Multicut is FPT

Nicolas Bousquet

Joint work with: Jean Daligault, Stéphan Thomassé

FPT

A parameterized problem is *FPT* (Fixed Parameter Tractable) iff there is an algorithm which runs in time $Poly(n) \cdot f(k)$ for an instance of size n and of parameter k.

Example

Theorem

Vertex Cover parameterized by the size of the solution is FPT.

Example

Theorem

Vertex Cover parameterized by the size of the solution is FPT.

Proof :

- Pick an edge xy : x or y are in the Vertex Cover. Hence we can branch to decide which one is selected in the Vertex Cover. Decrease k by one and delete the edges adjacent to the choosen vertex.

Example

Theorem

Vertex Cover parameterized by the size of the solution is FPT.

Proof :

- Pick an edge xy : x or y are in the Vertex Cover. Hence we can branch to decide which one is selected in the Vertex Cover. Decrease k by one and delete the edges adjacent to the choosen vertex.
- Binary tree of depth k : at most 2^k branches.

Courcelle's Theorem

Theorem (Courcelle)

All the problems definable in the Monadic Second Order Logic parameterized by the treewidth are FPT.

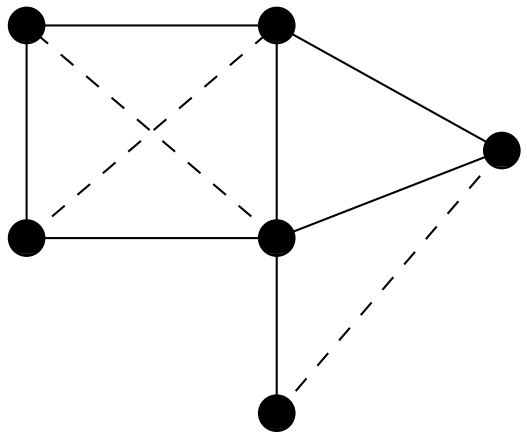
Multicut

Definition

Let $G = (V, E)$ be a graph and R be a set of pairs of vertices called *requests*. A subset E' of E is a Multicut iff for each pair $xy \in R$ there is no path from x to y in $G' = (V, E \backslash E')$.

Definition

A pair of vertices of R is called a *request*.
A vertex which is in a request is called a *terminal*.



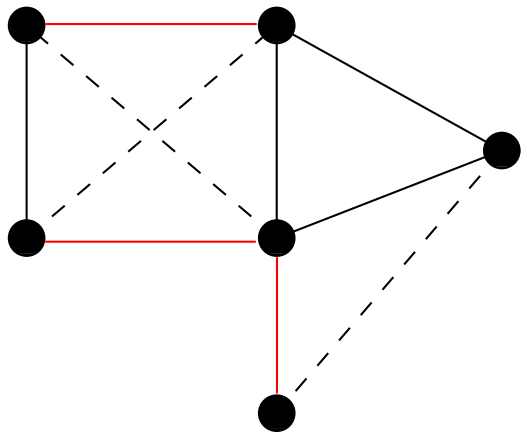
Multicut

Definition

Let $G = (V, E)$ be a graph and R be a set of pairs of vertices called *requests*. A subset E' of E is a Multicut iff for each pair $xy \in R$ there is no path from x to y in $G' = (V, E \backslash E')$.

Definition

A pair of vertices of R is called a *request*.
A vertex which is in a request is called a *terminal*.



Multicut

Multicut problem

Input : A graph G, a set of requests R, an integer k.
Output : YES iff there exists a Multicut of (G, R) of size at most k.

Multicut

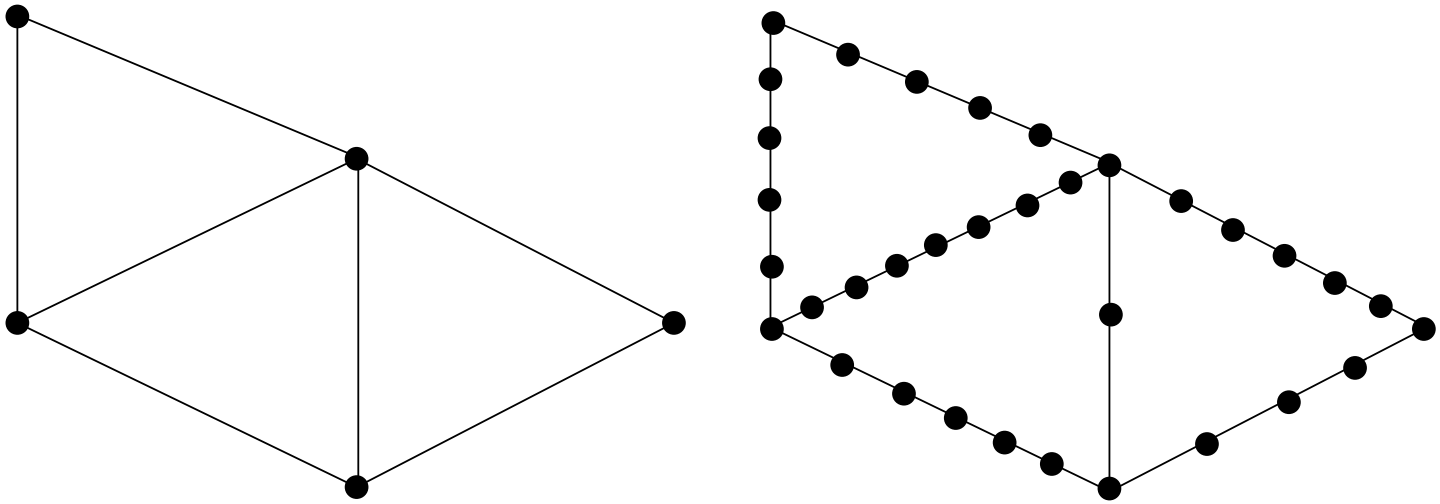
Multicut problem

Input : A graph G, a set of requests R, an integer k.
Output : YES iff there exists a Multicut of (G, R) of size at most k.

Theorem (B.,Daligault, Thomassé and Marx, Razgon)

Multicut parameterized by the size of the solution is FPT.

Subdivision of a graph



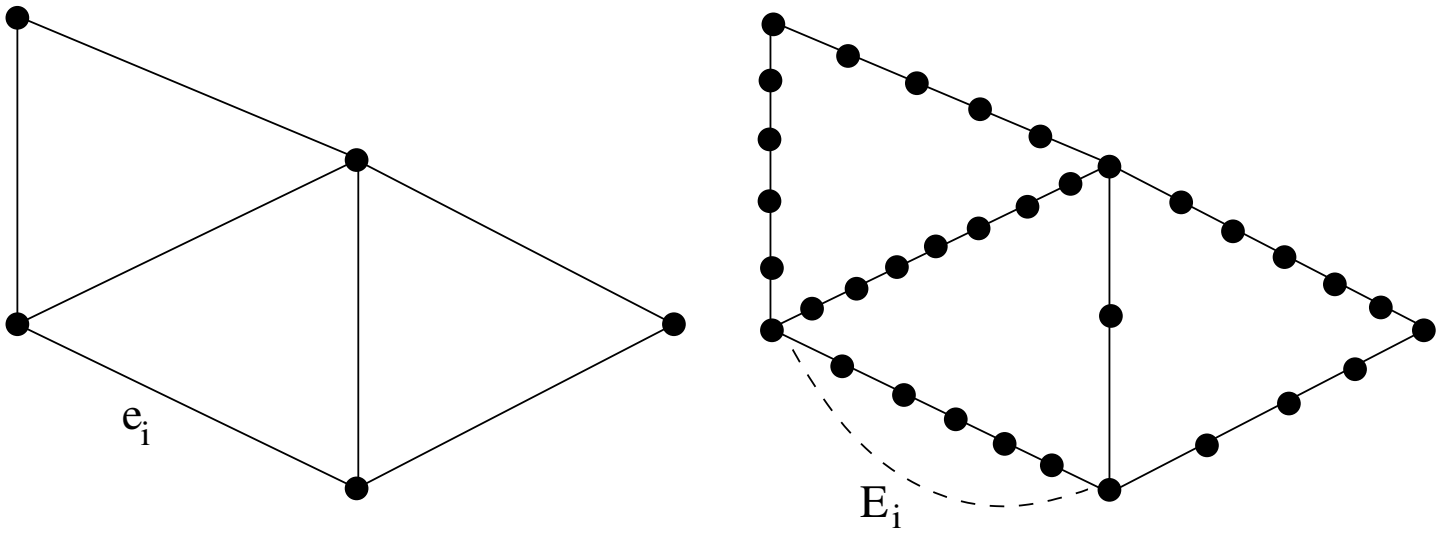
Definition

A graph H is a subdivision of a graph G iff H can be obtained by a subdivision of the edges of G.

Subdivided Multicut

Notation

We denote by E_i the set of edges of H associated to an edge e_i of G.



Subdivided Multicut

Notation

We denote by E_i the set of edges of H associated to an edge e_i of G.

Subdivided Multicut

Input : A graph H which is a subdivision of a graph G with k edges. A set of requests with endpoints which do not belong to the same E_i.

Output : YES iff there is a multicut which selects exactly one edge in each E_i.

Subdivided Multicut

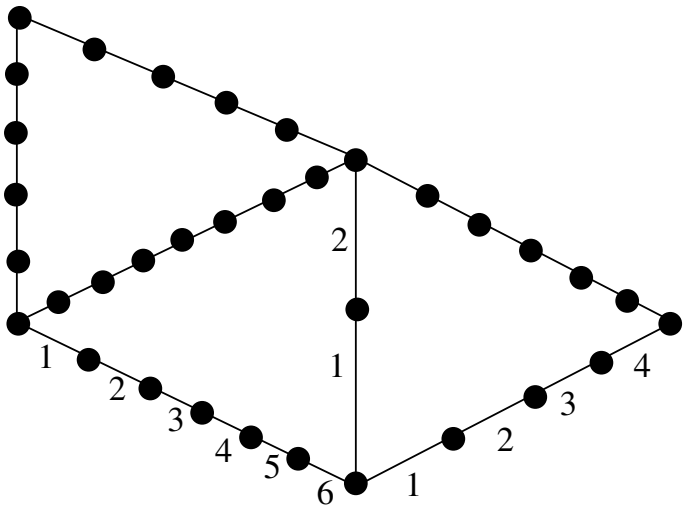
Notation

We denote by E_i the set of edges of H associated to an edge e_i of G.

Theorem

Subdivided Multicut can be decided in polynomial time.

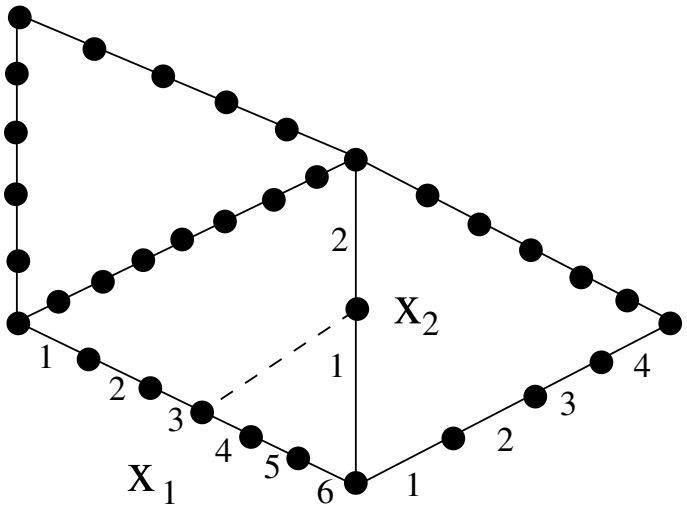
Proof



Proof :

- Number the edges of each E_i.

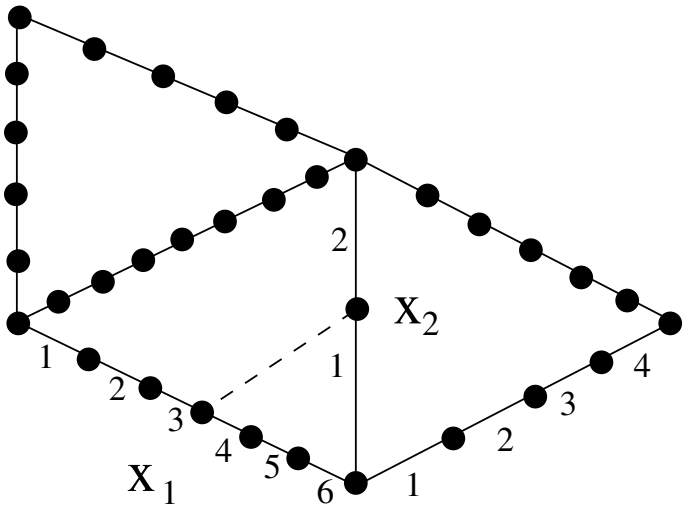
Proof



Proof :

- Number the edges of each E_i.
- Create a variable x_i for each E_i : the value of x_i refer to the edge selected by the Multicut.

Proof



Proof :

- Number the edges of each E_i.
- Create a variable x_i for each E_i : the value of x_i refer to the edge selected by the Multicut.
- Encode the requests with 2-SAT constraints : $x_1 \leq 3 \Rightarrow x_2 \leq 1$.

Tansformation into a boolean instance

Objective

Transformation into a boolean instance.

- Create boolean instances "$x_i \geq k$" for each pair (i, k).

Tansformation into a boolean instance

Objective

Transformation into a boolean instance.

- Create boolean instances "$x_i \geq k$" for each pair (i, k).
- Encode the requests in the same way : $x_1 \leq 3 \Rightarrow x_2 \leq 1$.

Tansformation into a boolean instance

Objective

Transformation into a boolean instance.

- Create boolean instances "$x_i \geq k$" for each pair (i, k).
- Encode the requests in the same way : $x_1 \leq 3 \Rightarrow x_2 \leq 1$.
- Encode the constraints of the order :
  - $x_i \geq k \Rightarrow x_i \geq k - 1$.
  - $x_i \leq k \Rightarrow x_i \leq k + 1$.

Iterative Compression

Main idea

Compute a solution of an instance using a solution of a smaller instance.

Iterative Compression

Main idea

Compute a solution of an instance using a solution of a smaller instance.

Theorem

Let P be a problem parameterized by the number of vertices k in the solution. If the problem given a solution of size $k+1$ is FPT, then the problem is FPT.

Iterative Compression

Main idea

Compute a solution of an instance using a solution of a smaller instance.

Theorem

Let P be a problem parameterized by the number of vertices k in the solution. If the problem given a solution of size $k+1$ is FPT, then the problem is FPT.

Proof : By induction on the size of the graph.

- Solve the problem on a graph of size 1 : ok.
- Solve the problem on the graph restricted to $V \backslash v$: the solution on $V \backslash v$ plus v is a solution of size $k+1$.

Iterative compression

Theorem

If Multicut given a Vertex Multicut of size at most $k+1$ can be solved in $f(k) \cdot n^c$ then Multicut can be solved in $f(k) \cdot n^{c+1}$.

Iterative compression

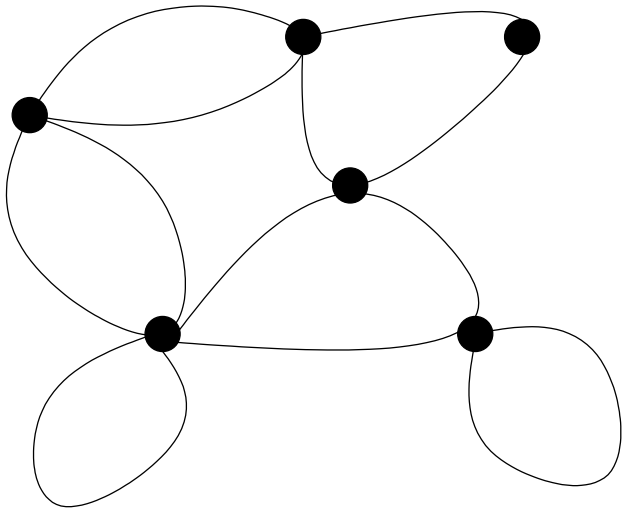
Theorem

If Multicut given a Vertex Multicut of size at most $k+1$ can be solved in $f(k) \cdot n^c$ then Multicut can be solved in $f(k) \cdot n^{c+1}$.

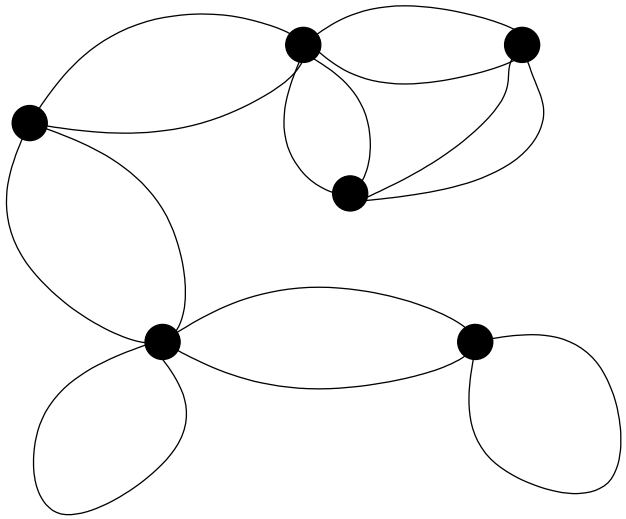
Proof by induction on the size of the graph :

- Solve the problem on the graph exept one vertex.
- One endpoint of each edge and the new vertex is a vertex multicut of size $k+1$.
- Solve the problem Multicut given a vertex multicut of size $k+1$.

Total time : $f(k) \cdot (n-1)^{c+1} + f(k) \cdot n^c \leq f(k) \cdot n^{c+1}$.



- We can assume that the vertices of the Vertex Multicut are separated by the Multicut.



- We can assume that the vertices of the Vertex Multicut are separated by the Multicut.
- We can assume that the components have one or two attachment vertices.

Left cuts

Let G be a connected graph and x be a vertex called root.

Definition

A *cut* S is a subset of vertices containing x.
The *border* Δ of a cut S is the set of edges with one endpoint in S.
We denote by δ its size.

Left cuts

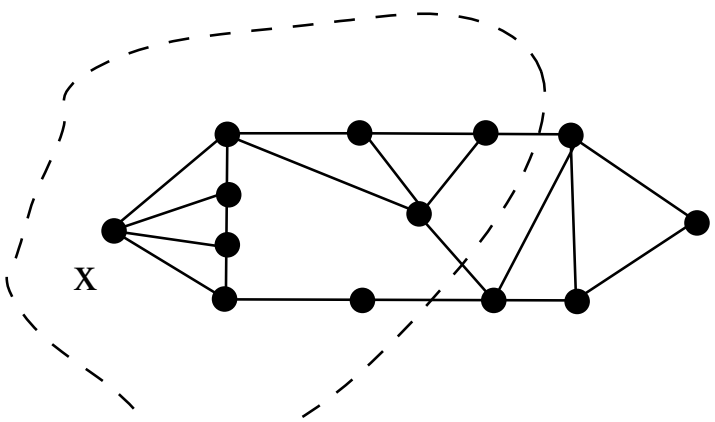
Let G be a connected graph and x be a vertex called root.

Definition

A *cut* S is a subset of vertices containing x.
The *border* Δ of a cut S is the set of edges with one endpoint in S.
We denote by δ its size.

Left cut

A *left cut* S such that if $T \subsetneq S$ then $\delta(T) > \delta(S)$.



Left cuts

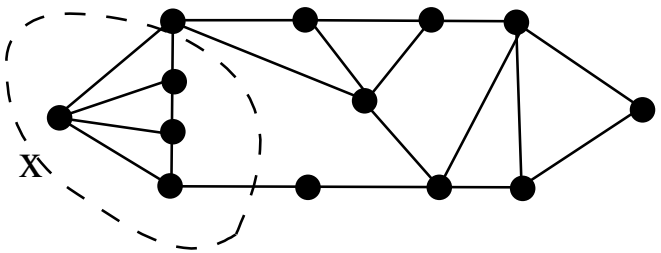
Let G be a connected graph and x be a vertex called root.

Definition

A *cut* S is a subset of vertices containing x.
The *border* Δ of a cut S is the set of edges with one endpoint in S.
We denote by δ its size.

Left cut

A *left cut* S such that if $T \subsetneq S$ then $\delta(T) > \delta(S)$.



Indivisible left cuts

Definition

A cut is *indivisible* S iff $G \backslash S$ is connected.

Theorem

Let y be a vertex. There is a bounded number (in k) of indivisible left cuts of size at most k which separate x from y.

Active sets

Active sets

An active set $\mathcal{L}$ of edges a set such that if there is a solution of the Multicut problem, there is a solution which use only edges of $\mathcal{L}$.

Theorem

Let C be a component attached on x. There is an active set $\mathcal{L}(C)$ which have a bounded size.

Active sets

Active sets

An active set $\mathcal{L}$ of edges a set such that if there is a solution of the Multicut problem, there is a solution which use only edges of $\mathcal{L}$.

Theorem

Let C be a component attached on x. There is an active set $\mathcal{L}(C)$ which have a bounded size.

Theorem

Let $C_1,...,C_p$ be p disjoint components attached on x. Let $U_i = \bigcup_{k=1}^{i} C_i$. Then each U_i has a bounded active set $\mathcal{L}_i$ and $\mathcal{L}_j \cap U_i \subseteq \mathcal{L}_i$ if $i \leq j$.

Backbone

Let C be a xy connected component. A backbone is a path from x to y in C in which only one edge of the Multicut is deleted.

We can assume that :

- Each component has a backbone.
- At most $2\lambda - 1$ edges are deleted (where λ is the xy-connectivity).
- Each vertex of the backbone is a vertex cutset.

Dilworth's Theorem

Partial Order

A *partial order* is an acyclic transitive oriented graph.
A *chain* is a total order.

Antichain

An antichain of a partial order P is a subset of P with pairwise incomparable elements.

Dilworth's Theorem

Partial Order

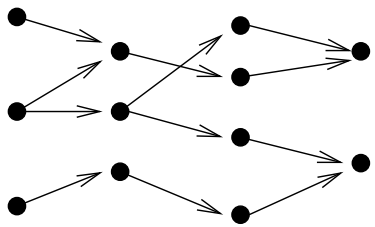
A *partial order* is an acyclic transitive oriented graph.
A *chain* is a total order.

Antichain

An antichain of a partial order P is a subset of P with pairwise incomparable elements.

Dilworth's Theorem

A partial order P can be covered by n chains iff the maximum size of an antichain is n.



Dilworth's Theorem

Partial Order

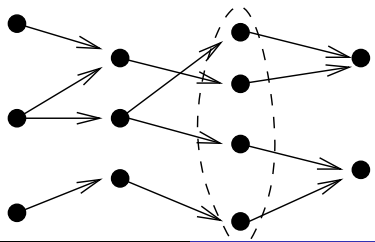
A *partial order* is an acyclic transitive oriented graph.
A *chain* is a total order.

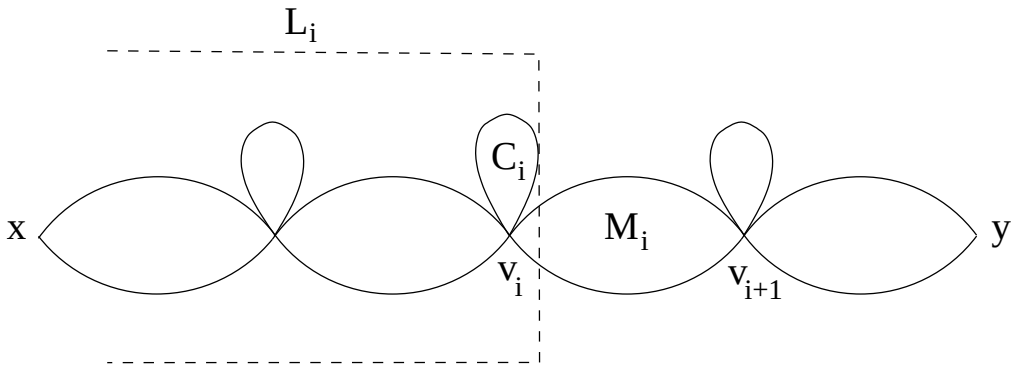
Antichain

An antichain of a partial order P is a subset of P with pairwise incomparable elements.

Dilworth's Theorem

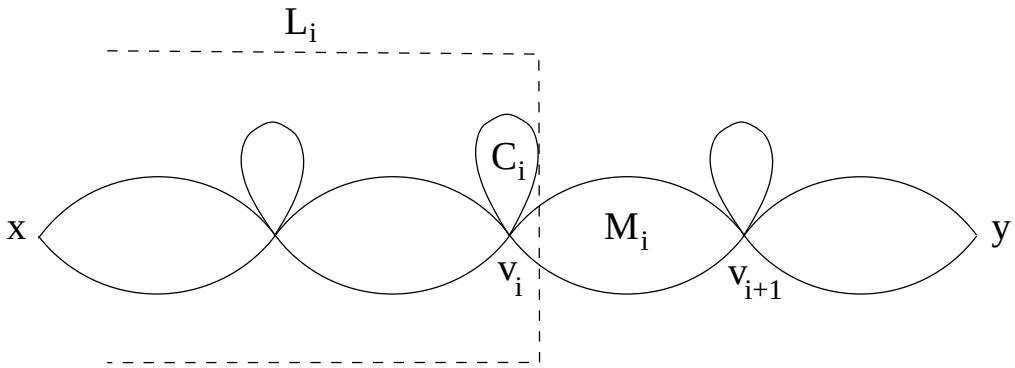
A partial order P can be covered by n chains iff the maximum size of an antichain is n.





Our goal

We want to reduce to the polynomial instance.



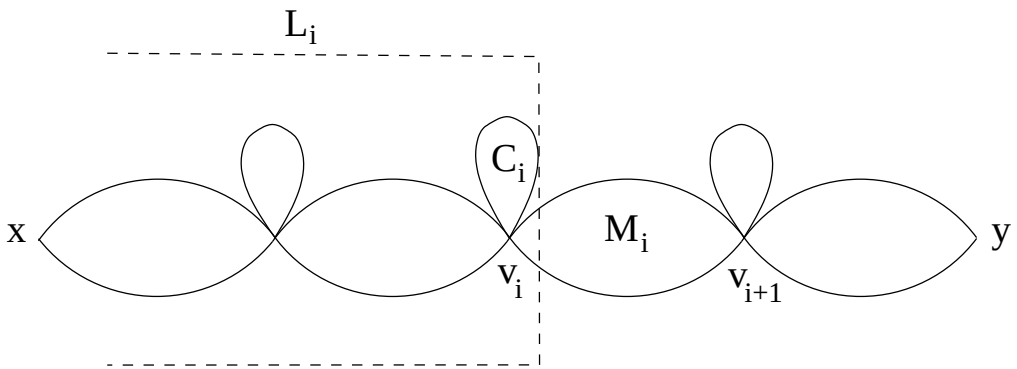
Our goal

We want to reduce to the polynomial instance.

Theorem

We can assume that there is no components C_i.

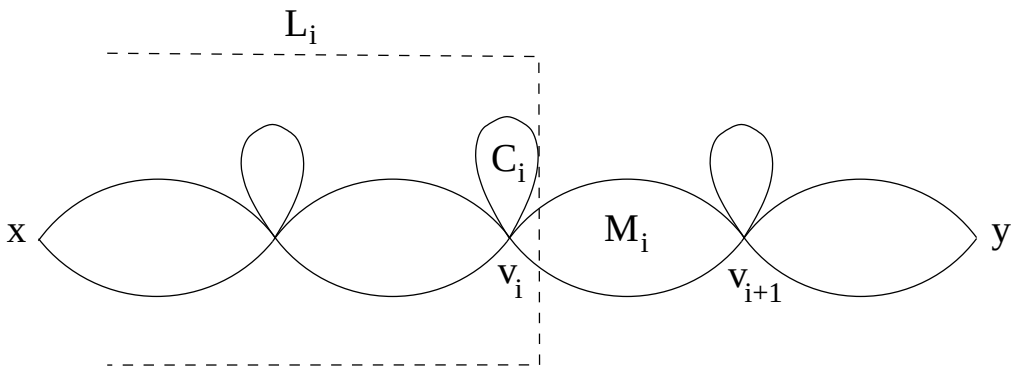
Application of Dilworth's Theorem



If the choosen edge in the backbone is $v_i v_{i+1}$, then :

- The edges before $v_i v_{i+1}$ can be contracted (hence L_i becomes a cherry).

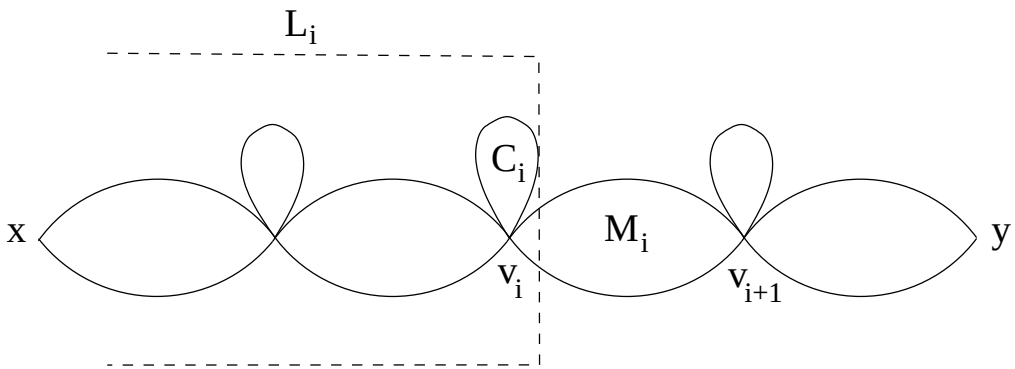
Application of Dilworth's Theorem



If the choosen edge in the backbone is $v_i v_{i+1}$, then :

- The edges before $v_i v_{i+1}$ can be contracted (hence L_i becomes a cherry).
- If $i \leq j$ then $L_i \subseteq L_j$.

Application of Dilworth's Theorem



If the choosen edge in the backbone is $v_i v_{i+1}$, then :

- The edges before $v_i v_{i+1}$ can be contracted (hence L_i becomes a cherry).
- If $i \leq j$ then $L_i \subseteq L_j$.

Theorem

Let $K_1,\ldots,K_p$ be p disjoint components attached on x. Let $L_i = \bigcup_{k=1}^{i} K_i$. Then each L_i has a bounded active set $\mathcal{L}_i$ and $\mathcal{L}_j \cap L_i \subseteq \mathcal{L}_i$ if $i \leq j$.

Application of Dilworth's Theorem

The sets $\mathcal{L}_i$ satisfy $\mathcal{L}_j \cap L_i \subseteq \mathcal{L}_i$.

The order $\preceq$

Let $F_i \subseteq \mathcal{L}_i$ and $F_j \subseteq \mathcal{L}_j$ with $j \geq i$.
$F_i \preceq F_j$ iff $F_j \cap L_{i+1} \subseteq F_i$.

Application of Dilworth's Theorem

The sets $\mathcal{L}_i$ satisfy $\mathcal{L}_j \cap L_i \subseteq \mathcal{L}_i$.

The order $\preceq$

Let $F_i \subseteq \mathcal{L}_i$ and $F_j \subseteq \mathcal{L}_j$ with $j \geq i$.
$F_i \preceq F_j$ iff $F_j \cap L_{i+1} \subseteq F_i$.

Theorem

The order $\preceq$ can be covered by a bounded number of chains.

Application of Dilworth's Theorem

Proof : Let us prove by induction on k that the maximum size of an antichain is bounded.

Dilworth's Theorem

A partial order P can be covered by n chains iff the maximum size of an antichain is n.

Application of Dilworth's Theorem

Proof : Let us prove by induction on k that the maximum size of an antichain is bounded.

Dilworth's Theorem

A partial order P can be covered by n chains iff the maximum size of an antichain is n.

Assume that the antichain for cuts of size $k+1$ is unbounded. Let F_i be the elements of such an antichain. Enumerate the $F_i \subseteq \mathcal{L}_{t_i}$ of the antichain such that if $i < j$ then $t_i \leq t_j$.

- Since F_1 is not comparable with F_i, $F_i \cap \mathcal{L}_{t_1+1} \subsetneq F_1$.

Application of Dilworth's Theorem

Proof : Let us prove by induction on k that the maximum size of an antichain is bounded.

Dilworth's Theorem

A partial order P can be covered by n chains iff the maximum size of an antichain is n.

Assume that the antichain for cuts of size $k+1$ is unbounded. Let F_i be the elements of such an antichain. Enumerate the $F_i \subseteq \mathcal{L}_{t_i}$ of the antichain such that if $i < j$ then $t_i \leq t_j$.

- Since F_1 is not comparable with F_i, $F_i \cap \mathcal{L}_{t_1+1} \subsetneq F_1$.
- Hence F_i has a vertex in $\mathcal{L}_{t_1+1}$.

Application of Dilworth's Theorem

Proof : Let us prove by induction on k that the maximum size of an antichain is bounded.

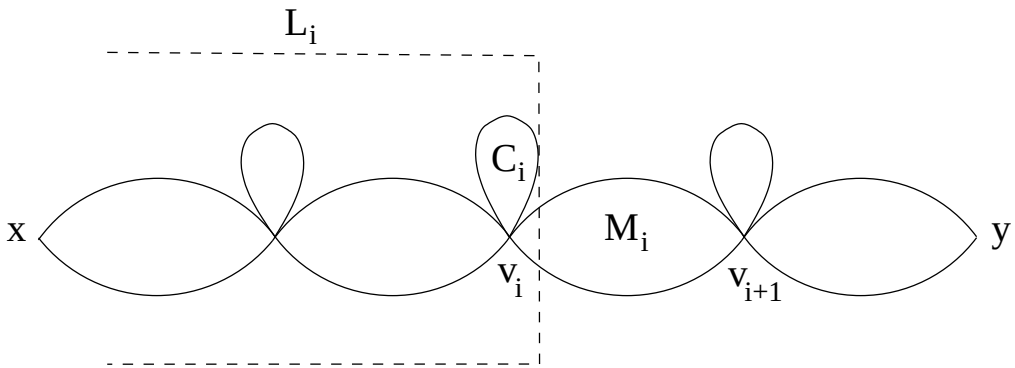
Dilworth's Theorem

A partial order P can be covered by n chains iff the maximum size of an antichain is n.

Assume that the antichain for cuts of size $k+1$ is unbounded. Let F_i be the elements of such an antichain. Enumerate the $F_i \subseteq \mathcal{L}_{t_i}$ of the antichain such that if $i < j$ then $t_i \leq t_j$.

- Since F_1 is not comparable with F_i, $F_i \cap \mathcal{L}_{t_1+1} \subsetneq F_1$.
- Hence F_i has a vertex in $\mathcal{L}_{t_1+1}$.
- Since $\mathcal{L}_{t_1+1}$ has a bounded size, an arbitrarily large number of F_i share the same vertex.
- Apply the induction hypothesis on this set (the sets have size k since they share an edge).

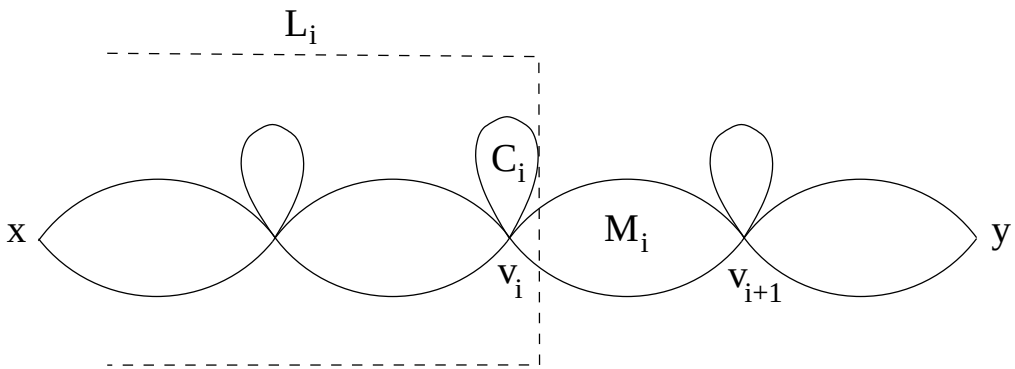
Projection of the requests



Consider a chain of P and a request from $y \in C_k$ passing through x then :

- When $i < k$, the request is cut.
- When $i \geq k$, the request is cut iff it is cut by $F_i \cap \mathcal{L}_k$.
- When the request becomes uncut, it is still uncut when i increases : indeed if $j > i$, $F_j \cap \mathcal{L}_i \subseteq F_i$.

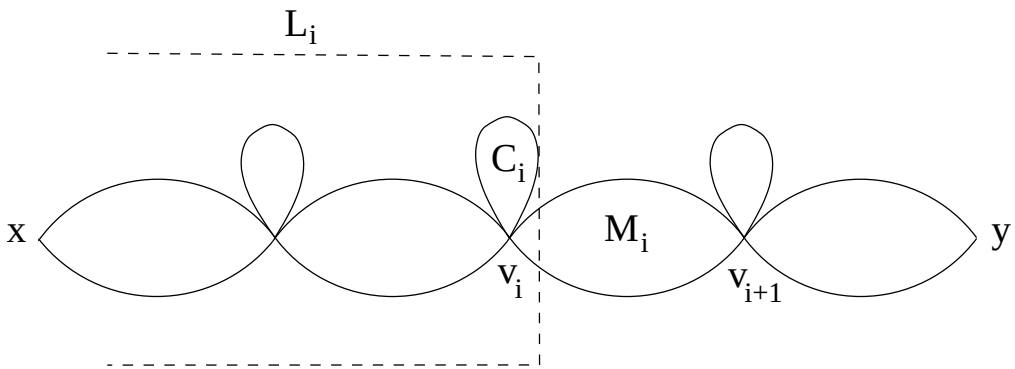
Projection of the requests



Projection of the requests

If the request is cut before v_i and uncut after v_{i+1}, the request can be projected on v_{i+1}.

Projection of the requests



Projection of the requests

If the request is cut before v_i and uncut after v_{i+1}, the request can be projected on v_{i+1}.

Theorem

We can assume that no component is attached on the vertices v_i.

Comparison with Marx and Razgon's proof

Our positive points :

- We have a branching algorithm.
- Our proof is self-contained.

Their positive points :

- Their complexity is better : $O(2^{k^3})$.
- Their proof is written for Vertex-Multicut.

Multiflow

Multiflow

Input : A graph G, a set of requests R, an integer k.
Output :YES iff there are k edge-disjoint paths between pairs of vertices of R.

Multiflow

Multiflow

Input : A graph G, a set of requests R, an integer k.
Output :YES iff there are k edge-disjoint paths between pairs of vertices of R.

Open problem

Is the Multiflow problem FPT parameterized by the size of the solution ?

Polynomial kernels

Definition

A problem parameterized by k has a polynomial kernel iff there is an algorithm running in polynomial time which transforms an instance (n, k) into an instance (n', k') such that :

- $n' \leq Poly(k)$ and $k' \leq k$.
- The new instance is positive iff the original instance is positive.

Polynomial kernels

Definition

A problem parameterized by k has a polynomial kernel iff there is an algorithm running in polynomial time which transforms an instance (n, k) into an instance (n', k') such that :

- $n' \leq Poly(k)$ and $k' \leq k$.
- The new instance is positive iff the original instance is positive.

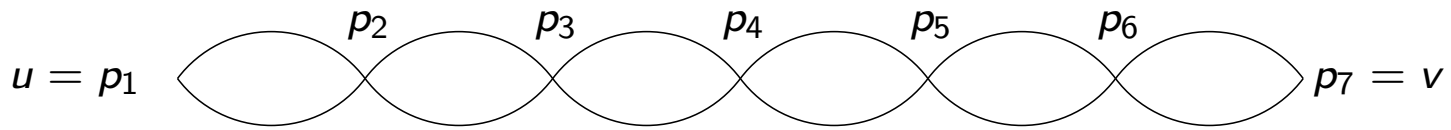
Open problem

Does the Multicut problem have a polynomial Kernel ?

Another approach for Multicut

Open problem

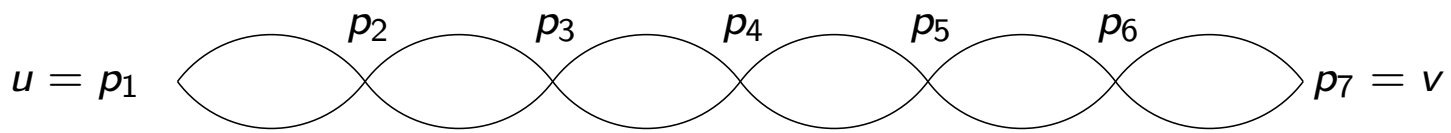
Is it possible to encode a request by a bounded (in k) number of paths and to compute them in FPT-time ?



Another approach for Multicut

Open problem

Is it possible to encode a request by a bounded (in k) number of paths and to compute them in FPT-time ?



Hitting path problem

Input : A graph G, a set of paths $\mathcal{P}$, an integer k.
Output : A set of k edges which intersects all the paths.

Open problem

Is Hitting path parameterized by the size of the solution FPT ?

Thanks for your attention

Questions ?