Tutorial Exact Algorithms

Exercise T19

Design an algorithm that solves Minimum Exact Set Cover in time $O^*(2^{m/2})$ using Split&List.
Recall that this problem is defined as follows: given a finite set $\mathcal{U} = \{1, \ldots, n\}$ and a collection $\mathcal{F}$ of $m > 0$ subsets of $\mathcal{U}$, you have to find a subset $\mathcal{F}' \subseteq \mathcal{F}$ of minimum size that partitions $\mathcal{U}$.

Solution

We define the characteristic vector $v(U)$ of a subset $U \subseteq \mathcal{U}$ as $(i_1, \ldots, i_n)$ where $i_j = |\{u \in U \mid j \in u\}|$. A subset of $\mathcal{U}$ is a exact set cover if $v(U) = (1, \ldots, 1)$.
We can now easily compute a minimum exact cover in time $O^*(2^{m/2})$ by partitioning $\mathcal{U}$ into L and R and computing the $v(U)$ for each $U \subseteq L$ and each $U' \subseteq L$. Analogously to Exact SAT, checking whether $v(U) + v(U') = (1, \ldots, 1)$ can be done in time $O^*(2^{|L|} + 2^{|R|}) = O^*(2^{m/2})$.

Exercise T20

Show that Vertex Cover cannot be solved in time $O(2^{o(n+m)})$ unless the ETH fails.

Solution

We know that Independent Set cannot be solved in time $O(2^{o(n+m)})$. Since a graph $G = (V, E)$ has a vertex cover of size k iff G has an independent set of size $|V| - k$, Vertex Cover cannot be solved in time $O(2^{o(n+m)})$ unless the ETH fails.

Exercise T21

Show that Dominating Set cannot be solved in time $O(2^{o(n+m)})$ unless the ETH fails.

Solution

Let $G = (V, E)$ and k be an input instance for Vertex Cover. We construct $G' = (V', E')$ such that G' has a dominating set of size $k + 1$ iff G has a vertex cover of size k. Moreover, $|V'| + |E'| \in O(|V| + |E|)$.
We set $V' = \{x_1, x_2\} \cup V \cup E$ and include the following edges in E':

- $\{x_1, x_2\}$
- $\{x_1, v\}$ forall $v \in V$
- $\{x_1, e\}$ if $e \in E$ and $x_1 \in e$

Let C be a vertex cover in G. Then C dominates E in G'. Hence, $C \cup \{x_1\}$ dominates G', since x_1 is adjacent to all nodes in V.

Now let D be a dominating set in G' of size $k+1$. Since x_2 is dominated, we have either $x_1 \in D$ or $x_2 \in D$. But since $N[x_2] \subset N[x_1]$, we can assume wlog that $x_1 \in D$. But this implies, that all nodes in $V' \setminus E$ are dominated. Since each $e \in E$ has only neighbors in V and all these nodes are already dominated, we can assume that $D \cap E = \emptyset$ by simply exchanging e with one of its neighbors otherwise.

Since D is dominating and since $D \cap E = \emptyset$, each $e \in E$ is adjacent to some $v \in V \cap D$ in the graph G'. But this implies that $D \setminus \{x_1\}$ is a vertex cover in G of size k.

Clearly $|E'| = 2|E| + |V| + 1$ and $|V'| = |E| + |V| + 2$. Therefore, Dominating Set cannot be solved in time $O(2^{o(n+m)})$ unless Vertex Cover can be solved in time $O(2^{o(n+m)})$. But the latter would imply that the ETH fails.

Homework Assignment H19 (10 Points)

Let $a, b, c, b' \in \mathbf{N}^m$ and $\leq_l$ the lexicographical ordering of vectors. Proof that $a + b > c$ and $b' > b$ implies $a + b' > c$.

Solution

Let for a vector v the i-th position of v denoted by $v[i]$.

Let $a + b > c$ and i the first position such $(a+b)[i] > c[i]$. Such a position exists since all vectors are of length m. Then we have $(a+b)[j] = c[j]$ for all $j < i$.

Moreover, let s be the first position such that $b'[s] > b[s]$. Again, such a position must exist.

If $s \leq i$, we have $(a+b')[j] = (a+b)[j] = c[j]$ for all $j < s$ and $(a+b')[s] > (a+b)[s] = c[s]$, and therefore $a + b' > c$.

If $s > i$, we have $(a+b')[j] = (a+b)[j] = c[j]$ for all $j < i$ and $(a+b')[i] = (a+b)[i] > c[i]$, which implies $a + b' > c$.

Homework Assignment H20 (10 Points)

Proof that in Strassen's Algorithm, we have

$$\begin{aligned} C1,2 &= M3 + M5 \\ C2,1 &= M2 + M4 \\ C2,2 &= M1 + M3 + M6 - M2 \end{aligned}$$

Solution

We have

$$M_3 + M_5 = A_{1,1}(B_{1,2} - B_{2,2}) + (A_{1,1} + A_{1,2})B_{2,2} = A_{1,1}B_{1,2} + A_{1,2}B_{2,2} = C_{1,2}$$

and

$$M_4 + M_2 = A_{2,2}(B_{2,1} - B_{1,1}) + (A_{2,1} + A_{2,2})B_{1,1} = A_{2,2}B_{2,1} + A_{2,1}B_{1,1} = C_{2,1}.$$

Moreover,

$$\begin{aligned}
& M1 + M3 + M6 - M2 \\
= \; & (A_{1,1} + A_{2,2})(B_{1,1} + B_{2,2}) + A_{1,1}(B_{1,2} - B_{2,2}) + (A_{2,1} - A_{1,1})(B_{1,1} + B_{1,2}) \\
& -(A_{2,1} + A_{2,2})B_{1,1} \\
= \; & A_{1,1}B_{1,1} + A_{1,1}B_{2,2} + A_{2,2}B_{1,1} + A_{2,2}B_{2,2} + A_{1,1}B_{1,2} - A_{1,1}B_{2,2} \\
& +A_{2,1}B_{1,1} + A_{2,1}B_{1,2} - A_{1,1}B_{1,1} - A_{1,1}B_{1,2} - A_{2,1}B_{1,1} - A_{2,2}B_{1,1} \\
= \; & A_{1,1}B_{1,1} - A_{1,1}B_{1,1} + A_{1,1}B_{2,2} - A_{1,1}B_{2,2} + A_{2,2}B_{1,1} - A_{2,2}B_{1,1} \\
& +A_{2,2}B_{2,2} + A_{1,1}B_{1,2} - A_{1,1}B_{1,2} + A_{2,1}B_{1,1} - A_{2,1}B_{1,1} + A_{2,1}B_{1,2} \\
= \; & A_{2,2}B_{2,2} + A_{2,1}B_{1,2} = C_{2,2}
\end{aligned}$$

Homework Assignment H21 (10 Bonus Points)

Show that Maximum Leaf Spanning Tree cannot be solved in time $O(2^{o(n+m)})$ unless the ETH fails.

The Maximum Leaf Spanning Tree is defined as:

Input:	A graph $G = (V, E)$, a number $k \in \mathbf{N}$
Question:	Is there a spanning tree in G with at least k leafs.

Solution

Recall that the solution to Exercise T21. In the proof, we can always assume that the dominating set is always connected, as x_1 belongs to the solution and otherwise only nodes from V. Hence, the graph induced by the dominating set is connected.

Now we show that a graph G has a connected dominating set of size k iff G has a spanning tree with at least $|V| - k$ nodes.

Let $V' \subseteq V$ be a connected dominating set in G. Thus, V' contains a spanning tree T'. All other nodes are adjacent to V', since V' is dominating. By connecting each $v \in V \setminus V'$, we obtain a spanning tree T for G. Clearly, this tree has at least $|V| - k$ leafs.

Now let T be a spanning tree in G with at least $|V| - k$ leafs. Since T is a spanning tree, all nodes in V are adjacent to the inner nodes of T. These nodes are thus dominating and also connected. Thus, the graph has a connected dominating set of size k.

Thus, instead of finding a dominating set of size $k + 1$ in the reduction from T21, we can ask for a spanning tree with at least $|V| - (k + 1)$ nodes.

If we could find such a tree in $O(2^{o(n+m)})$, we could also solve Vertex Cover in $O(2^{o(n+m)})$. But this would imply that the ETH fails.

Homework Assignment H22 (10 Bonus Points)

Design an algorithm that solves Partial Vertex Cover in time $O^*(2^{\omega n/3})$.

The Partial Vertex Cover is defined as:

Input:	A graph $G = (V, E)$, numbers $k, t \in \mathbf{N}$
Question:	Is there a $V' \subseteq V$ with $|V| = k$ such that $|\{e \in E \mid e \cap V' \neq \emptyset\}| \geq t$

Solution

We will solve this problem analogously to MaxCut. That is, we will construct a graph G' that contains a triangle of cost t iff there is a vertex set covering t in G. Finding the triangle is than the same as in the algorithm for MaxCut.

First, we partition V into V_1,V_2, and V_3. Then for each $k_1 + k_2 + k_3$ we construct the graph $G = (V', E')$ as follows: For each subset $V' \subseteq V_i$ with $|V'| = k_i$, there is a node in V'.

(In the following, we let $V_4 = V_1$.

For each $X \subseteq V_i$ and $Y \subseteq V_{i+1}$, we add an edge with cost $c_i + c_{i,i+1}$, where $c_i = |\{e \in E \mid e \cap X \neq \emptyset, e \in V_i\}|$ denotes the number of edges covered by X that are completely contained in V_i and

$c_{i,i+1} = |\{e \in E \mid e \cap (X \cup Y) \neq \emptyset, e \in V_i \cup V_{i+1}\}|$ denotes the number of edges covered by X and Y that are completely contained in $V_i \cup V_{i+1}$.

Now let v_1, v_2, v_3 be a triangle in G and V' be the corresponding node set in G. Since each edge that is covered by V' is counted only in the edge weight of a single edge v_i, v_j, V' covers at least t edges.

Moreover, if V' covers t edges, the triangle v_1, v_2, v_3 has costs t, since each covered edge is counted in some edge.

Hence, we can solve Partial Vertex Cover in time $O^*(2^{\omega n/3})$ by using the algorithm for MaxCut with an additional overhead of k^3.