Course 02249 Computationally Hard Problems
Fall 2013, DTU Compute

Solution to assignment PROJECT

Students' names: Andreas Hallberg Kjeldsen and Morten Chabert Eskesen

Study nr. s092638 and s133304

Date. 04.11.2013

Not quite adequate	Adequate	Good

Problem: [MirrorFriendlyMinimumSpanningTree (MFMST)]
Input: An undirected, connected weighted graph $G = (V, E, w)$, where $V = \{1, \ldots, n\}$, $E = \{e_1, \ldots, e_m\}$ and $w : E \to \mathbb{N}_0$, and a number $B \in \mathbb{N}$.
Output: YES if there is a spanning tree $T \subseteq E$ for G such that

$$max \left\{ \sum_{e_i \in T} w(e_i), \sum_{e_i \in T} w(e_{m+1-i}) \right\} \leq B$$

and NO otherwise.

a) Description of the problem in colloquial terms

A minimum spanning tree is a subgraph within a undirected, connected weighted graph that is a tree and connects all the vertices together with a weight less or equal to the weight of every other spanning tree. The main difference between a minimum spanning tree and a mirror friendly minimum spanning tree is the inequality described above. In a mirror friendly minimum spanning tree the inequality must be satisfied. It should be possible to mirror the spanning tree in such a way that the maximum of the spanning tree and the mirrored spanning tree is less than or equal to a fixed value, B. This also means that the mirror friendly minimum spanning tree may not be equal to the minimum spanning tree in the graph, i.e. it may have a larger weight than the minimum spanning tree.

Solve an example problem

Input: $V = \{1, 2, 3\}$, $E = \{e_1 = \{1, 2\}, e_2 = \{2, 3\}, e_3 = \{1, 3\}\}$, $w(e_i) = i$ for $i \in \{1, 2, 3\}$ and $B = 4$.

Spanning Tree	Mirrored Spanning Tree
$e_1 + e_2 = 3$	$e_{3+1-1} + e_{3+1-2} = 5$
$e_3 + e_1 = 4$	$e_{3+1-3} + e_{3+1-1} = 4$

$$max \{e_3 + e_1, e_{3+1-3} + e_{3+1-1}\} \leq 4$$

$$max \{4, 4\} \leq 4$$

Output would be a spanning tree consisting of the edges: e_3 and e_1.

b) Show that MFMST is in NP

1. Design a deterministic algorithm A which takes as input a problem instance X and random sequence R

Specify what the random sequence R consists of
Let the string R consist of bits: $R = r_1, r_2, \dots, r_n$.

Specify how A interprets R as a guess
Consider the first m bits. If the i-th bit is 1, mark the edge e_i.

Specify how A verifies the guess
If the marked edges create a mirror friendly minimum spanning tree with a weight less than or equal to B, answer YES, otherwise NO.

2. Show that the two conditions are met

If the answer to X is YES, then there is a string R^* with positive probability such that $A(X, R^*) = YES$

> Asssume that the answer is YES
>
> Then there is a subset of the edges that creates a mirror friendly minimum spanning tree with a weight less than or equal to B.
>
> Let $S \subseteq \{1, \dots, m\}$ be the set that describe the edges' index
>
> Construct the bit string $R^* = r_1, r_2, \dots, r_m$ where $r_i = 1$ if and if only if $i \in S$
>
> When A receives R^*, it will select the edges in S, check that the weight of the edges are less than or equal to B and answer YES.
>
> Altogether there is a string of length at most $p(n)$ that will give YES. The probability of randomly creating it is positive.

If the answer to X is NO, then $A(X, R) = NO$ for all R

> Assume that the answer is NO
>
> Then no set of the edges create a mirror friendly minimum spanning tree with a weight less than or equal to B.
>
> If R does not contain enough bits, the algorithm will correctly answer NO.

Otherwise the algorithm will mark some edges and compute their weight. This will be compared to B. But as no set of edges has a weight less than or equal to B, the answer is NO.

3. Show that A is p-bounded for some polynomial p

There are m edges.

It is checked that the string R consists of at least m bits. Time: $O(m)$.

Every edge is marked or not marked. Time: $O(m)$.

The weights of the marked edges are added. Time: $O(m)$.

The computed total weight is compared to B and the answer is returned. Time: $O(1)$.

Altogether the time is: $O(m)$.

c) Show that MFMST is NP-complete

Suitable problem P_c known to be NP-complete

Problem: [1-In-3-Satisfiability]
Input: A set of clauses $C = \{c_1, \ldots, c_k\}$ over n boolean variables $x_1, \ldots, x_n$, where every clause contains exactly three literals.
Output: YES if there is a satisfying assignment such that every clause has exactly one true literal, i.e., if there is an assignment

$$a : \{x_1, \ldots, x_n\} \to \{0, 1\}$$

such that every clause c_j is satisfied and no clause has two or three satisfied literals, and NO otherwise.

Prove 1-In-3-Satisfiability $\leq_p$ MirrorFriendlyMinimumSpanningTree.
In order to prove this we use component design. When given an instance (X, C) of 1-In-3-Satisfiability we construct an instance $(G = (V, E, w), B)$ of MFMST. This is done by building small "component" graphs that are later connected to form the desired graph G. The components will have different "responsibilities". Some ensure a

correct setting of the truth values, others which test satisfiability and components to connect them.

Outline of the transformation

Let $X = \{x_1, \ldots, x_n\}$ be the boolean variables and $C = \{c_1, \ldots, c_m\}$ the clauses over X. We set $B = n + 2m$ for the MFMST instance. We set $w(e_i) = i$. The vertex set of G:

$$V = \{x_1, \ldots, x_n\} \cup \{\overline{x}_1, \ldots, \overline{x}_n\} \cup \bigcup_{j=1}^{m} \{a_1(j), a_2(j), a_3(j)\}$$

The truth setting components are defined to be the edges.

$$\forall x_1 \in X \text{ let } T_1 \text{ be the vertex with incoming edges } x_1 \text{ and } \overline{x}_1$$

These components ensure that every spanning tree has to contain at least one of the edges x_1 or $\overline{x}_1$.
The components for clause satisfiability are defined by the following.

$$\forall c_j \in C\text{: let } S_j \text{ be the set of vertices}$$
$$S_j = \{v_1(j), v_2(j), v_3(j)\}$$

Two edges per triangle are needed to make a spanning tree. The edge not chosen is the *true* literal which satisfies c_j.
The connecting components are defined by the following. Let $c_j = l_1 \vee l_2 \vee l_3$ where l_k are literals. More specifically l_k is a boolean variable x_i or negated boolean variable $\overline{x}_i$. For every clause c_j the following three vertices are added.

$$K_j = \{(v_1(j) \vee l_1), (v_2(j) \vee l_2), (v_3(j) \vee l_3)\}$$

The vertex set V is therefore:

$$V = \bigcup_{i=1}^{n} T_i \cup \bigcup_{j=1}^{m} S_j \cup \bigcup_{j=1}^{m} K_j$$

The transformation T can be performed in time polynomial in n and m.

Answer to X is YES then answer to $T(X)$ is YES

Assume that $a : X \to \{0, 1\}$ is an assignment which satisfies all clauses c_j. Consider the graph $G = (V, E, w)$ constructed above. We begin to construct $T \subseteq E$ by selecting n edges as follows:

$$x_i \in T \iff a(x_i) = 1,$$
$$\overline{x}_i \in T \iff a(x_i) = 0.$$

Then all T_i are covered. For every clause c_j at least one connecting vertex is covered by an l_k (by virtue of some literal in c_j is set to 1). Assume that for clause c_j this is l_1. We add $a_2(j)$ and $a_3(j)$ to T. These two edges cover the other two connecting vertices and the three vertices of the triangle S_j. The set T is a spanning tree and has weight $B = n + 2m$.

Answer to $T(X)$ is YES then answer to X is YES
Lets assume that $T \subseteq E$ is a spanning tree for G with a mirrored spanning tree where both have a weight less than or equal to B. In order to cover the vertices at least one of the edges x_i or $\overline{x}_i$ has to be in T. In order to cover the vertices of S_j, the spanning tree T has to contain at least two of the edges. Therefore $|T| = B = n + 2m$, and we have that T contains exactly one edge from every T_i and exactly two edge from every S_j.
We define an assignment: $a(x_i) = 1$ if $x_i \in T$ and 0 if $\overline{x}_i \in T$
As T contains exactly one of x_i or $\overline{x}_i$ the assignment a is well defined. We still need to show that the assignment satisfies all clauses c_j. Let $c_j = l_1 \vee l_2 \vee l_3$ be a clause where l_k are literals. For every component S_j exactly two edges belong to T, say $\{v_1(j), v_2(j)\}$ and $\{v_2(j), v_3(j)\}$. They cover also the connecting vertices $(v_1(j) \vee l_1)$ and $(v_2(j) \vee l_2)$. The third connecting vertex $(v_3(j) \vee l_3)$ attached to S_j has to be covered by l_3. By the construction of G l_3 corresponds to a literal in c_j and by the construction of the assignment a, l_3 is set to true and clause c_j is then satisfied because only one literal in a clause must be true.

d) Find an algorithm which solves the optimizing version of the problem

e) Prove the worst-case running time of the algorithm

f) Implement the algorithm developed in d)