Mixed minors, compact representations and χ-boundedness

Marek Sokołowski

25 May 2023

Part 1

Mixed minors

Twin-width and matrices

A 0/1-matrix can be:

Twin-width and matrices

A 0/1-matrix can be:

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{matrix}$$

horizontal

Twin-width and matrices

A 0/1-matrix can be:

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{matrix} \qquad \begin{matrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{matrix}$$

horizontal vertical

Twin-width and matrices

A 0/1-matrix can be:

0	0	0	0
1	1	1	1
1	1	1	1
0	0	0	0
1	1	1	1

horizontal

0	1	1	0
0	1	1	0
0	1	1	0
0	1	1	0
0	1	1	0

vertical

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

constant

Twin-width and matrices

A 0/1-matrix can be:

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{matrix} \qquad \begin{matrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

horizontal   vertical   constant   mixed

Twin-width and matrices

A 0/1-matrix can be:

$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{matrix}$	$\begin{matrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{matrix}$	$\begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$	$\begin{matrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$
horizontal	vertical	constant	mixed

Twin-width and matrices

A 0/1-matrix can be:

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{matrix} \qquad \begin{matrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

horizontal | vertical | constant | mixed

Note: mixed $\iff$ has a 2×2 contiguous mixed submatrix (**corner**).

Twin-width and matrices

Divisions

Division $\mathcal{D}$ – partitioning of columns and rows into intervals (*blocks*).

$$
\begin{array}{cc|ccc|cccc}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}
$$

Twin-width and matrices

Divisions

Division $\mathcal{D}$ – partitioning of columns and rows into intervals (*blocks*).

$$\begin{array}{cc|ccc|cccc}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}$$

Zone – intersection of a row block and a column block

Twin-width and matrices

Divisions

Division $\mathcal{D}$ – partitioning of columns and rows into intervals (*blocks*).

$$\begin{array}{cc|ccc|cccc}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}$$

Zone – intersection of a row block and a column block

Mixed minors

$\mathcal{D}$ is a **mixed minor** if each zone of $\mathcal{D}$ is mixed.

Twin-width and matrices

Divisions

Division $\mathcal{D}$ – partitioning of columns and rows into intervals (*blocks*).

$$
\begin{array}{cc|ccc|cccc}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}
$$

Zone – intersection of a row block and a column block

Mixed minors

$\mathcal{D}$ is a **mixed minor** if each zone of $\mathcal{D}$ is mixed.

Mixed freeness

Matrix M is ***d*-mixed free** if it has no $d \times d$ mixed minor.

Grid theorem for twin-width

Mixed freeness

Matrix M is d-**mixed free** if it has no $d \times d$ mixed minor.

Theorem (Twin-width I)

Let $d \in \mathbb{N}$ be an integer and G be a graph. Then:

- $\operatorname{tww}(G) \leq d \implies G$ has a $(2d+2)$-**mixed free** adjacency matrix.
- G has a d-**mixed free** adjacency matrix $\implies \operatorname{tww}(G) \leq 2^{2^{O(d)}}$.

Marcus–Tardos theorem and twin-width

Mixed freeness

Matrix M is ***d*-mixed free** if it has no $d \times d$ mixed minor.

Theorem (Twin-width I, "Marcus–Tardos")

If: M – a d-mixed free matrix,
$\mathcal{D}$ – an $n \times n$ division of M

$\Rightarrow \mathcal{D}$ has at most $c_d \cdot n$ mixed zones ($c_d = \mathrm{const}(d)$).

Marcus–Tardos theorem and twin-width

Mixed freeness

Matrix M is **d-mixed free** if it has no $d \times d$ mixed minor.

Theorem (Twin-width I, "Marcus–Tardos")

If: M – a d-mixed free matrix,
$\mathcal{D}$ – an $n \times n$ division of M

$\Rightarrow \mathcal{D}$ has at most $c_d \cdot n$ mixed zones ($c_d = \text{const}(d)$).

Number of mixed zones: **linear** instead of **quadratic**!

Part 2

Compact representations

Pilipczuk, Sokołowski, Zych-Pawlewicz,
Compact Representation for Matrices of Bounded Twin-Width

Twin-width of matrices

Twin-width of matrices

t-twin-ordered matrices

0	1	1	0	0	0
1	0	1	1	1	0
0	0	1	0	0	0
0	0	1	0	0	1
1	1	1	1	1	1
0	0	0	1	1	1

Maximum $\times$'s in any row/column now: 0

Maximum so far: 0

Twin-width of matrices

t-twin-ordered matrices

0	1	1	0	0	0
1	0	1	1	1	0
0	0	1	0	0	0
0	0	1	0	0	1
1	1	1	1	1	1
0	0	0	1	1	1

Maximum $\times$'s in any row/column now: 0

Maximum so far: 0

Twin-width of matrices

t-twin-ordered matrices

0	1	1	0	0
1	0	1	1	0
0	0	1	0	0
0	0	1	0	1
1	1	1	1	1
0	0	0	1	1

Maximum $\times$'s in any row/column now: 0

Maximum so far: 0

Twin-width of matrices

t-twin-ordered matrices

0	1	1	0	0
1	0	1	1	0
0	0	1	0	0
0	0	1	0	1
1	1	1	1	1
0	0	0	1	1

Maximum $\times$'s in any row/column now: 0

Maximum so far: 0

Twin-width of matrices

t-twin-ordered matrices

0	1	1	0	0
1	0	1	1	0
0	0	1	0	$\times$
1	1	1	1	1
0	0	0	1	1

Maximum $\times$'s in any row/column now: 1

Maximum so far: 1

Twin-width of matrices

t-twin-ordered matrices

0	1	1	0	0
1	0	1	1	0
0	0	1	0	$\times$
1	1	1	1	1
0	0	0	1	1

Maximum $\times$'s in any row/column now: 1
Maximum so far: 1

Twin-width of matrices

t-twin-ordered matrices

0	1	1	0	0
1	0	1	1	0
0	0	1	0	×
1	1	1	1	1
0	0	0	1	1

Maximum ×'s in any row/column now: 1

Maximum so far: 1

Twin-width of matrices

t-twin-ordered matrices

×	1	0	0
×	1	1	0
0	1	0	×
1	1	1	1
0	0	1	1

Maximum ×'s in any row/column now: 2

Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

$\times$	1	0	0
$\times$	1	1	0
0	1	0	$\times$
1	1	1	1
0	0	1	1

Maximum $\times$'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

×	1	×	0
0	1	0	×
1	1	1	1
0	0	1	1

Maximum ×'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

×	1	×	0
0	1	0	×
1	1	1	1
0	0	1	1

Maximum ×'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

$\times$	$\times$	0
$\times$	0	$\times$
1	1	1
0	1	1

Maximum $\times$'s in any row/column now: 2

Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

$\times$	$\times$	0
$\times$	0	$\times$
1	1	1
0	1	1

Maximum $\times$'s in any row/column now: 2

Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

$\times$	$\times$
$\times$	$\times$
1	1
0	1

Maximum $\times$'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

$\times$	$\times$
$\times$	$\times$
1	1
0	1

Maximum $\times$'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

×	×
1	1
0	1

Maximum ×'s in any row/column now: 2

Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

×	×
1	1
0	1

Maximum ×'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices

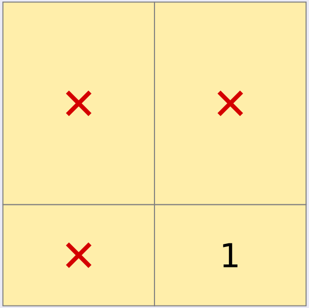
×	×
×	1

Maximum ×'s in any row/column now: 2

Maximum so far: 2

Twin-width of matrices

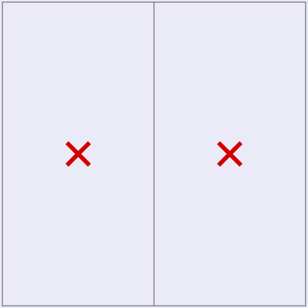
t-twin-ordered matrices



Maximum $\times$'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

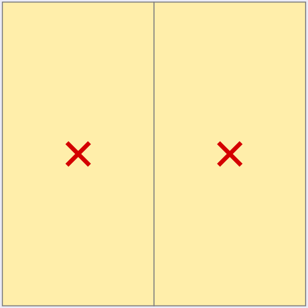
t-twin-ordered matrices



Maximum $\times$'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

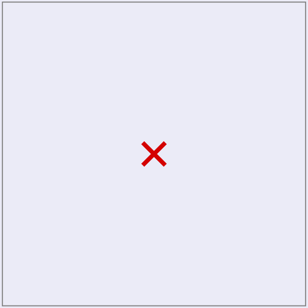
t-twin-ordered matrices



Maximum $\times$'s in any row/column now: 2
Maximum so far: 2

Twin-width of matrices

t-twin-ordered matrices



Maximum $\times$'s in any row/column now: 1

Maximum so far: 2

Twin-width of matrices

Red number of a contraction sequence

Red number = maximum $\times$'s in any row/column during the contraction.

Twin-width of matrices

Red number of a contraction sequence

Red number = maximum $\times$'s in any row/column during the contraction.

Twin-width of a matrix

M is d-**twin-ordered** $\iff$ M has a contraction with red number $\leq d$.

Twin-width of matrices

Red number of a contraction sequence

Red number = maximum $\times$'s in any row/column during the contraction.

Twin-width of a matrix

M is **d-twin-ordered** $\iff M$ has a contraction with red number $\leq d$.

$\operatorname{tww}(M) \leq d \iff M$ is **d-twin-ordered** for some permutation of rows/columns.

Twin-width of matrices

Red number of a contraction sequence

Red number $=$ maximum $\times$'s in any row/column during the contraction.

Twin-width of a matrix

M is d-**twin-ordered** $\iff$ M has a contraction with red number $\leq d$.

$\text{tww}\,(M) \leq d \iff M$ is d-**twin-ordered** for some permutation of rows/columns.

Note: M is d-**twin-ordered** $\implies$ M is $(2d+2)$-**mixed-free**.

Compact data structures

Objectives

A *compact data structure* for a dynamic problem should:

Compact data structures

Objectives

A *compact data structure* for a dynamic problem should:

- consume little space: closer to the optimum = better,

Compact data structures

Objectives

A *compact data structure* for a dynamic problem should:

- consume little space: closer to the optimum = better,
- answer queries efficiently,

Compact data structures

Objectives

A *compact data structure* for a dynamic problem should:

- consume little space: closer to the optimum $=$ better,
- answer queries efficiently,
- *(preferably)* be constructed efficiently.

Compact data structures

Objectives

A *compact data structure* for a dynamic problem should:

- consume little space: closer to the optimum = better,
- answer queries efficiently,
- *(preferably)* be constructed efficiently.

Our case:

- Given: M – a d-**twin-ordered** matrix
- Want: to query for entries of M

Compact data structures

Objectives

A *compact data structure* for a dynamic problem should:

- consume little space: closer to the optimum = better,
- answer queries efficiently,
- *(preferably)* be constructed efficiently.

Our case:

- Given: M – a d-**twin-ordered** matrix
- Want: to query for entries of M

Compact: bitsize $\mathcal{O}(S)$ bits if S = information-theoretic min bitsize.

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

$\approx$ Twin-width II

The number of binary d-**twin-ordered** $n \times n$ matrices is $2^{\Theta_d(n)}$.

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

$\approx$ Twin-width II

The number of binary d-**twin-ordered** $n \times n$ matrices is $2^{\Theta_d(n)}$.

$\implies$ Bitsize $\Omega_d(n)$ is required.

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

$\approx$ Twin-width II

The number of binary d-**twin-ordered** $n \times n$ matrices is $2^{\Theta_d(n)}$.

$\implies$ Bitsize $\Omega_d(n)$ is required.

	Bitsize	**Query time**
just store the matrix	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

≈ Twin-width II

The number of binary d-**twin-ordered** $n \times n$ matrices is $2^{\Theta_d(n)}$.

$\Longrightarrow$ Bitsize $\Omega_d(n)$ is required.

	Bitsize	**Query time**
just store the matrix	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
store the idx of d-twin-ordered matrix	$\mathcal{O}_d(n)$	*huge*

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

$\approx$ Twin-width II

The number of binary d-**twin-ordered** $n \times n$ matrices is $2^{\Theta_d(n)}$.

$\implies$ Bitsize $\Omega_d(n)$ is required.

	Bitsize	**Query time**
just store the matrix	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
store the idx of d-twin-ordered matrix	$\mathcal{O}_d(n)$	*huge*
adjacency labeling (Twin-width II)	$\mathcal{O}_d(n \log n)$	$\mathcal{O}_d(\log n)$

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

≈ Twin-width II

The number of binary d-**twin-ordered** $n \times n$ matrices is $2^{\Theta_d(n)}$.

$\implies$ Bitsize $\Omega_d(n)$ is required.

	Bitsize	**Query time**
just store the matrix	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
store the idx of d-twin-ordered matrix	$\mathcal{O}_d(n)$	*huge*
adjacency labeling (Twin-width II)	$\mathcal{O}_d(n \log n)$	$\mathcal{O}_d(\log n)$
Orthogonal Point Location (Chan, 2013)	$\mathcal{O}_d(n \log n)$	$\mathcal{O}_d(\log \log n)$

Given: M – a d-**twin-ordered** matrix
Want: to query for entries of M compactly

≈ Twin-width II

The number of binary d-**twin-ordered** $n \times n$ matrices is $2^{\Theta_d(n)}$.

$\Longrightarrow$ Bitsize $\Omega_d(n)$ is required.

	Bitsize	**Query time**
just store the matrix	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
store the idx of d-twin-ordered matrix	$\mathcal{O}_d(n)$	*huge*
adjacency labeling (Twin-width II)	$\mathcal{O}_d(n \log n)$	$\mathcal{O}_d(\log n)$
Orthogonal Point Location (Chan, 2013)	$\mathcal{O}_d(n \log n)$	$\mathcal{O}_d(\log \log n)$
our result (PSZ-P, 2022)	$\mathcal{O}_d(n)$	$\mathcal{O}_d(\log \log n)$

Different zones in a division

M — a d-twin-ordered $n \times n$ matrix;

$s \mid n$;

$\mathcal{D}$ — an $\frac{n}{s} \times \frac{n}{s}$ division of M where each zone is an $s \times s$ submatrix.

1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1
0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1

Different zones in a division

M — a d-twin-ordered $n \times n$ matrix;

$s \mid n$;

$\mathcal{D}$ — an $\frac{n}{s} \times \frac{n}{s}$ division of M where each zone is an $s \times s$ submatrix.

Small s

$\mathcal{D}$ has at most $2^{\mathcal{O}_d(s)}$ different zones *(Twin-width II)*.

Different zones in a division

M — a d-twin-ordered $n \times n$ matrix;

$s \mid n$;

$\mathcal{D}$ — an $\frac{n}{s} \times \frac{n}{s}$ division of M where each zone is an $s \times s$ submatrix.

Small s

$\mathcal{D}$ has at most $2^{\mathcal{O}_d(s)}$ different zones *(Twin-width II)*.

$\Longrightarrow$ for $s \ll \log n$, at most $\sqrt{n}$ different matrices of size s.

Different zones in a division

M — a d-twin-ordered $n \times n$ matrix;

$s \mid n$;

$\mathcal{D}$ — an $\frac{n}{s} \times \frac{n}{s}$ division of M where each zone is an $s \times s$ submatrix.

Small s

$\mathcal{D}$ has at most $2^{\mathcal{O}_d(s)}$ different zones *(Twin-width II)*.

$\Longrightarrow$ for $s \ll \log n$, at most $\sqrt{n}$ different matrices of size s.

Large s

We prove: $\mathcal{D}$ has at most $\mathcal{O}_d(\frac{n}{s})$ different zones.

Different zones in a division

M — a d-twin-ordered $n \times n$ matrix;

$s \mid n$;

$\mathcal{D}$ — an $\frac{n}{s} \times \frac{n}{s}$ division of M where each zone is an $s \times s$ submatrix.

Small s

$\mathcal{D}$ has at most $2^{\mathcal{O}_d(s)}$ different zones *(Twin-width II)*.

$\Longrightarrow$ for $s \ll \log n$, at most $\sqrt{n}$ different matrices of size s.

Large s

We prove: $\mathcal{D}$ has at most $\mathcal{O}_d(\frac{n}{s})$ different zones.

- "Marcus–Tardos": at most $\mathcal{O}_d(\frac{n}{s})$ **mixed** zones in total;

Different zones in a division

M — a d-twin-ordered $n \times n$ matrix;

$s \mid n$;

$\mathcal{D}$ — an $\frac{n}{s} \times \frac{n}{s}$ division of M where each zone is an $s \times s$ submatrix.

Small s

$\mathcal{D}$ has at most $2^{\mathcal{O}_d(s)}$ different zones *(Twin-width II)*.

$\implies$ for $s \ll \log n$, at most $\sqrt{n}$ different matrices of size s.

Large s

We prove: $\mathcal{D}$ has at most $\mathcal{O}_d(\frac{n}{s})$ different zones.

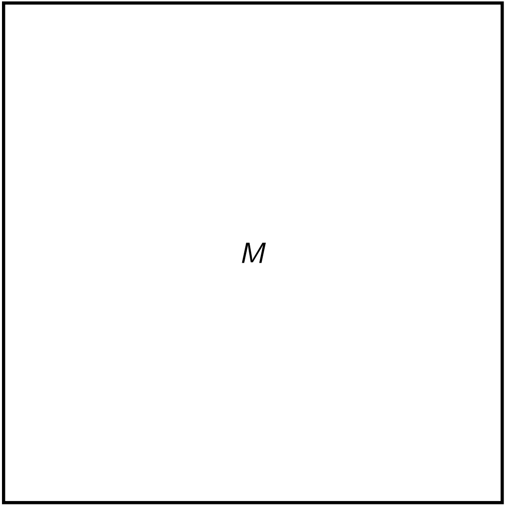
- "Marcus–Tardos": at most $\mathcal{O}_d(\frac{n}{s})$ **mixed** zones in total;
- Now (blackboard): at most $\mathcal{O}_d(\frac{n}{s})$ different **non-mixed** zones.

Data structure

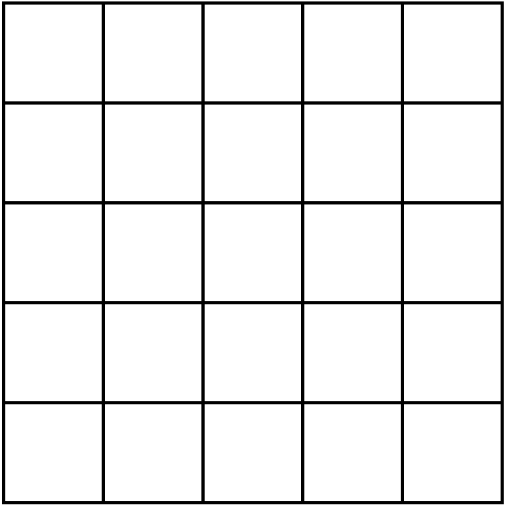
Reminder

- Fixed: $d \in \mathbb{N}$.
- Input: M – an $n \times n$ matrix that is **d-twin-ordered**.
- Target:

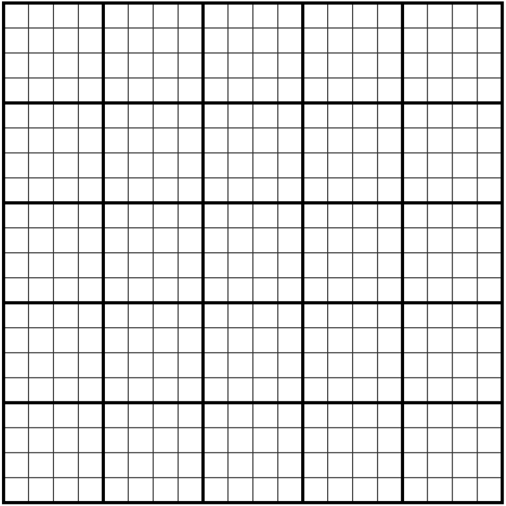
  $\mathcal{O}_d(n)$ bits of memory,
  $\mathcal{O}(\log \log n)$ per query.


M

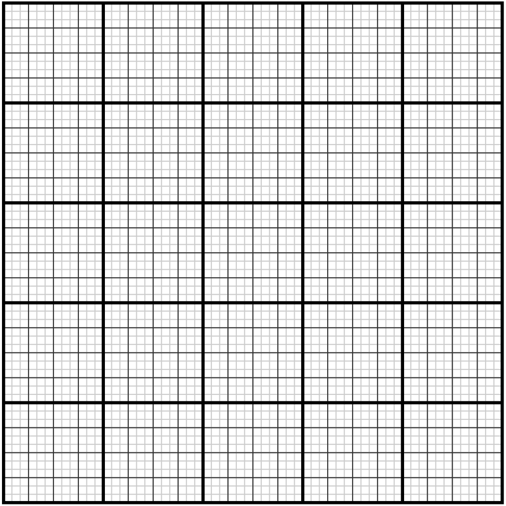
$\mathcal{D}_1$ – a division of M where each zone is an $n^{2/3} \times n^{2/3}$ submatrix.



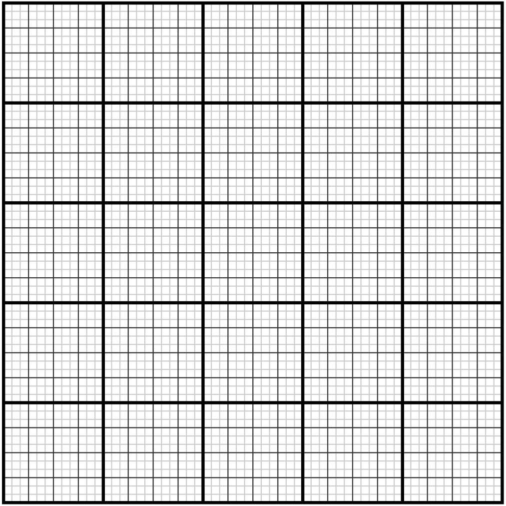
$\mathcal{D}_1$ – a division of M where each zone is an $n^{2/3} \times n^{2/3}$ submatrix.
$\mathcal{D}_2$ – a division of M where each zone is an $n^{4/9} \times n^{4/9}$ submatrix.



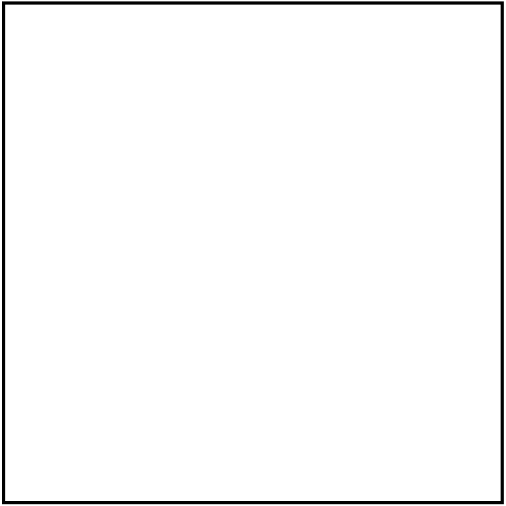
$\mathcal{D}_1$ – a division of M where each zone is an $n^{2/3} \times n^{2/3}$ submatrix.
$\mathcal{D}_2$ – a division of M where each zone is an $n^{4/9} \times n^{4/9}$ submatrix.
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.



$\mathcal{D}_1$ – a division of M where each zone is an $n^{2/3} \times n^{2/3}$ submatrix.
$\mathcal{D}_2$ – a division of M where each zone is an $n^{4/9} \times n^{4/9}$ submatrix.
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
$\mathcal{D}_{\text{last}}$ – a division where each zone is an $\mathcal{O}_d(\log n) \times \mathcal{O}_d(\log n)$ submatrix.

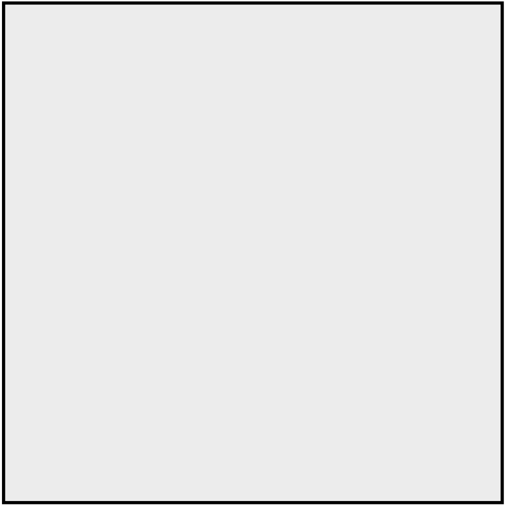


$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Next, mark unique zones in each division.

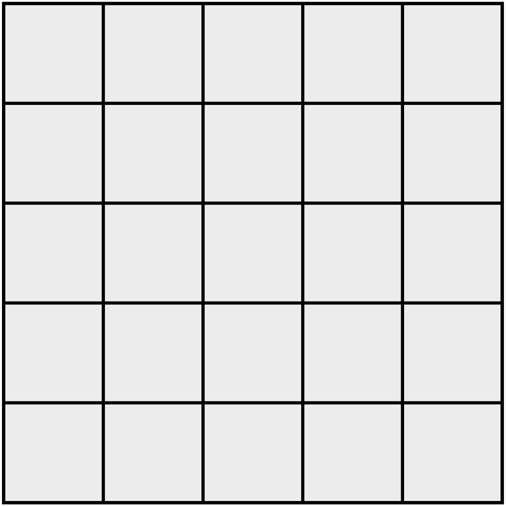


$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

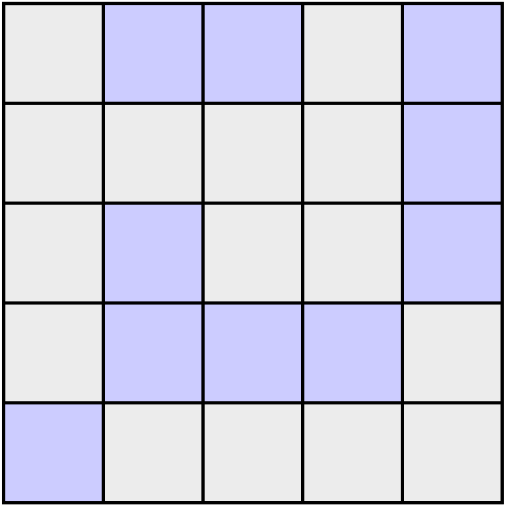
Next, mark unique zones in each division.



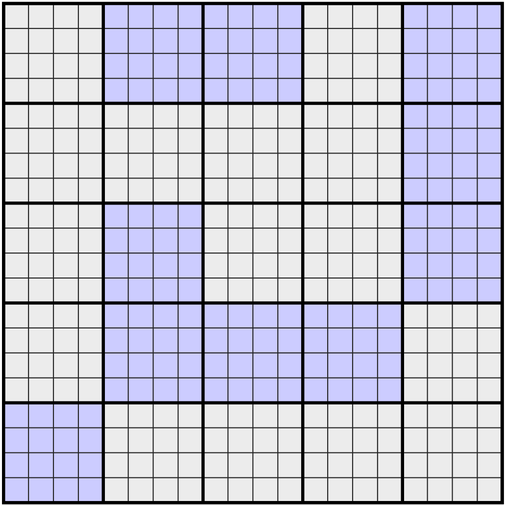
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Next, mark unique zones in each division.



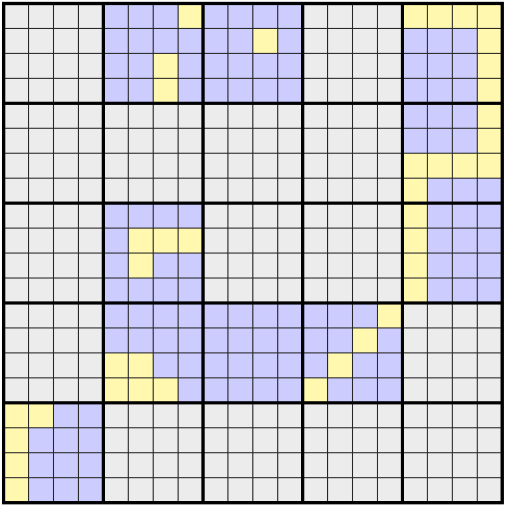
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Next, mark unique zones in each division.



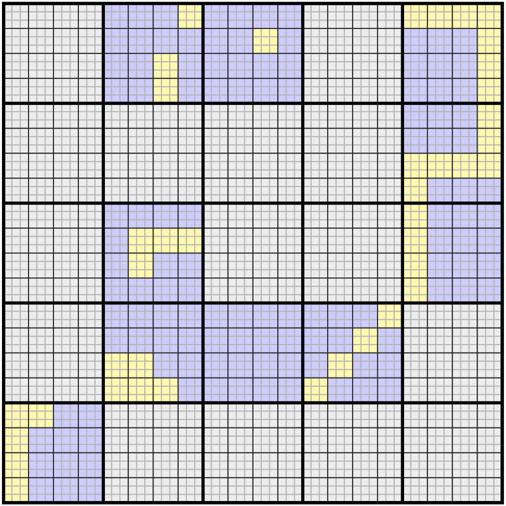
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Next, mark unique zones in each division.



$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Next, mark unique zones in each division.

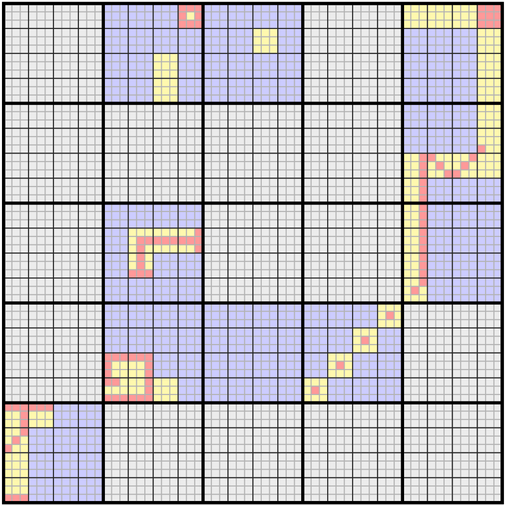


$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Next, mark unique zones in each division.



$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

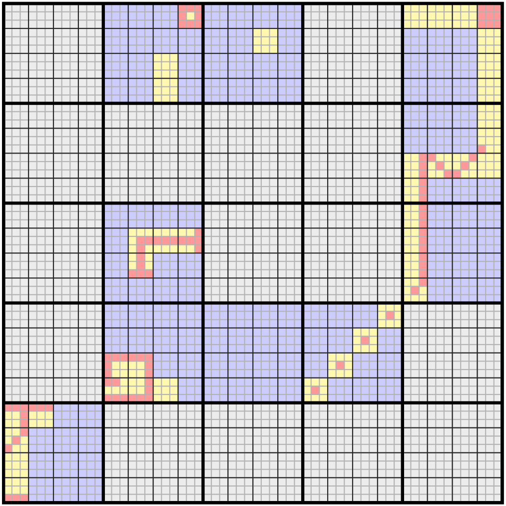
Next, mark unique zones in each division.



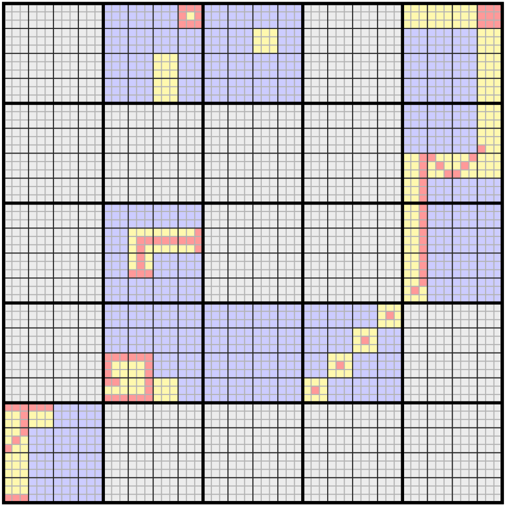
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

Next, mark unique zones in each division.

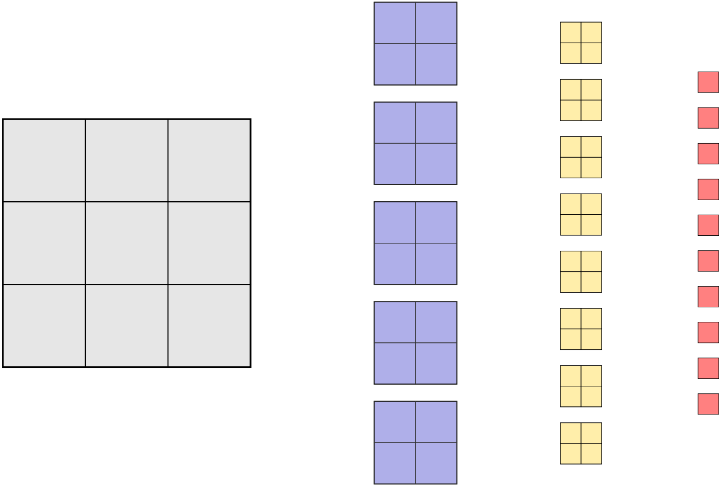
$\mathcal{D}_k$ has at most $\mathcal{O}_d(n/n^{(2/3)^k})$ unique zones. $(n/n,\ n/n^{2/3},\ n/n^{4/9},\ \ldots)$



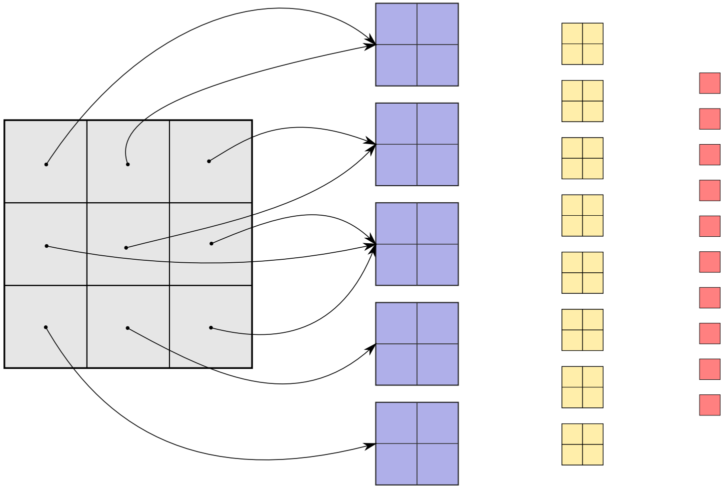
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Next, mark unique zones in each division.
$\mathcal{D}_k$ has at most $\mathcal{O}_d(n/n^{(2/3)^k})$ unique zones. $(n/n,\ n/n^{2/3},\ n/n^{4/9},\ \ldots)$
But: $\mathcal{D}_{\text{last}}$ has at most $\mathcal{O}(\sqrt{n})$ unique zones.



$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Now, create an object for each unique zone...



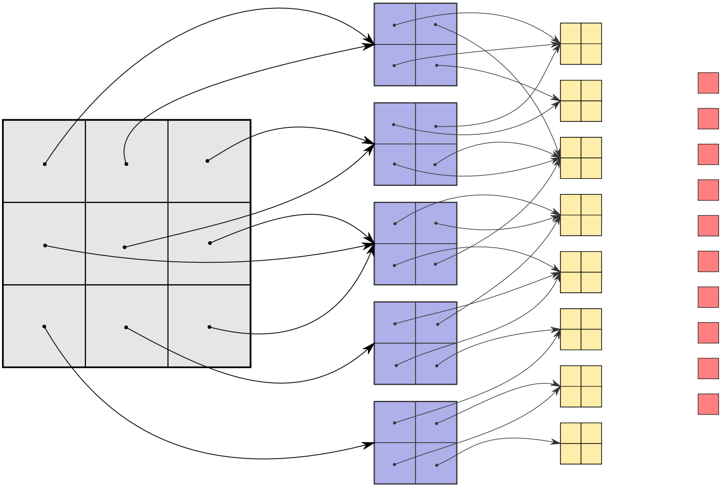
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Now, create an object for each unique zone...
Add pointers (each of size $\mathcal{O}(\log n)$ bits)...



$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

Now, create an object for each unique zone...

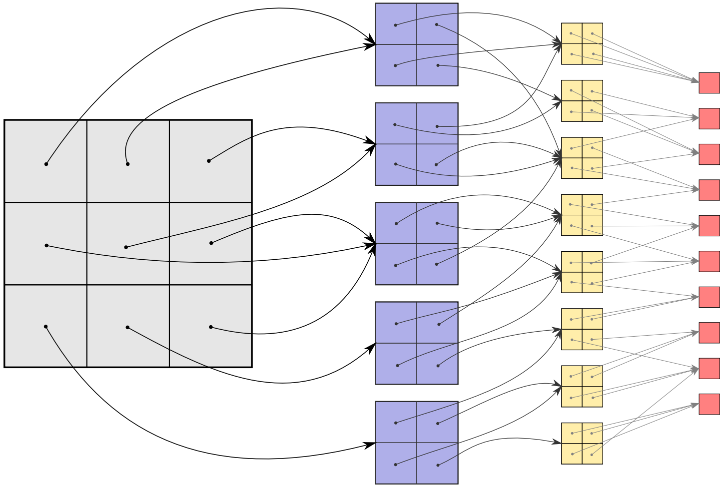
Add pointers (each of size $\mathcal{O}(\log n)$ bits)...



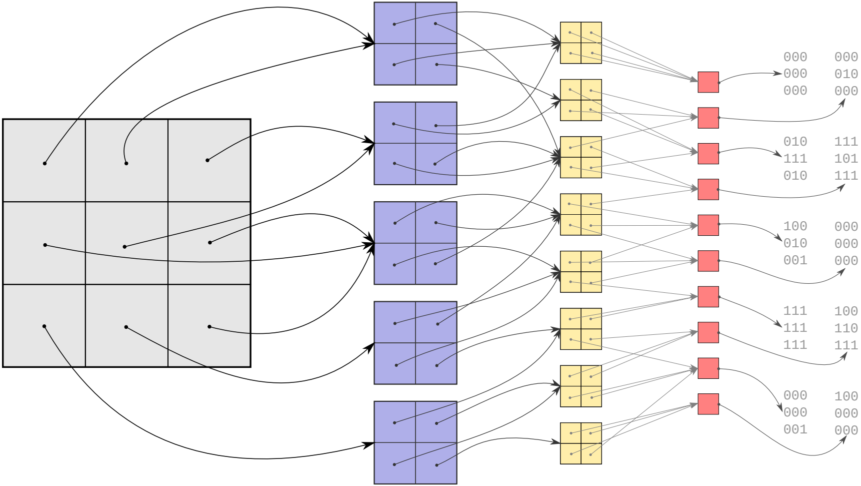
$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

Now, create an object for each unique zone...

Add pointers (each of size $\mathcal{O}(\log n)$ bits)...

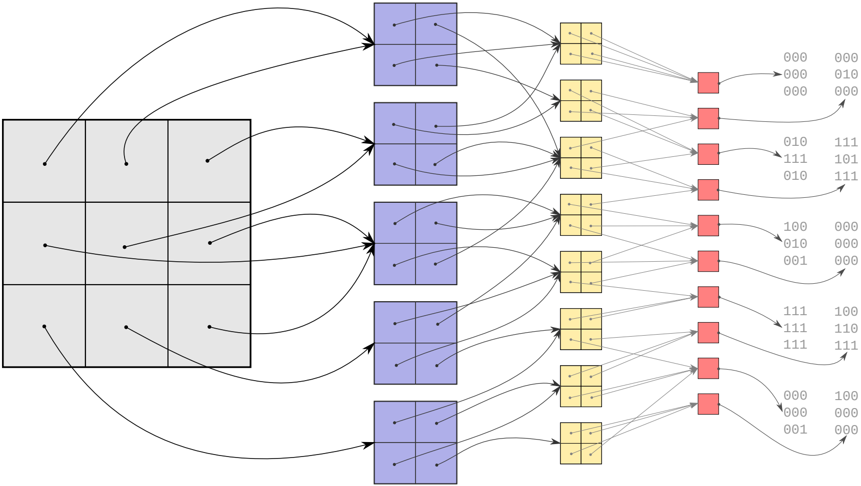


$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Now, create an object for each unique zone...
Add pointers (each of size $\mathcal{O}(\log n)$ bits)...
And store each unique zone of $\mathcal{D}_{\text{last}}$ explicitly.

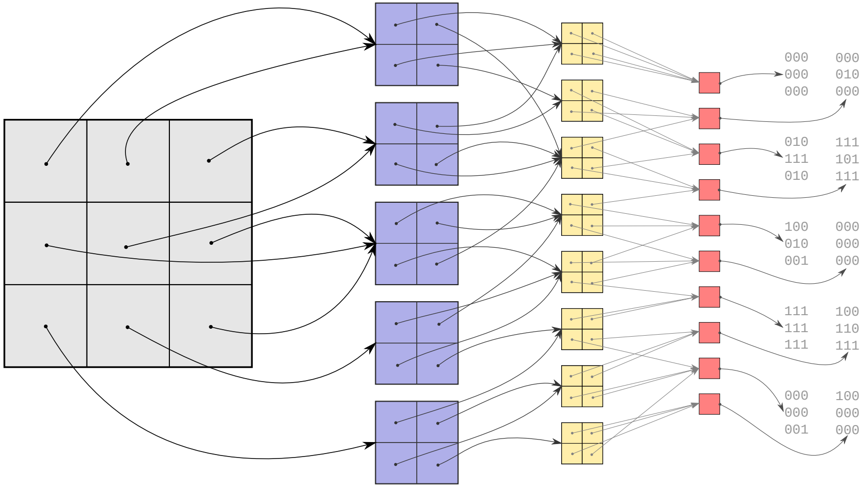


$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

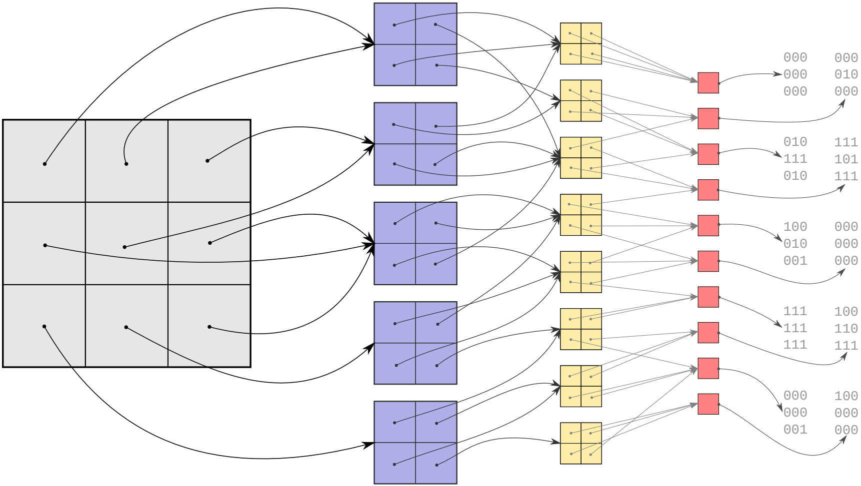
Query ($M[i, j] = ?$):



$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Query ($M[i,j] =?$): follow the pointers and return a bit from $\mathcal{D}_{\text{last}}$.



$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.
Query ($M[i,j] = ?$): follow the pointers and return a bit from $\mathcal{D}_{\text{last}}$.
Time complexity: $\mathcal{O}(\text{last}) = \mathcal{O}(\log \log n)$.



Memory complexity

$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

The proof has three parts:

Memory complexity

$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

The proof has three parts:

- $\mathcal{D}_{\text{last}}$ has only $\mathcal{O}(\sqrt{n})$ **small** zones and we can store them explicitly;

Memory complexity

$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

The proof has three parts:

- $\mathcal{D}_{\text{last}}$ has only $\mathcal{O}(\sqrt{n})$ **small** zones and we can store them explicitly;
- The layers with **large** zones occupy $\mathcal{O}_d(n)$ bits in total;

Memory complexity

$\mathcal{D}_k$ – a division of M where each zone is an $n^{(2/3)^k} \times n^{(2/3)^k}$ submatrix.

The proof has three parts:

- $\mathcal{D}_{\text{last}}$ has only $\mathcal{O}(\sqrt{n})$ **small** zones and we can store them explicitly;
- The layers with **large** zones occupy $\mathcal{O}_d(n)$ bits in total;
- The data structure needs to be modified slightly for **medium** zones.

Part 3

χ-boundedness

Pilipczuk, Sokołowski,
Graphs of Bounded Twin-Width are Quasi-Polynomially χ-Bounded

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\chi(G) \leq f(\omega(G)).$$

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

Examples:

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

Examples:

- perfect graphs ($f(x) = x$),

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

Examples:

- perfect graphs ($f(x) = x$),
- sparse graph classes (bounded degeneracy, bounded expansion),

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

Examples:

- perfect graphs ($f(x) = x$),
- sparse graph classes (bounded degeneracy, bounded expansion),
- bounded clique-width ($f(x) = \mathrm{poly}(x)$),

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

Examples:

- perfect graphs ($f(x) = x$),
- sparse graph classes (bounded degeneracy, bounded expansion),
- bounded clique-width ($f(x) = \mathrm{poly}(x)$),
- some geometric intersection graph classes...

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

Examples:

- perfect graphs ($f(x) = x$),
- sparse graph classes (bounded degeneracy, bounded expansion),
- bounded clique-width ($f(x) = \mathrm{poly}(x)$),
- some geometric intersection graph classes...

Esperet: *maybe if $\mathcal{C}$ is χ-bounded, then it is polynomially χ-bounded?*

χ-boundedness

Let $\mathcal{C}$ – a hereditary class of graphs.

Definition

$\mathcal{C}$ is χ-*bounded* by a function $f : \mathbb{N} \to \mathbb{N}$ if for every graph $G \in \mathcal{C}$,

$$\underbrace{\chi(G)}_{\text{coloring}} \leq f(\underbrace{\omega(G)}_{\text{clique}}).$$

Examples:

- perfect graphs ($f(x) = x$),
- sparse graph classes (bounded degeneracy, bounded expansion),
- bounded clique-width ($f(x) = \mathrm{poly}(x)$),
- some geometric intersection graph classes...

Esperet: *maybe if $\mathcal{C}$ is χ-bounded, then it is polynomially χ-bounded?*

Briański, Davies, Walczak (2022): **no.**

Twin-width and χ-boundedness

Theorem (Twin-width III)

Graphs of twin-width $\leq d$ are χ-bounded by $f_d(\omega) = (d+2)^{\omega-1}$.

Twin-width and χ-boundedness

Theorem (Twin-width III)

Graphs of twin-width $\leq d$ are χ-bounded by $f_d(\omega) = (d+2)^{\omega-1}$.

Note: the proof shows that

$$f_d(\omega) \leq (d+2) \cdot f_d(\omega - 1).$$

Twin-width and χ-boundedness

Theorem (Twin-width III)

Graphs of twin-width $\leq d$ *are* χ*-bounded by* $f_d(\omega) = (d+2)^{\omega-1}$.

Conjecture (Twin-width III)

Are graphs of twin-width $\leq d$ polynomially χ-bounded?

Twin-width and χ-boundedness

Theorem (Twin-width III)

Graphs of twin-width $\leq d$ are χ-bounded by $f_d(\omega) = (d+2)^{\omega-1}$.

Conjecture (Twin-width III)

Are graphs of twin-width $\leq d$ polynomially χ-bounded?

Our result (PS22)

Fix $d \in \mathbb{N}$. There exists a constant $\beta_d > 0$ such that graphs of twin-width $\leq d$ are χ-bounded by a **quasi-polynomial** function $f_d : \mathbb{N} \to \mathbb{N}$:

$$f_d(\omega) = 2^{\beta_d \cdot \log^{\mathcal{O}(d)} \omega}.$$

Twin-width and χ-boundedness

Theorem (Twin-width III)

Graphs of twin-width $\leq d$ are χ-bounded by $f_d(\omega) = (d+2)^{\omega-1}$.

Conjecture (Twin-width III)

Are graphs of twin-width $\leq d$ polynomially χ-bounded?

Our result (PS22)

Fix $d \in \mathbb{N}$. There exists a constant $\beta_d > 0$ such that graphs of twin-width $\leq d$ are χ-bounded by a **quasi-polynomial** function $f_d : \mathbb{N} \to \mathbb{N}$:

$$f_d(\omega) = 2^{\beta_d \cdot \log^{\mathcal{O}(d)} \omega}.$$

Theorem (Bourneuf, Thomassé 2023)

Graphs of twin-width $\leq d$ are polynomially χ-bounded.

Ingredient: d-almost mixed minors

$$
\begin{array}{cc|ccc|cccc|ccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
\hline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1
\end{array}
$$

Ingredient: d-almost mixed minors

$$
\begin{array}{cc|ccc|cccc|ccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
\hline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1
\end{array}
$$

Fact

M has a $2d$-almost mixed minor $\implies$ M has a d-mixed minor.

Ingredient: d-almost mixed minors

$$
\left.\begin{array}{cc|ccc|cccc|ccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
\hline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1
\end{array}\right.
$$

Fact

M has a $2d$-almost mixed minor $\implies$ M has a d-mixed minor.

Corollary

$$\mathrm{tww}(G) \le d \implies G \text{ has a } (2d+2)\text{-mixed free adjacency matrix}$$
$$\implies G \text{ has a } (4d+4)\text{-almost mixed free adjacency matrix.}$$

Idea

If we had

$$f_d(\omega) \leq \mathsf{const}(d) \cdot f_d(0.9\,\omega),$$

then we would get $f_d(\omega) = \mathrm{poly}(\omega)$!

Idea

If we had

$$f_d(\omega) \leq \mathrm{const}(d) \cdot f_d(0.9\,\omega),$$

then we would get $f_d(\omega) = \mathrm{poly}(\omega)$!

Induction on $\omega(G) \geq 1$.

G – a graph,
M – a d-almost mixed free adjacency matrix of G,
$V(G) = \{1, \ldots, n\}$ ordered according to M.

Idea

If we had

$$f_d(\omega) \le \mathrm{const}(d) \cdot f_d(0.9\,\omega),$$

then we would get $f_d(\omega) = \mathrm{poly}(\omega)$!

Induction on $\omega(G) \ge 1$.

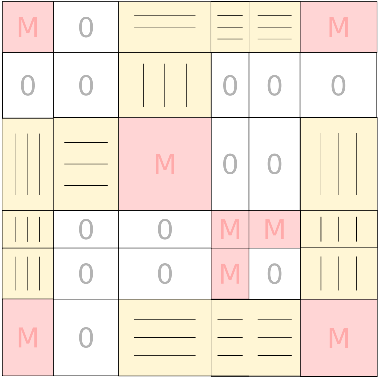
G – a graph,
M – a d-almost mixed free adjacency matrix of G,
$V(G) = \{1, \ldots, n\}$ ordered according to M.

Partition $V(G)$ into intervals $A_1 \cup A_2 \cup \cdots \cup A_k$ (**blobs**) so that

$$\omega(G[A_i]) = 0.9\,\omega \qquad \text{for } i = 1, 2, \ldots, k.$$

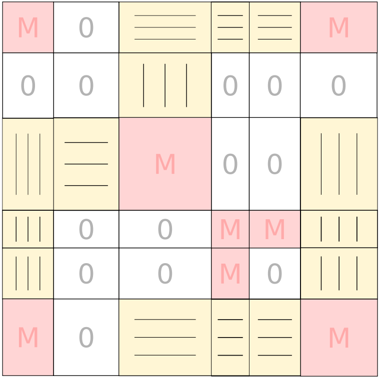
$$V(G) = A_1 \cup \cdots \cup A_k, \qquad \omega(G[A_i]) = 0.9\,\omega \quad \text{for } i = 1, 2, \ldots, k.$$

$\mathcal{D} :=$ a (symmetric) division of M from the partition $A_1, \ldots, A_k$.



$$V(G) = A_1 \cup \cdots \cup A_k, \qquad \omega(G[A_i]) = 0.9\,\omega \quad \text{for } i = 1, 2, \ldots, k.$$

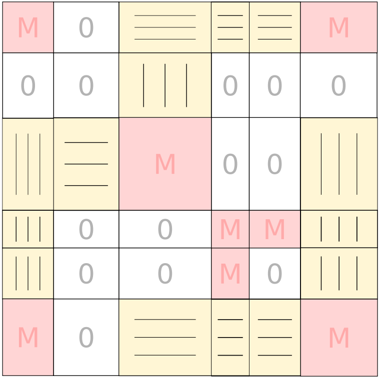
$\mathcal{D} :=$ a (symmetric) division of M from the partition $A_1, \ldots, A_k$.



First, paint each **blob**

$$V(G) = A_1 \cup \cdots \cup A_k, \qquad \omega(G[A_i]) = 0.9\,\omega \quad \text{for } i = 1, 2, \ldots, k.$$

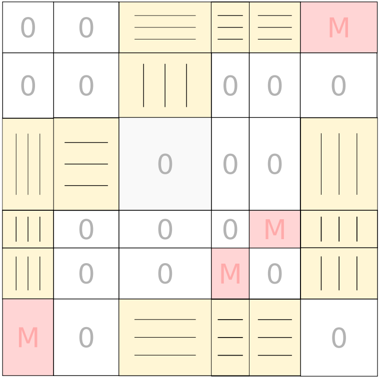
$\mathcal{D} :=$ a (symmetric) division of M from the partition $A_1, \ldots, A_k$.



First, paint each **blob** using $f_d(0.9\omega)$ colors.

$$V(G) = A_1 \cup \dots \cup A_k, \qquad \omega(G[A_i]) = 0.9\,\omega \quad \text{for } i = 1, 2, \dots, k.$$

$\mathcal{D} :=$ a (symmetric) division of M from the partition $A_1, \dots, A_k$.



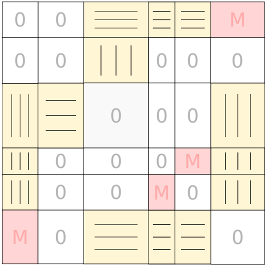
First, paint each **blob** using $f_d(0.9\omega)$ colors.

For each color class C: each intersection $C \cap A_i$ is an **independent set**!

Blob-blob connections

$$V(G) = A_1 \cup \cdots \cup A_k, \qquad \omega(G[A_i]) = 0.9\,\omega \quad \text{for } i = 1, 2, \ldots, k.$$

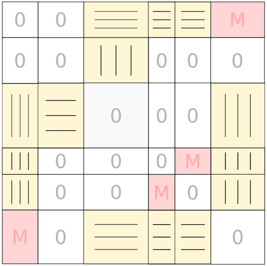
$\mathcal{D} :=$ a (symmetric) division of M from the partition $A_1, \ldots, A_k$.
Given a set (color class) C s.t. each $A_i \cap C$ is an **independent set**.



Blob-blob connections

$$V(G) = A_1 \cup \cdots \cup A_k, \qquad \omega(G[A_i]) = 0.9\,\omega \quad \text{for } i = 1, 2, \ldots, k.$$

$\mathcal{D} :=$ a (symmetric) division of M from the partition $A_1, \ldots, A_k$.
Given a set (color class) C s.t. each $A_i \cap C$ is an **independent set**.



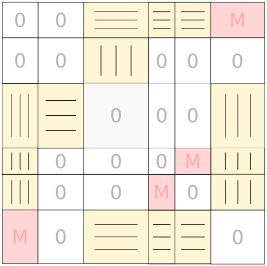
Lemma

$$\chi(G[C]) \leq \mathrm{const}(d) \cdot f_d(0.2\omega).$$

Blob-blob connections

$$V(G) = A_1 \cup \cdots \cup A_k, \qquad \omega(G[A_i]) = 0.9\,\omega \quad \text{for } i = 1, 2, \ldots, k.$$

$\mathcal{D}$:= a (symmetric) division of M from the partition $A_1, \ldots, A_k$.
Given a set (color class) C s.t. each $A_i \cap C$ is an **independent set**.



Lemma

$$\chi(G[C]) \le \underbrace{\mathrm{const}(d)}_{\text{remove } \boxed{\text{M}} \text{ with Marcus–Tardos}} \cdot \underbrace{f_d(0.2\omega)}_{\text{afterwards, clique number } \le 0.2\omega}$$

Solution attempt (summary)

- We assign each vertex a product coloring of 2 colorings:

Solution attempt (summary)

- We assign each vertex a product coloring of 2 colorings:
  - (**within blobs**) $f_d(0.9\omega)$ colors,

Solution attempt (summary)

- We assign each vertex a product coloring of 2 colorings:
  - (**within blobs**) $f_d(0.9\omega)$ colors,
  - (**between blobs**) $\mathrm{const}(d) \cdot f_d(0.2\omega)$ colors.

Solution attempt (summary)

- We assign each vertex a product coloring of 2 colorings:
  - (**within blobs**) $f_d(0.9\omega)$ colors,
  - (**between blobs**) $\mathrm{const}(d) \cdot f_d(0.2\omega)$ colors.
- Thus:

$$f_d(\omega) \leq \mathrm{const}(d) \cdot f_d(0.2\,\omega) \cdot f_d(0.9\,\omega).$$

Solution attempt (summary)

- We assign each vertex a product coloring of 2 colorings:
  - (**within blobs**) $f_d(0.9\omega)$ colors,
  - (**between blobs**) $\mathrm{const}(d) \cdot f_d(0.2\omega)$ colors.
- Thus:

$$f_d(\omega) \leq \mathrm{const}(d) \cdot f_d(0.2\,\omega) \cdot f_d(0.9\,\omega).$$

- Even worse than exponential...

Solution: attempt 2

Maybe not all hope is lost. What additional assumptions on G and M would help us?

Solution: attempt 2

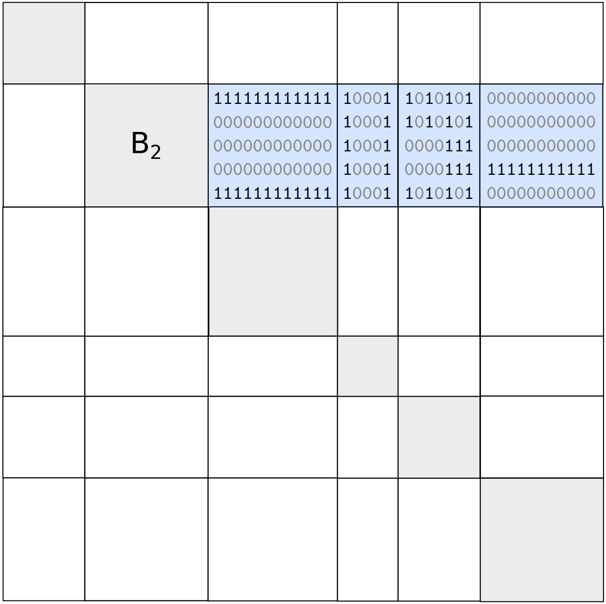
Maybe not all hope is lost. What additional assumptions on G and M would help us?

Rich blobs

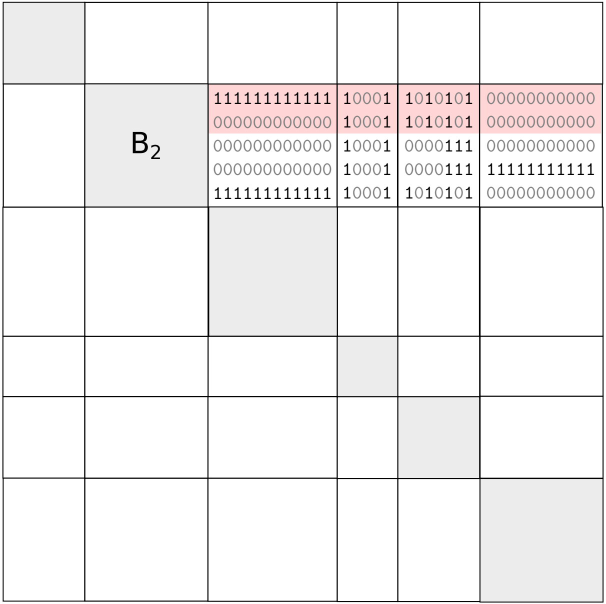
Fix a blob B_i ($i < k$), and let $B_i = \{\ell_i, \ell_i + 1, \dots, r_i\}$. We call B_i **rich** if:

- every vertex $v \in B_i$ is adjacent to any $s \in \{r_i + 1, r_i + 2, \dots, n\}$,
- no two consecutive vertices of B_i are twins with respect to $\{r_i + 1, r_i + 2, \dots, n\}$.

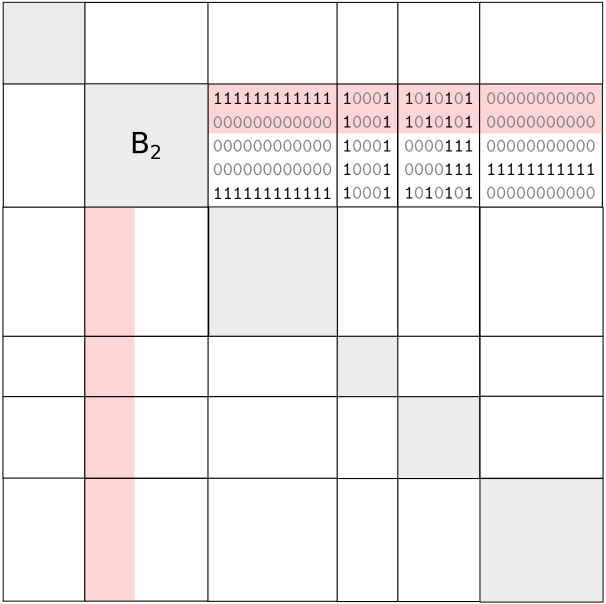
Solution: attempt 2 (rich blobs)



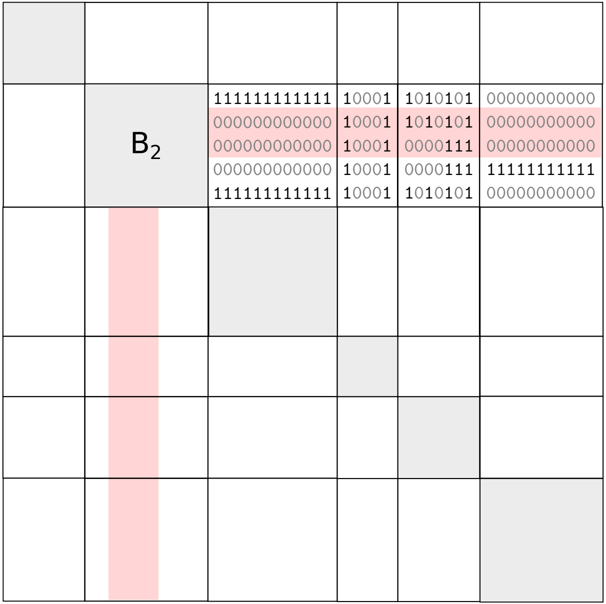
Solution: attempt 2 (rich blobs)



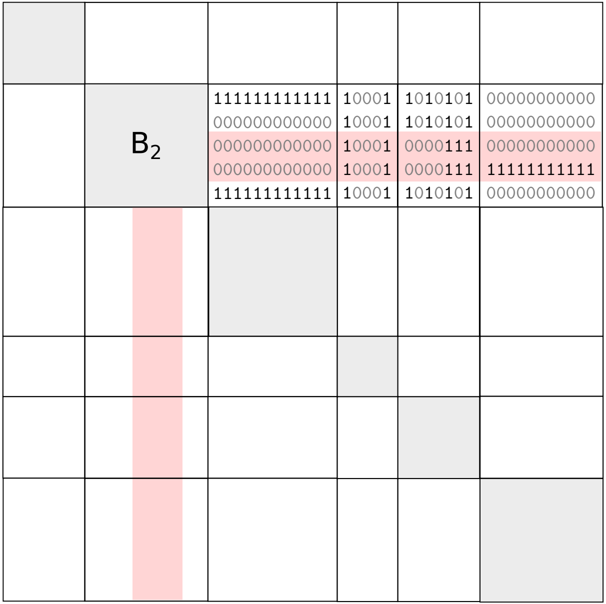
Solution: attempt 2 (rich blobs)



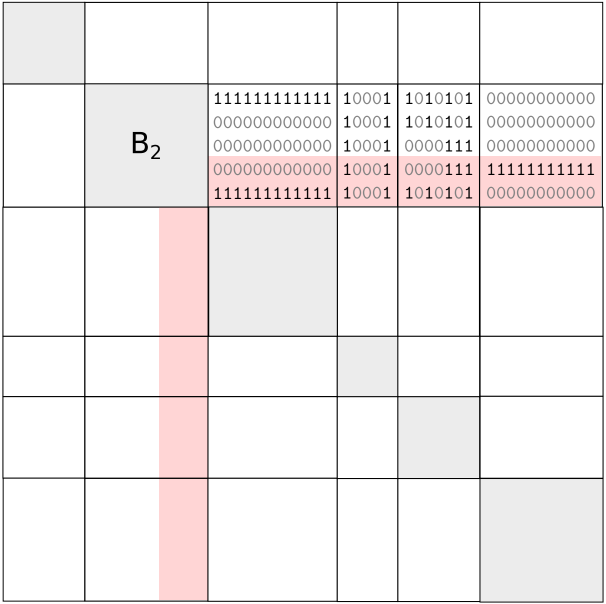
Solution: attempt 2 (rich blobs)



Solution: attempt 2 (rich blobs)



Solution: attempt 2 (rich blobs)



Solution: attempt 2 (rich blobs)

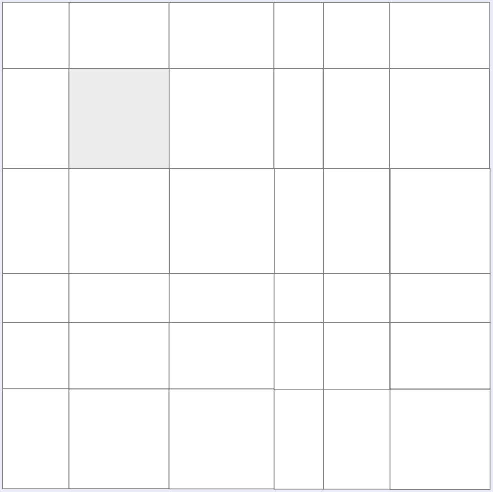
Rich blob lemma

If B_i is rich, then the adjacency matrix of B_i is $(d-1)$-almost mixed free.

Solution: attempt 2 (rich blobs)

Rich blob lemma

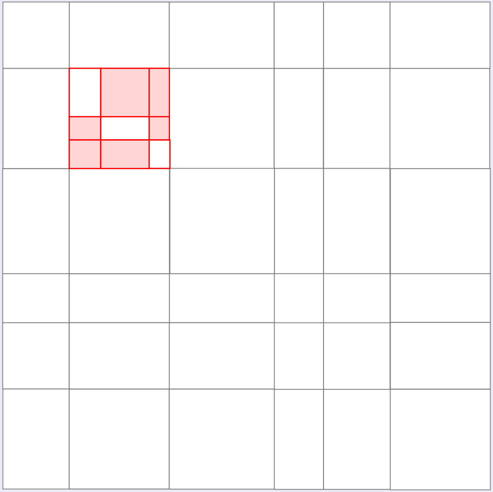
If B_i is rich, then the adjacency matrix of B_i is $(d-1)$-almost mixed free.



Solution: attempt 2 (rich blobs)

Rich blob lemma

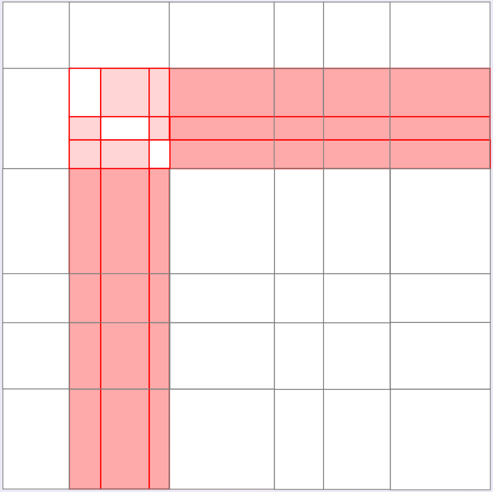
If B_i is rich, then the adjacency matrix of B_i is $(d-1)$-almost mixed free.



Solution: attempt 2 (rich blobs)

Rich blob lemma

If B_i is rich, then the adjacency matrix of B_i is $(d-1)$-almost mixed free.



Solution: attempt 2 (rich blobs)

If all blobs are rich:

As before, assign each vertex a product coloring of 2 colorings:

Solution: attempt 2 (rich blobs)

If all blobs are rich:

As before, assign each vertex a product coloring of 2 colorings:

- (**within blobs**) $f_{\mathbf{d\text{-}1}}(0.9\omega)$ colors (instead of $f_d(0.9\omega)$),

Solution: attempt 2 (rich blobs)

If all blobs are rich:

As before, assign each vertex a product coloring of 2 colorings:

- (**within blobs**) $f_{\mathbf{d\text{-}1}}(0.9\omega)$ colors (instead of $f_d(0.9\omega)$),
- (**between blobs**) $\mathrm{const}(d) \cdot f_d(0.2\omega)$ colors.

Solution: attempt 2 (rich blobs)

If all blobs are rich:

As before, assign each vertex a product coloring of 2 colorings:

- (**within blobs**) $f_{\mathbf{d\text{-}1}}(0.9\omega)$ colors (instead of $f_d(0.9\omega)$),
- (**between blobs**) $\mathrm{const}(d) \cdot f_d(0.2\omega)$ colors.

Thus:

$$f_d(\omega) \leq \mathrm{const}(d) \cdot f_d(0.2\,\omega) \cdot f_{d-1}(0.9\,\omega).$$

Solution: attempt 2 (rich blobs)

If all blobs are rich:

As before, assign each vertex a product coloring of 2 colorings:

- (**within blobs**) $f_{\mathbf{d\text{-}1}}(0.9\omega)$ colors (instead of $f_d(0.9\omega)$),
- (**between blobs**) $\mathrm{const}(d) \cdot f_d(0.2\omega)$ colors.

Thus:

$$f_d(\omega) \leq \mathrm{const}(d) \cdot f_d(0.2\,\omega) \cdot f_{d-1}(0.9\,\omega).$$

By induction: quasi-polynomial!

Solution: attempt 2

Another extreme: **poor** blobs

Fix a blob B_i ($i < k$). We call B_i **poor** if all its adjacencies with other blobs are empty or mixed.

Solution: attempt 2

Another extreme: poor blobs

Fix a blob B_i ($i < k$). We call B_i **poor** if all its adjacencies with other blobs are empty or mixed.

If all blobs are rich or poor:

- Purge mixed connections by painting each blob into const(d) colors (Marcus–Tardos);

Solution: attempt 2

Another extreme: **poor** blobs

Fix a blob B_i ($i < k$). We call B_i **poor** if all its adjacencies with other blobs are empty or mixed.

If all blobs are rich or poor:

- Purge mixed connections by painting each blob into const(d) colors (Marcus–Tardos);
- Paint vertices of **poor** blobs with a shared palette of $f_d(0.9\,\omega)$ colors;

Solution: attempt 2

Another extreme: **poor** blobs

Fix a blob B_i ($i < k$). We call B_i **poor** if all its adjacencies with other blobs are empty or mixed.

If all blobs are rich or poor:

- Purge mixed connections by painting each blob into const(d) colors (Marcus–Tardos);
- Paint vertices of **poor** blobs with a shared palette of $f_d(0.9\,\omega)$ colors;
- **Rich** blobs: $f_{d-1}(0.9\,\omega) \cdot f_d(0.2\,\omega)$ colors (as before);

Solution: attempt 2

Another extreme: poor blobs

Fix a blob B_i ($i < k$). We call B_i **poor** if all its adjacencies with other blobs are empty or mixed.

If all blobs are rich or poor:

- Purge mixed connections by painting each blob into const(d) colors (Marcus–Tardos);
- Paint vertices of **poor** blobs with a shared palette of $f_d(0.9\,\omega)$ colors;
- **Rich** blobs: $f_{d-1}(0.9\,\omega) \cdot f_d(0.2\,\omega)$ colors (as before);
- Thus:

$$f_d(\omega) \leq \mathrm{const}(d) \cdot \{f_d(0.9\,\omega) \ + \ f_d(0.2\,\omega) \cdot f_{d-1}(0.9\,\omega)\}\,.$$

Solution: attempt 2

Another extreme: **poor** blobs

Fix a blob B_i ($i < k$). We call B_i **poor** if all its adjacencies with other blobs are empty or mixed.

If all blobs are rich or poor:

- Purge mixed connections by painting each blob into $\mathrm{const}(d)$ colors (Marcus–Tardos);
- Paint vertices of **poor** blobs with a shared palette of $f_d(0.9\,\omega)$ colors;
- **Rich** blobs: $f_{d-1}(0.9\,\omega) \cdot f_d(0.2\,\omega)$ colors (as before);
- Thus:

$$f_d(\omega) \leq \mathrm{const}(d) \cdot \{f_d(0.9\,\omega) \; + \; f_d(0.2\,\omega) \cdot f_{d-1}(0.9\,\omega)\}\,.$$

- Quasi-polynomial again!

Solution: endgame

Not all blobs must be **rich** or **poor**.

Solution: endgame

Not all blobs must be **rich** or **poor**.

But each blob can be **split** into a poor part and an (almost) rich part!

Solution: endgame

Not all blobs must be **rich** or **poor**.

But each blob can be **split** into a poor part and an (almost) rich part!

The argument with (almost) **rich** blobs is involved and produces a worse bound:

$$\chi(B_i) \leq f_d(0.1\,\omega) \cdot f_{d-1}(\omega^d)^2 \qquad \text{instead of} \qquad \chi(B_i) \leq f_{d-1}(0.9\,\omega).$$

Solution: endgame

Not all blobs must be **rich** or **poor**.

But each blob can be **split** into a poor part and an (almost) rich part!

The argument with (almost) **rich** blobs is involved and produces a worse bound:

$$\chi(B_i) \leq f_d(0.1\,\omega) \cdot f_{d-1}(\omega^d)^2 \qquad \text{instead of} \qquad \chi(B_i) \leq f_{d-1}(0.9\,\omega).$$

Then:

$$f_d(\omega) \leq \mathrm{const}(d) \cdot \{f_d(0.9\,\omega) + \max \chi(B_i) \cdot f_d(0.2\,\omega)\}$$

Solution: endgame

Not all blobs must be **rich** or **poor**.

But each blob can be **split** into a poor part and an (almost) rich part!

The argument with (almost) **rich** blobs is involved and produces a worse bound:

$$\chi(B_i) \leq f_d(0.1\,\omega) \cdot f_{d-1}(\omega^d)^2 \qquad \text{instead of} \qquad \chi(B_i) \leq f_{d-1}(0.9\,\omega).$$

Then:

$$\begin{aligned} f_d(\omega) &\leq \text{const}(d) \cdot \{f_d(0.9\,\omega) + \max \chi(B_i) \cdot f_d(0.2\,\omega)\} \\ &\leq \text{const}(d) \cdot \left\{ f_d(0.9\,\omega) + f_d(0.1\,\omega) \cdot f_d(0.2\,\omega) \cdot f_{d-1}(\omega^d)^2 \right\} \end{aligned}$$

Subexponential ($2^{O(\omega^\varepsilon)}$ for any $\varepsilon > 0$ if parameters chosen carefully).

Solution: endgame

$$f_d(\omega) \leq \text{const}(d) \cdot \left\{ f_d(0.9\,\omega) + f_d(0.1\,\omega) \cdot f_d(0.2\,\omega) \cdot f_{d-1}(\omega^d)^2 \right\}$$

Solution: endgame

$$f_d(\omega) \leq \mathrm{const}(d) \cdot \left\{ f_d(0.9\,\omega) + f_d(0.1\,\omega) \cdot f_d(0.2\,\omega) \cdot f_{d-1}(\omega^d)^2 \right\}$$

How to reach a quasi-polynomial bound on χ?

Inspired by a work of Chudnovsky, Penev, Scott, Trotignon (*Substitution and χ-boundedness*, JCTB, 2013).

Solution: endgame

$$f_d(\omega) \leq \mathrm{const}(d) \cdot \left\{ f_d(0.9\,\omega) + f_d(0.1\,\omega) \cdot f_d(0.2\,\omega) \cdot f_{d-1}(\omega^d)^2 \right\}$$

How to reach a quasi-polynomial bound on χ?

Inspired by a work of Chudnovsky, Penev, Scott, Trotignon (*Substitution and χ-boundedness*, JCTB, 2013).

Intuition: the blobs are **more complicated** $\Longrightarrow$ the connections between the blobs are **less complex** $\Longrightarrow$ tradeoff between $f_d(0.1\,\omega)$ and $f_d(0.2\,\omega)$.

Solution: endgame

$$f_d(\omega) \leq \text{const}(d) \cdot \left\{ f_d(0.9\,\omega) + f_d(0.1\,\omega) \cdot f_d(0.2\,\omega) \cdot f_{d-1}(\omega^d)^2 \right\}$$

How to reach a quasi-polynomial bound on χ?

Inspired by a work of Chudnovsky, Penev, Scott, Trotignon (*Substitution and χ-boundedness*, JCTB, 2013).

Intuition: the blobs are **more complicated** $\implies$ the connections between the blobs are **less complex** $\implies$ tradeoff between $f_d(0.1\,\omega)$ and $f_d(0.2\,\omega)$.

We eventually get:

$$f_d(\omega) \leq \text{const}(d) \cdot \left\{ f_d(0.9\,\omega) + f_{d-1}(\omega^d)^2 \cdot \sum_{u=0}^{\lfloor \log_2(0.1\,\omega) \rfloor} f_d(2^{u+1}) \cdot f_d\left(\frac{0.2\,\omega}{2^u} + 1\right) \right\}$$

This resolves to $f_d(\omega) = 2^{\beta_d \cdot \log^d \omega}$.

Thank you!