CSE332: Data Abstractions
Lecture 24.5: Interlude on Intractability

Dan Grossman
Spring 2012
No, the material in lecture 24.5 (this one) won’t be on the final
– But it’s still an important high-level idea
Intractable Graph Problems

• Graph problems we studied all had efficient solutions (roughly $O(|V|^2)$ or better)
  – Topological sort
  – Traversals/connectedness
  – Shortest paths
  – Minimum Spanning Tree

• But there are plenty of intractable graph problems
  – Worst-case exponential in some aspect of the problem as far as we know
  – Topic studied in CSE312 and CSE421, but do not want to give false impression that there is always an efficient solution to everything
  – Common instances or approximate solutions can be better
**Vertex Cover: Optimal**

Input: A graph \((V, E)\)

Output: A minimum size subset \(S\) of \(V\) such that

for every edge \((u, v)\) in \(E\), at least one of \(u\) or \(v\) is in \(S\)

\(O(2^{|V|})\) algorithm: Try every subset of vertices; pick smallest one

\(O(|V|^k)\) algorithm: Unknown, probably does not exist
**Vertex Cover: Decision Problem**

Input: A graph \((V, E)\) and a number \(m\)
Output: A subset \(S\) of \(V\) such that for every edge \((u, v)\) in \(E\), at least one of \(u\) or \(v\) is in \(S\) and \(|S| = m\) (if such an \(S\) exists)

\(O(2^m)\) algorithm: Try every subset of vertices of size \(m\)
\(O(m^k)\) algorithm: Unknown, probably does not exist

Good enough: Binary search on \(m\) can solve the original problem

Easy to **verify** a solution: See if \(S\) has size \(m\) and covers edges
Traveling Salesman

[Like vertex cover, usually interested in the optimal solution, but we can ask a yes/no question and rely on binary search for optimal]

Input: A complete directed graph \((V,E)\) and a number \(m\)
Output: A path that visits each vertex exactly once and has total cost \(< m\) if one exists

\(O(|V|!)\) algorithm: Try every path, stop if find cheap enough one

Verifying a solution: Easy
Clique

Input: An undirected graph \((V, E)\) and a number \(m\)
Output: Is there a subgraph of \(m\) nodes such that every edge in the subgraph is present?

Naïve algorithm: Try all subsets of \(m\) nodes

Verifying a solution: Easy
Not Just Graph Problems

• Every problem studied in CSE332 has an efficient solution
  – Correct cause and effect: Chose to study problems for which we know efficient solutions!

• There are plenty of intractable problems…
**Subset Sum**

Input: An *array* of *n* numbers and a target-sum *sum*
Output: A subset of the numbers that add up to *sum* if one exists

\[ O(2^n) \] algorithm: Try every subset of array
\[ O(n^k) \] algorithm: Unknown, probably does not exist
Satisfiability

\[(\neg x_1 \lor x_2 \lor x_4) \land (x_1 \lor \neg x_3 \lor x_4) \land (x_2 \lor \neg x_4 \lor \neg x_5)\]

Input: a logic formula of size $m$ containing $n$ variables
Output: An assignment of Boolean values to the variables in the formula such that the formula is true

$O(m \cdot 2^n)$ algorithm: Try every variable assignment
$O(m^k n^k)$ algorithm: Unknown, probably does not exist
So… what to do?

• Given a problem, how can you:
  – Find an efficient solution…
  – … or prove that one (probably) does not exist?

• See CSE312, CSE421, CSE431