RWTH Aachen University **WS 2010/11**
Lehrgebiet Theoretische Informatik Tutorial 6
Kneis—Langer—Sikdar 23.11.2010

Tutorial Exact Algorithms

Exercise T10 CHROMATIC NUMBER. A k-coloring c of an undirected graph $G = (V, E)$ is an assignment of colors to the vertices of the graph $c\colon V \to \{1, \ldots, k\}$ such that adjacent vertices receive different colors. The smallest k for which G has a k-coloring is called the *chromatic number* of G and is denoted by $\chi(G)$.
Note that each monochromatic color class is an independent set and this is a typical partition problem that can be solved by brute-force by trying all possible colors for each vertex. The running time of such a brute-force algorithm is $O^*(n^n)$.
The dynamic programming algorithm that we discussed in class started out by defining the following subproblem: For $S \subseteq V$, we let $\mathrm{OPT}[S]$ denote the chromatic number of the graph $G[S]$. The DP algorithm then computes the values of $\mathrm{OPT}[S]$ in order of increasing cardinalities of the set S. For $S = \emptyset$, it is clear that $\mathrm{OPT}[S] = 0$. For $|S| \geq 1$,

$$\mathrm{OPT}[S] = 1 + \min_X \{\mathrm{OPT}[S \setminus X]\colon X \text{ is a maximal independent set of } G[S]\}.$$

Note that deleting a (maximal) independent set from a graph does not necessarily decrease the chromatic number. If this were so, one could compute the chromatic number of any graph in polynomial time! What is true is the following: if X is a maximal independent set in a graph G then

$$\chi(G) - 1 \leq \chi(G \setminus X) \leq \chi(G).$$

Now the running time of the above algorithm can be bounded by

$$\sum_{i=1}^{n} \binom{n}{i} \cdot 2^i = (1+2)^n = 3^n.$$

The 2^i in the summation occurs because we look at all possible subsets of a set S and check whether it is maximal independent. However if we use Moon and Moser's algorithm to compute maximal independent sets then we do better. Recall that Moon and Moser's Theorem states that the number of maximal independent sets in an n-vertex graph is bounded above by $3^{n/3}$ and these sets can be output in time $O^*(3^{n/3})$. Using this result, the running time of the algorithm can be bounded by

$$\sum_{i=1}^{n} \binom{n}{i} \cdot 3^{i/3} = (1 + 3^{1/3})^n < 2.4422^n.$$

Exercise T11 BALANCED PARTITION. You have a set $\{a_1, \ldots, a_n\}$ of n integers in the range $0 \ldots k$. You are to partition this set into two subsets S_1, S_2 such that $|\mathrm{sum}(S_1) - \mathrm{sum}(S_2)|$ is minimized, where $\mathrm{sum}(X)$ for $X \subseteq \{a_1, \ldots, a_n\}$ denotes the sum of the integers in the set X. Design a dynamic programming algorithm to solve this problem in time $O(n^2k)$. Note that BALANCED PARTITION is NP-complete and that this running time is *not* polynomial in the input size.

Homework Assignment H11 (10 Points) The DOMATIC NUMBER problem is defined as follows: given an undirected graph $G = (V, E)$, compute the largest integer k such that there is a partition of V into pairwise disjoint sets $V_1, \ldots, V_k$ such that $V_1 \cup \cdots \cup V_k = V$ and each V_i is a dominating set for G. The largest such integer is called the *domatic number* of G. Show how to compute the domatic number of an n-vertex graph in time $O^*(3^n)$.

Homework Assignment H12 (10 Points) You are given an *ordered* sequence $c_1 \ldots, c_n$ of n cities, and the distances d between every pair of cities. You must partition the cities into two subsequences (not necessarily contiguous) such that person A visits all cities in the first sequence, in order, and person B visits all cities in the second sequence, in order, and such that the sum of the total distances travelled by A and B is minimized.