RWTH Aachen University **WS 2010/11**
Lehrgebiet Theoretische Informatik Tutorial 2
Kneis—Langer—Sikdar 26.10.2010

Tutorial Exact Algorithms

Exercise T3

Let F be a formula in CNF. A clause $\{l_1, \ldots, l_k\}$ is exactly satisfied if exactly one of the literals is true. An assignment satisfies F exactly, if each clause is exactly satisfied.
Find an algorithm that decides whether F can be exact satisfied in time $O^*(\tau(3,3,3)^n)$.

Solution

Obviously, we can eliminate clauses of length 1.
Choose a clause of maximal length l. If $l = 2$, the problem can be solved in polynomial time, since the corresponding connectivity graph must be bipartite in order to satisfy the formula.
If $l \geq 3$, we simply branch on all possibilities, which yields $\tau(l, l, \ldots, l) \leq \tau(3,3,3)$.

Exercise T4

The problem Set Packing is defined as follows:

Input:	A family $\mathcal{S} = \{S_1, \ldots, S_m\}$ of sets.
Question:	Is there a subset $S' \subset \mathcal{S}$ of size at least k such that $S_i \cap S_j = \emptyset$ for all $S_i, S_j \in S'$ with $i \neq j$?

Design an algorithm that solves this problem in $O^*(1.47^m)$.

Solution

Clearly, if a set S_i is disjoint to all other sets, we can use it for the packing. If a set has a conflict with exactly one other set, we can also choose it, since we don't loose anything. Thus, tacking any remaining set blocks at least two other sets and we obtain the branching vector $(1, 3)$.
The problem can easily be modeled as an input for independent set. Hence, any algorithm that solves Independent Set with a runtime of $O^*(c^n)$ also solves Set Packing in $O^*(c^m)$ steps.

Homework Assignment H3 (10 Points)

The problem Hitting Set is defined as follows:

Input:	A family $\mathcal{S} = \{S_1, \ldots, S_m\}$ of subsets of a universe $\mathcal{U}$.
Question:	Is there a $U \subseteq \mathcal{U}$ of size at most k such that $U \cap S_i \neq \emptyset$ for all $S_i \in \mathcal{S}$

Design an algorithm that solves this problem in at most $O^*(\sqrt{2}^{|\mathcal{U}|+|\mathcal{S}|})$ steps. Hint: This problem is similar to Set Cover.

Solution

If there is $S \in \mathcal{S}$ with $|S| = 1$, then obviously the unique element in S must be chosen to hit S. We can therefore assume $|S| \geq 2$ for all $S \in \mathcal{S}$. If now there is an element $u \in \mathcal{U}$ that is contained in only one set $S \in \mathcal{S}$, then we can safely remove u from the instance, since we can always replace u by any $v \in S$ (which exists since $|S| > 1$) to get a solution of the same size.

If each element $u \in \mathcal{U}$ is contained in at most two sets, i.e., in exactly two sets, then we can again solve the problem by a simple reduction to Edge Cover, where the elements become edges between the respective two sets they cover.

If otherwise each $u \in \mathcal{U}$ is contained in at least three sets, we can simply branch on whether an element $u \in \mathcal{U}$ is contained in the solution U or not. If l is the number of sets that u is contained in, this yields a branching vector of $(1, 1 + l)$ when we analyze the algorithm in $|\mathcal{S}| + |\mathcal{U}|$. Here, $\tau(1, l + 1)$ reaches its maximum for $l = 3$ (assuming $l > 2$), and $\tau(1, 4) \approx 1.3803 < \sqrt{2}$.

Homework Assignment H4 (10 Points)

Give a formal proof for the following lemmas:

Lemma 1 *Let $G = (V, E)$ be a graph and $v \in V$ with $\deg(v) = 1$. Then*

$$\alpha(G) = \alpha(G \setminus \{v\}) + 1.$$

Lemma 2 *Let $G = (V, E)$ and let G consist of connected components $G_1, \ldots, G_l$. Then*

$$\alpha(G) = \sum_{i=1}^{l} \alpha(G_i).$$

Lemma 3 *Let $G = (V, E)$ be a graph and $\Delta(G) \leq 2$. Then $\alpha(G)$ can be computed in polynomial time.*

Solution Lemma 1 as stated above is wrong, as can easily be seen in a graph $G = (V, E)$ that contains a single edge. We will therefore show the intended lemma instead, i.e., $\alpha(G) = \alpha(G \setminus N[v]) + 1$.

1. If $v \in V$ has degree one, let u be the unique neighbor of v in G. Let I be an maximum independent set of $G \setminus N[v]$. Then $I' := I \cup \{v\}$ is an independent set of G of size $\alpha(G \setminus N[v]) + 1$. To see that $|I'| = \alpha(G)$, let I'' be an maximum independent set of G. If $v \in I''$, then $|I \setminus \{v\}| = \alpha(G) - 1 = \alpha(G \setminus N[v]) = |I| + 1 = |I'|$. If $v \notin I''$, then $u \in I''$ by the maximality of I'' (otherwise we could add v to I''). We can then consider $(I'' \setminus \{u\}) \cup \{u\}$ instead, which is an independent set of the same size of I'' that contains v, which yields the first case again.

2. Since the union of any independent sets $I_1, \ldots, I_l$ for $G_1, \ldots, G_l$, respectively, obviously yields an independent set for G, $\alpha(G) \geq \sum_{i=1}^{l} \alpha(G_i)$ is clear. To see that $\alpha(G) \leq \sum_{i=1}^{l} \alpha(G_i)$, consider a maximum independent set I of G and let, for $1 \leq i \leq l$, be $I_i = V(G_i) \cap I$. Then, for each $1 \leq i \leq l$, the respective I_i is an independent set for G_i, i.e., $|I_i| \leq \alpha(G_i)$. Therefore, $\alpha(G) = |I| = |I_1| + \cdots + |I_l| \leq \alpha(G_1) + \cdots + \alpha(G_l)$.

3. If $\Delta(G) = 2$, then G is a disjoint union of connected components $G_1, \ldots, G_l$, where each G_i is either an isolated node, a path or a cycle. In each of these three cases $\alpha(G_i)$ can easily be computed in polynomial time. The statement then follows with Lemma 2.