The k-disjoint paths problem in directed planar graphs

Dániel Marx[1]

[1]Computer and Automation Research Institute,
Hungarian Academy of Sciences (MTA SZTAKI)
Budapest, Hungary

(Joint work with Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk)

Dagstuhl Seminar 13121
March 21, 2013
Schloss Dagstuhl, Germany

Main result

Result of Schrijver:

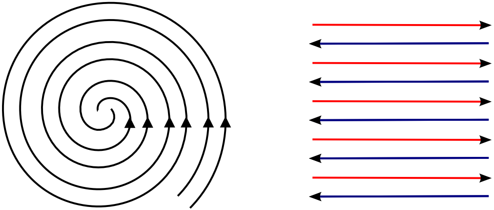
A $n^{O(k)}$ time algorithm for the k-vertex-disjoint paths problem in directed planar graphs.

New result:

A $f(k) \cdot n^{O(1)}$ time algorithm for the k-vertex-disjoint paths problem in directed planar graphs.

Overview

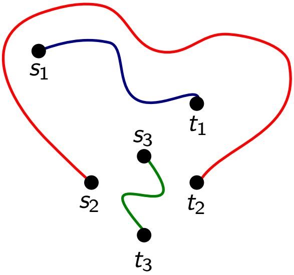
1. Undirected planar graphs.
2. Directed planar graphs: Schrijver's Algorithm.
3. Directed planar graphs: new algorithm.



Undirected graphs

k-disjoint paths problem

Given a graph G and pairs $(s_1, t_1), \ldots, (s_k, t_k)$, find k pairwise vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i and t_i.



Theorem [Robertson and Seymour GMXIII]

The k-disjoint paths problem can be solved in time $f(k) \cdot n^3$.

Undirected planar graphs

An algorithm for the special case of planar graphs appears already in [Robertson and Seymour GMVII]. A self-contained presentation:

Theorem [Adler et al. 2011]

The k-disjoint paths problem on undirected planar graphs can be solved in time $2^{2^{O(k)}} \cdot n^{O(1)}$.

Undirected planar graphs

An algorithm for the special case of planar graphs appears already in [Robertson and Seymour GMVII]. A self-contained presentation:

Theorem [Adler et al. 2011]

The k-disjoint paths problem on undirected planar graphs can be solved in time $2^{2^{O(k)}} \cdot n^{O(1)}$.

Main argument:

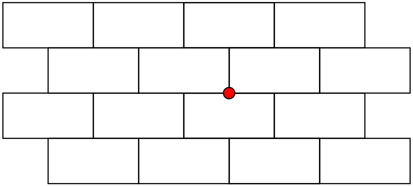
- either treewidth is $2^{O(k)}$ and we can use standard algorithmic techniques of bounded treewidth graphs, or
- treewidth is $2^{\Omega(k)}$ and we can find an **irrelevant vertex** whose deletion does not change the problem.

Irrelevant vertices

A vertex is **irrelevant** if its deletion does not change the problem, i.e., does not make it harder.

Theorem

If treewidth of a planar graph is $\Omega(k)$, then it contains the subdivision of a $k \times k$ wall.

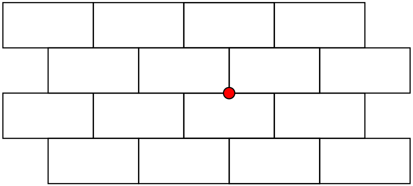


Irrelevant vertices

A vertex is **irrelevant** if its deletion does not change the problem, i.e., does not make it harder.

Theorem

If treewidth of a planar graph is $\Omega(k)$, then it contains the subdivision of a $k \times k$ wall.



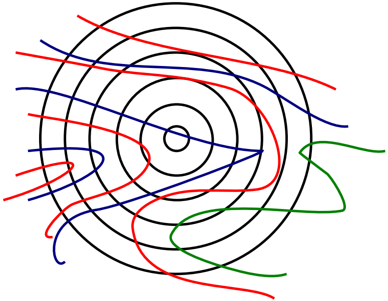
Lemma [Adler et al. 2011]

If a $2^{O(k)} \times 2^{O(k)}$ wall of a planar graph does not enclose any terminals, then the middle vertex of the wall is irrelevant to the k-disjoint paths problem.

Irrelevant vertices

Lemma [Adler et al. 2011]

If there are $2^{O(k)}$ concentric cycles in a planar graph not enclosing any terminals, then the innermost cycle is irrelevant to the k-disjoint paths problem.



Any solution can be rerouted to avoid the innermost cycle.

Undirected planar graphs

Algorithm:

- If treewidth is $2^{\Omega(k)}$, we can find an irrelevant vertex.
- By repeatedly removing irrelevant vertices, we can reduce treewidth to $2^{O(k)}$.
- If treewidth is $2^{O(k)}$, standard algorithmic techniques can be used.

Running time is $2^{2^{O(k)}} \cdot n^{O(1)}$.

Undirected planar graphs

Algorithm:

- If treewidth is $2^{\Omega(k)}$, we can find an irrelevant vertex.
- By repeatedly removing irrelevant vertices, we can reduce treewidth to $2^{O(k)}$.
- If treewidth is $2^{O(k)}$, standard algorithmic techniques can be used.

Running time is $2^{2^{O(k)}} \cdot n^{O(1)}$.

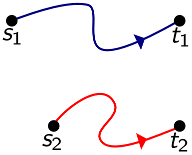
Note: [Adler et al. 2011] show that there are instances with treewidth $2^{\Omega(k)}$ and no irrelevant vertex, so double-exponential dependence on k cannot be avoided with this approach.

Directed graphs

There is no analog of [Robertson and Seymour GMXIII] on directed graphs:

Theorem [Fortune, Hopcroft, and Wyllie 1980]

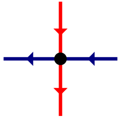
The directed 2-disjoint paths problem is NP-hard.



As the directed problem is hard in general, it can be important to distinguish between slightly different versions of the problem.

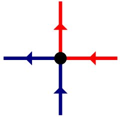
Different planar versions

- Edge-disjoint planar

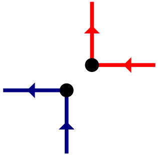


  [Open: Is the planar directed edge-disjoint problem NP-hard for $k = 2$?]

- Noncrossing edge-disjoint planar



- Vertex-disjoint planar



  [More general than the noncrossing edge-disjoint planar problem]

Planar graphs

Fact

Polynomial-time greedy algorithm if all the terminals are on a single face.

Theorem [Schrijver 1994]

The k-disjoint paths problem in directed planar graphs can be solved in time $n^{O(k)}$.

New result

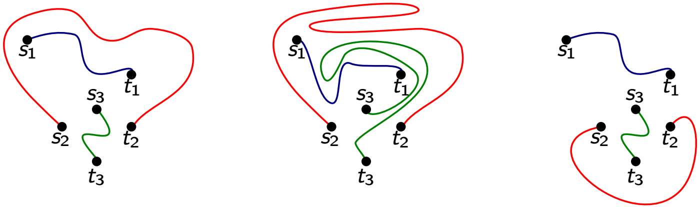
The k-disjoint paths problem in directed planar graphs can be solved in time $f(k) \cdot n^{O(1)}$.

Schrijver's result

Main idea

Guess the homology type of the solution and try to realize it.

Informally, two solutions are homologous if they can be "continuously transformed" into each other.



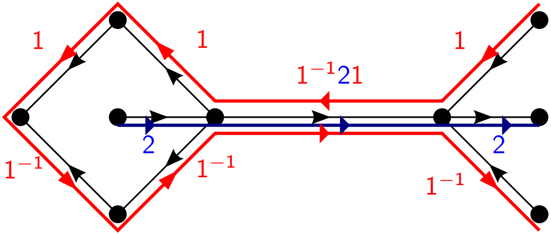
Flows

Flow

Informally: paths are allowed to share edges without crossing and to go in the wrong direction on an edge.

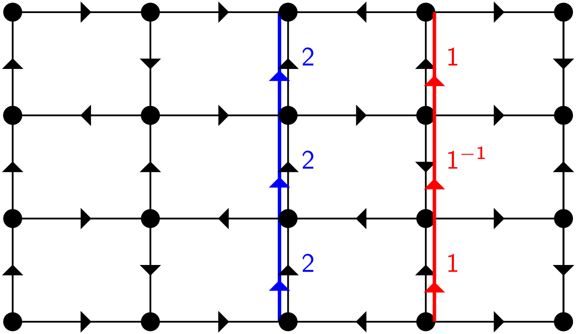
Formally:

- a word with letters from $1, 2, \ldots, k, 1^{-1}, 2^{-1}, \ldots, k^{-1}$ (or the empty word ϵ) on each edge,
- flow conservation and noncrossing conditions hold at each vertex.



Homology types

Two flows f and g are **homologous** if there is a word $w(F)$ for each face F such that $w(F)^{-1} \cdot f(a) \cdot w(F') = g(a)$ for each edge a, where F and F' are the left-hand and right-hand side of a, respectively.

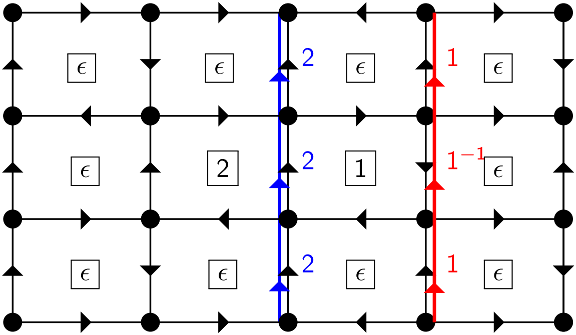


Lemma [Schrijver]

Given a flow f, we can check in polynomial time if there is a flow g homologous to f such that $g(a) \in \{1, 2, \ldots, k, \epsilon\}$ for every edge a.

Homology types

Two flows f and g are **homologous** if there is a word $w(F)$ for each face F such that $w(F)^{-1} \cdot f(a) \cdot w(F') = g(a)$ for each edge a, where F and F' are the left-hand and right-hand side of a, respectively.

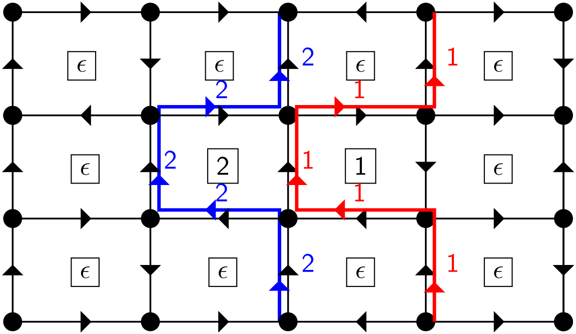


Lemma [Schrijver]

Given a flow f, we can check in polynomial time if there is a flow g homologous to f such that $g(a) \in \{1, 2, \ldots, k, \epsilon\}$ for every edge a.

Homology types

Two flows f and g are **homologous** if there is a word $w(F)$ for each face F such that $w(F)^{-1} \cdot f(a) \cdot w(F') = g(a)$ for each edge a, where F and F' are the left-hand and right-hand side of a, respectively.

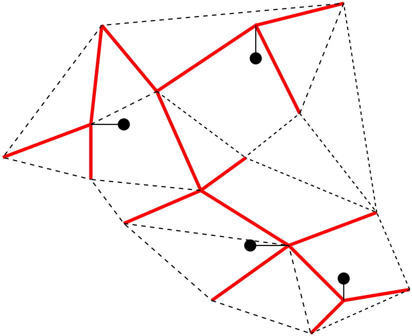


Lemma [Schrijver]

Given a flow f, we can check in polynomial time if there is a flow g homologous to f such that $g(a) \in \{1, 2, \ldots, k, \epsilon\}$ for every edge a.

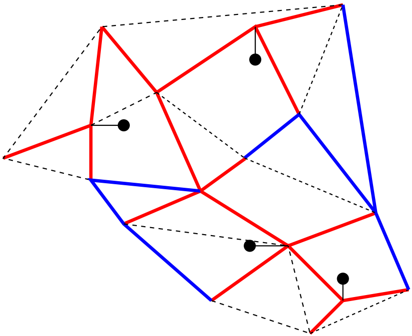
Enumerating homology types

- We may assume that every terminal has degree 1.
- Find a spanning tree of the graph minus the terminals.
- If the fundamental cycle of an edge encloses a terminal, we call it an "ear."



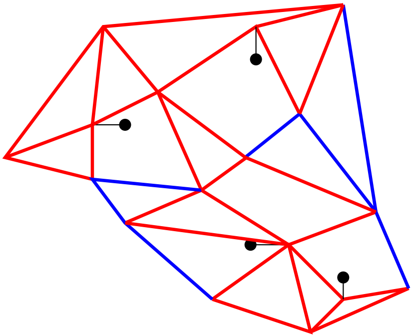
Enumerating homology types

- We may assume that every terminal has degree 1.
- Find a spanning tree of the graph minus the terminals.
- If the fundamental cycle of an edge encloses a terminal, we call it an "ear."



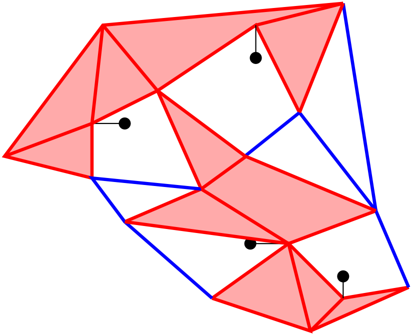
Enumerating homology types

- We may assume that every terminal has degree 1.
- Find a spanning tree of the graph minus the terminals.
- If the fundamental cycle of an edge encloses a terminal, we call it an “ear.”



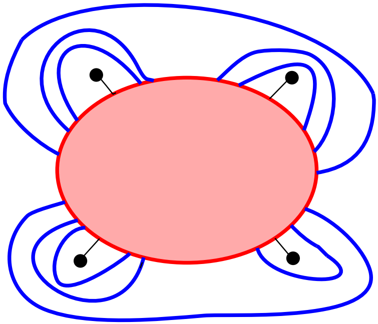
Enumerating homology types

- We may assume that every terminal has degree 1.
- Find a spanning tree of the graph minus the terminals.
- If the fundamental cycle of an edge encloses a terminal, we call it an "ear."



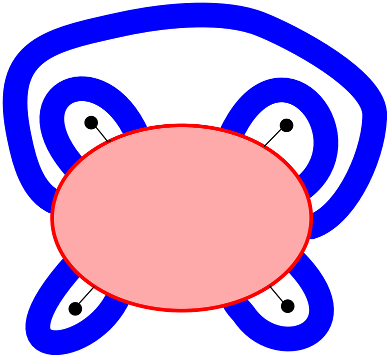
Enumerating homology types

$O(k)$ parallel classes of ears:



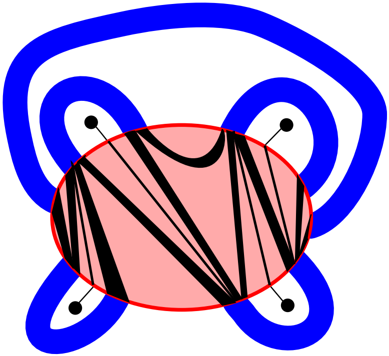
Enumerating homology types

$O(k)$ parallel classes of ears:



Enumerating homology types

$O(k)$ parallel classes of ears:



Homology type of the solution is described by

- the number of connections between any two ear classes.
- specifying which terminal is connected to which ear.

$\Rightarrow n^{O(k)}$ homology types.

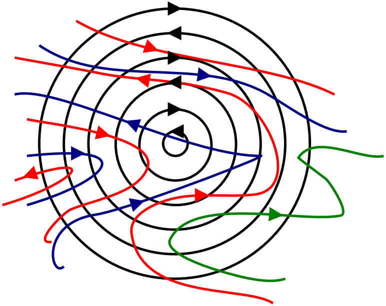
New algorithm

1. Irrelevant vertex rule.
2. Duality of alternation.
3. Decomposition.
4. Rerouting in rings.
5. Guessing the homology type.

Irrelevant vertex rule

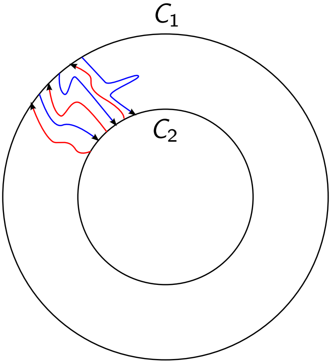
Theorem

If an alternating sequence of $f(k)$ cycles does not enclose any terminals, then the middle vertex is irrelevant.



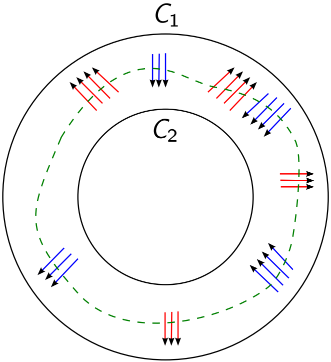
Duality theorem 1

Given two concentric cycles C_1 and C_2, either...



...there is an alternating sequence of k paths connecting C_1 and C_2 ...

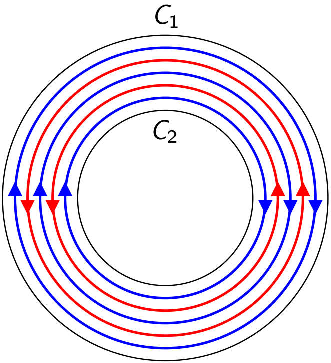
or



...there is a closed curve separating C_1 from C_2 and intersecting a sequence of edges with at most $k+O(1)$ alternations.

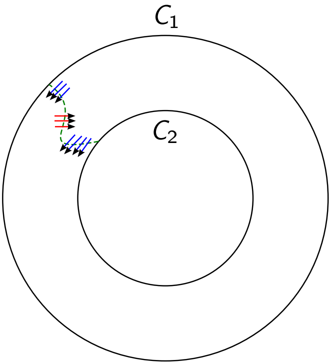
Duality theorem 2

Given two concentric cycles C_1 and C_2, either...



...there is an alternating sequence of k concentric cycles between C_1 and C_2...

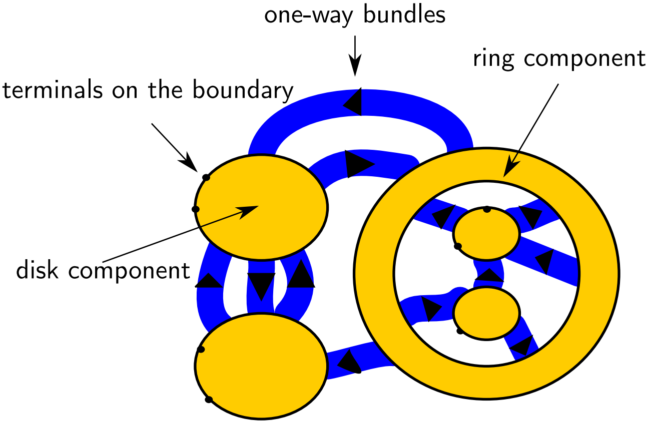
or



...there is a curve from C_1 to C_2 intersecting a sequence of edges with at most $k+O(1)$ alternations.

Decomposition

With some preprocessing, we can assume that the instance has a decomposition of the following form into $f(k)$ components and $f(k)$ connecting bundles:



Decomposition

Suppose that there is a terminal not on the outer boundary of its component.

- If there is a curve with bounded alternation to the boundary of the component, we can move the terminal to the boundary by introducing a bounded number of new bundles.

Decomposition

Suppose that there is a terminal not on the outer boundary of its component.

- If there is a curve with bounded alternation to the boundary of the component, we can move the terminal to the boundary by introducing a bounded number of new bundles.
- If there is no such curve, by duality a large sequence of alternating cycles separate the terminal from the boundary.

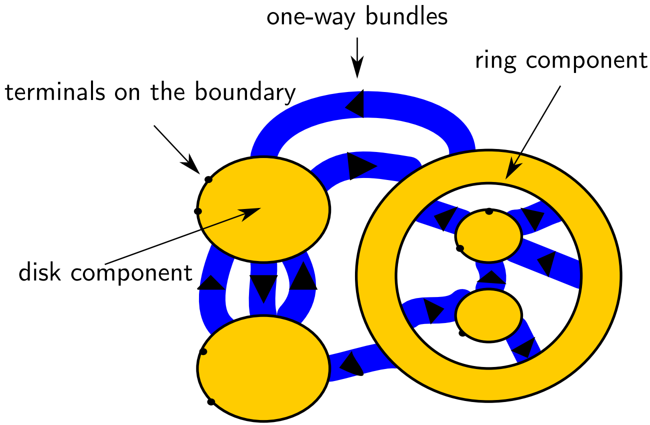
Decomposition

Suppose that there is a terminal not on the outer boundary of its component.

- If there is a curve with bounded alternation to the boundary of the component, we can move the terminal to the boundary by introducing a bounded number of new bundles.
- If there is no such curve, by duality a large sequence of alternating cycles separate the terminal from the boundary.
  - If there is a large alternating set of paths through these cycles, then we can find an irrelevant vertex.
  - Otherwise, we can find a cut of bounded alternation (creating a ring) and a curve of small alternation to this cut (moving the terminal to the boundary).

Decomposition

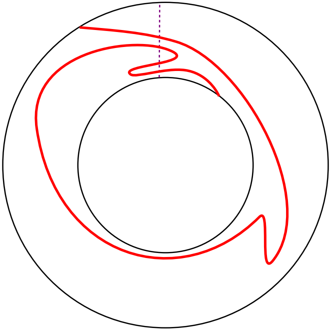
We claim that we can enumerate $f(k)$ homology types such that if there is a solution, then there is a solution with one of these types.



Rerouting in a ring

Consider the subpaths crossing a "fat" ring: the number of different homologies cannot be bounded by $f(k)$.

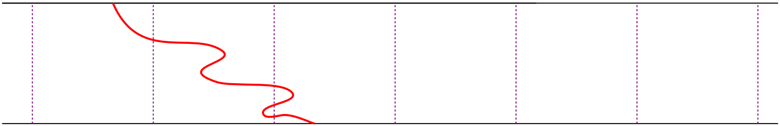
Number of turns: the (signed) number of times a path crosses a reference path connecting the inside and outside.



Rerouting in a ring

Consider the subpaths crossing a "fat" ring: the number of different homologies cannot be bounded by $f(k)$.

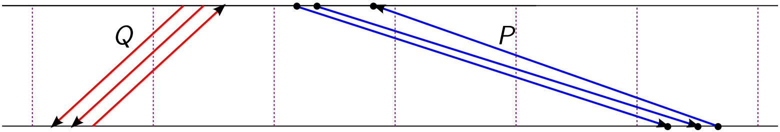
Number of turns: the (signed) number of times a path crosses a reference path connecting the inside and outside.



Rerouting in a ring

Consider the subpaths crossing a “fat” ring: the number of different homologies cannot be bounded by $f(k)$.

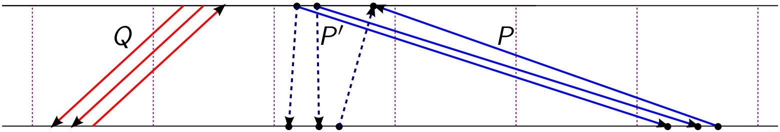
Number of turns: the (signed) number of times a path crosses a reference path connecting the inside and outside.



Rerouting in a ring

Consider the subpaths crossing a "fat" ring: the number of different homologies cannot be bounded by $f(k)$.

Number of turns: the (signed) number of times a path crosses a reference path connecting the inside and outside.



Lemma

Let P and Q be two sets of at most k paths with the same pattern. Suppose that P and Q cross a ring having $f(k)$ alternating cycles. Then P can be rerouted (without changing its endpoints) such that it does the same number of turns (maybe $\pm O(k)$) as Q.

Routing on the torus

Observation: Routing on a ring between the inside and the outside can be considered as finding disjoint cycles on the torus.

Theorem [Ding, Schrijver, Seymour 1993]

Given pairwise disjoint non-nullhomotopic curves on a torus, a sufficient and necessary condition for being able to shift the curves into pairwise disjoint cycles.

Routing on the torus

Observation: Routing on a ring between the inside and the outside can be considered as finding disjoint cycles on the torus.

Theorem [Ding, Schrijver, Seymour 1993]

Given pairwise disjoint non-nullhomotopic curves on a torus, a sufficient and necessary condition for being able to shift the curves into pairwise disjoint cycles.

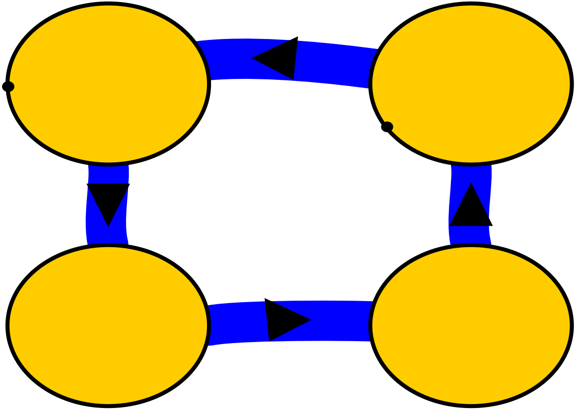
Lemma

Let P and Q be two sets of at most k paths with the same pattern. Suppose that P and Q cross a ring having $f(k)$ alternating cycles. Then P can be rerouted (without changing its endpoints) such that it does the same number of turns (maybe $\pm O(k)$) as Q.

Main idea: If P realizes the pattern with turning number x and Q realizes it with turning number Q, then a witness showing that P cannot be rerouted with turning number $(x+y)/2$ gives a contradiction.

Guessing a homology type

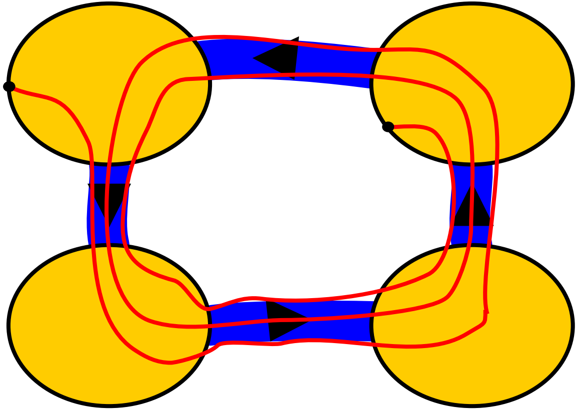
Suppose that there are no ring components:



Main problem: a path can spiral even if there are no ring components.

Guessing a homology type

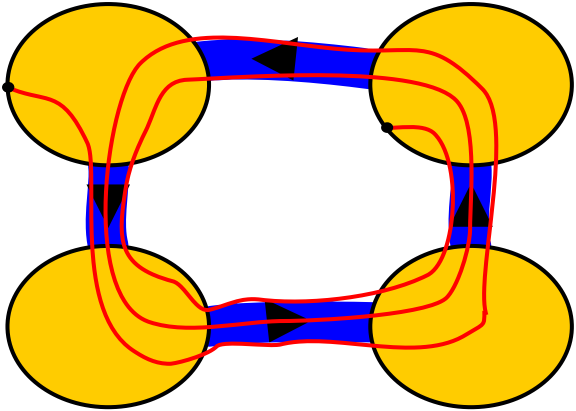
Suppose that there are no ring components:



Main problem: a path can spiral even if there are no ring components.

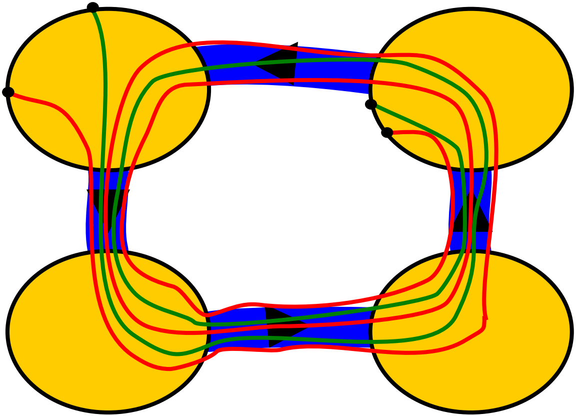
One-way spirals

Observation: if a path creates a spiral with many turns, then the other paths in between do similar spirals.



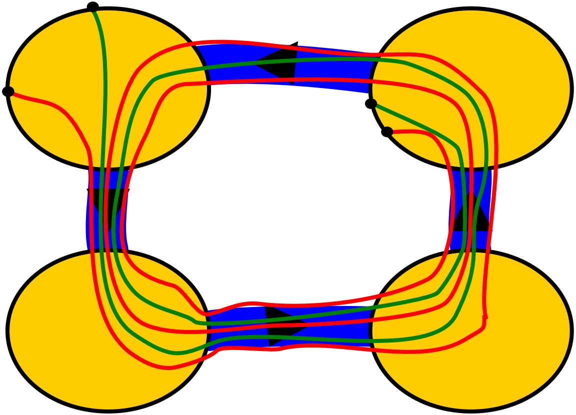
One-way spirals

Observation: if a path creates a spiral with many turns, then the other paths in between do similar spirals.



One-way spirals

Observation: if a path creates a spiral with many turns, then the other paths in between do similar spirals.



We may assume that the number of turns these i paths do is the minimum number of turns that i paths can do from the outside to the inside.

Summary of the algorithm

- Remove irrelevant vertices inside concentric cycles.
- Find a decomposition into a bounded number of components and bundles.
- Guess the number of turns in rings.
- Guess the global structure (including the structure of one-way spirals).
- Compute the number of turns for the one-way spirals.
- Determine if there is a solution with this homology type.

A note on complexity

It could have been that the $n^{O(k)}$ algorithm is best possible.

W[1]-hardness: strong evidence that there is no $f(k) \cdot n^{O(1)}$ time algorithm (similar to NP-hardness).

A note on complexity

It could have been that the $n^{O(k)}$ algorithm is best possible.

W[1]-hardness: strong evidence that there is no $f(k) \cdot n^{O(1)}$ time algorithm (similar to NP-hardness).

Example:

Theorem [Dalhaus et al. 1994]

Planar Multiterminal Cut (find the minimum number of edges pairwise separating k given terminals) can be solved in time $n^{O(k)}$.

Theorem [M. 2012]

Planar Multiterminal Cut is W[1]-hard.

A note on complexity

It could have been that the $n^{O(k)}$ algorithm is best possible.

W[1]-hardness: strong evidence that there is no $f(k) \cdot n^{O(1)}$ time algorithm (similar to NP-hardness).

Example:

Theorem [Dalhaus et al. 1994]

Planar Multiterminal Cut (find the minimum number of edges pairwise separating k given terminals) can be solved in time $n^{O(k)}$.

Theorem [M. 2012]

Planar Multiterminal Cut is W[1]-hard.

Our goal was either

- to find an $f(k) \cdot n^{O(1)}$ time algorithm or
- to show that the problem is W[1]-hard.