Hitting paths in graphs

Radovan Červený | Ondřej Suchý (supervisor)

FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

Problem definition

- The *5-Path Vertex Cover*, *5-PVC* problem: Given a graph G, find a subset F of its vertices such that each path on 5 vertices in G has at least one vertex in F.

e.g.: Given a map of cities (vertices) connected with roads (edges), determine in which cities we need to build a charging station to ensure that when travelling through 5 cities (a path of 5 vertices) there is a charging station in at least one of those cities.

- This problem is computationally very hard (NP-complete), so we further *parameterize* the problem by the *size of the solution* k, i.e., we want the subset F to contain at most k vertices.

Problem motivation

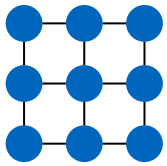
- The problem is motivated by the design of *secure wireless communication protocols* or in route planning and speeding up shortest path queries in graphs.

e.g.: In a sensor network, it is typically very costly or even impossible to secure all the sensors (protect them from an adversary), thus we want to protect only a convenient subset of the sensors.

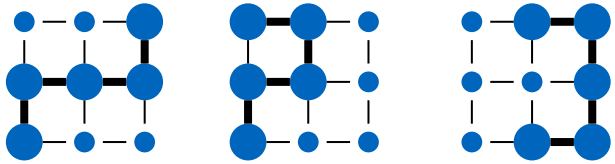
Our contribution

- Specifically for 5-PVC, only a trivial branching algorithm with $O(5^k n^{O(1)})$ running time was previously known.
- However, there exists an algorithm which involves a reduction from the 5-PVC to the 5-Hitting Set problem and achieves $O(4.0755^k n^{O(1)})$ running time.
- We created an algorithm that solves the 5-PVC in $O(4^k n^{O(1)})$ running time, thus giving a new upper bound for the 5-PVC problem.

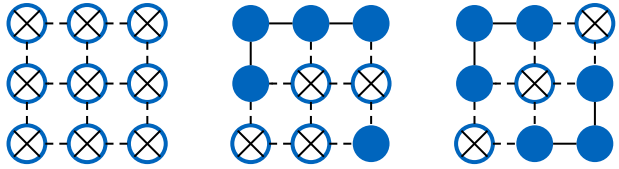
The input graph G ....



... has many paths on 5 vertices ...



... and many possible solutions F.



(crossed vertices are in the solution)

Our algorithm

- We used a general technique called *iterative compression*, which starts with an empty solution and builds it by adding vertices one by one to it. When the current solution becomes too big, it uses a so called *compression routine*, which finds a smaller solution or proves that no smaller solution exists.

Compression routine

- We designed the compression routine as a branching procedure which uses a ordered set of rules and repeatedly does the following: If the graph contains a path on 5 vertices, find the first rule that can be applied and apply it.
- There are two types of rules: *reduction* rules simplify the current problem instance, and *branching* rules that make at least two recursive calls to our procedure.
- We designed *50 rules* to deal with the problem. The key idea is that if there still is a path on 5 vertices in the graph, then *there is always at least one rule that can be applied*. Together with the proofs of correctness of each rule, the proof of this idea constitutes the main body of our work.