Problem Set 1
Algorithms Advanced Course TDA250

Per Lindstrand

7th November 2005

1.1 Introduction

Problem Set 1 by Per Lindstrand (muusu@dtek.chalmers.se), 820921-5576.

1.2 [NP] Problem 1.1 [KT 8.9.6]

Definition M-SAT is the problem of minimal truth-assignment for monotone SAT. Given a monotone instance of SAT ϕ and a number k, is there an assignment ρ where at most k variables are set to 1 that satisfies ϕ?

Proposition 1.2.1 (M-SAT $\in$ NP) *There exists a polynomial time certifier for a solution ρ to an instance of (ϕ, k) M-SAT.*

Proof Given a solution ρ it is "trivial" to verify that ϕ is satisfied and that ρ contains at most k variables that are set to 1. Construct an assignment τ where all variables are set to 0 except those that are set to 1 in ρ. Verify that at most k variables are assigned to 1 then evaluate ϕ for assignment τ. ∎

Proposition 1.2.2 (M-SAT is NP-complete) *Given a monotone instance of SAT ϕ, together with a number k, the problem of Monotone Satisfiability with Few True Variables (M-SAT) is NP-complete.*

Consider the reduction SET-COVER $\leq_P$ M-SAT. Given a black box to solve M-SAT and an instance of SET-COVER with n subsets $S_1 \cdots S_n \in U$ and a number k, is it possible to pick at most k subsets S_i to cover U?

The problem of SET-COVER is to pick a minimal number of subsets of U to cover a set U. Similarly, the problem of M-SAT is to pick a minimal truth-assignment of variables to cover a set of clauses.

Let all subsets $S_1 \cdots S_n$ correspond to boolean variables $x_1 \cdots x_n$ and all elements $a_i \in U$ correspond to clauses C_i such that

$$C_i = \bigvee x_j \qquad \forall j \text{ such that } a_i \in S_j \tag{1.1}$$

That is, element a_i is covered by subset S_j which corresponds to clause C_i being "covered" by setting variable x_j to 1.

Proof ($\rightarrow$) If at most k subsets are needed to cover U then these contain all elements in U. Then there must exist a truth-assignment of at most k variables that satisfies all clauses. As each element $a_i \in U$ corresponds to a clause C_i and k subsets S_j where $j \in J$ contains all elements a_i then by setting the variables x_j where $j \in J$ to 1 all clauses C_i are satisfied.

Suppose that there does not exist a truth-assignment of at most k variables. This implies that there exists a clause C_i that is not satisfied by setting variables x_j to 1, that is, there exists an element a_i that is not contained in a subset S_j. This is a contradiction.

Proof ($\leftarrow$) Conversely, if there exists a truth-assignment with at most k variables x_j where $j \in J$ then all elements $a_i \in U$ are covered by picking subsets S_j where $j \in J$. ∎

1.3 [NP] Problem 1.2

We divide non-trivial reduction CLIQUE $\leq_P$ 3-SAT into two reductions; CLIQUE $\leq_P$ SAT and SAT $\leq_P$ 3-SAT. The latter being subject for reference perhaps.

The idea is to perform a colouring of vertices in the graph $G = (V, E)$. This colouring will describe relations between vertices and enforce restrictions that are an essential part of the construction.

Each vertex will be coloured uniquely, it will be part of its own colour only. If we "pick" a vertex, we must colour it, corollary, if don't "pick" a vertex, it doesn't matter. The plan is to see if we can pick k vertices with k different colours.

Let variables x_i correspond to vertices v_i, if a vertex is "picked" then $x_i = 1$. Let variables x_{ij} denote that vertex v_i has colour j.

Every picked vertex v_i must have a colour.

$$C^p = \bigwedge_{i=1}^{n} \left[x_i \wedge \left(\bigvee_{l=1}^{k} x_{il} \right) \vee \overline{x_i} \wedge \overline{\left(\bigvee_{l=1}^{k} x_{il} \right)} \right] \tag{1.2}$$

A picked vertex v_i must only have one colour.

$$C^{p'} = \bigwedge_{i=1}^{n} \bigwedge_{l=1}^{k} \left(\overline{x_{il}} \vee x_{il} \wedge \overline{\left(\bigvee_{l'=1, l' \neq l}^{k} x_{il'} \right)} \right) \tag{1.3}$$

No nodes connected by an edge $e \in E$ may have the same colour.

$$C^e = \bigwedge_{l=1}^{k} \overline{(x_{il} \wedge x_{jl})} \qquad \forall e = (v_i, v_j) \in E \tag{1.4}$$

All colours $1 \ldots k$ must be present.

$$C^c = \bigwedge_{l=1}^{k} \left(\bigvee_{i=1}^{n} x_{il} \right) \tag{1.5}$$

If two vertices i and j do not have an edge $e \in E$ such that $v_i \in e$ and $v_j \in e$ both may not be picked at the same time. We define E' to be all edges not in E, that is, the complement edge set.

$$C^v = \bigwedge_{i=1}^{n} \left[x_i \wedge \overline{\left(\bigvee_{j | e \in E', v_i \in e, v_j \in e} x_j \right)} \vee \overline{x_i} \right] \tag{1.6}$$

We need to transform (1.6) into CNF (Conjunctive Normal Form), by expanding the expression in the construction it is possible to transform it into CNF. Using this construction the CLIQUE problem has been transformed into a SAT problem. The second reduction is from SAT to 3-SAT. Given an instance of SAT and a black box for solving 3-SAT we need to show a polynomial time reduction from SAT to 3-SAT.

Let the instance of SAT be described by the set of clauses $C_1 \cdots C_n$ over the variables $x_1 \cdots x_m$. We will construct a collection of clauses C'_i over the variables $x_1 \cdots x_m$ and some additional variables $y_{i,1} \cdots y_{i,k}$ for each clause $1 \leq i \leq n$. Replace all clauses C_i by a set of three literal clauses C'_i using the variables from the clause C_i and some additional variables depending on the size of C_i.

Let a clause C_i be given by $z_1 \vee \cdots \vee z_k$ where z_j are literals in C_i. For a clause of size k

(i) $k = 1 \Rightarrow C_i = z_1$. Use two variables $y_{i,1}$ and $y_{i,2}$ and define $C'_i = (z_1 \vee y_{i,1} \vee y_{i,2}) \wedge (z_1 \vee y_{i,1} \vee \overline{y_{i,2}}) \wedge (z_1 \vee \overline{y_{i,1}} \vee y_{i,2}) \wedge (z_1 \vee \overline{y_{i,1}} \vee \overline{y_{i,2}})$.

(ii) $k = 2 \Rightarrow C_i = z_1 \vee z_2$. Use one variable $y_{i,1}$ and define $C'_i = (z_1 \vee z_2 \vee y_{i,1}) \wedge (z_1 \vee z_2 \vee \overline{y_{i,1}})$.

(iii) $k = 3 \Rightarrow C_i = z_1 \vee z_2 \vee z_3$. No additional variables are needed and we define $C'_i = C_i$.

(iv) $k > 3 \Rightarrow C_i = z_1 \vee \cdots \vee z_k$. Use variables $y_{i,1} \cdots y_{i,k-3}$ and define $C'_i = (z_1 \vee z_2 \vee y_{i,1}) \wedge (\overline{y_{i,1}} \vee z_3 \vee y_{i,2}) \wedge (\overline{y_{i,2}} \vee z_4 \vee y_{i,3}) \wedge \cdots \wedge (\overline{y_{i,k-3}} \vee z_{k-1} \vee z_k)$.

which concludes the construction. Below follows a short proof.

Proof ($\leftarrow$) Suppose that the constructed expression ϕ is satisfiable. From (1.3) and (1.4) we know that all vertices in the solution have a unique colour because they are connected by an edge. From (1.5) we know that all colours are present in the solution and from (1.6) we know that no two vertices without an edge can be picked at the same time. We can now infer that the solution has k vertices with k different colours that are fully connected. ∎

Proof ($\rightarrow$) Suppose we have a solution V' to an instance of CLIQUE with $G = (V, E)$ and a number k. We know that V' contains k vertices that are fully connected.

For any set of vertices (1.3) can be satisfied by picking a colour. As there are k (picked) vertices and we have k unique colours we can satisfy (1.4). Since all vertices are fully connected we must use k colours and hence satisfy (1.5). Since all vertices in the solution are fully connected there are no two vertices that are not connected, hence (1.6) is satisfied. ∎

1.4 [A] Problem 1.3 [KT 11.9.1]

1.4.1 Part a

Proposition 1.4.1 *The greedy algorithm does not use the minimum possible number of trucks for all sets of weights $w_1 \cdots w_n$ and number K.*

Proof Suppose $n > 2$ weights are given by the set $W = \{1, K, 1, K \cdots\}$ for any $K > 1$. The greedy algorithm will use n trucks. We need $\frac{n}{2}$ trucks for all weights of K and $\frac{n}{2}\frac{1}{K}$ trucks for all weights of 1. That is,

$$\frac{1}{K}\left(\frac{n}{2} \cdot 1 + \frac{n}{2} \cdot K\right) = \frac{n}{2}\left(\frac{1}{K} + 1\right) \tag{1.7}$$

and $n < \frac{n}{2}(\frac{1}{K} + 1)$ if $K > 1$. ∎

Consider the instance $K = 2$ and $W = \{1, 2, 1, 2\}$. The greedy algorithm would load 1 onto the first truck. The second weight is too heavy for the first truck and the greedy algorithm would send it away and load 2 onto the next truck. This truck is now fully loaded and must be sent off. This continues with the third and fourth truck as well.

The optimal number of trucks is three. By loading the two weights of 1 onto a single truck and loading the two weights of 2 onto two other trucks we achieve the optimal number of trucks, three.

1.4.2 Part b

Proposition 1.4.2 *An optimal solution* $T^* \geq \frac{1}{K}\sum_{i=1}^{n} w_i$.

Proof Suppose that we have a solution $T' < T^*$, that is $T' < \frac{1}{K}\sum_i w_i$ then $KT' < \sum_i w_i$ but the total weight that can be loaded with T' trucks is KT' which means that one or more trucks must load a weight $K' > K$. This is a contradiction. ∎

Proposition 1.4.3 *The number of trucks that the greedy algorithm uses is within a factor two of the minimum possible number, for any set of weights and any value of* K. *That is,* $T \leq 2T^*$.

Proof Let S_i denote the set of weights that truck i loads. By analyzing the greedy algorithm we can conclude that two sequential trucks loads S_i and S_{i-1} respectively and that the following must hold for any i:

$$|S_i| + |S_{i-1}| = \sum_{w_j \in S_i} w_j + \sum_{w_k \in S_{i-1}} w_k > K \tag{1.8}$$

which means that a truck i loads $l_i > \frac{K}{2}$ on average. Furthermore, we have that

$$\sum_{k=1}^{n} w_k = \sum_{i=1}^{T} \sum_{w_j \in S_i} w_j = \sum_{i=1}^{T} |S_i| \tag{1.9}$$

If we have an even number of trucks $T = 2T'$ then

$$\sum_{i=1}^{T} |S_i| = \sum_{i=1}^{T'} (|S_{2i}| + |S_{2i-1}|) > \sum_{k=1}^{T'} K = \frac{1}{2}TK \tag{1.10}$$

and for an odd number of trucks $T = 2T' + 1$ then

$$\sum_{i=1}^{T} |S_i| = |S_T| + \sum_{i=1}^{T'} (|S_{2i}| + |S_{2i-1}|) > \sum_{k=1}^{T'} K = \frac{1}{2}(T-1)K \tag{1.11}$$

From (1.10) and (1.11) we can infer that for any solution T

$$\sum_{i=1}^{T} |S_i| > \frac{1}{2}(T-1)K \tag{1.12}$$

since, clearly, $\frac{1}{2}(T-1)K < \frac{1}{2}TK$. Using (1.9)

$$\frac{1}{2}(T-1)K < \sum_{k=1}^{n} w_k \Rightarrow T \leq 2\frac{1}{K}\sum_{i=1}^{n} w_i \tag{1.13}$$

but $T^* \geq \frac{1}{K}\sum_i^n w_i$ then $T \leq 2T^*$. ∎

1.5 [A] Problem 1.4 [KT 11.9.3]

1.5.1 Part a

Proposition 1.5.1 *UNWEIGHTED-HITTING-SET ∈ NP.*

Proof Given a solution H to an instance of UNWEIGHTED-HITTING-SET with $A = \{a_1 \cdots a_n\}$, $B_1 \cdots B_m \in A$, $w_1 \cdots w_n$ where $w_i = 1$ for all $1 \leq i \leq n$ and a number w we can certify it in polynomial time. We check that

$$\sum_{a_i \in H} w_i \leq w \tag{1.14}$$

and

$$H \cap B_i \neq \emptyset \quad \forall B_i \tag{1.15}$$

and answer "yes" iff both statements are true and "no" otherwise. ∎

Proposition 1.5.2 *UNWEIGHTED-HITTING-SET is NP-complete.*

Consider the reduction M-SAT $\leq_P$ UNWEIGHTED-HITTING-SET. Given a black box to solve UNWEIGHTED-HITTING-SET and an instance ϕ of M-SAT with variables $x_1 \cdots x_n$, clauses $C_1 \cdots C_m$ and a number k.

Let variables $x_1 \cdots x_n$ correspond to elements $a_1 \cdots a_n \in A$. Let clauses $C_1 \cdots C_m$ correspond to subsets $B_1 \cdots B_m \in A$ such that

$$B_i = \{a_j | x_j \in C_i\} \tag{1.16}$$

Solve UNWEIGHTED-HITTING-SET with $A = \{a_1 \cdots a_n\}$, $B_1 \cdots B_m$ and k using the black box. If the answer is "yes" then there is a satisfying assignment of size at most k variables x_i set to 1.

Proof ($\rightarrow$) Suppose there is a satisfying assignment ρ of size k for ϕ. Since this is a satisfying assignment we have satisfied all clauses $C_1 \cdots C_m$ by applying assignment ρ. From the construction we have that if a variable $x_i \in C_j$ then $a_i \in B_j$. Consider all elements $a_i \in A$ corresponding to variables $x_i \in \rho$. This is a *hitting set* of size k since by applying ρ we satisfy all clauses C_j and therefore cover all subsets B_j. ∎

Proof ($\leftarrow$) Conversely, suppose we have a minimal hitting set H of size k. Since this is a minimal hitting set, for each subset B_j, there is an element a_i such that $a_i \in H$ and $a_i \in B_j$. From the construction we have that for each variable $a_i \in H$ there is a variable x_i. Since we have picked at least one element a_i from every subset B_j then we have at least one variable x_i from every clause C_j. Let ρ be an assignment such that $\rho = \{x_i = 1 | a_i \in H\}$, then ρ is a satisfying assignment of size k for ϕ. ∎

1.5.2 Part b

A decision version of the weighted problem would be "given a set $A = \{a_1 \cdots a_n\}$, a set of subsets $B_1 \cdots B_m \in A$, a set of weights $w_1 \cdots w_n$ where all $w_i \geq 0$ and a number w, is there a hitting set H such that $H \cap B_i \neq \emptyset$ for all B_i and $\sum_{a_i \in H} w_i \leq w$.

Proposition 1.5.3 *WEIGHTED-HITTING-SET ∈ NP.*

Proof Given a solution H to an instance of WEIGHTED-HITTING-SET with $A = \{a_1 \cdots a_n\}$, $B_1 \cdots B_m \in A$, $w_1 \cdots w_n$ where $w_i \geq 0$ for all $1 \leq i \leq n$ and a number w we can certify it in polynomial time. We check that

$$\sum_{a_i \in H} w_i \leq w \tag{1.17}$$

and

$$H \cap B_i \neq \emptyset \quad \forall B_i \tag{1.18}$$

and answer "yes" iff both statements are true and "no" otherwise. ∎

Proposition 1.5.4 *WEIGHTED-HITTING-SET is NP-complete.*

Consider the reduction UNWEIGHTED-HITTING-SET $\leq_P$ WEIGHTED-HITTING-SET. Given a black box to solve WEIGHTED-HITTING-SET and an instance of UNWEIGHTED-HITTING-SET with $A = \{a_1 \cdots a_n\}$, a set of subsets $B_1 \cdots B_m \in A$, a set of weights $w_1 \cdots w_n$ where all $w_i = 1$ and a number w.

There is no construction, the problem can be directly translated.

Proof Since UNWEIGHTED-HITTING-SET is a special case of WEIGHTED-HITTING-SET the solution to the weighted problem will be the solution to the unweighted problem. ∎

1.5.3 Part c

Proposition 1.5.5 *An optimal solution* $w^* \geq \max\{\min\{w_i | a_i \in B_j\} | B_j, 1 \leq j \leq m\}$. *(The maximum of all minimum elements in all subsets.)*

Proof (Proof by contradiction.) Suppose $w^* < \max\{\min\{w_i | a_i \in B_j\} | B_j, 1 \leq j \leq m\}$ then the optimal solution would contain elements from all subsets but have a total weight less than the smallest element in a subset. Since there are only larger elements in that subset and we know that $H \cap B_j \neq \emptyset$ for all $j = 1 \cdots m$ we must have chosen a different element from that subset. But this element has weight larger than or equal to the smallest in that subset. This is a contradiction.

A greedy rule for this problem is to minimize the negative impact of picking an element a_i in each step. We define negative impact as (i) weighing and (ii) being in few subsets. That is, we want to minimize $w_i / |\{a_i | a_i \in B_j\}|$ which of, at least numerically, assumes that all elements are present in at least one subset. By defining a division by zero as infinity the equation is consistent.

We pick the elements according to this greedy rule in each step. Also, in each step, all subsets containing the picked element a_i are marked as "hit". This is done until all subsets B_j have been "hit"

Algorithm Greedy-Approximate-Weighted-Hitting-Set

Let `H` = $\emptyset$
Let `V` = $\emptyset$
for $a_i \in A$ **do**
  $v_i = w_i / |\{a_i | a_i \in B_j\}|$

$\mathtt{V} \leftarrow (i, v_i)$
end for
Sort V in ascending order with respect to v_i (if $v_i = v_j$, favour element with largest $|\{a_i|a_i \in B_j\}|$)
while All B_j not marked as hit **do**
Pick smallest element $(i, v_i) \in \mathtt{V}$ such that $a_i \in B_j$ for any B_j not marked as hit and remove it from V
$\mathtt{H} \leftarrow a_i$
Mark all B_j such that $a_i \in B_j$ as hit
end while
Return H

Proposition 1.5.6 *The greedy algorithm produces a hitting set H with weight $w \leq bw^*$ where $b = \max\{|B_j||B_j, 1 \leq j \leq m\}$.*

Suppose the worst case for the greedy rule in the algorithm, two subsets $B_1 = \{1, \ldots, n\}$ and $B_2 = \{n+1\}$. The greedy rule would pick all elements in B_1 first since $\frac{1}{1} \leq \frac{2}{1} \leq \cdots \leq \frac{n+1}{1}$ and then the element in B_2. The optimal solution would be, according to the lower bound, $w^* \geq n+1$ (specifically $n+2$.) The solution produced by the greedy algorithm would contain all elements in B_1 and the first element of B_2, that is

$$w = \sum_{a_i \in B_1} w_i + (w_{n+1}) = \sum_{i=1}^{n} i + (n+1) = \frac{n(n+1)}{2} + (n+1) = \frac{(n+2)(n+1)}{2} \tag{1.19}$$

and since $b = \max\{|B_j||B_j, 1 \leq j \leq m\} = n$ we have that

$$w \leq bw^* \Rightarrow \frac{(n+2)(n+1)}{2} \leq n(n+2) \Rightarrow \frac{(n+1)}{2} \leq n \Rightarrow n \leq 2n-1 \tag{1.20}$$

which is true for $n \geq 1$.

Proof Consider each step of the greedy algorithm and the optimal solution

$$w^* \leq w \Rightarrow w^* + w_i^* \leq w + w_i' \tag{1.21}$$

where

$$w_i' = \min\{w_i|w_i/|\{a_i|a_i \in B_j\}|, 1 \leq j \leq m\} \tag{1.22}$$

and since the greedy algorithm picked w_i' and not w_i^* we know that

$$\frac{w_i'}{|\{a_i|a_i \in B_k\}|} \leq \frac{w_i^*}{|\{a_i|a_i \in B_{k'}\}|} \tag{1.23}$$

therefore

$$w_i' \leq \frac{|\{a_i|a_i \in B_k\}|}{|\{a_i|a_i \in B_{k'}\}|} w_i^* \tag{1.24}$$

and since $|\{a_i|a_i \in B_k\}| \leq b$ for all k we know that in the worst case $|\{a_i|a_i \in B_k\}| = b$ and $|\{a_i|a_i \in B_{k'}\}| = 1$

$$w_i' \leq bw_i^* \tag{1.25}$$

which holds for all picked weights w_i' and w_i^*, therefore

$$\sum_i w_i' \leq \sum_i bw_i^* \tag{1.26}$$

where $w = \sum_i w'_i$ and $w^* = \sum_i w^*_i$, that is

$$w \leq bw^* \tag{1.27}$$

for any solution of the greedy algorithm. ∎