Solution Assignment 4

Maxime Descoteaux

April 16, 2004

Question 1 (15 pts)

(10 pts)(a) Reduce 3-SAT to VERTEX COVER (VC)

We construct the graph in the following way:

1. For each literal x_i in each clause j, create a vertex t_{ij} and connect them with an edge. i.e. we get a triangle-like subgraphs for every clause.

2. For every literal x_i, create a vertex v_i and $\sim$v$_i$ with an edge between them.

3. For each vertex t_{ij}, create an edge going to its corresponding v_i literal. Note that t_{ij} and v_i are two different vertex which represent the same literal. The v_i vertices selected in the vertex cover constitute the truth assignment to 3-SAT.

4. Note that if N is the number of literals and M the number of clause, we have a total of $2N + 3M$ vertices and $N + 3M$ edges in the graph.

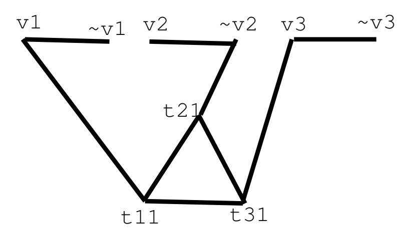

Figure 1: Example reduction of the 3-SAT clause $(x_1 + \sim x_2 + x_3)$

Let the graph constructed in this transformation be G_{3-SAT}. Figure 1 is an example for a particular clause $(x_1 + \sim x_2 + x_3)$. Note also that this transformation can be done in polynomial time.

Claim: 3-SAT is satisfied iff G_{3-SAT} has a vertex cover of size $N + 2M$

intuition: We know we need two vertices to cover a triangle and one vertex for each pair v_i, $\sim$v$_i$ => exactly$N + 2M$ needed to cover the graph. Hence, with this construction, we cannot have both v_i and $\sim$v$_i$ in the vertex cover.

Must show that if 3-SAT is satisfied then we have a vertex cover of size $N + 2M$ and also show that if there is a VC of size $N + 2M$ in G_{3-SAT} then it satisfies 3-SAT.

Proof:

=>

If 3-SAT is satisfied, we have a truth assignment. We know that either x_i = TRUE or $\sim$x$_i$ = TRUE. So, if x_i = TRUE we add v_i to the vertex cover VC and add $\sim$v$_i$ otherwise. Also, since each clause j has to return TRUE, we find the first TRUE literal in the clause. Let the other two literals in the clause be, x_p and x_q. We add the associated vertices t_{pj} and t_{qj} to VC.

Is VC a vertex cover of G_{3-SAT}? Yes! Each triangle subgraph is covered by exactly two vertex and each pair v_i, $\sim$v$_i$ is also covered by exactly one vertex selected from the truth assignment. Hence, VC is a vertex cover of size $N + 2M$.

<=

We go with the contrapositive. i.e if 3-SAT is NOT satisfied I claim there cannot be a VC of size N + 2M in G_{3-SAT}. This is because one of the triangle subgraph cannot be covered with only two vertex.

Why? If 3-SAT is not satisfied, there exist a clause j where each literal is false. WLOG, let the three FALSE literals be x_a, x_b, x_c. In order to cover all three v_i, $\sim$v$_i$ pairs with the least number of vertices possible, we need to have $\sim$v$_a$, $\sim$v$_b$, $\sim$v$_c$ in the vertex cover. Otherwise, the 3-SAT problem would be satisfied. Now, I claim we cannot cover the triangle subgraph part with only two vertex. No matter how you pick two vertex in the triangle (say t_{aj} and t_{bj}) the other edge between the t vertex and v vertex ($t_{cj}v_c$) will NOT be covered. This is because neither t_{cj} nor v_c are in the vertex cover. You would need all three vertex of the triangle to have a valid vertex cover. Therefore, there cannot be a vertex cover of size $N + 2M$.

Illustrate the G_{3-SAT} graph construction for $(\bar{x} + y + \bar{z}) \cdot (x + \bar{y} + z) \cdot (\bar{x} + \bar{y} + z) \cdot (\bar{x} + y + z)$

If we let $x = x_1, y = x_2, z = x_3$ to agree with our notation, a valid truth assignment is $x_1 = x_2 = x_3 =$ TRUE.

1. We ADD v_1, v_2, v_3 in the vertex cover since x_1, x_2, x_3 are TRUE.
2. We go to close 1 (t_{i1} triangle) and pick first TRUE literal. This is y which corresponds to x_2 which coresponds to t_{21}, in our notation.
3. ADD other two literal, t_{11} and t_{31}, to vertex cover
4. Repeat step 2. and 3. for every clause. The resulting vertex cover is the set of all circled vertices. The vertex cover $\{v_1, v_2, v_3, t_{11}, t_{31}, t_{22}, t_{32}, t_{13}, t_{23}, t_{14}, t_{34}\}$ has size

$$N + 2M = 3 + 2 * 4 = 11$$

as expected.

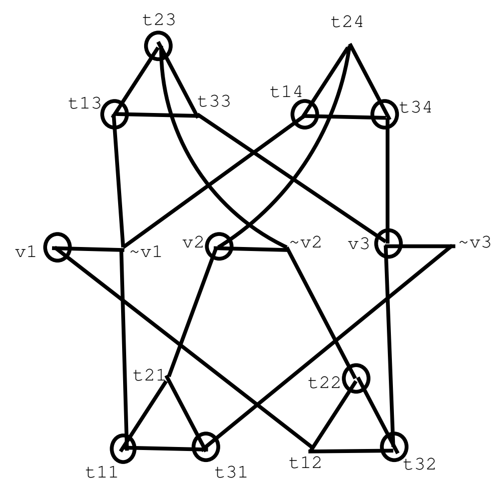

Figure 2: Transformation graph for 3-SAT to VC reduction

(5 pts)(b) illustrate the reduction from 3-SAT to SS

We construct the table corresponding to

$$(\bar{x} + y + \bar{z}) \cdot (x + \bar{y} + z) \cdot (\bar{x} + \bar{y} + z) \cdot (\bar{x} + y + z)$$

	v_1	v_1'	v_2	v_2'	v_3	v_3'	s_1	s_1'	s_2	s_2'	s_3	s_3'	s_4	s_4'	t
x	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1
y	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1
z	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1
c_1	0	1	1	0	0	1	1	2	0	0	0	0	0	0	4
c_2	1	0	0	1	1	0	0	0	1	2	0	0	0	0	4
c_3	0	1	0	1	1	0	0	0	0	0	1	2	0	0	4
c_4	0	1	1	0	1	0	0	0	0	0	0	0	1	2	4

Subset Sum Instance:
$S = \{1000100, 1001011, 101001, 100110, 10111, 11000, 1000, 2000, 100, 200, 10, 20, 1, 2\}$
$t = 1114444$

1. Truth assignment: x = TRUE, y = TRUE, z = TRUE. Easy to check that this satisfies 3-SAT.

   Corresponding Subset Sum: (need columns $v_1, v_2, v_3, s_1, s_1', s_2', s_3, s_3', s_4'$)
   $S' = \{1000100, 101001, 10111, 1000, 2000, 200, 10, 20, 2\}$
   $\text{sum}(S') = 1114444$

2. Truth assignment: x = FALSE, y = FALSE, z = FALSE. Easy to check that this satisfies 3-SAT.

   Corresponding Subset Sum: (need columns $v_1', v_2', v_3, s_1, s_1', s_2', s_3, s_4$
   $S'' = \{1001011, 100110, 10111, 1000, 2000, 200, 10, 2\}$
   $\text{sum}(S'') = 1114444$

Question 2: Prove that 3-PARTITION is NP-complete (15 pts)

3-PARTITION is in NP (5 pts)

- 3-PARTITION is a decision problem since the output is either "yes" or "no" if the instance can be partionned or not in three disjoint subsets of equal weight.
- Can the "yes" answer be verified with a certificate in poly-time? Yes!

  certificate: subsets A_1, A_2, A_3 partionning the instance $\{a_1, ..., a_n\}$ into three disjoint subsets of sum N

  1. check that $A_1 \cup A_2 \cup A_3 = \{a_1, ..., a_n\}$. Done in $O(n)$.
  2. check that $A_1 \cap A_2 \cap A_3 = \emptyset$. Done in $O(n^2)$ or better $O(n\text{log}n)$.
  3. check that $\text{sum}(A_1) = \text{sum}(A_2) = \text{sum}(A_3) = N$. Done in $O(n)$.

  Therefore, 3-PARTITION is in NP.

Can we reduce a known NP-complete problem to 3-PARTITION? (10 pts)

We transform Subset Sum (SS) to 3-PARTITION.

SS $\leq$ 3-PARTITION (5 pts)

Given instance of subset sum $A = \{a_1, \ldots, a_n\}$ and integer k. Suppose $t = \text{sum}(A) = a_1 + \cdots + a_n$ and assume for simplicity that $t \geq k$ (if not, I leave it to you to work out the little modifications). We have two case to consider:
case I: Let $a_{n+1} = t - 2k$ and $a_{n+2} = t - k$. There are both non-negative as $t > 2k \implies t - 2k \geq 0 \implies t - k > 0$. case II: Let $a_{n+1} = 2k - t$ and $a_{n+2} = k$. There are both non-negative as $t \leq 2k \implies 2k - t > 0$.

Let the new instance for 3-PARTITION be

$$S = \{a_1, \ldots, a_n, a_{n+1}, a_{n+2}\} \qquad N = a_{n+2}$$

In any case, this is a valid instance of 3-PARTITION because S is a non-negative set of integers. I claim that $\exists\ A' \subseteq A$ with $\text{sum}(A') = k \iff \exists$ a 3-PARTITION A_1, A_2, A_3 of S with $\text{sum}(A_1) = \text{sum}(A_2) = \text{sum}(A_3) = N$.

Proof: (5pts)

(=>) Suppose we have $A' \subseteq A$ with sum$(A') = k$. Let the complement of A' be $\bar{A}'$. Thus, sum$(A') = k$, and sum$(\bar{A}') = t - k$.
case I:

$$A_1 = A' \cup \{a_{n+1}\} \qquad A_2 = \bar{A}' \qquad A_3 = \{a_{n+2}\}$$

is a 3-PARTITION. It satisfies conditions 1, 2, 3 above. In particular, sum$(A_1) = k + (t - 2k) =$ sum(A_2) $= t - k =$ sum$(A_3) = t - k = a_{n+2}$.

case II:

$$A_1 = A' \qquad A_2 = \bar{A}' \cup \{a_{n+1}\} \qquad A_3 = \{a_{n+2}\}$$

is a 3-PARTITION. It satisfies conditions 1, 2, 3 above. In particular, sum$(A_1) = k =$ sum$(A_2) =$ $(t - k) + (2k - t) =$ sum$(A_3) = k = a_{n+2}$.

Therefore, in any case, 3-PARTITION is satisfied.

(<=) We go with the contrapositive. Suppose we have NO subset of A with sum k. Now, for a contradiction, suppose we have 3-PARTITION A_1, A_2, A_3 of S with equal weight a_{n+2}. In any case, as S is a set of non-negative integers, a_{n+2}, must be by itself in a subset (assuming none of $\{a_1, \ldots, a_n, a_{n+1}\}$ are zero). Thus, we need two subsets $A', A'' \subseteq \{a_1, \ldots, a_n, a_{n+1}\}$ with equal weight a_{n+2}. Note that either $a_{n+1} \in A'$ or $a_{n+1} \in A''$. Hence, in one case, we must have a subset of A with sum $t - k$ and in the other case a subset of A with sum k. In any case, there exists a subset of $A = \{a_1, \ldots, a_n\}$ with sum$(A) = k$. CONTRADICTION! Therefore, if we have no subset of A with sum k then we have no 3-PARTITION of S with equal sum N.

Question 3 (10 pts)

Formulate 3-SAT as ILP

- For every literal x_i. we introduce an integer variable $y_i \in \mathbf{Z}$ and for every literal $\bar{x}_i$,we introduce an integer variable $\bar{y}_i \in \mathbf{Z}$

$$y_i = \begin{cases} 1 & \text{if} x_i = \text{TRUE} \\ 0 & \text{otherwise} \end{cases}$$

$$\bar{y}_i = \begin{cases} 1 & \text{if} \bar{x}_i = \text{TRUE} \\ 0 & \text{otherwise} \end{cases}$$

- For every clause, $(x_i + x_j + x_k)$, we introduce a constraint

$$y_i + y_j + y_k \geq 1$$

  This ensures that every clause is satisfied.

- Introduce a constraint that makes sure that each variable y_i is either 0 or 1.

$$0 \leq y_i \leq 1 \qquad 0 \leq \bar{y}_i \leq 1$$

- We must make sure that y_i and $\bar{y}_i$ are not both zero or not both 1.

$$y_i + \bar{y}_i = 1$$

- We have a dummy objective function

$$\max \sum_{\forall i} y_i$$

Formulate VERTEX COVER as ILP

Given $G = (V, E)$ and integer k

- $\forall v_i \in V$ introduce an integer variable $x_i \in \mathbf{Z}$

$$x_i = \begin{cases} 1 & \text{if} \, v_i \in \text{vertex cover} \\ 0 & \text{otherwise} \end{cases}$$

- $\forall e_{ij} \in E$ (e_{ij} is the edge connecting vertex $v_i, v_j \in V$), we introduce a constraint

$$x_i + x_j \geq 1$$

  This enforces that for every edge, we select at least one the the two endpoint vertex.

- $\sum_{\forall i} x_i = k$, enforces the vertex cover to be of size k.
- $0 \leq x_i \leq 1$
- Ojective function: max $\sum_{\forall i} x_i$

Formulate SUBSET SUM as ILP

We are given a set of non-negative integers $A = \{a_i, \ldots, a_n\}$ and integer k

- for every a_i, we introduce an integer variable $x_i \in \mathbf{Z}$.

$$x_i = \begin{cases} 1 & \text{if} \, a_i \in A' \subseteq A \, \text{with sum}(A') = k \\ 0 & \text{otherwise} \end{cases}$$

- $\sum_{\forall i} a_i x_i = k$
- $x_i \geq 0$
- Objective function: max $\sum_{\forall i} x_i$

Using lp_solve on question #1 example

It is simply a matter of creating input files with valid syntax and et voila! The files are called 3-SAT.lp, VC.lp and SS.lp. You can run them to get my output.
IMPORTANT: equality constraints must be put into standard linear program inequality form. i.e.

$$x_1 + \cdots + x_n = t \quad \text{becomes} \quad x_1 + \cdots + x_n \leq t \quad \text{AND} \quad -x_1 - \cdots - x_n \leq -t$$

Results:
3-SAT: x = 1, y = 1, z = 1, $\bar{x} = 0$, $\bar{y} = 0$, $\bar{z} = 0$

Vertex Cover: You can verify that it gives a valid vertex cover. The output is $VC = \{\sim\text{v}_1, \sim\text{v}_2, \sim\text{v}_3, t_{13}, t_{14}, t_{21}, t_{22}, t_{23}, t$

Subset Sum: $S = \{1000100, 101001, 10111, 1000, 2000, 200, 10, 20, 2\}$
sum(S) = 1114444.