Covering a graph using shortest paths

Florent Foucaud[1]
joint work with:

Dibyayan Chakraborty[2], Antoine Dailly[1], Sandip Das[3],
Harmender Gahlawat[4], Subir Kumar Ghosh[5]

AND

Maël Dumas[6], Anthony Perez[6], Ioan Todinca[6]

AND

Dibyayan Chakraborty[2], Jérémie Chalopin[7], Yann Vaxès[7]

[1] LIMOS, Université Clermont-Auvergne, Clermont-Ferrand, France
[2] University of Leeds, United Kingdom
[3] Indian Statistical Institute, Kolkata, India
[4] G-SCOP, Université Grenoble-Alpes, France
[5] Ramakrishna Mission Vivekananda Edu. and Res. Institute, Kolkata, India
[6] LIFO, Université d'Orléans, Orléans, France
[7] LIS, Université d'Aix-Marseille, France


PARIS
CLERMONT-FERRAND
LYON
BORDEAUX
MARSEILLE


MICHELIN


Pascal

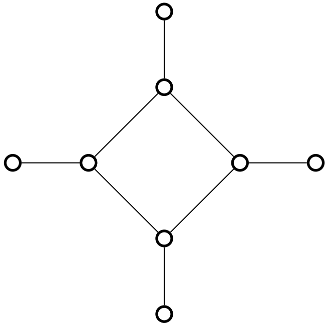

Saint Nectaire

Definitions

isometric path = shortest path between its endpoints

Isometric Path Cover

A set of shortest paths covering every vertex from a graph.

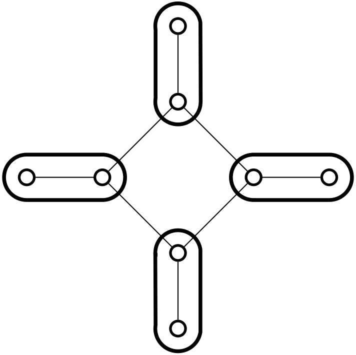


Definitions

isometric path = shortest path between its endpoints

Isometric Path Cover

A set of shortest paths covering every vertex from a graph.

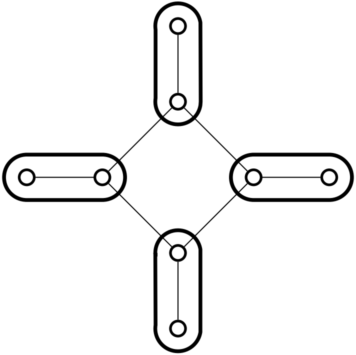


Definitions

isometric path = shortest path between its endpoints

Isometric Path Cover

A set of shortest paths covering every vertex from a graph.
We want to minimize the number of paths.

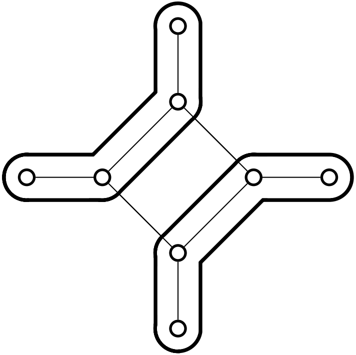


Definitions

isometric path = shortest path between its endpoints

Isometric Path Cover

A set of shortest paths covering every vertex from a graph.
We want to minimize the number of paths.

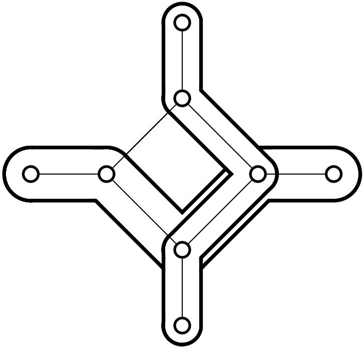


Definitions

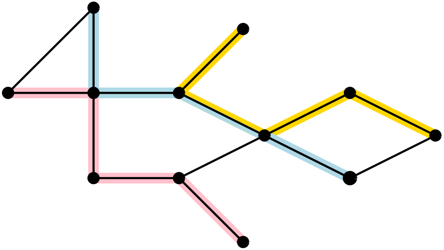
isometric path = shortest path between its endpoints

Isometric Path Cover

A set of shortest paths covering every vertex from a graph.
We want to minimize the number of paths.



Covering a city by bus routes



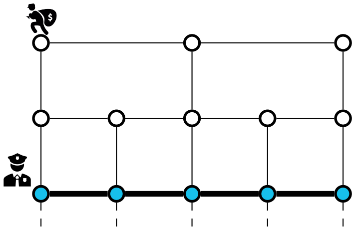
→ The shortest paths represent optimal bus routes

Context: pursuit-evasion problems

Cops and robber game: k cops and one robber are placed on a graph, and alternate their moves (along edges of the graph). The cops win if they can eventually catch the robber.

Lemma [Aigner & Fromme, 1983]

In *cops and robber*, one cop can "protect" a shortest path.

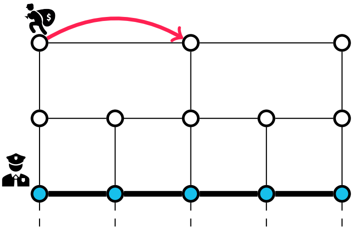


Context: pursuit-evasion problems

Cops and robber game: k cops and one robber are placed on a graph, and alternate their moves (along edges of the graph). The cops win if they can eventually catch the robber.

Lemma [Aigner & Fromme, 1983]

In *cops and robber*, one cop can "protect" a shortest path.

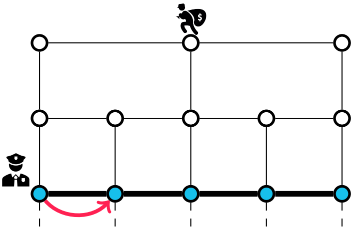


Context: pursuit-evasion problems

Cops and robber game: k cops and one robber are placed on a graph, and alternate their moves (along edges of the graph). The cops win if they can eventually catch the robber.

Lemma [Aigner & Fromme, 1983]

In *cops and robber*, one cop can "protect" a shortest path.

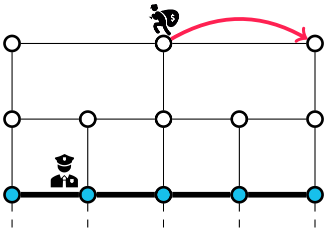


Context: pursuit-evasion problems

Cops and robber game: k cops and one robber are placed on a graph, and alternate their moves (along edges of the graph). The cops win if they can eventually catch the robber.

Lemma [Aigner & Fromme, 1983]

In *cops and robber*, one cop can "protect" a shortest path.

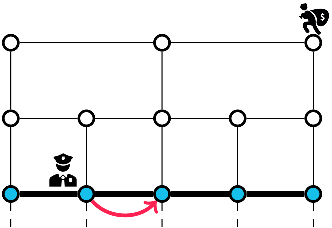


Context: pursuit-evasion problems

Cops and robber game: k cops and one robber are placed on a graph, and alternate their moves (along edges of the graph). The cops win if they can eventually catch the robber.

Lemma [Aigner & Fromme, 1983]

In *cops and robber*, one cop can "protect" a shortest path.

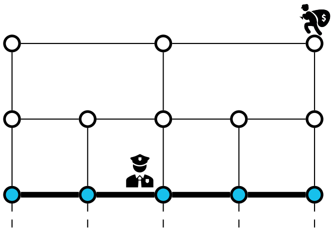


Context: pursuit-evasion problems

Cops and robber game: k cops and one robber are placed on a graph, and alternate their moves (along edges of the graph). The cops win if they can eventually catch the robber.

Lemma [Aigner & Fromme, 1983]

In *cops and robber*, one cop can "protect" a shortest path.



Context: pursuit-evasion problems

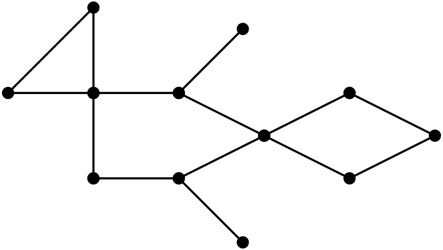
Cops and robber game: k cops and one robber are placed on a graph, and alternate their moves (along edges of the graph). The cops win if they can eventually catch the robber.

Lemma [Aigner & Fromme, 1983]

In *cops and robber*, one cop can "protect" a shortest path.

$\Rightarrow$ The minimum size of an Isometric Path Cover is an upper bound for the number of cops required to catch the robber

Formal problem statement

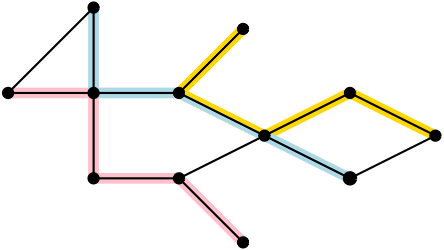


ISOMETRIC PATH COVER (IPC)

Input : A graph G and an integer k.

Question : Is there a set of k shortest paths of G, such that each vertex of G belongs to at least one of the shortest paths?

Formal problem statement

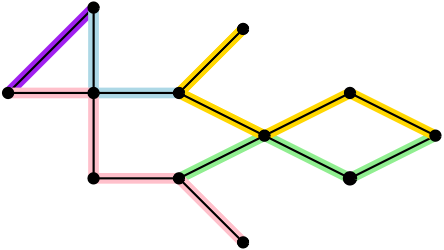


Isometric Path Cover (IPC)

Input : A graph G and an integer k.

Question : Is there a set of k shortest paths of G, such that each vertex of G belongs to at least one of the shortest paths?

Formal problem statement

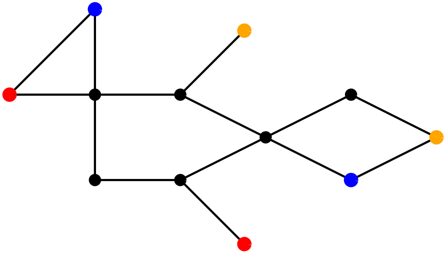


ISOMETRIC PATH COVER (IPC)

Input : A graph G and an integer k.

Question : Is there a set of k shortest paths of G, such that each vertex of G belongs to at least one of the shortest paths?

with terminals



Isometric Path Cover with Terminals (IPC with Terminals)

Input :A graph G, and k pairs of vertices $(s_1, t_1), \ldots, (s_k, t_k)$, the terminals.

Question : Is there a set of k shortest paths of G, the ith path being an s_i-t_i shortest path, such that each vertex of G belongs to at least one of the shortest paths ?

Algorithmic questions

A c-**approximation algorithm** for a given problem is a polynomial-time algorithm producing a feasible solution whose value is at most c times the optimum.

A problem with parameter k is called **FPT** (fixed-parameter tractable) if it has an algorithm of complexity $f(k) \cdot n^{O(1)}$.

It is called **XP** if it has an algorithm with running time $n^{f(k)}$.

Questions

- Is IPC polynomial-time solvable?
- If not, is it approximable?
- Are IPC and IPC WITH TERMINALS FPT? Or at least XP ?

Related problems

Covering:

- Path Cover (NP-c for 1 path : Hamiltonian Path)

Related problems

Covering:

- PATH COVER (NP-c for 1 path : HAMILTONIAN PATH)

Packing (with terminals):

- k DISJOINT PATHS (NP-c [Karp, 1975],
  FPT algorithm: $f(k)n^3$ [Robertson & Seymour, 1995])
- k DISJOINT SHORTEST PATHS (W[1]-hard,
  XP algorithm: $O(kn^{16k \cdot k! + k + 1})$ [Bentert *et al.*, 2021])

Related problems

Covering:

- Path Cover (NP-c for 1 path : Hamiltonian Path)

Packing (with terminals):

- k Disjoint Paths (NP-c [Karp, 1975],
  FPT algorithm: $f(k)n^3$ [Robertson & Seymour, 1995])
- k Disjoint Shortest Paths (W[1]-hard,
  XP algorithm: $O(kn^{16k \cdot k! + k + 1})$ [Bentert *et al.*, 2021])

Partitioning:

- Isometric Path Partition (NP-c [Manuel, 2021])

State of the art on ISOMETRIC PATH COVER

Surprisingly few results!

State of the art on Isometric Path Cover

Surprisingly few results!

Exact values

- Trees, cycles, complete bipartite graphs, several cartesian products of paths [Fitzpatrick, 1997 & 1999]
- Some hypercubes [Fitzpatrick et al, 2001]
- Complete *k*-partite graphs [Pan & Chang, 2006]
- Some cartesian products [Manuel, 2018]

State of the art on Isometric Path Cover

Surprisingly few results!

Exact values

- Trees, cycles, complete bipartite graphs, several cartesian products of paths [Fitzpatrick, 1997 & 1999]
- Some hypercubes [Fitzpatrick et al, 2001]
- Complete k-partite graphs [Pan & Chang, 2006]
- Some cartesian products [Manuel, 2018]

Algorithms

- Linear-time algorithm for block graphs [Pan & Chang, 2005]
- poly-time $\log(d)$-approximation for graphs of diameter d [Thiessen & Gaertner, 2021]

Our results

Our results

NP-hardness

Isometric Path Cover is NP-complete, even on chordal graphs with a dominating vertex.

Our results

NP-hardness

ISOMETRIC PATH COVER is NP-complete, even on chordal graphs with a dominating vertex.

Approximation for chordal graphs (and beyond)

Polynomial-time 4-approximation algorithm on chordal graphs.

Our results

NP-hardness

Isometric Path Cover is NP-complete, even on chordal graphs with a dominating vertex.

Approximation for chordal graphs (and beyond)

Polynomial-time 4-approximation algorithm on chordal graphs.

FPT for chordal graphs

Exact algorithm in $2^{k2^{\mathcal{O}(w)}}n$ and $2^{2^{\mathcal{O}(k)}}n$ on chordal graphs (k = solution size, w = treewidth).

Our results

NP-hardness

Isometric Path Cover is NP-complete, even on chordal graphs with a dominating vertex.

Approximation for chordal graphs (and beyond)

Polynomial-time 4-approximation algorithm on chordal graphs.

FPT for chordal graphs

Exact algorithm in $2^{k2^{\mathcal{O}(w)}}n$ and $2^{2^{\mathcal{O}(k)}}n$ on chordal graphs (k = solution size, w = treewidth).

XP algorithm in general

Structural result: $IPC = k \Rightarrow w \leq f(k)$. This implies a $g(k)n^k$ XP algorithm for Isometric Path Cover.

NP-completeness

chordal graph: every cycle of length at least 4 has a *chord*

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

ISOMETRIC PATH COVER is NP-complete, even on chordal graphs with a dominating vertex.

NP-completeness

chordal graph: every cycle of length at least 4 has a *chord*

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

ISOMETRIC PATH COVER is NP-complete, even on chordal graphs with a dominating vertex.

Proof

Reduction from INDUCED P_3-PARTITION (NP-complete even on chordal graphs with $3k$ vertices [van Bevern *et al.*, 2017])

NP-completeness

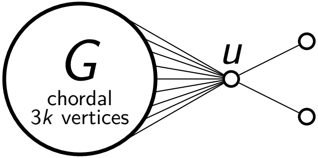
chordal graph: every cycle of length at least 4 has a *chord*

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

ISOMETRIC PATH COVER is NP-complete, even on chordal graphs with a dominating vertex.

Proof

Reduction from INDUCED P_3-PARTITION (NP-complete even on chordal graphs with $3k$ vertices [van Bevern *et al.*, 2017])



We are looking for an ISOMETRIC PATH COVER of size $k + 1$.

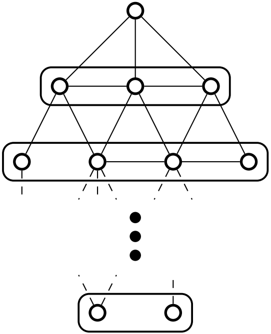
Part 1: approximation algorithm for chordal graphs and beyond

Approximation algorithm for chordal graphs

Approximation algorithm for chordal graphs

Algorithm

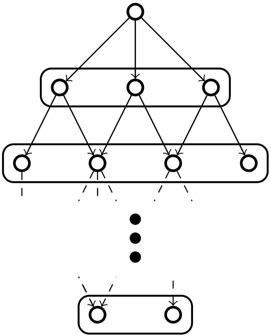
1. Do a Breadth-First Search (BFS) of G



Approximation algorithm for chordal graphs

Algorithm

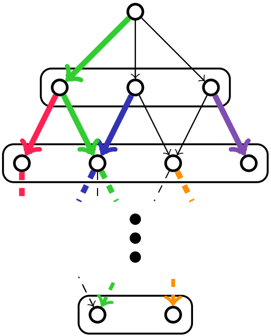
1. Do a Breadth-First Search (BFS) of G
2. Search graph $\equiv$ Hasse diagram of a poset



Approximation algorithm for chordal graphs

Algorithm

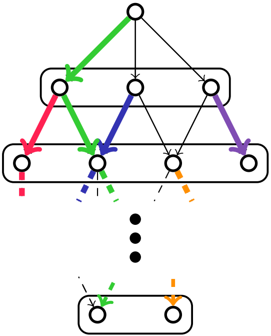
1. Do a Breadth-First Search (BFS) of G
2. Search graph $\equiv$ Hasse diagram of a poset
   $\Rightarrow$ Chain covering of the poset $\mathcal{C}_{\min}$
   [Fulkerson, 1956]



Approximation algorithm for chordal graphs

Algorithm

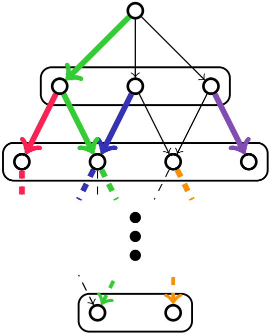
1. Do a Breadth-First Search (BFS) of G
2. Search graph $\equiv$ Hasse diagram of a poset
   $\Rightarrow$ Chain covering of the poset $\mathcal{C}_{\min}$
   [Fulkerson, 1956]
3. $|\mathcal{C}_{\min}| = |A_{\max}|$ [Dilworth, 1950]



Approximation algorithm for chordal graphs

Algorithm

1. Do a Breadth-First Search (BFS) of G
2. Search graph $\equiv$ Hasse diagram of a poset $\Rightarrow$ Chain covering of the poset $\mathcal{C}_{\min}$ [Fulkerson, 1956]
3. $|\mathcal{C}_{\min}| = |A_{\max}|$ [Dilworth, 1950]
4. **Main Idea:** If any isometric path can contain at most ℓ vertices of any antichain, then the algorithm gives an ℓ-approximation.



Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

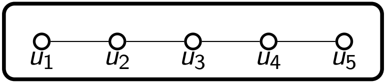
(proof by contradiction)

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

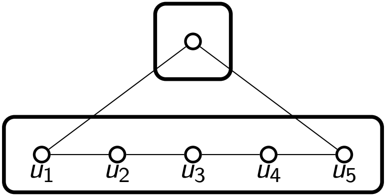


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

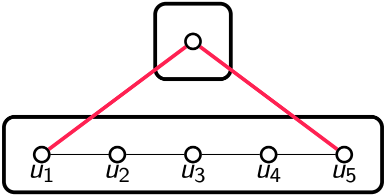


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



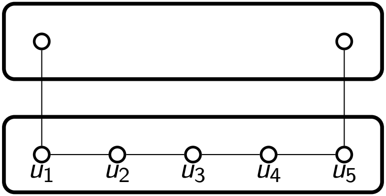
Impossible since $u_1, \ldots, u_5$ in a shortest path

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

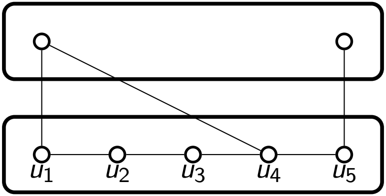


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

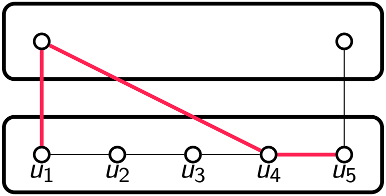


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



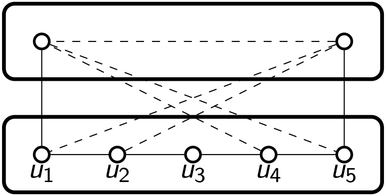
Impossible since $u_1, \ldots, u_5$ in a shortest path

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

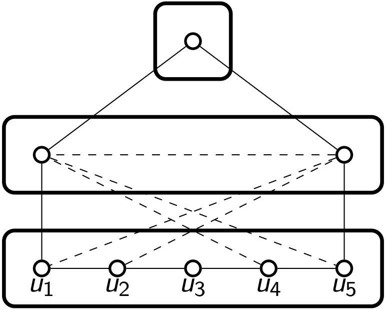


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

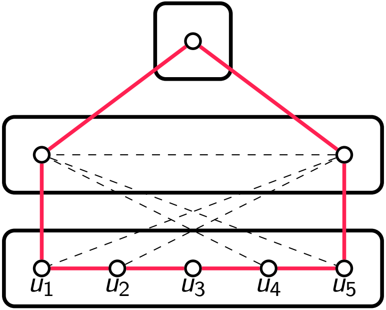


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



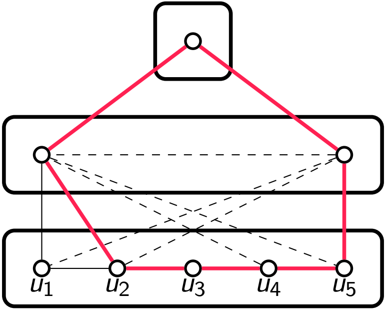
Impossible since the graph is chordal

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



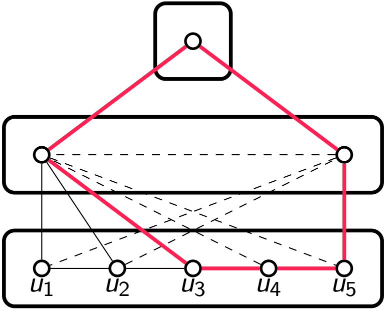
Impossible since the graph is chordal

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



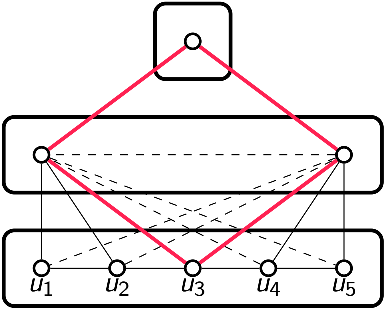
Impossible since the graph is chordal

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



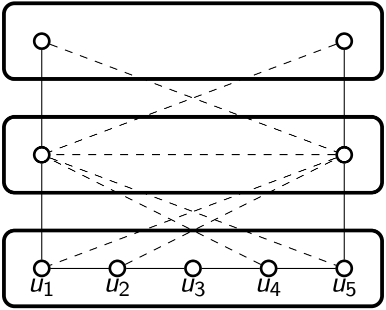
Impossible since the graph is chordal

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

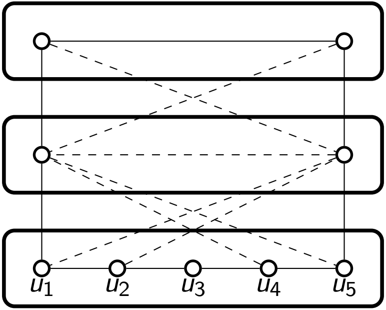


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

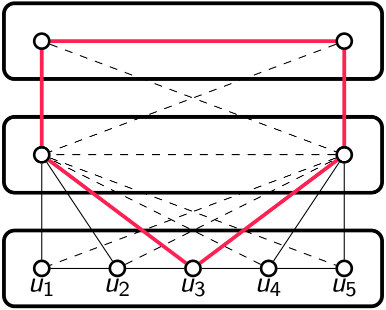


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



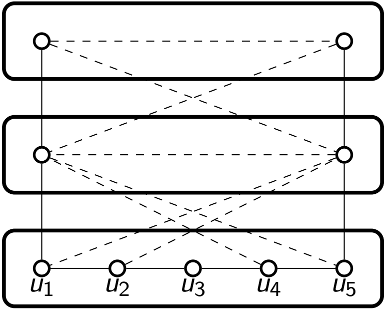
Impossible since the graph is chordal

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph

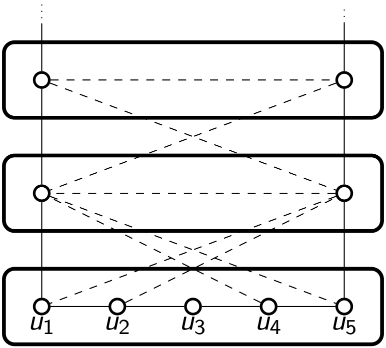


Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



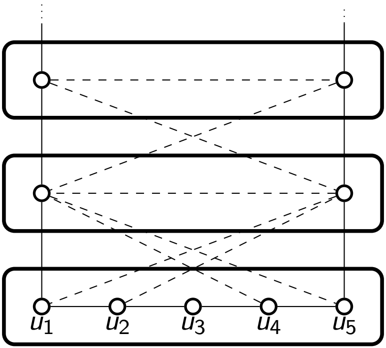
Impossible to have a common ancestor $\Rightarrow$ contradiction

Approximation for chordal graphs: proof idea

No shortest path can contain 5 antichain vertices

(proof by contradiction)

Case 1: a shortest path contains 5 vertices of an antichain on the same level of the search graph



Impossible to have a common ancestor ⇒ contradiction

→ Other cases follow a similar reasoning

Approximation: Extension

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

The algorithm yields the following approximation ratios:

- 4 on chordal graphs

Approximation: Extension

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

The algorithm yields the following approximation ratios:

- 4 on chordal graphs
- 3 on interval graphs
- 2 on proper interval graphs

Approximation: Extension

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

The algorithm yields the following approximation ratios:

- 4 on chordal graphs
- 3 on interval graphs
- 2 on proper interval graphs
- $k + 7$ on k-chordal graphs (with $k \geq 4$)
- $6\ell + 2$ on graphs of treelength at most ℓ

Approximation: Extension

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

The algorithm yields the following approximation ratios:

- 4 on chordal graphs $\rightarrow$ tight
- 3 on interval graphs $\rightarrow$ tight
- 2 on proper interval graphs $\rightarrow$ tight
- $k+7$ on k-chordal graphs (with $k \geq 4$)
- $6\ell + 2$ on graphs of treelength at most ℓ

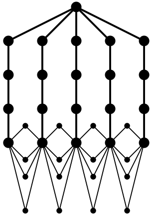
Approximation: Extension

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

The algorithm yields the following approximation ratios:

- 4 on chordal graphs $\rightarrow$ tight
- 3 on interval graphs $\rightarrow$ tight
- 2 on proper interval graphs $\rightarrow$ tight
- $k + 7$ on k-chordal graphs (with $k \geq 4$)
- $6\ell + 2$ on graphs of treelength at most ℓ

General graphs (n vertices): the algorithm can yield $\Omega(\sqrt{n})$-factor



A new graph parameter: IPCO

isometric path complexity of graph G

Minimum integer k such that there exists $v \in V(G)$ s.t: the vertices of any isometric path P of G can be covered by k many v-rooted isometric paths. $\rightarrow$ Denoted $ipco(G)$

A new graph parameter: IPCO

isometric path complexity of graph G

Minimum integer k such that there exists $v \in V(G)$ s.t:
the vertices of any isometric path P of G can be covered by k many v-rooted isometric paths. $\rightarrow$ Denoted $ipco(G)$

Equal to max. number of antichain vertices in an isometric path!

A new graph parameter: IPCO

isometric path complexity of graph G

Minimum integer k such that there exists $v \in V(G)$ s.t:
the vertices of any isometric path P of G can be covered by k many v-rooted isometric paths. $\rightarrow$ Denoted $ipco(G)$

Equal to max. number of antichain vertices in an isometric path!

Theorem [Chakraborty, Chalopin, F, Vaxès 2023]

$ipco(G)$ can be computed in time $O(n^2m)$

A new graph parameter: IPCO

isometric path complexity of graph G

Minimum integer k such that there exists $v \in V(G)$ s.t:
the vertices of any isometric path P of G can be covered by k many v-rooted isometric paths. $\rightarrow$ Denoted $ipco(G)$

Equal to max. number of antichain vertices in an isometric path!

Theorem [Chakraborty, Chalopin, F, Vaxès 2023]

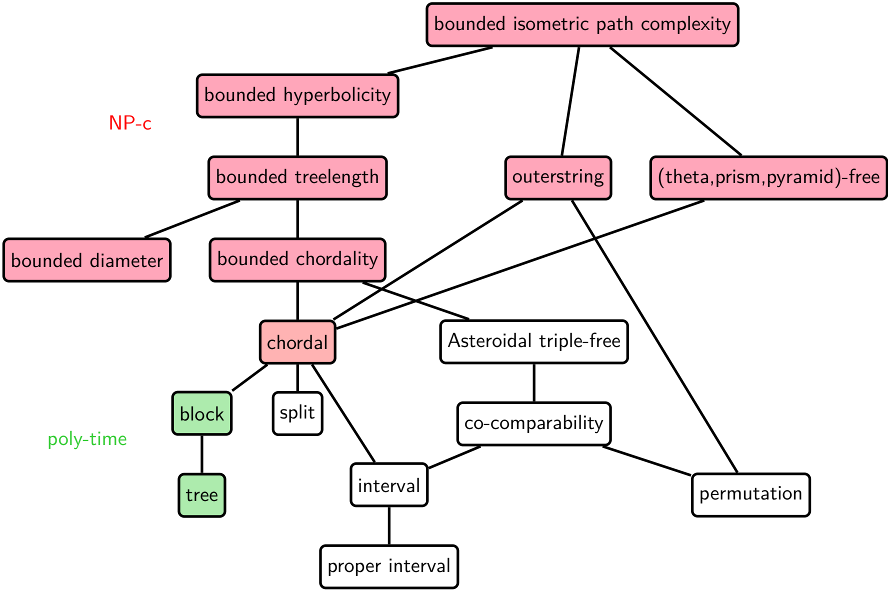
$ipco(G)$ can be computed in time $O(n^2m)$

Theorem [Chakraborty, Chalopin, F, Vaxès 2023]

$ipco(G)$ is bounded for:

- graphs of bounded hyperbolicity
- outerstring graphs
- (theta, prism, pyramid)-free graphs

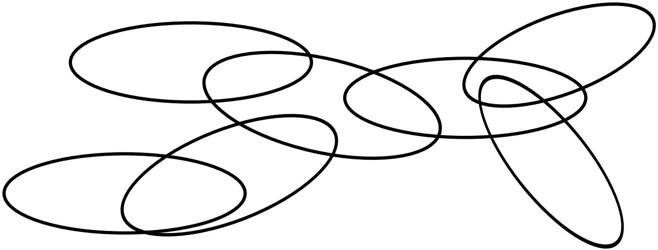
Graph classes where IPC is constant-factor approximable



Part 2: relation with tree-width

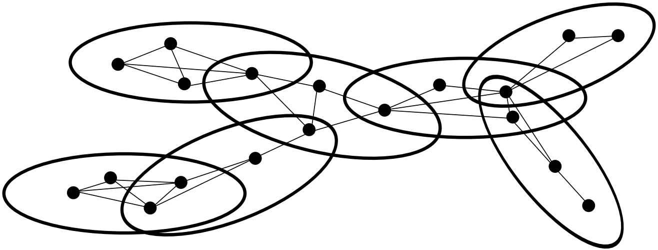
Graphs of bounded tree-width

Graphs of tree-width k: look like a tree where each edge is replaced by a set of k vertices



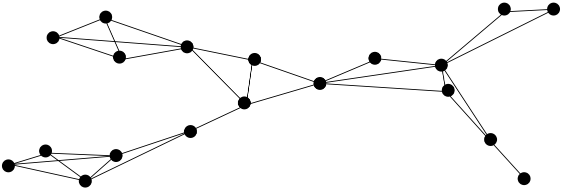
Graphs of bounded tree-width

Graphs of tree-width k: look like a tree where each edge is replaced by a set of k vertices



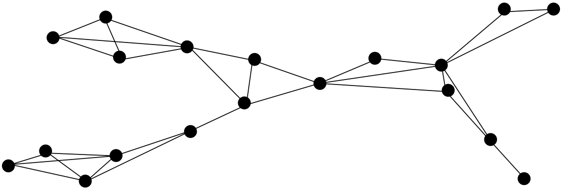
Graphs of bounded tree-width

Graphs of tree-width k: look like a tree where each edge is replaced by a set of k vertices



Graphs of bounded tree-width

Graphs of tree-width k: look like a tree where each edge is replaced by a set of k vertices



For graphs of treewidth k and n vertices, many problems can be solved in time $f(k)poly(n)$, for some (potentially exponential) function k.

$\rightarrow$ dynamic programming on the tree-like structure

$\rightarrow$ Courcelle's theorem (1990)

FPT algorithm for chordal graphs using tree-width

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

IPC can be solved in time $2^{k2^{\mathcal{O}(w)}}n$ and $2^{2^{\mathcal{O}(k)}}n$ on chordal graphs ($k =$ solution size, $w =$ treewidth).

- Dynamic programming on tree decomposition. For chordal graphs: $TW =$ Clique Number-1.
- Each solution path can contain at most 2 vertices from each clique. So $k \geq \frac{TW+1}{2}$.
- Hence, FPT algorithm by TW implies FPT algorithm by solution size.

FPT algorithm for chordal graphs using tree-width

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

IPC can be solved in time $2^{k2^{\mathcal{O}(w)}}n$ and $2^{2^{\mathcal{O}(k)}}n$ on chordal graphs (k = solution size, w = treewidth).

- *close*(X, y): vertices in X which are closer to y.
- Main idea: P is isometric $\iff$ for every vertex v of P and clique X intersecting P, P contains exactly one vertex from $V(P) \cap X$ in *close*(X, v)

FPT algorithm for chordal graphs using tree-width

Theorem [Chakraborty, Dailly, Das, F, Gahlawat, Ghosh 2022]

IPC can be solved in time $2^{k2^{\mathcal{O}(w)}}n$ and $2^{2^{\mathcal{O}(k)}}n$ on chordal graphs (k = solution size, w = treewidth).

- $close(X, y)$: vertices in X which are closer to y.
- Main idea: P is isometric $\iff$ for every vertex v of P and clique X intersecting P, P contains exactly one vertex from $V(P) \cap X$ in $close(X, v)$
- For each bag X and vertex y, the DP table maintains:
  - the intersection between each path and X (at most 2 vertices per path)
  - for each path intersecting X, if it continues "above" or "below" X
  - for each path, if it has already been used entierly "below" or not
  - for each path P_i and each subset S of X, if there is a vertex y "above" and "below" with $close(X, y) = S$

Combinatorial result

Theorem [Dumas, F, Perez, Todinca 2022]

If G is coverable by k shortest paths then, for any vertex a and any fixed distance D, the number of vertices at distance exactly D from a is upper bounded by some function $g(k)$.

Combinatorial result

Theorem [Dumas, F, Perez, Todinca 2022]

If G is coverable by k shortest paths then, for any vertex a and any fixed distance D, the number of vertices at distance exactly D from a is upper bounded by some function $g(k)$.

- Edge coverable : $g(k) = O(3^k)$.

Combinatorial result

Theorem [Dumas, F, Perez, Todinca 2022]

If G is coverable by k shortest paths then, for any vertex a and any fixed distance D, the number of vertices at distance exactly D from a is upper bounded by some function $g(k)$.

- Edge coverable : $g(k) = O(3^k)$.
- Vertex coverable : $g(k) = O(k \cdot 3^k)$.

Combinatorial result

Theorem [Dumas, F, Perez, Todinca 2022]

If G is coverable by k shortest paths then, for any vertex a and any fixed distance D, the number of vertices at distance exactly D from a is upper bounded by some function $g(k)$.

- Edge coverable : $g(k) = O(3^k)$.
- Vertex coverable : $g(k) = O(k \cdot 3^k)$.

Corollary

G is of **treewidth** at most $2 \cdot g(k)$.

Combinatorial result

Theorem [Dumas, F, Perez, Todinca 2022]

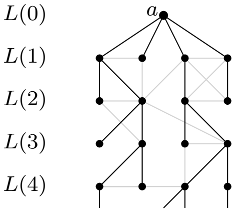
If G is coverable by k shortest paths then, for any vertex a and any fixed distance D, the number of vertices at distance exactly D from a is upper bounded by some function $g(k)$.

- Edge coverable : $g(k) = O(3^k)$.
- Vertex coverable : $g(k) = O(k \cdot 3^k)$.

Corollary

G is of **treewidth** at most $2 \cdot g(k)$.

- Decomposition based on a breadth-first search (BFS).



Combinatorial result

Theorem [Dumas, F, Perez, Todinca 2022]

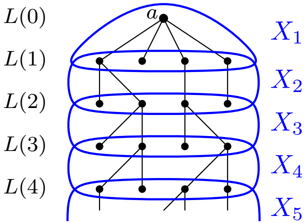
If G is coverable by k shortest paths then, for any vertex a and any fixed distance D, the number of vertices at distance exactly D from a is upper bounded by some function $g(k)$.

- Edge coverable : $g(k) = O(3^k)$.
- Vertex coverable : $g(k) = O(k \cdot 3^k)$.

Corollary

G is of **treewidth** at most $2 \cdot g(k)$.

- Decomposition based on a breadth-first search (BFS).
- Each bag: two consecutive layers.



Algorithmic consequences

Theorem [Dumas, F, Perez, Todinca 2022]

IPC WITH TERMINALS is **FPT**, with running time $O(f(k) \cdot n)$.

Algorithmic consequences

Theorem [Dumas, F, Perez, Todinca 2022]

IPC WITH TERMINALS is **FPT**, with running time $O(f(k) \cdot n)$.

- Yes-instances have bounded treewidth by the corollary
- Courcelle's theorem to solve this problem on bounded treewidth graphs.

Algorithmic consequences

Theorem [Dumas, F, Perez, Todinca 2022]

IPC WITH TERMINALS is **FPT**, with running time $O(f(k) \cdot n)$.

- Yes-instances have bounded treewidth by the corollary
- Courcelle's theorem to solve this problem on bounded treewidth graphs.

Corollary

IPC is **XP** for parameter k: $f(k) \cdot n^{O(k)}$.

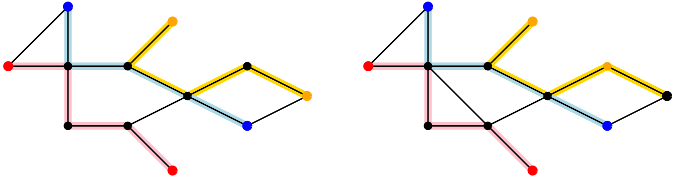
- Brute force: try all possible pairs of k terminals + above theorem

IPC with Terminals is FPT

Theorem [Courcelle, 1990]

Every problem expressible in MSOL_2 can be solved in $f(w) \cdot n$ time on graphs of treewidth at most w.

- Compute a tree decomposition by BFS. If $w > 2g(k)$ return `false`.
- Express in MSOL_2: "find paths $P_1, \ldots, P_k$ s.t. $\forall i, s_i, t_i \in P_i$, each vertex of G is covered". Minimize sum of paths' lengths.

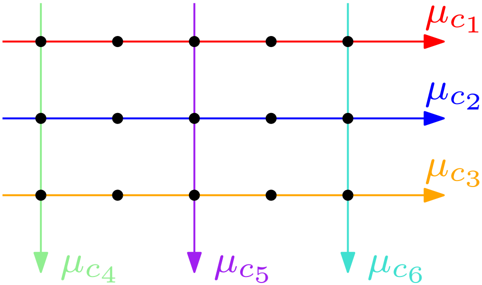


- Use Courcelle's theorem to solve the problem (optimization version).
- **If** $\forall i, |P_i| = dist(s_i, t_i)$ **then** answer `true`, **else** answer `false`.

Proof sketch: base paths colouring

Base paths : the k shortest paths $\mu_1, \ldots, \mu_k$ that cover the graph
To each base path μ_c we give :

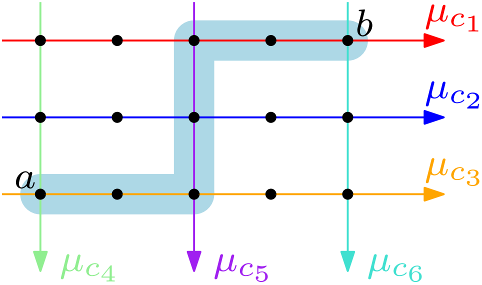
- A colour c, $1 \leq c \leq k$,
- An arbitrary direction.



Each edge of the graph receives a set of colours
colours(e) $\subseteq \{1, \ldots, k\}$.

Colouring of a path

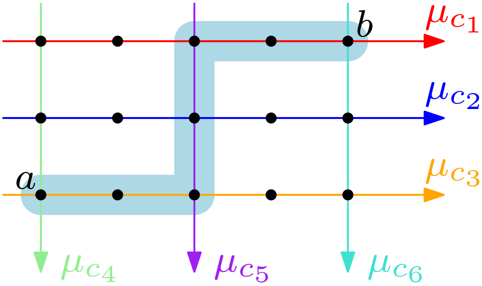
Given a path P from a to b, we can pick for each edge one of its possible colours, it is a colouring col of P.
(P, col) is a coloured path



Colours-signs word

(P, col) can be divided in monochromatic subpaths. These subpaths induce either a "$+$" sign or a "$-$" sign w.r.t. the direction of the base path.

We associate to (P, col) with colours $\{c_1, \ldots, c_\ell\}$ a colours-signs word $\omega = ((c_1, s_1), \ldots, (c_\ell, s_\ell))$ on the alphabet $\{1, \ldots, k\} \times \{+, -\}$.



$$\omega = ((c_3, +), (c_5, -), (c_1, +))$$

Good colouring of a path

(P, col) is well-coloured if the set of edges using a colour c form a **connected subpath** of P.

A path well-coloured :

A path not well-coloured :

Good colouring of a path

Good colouring Lemma

For every pair of vertices a, b of G, there exists a **shortest well-coloured** a-b path.

Good colouring of a path

Good colouring Lemma

For every pair of vertices a, b of G, there exists a **shortest well-coloured** a-b path.

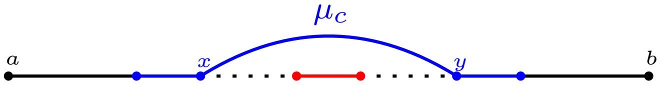
If a a-b shortest path (P, col) isn't well-coloured :

Good colouring of a path

Good colouring Lemma

For every pair of vertices a, b of G, there exists a **shortest well-coloured** a-b path.

If a a-b shortest path (P, col) isn't well-coloured :



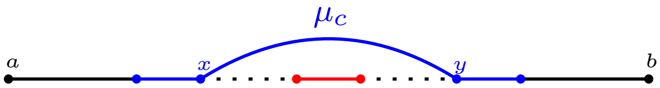
$\Rightarrow$ Replace $P[x, y]$ by $\mu_c[x, y]$.

Good colouring of a path

Good colouring Lemma

For every pair of vertices a, b of G, there exists a **shortest well-coloured** a-b path.

If a a-b shortest path (P, col) isn't well-coloured :



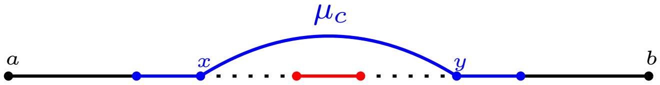
$\Rightarrow$ Replace $P[x, y]$ by $\mu_c[x, y]$.
The constructed path is still a shortest path
$(|\mu_c[x, y]| \leq |P[x, y]|)$.

Good colouring of a path

Good colouring Lemma

For every pair of vertices a, b of G, there exists a **shortest well-coloured** a-b path.

If a a-b shortest path (P, col) isn't well-coloured :

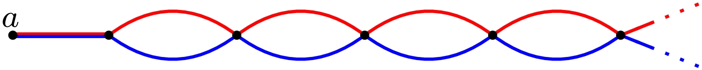


$\Rightarrow$ Replace $P[x, y]$ by $\mu_c[x, y]$.
The constructed path is still a shortest path
$(|\mu_c[x, y]| \leq |P[x, y]|)$.

The number of possible colours-signs words is upper-bounded by $g(k) = O(3^k)$.

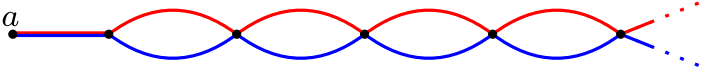
The bound for isometric path edge-covers

Multiple shortest paths of same length may have the same colours-signs word :



The bound for isometric path edge-covers

Multiple shortest paths of same length may have the same colours-signs word :

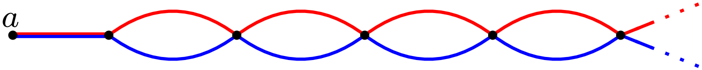


Colours-signs word Lemma

The shortest paths starting at a vertex a, of length D and colours-signs word ω all end at the same vertex b.

The bound for isometric path edge-covers

Multiple shortest paths of same length may have the same colours-signs word :



Colours-signs word Lemma

The shortest paths starting at a vertex a, of length D and colours-signs word ω all end at the same vertex b.

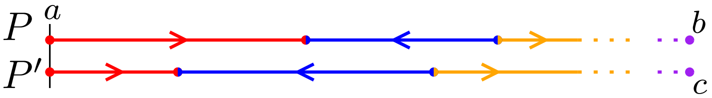
Theorem [Dumas, F, Perez, Todinca 2022]

For any vertex a and any fixed distance D, the number of vertices at distance exactly D from a is upper bounded by $g(k)$ (the number of colours-signs words).

Proof of the Colours-signs word Lemma

Let b and c be vertices at distance D from vertex a. Let (P, col), (P', col') be two well-coloured shortest a-b and a-c paths.

Claim : If they have the same colours-signs word, then $b = c$.

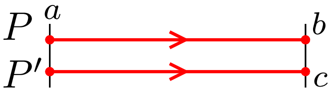


Proof of the Colours-signs word Lemma

Let b and c be vertices at distance D from vertex a. Let (P, col), (P', col') be two well-coloured shortest a-b and a-c paths.

Claim : If they have the same colours-signs word, then $b = c$.

Proof by induction on the number of letters ℓ of the colours-signs word.
If $\ell = 1$



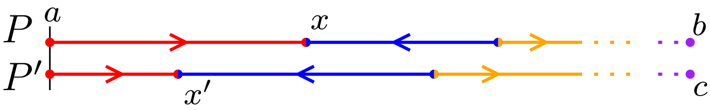
$dist(a, b) = dist(a, c)$ thus $b = c$.

Proof of the Colours-signs word Lemma

Let b and c be vertices at distance D from vertex a. Let (P, col), (P', col') be two well-coloured shortest a-b and a-c paths.

Claim : If they have the same colours-signs word, then $b = c$.

Proof by induction on the number of letters ℓ of the colours-signs word.
If $\ell > 1$



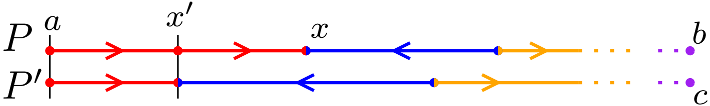
Take P the path with the longest subpath of the colour c_1.

Proof of the Colours-signs word Lemma

Let b and c be vertices at distance D from vertex a. Let (P, col), (P', col') be two well-coloured shortest a-b and a-c paths.

Claim : If they have the same colours-signs word, then $b = c$.

Proof by induction on the number of letters ℓ of the colours-signs word.
If $\ell > 1$



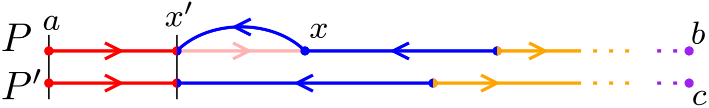
The vertex x' is in the path $\mu_{c_1}[a, x]$, thus in the path P

Proof of the Colours-signs word Lemma

Let b and c be vertices at distance D from vertex a. Let (P, col), (P', col') be two well-coloured shortest a-b and a-c paths.

Claim : If they have the same colours-signs word, then $b = c$.

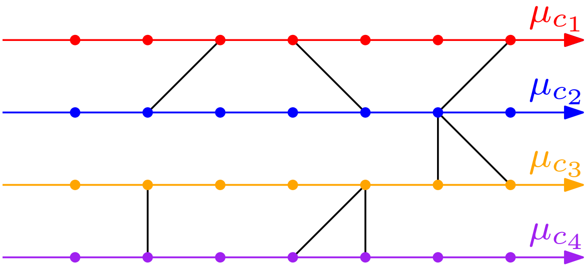
Proof by induction on the number of letters ℓ of the colours-signs word.
If $\ell > 1$



Replace $P[x', x]$ by $\mu_{c_2}[x', x]$.
$P[x', b]$ and $P'[x', c]$ have $\ell - 1$ colours, by the induction hypothesis, $b = c$.

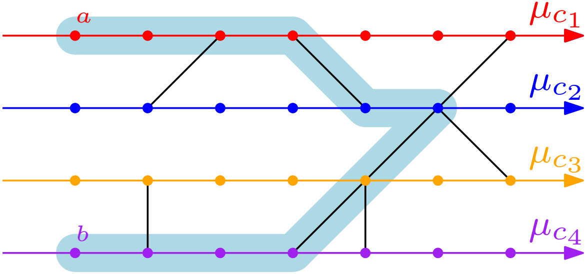
Vertex-covering case: colouring of a path

A colour and a direction given to each base path.



Vertex-covering case: colouring of a path

A colour and a direction given to each base path.



A colouring **col** associate to each vertex $v \in P$ a colour $c \in \text{colours}(v)$.

Colours-signs words are defined the same way as in the edge case.

Problem

Colours-signs word Lemma

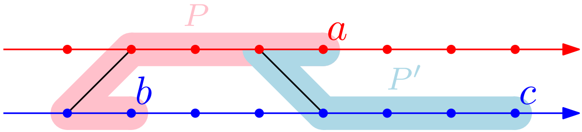
The shortest paths starting at vertex a, of length D and colours-signs word ω all end in the same vertex b.

Problem

Colours-signs word Lemma

The shortest paths starting at vertex a, of length D and colours-signs word ω all end in the same vertex b.

$\Rightarrow$ FALSE in the vertex case

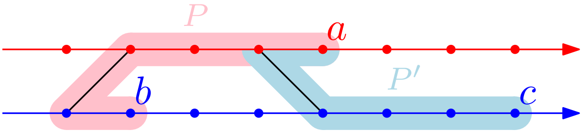


Problem

Colours-signs word Lemma

The shortest paths starting at vertex a, of length D and colours-signs word ω all end in the same vertex b.

$\Rightarrow$ FALSE in the vertex case



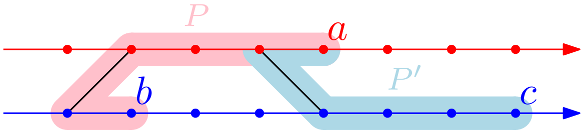
- BUT: number of vertices at distance D from a that share the same colours-signs word is at most $2k$.

Problem

Colours-signs word Lemma

The shortest paths starting at vertex a, of length D and colours-signs word ω all end in the same vertex b.

$\Longrightarrow$ FALSE in the vertex case

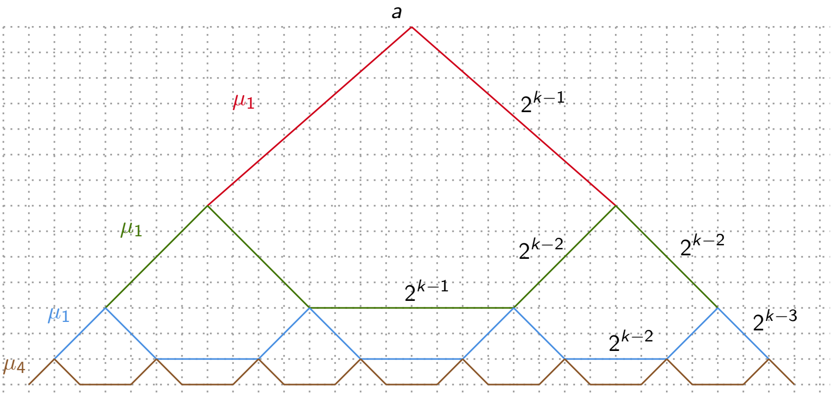


- BUT: number of vertices at distance D from a that share the same colours-signs word is at most $2k$.
- There are at most $g(k) = O(k \cdot 3^k)$ vertices at a given distance of vertex a.

Lower bound

In a graph vertex-coverable by k shortest paths, the number of vertices at same distance of a source is upper bounded by $g(k)$.

- We have shown that $g(k) = O(k \cdot 3^k)$.
- Lower bound : $g(k) \geq 2^k$



Conclusion

- NP-completeness on chordal graphs
- Approximation algorithm on several classes
- FPT algorithm by solution size on chordal graphs
- Bounding the treewidth by a function of solution size
- FPT algorithm by solution size when terminals fixed
- XP algorithm by solution size when terminals free

Conclusion

- NP-completeness on chordal graphs
- Approximation algorithm on several classes
- FPT algorithm by solution size on chordal graphs
- Bounding the treewidth by a function of solution size
- FPT algorithm by solution size when terminals fixed
- XP algorithm by solution size when terminals free

Open problems

- Complexity on interval graphs, split graphs... Planar graphs?
- Constant factor approximation algorithm for all graphs?
- Is Isometric Path Cover FPT or W[1]-hard? (sol. size)
- Can we improve the bound on the treewidth? Polynomial?
- Approximation for Isometric Path Partition?

Conclusion

- NP-completeness on chordal graphs
- Approximation algorithm on several classes
- FPT algorithm by solution size on chordal graphs
- Bounding the treewidth by a function of solution size
- FPT algorithm by solution size when terminals fixed
- XP algorithm by solution size when terminals free

Open problems

- Complexity on interval graphs, split graphs... Planar graphs?
- Constant factor approximation algorithm for all graphs?
- Is Isometric Path Cover FPT or W[1]-hard? (sol. size)
- Can we improve the bound on the treewidth? Polynomial?
- Approximation for Isometric Path Partition?

THANK YOU!