ADFOCS 2013

Algebraic Approaches to Exact Algorithms

Exercises, Monday, 5th August 2013

1. You are given a "black-box" algorithm A which *decides* whether a given graph is k-colorable in $O(\alpha^n)$ time, for some constant $\alpha > 1$.

   a) (Easy) Show that one can *find* a k-coloring (if it exists) in $O^*(\alpha^n)$ time.

   b) (Tricky) The same in $O(\alpha^n \log n)$ time.

2. You are given a "black-box" algorithm A which *decides* whether a given n-vertex graph has a k-path in $O(f(k)p(n,k))$ time, for some function f and polynomial p. Show that one can *find* a k-path (if it exists) in $O(f(k)p'(n,k))$ time for some polynomial p'.

3. In the TSP problem, we are given a complete graph with a weight function $w : V^2 \to \{0, \dots, W\}$ and the goal is to find a Hamiltonian cycle H of smallest weight (i.e. $\sum_{uv \in E(H)} w(u,v)$). Describe a $O^*(2^n \cdot W)$-time, $O^*(W)$-space algorithm for the TSP problem.

4. Show an algorithm which computes the number of perfect matchings in a given n-vertex *bipartite* graph in $O^*(2^{n/2})$ time. (The solution is called Ryser's Formula.)

5. Let $\mathcal{A}, \mathcal{B} \subseteq 2^U$ for some finite set U.

   a) Show that $|\{(A,B) \in \mathcal{A} \times \mathcal{B} \ : \ \mathcal{A} \cap \mathcal{B} = \emptyset\}|$ can be computed in $O^*(|\downarrow\!\mathcal{A}| + |\downarrow\!\mathcal{B}|)$ time.

   b) Generalize a) to computing

   $$\alpha \boxtimes \beta = \sum_{\substack{A \in \mathcal{A} \\ B \in \mathcal{B} \\ A \cap B = \emptyset}} \alpha(A)\beta(B)$$

   for two functions $\alpha : \mathcal{A} \to \mathbb{N}$ and $\beta : \mathcal{B} \to \mathbb{N}$.

   c) Use b) to *count* k-paths in an n-vertex graph in $O^*(n^{k/2})$ time.

6. In the 2-bounded channel assignment problem, we are given an undirected graph $G = (V, E)$, a function $\ell : E \to \{1, 2\}$ and the number $s \in \mathbb{N}$. The goal is to find a coloring $c : V \to \mathbb{N}$ such that for every $uv \in E$, $|c(u) - c(v)| \geq \ell(uv)$ and the *span* of c, i.e. $\max_{v \in V} c(v) - \min_{v \in V} c(v)$, is at most s. Give a $O^*(3^n)$-time algorithm for this problem.

See page 2 for some hints to exercises if you need them.

Hints

1. a) Easy, b) Note that $\sum_{i=1}^{n} \alpha^n = O(\alpha^n)$.

2. Easy.

3. Adapt the $O^*(2^n)$-time polynomial space inclusion-exclusion based algorithm from the lecture.

4. Inclusion-Exclusion principle.

5. a) Recall that for a set $\mathcal{A} \subseteq 2^U$ and a function $f : \mathcal{A} \to \mathbb{N}$ the up-zeta transform $(\zeta^{\uparrow} f)(X) = \sum_{Y \supseteq X} f(Y)$ can be computed for all sets $X \in \downarrow\mathcal{A}$ in $O^*(|\downarrow\mathcal{A}|)$ time. Begin with applying the inclusion-exclusion principle and then use this fast up-zeta transform algorithm.

   b) once you solved a) this should be straightforward.

   c) partition k-paths into two halves.

6. Similarly as in the $O^*(2^n)$ coloring algorithm, use the inclusion-exclusion principle compute the number of covers of V with tuples $(I_0, \ldots, I_s)$ such that for every $uv \in E$, for every i, j, if $u \in I_i$ and $v \in I_j$ then $|i-j| \geq \ell(uv)$. Using dynamic programming this should give $O^*(4^n)$ time. Then speed-up the DP algorithm using the fast zeta transform to $O^*(3^n)$ time.