Assignment 4
COMP6741: Parameterized and Exact Computation

Serge Gaspers
19T3

Assignment 4 is a group assignment. The lecturer-in-charge will announce the groups on 11 Oct 2019. Each group consists of at least 3 members.

For the solutions to this assignment, you may rely on all theorems, lemmas, and results from the lecture notes. If any other works (articles, Wikipedia entries, lecture notes from other courses, etc.) inspired your solutions, please cite them and give a list of references at the end. You may use any result you find in the literature, without re-proving it. Existing implementations and libraries may also be used, as long as their licenses allow unrestricted academic use.

If you have questions about this assignment, please post them to the Forum.

Due date. This assignment is due on Friday, 25 Oct 2019, at 17:59 Sydney time. Submitting $x$ hours after the deadline, with $x > 0$, reduces the grade by $\lfloor x \rfloor$ per cent.

How to submit. There will be one Bitbucket GIT repository per group. The Readme.md file in this repository describes where to put various files, including the report answering the questions below, by the submission deadline.

We will consider two parameters for Feedback Vertex Set and its optimization version, Minimum Feedback Vertex Set.

A feedback vertex set of a graph $G$ is a subset $S$ of vertices such that $G - S$ is acyclic.

<table>
<thead>
<tr>
<th>Feedback Vertex Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: A graph $G = (V, E)$ and an integer $k$</td>
</tr>
<tr>
<td>Question: Does $G$ have a feedback vertex set of size at most $k$?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Minimum Feedback Vertex Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: A graph $G = (V, E)$</td>
</tr>
<tr>
<td>Output: A smallest feedback vertex set of $G$</td>
</tr>
</tbody>
</table>

We use the notation from [SW13] to compare graph parameters. A graph parameter $p$ is a function that assigns a real number to each graph. If $p, q$ are graph parameters, then the parameter $p$ upper bounds the parameter $q$ if there is a function $f$ such that for every graph $G$ we have that $q(G) \leq f(p(G))$. If $p$ does not upper bound $q$ and $q$ does not upper bound $p$, then $p$ and $q$ are said to be incomparable.

Let $G$ be a graph. The feedback vertex set number of $G$ is the size of a smallest feedback vertex set of $G$. The vertex cover number of $G$ is the size of a smallest vertex cover of $G$. The degree-2 deletion number of $G$ is the size of a smallest degree-2 deletion set, that is a smallest set of vertices $S$ such that $G - S$ has maximum degree at most 2.

Exercise 1. Prove that the feedback vertex set number and the degree-2 deletion number are incomparable. Are the vertex cover number and the feedback vertex set number incomparable? [20 points]

Exercise 2. Prove that Feedback Vertex Set is FPT for parameter vertex cover number. [20 points]

Hint: One approach is to compute a smallest vertex cover, then kernelize the instance.

Exercise 3. Prove that the following problem is FPT parameterized by $k$: Given a graph $G$ and an integer $k$, decide whether $G$ has a degree-2 deletion set of size at most $k$. [20 points]
In Assignment 5, you will be asked to implement an algorithm for Minimum Feedback Vertex Set and analyse the performance of the implementation with respect to the parameters vertex cover number and degree-2 deletion number. In particular, we would like to find out whether your implementation is fast if the graph has small vertex cover number or small degree-2 deletion number. You will run your implementation on benchmark instances.

We will use sage for all implementations. See [http://www.sagemath.org/](http://www.sagemath.org/).

**Exercise 4.** Obtain a selection of 40 benchmark instances by (1) generating 10 random instances (for what average degree do you expect the parameters to be small?), (2) finding 10 real-world instances (for example from existing benchmark graphs), and (3) crafting/generating your own new instances, of which 10 have a small vertex cover and 10 have a small degree-2 deletion set. Ideally your instances have diverse properties (except maybe the random ones), have varying degrees of difficulty (the easiest ones should just be solvable by brute-force methods, and your hardest ones should be expected to be slightly out of reach for competitive algorithms within a cutoff time of 2 minutes), and the generation of instances should be reproducible.

Give a high-level description of the instances, estimate how challenging it is to solve them, and explain how they are designed / where they were found.

Save each such graph as sage graph object (of type `sage.graphs.graph.Graph`) into the sub-folder with your benchmark instances (GIT repository). Name them `RandomX.graph.sobj`, `RealX.graph.sobj`, `CraftedVCX.graph.sobj`, and `CraftedD2DX.graph.sobj`, where X are digits ranging from 0 to 9. [20 points]

**Exercise 5.** Implement a sage function named `checkFVS` that takes as arguments a graph $G$ (of type `sage.graphs.graph.Graph`) and a set of vertices $S$ (of type `list`) and checks whether $S$ is a feedback vertex set of $G$.

[10 points]

**Exercise 6.** Describe how you divided the work of Assignment 4, the contributions of each group member (this should be self-reported by each group member), what milestones you set, and what was your timeline. Did any of these evolve over time? If any issues or setbacks arose, how did you handle them? [10 points]

**References**