Optimization (2MMD10/2DME20), lecture 4

Gerhard Woeginger

Technische Universiteit Eindhoven

Fall 2015, Q1

Program for this week

- Basic definitions: discrete problems, algorithms, time complexity
- P versus NP
- Reductions
- NP-hardness
- A catalogue of NP-hard problems

Basic concepts (1)

Discrete problem:

- Optimization problem (min/max)
- Decision problem (with answer YES/NO)

Example: Optimization problem

Instance: a graph $G = (V, E)$
Goal: find a clique of maximum size in G

Example: Decision problem

Instance: a graph $G = (V, E)$; a bound k
Question: does G contain a clique of size (at least) k?

Basic concepts (2)

Instance:

- specification of problem data

Example: Instance of decision version of clique

$V = \{1, 2, 3, 4, 5\}$;
$E = \{[1, 2],\ [1, 3],\ [4, 5],\ [2, 3],\ [3, 5]\}$;
$k = 3$

Basic concepts (3)

Problem size:

- length (number of symbols) of reasonable encoding of instance

Example

- Graph: adjacency list; adjacency matrix
- Set: list of elements; bit vector
- Number: decimal; binary; hex; unary

We do not really care whether
an n-vertex graph is encoded with $4n^2 + 3n$ or with $7n^2 + 2$ symbols.

Recall: big-Oh notation; $4n^2 + 3n \in O(n^2)$ and $7n^2 + 2 \in O(n^2)$

Basic concepts (4)

Algorithm:

- an unambiguous recipe for solving a discrete problem

(If you want: just think of 'algorithm' as C++ program)

Time complexity of an algorithm:

- number of elementary steps an algorithm makes

The time complexity is measured as a function of the instance size:

- $T_A(I)$ = number of steps that algorithm A makes on instance I
- $T(n)$ = maximum number of steps that algorithm A makes on any instance I of size $O(n)$

Basic concepts (5): Polynomial versus exponential

Polynomial growth rate:

- $O(\text{poly}(n))$ for some polynomial poly

Example: $O(n)$; $O(n \log n)$; $O(n^3)$; $O(n^{100})$

Exponential growth rate:

- everything that grows faster than polynomial

Example: 2^n; 3^n; $n!$; 2^{2^n}; n^n

Intuition:

Polynomial = desirable, good, harmless, fast, short, small

Exponential = undesirable, bad, evil, slow, wasteful, horrible

Basic concepts (6)

Observation

Every discrete optimization problem can be rewritten into a short sequence of decision problems:
use bisection search on the interval of objective values

Example

Let G be a graph on n vertices.

Does G contain a clique of size at least $n/2$? – YES
Does G contain a clique of size at least $3n/4$? – YES
Does G contain a clique of size at least $7n/8$? – NO
Does G contain a clique of size at least $13n/16$? – YES
Etc.

Search takes logarithmic number of steps –> fast and simple

P versus NP

Definition

A decision problem X lies in the complexity class P,
if X is solved by an algorithm with polynomial time complexity

Definition

A decision problem X lies in the complexity class NP,
if for every YES-instance of X
there exists a certificate of polynomial length
that can be verified in polynomial time

Example

A certificate for the decision version of clique:
subset $C \subseteq V$ of size k that induces a clique

Exercise: Satisfiability

Satisfiability (SAT)

Instance:

a logical formula Φ in CNF over logical variable set $X = \{x_1, \ldots, x_n\}$

Question: does there exist a truth setting for X that satisfies Φ?

Examples

$$\Phi = (x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$$
$$\Phi = (x \vee y) \wedge (\neg x \vee y) \wedge (x \vee \neg y) \wedge (\neg x \vee \neg y)$$

Question

What's a good NP-certificate for SAT?

Exercise: Integer programming

Integer linear programming (ILP)

Instance: an integer matrix A; an integer vector b

Question: does there exist an integer vector x with $Ax \leq b$?

Question

What's a good NP-certificate for ILP?

Exercise: Hamiltonian cycle / TSP

Hamiltonian cycle (HC)

Instance: an undirected graph $G = (V, E)$
Question: does G contain a Hamiltonian cycle?
(a simple cycle that visits every vertex exactly once)

Travelling Salesman Problem (TSP)

Instance: cities $1, \ldots, n$; distances $d(i, j)$; a bound B
Question: does there exist a roundtrip of length at most B?

Question

What's a good NP-certificate for HC?
What's a good NP-certificate for TSP?

Exercise: Exact cover

Exact cover (Ex-Cov)

Instance: a ground set X; subsets $S_1, \ldots, S_m$ of X

Question: do there exist some subsets S_i that form a partition of X?

Question

What's a good NP-certificate for Ex-Cov?

Exercise: Subset Sum

Subset Sum (SS)

Instance: positive integers $a_1, \ldots, a_n$; a bound b

Question: does there exist an index set $I \subseteq \{1, \ldots, n\}$ with $\sum_{i \in I} a_i = b$?

Question

What's a good NP-certificate for SS?

Back to P versus NP

- P = class of all problems that are easy to solve
  P stands for Polynomial Time
- NP = huge class of problems that fulfill some soft condition
  NP contains lots of interesting and important decision problems
  NP stands for Non-deterministic Polynomial Time

Big open question

P=NP ????

Answer YES:

- would trigger a revolution in computing
- if a short solution exists, it can be found quickly

Answer NO:

- that's what most people expect
- even very short solutions may be very hard to find

NP-hardness (1)

Definition

For two decision problems X and Y,
we say that X reduces to Y (and we write $X \leq_p Y$)
if there exists a polynomial time transformation f
that translates instance of X into instances of Y
with $I \in$YES(X) $\iff$ $f(I) \in$YES(Y).

Intuition:

- X can be modelled as a special case of Y
- If Y is easy, then also X is easy
- If X is difficult, then also Y is difficult

NP-hardness (2)

Problem: EvenPath

Instance: an undirected graph $G = (V, E)$; two vertices $s, t \in V$
Question: does there exist a simple path from s to t
that uses an even number of edges?

Problem: OddPath

Instance: an undirected graph $G' = (V', E')$; two vertices $s', t' \in V'$
Question: does there exist a simple path from s' to t'
that uses an odd number of edges?

Lemma

(a) EvenPath $\leq_p$ OddPath.
(b) OddPath $\leq_p$ EvenPath.

NP-hardness (3)

Lemma

Reducibility is a transitive relation:
$X \leq_p Y$ and $Y \leq_p Z$ implies $X \leq_p Z$

Proof: by putting the two tranformations into series

NP-hardness (4)

Definition

A decision problem X is *NP-hard*,
if all problems $Y \in NP$ can be reduced to it
(that is, if $Y \leq_p X$ holds for all $Y \in NP$)

Definition

A decision problem X is *NP-complete*,
if $X \in NP$ and X is NP-hard.

Intuition:

- NP-complete problems are the hardest problems in NP
- Recall: NP is huge and contains tons of important problems
- NP-complete problems are considered to be intractable

NP-hardness (5)

Theorem

If one NP-complete problem X has a polynomial time algorithm then all NP-complete problems have polynomial time algorithms (and hence P=NP)

Cook's theorem (1971)

SAT is NP-complete.

- Stephen Cook (born 1939):
  American-Canadian computer scientist and mathematician

NP-hardness: Satisfiability

Satisfiability (SAT)

Instance:
a logical formula Φ in CNF over logical variable set $X = \{x_1, \ldots, x_n\}$

Question: does there exist a truth setting for X that satisfies Φ?

- 3-SAT: all clauses contain three literals

Examples

$$\begin{aligned}\Phi &= (x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z) \\ \Phi &= (x \vee y) \wedge (\neg x \vee y) \wedge (x \vee \neg y) \wedge (\neg x \vee \neg y)\end{aligned}$$

Theorem

SAT is NP-complete.
3-SAT is NP-complete.

NP-hardness: Integer programming

Integer programming (ILP)

Instance: an integer matrix A; an integer vector b

Question: does there exist an integer vector x with $Ax \leq b$?

Theorem

ILP is NP-complete.

Consequence: Every problem in NP can be modelled as an ILP.

NP-hardness: Clique

Clique

Instance: a graph $G = (V, E)$; an integer k

Question: does G contain a clique of size (at least) k?

Theorem

CLIQUE is NP-complete.

NP-hardness: Independent set / Vertex cover

Independent set (IS)

Instance: a graph $G = (V, E)$; an integer k
Question: does G contain an independent set of size (at least) k?
(a set of vertices that does not span any edge)

Vertex cover (VC)

Instance: a graph $G = (V, E)$; an integer k
Question: does G contain a vertex cover of size (at most) k?
(a set of vertices that touches every edge)

Theorem

IS is NP-complete.
VC is NP-complete.

NP-hardness: Exact cover

Exact cover (Ex-Cov)

Instance: a ground set X; subsets $S_1, \ldots, S_m$ of X

Question: do there exist some subsets S_i that form a partition of X?

Theorem

Ex-Cov is NP-complete.

NP-hardness: Subset Sum

Subset Sum (SS)

Instance: positive integers $a_1, \ldots, a_n$; a bound b

Question: does there exist an index set $I \subseteq \{1, \ldots, n\}$ with $\sum_{i \in I} a_i = b$?

Theorem

SS is NP-complete.

NP-hardness: Hamiltonian cycle / TSP

Directed Hamiltonian cycle (dir-HC)

Instance: a directed graph (X, A)
Question: does this graph contain a directed Hamiltonian cycle?

Hamiltonian cycle (HC)

Instance: an undirected graph $G = (V, E)$
Question: does G contain a Hamiltonian cycle?

Travelling Salesman Problem (TSP)

Instance: cities $1, \ldots, n$; distances $d(i, j)$; a bound B
Question: does there exist a roundtrip of length at most B?

Theorem

dir-HC is NP-complete. HC is NP-complete. TSP is NP-complete.

NP versus coNP (1)

Recall:

Definition

A decision problem X lies in the complexity class NP,
if the YES-instances of X possess certificates of polynomial length that can be verified in polynomial time

A decision problem X is *NP*-complete,
if $X \in NP$ and all problems $Y \in NP$ can be reduced to it.

Now we define:

Definition

A decision problem X lies in the complexity class coNP,
if the NO-instances of X possess certificates of polynomial length that can be verified in polynomial time

A decision problem X is *coNP*-complete,
if $X \in coNP$ and all problems $Y \in coNP$ can be reduced to it.

NP versus coNP (2)

Problems in $NP \cap coNP$ have

- good certificates for YES-instances
- good certificates for NO-instances

Example

Linear Programming (LP):
Instance: a matrix A; vectors c and b; a bound t
Question: does there exist a real vector x with $Ax \leq b$ and $cx \leq t$?

- LP lies in NP
- LP lies in coNP

- Similar: MaxFlow in NP and in coNP
- Recall: Duality theorems

NP versus coNP (3)

- FACT: $P \subseteq NP \cap coNP$
- Some people think that $P \neq NP \cap coNP$
- Some people think that $P = NP \cap coNP$

- Most people think that $NP \neq coNP$.

Theorem

If coNP contains some NP-complete problem X, then NP=coNP.

Hence:

- X being NP-complete is indication for $X \notin coNP$
- X being coNP-complete is indication for $X \notin NP$

Homework 4

- Read the paper by Lenstra & Rinnooy Kan
- Recommended Exercises:
  69, 73, 75, 78, 80, 84, 86, 88

Collection of exercises can be downloaded from:
  `http://www.win.tue.nl/~gwoegi/optimization/`

Attention!
Weeks 2-5 (Sep 8; Sep 15; Sep 22; Sep 29):

- Tuesday 1+2: instructions
- Tuesday 3+4: lecture