Problem Set 4

Jaan Tollander de Balsch

March 21, 2020

H4.1

(a)

The problem is in **NP** because the problem is a special case on SAT which is in **NP**. Next, we will show that the problem is **NP**-complete by reducing SAT to this problem in polynomial time. For this reduction, we use the idea by Jones (2018).

Let ϕ be a SAT formula in CNF form. Then, new formula ϕ' in CNF that is logically equivalent to ϕ is constructed as follows.

Add all clauses c in ϕ that have 1 or 2 literals to ϕ' unmodified.

For all clauses c in ϕ with more than 3 literals

1) Create new clause c' by substituting each negative literal $\neg x$ in clause c by new variable z.
2) Add clauses c', $(x \vee z)$ and $(\neg x \vee \neg z)$ to the formula ϕ'.

---

x	z	(z ∧ ((x ∨ z) ∧ (¬x ∨ ¬z)))
F	F	F
F	T	T
T	F	F
T	T	F

Proof: Literal z is forced to act as literal $\neg x$ because

$$z \land (x \lor z) \land (\neg x \lor \neg z)$$

is true only if z and $\neg x$ are true, which is evident from its truth table.

(b)

1-IN-3SAT is in **NP** since verifying the correct solution is same as for normal SAT. We show **NP**-completeness by reducing 3SAT to 1-IN-3SAT.

We can reduce 3SAT to 1-IN-3-SAT by replacing each clause $c = (x \lor y \lor z)$ in 3SAT with the clause

$$c' = (\neg x \lor a \lor b) \land (b \lor y \lor c) \land (c \lor d \lor \neg z).$$

We can prove that c' is 1-IN-3 satisfiable if and only if c is satisfiable by using a truth table.

(	¬	x	∨	a	∨	b	)	∧	(	b	∨	y	∨	c	)	∧	(	c	∨	d	∨	¬	z	)
	0	1		1		0				0		1		0				0		1		0	1	
	0	1		1		0				0		1		0				0		0		1	0	
	0	1		1		0				0		0		1				1		0		0	1	
	0	1		0		1				1		0		0				0		0		1	0	
	1	0		0		0				0		1		0				0		1		0	1	
	1	0		0		0				0		1		0				0		0		1	0	
	1	0		0		0				0		0		1				1		0		0	1	
	1	0		0		0				0		0		0				0		0		1	0	

As can be seen, each of the first 7 rows are 1-in-3 satisfiable, but the last row representing assignment $x = 0, y = 0, z = 0$ is not 1-in-3 satisfiable.

H4.2

The solution was inspired by Mount (2017).

i) We show that dominating set (**DS**) is in **NP**.
ii) We reduce vertex cover (**VS**) to dominating set in logspace.

(i)

Given a graph $G = (V, E)$, set $U \subseteq V$ is a dominating set if $U \cup W = V$ where W is the set of vertices adjacent to vertices in U, defined as

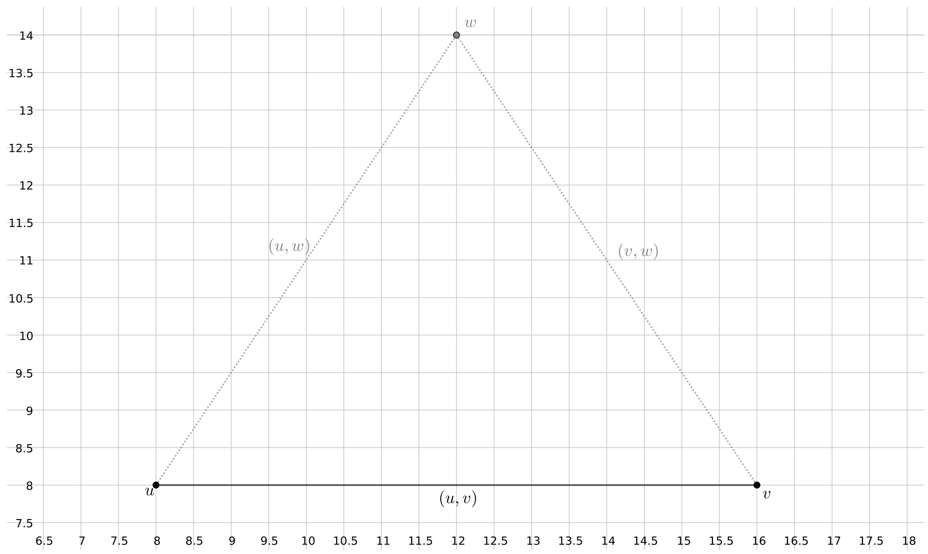
$$W = \{w \mid (u, w) \in E, u \in U\}.$$

The verification can be done in polynomial time $O(|U||E|) = O(|V||E|)$, since $|U| \leq |V|$.

(ii)

We reduce deciding the vertex cover for undirected graph $G = (V, E)$ and integer k to deciding dominating set for undirected graph $G' = (V \cup V', E \cup E')$ and integer $k + n$ where n is the number of isolated nodes. We include n in the size since the dominating set must include all isolated nodes by definition, but the vertex cover does not include them.

The graph G' contains the original graph G and auxiliary vertices V' and auxiliary edges E'.



We can force the inclusion of atleast one of the vertices $(u, v) \in E$ in the dominating set such that it satisfies the condition for vertex cover as follows.

For each edge $(u, v) \in E$, we add vertex w to V' and edges (u, w) and (v, w) to E'. The dominating set must now include atleast one of the vertices in u, v or w. If w is included, we can replace it with u or v because the vertices u, v and w are adjacent, thus, the set still remains a dominating set.

Now, we find a dominating set $U' \subseteq V \cup V'$ of size $k + n$ in graph G'. Then, we create a new dominating set $U \subseteq V$ such that for each vertex $u \in U'$

1) If $u \in V$ and u is not isolated, add u to U.
2) If $u \in V'$ add one of its adjacent vertices to U.

The dominating set U is a vertex cover of size k of graph G.

The reduction is computable in constant space, and is therefore logspace computable.

H4.3

(i)

A Turing machine M (with input and output) on input $k; x$, where $k \geq 0$, computes the mapping $x \mapsto 1^{k|x|}$ in linear time and constant space as follows.

1) Copy k to work tape 1.
2) Copy k from work tape 1 to work tape 2, which will act as a counter. Input tape cursor should be now in the first symbol of x.
3) Write 1 for k times on the output tape by reducing the counter each time 1 is written until it reaches zero.
4) Move the input tape cursor to the next symbol of x.
5) Copy k from work tape 1 to work tape 2.
6) Repeat from step (3) or halt if input tape cursor is on $\sqcup$.

(ii)

We have padded universal set

$$U = \{M; x; 1^{|M||x|} \mid \text{Turing machine } M \text{ accepts input } x \text{ in space } |x|\}$$

(iii)

(iiii)

References

Jones, K., 2018. *What is the complexity of determining whether or not conjunction of positive cnf and negative cnf is satisfiable?* Available at: <https://cs.stackexchange.com/q/79218>.

Mount, D., 2017. CMSC 451 : Lecture 22 Clique , Vertex Cover , and Dominating Set. [online] pp.1–5. Available at: <http://www.cs.umd.edu/class/fall2017/cmsc451-0101/Lects/lect21-np-clique-vc-ds.pdf>.