Summer term 2026
Prof. Dr. S. Held
Dr. U. Brenner

Research Institute for Discrete Mathematics
University of Bonn

Linear and Integer Optimization
Programming Exercise 3

Implement the branch-&-bound algorithm for the MAXIMUM-WEIGHT-STABLE-SET-PROBLEM that is defined as follows. Given a graph G and weights on the vertices $\alpha : V(G) \to \mathbb{N}$, we are looking for a stable set $S \subseteq V(G)$ of maximum weight $\sum_{v\in S} \alpha(v)$. It should be modeled by the following ILP:

$$\max \sum_{v\in V(G)} \alpha(v)x_v \tag{1}$$

$$s.d. \quad x_v + x_w \leq 1 \qquad \forall \{v,w\} \in E(G) \tag{2}$$

$$x_v \in \{0,1\} \qquad \forall v \in V(G) \tag{3}$$

As an LP solver you must use the academically free program **QSopt** through the API in `lp.h` that is available on the web-site. To make the implementation easier, you find a program that

- Reads an instance,
- creates the above LP-relaxation using the API in `lp.h`,
- solves it and prints the solution vector to the console.

(see `mss.zip` in `https://uni-bonn.sciebo.de/s/TNkJyPidkG67Bdw`). The ZIP file contains also test instances. **Read the README file for further information!**. For compiling type 'make' in the extracted directory 'mss'. If you encounter problems building mss, do not hesitate to contact me: held@dm.uni-bonn.de.
Note that the matrix A is usually sparse, i.e. most coefficients are zero. Thus, in `lp.h` new rows/constraints are always defined by their non-zero entries. The corresponding functions in `lp.h` are commented and their use becomes clear in mss.c.
As shown in the lecture, the LP relaxation has a large integrality gap, which is problematic for branch-&-bound. Thus you should first try to tighten the gap in the root LP, by adding clique inequalities and inequalities of Exercise 9.4. To this end you should implement a simple greedy algorithms. E.g. for cliques you can start with $C = \{v\}$ for a $v \in V(G)$ and add vertices $w \in V(G) \setminus C, C \subseteq \Gamma(w)$, with a large value x_w to C.
This should be started iteratively with different vertices $v \in V$ until all vertices are part of some (inclusion-wise) maximal clique. You should iterate the two steps

- solving the root lp and
- adding clique inequalities and circuit inequalities from Exercise 10.4

until no more clique inequalities are found before starting branch-&-bound. The algorithm should write

1. the value of the root LP without clique inequalities,
2. the value of the root LP with clique inequalities, and
3. the value and vertex indices of a maximum-weight stable set S

to the console.

The program has to be implemented in C++. Your submission must include all source files and a command that can be used to compile the program. The program will be invoked as `./branch_and_bound input`. Your code must compile cleanly with the flags `-Wall -Wextra -Wpedantic -Werror -std=c++20` using `GCC 12+` or `Clang 15+`. Note that compiling **QSopt** creates warnings but your code may not add warnings. No third-party libraries beyond the C++ standard library are permitted.

For this programming exercise, you can get 30 points.

Due date: Tuesday, July 14, 16:00 s.t. The submission is done via the eCampus page of your exercise group.

Web page of the lecture: `https://www.or.uni-bonn.de/lectures/ss26/lgo_ss26.html`