Shortest Disjoint Paths on a Grid

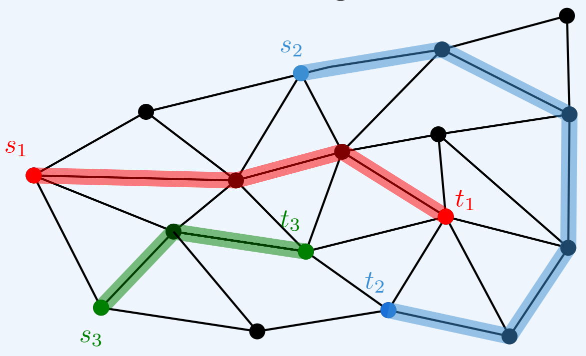
Mathieu Mari U. Montpellier France
Anish Mukherjee U. Warwick UK
Michał Pilipczuk U. Warsaw Poland
Piotr Sankowski U. Warsaw IDEAS-NCBR Poland

Context.

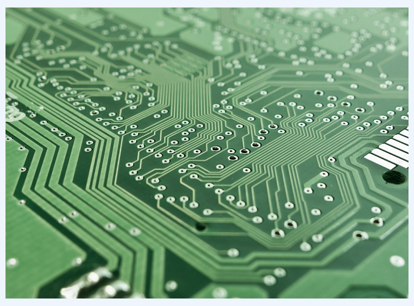
The (classic) Disjoint Paths Problem.

Input: a graph G, and a set T of $2k$ vertices $s_1, t_1, \dots, s_k, t_k$ called *terminals*.
Goal: Find k paths connecting every s_i to t_i that are pairwise vertex-disjoint, if such paths exist.

- actively studied for almost 50 years already
- central application : VLSI design
- NP-complete, even on an infinite grid [Kramer, Leeuwen, 84]
- there is an FPT-algorithm [Graphs Minors XIII, Robertson, Seymour, 95]



a routing



A Printed circuit board can be considered as a rectangular **grid** graph.

The **shortest** Disjoint Paths Problem.

= min-sum

Input: a graph G, and a set of $2k$ vertices $s_1, t_1, \dots, s_k, t_k$ called *terminals*.
Goal: Find k paths connecting every s_i to t_i that are pairwise vertex-disjoint, and such that **the sum of the lengths of the paths is minimized**, if such paths exist.

Motivation

- natural optimisation version
- relevant for applications to VLSI design
- Complexity poorly understood

What is known?

- $k = 3$: NP-hard? polytime?
- Randomized polytime algo for $k = 2$. [Björklund, Husfeldt, 19]
- W[1]-hard (and XP) when we ask each path to be a shortest path. [Lochet, 21]
- $O(kn \log n)$-time algo if the graph is planar, sources are on one face, and sinks are in another face. [Colin de Verdière, Schrijver, 11]
- few other results that require some restriction in the placement of the terminals.

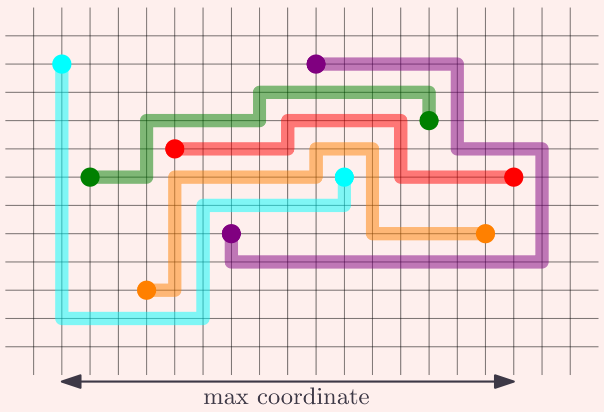
Our contribution.

Shortest Disjoint paths on a **grid**

Input: a set T of $2k$ terminals with integral coordinates.
Output: the length of a shortest routing of T on the grid.

Theorem. There exists an algorithm that solves this problem in time $2^{2^{O(k)}} O([T])$.

- our algorithm additionnally gives the paths.
- no prior, even XP-algorithm was known in any setting without restrictions on the placement of the terminals



$[T] \leq 4k \log_2(\text{max coordinate})$ is the number of bits to encore the terminals' coordinates.

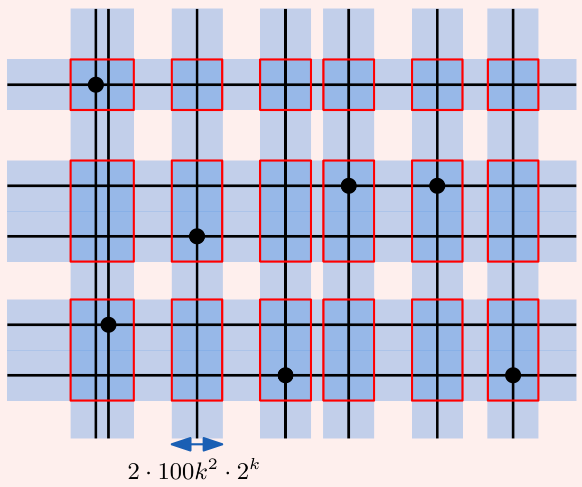
Plan of the proof. We assume that there is exists a routing of T.

Structural Lemma. We show that there exists an optimal routing that is "well-structured"

Algorithm. We enumerate all $2^{2^{O(k)}}$-many well-structured routings, and return one of minimum length.

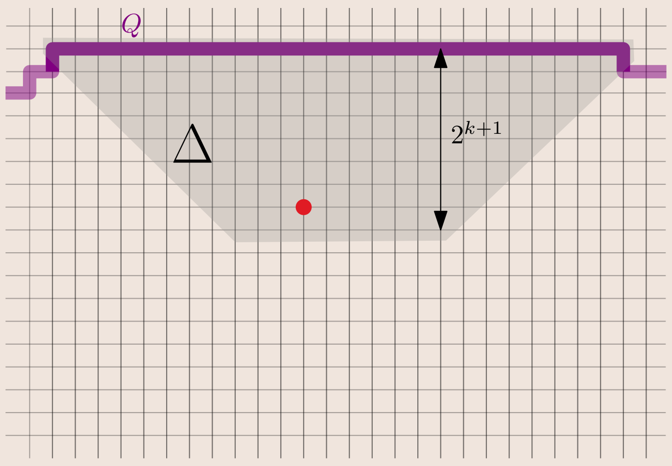
Structural Lemma.

If there is a routing for T, then there exists an shortest routing such that for every corner c, there exist terminals $t_1, t_2 \in T$ such that $|x(c) - x(t_1)| < 100k^2 \cdot 2^k$ and $|y(c) - y(t_2)| < 100k^2 \cdot 2^k$

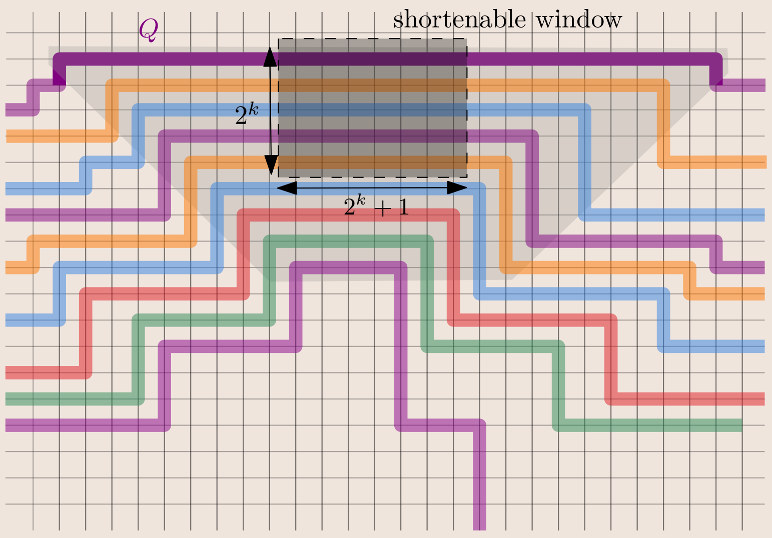


Idea of the proof.

In a shortest routing, every "U-turn" is close to a terminal.

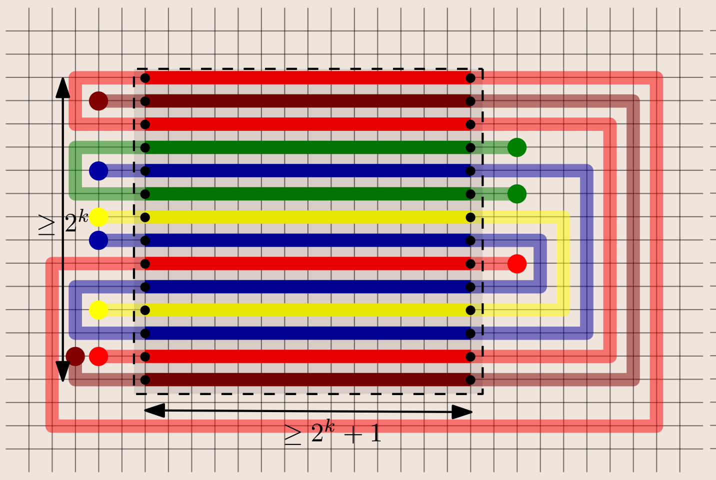


Otherwise, we can find a shortenable window...

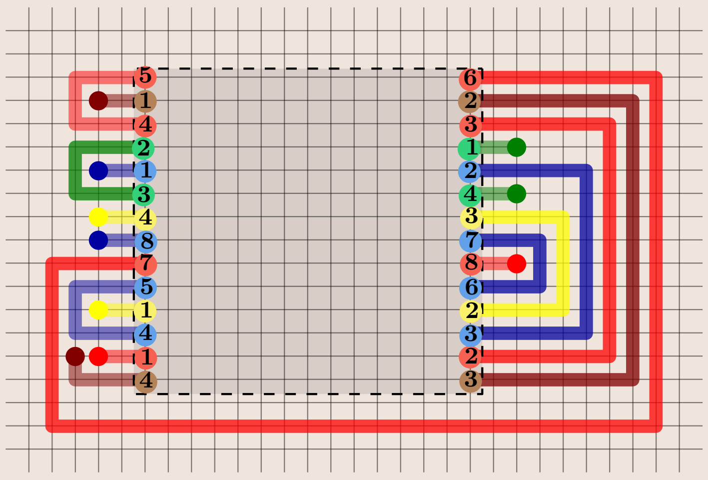


...that we can use to shorten the routing:

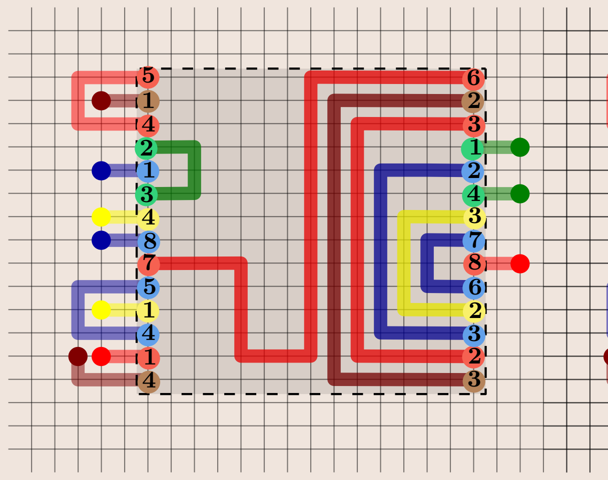
1. Find an "even" sub-window, that each path crosses an even number of times.



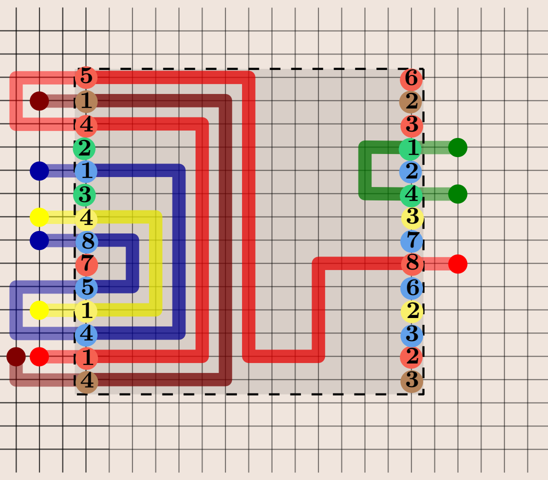
2. Remove the inside part and order the intersection points.



3. Apply a out-to-in continuous bijection to the $(4i + 2, 4i + 3)$-subpaths.



4. Apply a vertical symmetry, and remove useless parts.



Open questions

- Improve the dependence on k?
- Extension to a grid with holes, or a two-layer grid?
- A polytime algorithm for $k = 3$? open already for planar graphs.
- A deterministic and combinatorial polytime algorithm for planar graphs and $k = 2$?
- Approximation algorithms?

the paper