The world as we know it …

All computational problems

Solvable

Unsolvable

Why polynomial?

• Someone said so…
• It makes a lot of sense…
What’s in P?

Every decision problem that has a polynomial-time algorithm: e.g.
– Is the list S of integers sorted in ascending order?
– Is the graph G connected?
– Does graph G have a MST with cost K or less?
– Does tree T have a vertex cover of size K or less?

What about Search problems?

• We’ll come back to that.

What is not in P?

• Recognizing true statements in Presburger arithmetic
• The circularity problem for attribute grammars
• Inequivalence for regular expressions with squaring
• And others

Huh?

The world of solvable problems…

Don’t know

The world of solvable problems…as it seems

Don’t know
NP: all tractable + some don’t know

- Known to be tractable
- Don’t know
- Known to be intractable

Is P=NP?

- P
- NP

What is NP?

- NP is the class of decision problems that have polynomial-time verifiable proofs.
- HUH?

About NP

- P is in NP
- Some of NP is in don’t know
- NP as a class has some nice properties
- NP is the smallest class containing some don’t knows that has these properties

What is NP?

- A language over an alphabet $\Sigma$ is a subset of $\Sigma^*$
- Examples for $\Sigma=\{0,1\}$
  - $\{01,0101,010101,\ldots\}$
  - $\{0, 11, 110, 1001, \ldots\}$
  - $\emptyset$
  - $\Sigma^*$

Some classes of languages

- Regular
- Context-free
- Recursive
- Recursive-enumerable
Language classes

• Language classes are typically defined by the computational power needed to answer membership queries:

  Is x in L?

Some classes of languages

• Regular – Finite Automata
• Context-free – Pushdown Automata
• Recursive – Turing machine
• Recursive-enumerable – Turing machines can answer yes but not necessarily no.

Turing machine

• A simple model of a computer:
  – Finite state machine
  – R/W tape
  – Can be programmed to follow simple rules

Church-Turing thesis

• Any physically-realizable computing device can be modeled with at most polynomial-time blowup by a randomized Turing machine.

Note

• Church-Turing thesis may be disproved by quantum computers if they are found to be
  – Physically realizable
  – Provably more powerful than traditional Turing machines

More classes

• Membership questions can be answered by resource-bounded Turing machines
  – Limit time
  – Limit space
  – Limit randomness
P

- Membership questions can be answered by resource-bounded Turing machines
  - Limit time – polynomial
  - Limit space
  - Limit randomness – none

More classes

- Membership questions can be answered by resource-bounded non-deterministic Turing machines
  - Limit time
  - Limit space
  - Limit randomness

NP

- Membership questions can be answered by resource-bounded non-deterministic Turing machines
  - Limit time - polynomial
  - Limit space
  - Limit randomness - none

Decision Problems

- Computational problems in which the output is Yes or No.
- Decision problems can be posed as membership queries.

Vertex Cover

Input space: G,k

Yes instances
(G,k) such that G has a vertex cover of size k

No instances
(G,k) such that G does not have a vertex cover of size k

Input space: valid encodings

Yes instances
x: such that x encodes a yes-instance of V.C.

No instances
x: such that x encodes a no-instance of V.C.
Language over \{0,1\}^*

<table>
<thead>
<tr>
<th>Yes instances</th>
<th>Invalid encodings</th>
</tr>
</thead>
<tbody>
<tr>
<td>x: such that x encodes a yes-instance of V.C.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No instances</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>x: such that x encodes a no-instance of V.C.</td>
<td></td>
</tr>
</tbody>
</table>

Encoding Rule

- It is easy to determine whether or not a binary string is a valid encoding.
- A problem of size \(n\) can be encoded in \(\text{poly}(n)\) bits.
- Notion of tractability is preserved.

Our outlook

- Natural problems
  - Ignore coding issue unless it matters
- Decision version
  - What about search?
- Objective to distinguish tractable from intractable

Shopping Bag Problem

\(n\) items of weight at most \(B\)

- Natural problems
  - Ignore coding issue unless it matters
- Coding clarifies what we mean by “input size”

Shopping Bag Problem

\(n\) items of weight at most \(B\)

- Natural problems
  - Input size is \(n \log(B)\) bits
Shopping Bag Problem
n items of weight at most B
• Natural problems
  • Input size is n \lg(B) bits
• Objective: An O(nB) algorithm does not prove tractability

Our outlook
• Decision version
  – What about search?
    • If the decision problem is intractable then the search problem is intractable. Why?

Our outlook
• Decision version
  – What about search?
    • If the decision problem is intractable then the search problem is intractable. Why?

Typically, if the decision problem is tractable then so is the search problem. Why?

The world as it seems…

We don’t know

The world as we believe it to be…

P: Tractable

NP

Intractable
NP

• Languages that can be posed as
  \{ x \mid \exists y \text{ such that } P(x,y) \}

where \( P(x,y) \) is checkable in time poly(|x|)

Example

• VC:
  – \( x = (G,k) \)
  – \( y \) is a vertex cover of \( G \) containing \( k \) or fewer vertices

• 3-coloring:
  – \( x = G \)
  – \( y \) is a function mapping \( V \) to \{red,blue,green\}

NP: Other characterizations

• Languages decidable in polynomial-time by a non-deterministic Turing machine
• Languages that have probabilistically checkable proofs using a constant number of queries and logarithmic randomness

The world as we believe it to be…

But what could be…
The world as we believe it to be…

NP-complete

- If A is NP and B is NP-complete then $A \preceq_p B$
- If any NP-complete problem is tractable then every NP problem is tractable.

NP

- Is it in NP?
  - Is it also in P?
  - Is it NP-Complete?
  - Else?

NP-Completeness Map

Legend

- Vertex Cover
- Dominating Set

VC $\preceq_p$ DS:
1. If VC is NP-hard then so is DS.
2. If DS can be solved efficiently then so can VC.

NP-Completeness Map
Clique ↔ Independent Set

For $G = (V,E)$ the complement of $G$ is $G' = (V, V \times V - E)$.

\[ T_{\text{clique}} (n,m) = n^2 + T_{\text{ind-set}}(n,n^2-m) \]

Reduction is poly(n,m)

If $T_{\text{ind-set}}$ is polynomially-bounded then so is $T_{\text{clique}}$. 

\[ T_{\text{clique}} (n,m) = n^2 + T_{\text{ind-set}}(n,n^2-m) \]
NP-Completeness Map

- Clique
  - Independent Set, trivial
  - Vertex Cover, simple
  - Dominating Set, easy

NP-Completeness Map

- Hamiltonian Cycle, easy
  - Hamiltonian Path, trivial
  - Longest Path

NP-Completeness Map

- 3-Coloring, simple
  - 4-Coloring
  - K-Coloring, trivial