Exercise sheet 3 – Solutions

COMP6741: Parameterized and Exact Computation

Serge Gaspers

Semester 2, 2017

Exercise 1. Suppose there exists a $O^*(1.2^n)$ time algorithm, which, given a graph G on n vertices, computes **the size** of a largest independent set of G.

Design an algorithm, which, given a graph G, **finds** a largest independent set of G in time $O^*(1.2^n)$.

Solution sketch.

- Compute k, the size of a largest independent set of G
- Find a vertex v belonging to an independent set of size k
  - We can do this by going through each vertex u of G, and checking whether $G - N_G[u]$ has an independent set of size $k-1$
- Recurse on $(G - N_G[v], k-1)$

Exercise 2. Let A be a branching algorithm, such that, on any input of size at most n its search tree has height at most n and for the number of leaves $L(n)$, we have

$$L(n) = 3 \cdot L(n-2)$$

Upper bound the running time of A, assuming it spends only polynomial time at each node of the search tree.

Solution. We need to minimize $L(n) = 2^\alpha$ subject to $1 \geq 3 \cdot 2^{\alpha \cdot (-2)}$.

This solves to $2^\alpha = 3^{1/2} = \sqrt{3}$. The running time of A is $O^*(3^{n/2})$.

Exercise 3. Same question, except that

$$L(n) \leq \max \begin{cases} 2 \cdot L(n-3) \\ L(n-2) + L(n-4) \\ 2 \cdot L(n-2) \\ L(n-1) \end{cases}$$

Solution. By the Balance property, $(3,3) \leq (2,4)$. By the Dominance property, $(2,4) \leq (2,2)$. For every positive $alpha$, $1 \geq 2^{-\alpha}$ is satisfied.

Thus, it suffices to minimize $L(n) = 2^\alpha$ subject to $1 \geq 2 \cdot 2^{\alpha \cdot (-2)}$

This solves to $2^\alpha = 2^{1/2} = \sqrt{2}$. The running time of A is $O^*(2^{n/2})$.

Exercise 4. Consider the MAX 2-CSP problem

MAX 2-CSP

Input: A graph $G = (V, E)$ and a set S of *score functions* containing

- a score function $s_e : \{0,1\}^2 \to \mathbb{N}_0$ for each edge $e \in E$,
- a score function $s_v : \{0,1\} \to \mathbb{N}_0$ for each vertex $v \in V$, and
- a score "function" $s_\emptyset : \{0,1\}^0 \to \mathbb{N}_0$ (which takes no arguments and is just a constant convenient for bookkeeping).

Output: The maximum score $s(\phi)$ of an assignment $\phi : V \to \{0,1\}$:

$$s(\phi) := s_\emptyset + \sum_{v \in V} s_v(\phi(v)) + \sum_{uv \in E} s_{uv}(\phi(u), \phi(v)).$$

1. Design simplification rules for vertices of degree ≤ 2.
2. Using the simple analysis, design and analyze an $O^*(2^{m/4})$ time algorithm, where $m = |E|$.
3. Use the measure $\mu := w_e \cdot m + \left(\sum_{v \in V} w_{d_G(v)}\right)$ to improve the analysis to $O^*(2^{m/5})$.

Solution sketch. (a) Simplification rules

S0 If there is a vertex y with $d(y) = 0$, then set $s_\emptyset = s_\emptyset + \max_{C \in \{0,1\}} s_y(C)$ and delete y from G.

S1 If there is a vertex y with $d(y) = 1$, then denote $N(y) = \{x\}$ and replace the instance with (G', S') where $G' = (V', E') = G - y$ and S' is the restriction of S to V' and E' except that for all $C \in \{0,1\}$ we set

$$s'_x(C) = s_x(C) + \max_{D \in \{0,1\}} \{s_{xy}(C, D) + s_y(D)\}.$$

S2 If there is a vertex y with $d(y) = 2$, then denote $N(y) = \{x, z\}$ and replace the instance with (G', S') where $G' = (V', E') = (V - y, (E \setminus \{xy, yz\}) \cup \{xz\})$ and S' is the restriction of S to V' and E', except that for $C, D \in \{0,1\}$ we set

$$s'_{xz}(C, D) = s_{xz}(C, D) + \max_{F \in \{0,1\}} \{s_{xy}(C, F) + s_{yz}(F, D) + s_y(F)\}$$

if there was already an edge xz, discarding the first term $s_{xz}(C, D)$ if there was not.

(b) Branching rules

B Let y be a vertex of maximum degree. There is one subinstance (G', s^C) for each color $C \in \{0,1\}$, where $G' = (V', E') = G - y$ and s^C is the restriction of s to V' and E', except that we set

$$(s^C)_\emptyset = s_\emptyset + s_y(C),$$

and, for every neighbor x of y and every $D \in \{0,1\}$,

$$(s^C)_x(D) = s_x(D) + s_{xy}(D, C).$$

Simple analysis

- Branching on a vertex of degree ≥ 4 removes ≥ 4 edges from both subinstances
- Branching on a vertex of degree 3 removes ≥ 6 edges from both subinstances since G is 3-regular.

The recurrence $T(m) \leq 2 \cdot T(m - 4)$ solves to $2^{m/4}$

(c) Measure based analysis Using the measure

$$\mu := w_e \cdot m + \left(\sum_{v \in V} w_{d_G(v)}\right)$$

we constrain that

$$
\begin{aligned}
w_d &\leq 0 && \text{for all } d \geq 0 \text{ to ensure that } \mu \leq w_e m \\
d \cdot w_e/2 + w_d &\geq 0 && \text{for all } d \geq 0 \text{ to ensure that } \mu(G) \geq 0 \\
-w_0 &\leq 0 && \text{constraint for S0} \\
-w_2 - w_e &\leq 0 && \text{constraint for S2}
\end{aligned}
$$

$$1 - w_d - d \cdot w_e - d \cdot (w_j - w_{j-1}) \leq 0$$

for all $d, j \geq 3$.

Using $w_e = 0.2$, $w_0 = 0$, $w_1 = -0.05$, $w_2 = -0.2$, $w_3 = -0.05$, and $w_d = 0$ for $d \geq 4$, all constraints are satisfied and $\mu(G) \leq m/5$ for each graph G.