Exercise Sheet 2

Integer Programming
Academic Year 2022/23: Quarter 4

Lecturer: Christopher Hojny

TU/e

Problem Description

In the first assignment, we have derived a lower bound on the optimal objective value of the LP relaxation of a column generation model. However, we have not discussed how to compute this lower bound. As we will see next, we can do this by finding so-called stable sets in an auxiliary graph. The aim of this assignment is to develop integer programming techniques to find stable sets.

Let $G = (V, E)$ be an undirected graph. A *stable set* in G is a set $S \subseteq V$ such that, for any distinct $u, v \in S$, we have $\{u, v\} \notin E$. That is, the nodes in S are pairwise non-adjacent. A stable set S in G is called *maximum* if, for each stable set S' in G, we have $|S'| \leq |S|$.

Exercises

Exercise 1 (Relation between special sets and stable sets) 0.5 points

Consider a pair (X, Y) of positive and negative data points. Let $G = (Y, E)$, where Y is the set of negative data points and $E = \{\{y_1, y_2\} : \text{conv}(\{y_1, y_2\}) \cap \text{conv}(X) = \emptyset)\}$.

Show that there is a bijection between stable sets in G and special sets for (X, Y).

Exercise 2 (Integer programming formulation for stable sets) 1 point

Let $G = (V, E)$ be an undirected graph. Show that

$$\max \sum_{v \in V} x_v \tag{1a}$$

$$x_u + x_v \leq 1, \qquad \{u, v\} \in E, \tag{1b}$$

$$x_v \in \{0, 1\}, \qquad v \in V, \tag{1c}$$

is an integer programming formulation for finding a maximum stable set. That is, every optimal solution of (1) corresponds to a maximum stable set and vice versa.

Exercise 3 (Strengthening the formulation) 1+2 points

The feasible region of the LP relaxation of Model (1) is the polyhedron

$$P(G) = \{x \in [0, 1]^V : x_u + x_v \leq 1 \text{ for all } \{u, v\} \in E\}.$$

In general, $P(G)$ is not an integral polyhedron, i.e., we cannot just hope to find a maximum stable set by solving a linear program over $P(G)$. Therefore, we can aim to identify cutting planes. To investigate the strength of these cutting planes, let $P_I(G)$, be the integer hull of $P(G)$, to which we refer to as the *stable set polytope*.

1. An odd cycle of length k is an undirected graph $C_k = (V_k, E_k)$ such that the nodes can be labeled $v_1, \ldots, v_k$ and the edges can be labeled $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{k-1}, v_k\}, \{v_k, v_1\}$.
   Let $G = (V, E)$ be an undirected graph and let $C = (V', E')$ be an *induced* subgraph of G that is isomorphic to C_k. Show that the so-called *odd cycle inequality*

$$\sum_{v \in V'} x_v \leq \frac{k-1}{2} \tag{2}$$

is a valid inequality for $P_I(G)$.

2. Prove that (2) defines a facet of $P_I(C)$.

Exercise 4 (Finding odd cycle inequalities) 0.5+2 points

Given an undirected graph $G = (V, E)$, we can, in principle, strengthen the LP relaxation of (1) by adding *all* odd cycle inequalities to (1). In general, this approach is intractable though, because there might be exponentially many odd cycles in G. Therefore, one usually does not add all odd cycles to (1), but only those being "useful" within the following procedure:

Start with the formulation $P(G)$, i.e., without odd cycle inequalities, and use it within branch-and-bound to find a maximum stable set. The branch-and-bound algorithm is then enhanced as follows. Whenever a solution $x^\star$ of an LP relaxation is computed, one checks whether there exists *some* odd cycle inequality that is violated by $x^\star$. If a violated inequality exists, it is added to the model to strengthen the formulation.

This causes the next challenge: how to identify violated odd cycle inequalities?

Develop an algorithm that runs in polynomial time (polynomial in $|V|$ and $|E|$) that receives the following input and generates the following output:

Input an undirected graph $G = (V, E)$, a point $x^\star \in P(G)$;

Output an odd cycle $C = (V', E')$ in G for which $\sum_{v \in V'} x^\star_v - \frac{|V'|-1}{2}$ is as large as possible, or the correct statement that there does not exist any odd cycle in G.

In particular,

1. explain why such an algorithm allows to identify a violated odd cycle inequality;
2. prove correctness of your algorithm and explain why it runs in polynomial time.

Hint: It might be useful to derive an auxiliary graph such that odd cycles in G correspond to paths with an even number of nodes in the auxiliary graph.

Exercise 5 (A branch-and-cut algorithm) 3 points

Implement the algorithm developed in the previous exercise in Gurobi. To this end, make use of a so-called callback.

In a callback, users can, among others, implement separation routines. Gurobi will then call the callbacks at appropriate points of time and execute the separation routines. If these routines generate a violated inequality, Gurobi adds the inequality to the problem formulation. The aim of this exercise is to write such a separation callback for Gurobi. Your callback should be able to separate arbitrary points, i.e., your callback should be implemented in such a way that Gurobi can separate arbitrary solutions of an LP relaxation and does not rely on integrality.

Implement your separation routine as a Gurobi callback. More information on callbacks can be found at https://www.gurobi.com/documentation/10.0/refman/cb_codes.html and https://www.gurobi.com/documentation/10.0/refman/py_model_cbcut.html.

Important: If you could not find an efficient algorithm in the previous exercise, you are allowed to use an alternative algorithm. In this case, model the problem of finding a violated odd cycle inequality as a mixed-integer linear program that you solve with Gurobi. Note, however, that deriving this model does not compensate for missing points in the previous exercise.

Test Instances

We provided a template file on CANVAS as well as a few test instances. The template file already contains a routine for reading an instance of our problem in the section "Auxiliary Functions". Extend this template file by your own routines and optimization models. Please put your own methods in the corresponding sections (Auxiliary Functions, Exercise 1, Exercise 2,...) of this file and submit your solution via CANVAS.

To run the code using the template file, you have to specify a file containing data, see CANVAS for exemplary data; depending on your operating system, a call of the template might look like this

```
python3 template_separation.py /path/to/datafile
```

The format of the data files is as follows. Each line contains two integral numbers encoding the end points of an edge.

Note that the template changes some of Gurobi's default parameter values, because Gurobi is able to detect (some) odd cycles on its own. To measure the effect of your separation routines, Gurobi's built-in methods need to be disabled.

Submission and Grading

To submit your solution on CANVAS,

- for the practical exercises, extend the provided template file by adding your own methods and data structures to the corresponding sections of the template;
- for the theoretical exercises, prepare a PDF file containing your solution;
- work in groups of up to 3 students, list all group members in the header section of the template file and your PDF;
- submit your solution until June 4, 20:00.

To achieve a top score, your solution has to satisfy the following criteria:

- Answers to all exercises are provided;
- The implementation and answers to theoretical questions are correct;
- The implementation of the algorithm in Exercise 5 makes use of a Gurobi callback.