_4$)
Minimal fixings

3 variants

We want to satisfy a given property $\Pi$

Example: $\Pi = C_4$-free

(contains no induced $C_4$)
Minimal fixings

3 variants

We want to satisfy a given property $\Pi$
Example: $\Pi = C_4$-free
(contains no induced $C_4$)

Fixing by adding edges

**Min. $\Pi$-completion**

- Adding edges to the graph to satisfy the property $\Pi$.
Minimal fixings

3 variants

We want to satisfy a given property \( \Pi \)

Example : \( \Pi = C_4 \)-free

(contains no induced \( C_4 \))

Fixing by adding edges

\textbf{Min. \( \Pi \)-completion}

Fixing by deleting edges

\textbf{Min. \( \Pi \)-deletion}

\begin{itemize}
  \item Fixing by adding edges
  \item Fixing by deleting edges
  \item + 3 others
\end{itemize}
Minimal fixings

3 variants

We want to satisfy a given property $\Pi$

Example: $\Pi = C_4$-free
(contains no induced $C_4$)

Fixing by adding edges

Min. $\Pi$-completion

Fixing by deleting edges

Min. $\Pi$-deletion

+ 3 others

Fixing by deleting vertices

Max. $\Pi$-induced subgraph

+ 3 others
Chordal completion

Given a graph $G = (V, E)$, a chordal completion of $G$ is a completion of $G$ that is chordal.

Sometimes we refer to the chordal completion as a fill-in $F$ where $F$ denotes the set of non-edges of $G$ that must be turned into edges so that $(V, E \cup F)$ is chordal.

A chordal completion of $G$ is also called a triangulation of $G$. 
Chordal completion

Given a graph $G = (V, E)$, a chordal completion of $G$ is a completion of $G$ that is chordal.

Sometimes we refer to the chordal completion as a fill-in $F$ where $F$ denotes the set of non-edges of $G$ that must be turned into edges so that $(V, E \cup F)$ is chordal.

A chordal completion of $G$ is also called a triangulation of $G$. 
Algorithmic methods to enumerate

(Un petit quiz avant d’enchaîner?
http://wooclap.com/M2ORCRO)
General principle of an enumeration algorithm

- Metagraph of solutions: traversal of this metagraph

Example with maximal stable sets
Solution metagraph
General principle of an enumeration algorithm

- Metagraph of solutions: traversal of this metagraph
- Outputting each solution once

Example with maximal stable sets
Solution metagraph
General principle of an enumeration algorithm

- Metagraph of solutions: traversal of this metagraph
- outputting each solution once
- and only once

Example with maximal stable sets
Solution metagraph
General principle of an enumeration algorithm

- Metagraph of solutions: traversal of this metagraph
- Outputting each solution once
- And only once

Example with maximal stable sets
Solution metagraph
Three classical methods:

Goal: get polynomial delay
+ poly space for 1 et 2 (and sometimes 3)

1. Flashlight search or Binary partition
   [Read, Tarjan ’75]

2. Reverse search
   [Avis, Fukuda ’96]

3. Proximity Search
   [Conte, Uno ’19]
   [Conte, Grossi, Marino, Uno, Versari, ’21]
Three classical methods:

Goal: get polynomial delay
+ poly space for 1 et 2 (and sometimes 3)

1. **Flashlight search** or **Binary partition**
   [Read, Tarjan ’75]

2. **Reverse search**
   [Avis, Fukuda ’96]

3. **Proximity Search**
   [Conte, Uno ’19]
   [Conte, Grossi, Marino, Uno, Versari, ’21]
Flashlight search or Binary partition

1 ∈ S?
   Yes
   No

2 ∈ S?
   Yes
   No

3 ∈ S?
   Yes
   No

6 ∈ S?
   Yes
   No

V
{1,3,5}
{1,3,6}
{1,4}
{2,5}
{3,6}
{2,5,6}
{3,4 }
{4,5,6}
∅
**Flashlight search or Binary partition**

1 ∈ $S$?
- Yes
- No

2 ∈ $S$?
- Yes
- No

3 ∈ $S$?
- Yes
- No

6 ∈ $S$?
- Yes
- No

$V$
- $\{1,3,6\}$
- $\{1,3,5\}$
- $\{2,5,6\}$
- $\{2,5\}$
- $\{3,6\}$
- $\{3,4\}$
- $\{4,5,6\}$
- $\emptyset$
Flashlight search or Binary partition

1 ∈ S?

Yes

No

1 ∈ S?
Yes
No

13/18
**Flashlight search or Binary partition**

1 ∈ S?

- **Yes**
- **No**
Flashlight search or Binary partition

1 ∈ S?
Yes
No

2 ∈ S?
Yes
No
Flashlight search or Binary partition

1 ∈ S?
- Yes
- No

2 ∈ S?
- Yes
- No

No sol. here
Flashlight search or Binary partition

1 ∈ S?

Yes

No

2 ∈ S?

Yes

No

X
Flashlight search or Binary partition

1 ∈ S?
Yes
No
2 ∈ S?
Yes
No
3 ∈ S?
Yes
No

1 ∈ S?
Yes
No
2 ∈ S?
Yes
No
3 ∈ S?
Yes
No
Flashlight search or Binary partition

1 ∈ S?
- Yes
- No

2 ∈ S?
- Yes
- No

3 ∈ S?
- Yes
- No

{1,3,5}
**Flashlight search or Binary partition**

1 ∈ S?  
   Yes  
   No

2 ∈ S?  
   Yes  
   No

3 ∈ S?  
   No

{1, 3, 5}  
No sol.  
Yes  
No

Yes  
No
Flashlight search or Binary partition

1 ∈ S?

Yes

No

2 ∈ S?

Yes

No

3 ∈ S?

Yes

No

{1, 3, 5}

Yes

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No

No

Yes

No

No
Flashlight search or Binary partition

1 ∈ S?
1 ∈ S?
Yes
No

2 ∈ S?
Yes
No

3 ∈ S?
{1,3,5}
Yes
No
Yes

1, 3, 5 ∈ {1,3,5}
Flashlight search or Binary partition

For Flashlight search to run in poly delay (and space):

Answer in poly time to the extension problem of $\mathcal{P}$:

Let $A$ and $B$ be two disjoint sets
is there a solution $S$ of $\mathcal{P}$ such that $A \subseteq S$ and $S \cap B = \emptyset$?

All of $A$ must be in solution
and all of $B$ is forbidden in the solution.
Three classical methods:

Goal: get polynomial delay
+ poly space for 1 et 2 (and sometimes 3)

1. **Flashlight search** or **Binary partition**
   [Read, Tarjan ’75]

2. **Reverse search**
   [Avis, Fukuda ’96]

3. **Proximity Search**
   [Conte, Uno ’19]
   [Conte, Grossi, Marino, Uno, Versari, ’21]
Reverse search

Solution space

S1
S2
S3
S4
S5
S6
S7
S8
S9
S10
S12

Solution space
Reverse search

Solution metagraph
Reverse search

Solution tree

S1
S2
S3
S4
S5
S6
S7
S8
S9
S10
S12
Reverse search

Solution tree

\[ S_1 \]
Reverse search
Reverse search
Reverse search
Reverse search
Reverse search
Reverse search

- $S_1$
- $S_2$
- $S_3$
- $S_4$
- $S_5$
- $S_6$
- $S_7$
Reverse search

To have Reverse search run in poly delay and space:

Generate in poly time and space the children of a solution (each solution must have a single father)
Reverse Search

ALL stable sets