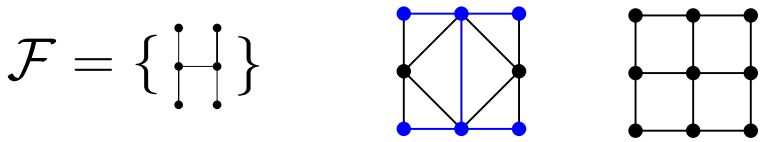
Forbidding subdivisions and line graphs towards faster algorithms

Jana Novotná

Imagine, at first, a map of cities and roads between them, houses connected by powerlines, an internet network, or conflicts between jobs that cannot be assigned to the same timeslot. From the mathematical point of view, we care only about the abstract structure behind the real-world setting, which describes the relationship involving pairs of objects. We call such a structure a *graph*, its objects *vertices* and connections between pairs of objects *edges*. Take the example with conflicts: each job is a vertex of our graph, and an edge between two jobs represents that the jobs cannot be assigned to the same timeslot.

One of the earliest studied and most center graph problems is the Graph Coloring problem that asks to label the vertices of a graph with the smallest possible number of *colors* so that no two neighboring vertices (those connected by an edge) are identically colored. The problem itself and its numerous modifications have many applications, including job scheduling. In our example, each color represents a set of conflict-free jobs, so we can assign them to a single time slot. Then a solution with the minimum number of colors gives us a schedule with the minimum number of time slots needed to perform all the jobs. However, in general, the problem is NP-complete, meaning that there is no efficient algorithm that solves the problem optimally (under the famous conjecture P $\neq$ NP). Moreover, if we consider, at first glance, an easier question of the 3-Coloring problem where we restrict ourselves to using only three colors, the problem stays NP-complete. On the positive side, many computationally hard graph problems become tractable when the input is restricted to a specific graph class (as we can use additional properties of the input to design an algorithm). The central question is: For which graph classes does a graph problem become tractable and for which graph classes is it still computationally hard?

In our project, we focus on graph classes that are characterized by forbidden patterns – they do not contain any "pattern" from a specific set $\mathcal{F}$ (not necessarily finite). In other words, we can never obtain any pattern from $\mathcal{F}$ by removing vertices from our graph. We call such graphs $\mathcal{F}$-free. Follow the example below: On the left, we have a specific set of forbidden "patterns" (here, a graph of shape "H"). The graph in the middle is not $\mathcal{F}$-free, but the graph on the right is as it contains "additional" edges.



In this way, many well-known graph classes can be characterized. The approach via forbidden patterns has been given significant attention in recent years with the goal to obtain more efficient algorithms or new intractability results. Particular emphasis has been given to forbidding a single, a few, or well-structured patterns.

In our project, we give particular attention to graph classes characterized by an infinite family of forbidden patterns, however, derived from only one pattern by two basic operations: taking a subdivision (replace single edges by arbitrarily long paths) and its line graph (informally, it represents the adjacences between the edges of the original graph).

The project's underlying goal is to identify certain forbidden patterns that give a nice specific structure and, in particular, enable more efficient algorithms in the corresponding graph classes. We aim to approach the goal from two different perspectives. First, for some specific patterns, obtain structural properties that are strong enough to bring efficient algorithms for many problems at once. Second, we consider single graph problems, and even with less structure, we believe we will be able to get efficient algorithms there, with the main emphasis on the 3-Coloring problem.