Computational Complexity; slides 5, HT 2022
Reductions, NP-hardness

Prof. Paul W. Goldberg (Dept. of Computer Science, University of Oxford)

HT 2022

NP-Completeness Proofs

To prove that a problem $\mathcal{X}$ is NP-complete, we now just have to perform two steps:

1. Show that $\mathcal{X} \in \mathrm{NP}$ usually easy
2. Find a known NP-complete problem $\mathcal{X}'$ and reduce $\mathcal{X}' \leq_p \mathcal{X}$. the FUN part

Thousands of problem have now been shown to be NP-complete (See Garey and Johnson for an early survey); Karp 1972, "reducibility among combinatorial problems" kicked-off this work

A relevant quote

Proving NP-completeness results is an important ingredient of our methodology for studying computational problems. It is also something of an art form.

start of chapter 9 of Papadimitriou's textbook

NP-Completeness Proofs

Coming up next: some examples.

CNF-SAT$\leq_p$3-SAT (BTW, goes back to Cook's paper)

$\{X_1, X_2, \ldots, X_n\} \mapsto \{X_1, X_2, X_{new}\}, \{\neg X_{new}, X_3, \ldots, X_n\}$
repeat until clause lengths ≤ 3

3-SAT is a more convenient starting-point of reductions than unrestricted SAT.

3-SAT$\leq_p$INTEGER PROGRAMMING (simple but important)
3-SAT$\leq_p$IND SET$\leq_p$CLIQUE
3-SAT$\leq_p$DIRECTED HAMILTONIAN PATH
3-SAT$\leq_p$SUBSET SUM$\leq_p$KNAPSACK

NP-Completeness of INTEGER PROGRAMMING

IP: Input: a set of linear constraints, Question: can we satisfy them with integer values?
3-SAT$\leq_p$IP

X_i in 3-SAT instance $\mapsto x_i$ in IP instance.

$\forall i$, include constraints $0 \leq x_i \leq 1$
(idea: 0 means F, 1 means T)

$\{X_i, X_j, X_k\} \mapsto x_i + x_j + x_k \geq 1$
$\{X_i, \neg X_j, X_k\} \mapsto x_i + (1 - x_j) + x_k \geq 1$
and similarly for more than one negated literal

Example

$\{\{X_1, X_2, X_3\}, \{\neg X_1, \neg X_2, X_4\}\}$
is reduced to the following IP:
$0 \leq x_1 \leq 1$, $0 \leq x_2 \leq 1$
$x_1 + x_2 + x_3 \geq 1$, $(1 - x_1) + (1 - x_2) + x_4 \geq 1$

NP-Completeness of CLIQUE

CLIQUE: Given G, k, does G contain a clique of order $\geq k$?

Theorem

CLIQUE *is* NP*-complete.*

It's convenient to reduce from 3-SAT to IND SET and from there to CLIQUE.

3-SAT$\leq_p$IND SET: each clause of a 3-SAT instance becomes a triangle in the graph. Label vertices with the literals.
If the formula had m clauses, the graph now has $3m$ vertices. Is there an independent set of size m?

(idea: choice of vertex in each triangle corresponds to choice of literal that gets satisfied)

Add new edges between any pair of vertices labelled by a variable X_i and its negation $\neg X_i$.
Any n-independent set corresponds to a satisfying assignment.

DIRECTED HAMILTONIAN PATH

Input: G: directed graph.

Problem: Is there a directed path in G containing every vertex exactly once?

Theorem

DIRECTED HAMILTONIAN PATH *is NP-complete*

NP-Completeness of Directed Hamiltonian Path

Directed Hamiltonian Path

Input:	G: directed graph.
Problem:	Is there a directed path in G containing every vertex exactly once?

Theorem

Directed Hamiltonian Path *is NP-complete*

Proof.

1. Directed Hamiltonian Path $\in$NP.

   Take the path to be the certificate.

NP-Completeness of Directed Hamiltonian Path

Directed Hamiltonian Path

Input: G: directed graph.

Problem: Is there a directed path in G containing every vertex exactly once?

Theorem

Directed Hamiltonian Path *is NP-complete*

Proof.

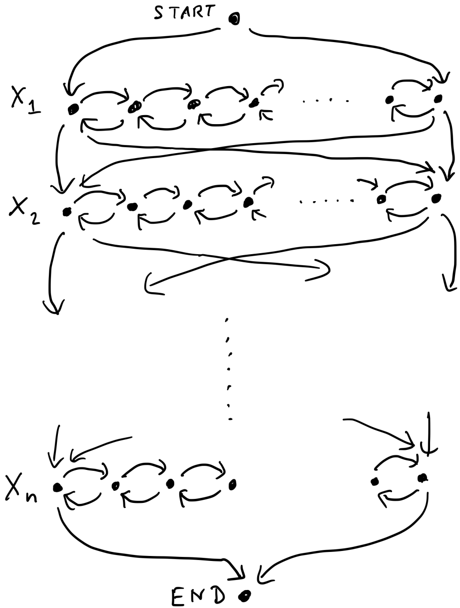
1. Directed Hamiltonian Path $\in$NP.

   Take the path to be the certificate.

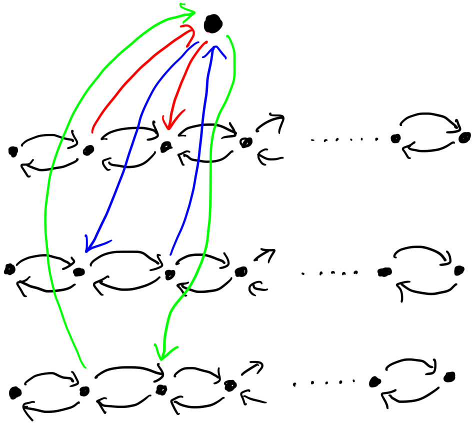
2. Directed Hamiltonian Path is NP-hard.

   3-Satisfiability $\leq_p$ Directed Hamiltonian Path

from 3-SAT to DIRECTED HAMILTONIAN PATH



from 3-SAT to DIRECTED HAMILTONIAN PATH



Digression: how to design reductions

Show that problem $\mathcal{X}$ (Dir. Hamiltonian Path) is NP-hard.

Which problem to reduce to $\mathcal{X}$:

- Arguably, the most important part is to decide where to start from; e.g. which problem to reduce to Directed Hamiltonian Path — something graph-theoretic?
- Considerations:
  - Is there an NP-complete problem similar to $\mathcal{X}$?
    (E.g. Clique and Independent Set)
  - It is not always beneficial to choose a problem of the same type
    (E.g. reducing a graph problem to a graph problem)
    - For instance, Clique, Independent Set are "local" problems (is there a set of vertices inducing some structure)
    - Hamiltonian Path is a "global" problem
      (find a structure containing all vertices)

How to design the reduction:

- Does your problem come from an optimisation problem?
  If so: a maximisation problem? a minimisation problem?

NP-Completeness of Subset Sum

Subset Sum

Input: A collection of positive integers $S := \{a_1, \dots, a_k\}$ and a target integer t.

Problem: Is there a subset $T \subseteq S$ such that $\sum_{a_i \in T} a_i = t$?

Theorem. Subset Sum is NP-complete

Proof.

1. Subset Sum $\in$ NP.

   Take T to be the certificate.

2. Subset Sum is NP-hard.

   CNF-Sat $\leq_p$ Subset Sum (example next slide)

Example

$$(X_1 \vee X_2 \vee X_3) \wedge (\neg X_1 \vee \neg X_4) \wedge (X_4 \vee X_5 \vee \neg X_2 \vee \neg X_3)$$

		X_1	X_2	X_3	X_4	X_5	C_1	C_2	C_3
t_1	=	1	0	0	0	0	1	0	0
f_1	=	1	0	0	0	0	0	1	0
t_2	=		1	0	0	0	1	0	0
f_2	=		1	0	0	0	0	0	1
t_3	=			1	0	0	1	0	0
f_3	=			1	0	0	0	0	1
t_4	=				1	0	0	0	1
f_4	=				1	0	0	1	0
t_5	=					1	0	0	1
f_5	=					1	0	0	0
$m_{1,1}$	=						1	0	0
$m_{1,2}$	=						1	0	0
$m_{2,1}$	=						0	1	0
$m_{3,1}$	=						0	0	1
$m_{3,2}$	=						0	0	1
$m_{3,3}$	=						0	0	1
t	=	1	1	1	1	1	3	2	4

$\text{SAT} \leq_p \text{SUBSET SUM}$ (the general construction)

Given: $\varphi := C_1 \wedge \cdots \wedge C_k$ in conjunctive normal form.

(for numbers in base 10: at most 9 literals per clause)

Let $X_1, \ldots, X_n$ be the variables in φ. For each X_i let

$$t_i := a_1 \ldots a_n c_1 \ldots c_k \quad \textit{where} \quad a_j := \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \quad c_j := \begin{cases} 1 & X_i \text{ occurs in } C_j \\ 0 & \text{otherwise} \end{cases}$$

$$f_i := a_1 \ldots a_n c_1 \ldots c_k \quad \textit{where} \quad a_j := \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \quad c_j := \begin{cases} 1 & \neg X_i \text{ occurs in } C_j \\ 0 & \text{otherwise} \end{cases}$$

$\textsc{Sat} \leq_p \textsc{Subset Sum}$ (the general construction)

Further, for each clause C_i take $r := |C_i| - 1$ integers $m_{i,1}, \ldots, m_{i,r}$

where $m_{i,j} := c_i \ldots c_k$ with $c_j := \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}$

Definition of S: Let

$$S := \{t_i, f_i : 1 \leq i \leq n\} \cup \{m_{i,j} : 1 \leq i \leq k, \quad 1 \leq j \leq |C_i| - 1\}$$

Target: Finally, choose as target

$$t := a_1 \ldots a_n c_1 \ldots c_k \text{ where } a_i := 1 \text{ and } c_i := |C_i|$$

Claim: There is $T \subseteq S$ with $\sum_{a_i \in T} a_i = t$ iff φ is satisfiable.

NP-Completeness of Subset Sum

Let $\varphi := \bigwedge C_i$ C_i: clauses

Show. If φ is satisfiable, then there is $T \subseteq S$ with $\sum_{s \in T} s = t$.

Let β be a satisfying assigment for φ

Set $T_1 := \{t_i : \beta(X_i) = 1 \quad 1 \leq i \leq m\} \cup \{f_i : \beta(X_i) = 0 \quad 1 \leq i \leq m\}$

Further, for each clause C_i let r_i be the number of satisfied literals in C_i

(with resp. to β).

Set $T_2 := \{m_{i,j} : 1 \leq i \leq k, \quad 1 \leq j \leq |C_i| - r_i\}$

and define $T := T_1 \cup T_2$.

It follows: $\sum_{s \in T} s = t$

NP-Completeness of SUBSET SUM

Show. If there is $T \subseteq S$ with $\sum_{s \in T} s = t$, then φ is satisfiable.

Let $T \subseteq S$ s.th. $\sum_{s \in T} s = t$

Define $\beta(X_i) = \begin{cases} 1 & \text{if } t_i \in T \\ 0 & \text{if } f_i \in T \end{cases}$

This is well defined as for all i: $t_i \in T$ or $f_i \in T$ but not both.

Further, for each clause, there must be one literal set to 1 as for all i, the $m_{i,j} : m_{i,j} \in S$ do not sum up to the number of literals in the clause.

NP-completeness of Knapsack

Knapsack

Input: A set $I := \{1, \ldots, n\}$ of items
each of value v_i and weight w_i for $1 \le i \le n$
target value t weight limit ℓ

Problem: Is there $T \subseteq I$ such that

- $\sum_{i \in T} v_i \ge t$
- $\sum_{i \in T} w_i \le \ell$

Theorem. Knapsack is NP-complete

1. Knapsack $\in$ NP
   Take T as certificate.
2. Knapsack is NP-hard
   By reduction Subset Sum $\le_p$ Knapsack

Key point: Knapsack is "more general/expressive" than Subset Sum

SUBSET SUM $\leq_p$ KNAPSACK (the details)

reminder: SUBSET SUM

Given:	$S := \{a_1, \ldots, a_n\}$	collection of positive integers
	t	target integer

Problem: Is there a subset $T \subseteq S$ such that $\sum_{a_i \in T} a_i = t$?

SUBSET SUM $\leq_p$ KNAPSACK (the details)

reminder: SUBSET SUM

Given: $S := \{a_1, \ldots, a_n\}$ collection of positive integers

t target integer

Problem: Is there a subset $T \subseteq S$ such that $\sum_{a_i \in T} a_i = t$?

Reduction: From this input to SUBSET SUM construct

- $I := \{1, \ldots, n\}$: set of items
- $v_i = w_i = a_i$ for all $1 \leq i \leq n$
- target value $t' := t$ weight limit $\ell := t$

SUBSET SUM $\leq_p$ KNAPSACK (the details)

reminder: SUBSET SUM

Given: $S := \{a_1, \ldots, a_n\}$ collection of positive integers

t target integer

Problem: Is there a subset $T \subseteq S$ such that $\sum_{a_i \in T} a_i = t$?

Reduction: From this input to SUBSET SUM construct

- $I := \{1, \ldots, n\}$: set of items
- $v_i = w_i = a_i$ for all $1 \leq i \leq n$
- target value $t' := t$ weight limit $\ell := t$

Clearly: For every $T \subseteq S$

$$\sum_{a_i \in T} a_i = t \quad \Longleftrightarrow \quad \begin{array}{l} \sum_{a_i \in T} v_i \geq t' \quad = t \\ \sum_{a_i \in T} w_i \leq \ell \quad = t \end{array}$$

Hence: The reduction is correct and in polynomial time.