CSE 417 HW 8 Q1&2

Yang Zhang, 1030416, zhy9036@uw.edu

Problem 1

(a) MaxCut can be testified in Polynomial time:
Suppose given 2 sets of vertices ***g1, g2*** (which are no overlapping with each other), integer ***k,*** and a set of edges ***e***. Then only need to spend **O(n)** time to loop over ***e***, then counts the number of edges that have vertices both in ***g1*** and ***g2***.
Therefore MaxCut is in **NP.**

(b) Algorithm for MaxCut:

```
globle k
func max_cut(subset, index i, vertexes v, edges e):
     if i < len(v)
          subset.append(v[i])
          verify(subset, e)
          max_cut(subset, i+1, v, e)
          subset.remove(v[i])
          verify(subset, e)
          max_cut(subset, i+1, v, e)
     endif
endfunc

func verify(subset, edge):
     count = 0
     for e in edge:
          if subset only contains one vertex in e
               count++
          endif
     endfor
     if count >= k
          return subset
     endif
endfunc
```

Proof:
Since the above algorithm contains all the cases that can divide a graph into 2 parts, and verify every single case. Therefore, the algorithm is correct.

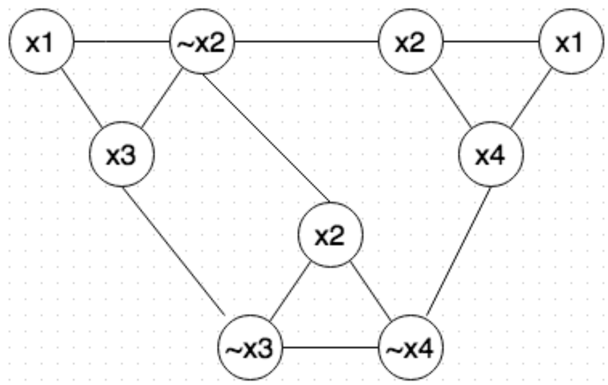
Running Time Analyze:

Each level of recursion reduce the size by 1 and there are n level of recursion. Therefore, the total time is n*(n-1)*(n-2)*…*1, which is $O(n^n)$

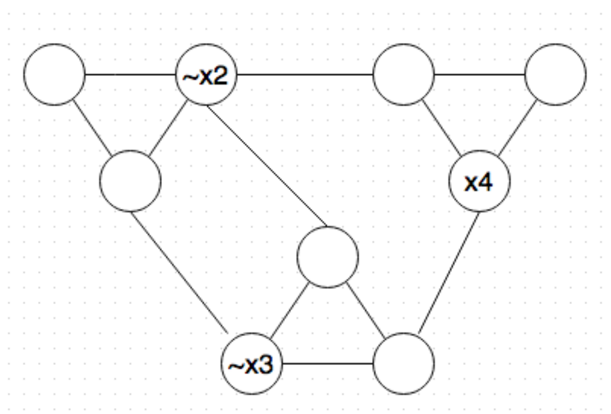
Problem 2

(x1 ∨ x2 ∨ x4) ∧ (x1 ∨ ~x2 ∨ x3) ∧ (x2 ∨ ~x3 ∨ ~x4)

(a) shown as figure below:



(b) x1 = x2 = x3 = False, x4 = True



The indSet = {~x2, ~x3, x4}
x1 doesn't show in the independent set, therefore x1 can be either True or False.

1) x1 = x2 = x3 = False, x4 = True
2) x1 = x4 = True, x2 = x3 = False

Both assignments satisfies the Boolean expression.

CSE 417 HW 8 Q3&4

Yang Zhang, 1030416, zhy9036@uw.edu

Problem 3

(x1 ∨ x2 ∨ x4) ∧ (x1 ∨ ~x2 ∨ x3) ∧ (x2 ∨ ~x3 ∨ ~x4)

(a) shown as table below:

	Variables				Clauses		
	x1	x2	x3	x4	(x1 ⋁ x2 ⋁ x4)	(x1 ⋁ ~x2 ⋁ x3)	(x2 ⋁ ~x3 ⋁ ~x4)
w1 (x1)	1	0	0	0	1	1	0
w2 (~x1)	1	0	0	0	0	0	0
w3 (x2)		1	0	0	1	0	1
w4 (~x2)		1	0	0	0	1	0
w5 (x3)			1	0	0	1	0
w6 (~x3)			1	0	0	0	1
w7(x4)				1	1	0	0
w8(~x4)				1	0	0	1
W9					1	0	0
W10					1	0	0
W11						1	0
w12						1	0
w13							1
w14							1
C	1	1	1	1	3	3	3

(b)

x1 = True → 1000 110 (w1)

x2 = False → 0100 010 (w4)

x3 = False → 0010 001 (w5)

x4 = True → 0001 100 (w7)

add above = **1111221 < 1111333, so the subset must contains slacks**

There are four subsets:

Set1: {w1, w4, w5, w7, w9, w11, w13, w14}

Set2: {w1, w4, w5, w7, w10, w11, w13, w14}

Set3: {w1, w4, w5, w7, w9, w12, w13, w14}

Set4: {w1, w4, w5, w7, w10, w12, w13, w14}

There is only one assignment:
x2 = x3 = False
x1 = x4 = True
And it satisfy Boolean expression

Problem 4

(a) Yes, because interval-scheduling problem is in NP and any problem in NP can be reduced to Vertex Cover.
(b) Unknown, because it would resolve the question of whether P = NP

CSE 417 HW 8 Q5

Yang Zhang, 1030416, zhy9036@uw.edu

Problem 5

(a) This argument is **false.** The reason is that the argument only proofed that

$$\mathbf{P} \leq_{\mathbf{p}} \mathbf{NP}$$

However, proofing P = NP also need to proof $\mathbf{NP} \leq_{\mathbf{p}} \mathbf{P}$

(b) This argument is **false.** The reason is that the transformation can be done in the length of **v** NOT IN u. Since KNAP is NP-complete, the length of v is also not in polynomial
(c)