Constructive algorithm for path-width of matroids

Jisu Jeong (Dept. of Math, KAIST)

joint work with

Eun Jung Kim (CNRS / Univ. Paris-Dauphine), Sang-il Oum (KAIST)

SODA16

2016.1.12 Arlington, USA

Consider the linear code C that is generated by (100001), (010100), and (001010).

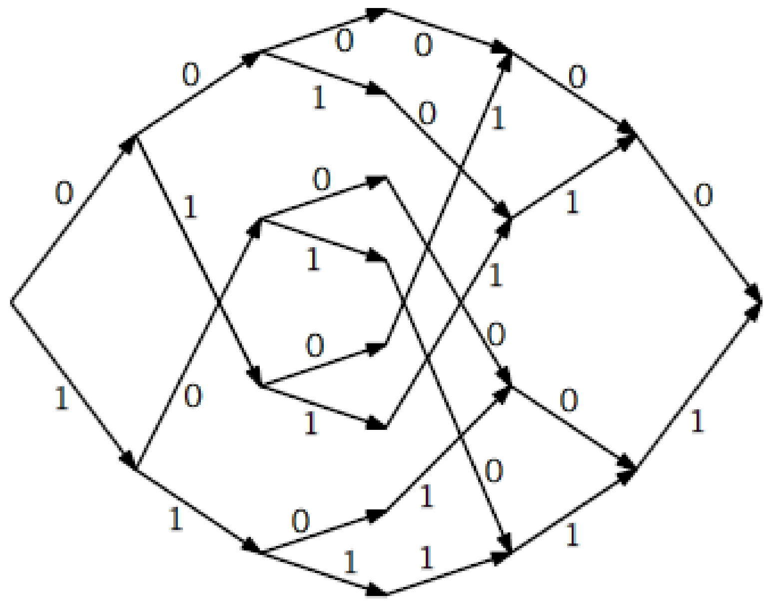
{(000000), (100001), (010100), (001010), (110101), (101011), (011110), (111111)}

The generator matrix is $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$.

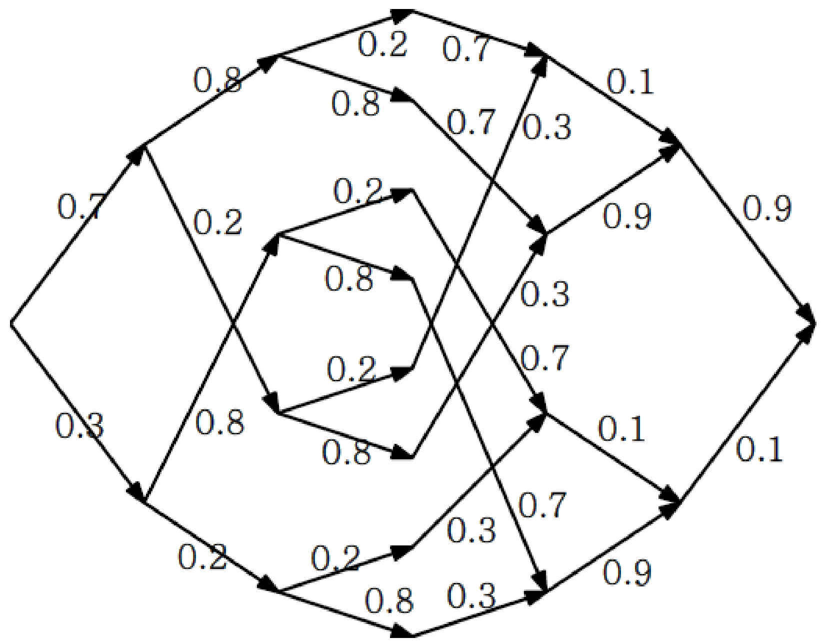
(110101)

(0.7 0.8 0.2 0.7 0.1 0.9)

{(000000), (100001), (010100), (001010), (110101), (101011), (011110), (111111)}

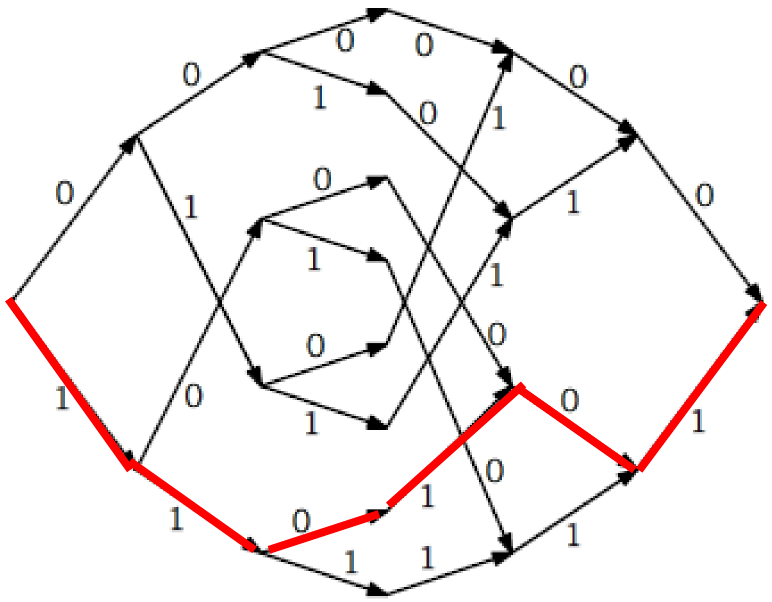


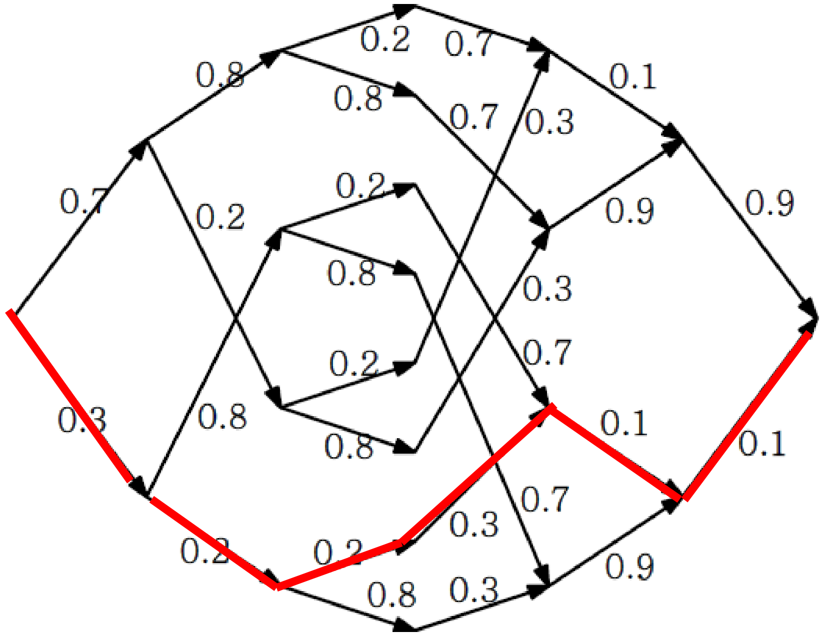
trellis



(0.7 0.8 0.2 0.7 0.1 0.9)

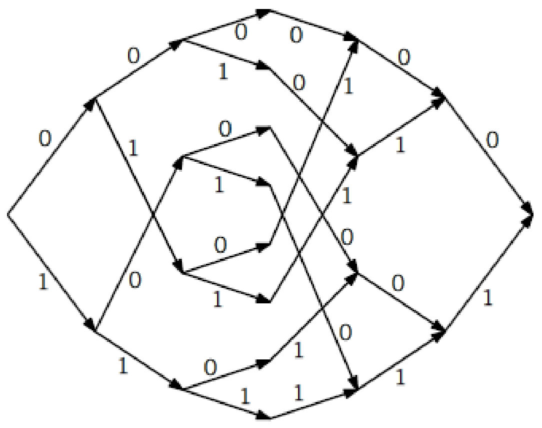
{(000000), (100001), (010100), (001010), (110101), (101011), (011110), (111111)}





(110101)

{(000000), (100001), (010100), (001010), (110101), (101011), (011110), (111111)}



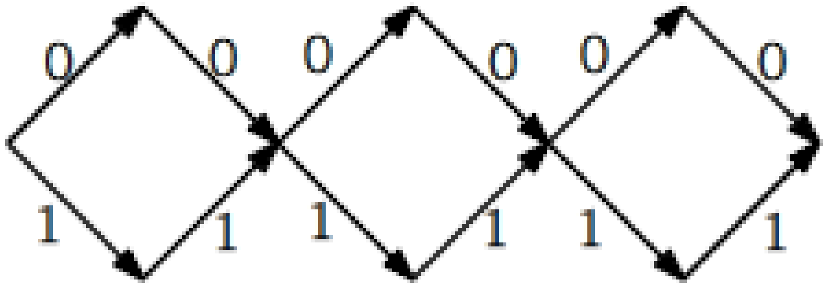
Want to make better trellis

Permute the columns of

$$\begin{pmatrix} 1 & \color{red}{0} & \color{red}{0} & 0 & 0 & \color{red}{1} \\ 0 & \color{red}{1} & \color{red}{0} & 1 & 0 & \color{red}{0} \\ 0 & \color{red}{0} & \color{red}{1} & 0 & 1 & \color{red}{0} \end{pmatrix}$$

$\pi = (1)(4)(5)(2,3,6)$

$$\begin{pmatrix} 1 & \color{red}{1} & \color{red}{0} & 0 & 0 & \color{red}{0} \\ 0 & \color{red}{0} & \color{red}{1} & 1 & 0 & \color{red}{0} \\ 0 & \color{red}{0} & \color{red}{0} & 0 & 1 & \color{red}{1} \end{pmatrix}$$



Use (0.7 0.9 0.8 0.7 0.1 0.2) instead of (0.7 0.8 0.2 0.7 0.1 0.9)

Definition (Path-width)
A **set V of n vectors** over a finite field F has ***path-width at most k*** if there exists a permutation $v_1, v_2, \dots, v_n$ of V satisfying that for all i

$$\dim\langle v_1, v_2, \dots, v_i\rangle \cap \langle v_{i+1}, v_{i+2}, \dots, v_n\rangle \leq k.$$

Note that such permutation is called a ***linear layout of path-width*** $\leq \boldsymbol{k}$.

$$\begin{array}{ccc|ccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array}$$
$$\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{array}\right)$$

3

$$\begin{array}{ccc|ccc} v_1 & v_6 & v_2 & v_4 & v_5 & v_3 \end{array}$$
$$\left(\begin{array}{ccc|ccc} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right)$$

1

Note that since we only consider `F-representable matroids' with a fixed finite field F, we can say that a **F-representable matroid** is a **set of vectors** over F.

Path-width Problem

Input : a set V of n vectors over F

Parameter : a nonnegative integer k

Output : a linear layout $v_1, v_2, \ldots, v_n$ of V satisfying that for all i

$$\dim\langle v_1, v_2, \ldots, v_i\rangle \cap \langle v_{i+1}, v_{i+2}, \ldots, v_n\rangle \leq k$$

if it exists.

Path-width Problem (decision version)

Input : a set V of n vectors over F

Parameter : a nonnegative integer k

Output : YES if there exists a linear layout $v_1, v_2, \ldots, v_n$ of V satisfying that for all i

$$\dim\langle v_1, v_2, \ldots, v_i\rangle \cap \langle v_{i+1}, v_{i+2}, \ldots, v_n\rangle \leq k$$

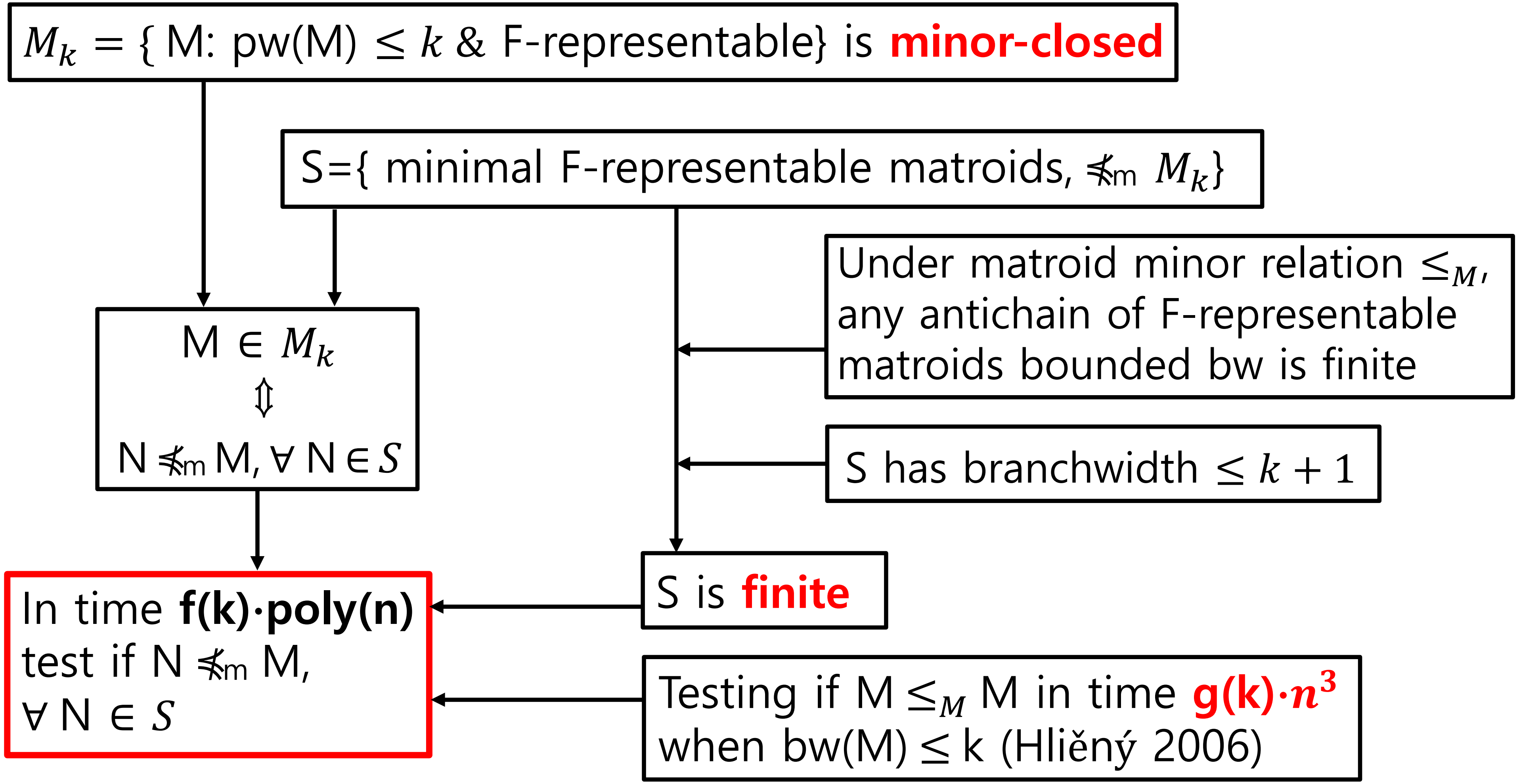
NO otherwise.

Note that this problem is NP-complete [Kashyap 2008].

Our algorithm is **FPT** algorithm.

Roughly speaking, we say **FPT** if the time complexity is $f(k) \cdot poly(n)$.

Decision FPT algorithm for an F-representable matroid path-width $\leq k$



Our results

Constructive algorithm for path-width of n subspaces

Input : n subspaces
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

$V_1, V_2, \cdots, V_n$
where V_i =span$(v_1, v_2, \cdots, v_j)$

Constructive algorithm for path-width of n vectors

Input : n vectors
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

$v_1, v_2, \cdots, v_n$

Our results

Constructive algorithm for path-width of n subspaces

Input : n subspaces
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

$V_1, V_2, \cdots, V_n$
where V_i =span$(v_1, v_2, \cdots, v_j)$

Constructive algorithm for trellis-width of linear codes

Input : linear code
Output : a linear layout of trellis-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Linear code whose generator matrix is
$(v_1 \quad v_2 \quad \cdots \quad v_n)$

Constructive algorithm for path-width of matroids

Input : matroid(F-representable)
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Matroid represented by
$v_1, v_2, \cdots, v_n$

Our results

Constructive algorithm for path-width of n subspaces

Input : n subspaces
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Constructive algorithm for trellis-width of linear codes

Input : linear code
Output : a linear layout of trellis-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Constructive algorithm for path-width of matroids

Input : matroid(F-representable)
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Constructive algorithm for *linear rank-width* of graphs

Input : graph
Output : a linear layout of linear rank-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Rank-width is a width-parameter introduced by Oum and Seymour, which is equivalent to **clique-width**.

Our results

Constructive algorithm for path-width of n subspaces

Input : n subspaces
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Constructive algorithm for trellis-width of linear codes

Input : linear code
Output : a linear layout of trellis-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Constructive algorithm for path-width of matroids

Input : matroid(F-representable)
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Constructive algorithm for linear rank-width of graphs

Input : graph
Output : a linear layout of linear rank-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Exact algorithm for path-width of matroids

Input : matroid (F-representable, **bounded branch-width**)
Output : path-width of given matroid
Time : poly(n)

Exact algorithm for linear rank-width of graphs

Input : graph of **bounded rw**
Output : linear rank-width of given graph
Time : poly(n)

Approximation algorithm for linear clique-width of graphs

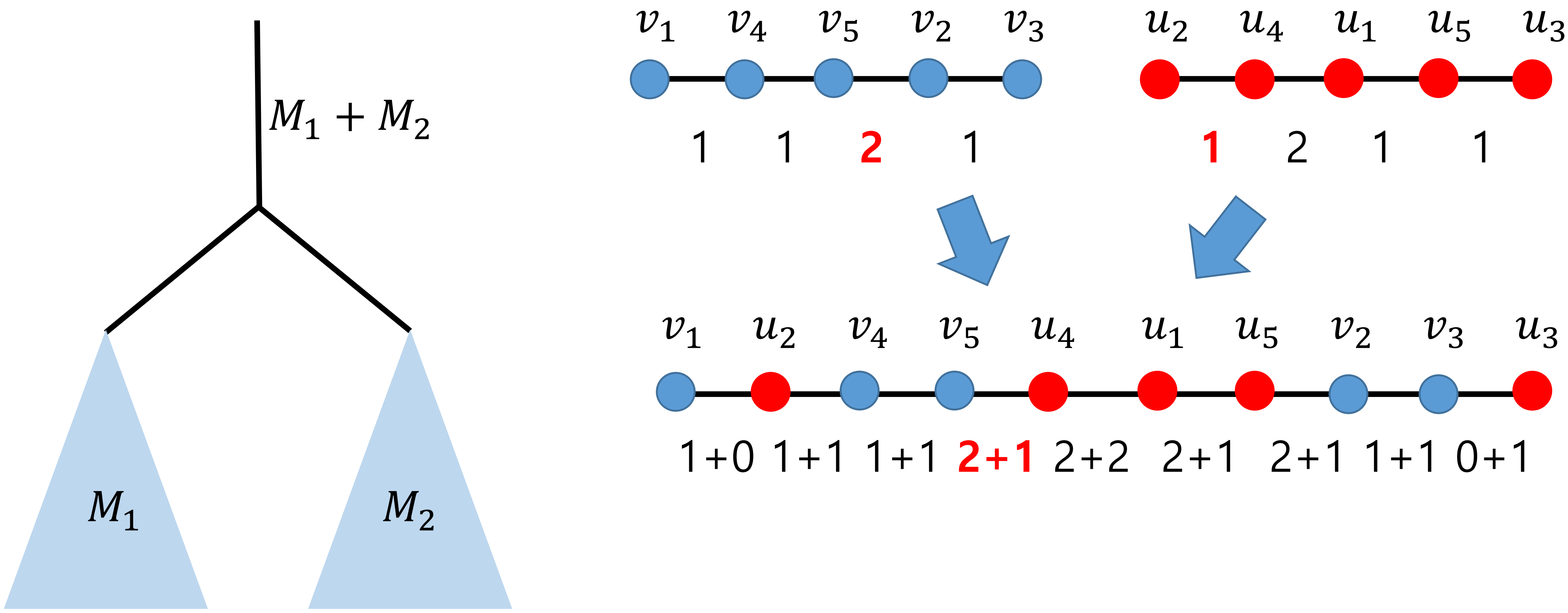
Input : graph
Output : linear $(2^k + 1)$-expression of given graph
Time : $f(k) \cdot n^3$

Proof ideas

1. Dynamic programming   2. Typical sequence   3. Subspace analysis (linear algebra)

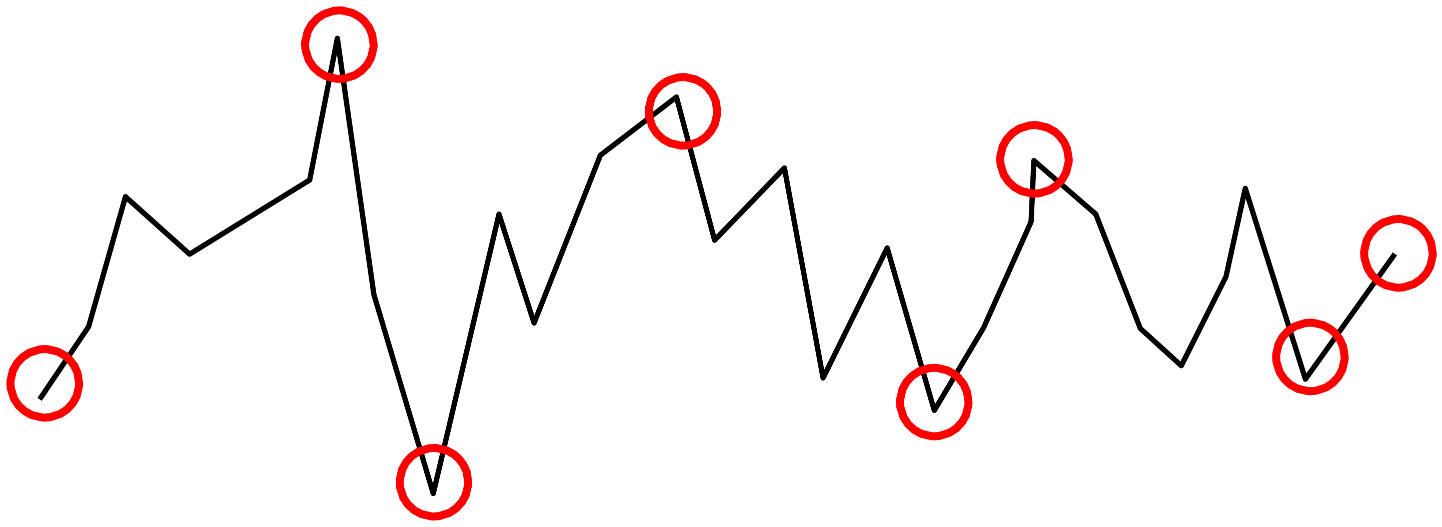
Proof ideas

1. Dynamic programming



Proof ideas

2. Typical sequence



3 6 9 0 5 4 8 5 6 2   3 9 0 5 4 8 5 6 2   3 9 0 8 5 6 2   3 9 0 8 2

Constructive algorithm for path-width of n subspaces

Input : n subspaces
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Constructive algorithm for path-width of matroids

Input : matroid(F-representable)
Output : a linear layout of path-width $\leq k$ if it exists
Time : $f(k) \cdot n^3$

Further questions

1. FPT algorithms for path-width of general matroids
2. Can $O(n^3)$ factor in the running time improved?
   e.g. $O(n^w)$ (w=matrix multiplication exponent)