Lecture 31: NP-Completeness

Agenda:

- NP-completeness: the main ideas
- Graph representations & size of an instance
- Decision problems & instances
- Polynomial time
- Classes P, NP

Reading:

- Textbook pages 966 – 983

The problems we have studied:

1. Sorting — can be done in $\Theta(n \log n)$
2. Longest common subsequence — can be done in $\Theta(n \times m)$
3. Minimum spanning tree — can be done in $\Theta(m \log n)$
4. Single-source shortest paths — can be done in $\Theta(m \log n)$

They can be solved in a <u>short</u> amount of time.

What does "short" mean?

1. Sorting

   n is the number of keys, measuring how big the sorting instance is

2. Longest common subsequence

   n, m are the lengths of the sequences, measuring how big the sorting instance is

3. Minimum spanning tree

   n, m are the numbers of vertices and the number of edges, measuring how big the sorting instance is

4. Single-source shortest paths

   n, m are the numbers of vertices and the number of edges, measuring how big the sorting instance is

"Short" is <u>polynomial</u> in the "size" of the instance

Some other computational problems:

- Eulerian tour — a cycle including every edge exactly once

  Determine if a graph is Eulerian

  can be done in $\Theta(n^2)$

- Hamiltionian cycle — a cycle including every vertex exactly once

  Determine if a graph is Hamiltonian

  so far no known algorithm in $O(n^k)$ for any k

- Shortest x-to-y path

  can be done in $\Theta(n^2)$

- Longest x-to-y path

  so far no known algorithm in $O(n^k)$ for any k

We need/want to classify the problems into "**easy**" and "**hard**" categories ...

Basic concepts:

- Size of an instance: in order to store the instance into the computer, how many memory units are necessary?

  Adjacency list representation of a graph containing 6 vertices and 7 edges:

1:	2, 3, 5
2:	1, 4, 6, 3
3:	2, 1
4:	5, 2
5:	4, 1
6:	2

  `6//2,3,5/1,4,6,3/2,1/5,2/4,1/2//` — 32 memory units

  In this problem: $\Theta(n+m)$

- Polynomial time — polynomial in the size(s) of the instance(s)

- Abstract problem
  - two parts:
    1. a set of <u>instances</u> — which are inputs
    2. a query — the question asked
  - instance solution: answer to the query — the output

  Example: minimum spanning tree problem
  1. instances: all edge-weighted (simple, undirected) graphs
  2. query: for input G, what is the length of an MST of G?

Basic concepts (2):

- Decision problem: the answer to the query is **yes** or **no**

  Example: minimum spanning tree problem

  1. instances: all (G, ℓ): G an edge-weighted (simple, undirected) graph, ℓ an integer
  2. query: for input (G, ℓ), is there a spanning tree of G of length at most ℓ?

- Optimization problem: abstract problem with an optimization goal

- Relations between optimization problem and its decision version
  - suppose you can solve the optimization problem, then you can solve the decision problem (how?)
  - suppose you can solve the decision problem, then generally you can solve the optimization problem as well (how?)

- Notes:
  - for some abstract problems, their decision version is the same. Example: `Hamiltonian graph problem`
    - instances: all (simple, undirected) graphs G
    - query: for input G, does it have a Hamiltonian cycle?
  - correspondence: optimization $\leftrightarrow$ decision
    1. minimization $\leftrightarrow$ *at most*
    2. maximization $\leftrightarrow$ *at least*

The classes of P and NP:

- Class P
  - decision problem
  - there exists some algorithm solving the problem in polynomial time
  - which of the above problems are in P:
    1. sorting
    2. longest common subsequence
    3. minimum spanning tree
    4. single-source shortest paths
    5. shortest x-to-y path
    6. determining Eulerian graphs

- Class NP
  - decision problem
  - for every **yes**-instance, there exists a proof that the answer is yes and the proof can be verified in polynomial time
  - which of the above problems are in NP:
    1. sorting
    2. longest common subsequence
    3. minimum spanning tree
    4. single-source shortest paths
    5. shortest x-to-y path
    6. determining Eulerian graphs
    7. longest x-to-y path
    8. determining Hamiltonian graphs

Have you understood the lecture contents?

well	ok	not-at-all	topic
☐	☐	☐	rough idea on ‘easy’ and ‘hard’
☐	☐	☐	size of an instance
☐	☐	☐	polynomial time
☐	☐	☐	abstract, optimization, decision problems
☐	☐	☐	optimization $\leftrightarrow$ decision
☐	☐	☐	P and NP