CS21
Decidability and Tractability

Lecture 22

February 28, 2014

Outline

- the class NP
  - NP-complete problems: subset sum
  - NP-complete problems: NAE-3-SAT, max-cut
- the class co-NP
- the class NP ∩ coNP

SUBSET-SUM is NP-complete

Theorem: the following language is NP-complete:

SUBSET-SUM = {(S … :

there is a subse…

- Proof:
  - Part 1: SUBSET-SU…
  - Part 2: SUBSET-SUM is NP-hard.
    - reduce from?

our reduction had better produce super-polynomially large B (unless we want to prove P=NP)

SUBSET-SUM is NP-complete

- We are reducing from the language:

3SAT = { φ : φ is a 3-CNF formula that has a satisfying assignment }

to the language:

SUBSET-SUM = {(S = {a_1, a_2, a_3, …, a_k}, B) : there is a subset of S that sums to B}

SUBSET-SUM is NP-complete

- $\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_3) \wedge \ldots \wedge (\ldots)$
- Need integers to play the role of truth assignments
- For each variable x_i include two integers in our set S:
  - x_i^{TRUE} and x_i^{FALSE}
- set B so that exactly one must be in sum

SUBSET-SUM is NP-complete

x_1^{TRUE} = 1 0 0 0 … 0

x_1^{FALSE} = 1 0 0 0 … 0

x_2^{TRUE} = 0 1 0 0 … 0

x_2^{FALSE} = 0 1 0 0 … 0

…

x_m^{TRUE} = 0 0 0 0 … 1

x_m^{FALSE} = 0 0 0 0 … 1

B = 1 1 1 1 … 1

- every choice of one from each $(x_i^{TRUE}, x_i^{FALSE})$ pair sums to B
- every subset that sums to B must choose one from each $(x_i^{TRUE}, x_i^{FALSE})$ pair

SUBSET-SUM is NP-complete

- $\varphi = (x_1 \lor x_2 \lor \neg x_3) \land (\neg x_1 \lor x_4 \lor x_3) \land \ldots \land (\ldots)$
- Need to force subset to "choose" at least one true literal from each clause
- Idea:
  - add more digits
  - one digit for each clause
  - set B to force each clause to be satisfied.

SUBSET-SUM is NP-complete

– $\varphi = (x_1 \lor x_2 \lor \neg x_3) \land (\neg x_1 \lor x_4 \lor x_3) \land \ldots \land (\ldots)$

x_1^{TRUE}	= 1 0 0 0 … 0	1
x_1^{FALSE}	= 1 0 0 0 … 0	0
x_2^{TRUE}	= 0 1 0 0 … 0	1
x_2^{FALSE}	= 0 1 0 0 … 0	0
x_3^{TRUE}	= 0 0 1 0 … 0	0
x_3^{FALSE}	= 0 0 1 0 … 0	1
…		:
B	= 1 1 1 1 … 1	? ? ? … ?

clause 1
clause 2
clause 3
:
clause k

SUBSET-SUM is NP-complete

– B = 1 1 1 1 … 1 ? ? ? ?
– if clause i is satisfied sum might be 1, 2, or 3 in corresponding column.
– want ? to "mean" ≥ 1
– solution: set ? = 3
– add two "filler" elements for each clause i:
– $FILL1_i$ = 0 0 0 0 … 0 0 … 0 1 0 … 0
– $FILL2_i$ = 0 0 0 0 … 0 0 … 0 1 0 … 0

column for clause i

SUBSET-SUM is NP-complete

- Reduction: m variables, k clauses
  - for each variable x_i:
    - x_i^{TRUE} has ones in positions k + i and {j : clause j includes literal x_i}
    - x_i^{FALSE} has ones in positions k + i and {j : clause j includes literal $\neg x_i$}
  - for each clause i:
    - $FILL1_i$ and $FILL2_i$ have one in position i
  - bound B has 3 in positions 1…k and 1 in positions k+1…k+m

SUBSET-SUM is NP-complete

- Reduction computable in poly-time?
- YES maps to YES?
  - choose one from each $(x_i^{TRUE}, x_i^{FALSE})$ pair corresponding to a satisfying assignment
  - choose 0, 1, or 2 of filler elements for each clause i depending on whether it has 3, 2, or 1 true literals
  - first m digits add to 1; last k digits add to 3

SUBSET-SUM is NP-complete

- NO maps to NO?
  - at most 5 ones in each column, so no carries to worry about
  - first m digits of B force subset to choose exactly one from each $(x_i^{TRUE}, x_i^{FALSE})$ pair
  - last k digits of B require at least one true literal per clause, since can only sum to 2 using filler elements
  - resulting assignment must satisfy φ

Not-All-Equal 3SAT

$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_3) \wedge \ldots \wedge (\ldots)$

Theorem: the following language is NP-complete:

NAE3SAT = {φ : φ is a 3-CNF formula for which there exists a truth assignment in which every clause has at least 1 true literal and at least 1 false literal}

- Proof:
  - Part 1: NAE3SAT ∈ NP. Proof?
  - Part 2: NAE3SAT is NP-hard. Reduce from?

NAE3SAT is NP-complete

- We are reducing from the language:

  CIRCUIT-SAT = {C : C is a Boolean circuit for which there exists a satisfying truth assignment}

  to the language:

  NAE3SAT = {φ : φ is a 3-CNF formula for which there exists a truth assignment in which every clause has at least 1 true literal and at least 1 false literal}

NAE3SAT is NP-complete

- Recall reduction to 3SAT
  - variables $x_1, x_2, \ldots, x_n$, gates $g_1, g_2, \ldots, g_m$
  - produce clauses:

not all true in a satisfying assignment

$\vee$ g_i, z_1 z_2 ⇒
- $(\neg z_1 \vee g_i)$
- $(\neg z_2 \vee g_i)$
- $(\neg g_i \vee z_1 \vee z_2)$

$\wedge$ g_i, z_1 z_2 ⇒
- $(\neg g_i \vee z_1)$
- $(\neg g_i \vee z_2)$
- $(\neg z_1 \vee \neg z_2 \vee g_i)$

$\neg$ g_i, z ⇒
- $(g_i \vee z)$
- $(\neg z \vee \neg g_i)$

output gate g_m:
- (g_m)

NAE3SAT is NP-complete

- modified reduction to NAE3SAT
  - variables $x_1, x_2, \ldots, x_n$, gates $g_1, g_2, \ldots, g_m$
  - produce clauses:

$\vee$ g_i, z_1 z_2 ⇒
- $(\neg z_1 \vee g_i \vee w)$
- $(\neg z_2 \vee g_i \vee w)$
- $(\neg g_i \vee z_1 \vee z_2)$

$\wedge$ g_i, z_1 z_2 ⇒
- $(\neg g_i \vee z_1 \vee w)$
- $(\neg g_i \vee z_2 \vee w)$
- $(\neg z_1 \vee \neg z_2 \vee g_i)$

$\neg$ g_i, z ⇒
- $(g_i \vee z \vee w)$
- $(\neg z \vee \neg g_i \vee w)$

output gate g_m:
- $(g_m \vee w)$

NAE3SAT is NP-complete

- Does the reduction run in polynomial time?
- YES maps to YES
  - already know how to get a satisfying assignment to the BLUE variables
  - set w = FALSE

- $(\neg z_1 \vee g_i \vee w)$
- $(\neg z_2 \vee g_i \vee w)$
- $(\neg g_i \vee z_1 \vee z_2)$
- $(\neg g_i \vee z_1 \vee w)$
- $(\neg g_i \vee z_2 \vee w)$
- $(\neg z_1 \vee \neg z_2 \vee g_i)$
- $(g_i \vee z \vee w)$
- $(\neg z \vee \neg g_i \vee w)$
- $(g_m \vee w)$

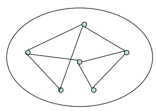
NAE3SAT is NP-complete

- NO maps to NO
  - given NAE assignment A
  - complement A' is a NAE assignment
  - A or A' has w = FALSE
  - must have TRUE BLUE variable in every clause
  - we know this implies C satisfiable

- $(\neg z_1 \vee g_i \vee w)$
- $(\neg z_2 \vee g_i \vee w)$
- $(\neg g_i \vee z_1 \vee z_2)$
- $(\neg g_i \vee z_1 \vee w)$
- $(\neg g_i \vee z_2 \vee w)$
- $(\neg z_1 \vee \neg z_2 \vee g_i)$
- $(g_i \vee z \vee w)$
- $(\neg z \vee \neg g_i \vee w)$
- $(g_m \vee w)$

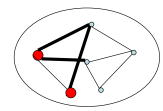
MAX CUT

- Given graph G = (V, E)
  - a cut is a subset $S \subset V$
  - an edge (x, y) crosses the cut if $x \in S$ and $y \in V - S$ or $x \in V - S$ and $y \in S$
  - search problem:
    find cut maximizing number of edges crossing the cut



MAX CUT

- Given graph G = (V, E)
  - a cut is a subset $S \subset V$
  - an edge (x, y) crosses the cut if $x \in S$ and $y \in V - S$ or $x \in V - S$ and $y \in S$
  - search problem:
    find cut maximizing number of edges crossing the cut



MAX CUT

Theorem: the following language is NP-complete:

MAX CUT = {(G = (V, E), k) : there is a cut $S \subset V$ with at least k edges crossing it}

- Proof:
  - Part 1: MAX CUT $\in$ NP. Proof?
  - Part 2: MAX CUT is NP-hard.
    - reduce from?

MAX CUT is NP-complete

- We are reducing from the language:

NAE3SAT = {φ : φ is a 3-CNF formula for which there exists a truth assignment in which every clause has at least 1 true literal and at least 1 false literal}

to the language:

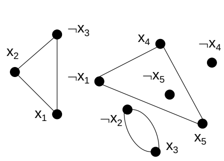
MAX CUT = {(G = (V, E), k) : there is a cut $S \subset V$ with at least k edges crossing it}

MAX CUT is NP-complete

- The reduction:
  - given instance of NAE3SAT (n nodes, m clauses):

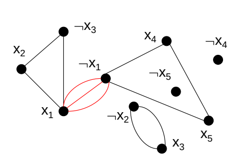
$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_5) \wedge \ldots \wedge (\neg x_2 \vee x_3 \vee x_3)$

  - produce graph G = (V, E) with node for each literal



- triangle for each 3-clause
- parallel edges for each 2-clause

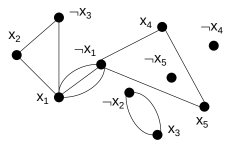
MAX CUT is NP-complete



- triangle for each 3-clause
- parallel edges for each 2-clause

- if cut selects TRUE literals, each clause contributes 2 if NAE, and < 2 otherwise
- need to penalize cuts that correspond to inconsistent truth assignments
- add n_i parallel edges from x_i to $\neg x_i$ (n_i = # occurrences)
  (repeat variable in 2-clause to make 3-clause for this calculation)

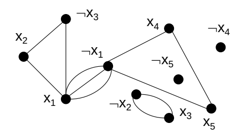
MAX CUT is NP-complete



- triangle for each 3-clause
- parallel edges for each 2-clause
- n_i parallel edges from x_i to $\neg x_i$
- set k = 5m

- YES maps to YES
  - take cut to be TRUE literals in a NAE truth assignment
  - contribution from clause gadgets: 2m
  - contribution from (x_i, $\neg x_i$) parallel edges: 3m

MAX CUT is NP-complete



- triangle for each 3-clause
- parallel edges for each 2-clause
- n_i parallel edges from x_i to $\neg x_i$
- set k = 5m

- NO maps to NO
  - **Claim**: if cut has x_i, $\neg x_i$ on same side, then can move one to opposite side without decreasing # edges crossing cut
  - contribution from (x_i, $\neg x_i$) parallel edges: 3m
  - contribution from clause gadgets must be 2m
  - conclude: there is a NAE assignment

MAX CUT is NP-complete

Claim: if cut has x_i, $\neg x_i$ on same side, then can move one to opposite side without decreasing # edges crossing cut

- Proof: