RWTH Aachen **SS 2014**
Lehrgebiet Theoretische Informatik Problem Set 2
Reidl—Rossmanith—Sánchez—Sikdar 23.10.2011

Parameterized Algorithms Tutorial

Tutorial Exercise T1

In this exercise, we wish to design an algorithm for the FEEDBACK VERTEX SET problem. An input to this problem is a graph $G = (V, E)$ and an integer k and the question is whether the graph has a vertex subset of size at most k whose deletion results in a forest. Such a vertex set is called a *feedback vertex set*.

1. A feedback vertex set hits all cycles in the graph. Therefore vertices which do *not* belong to cycles may be safely removed. Use this observation to design *reduction rules* to simplify the graph.
2. Can you bound the girth of the reduced graph? Use the bound to design an algorithm for the problem.
3. What is the running time of your algorithm?

Tutorial Exercise T2

Given a boolean formula φ in CNF with n variables and m clauses and an integer k, you have to decide whether there exists an assignment to the variables that satisfies at least k clauses. Assume that the literals appearing in a clause are all distinct and that no clause contains a literal and its negation. We first consider several special cases.

1. If φ contains only unit clauses (clauses with one literal), then how can you find the optimum assignment?
2. If there are k clauses in φ that each contain k literals, then show that one can find an assignment satisfying all these clauses in $|\varphi|$ time.
3. Show that one can always find an assignment that satisfies at least $m/2$ clauses of φ.

Use these facts to design an FPT-algorithm with k as parameter.

Homework H1

Consider the following algorithm for VERTEX COVER. Choose an arbitrary edge $e = \{u, v\}$ that has not yet been covered and branch on the two subcases: on one branch include u in the solution and in the other include v in the solution. Return the smaller of the two solutions.

Does this algorithm run in FPT-time if parameterized by the size of the *minimum* vertex cover? If yes, provide a formal proof. If no, provide a generic counterexample.

Homework H2

Consider the following algorithm for INDEPENDENT SET. Given a graph $G = (V, E)$, we first check whether the maximum degree is at most two. If this is the case, we can find a maximum independent set (in polynomial time) and return the solution. Else we find a vertex u of largest degree (which is at least 3) and branch on it. There are two cases: we can either include u in the independent set in which case we must delete all its neighbors; if we choose *not* to include it in the independent set then we can safely delete it from the graph. We branch on these two cases and return the bigger of the two solutions.

1. Does this algorithm compute a largest independent set? How would you prove it?
2. Write down the recurrence that governs the running time of this algorithm. Solve the recurrence using methods discussed in the tutorial. What is the exponential factor in the running time?

In case you're worried about what the parameter is in this case, it is simply the number of vertices in the graph. Remember that the parameter *must* decrease in each branch of your recursion tree, and with this choice of parameter, this requirement is satisfied.