Constructive algorithm for pathwidth of matroids

Eunjung KIM, CNRS / University Paris-Dauphine
jointwork with Sang-il Oum and Jisu Jeong (KAIST)

Workshop on Satisfiability Lower bounds and Tight Results for Parameterized and Exponential Time algorithm

Background

Matroid: primer

- $(E, \mathcal{I})$ - a ground set and a family of subsets of E called the independet sets - satisfying

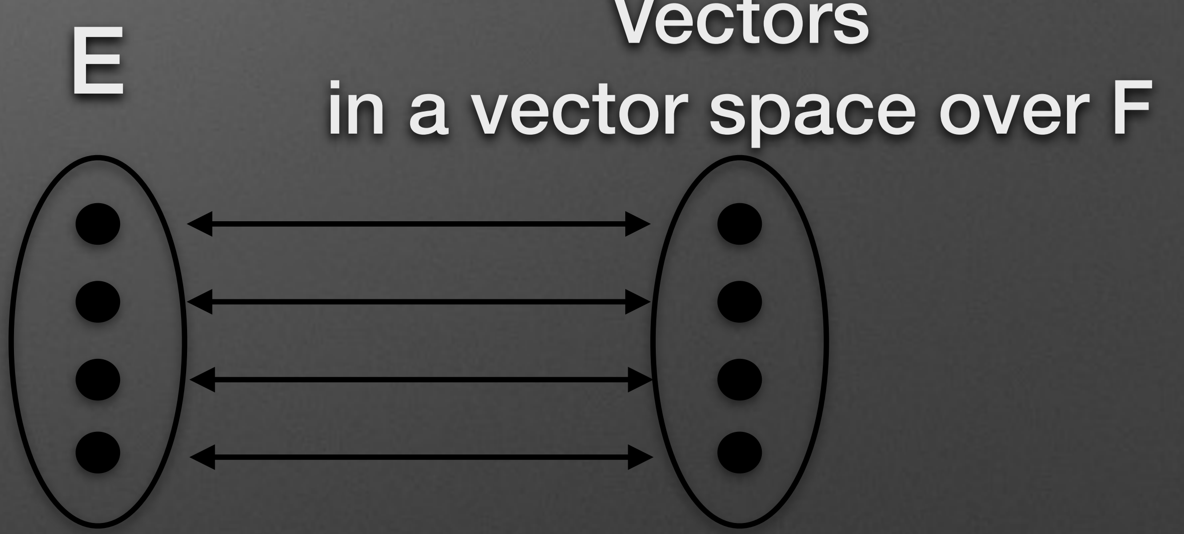
A. $\emptyset \in \mathcal{I}$

B. $X \subset Y$ and $Y \in \mathcal{I} \rightarrow X \in \mathcal{I}$

C. $\forall X,Y \in \mathcal{I}$ with $|X| < |Y|$,

$\exists y \in Y$ s.t. $X \cup \{y\} \in \mathcal{I}$

- A matroid (E, I) is representable in F if



… such that $X \in I$ iff corresponding vectors are independent.

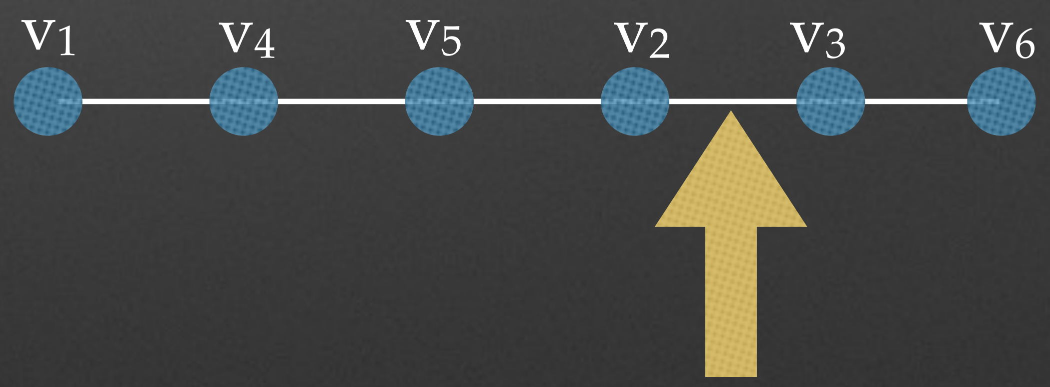
Vectors Arrangement

$\mathbb{F}$ is a finite field

Input: vectors $v_1, v_2, \ldots v_n \in F^r$, a positive integer k

Goal: find a permutation of $v_1, v_2, \ldots v_n$ such that for every i, $\dim(\langle v_1+\ldots+v_i\rangle \cap \langle v_2,\ldots,v_n\rangle) \leq k$.

linear layout
(or pathwidth)
of width≤k,



$\dim(\langle v_1,v_4,v_5,v_2\rangle \cap \langle v_3,v_6\rangle) \leq k$

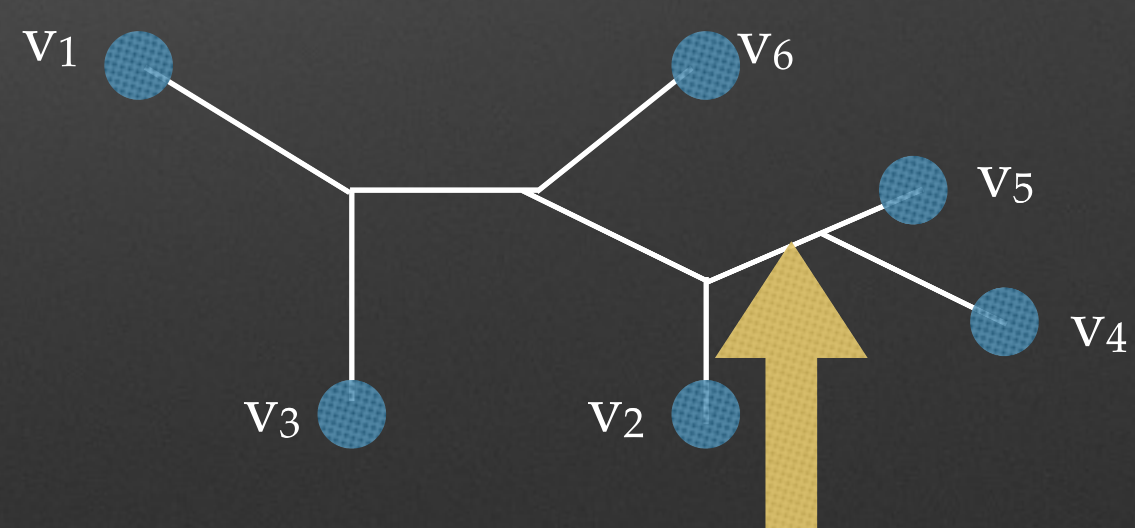
Branchwidth of F-represented matroids

$\mathbb{F}$ is a finite field

Input: vectors $v_1, v_2, \ldots v_n \in F^r$, a positive integer k

Goal: find a subcubic tree T with a bijection L:{leaves} → {vectors} such that for every e in T,

branch decomposition **of width≤k**

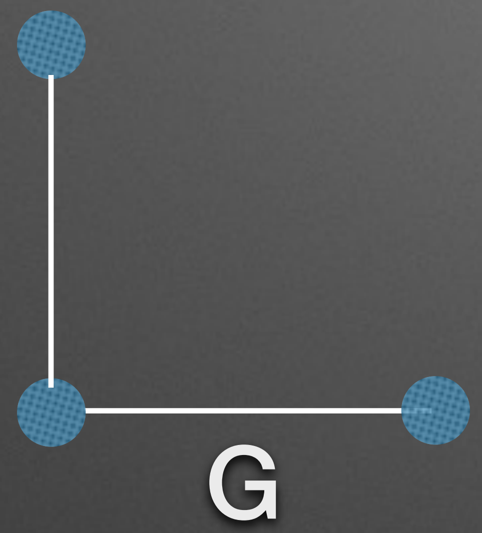


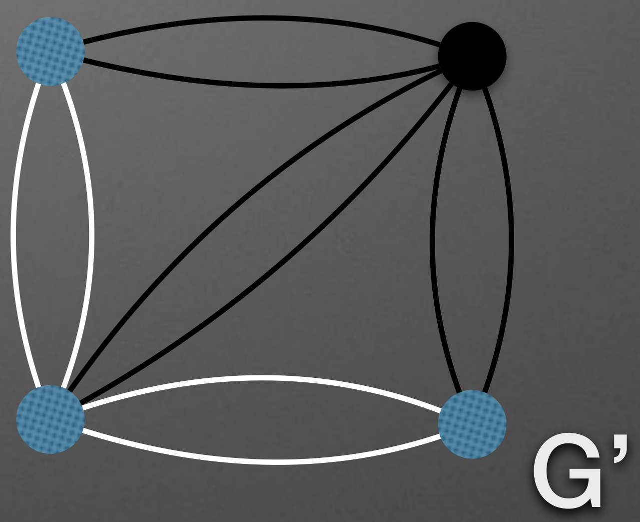
$\dim(\langle v_1,v_3,v_6,v_2\rangle \cap \langle v_4,v_5\rangle) \le k$

From graphs to matroids

pw (cycle matroid of G') = pw (G)

Kashap 2008





When G is not a tree

bw (cycle matroid of G) = bw (G)

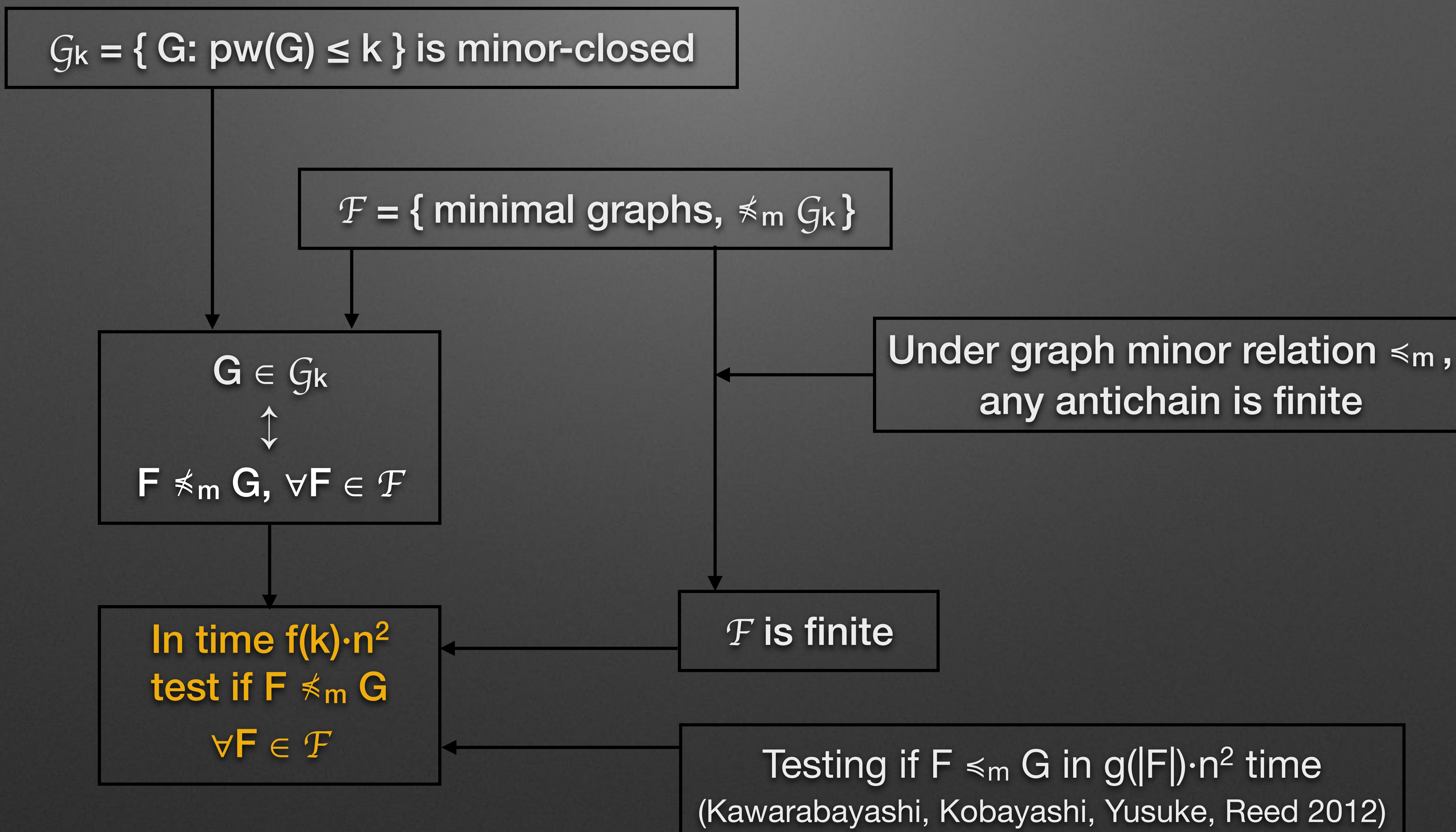
Hicks, McMurray, Nolan 2007
Mazoit, Thomassé 2007

XP algorithms

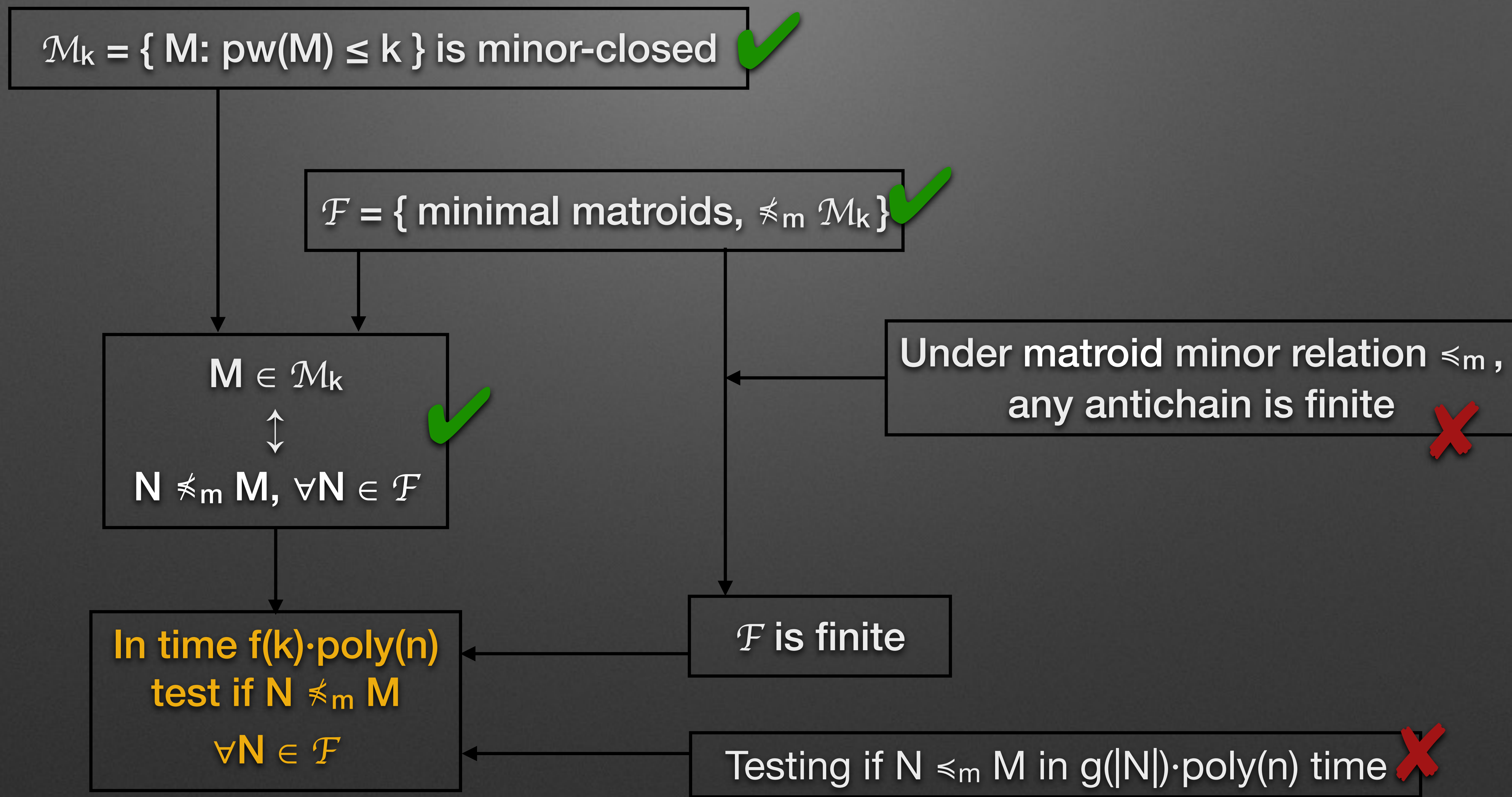
- When k is a fixed constant, polynomial-time algorithm exists for matroids in general (given an independence oracle)
- Pathwidth: Nagamochi 2012
- Branchwidth: Oum, Seymour 2007

What about FPT algorithms?

pw(G)≤k via minor testing



pw(M)≤k via minor testing



pw(M)≤k via minor testing

- Infinite antichain is not difficult to construct.

- Conjecture (RS): for each finite field F, F-representable matroids are w.q.o under $\preccurlyeq_m$
- Geelen, Gerards, Whittle (2014) announced the proof.

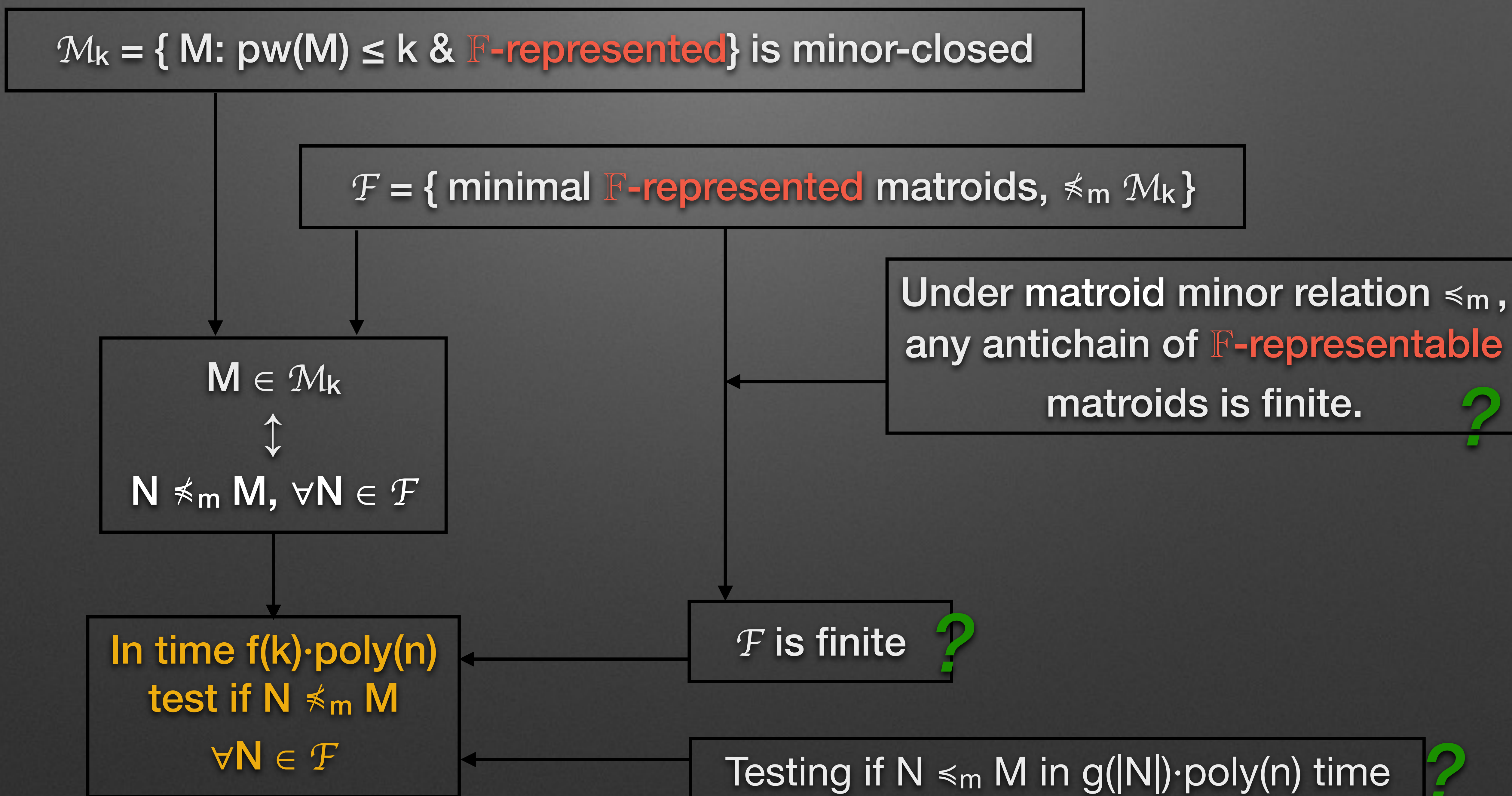
Under matroid minor relation $\preccurlyeq_m$, any antichain is finite ✗

- M is binary $\leftrightarrow$ $U_{2,4} \not\preccurlyeq_m M$
- Any algorithm (with independence oracle) requires exponential time (Seymour 1981)

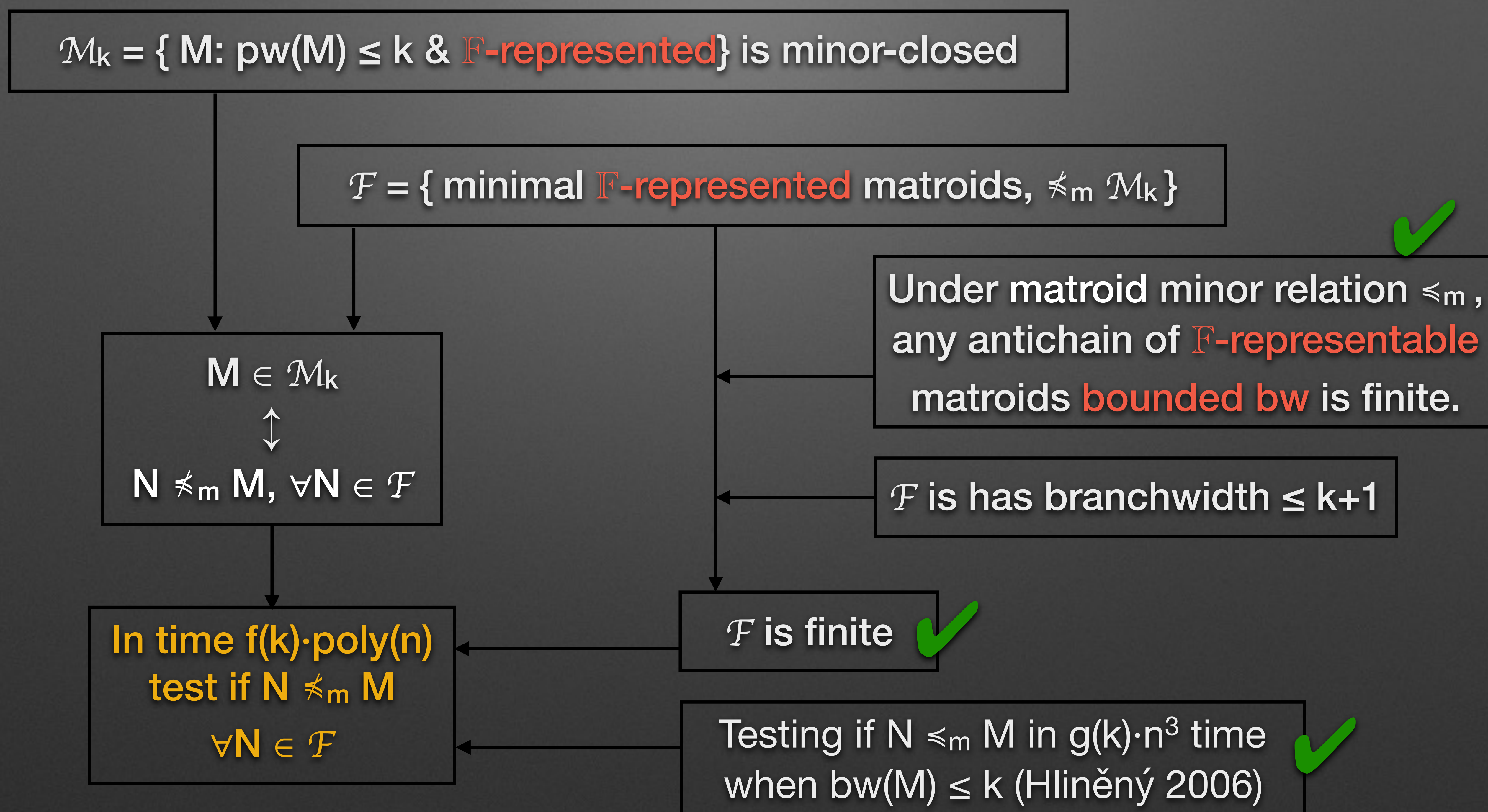
- Conjecture: for any finite field F and any F-representable N, testing $N \preccurlyeq_m M$ in time $g(|N|)\cdot poly(n)$
- Probably GGW taking care of it?

Testing if $N \preccurlyeq_m M$ in $g(|N|)\cdot poly(n)$ time ✗

pw(M)≤k when $\mathbb{F}$-represented



pw(M)≤k when 𝔽-represented



Issues with the approach

- Too many obstructions - at least $(k!)^2$
  (Koutsonas, Thilikos, Yamazaki 2014)
- No algorithm to generate all minor obstructions.
- Even if the complete obstruction list is known, minor testing algorithm (Hliněný 2006) hides gigantic function on k - relies on MSO checking.
- Only decision - no algorithm to actually produce a layout.

Our result

$O(f(k)\cdot n^3)$-time algorithm to decide a linear layout $v_1, v_2, \ldots, v_n$ of the input n vectors in $\mathbb{F}^m$, for any finite field $\mathbb{F}$, such that

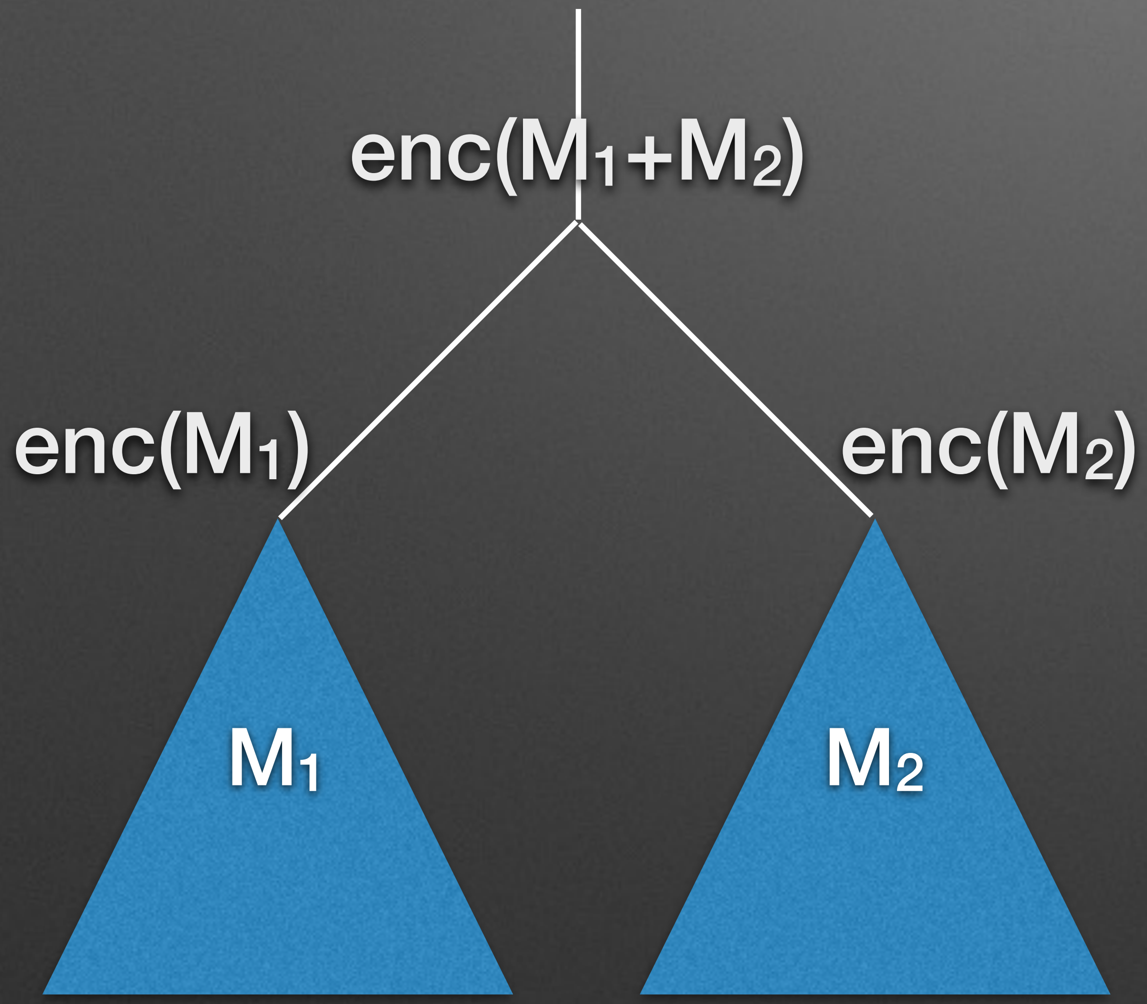
$$\dim(\langle v_1, v_2, \ldots, v_i \rangle \cap \langle v_{i+1}, \ldots, v_n \rangle) \le k \text{ for all } i$$

and output one if exists.

- Dynamic programming for pathwidth (Bodlaender, Kloks 1996)
- Does not depend on a heavy machinery
- Constructive
- Uses 3-approximation for branch-decomposition and can be made self-contained with O(n) overhead

Algorithm for Vector Arrangement

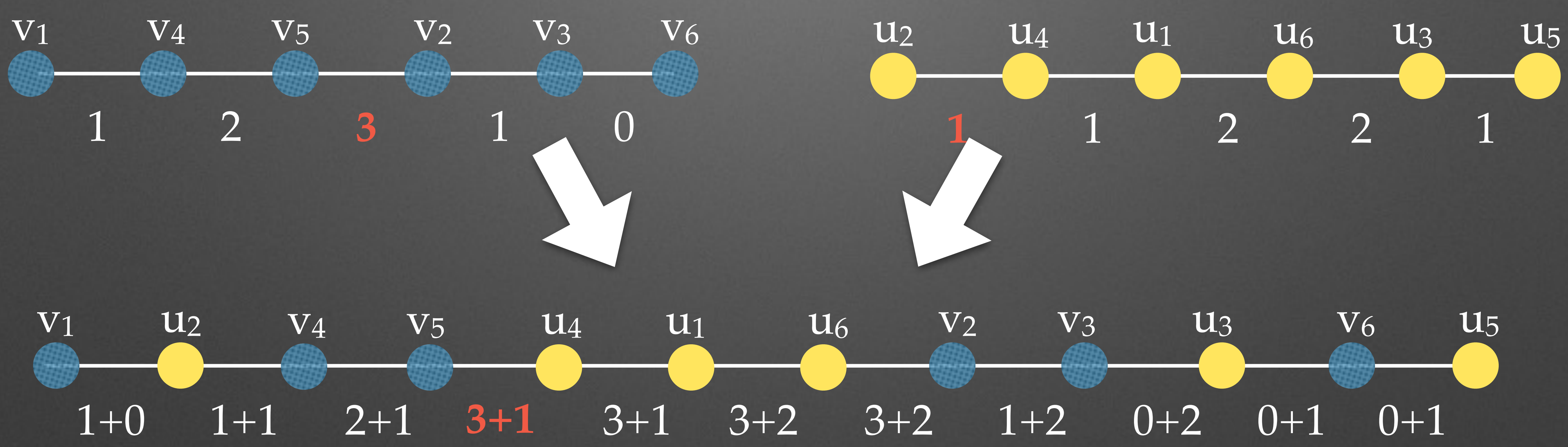
Dynamic programming on a branch-decomposition



- Want to encode all feasible linear layouts of M_1 & M_2
- ...in a compact way
- ...in a way s.t. enconding for $M_1 + M_2$ can be constructed from encodings for M_1 and M_2.

Encoding a linear layout

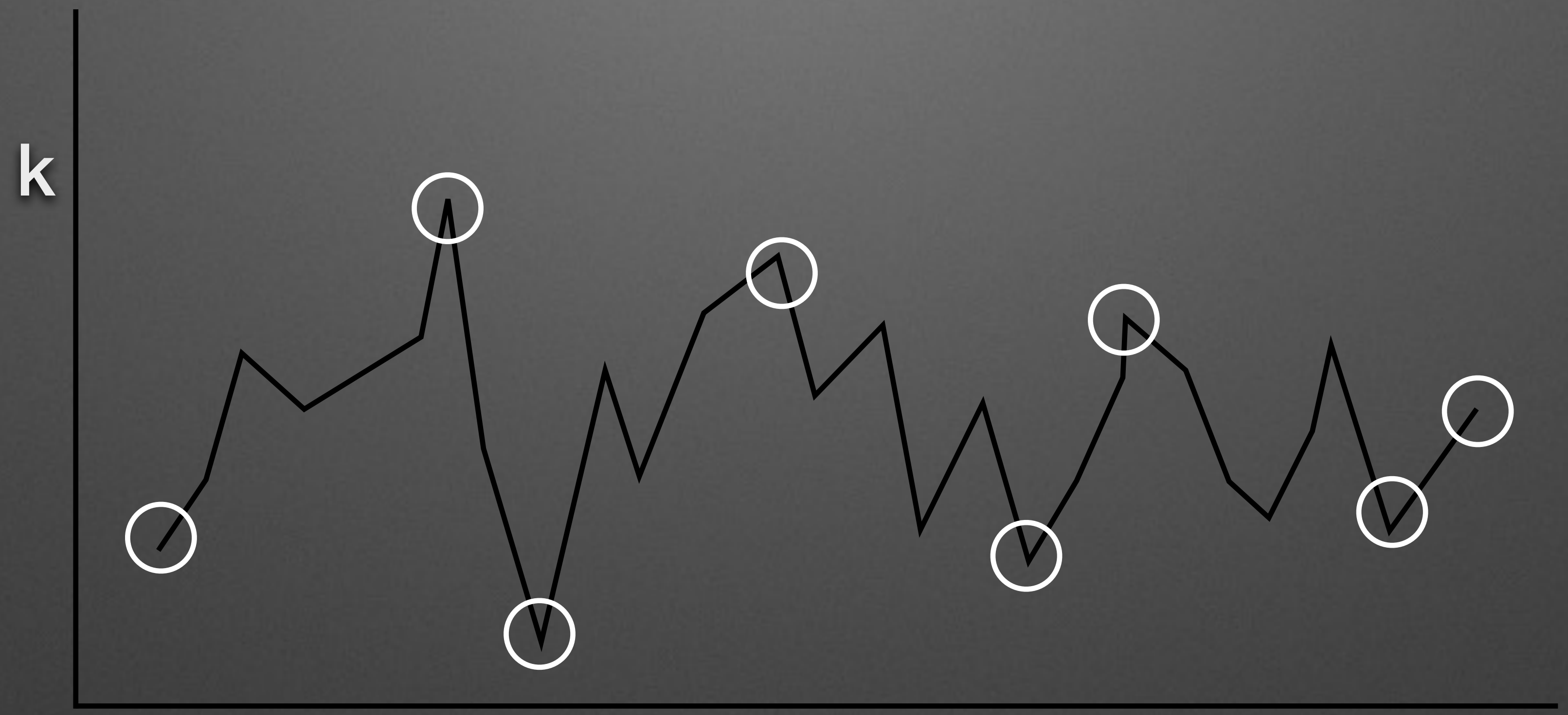
When $\langle M_1 \rangle \cap \langle M_2 \rangle = \emptyset$



Issues

1. How to shorten the length of the dimension sequence
2. How to handle the boundary space; $\langle M_1 \rangle \cap \langle M_2 \rangle \neq \emptyset$

Compressing a dimension sequence

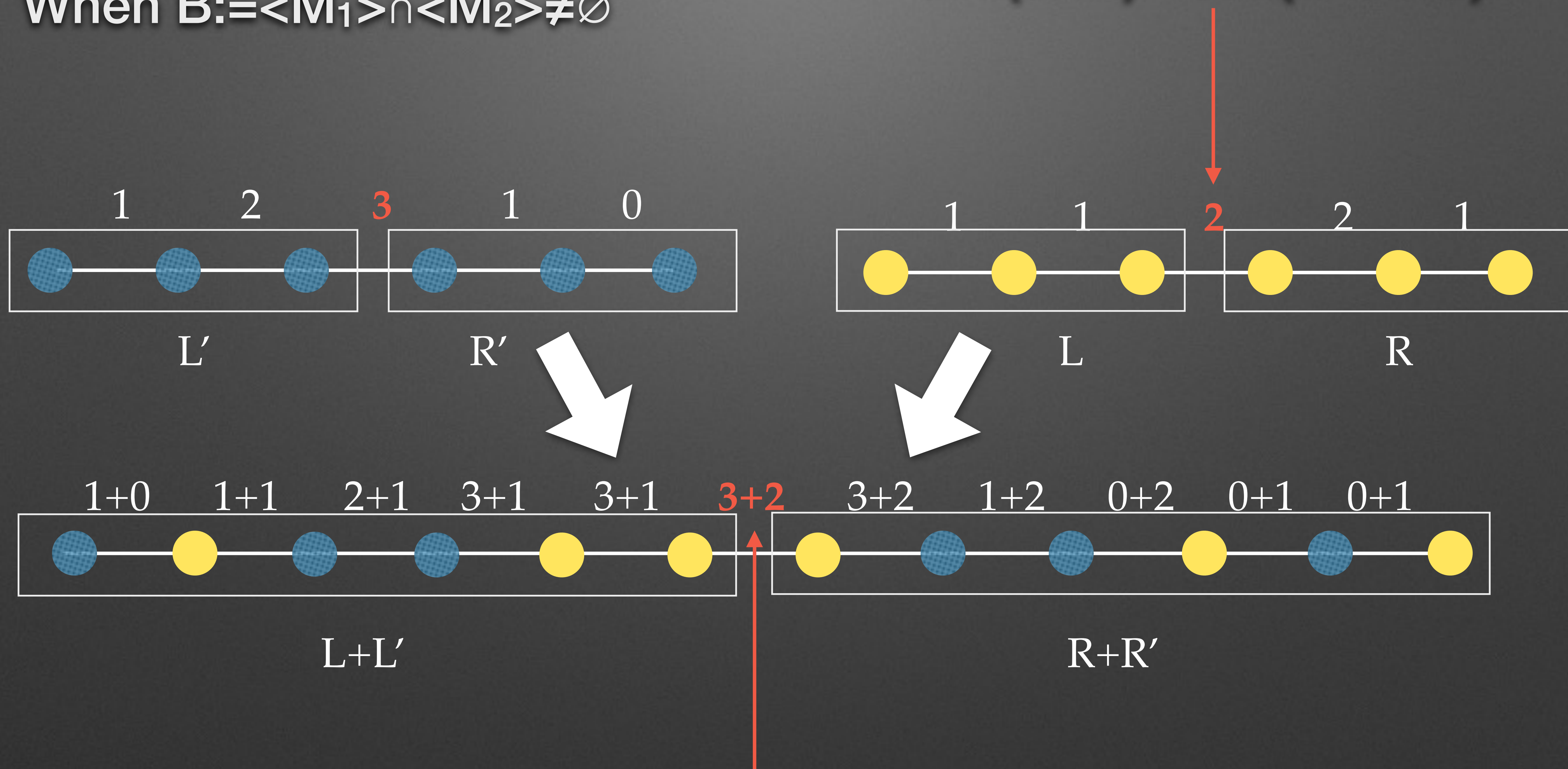


Typical sequence: idea from Bodlaender-Kloks (1996)
#typical sequences consisting of {0,1,2,…,k})≤ $(8/3)2^{2k}$.

Handling a boundary space

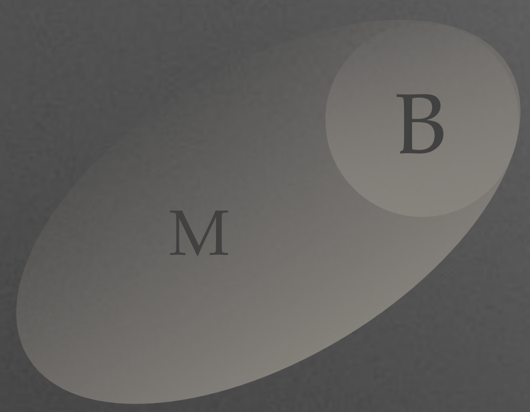
When $B := \langle M_1 \rangle \cap \langle M_2 \rangle \neq \emptyset$

dim(L∩R) - dim(L∩R∩B)

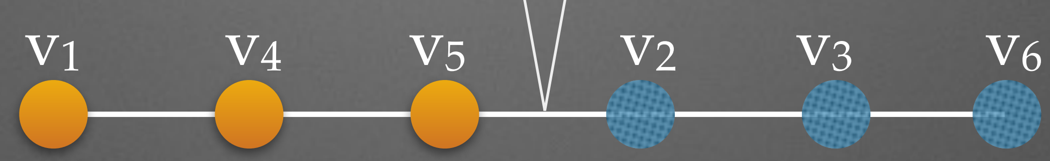


dim(L+L’)∩(R+R’) - dim(L+L’)∩(R+R’)∩B

Encoding of a linear layout



For each "gap", there is a triple (L,R,λ)



L: "Left subspace" shown on B

$L_3 = \langle v_1, v_4, v_5 \rangle \cap B$

R: "Right subspace" shown on B

$R_3 = \langle v_2 + v_3 + v_6 \rangle \cap B$

λ: Extra connectivity not shown in B

$\lambda_3 = \dim \langle v_1, v_4, v_5 \rangle \cap \langle v_2 + v_3 + v_6 \rangle - \dim \langle v_1, v_4, v_5 \rangle \cap \langle v_2 + v_3 + v_6 \rangle \cap B$

Applications

An alternative definition of pathwidth

- A path-decomposition of $\mathbb{F}^r$
  := a sequence $\pi = (S_1,S_2,\ldots,S_m)$ of **subspaces** of $\mathbb{F}^r$
  with an injective function $\mu:\{1,2,\ldots,n\}\to\{1,2,\ldots,m\}$ **s.t.** $V_i \subseteq S_{\mu(i)}$.

- Width of π := $\max (S_1+\ldots+S_i)\cap(S_{i+1}+\ldots+S_m)$

∃ linear layout of width ≤ k

⇕

∃ path-dec. of width≤k.

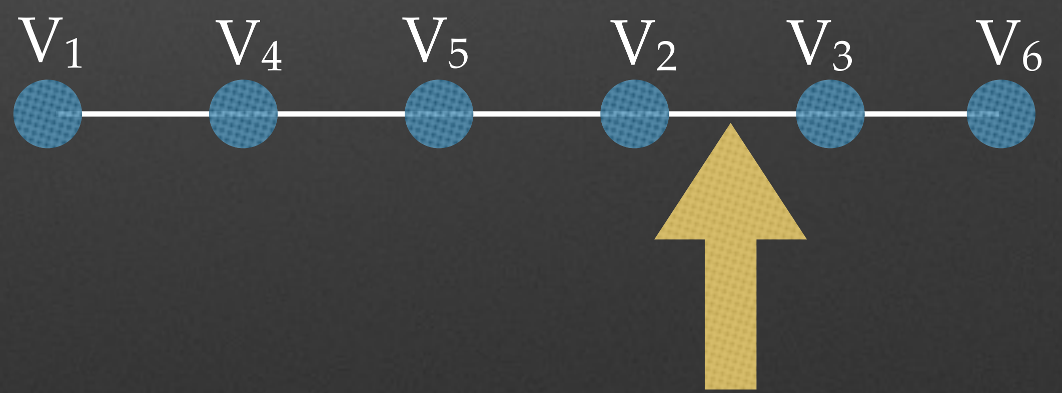
Subspaces Arrangement

$\mathbb{F}$ is a finite field

Input: subspaces V_1, V_2,…V_n of $\mathbb{F}^r$, a positive integer k

Goal: find a permutation of V_1, V_2,…, V_n such that for every i, $\dim(\langle V_1+\ldots+V_i\rangle \cap \langle V_2,\ldots,V_n\rangle) \leq k$.

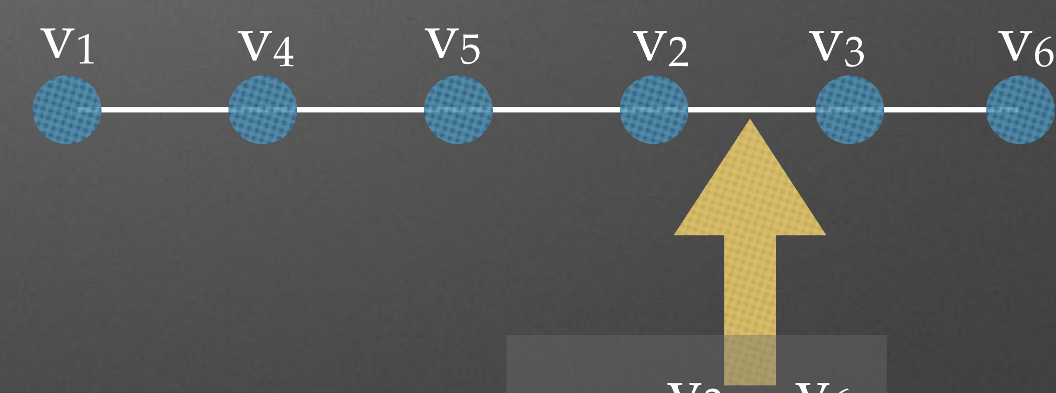
linear layout **of width≤k**



$\dim(\langle V_1,V_4,V_5,V_2\rangle \cap \langle V_3,V_6\rangle) \leq k$

Application to Linear rank-width

- Find a linear ordering of vertices of G so that the rank of adjacency matrix between $\{v_1,\ldots v_i\}$ and $\{v_{i+1},\ldots v_n\}$ is at most k.



rank(

	v_3	v_6
v_1	1	0
v_4	0	1
v_5	1	1
v_2	1	1

) $\leq$ k

Set $V_i := \{e_i , x_i\}$

Path-width of $\{V_1,V_2,\ldots,V_n\}$ = 2*(linear rank-width of G)

Application to Linear rank-width

For any fixed k, $O(n^3)$-time algorithm to find a linear rank decomposition of width ≤ k or confirms that linear rank-width > k.

Application to linear clique-width

- Linear clique-width, “linearized version of clique-width”
- Linear rank-width≤k ➡ linear clique-width ≤ 2^k+1.

For any fixed k, $O(n^3)$-time algorithm to find a linear clique-width expression of width≤2^k+1 or confirms that linear clique-width>k.

Further questions

- FPT algorithms for pathwidth / branchwidth on general matroids? (given independence oracle)
- Can $O(n^3)$ factor in the runtime improved? i.e. $O(n^w)$

THANK YOU!