RWTH Aachen
Lehrgebiet Theoretische Informatik
Dreier—Kuinke—Rossmanith

WS 2018
Problem Set 11
14.1.2018

Parameterized Algorithms Tutorial

Tutorial Exercise T1

Consider this problem:

- Input: A finite deterministic automaton M and a $k \in \mathbf{N}$
- Parameter: k
- Question: Is there a word of length at least k in $L(M)$?

Do not forget that the alphabet size of M can be arbitrarily large.

a) Is this problem NP-hard?

b) What is its parameterized complexity?

Proposed Solution

The problem lies in P. We iteratively compute sets Q_0, Q_1, Q_2, Q_{n+1} such that Q_i is the set of all states that are reachable with exactly i steps. If $k \leq n$, we return yes if Q_k is nonempty. Otherwise, wer return yes if Q_{n+1} is nonempty. If Q_{n+1} is non-empty then there is an accepting word wich visits a state twice. Using the pumping lemma we see that $L(M)$ contains arbitrary long words, especially words of length at least k.

Tutorial Exercise T2

We are analyzing a variant of the *k-leaf spanning tree problem.* Instead of looking for a spanning tree with at least k leafs, we are looking for one with *exactly* k leafs.

a) Should we distinguish two versions of this problem: "exact k-leaf subtree" and "exact k-leaf spanning tree"? For the original problem the two versions where equally hard to solve.

b) What is the parameterized complexity of both problems?

c) Find an efficient algorithm if a variant is in FPT.

Proposed Solution

The problem *exactly k-leaf spanning tree* with $k = 2$ asks for the existence of a Hamiltonian path and is therefore NP-complete. We say *exactly k-leaf spanning tree* is para-NP complete. Note that $W[1] \subseteq$ para-NP. Under the assumption $W[1] \neq$ FPT the problem does not lie in FPT. On the other hand, *exactly k-leaf subtree* lies in FPT: We compute a *k-leaf spanning tree* and remove leafs until the number of leaves is exactly k.

Tutorial Exercise T3

The MSO type of a structure S with a finite domain is the set of all MSO formulas ϕ with $S \models \phi$. Let us say that the q-type are the formulas in the type that have at most q variables. For simplicity we always assume that formulas are in prenex normal form.

a) Is the q-type of a structure finite or can it be infinite?

b) If it is infinite, are there only finitely many equivalence classes with regard to logical equivalence betweens formulas?

c) How could representatives of these equivalence classes look like?

Proposed Solution

a) The q-type of a structure can be infinite. If the q-type contains a formula ϕ we can construct an infinte sequence of equivalent formulas which all have at most q variables by conjuncting with true statements.

b) and c) There are at most finitely many equivalence classes. We take a formula ϕ with q variables. Then we rename these variables into $x_1, \ldots, x_q$. The quantifier free subformula is now converted into an equivalent short formula in CNF. The result is an equivalent formula whose size depends only on q.

Homework H1

Let t be a constant. Design an efficient algorithm that solves the following problem in polynomial time:

- Input: A graph G and a number k
- Output: A t-protrusion in G of size at least k or the answer that no such protrusion exists.

The degree of a polynomial that upper bounds the running time may depend on t.

Proposed Solution

We enumerate all subsets of vertices of size at most t. We remove the boundary from the graph and check if one of the components has size at least k and treewidth at most t. The treewidth of a graph can be recognized in FPT-time with parameter t. The subsets can be enumerated in $O(n^t)$.

Homework H2

Find a graph class that excludes some H as a topological minor, but contains *every* graph H as a minor (i.e., contains a graph that has H as a minor).

Proposed Solution

Let K_4 be the clique with 4 vertices. If a graph G contains K_4 as a topological minor then G contains vertices with degree at least 4. We need to construct a family of graphs of degree at most 3 which contains every graph H as a minor. One such family is the family of all 3-regular graphs.