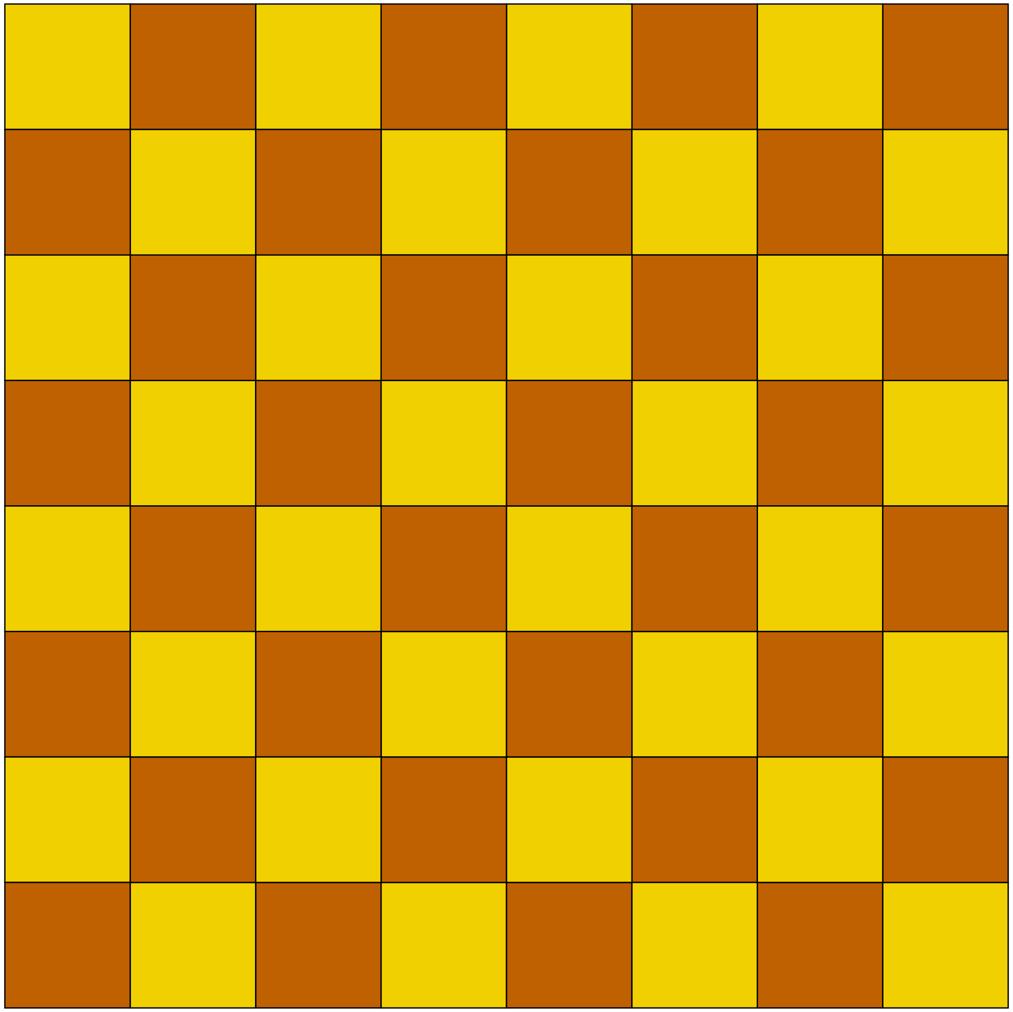
Should We Believe in Numbers Computed by Loopy Belief Propagation?

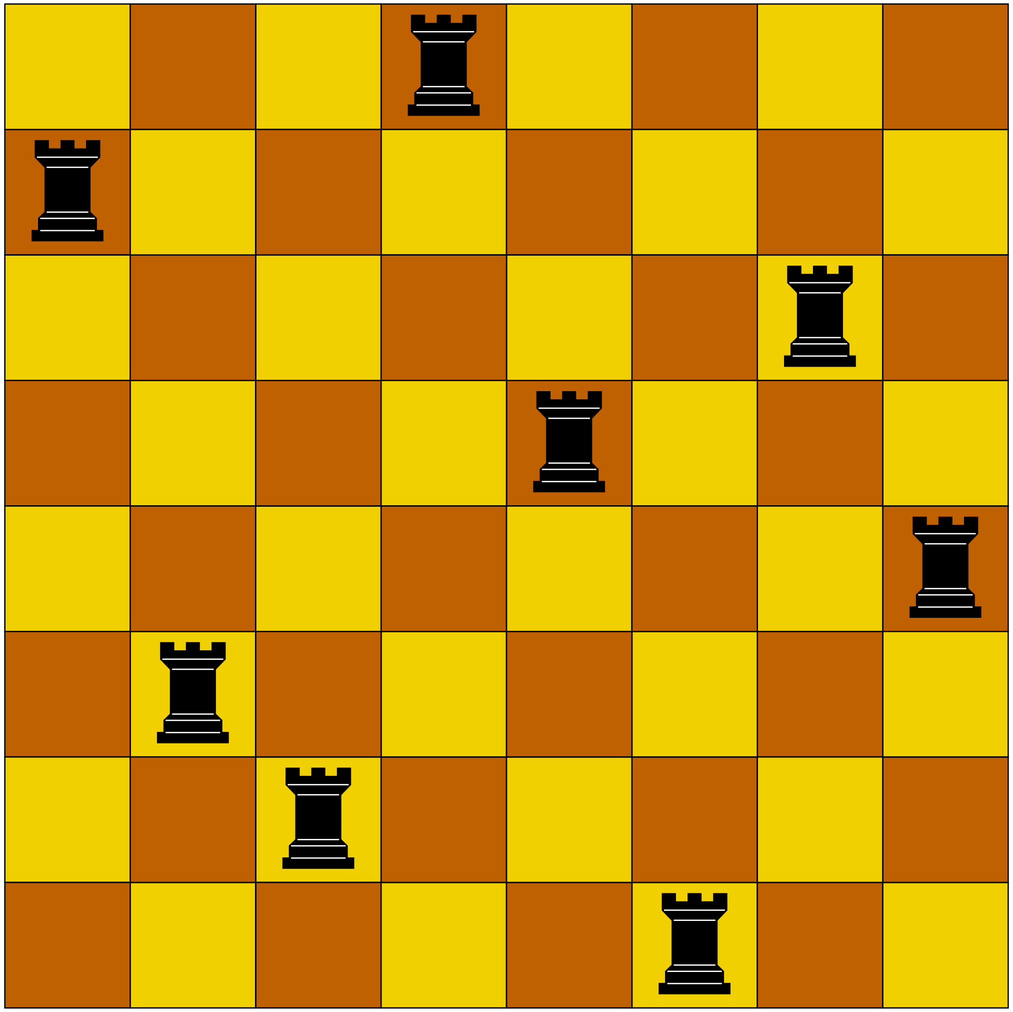
Pascal O. Vontobel
Information Theory Research Group
Hewlett-Packard Laboratories Palo Alto

CIOG Workshop, Princeton University, November 3, 2011

Chess Board



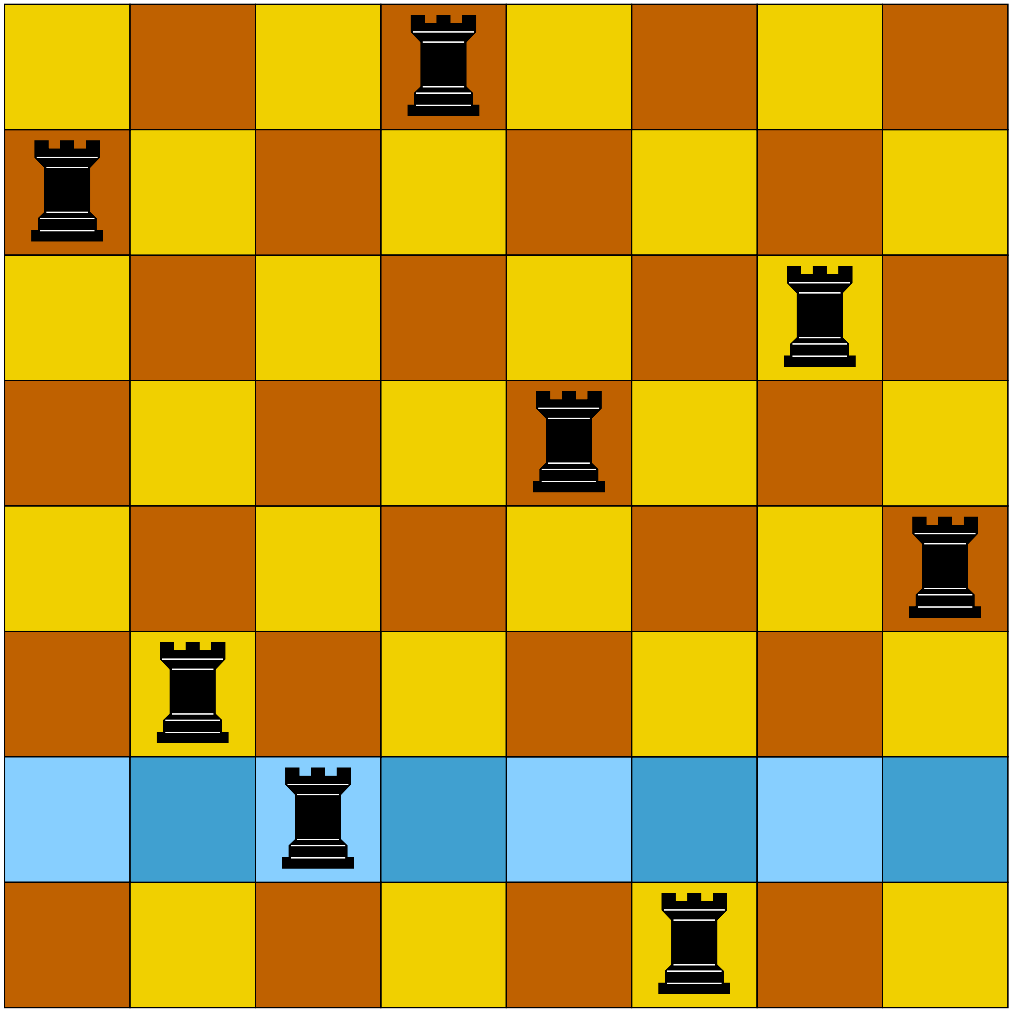
Chess Board



Question: in how many ways can we place 8 non-attacking rooks on a chess board?

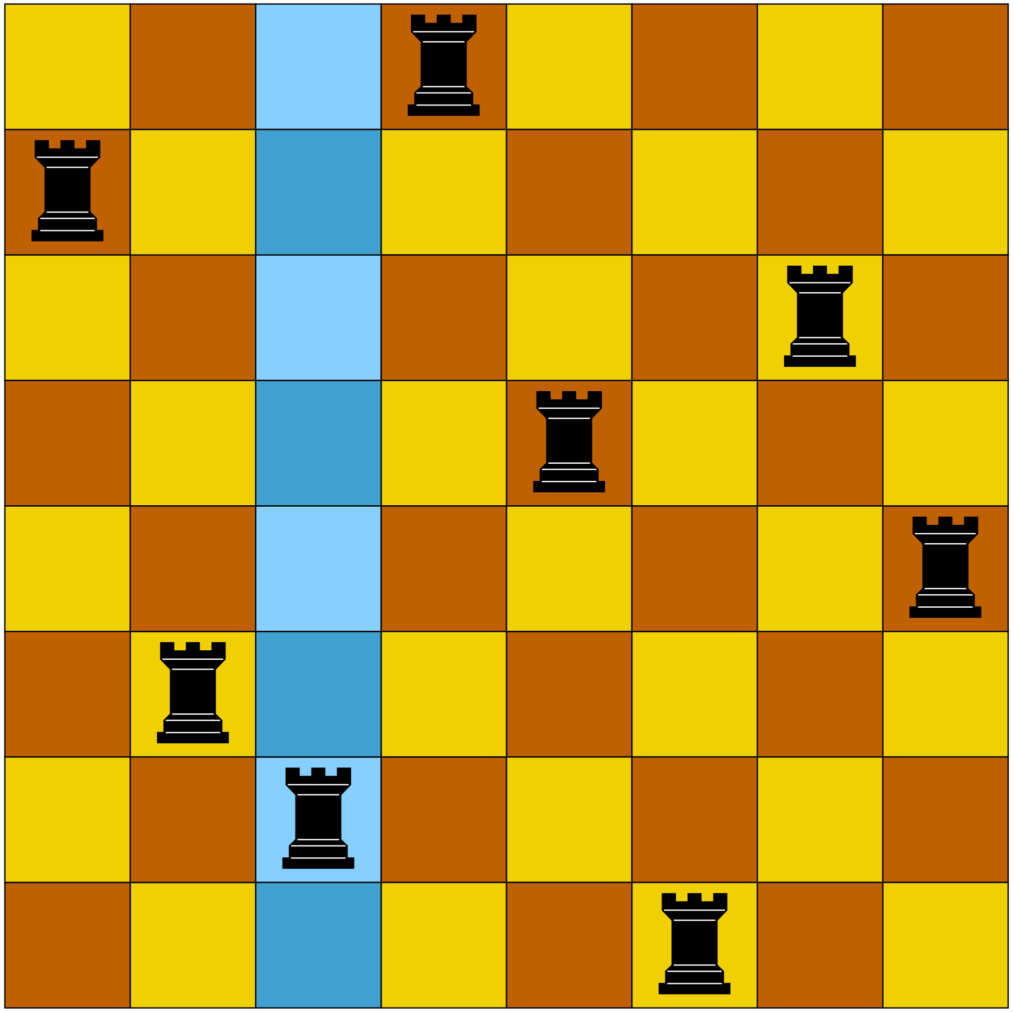
LABS hp

Chess Board



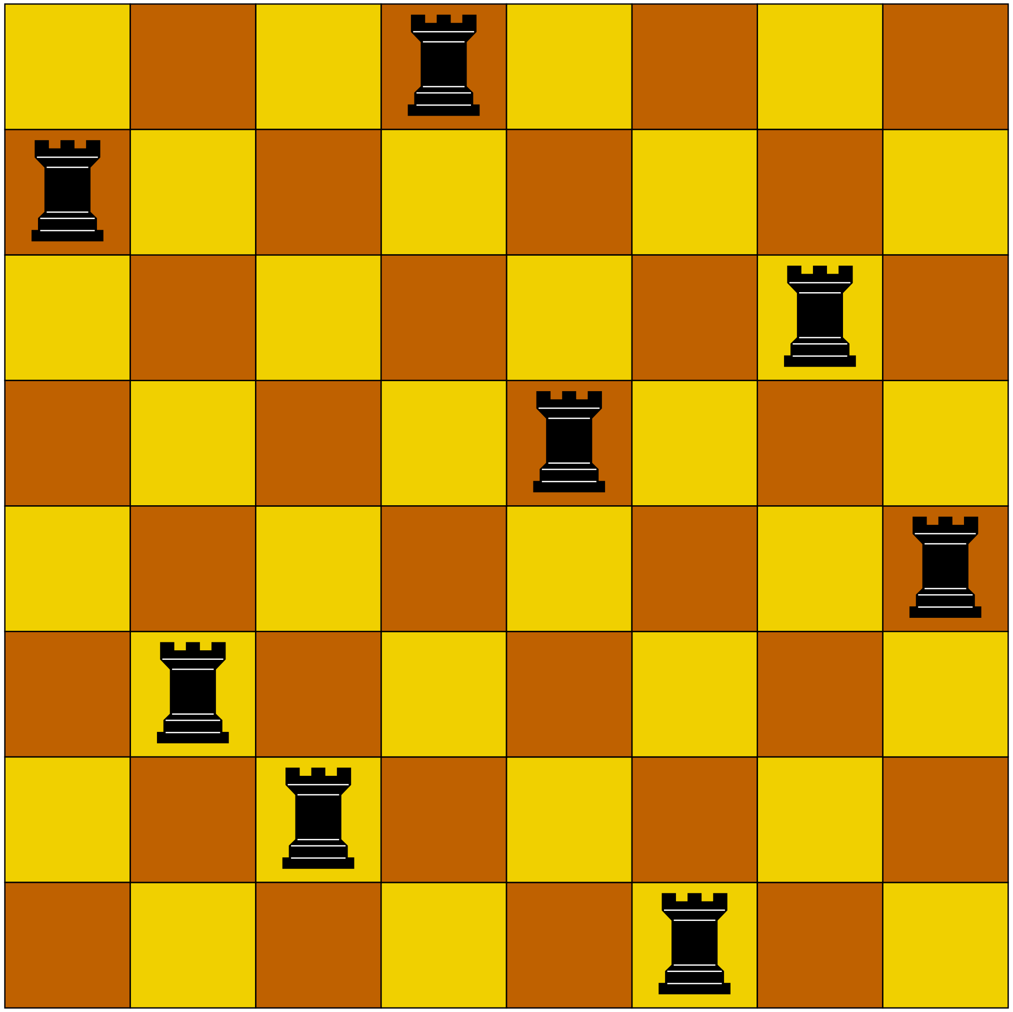
Row condition: exactly one rook per row.

Chess Board



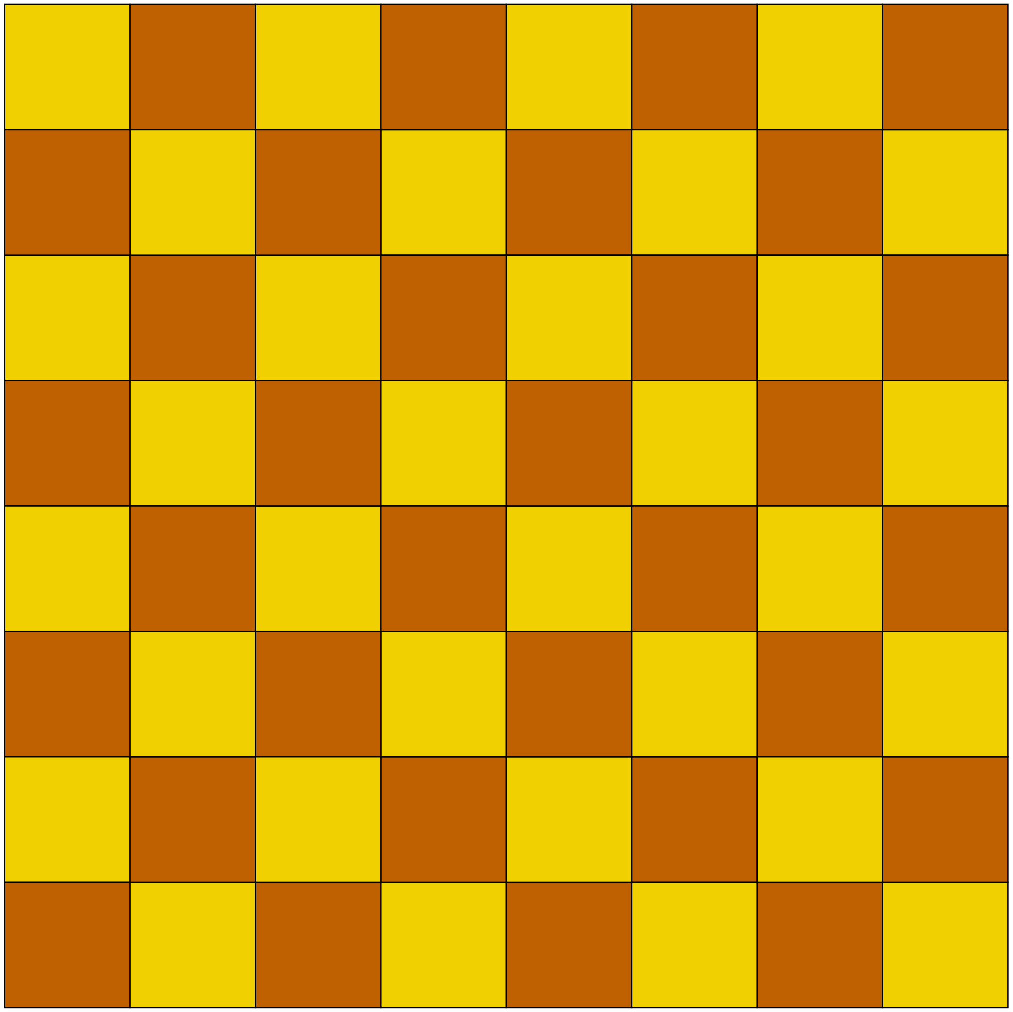
Column condition: exactly one rook per column.

Chess Board

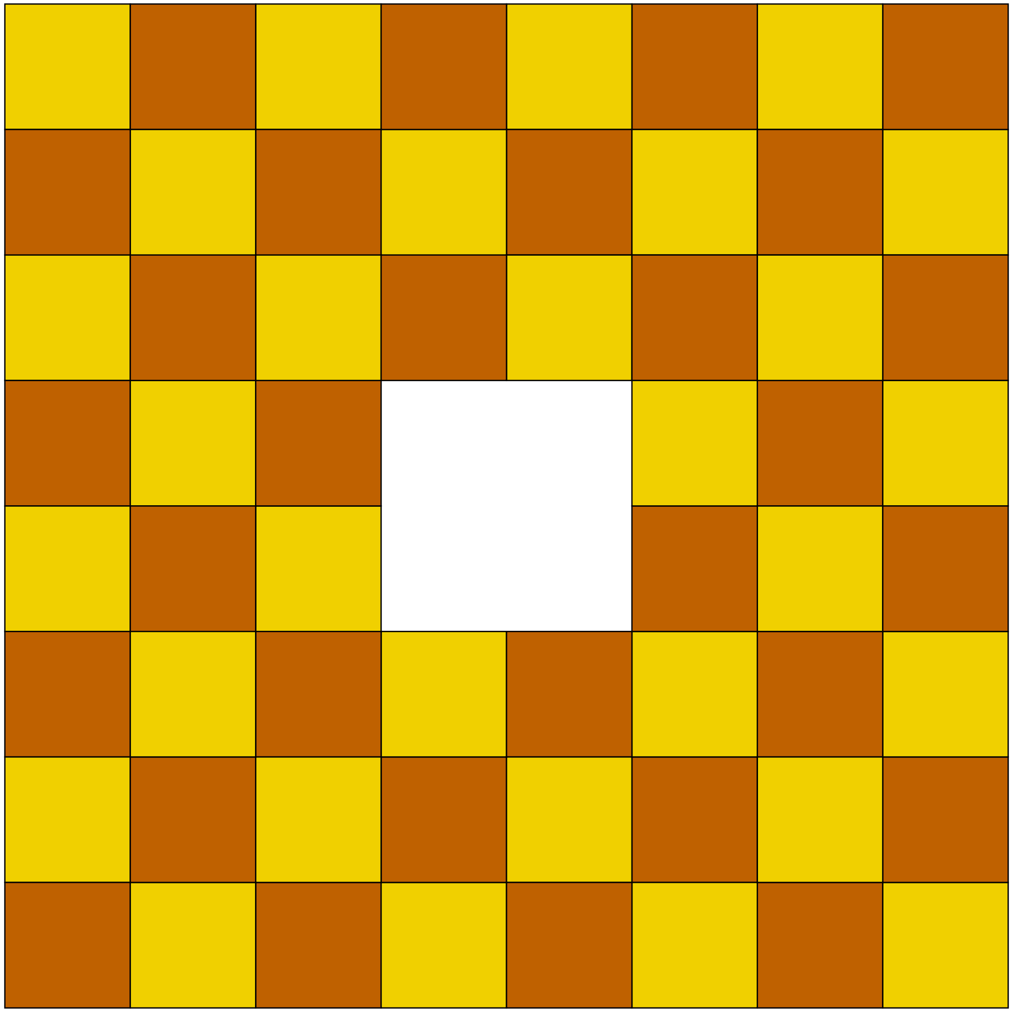


Question: in how many ways can we place 8 non-attacking rooks on a chess board?

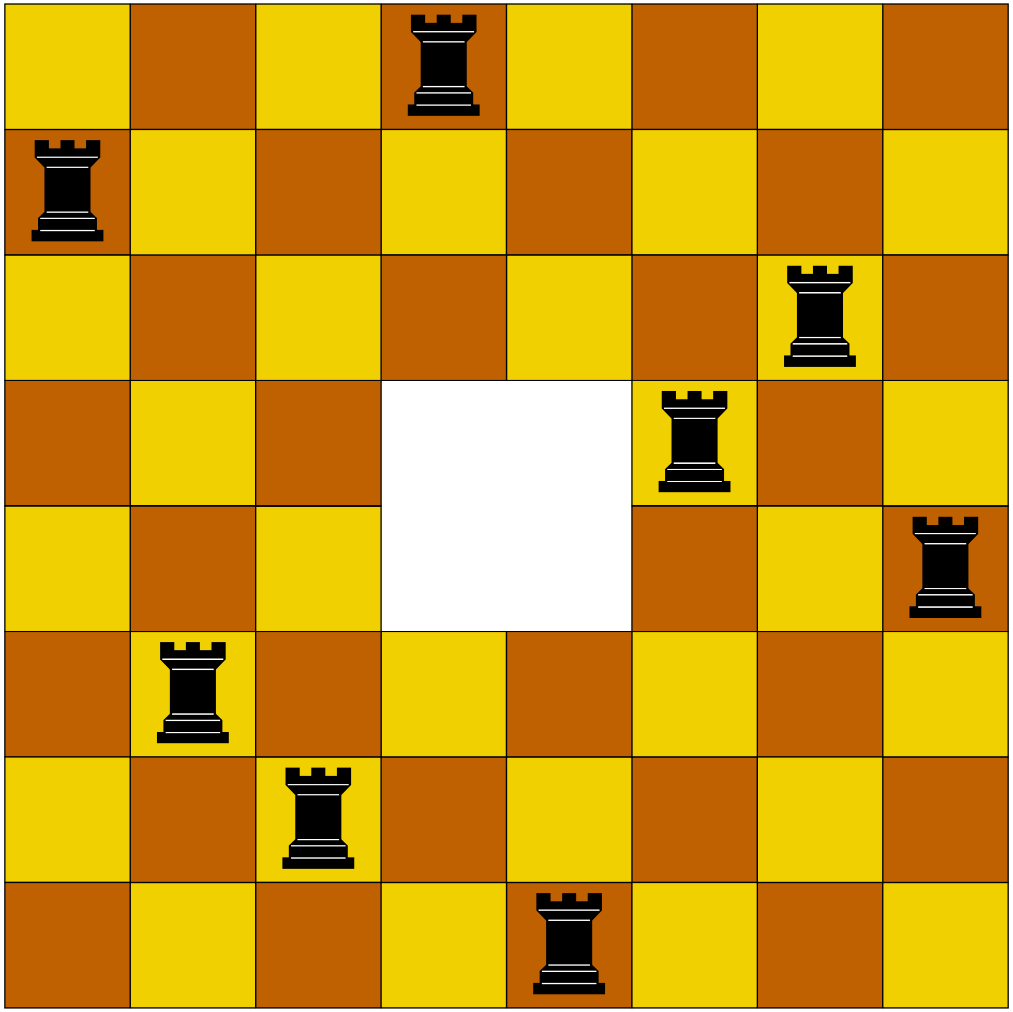
Chess Board



Chess Board

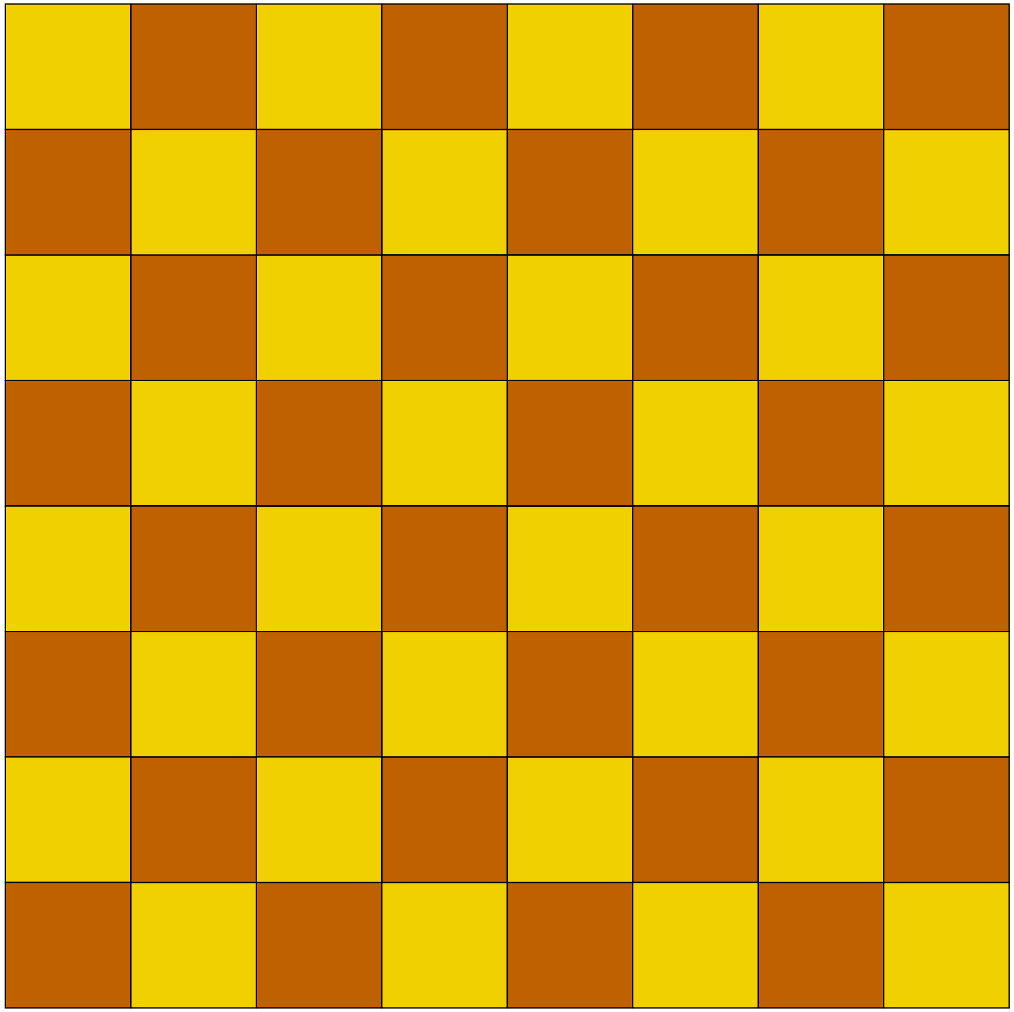


Chess Board

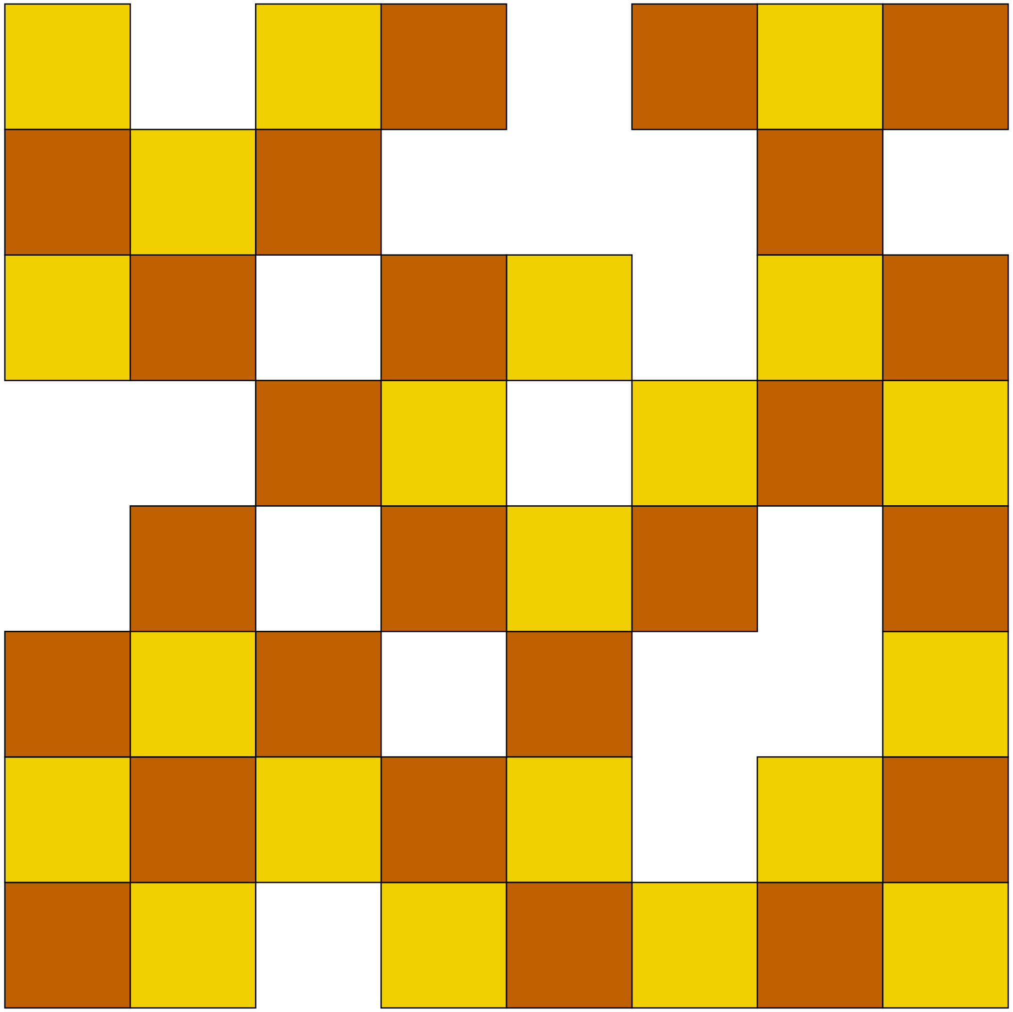


Question: in how many ways can we place 8 non-attacking rooks on this modified chess board?

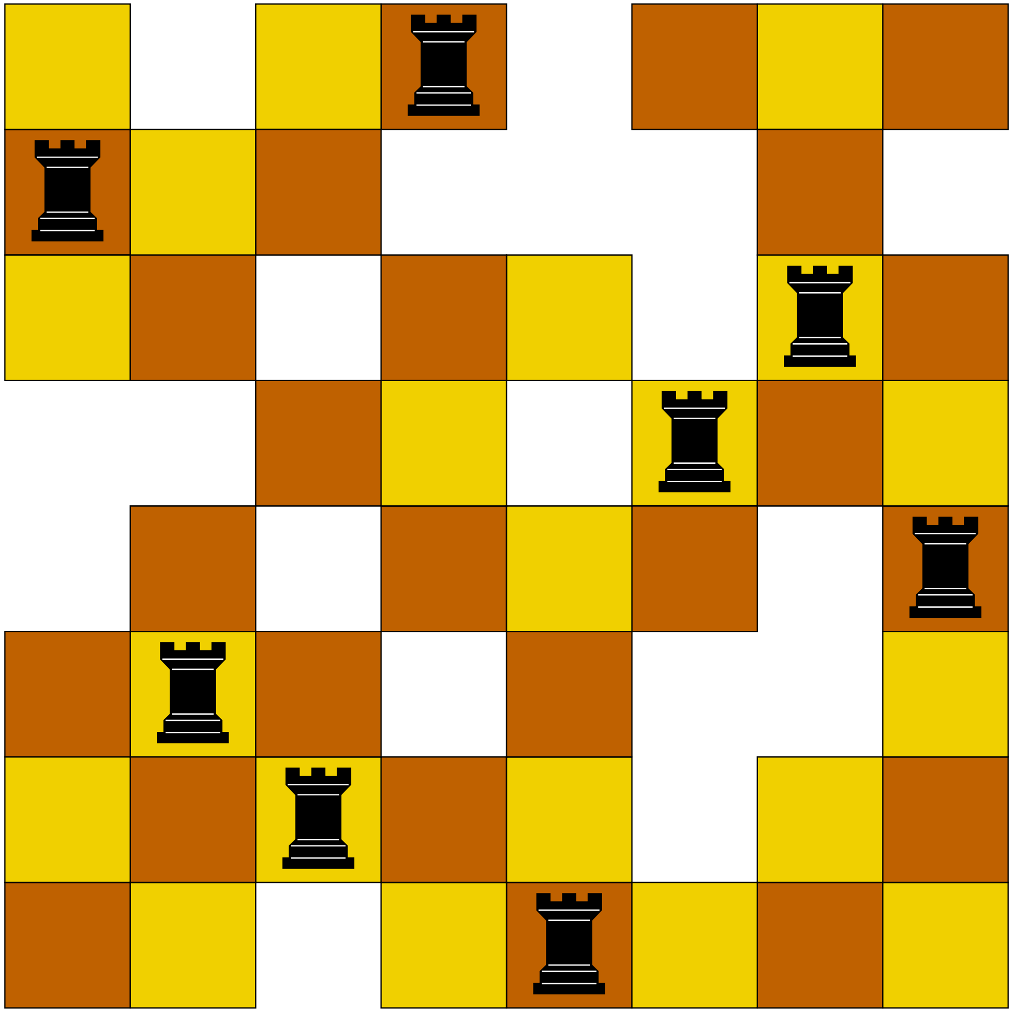
Chess Board



Chess Board

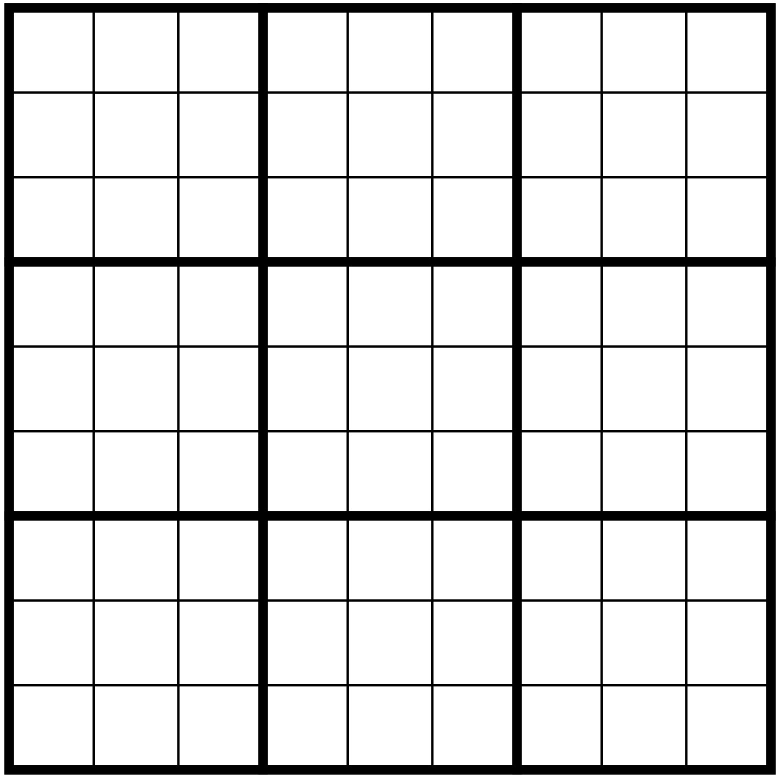


Chess Board



Question: in how many ways can we place 8 non-attacking rooks on this modified chess board?

Sudoku



Sudoku

1	2	5	3	9	6	8	7	4
4	6	3	8	7	5	2	9	1
7	9	8	2	4	1	5	3	6
5	4	7	6	1	2	9	8	3
2	3	9	5	8	4	1	6	7
8	1	6	9	3	7	4	2	5
6	8	1	7	5	9	3	4	2
3	7	4	1	2	8	6	5	9
9	5	2	4	6	3	7	1	8

Sudoku

1	2	5	3	9	6	8	7	4
4	6	3	8	7	5	2	9	1
7	9	8	2	4	1	5	3	6
5	4	7	6	1	2	9	8	3
2	3	9	5	8	4	1	6	7
8	1	6	9	3	7	4	2	5
6	8	1	7	5	9	3	4	2
3	7	4	1	2	8	6	5	9
9	5	2	4	6	3	7	1	8

Question: how many Sudoku arrays are there?

(More technically: how many valid configurations are there?)

Sudoku

1	2	5	3	9	6	8	7	4
4	6	3	8	7	5	2	9	1
7	9	8	2	4	1	5	3	6
5	4	7	6	1	2	9	8	3
2	3	9	5	8	4	1	6	7
8	1	6	9	3	7	4	2	5
6	8	1	7	5	9	3	4	2
3	7	4	1	2	8	6	5	9
9	5	2	4	6	3	7	1	8

Row condition: numbers $1, \ldots, 9$ appear exactly once.

Sudoku

1	2	5	3	9	6	8	7	4
4	6	3	8	7	5	2	9	1
7	9	8	2	4	1	5	3	6
5	4	7	6	1	2	9	8	3
2	3	9	5	8	4	1	6	7
8	1	6	9	3	7	4	2	5
6	8	1	7	5	9	3	4	2
3	7	4	1	2	8	6	5	9
9	5	2	4	6	3	7	1	8

Column condition: numbers $1, \ldots, 9$ appear exactly once.

Sudoku

1	2	5	3	9	6	8	7	4
4	6	3	8	7	5	2	9	1
7	9	8	2	4	1	5	3	6
5	4	7	6	1	2	9	8	3
2	3	9	5	8	4	1	6	7
8	1	6	9	3	7	4	2	5
6	8	1	7	5	9	3	4	2
3	7	4	1	2	8	6	5	9
9	5	2	4	6	3	7	1	8

Sub-block condition: numbers $1, \ldots, 9$ appear exactly once.

Sudoku

1	2	5	3	9	6	8	7	4
4	6	3	8	7	5	2	9	1
7	9	8	2	4	1	5	3	6
5	4	7	6	1	2	9	8	3
2	3	9	5	8	4	1	6	7
8	1	6	9	3	7	4	2	5
6	8	1	7	5	9	3	4	2
3	7	4	1	2	8	6	5	9
9	5	2	4	6	3	7	1	8

Question: how many Sudoku arrays are there?

(More technically: how many valid configurations are there?)

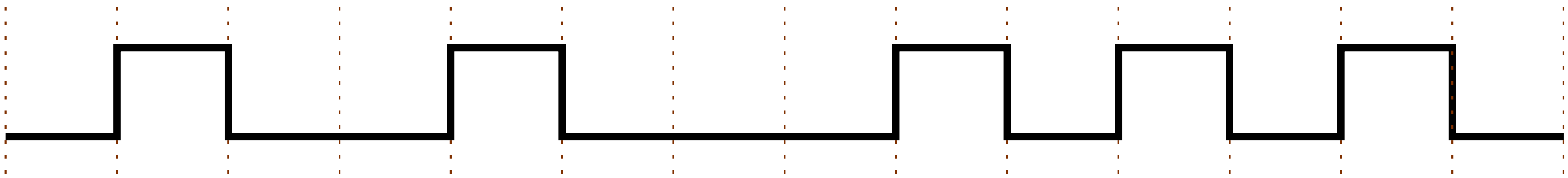
Other Sudoku Setups

Other Sudoku Setups

3	5	8	1	9	6	2	7	4
4	9	2	5	6	7	1	3	8
6	1	3	9	7	8	4	2	5
1	7	5	8	4	2	6	9	3
8	2	6	4	5	3	7	1	9
2	4	9	7	3	1	8	5	6
9	8	7	3	2	4	5	6	1
7	3	4	6	1	5	9	8	2
5	6	1	2	8	9	3	4	7

1D constraints in communications

RLL Constraints

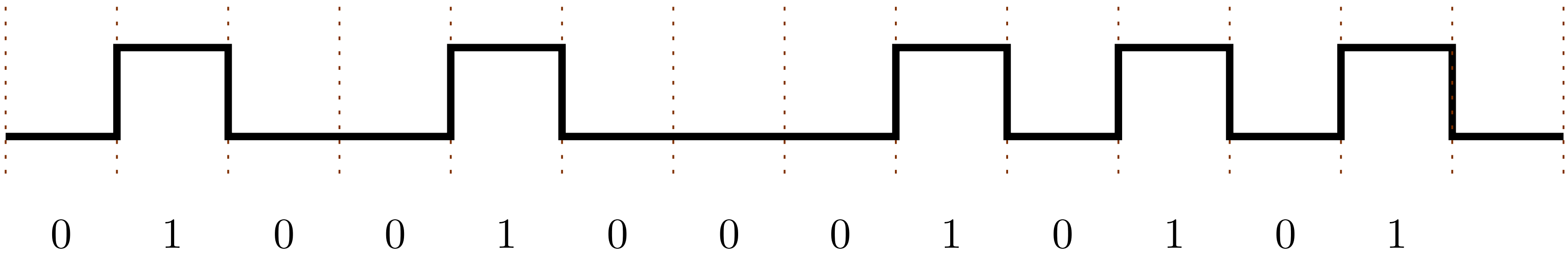


0 1 0 0 1 0 0 0 1 0 1 0 1

A (d, k) RLL constraint imposes:

- At least d zero symbols between two ones.
- At most k zero symbols between two ones.

RLL Constraints

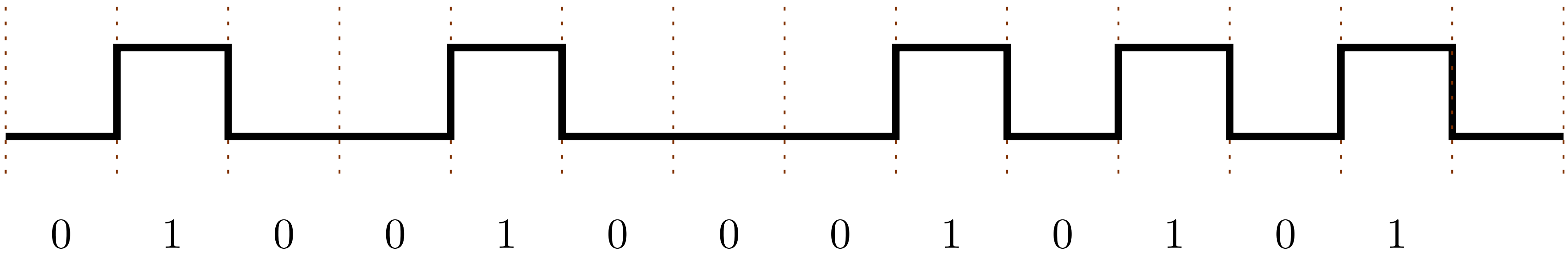


A (d, k) RLL constraint imposes:

- At least d zero symbols between two ones.
- At most k zero symbols between two ones.

Question: how many sequences of length T fulfill these constraints?

RLL Constraints



A (d, k) RLL constraint imposes:

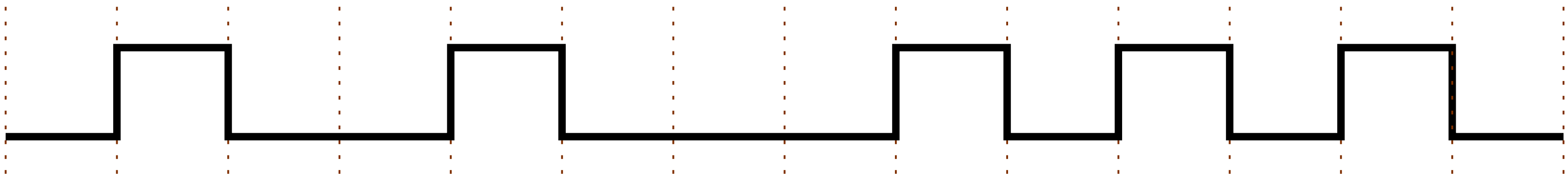
- At least d zero symbols between two ones.
- At most k zero symbols between two ones.

Question: how many sequences of length T fulfill these constraints?

Answer: typically, the answer to such questions looks like

$$N(T) = \exp\big(C \cdot T + o(T)\big).$$

RLL Constraints



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

A (d, k) RLL constraint imposes:

- At least d zero symbols between two ones.
- At most k zero symbols between two ones.

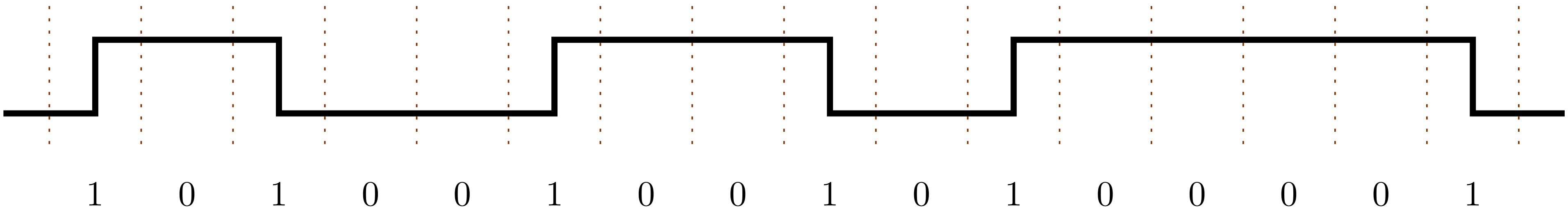
Question: how many sequences of length T fulfill these constraints?

Answer: typically, the answer to such questions looks like

$$N(T) = \exp\big(C \cdot T + o(T)\big).$$

C: "capacity" or "entropy."

RLL Constraints



A (d, k) RLL constraint imposes:

- At least d zero symbols between two ones.
- At most k zero symbols between two ones.

Question: how many sequences of length T fulfill these constraints?

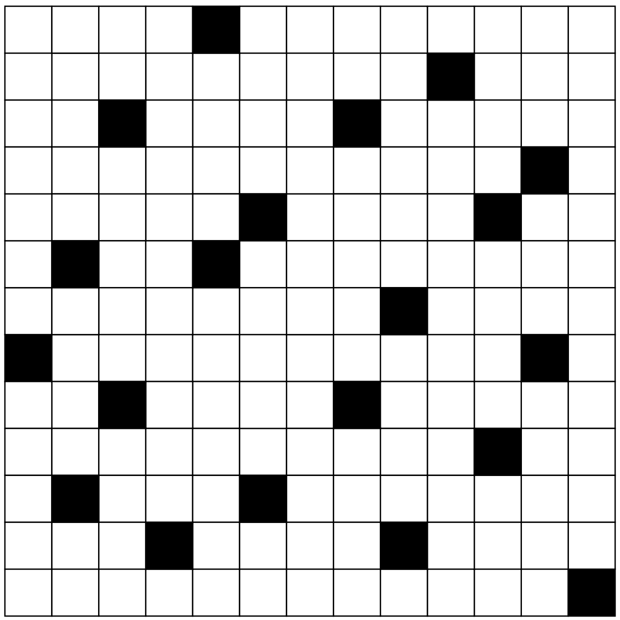
Answer: typically, the answer to such questions looks like

$$N(T) = \exp\big(C \cdot T + o(T)\big).$$

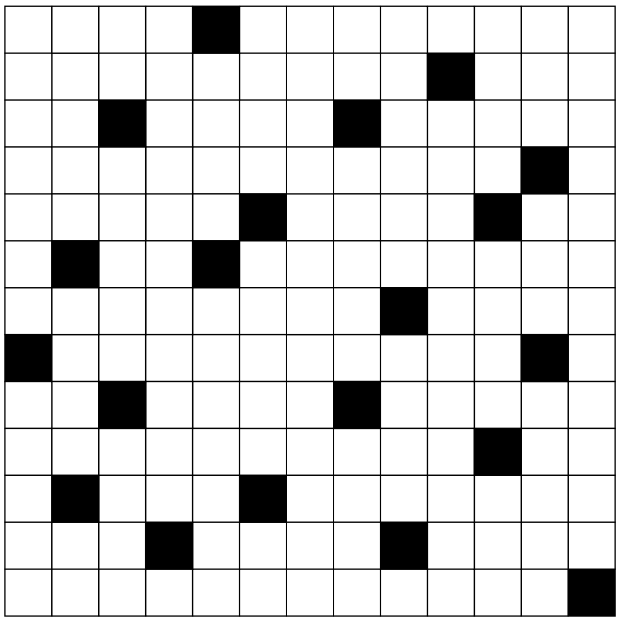
C: "capacity" or "entropy."

2D constraints in communications

Two-Dimensional RLL Constraints



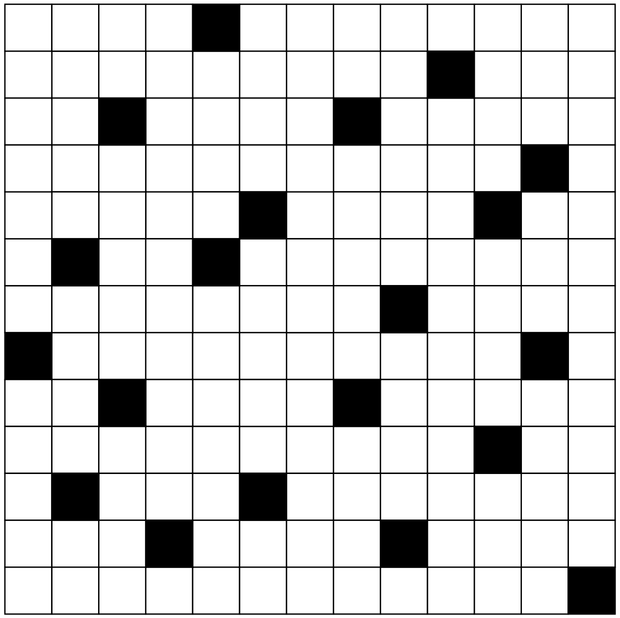
Two-Dimensional RLL Constraints



A $(d_1, k; d_2, k_2)$ RLL constraint imposes:

- ...
- ...

Two-Dimensional RLL Constraints

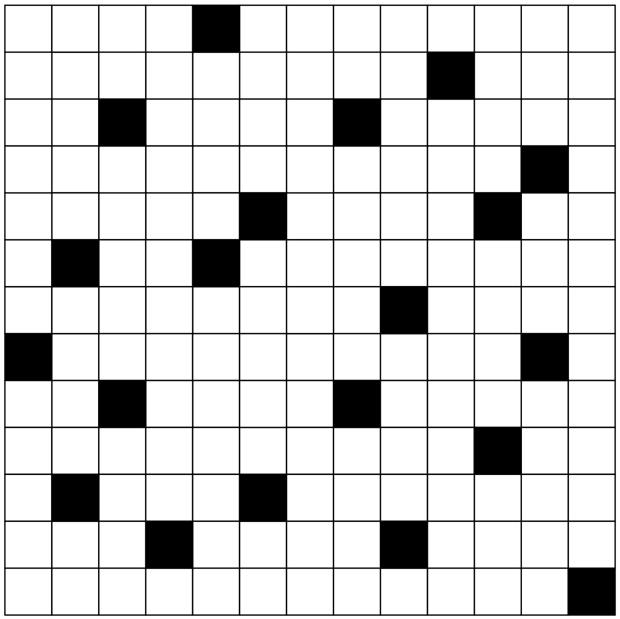


A $(d_1, k; d_2, k_2)$ RLL constraint imposes:

- ...
- ...

Question: How many arrays of size $m \times n$ fulfill these constraints?

Two-Dimensional RLL Constraints



A $(d_1, k; d_2, k_2)$ RLL constraint imposes:

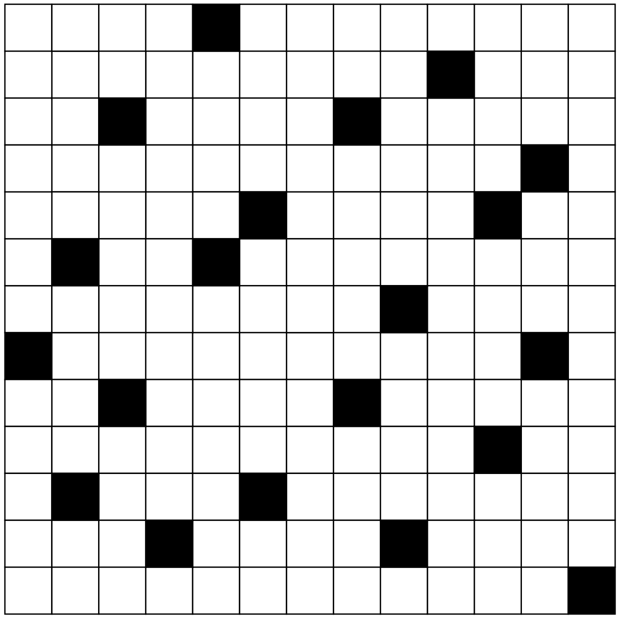
- $\ldots$
- $\ldots$

Question: How many arrays of size $m \times n$ fulfill these constraints?

Answer: Typically, the answer to such questions looks like

$$N(m,n) = \exp\left(C \cdot mn + o(mn)\right).$$

Two-Dimensional RLL Constraints



A $(d_1, k; d_2, k_2)$ RLL constraint imposes:

- ...
- ...

Question: How many arrays of size $m \times n$ fulfill these constraints?

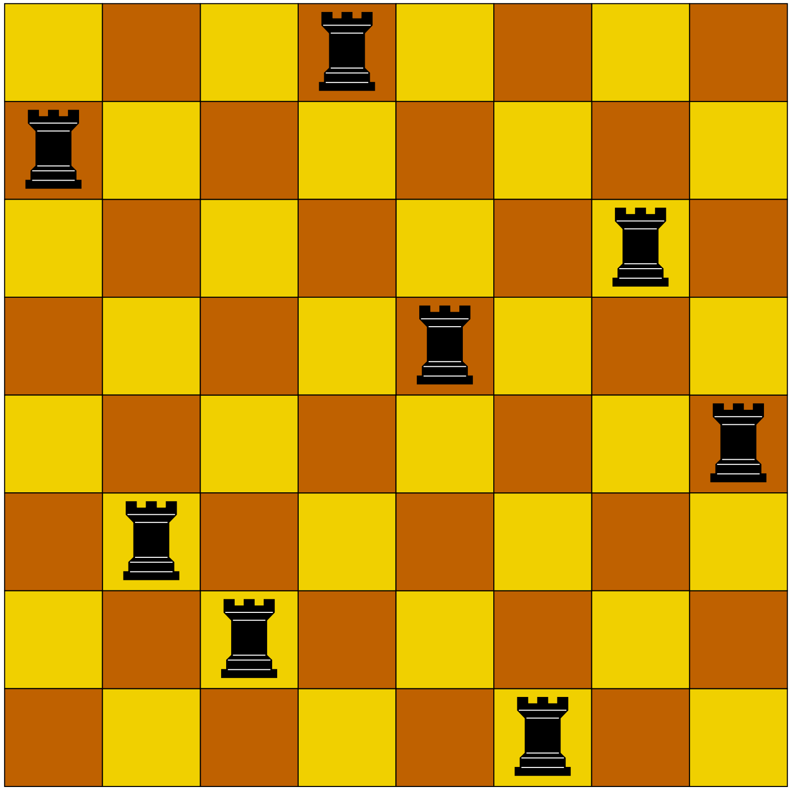
Answer: Typically, the answer to such questions looks like

$$N(m,n) = \exp\left(C \cdot mn + o(mn)\right).$$

C: "capacity" or "entropy."

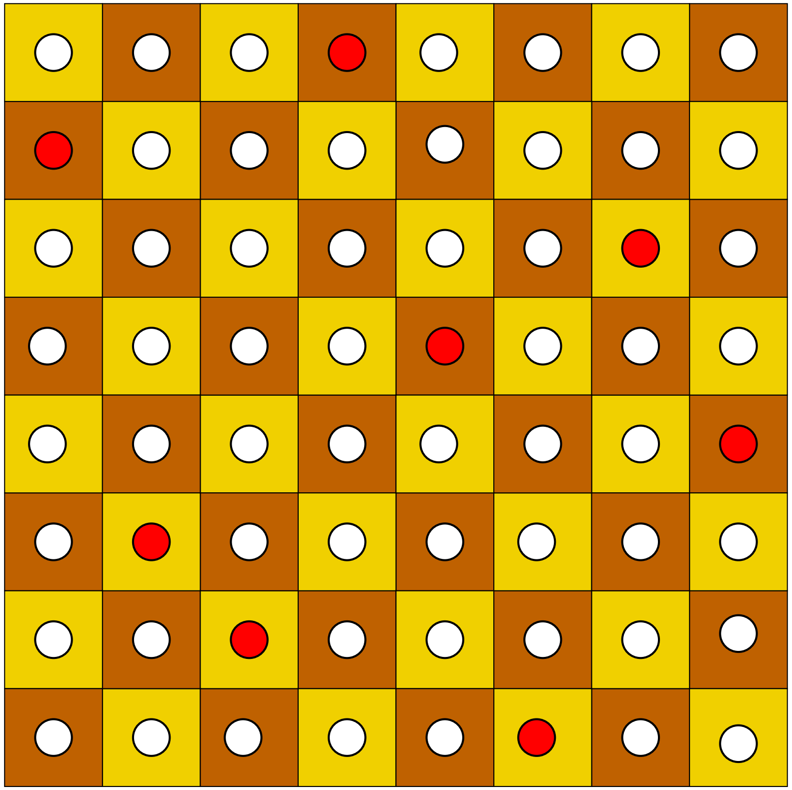
Towards a graphical model

Towards a Graphical Model

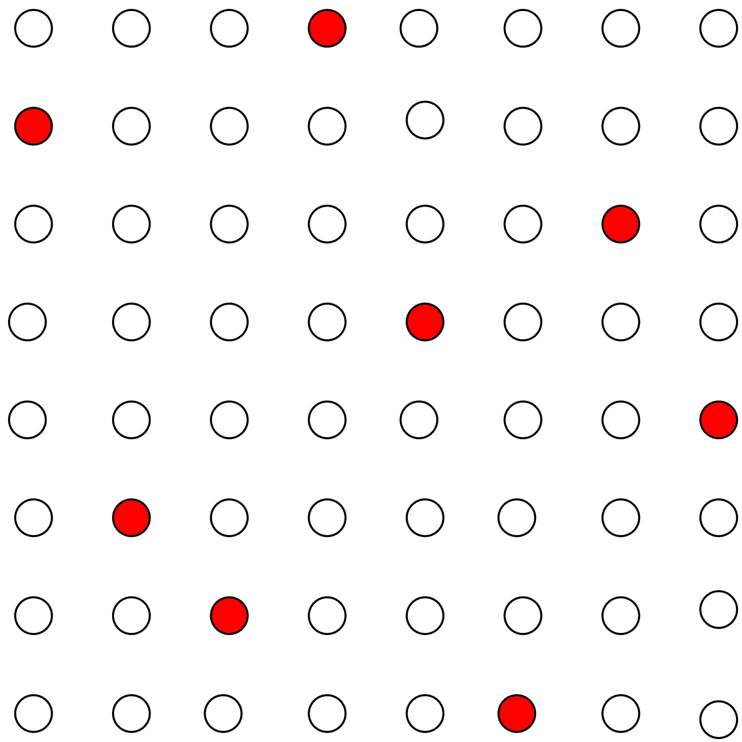


Question: in how many ways can we place 8 non-attacking rooks on a chess board?

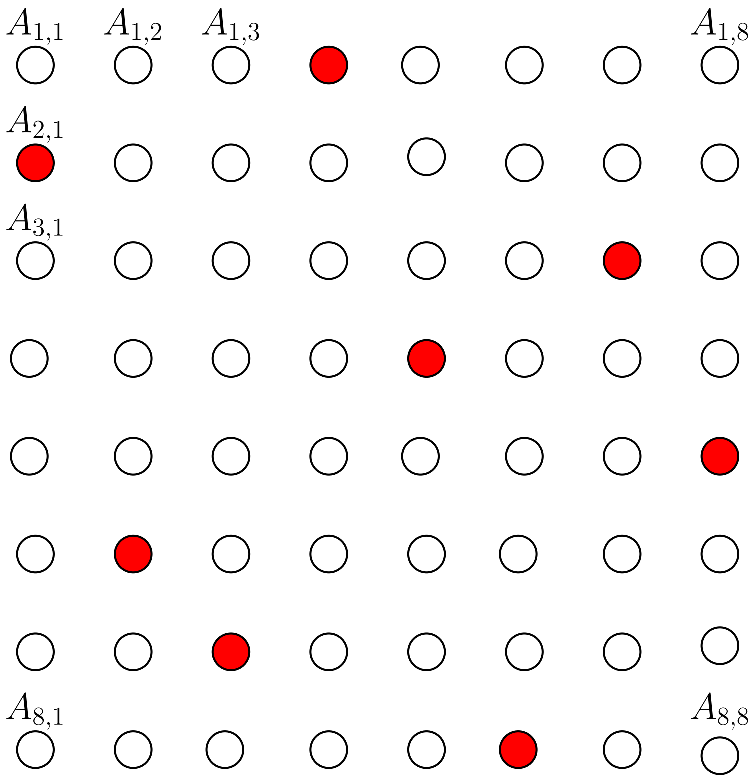
Towards a Graphical Model



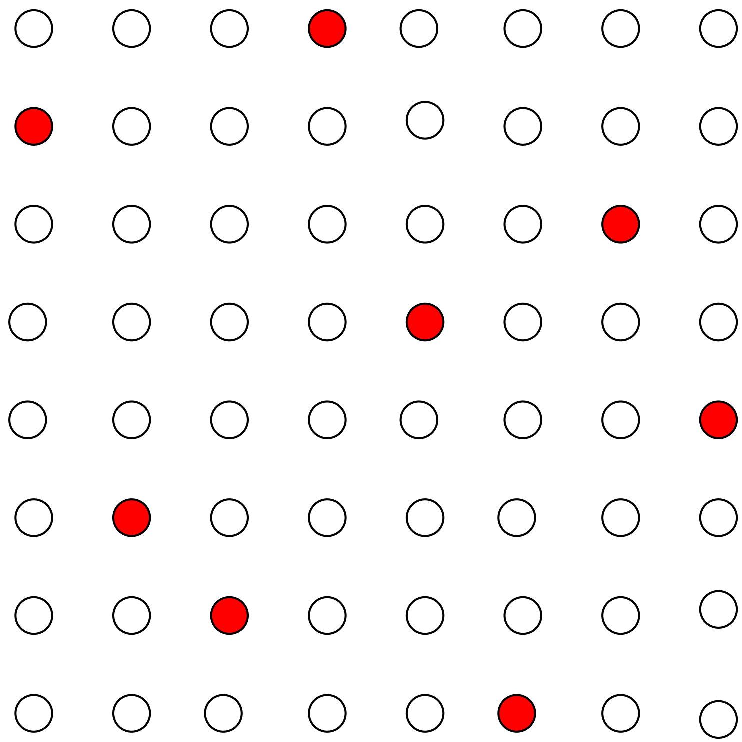
Towards a Graphical Model



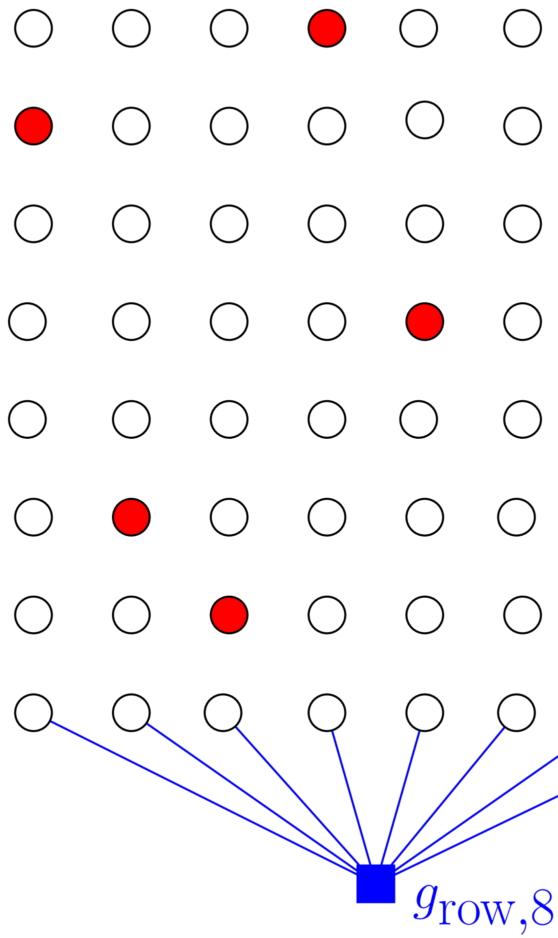
Towards a Graphical Model



Towards a Graphical Model

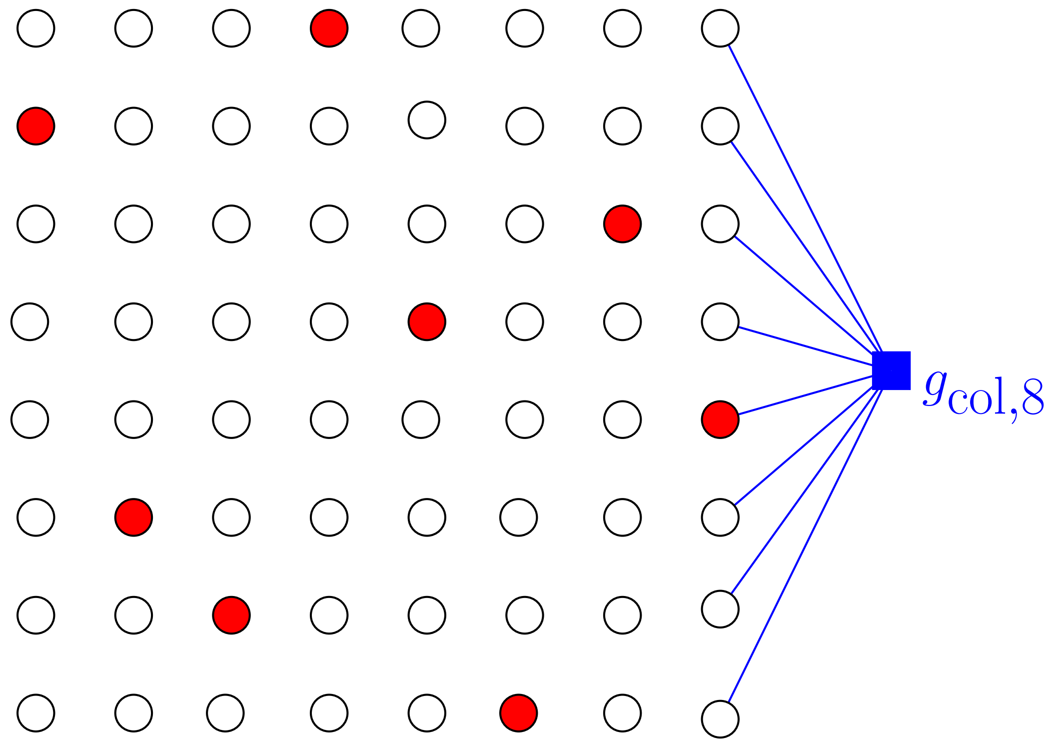


Towards a Graphical Model



$$g_{\mathrm{row},8}(a_{8,1},\ldots,a_{8,8}) \triangleq \begin{cases} 1 & \text{exactly one rook} \\ 0 & \text{otherwise} \end{cases}$$

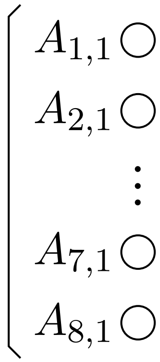
Towards a Graphical Model



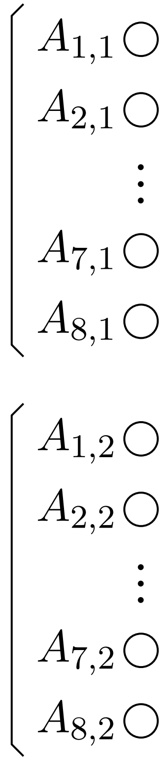
$$g_{\mathrm{col},8}(a_{1,8}, \ldots, a_{8,8}) \triangleq \begin{cases} 1 & \text{exactly one rook} \\ 0 & \text{otherwise} \end{cases}$$

Towards a Graphical Model

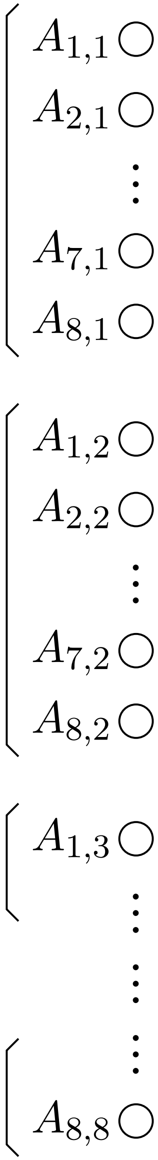
Towards a Graphical Model



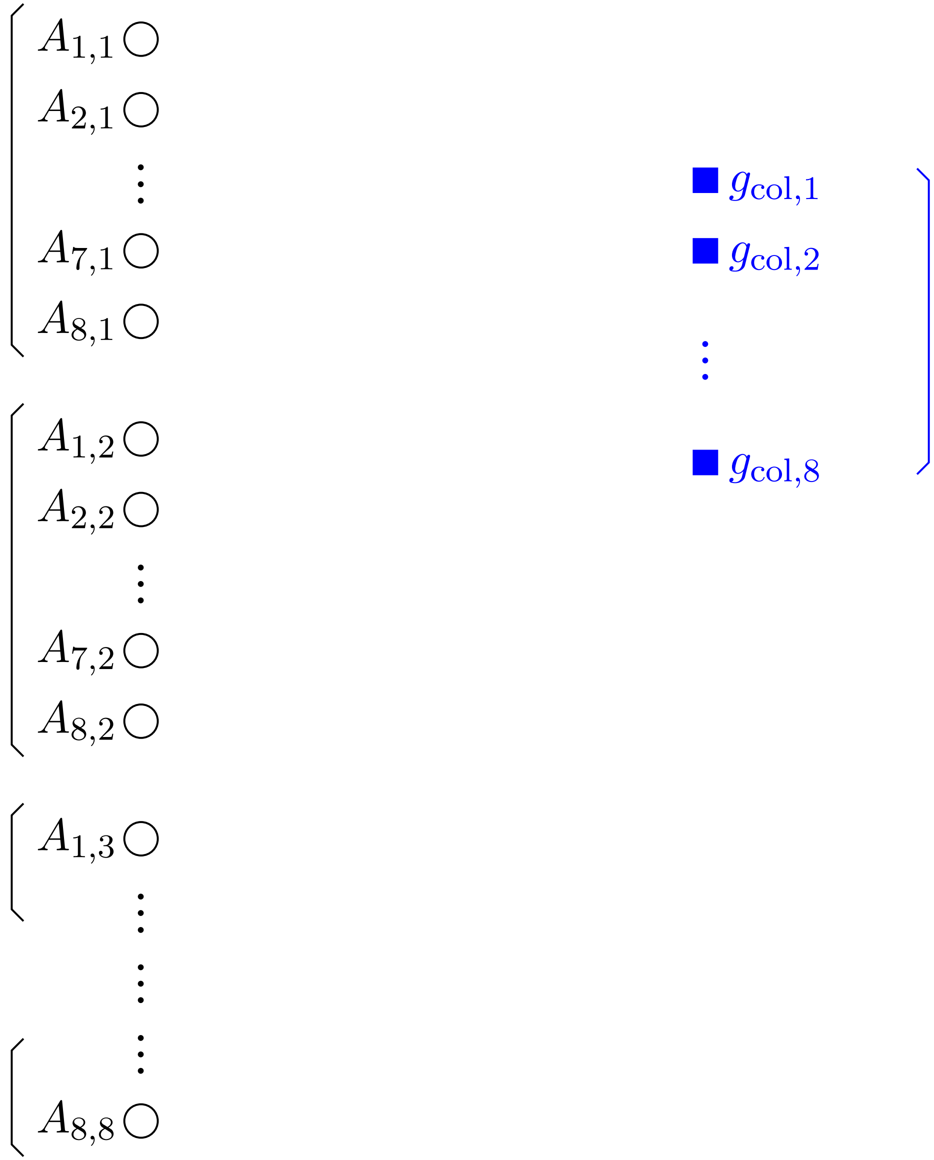
Towards a Graphical Model



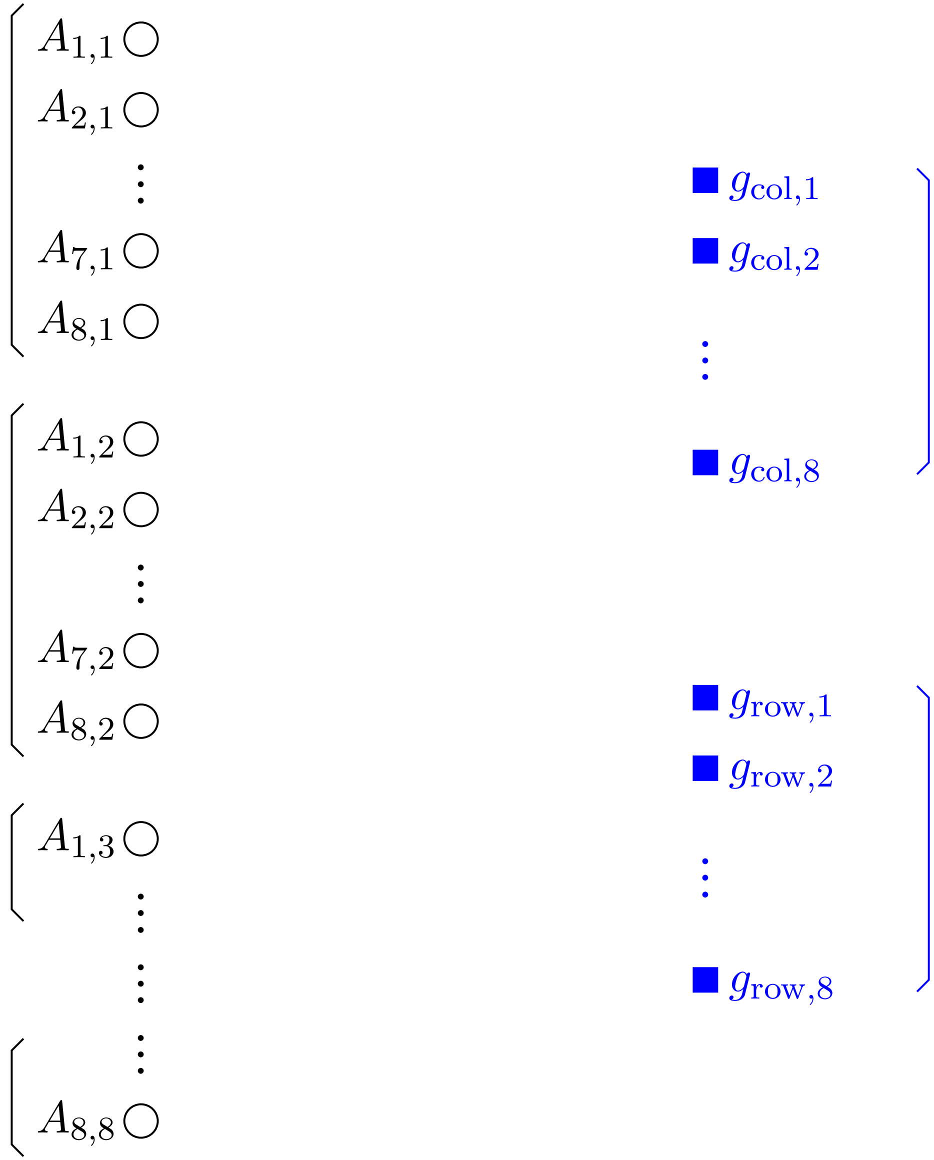
Towards a Graphical Model



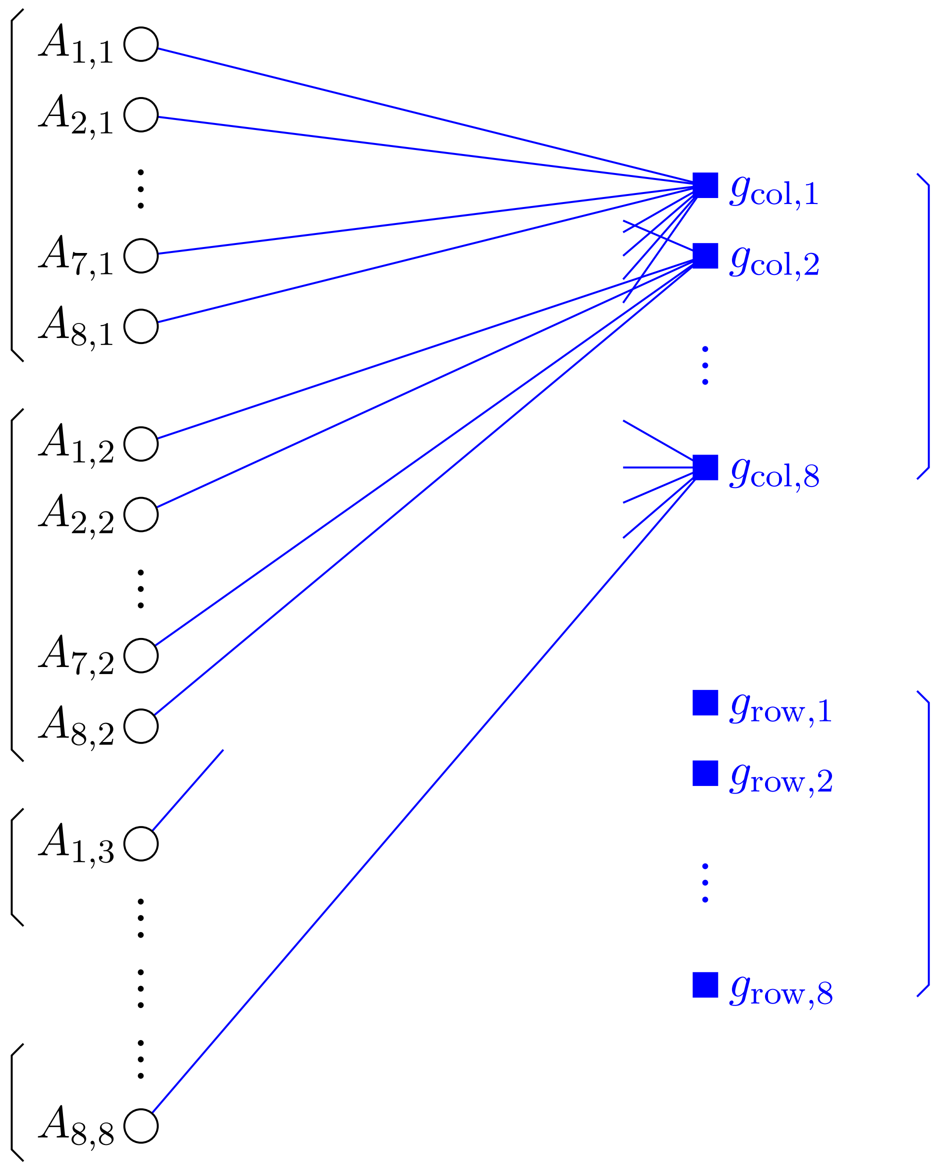
Towards a Graphical Model



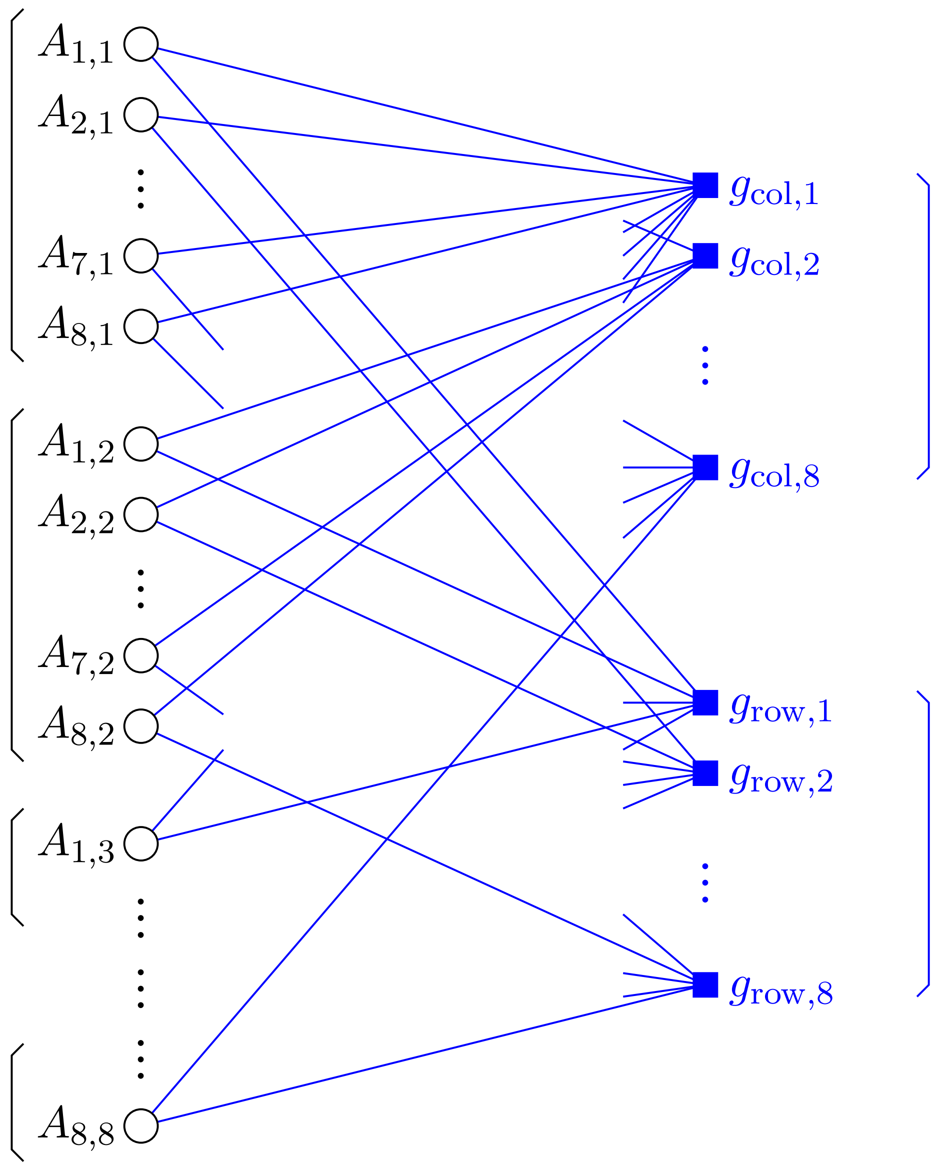
Towards a Graphical Model



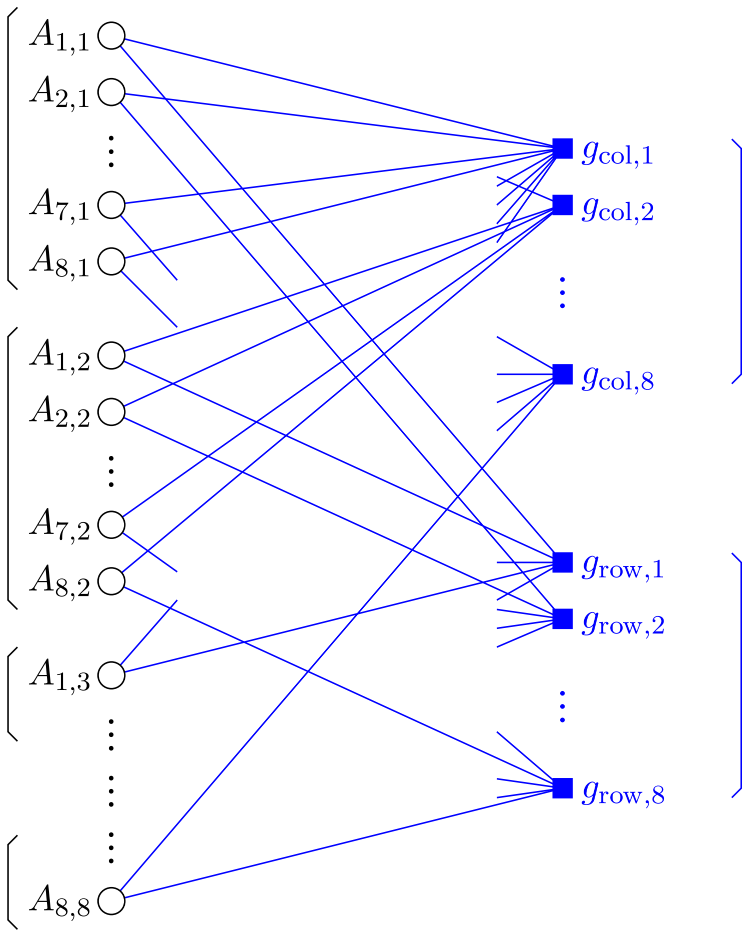
Towards a Graphical Model



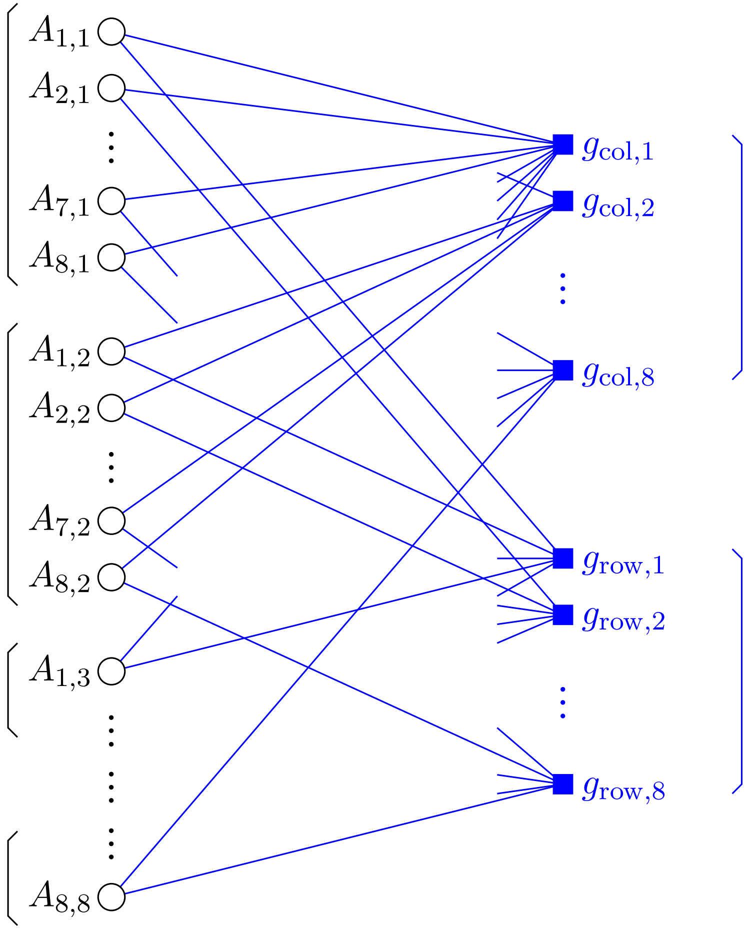
Towards a Graphical Model



Towards a Graphical Model



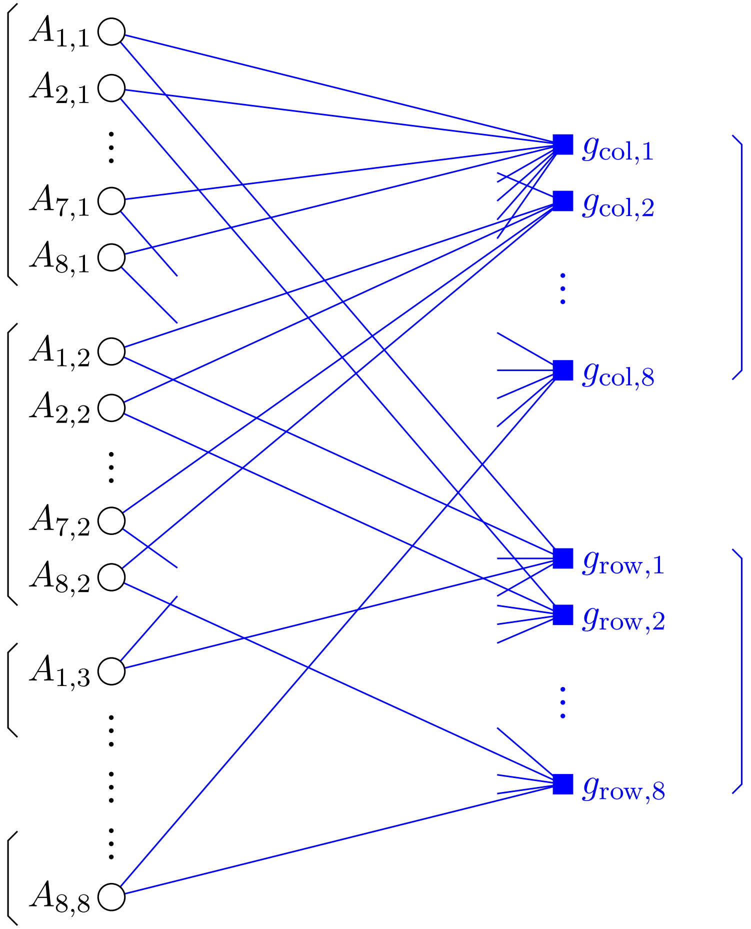
Towards a Graphical Model



Global function:

$$
\begin{aligned}
g(a_{1,1}, \ldots, a_{8,8}) \\
&= \prod_j g_{\mathrm{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \\
&\prod_i g_{\mathrm{row},i}(a_{i,1}, \ldots, a_{i,8})
\end{aligned}
$$

Towards a Graphical Model



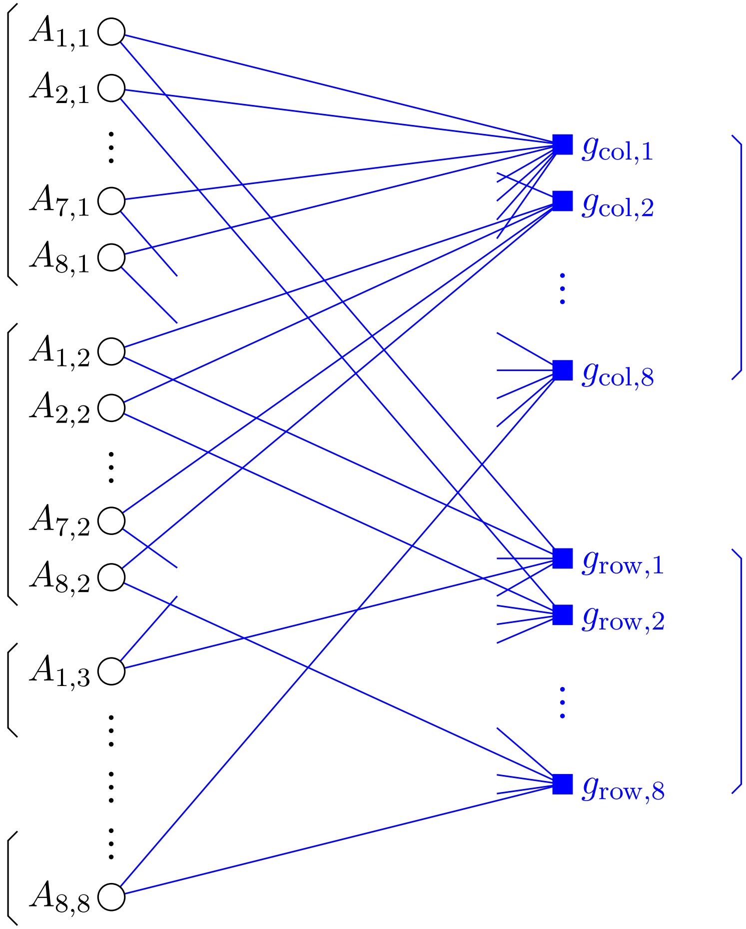
Global function:

$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\mathrm{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\mathrm{row},i}(a_{i,1}, \ldots, a_{i,8})$$

Total sum:

$$Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1}, \ldots, a_{8,8})$$

Towards a Graphical Model



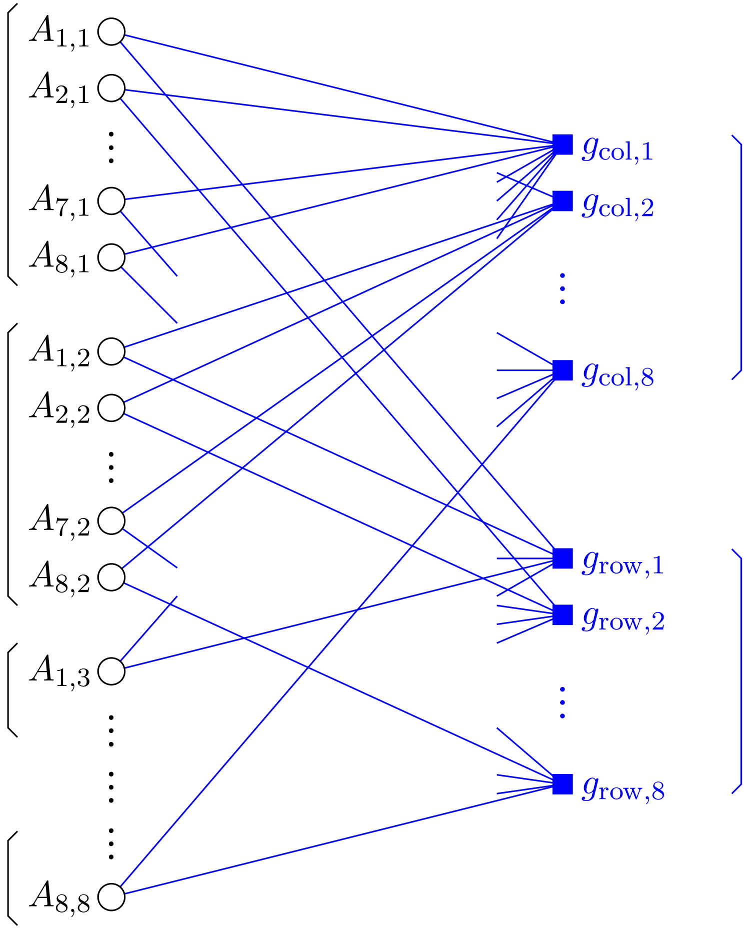
Global function:

$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\text{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\text{row},i}(a_{i,1}, \ldots, a_{i,8})$$

Total sum (partition function):

$$Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1}, \ldots, a_{8,8})$$

Towards a Graphical Model



Global function:

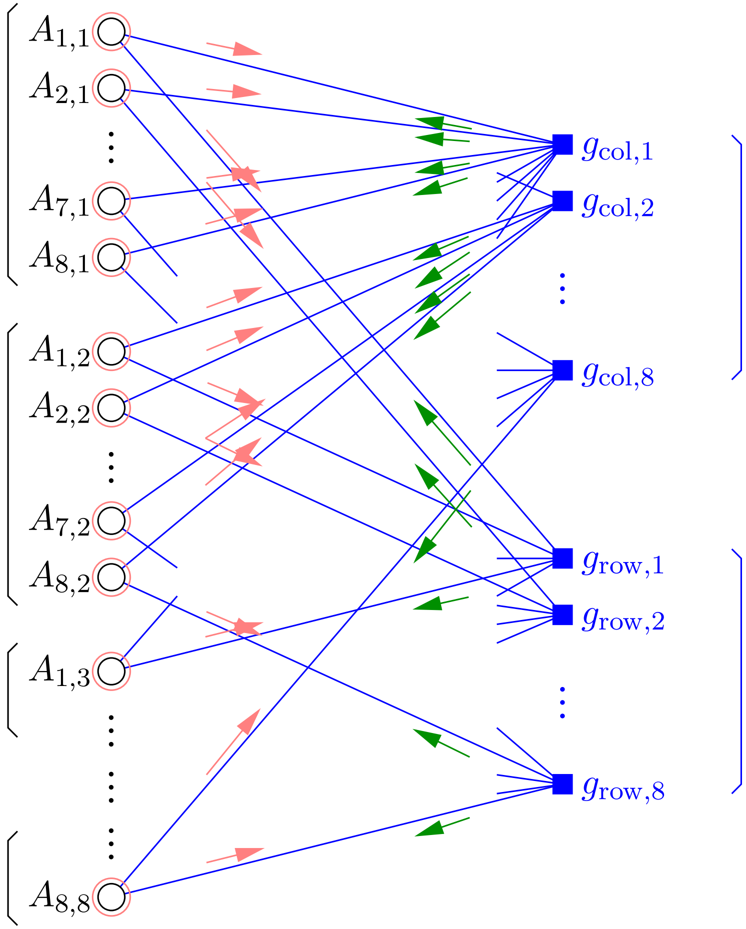
$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\mathrm{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\mathrm{row},i}(a_{i,1}, \ldots, a_{i,8})$$

Total sum (partition function):

$$Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1}, \ldots, a_{8,8})$$

Use of loopy belief propagation for approximating Z?

Towards a Graphical Model



Global function:

$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\mathrm{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\mathrm{row},i}(a_{i,1}, \ldots, a_{i,8})$$

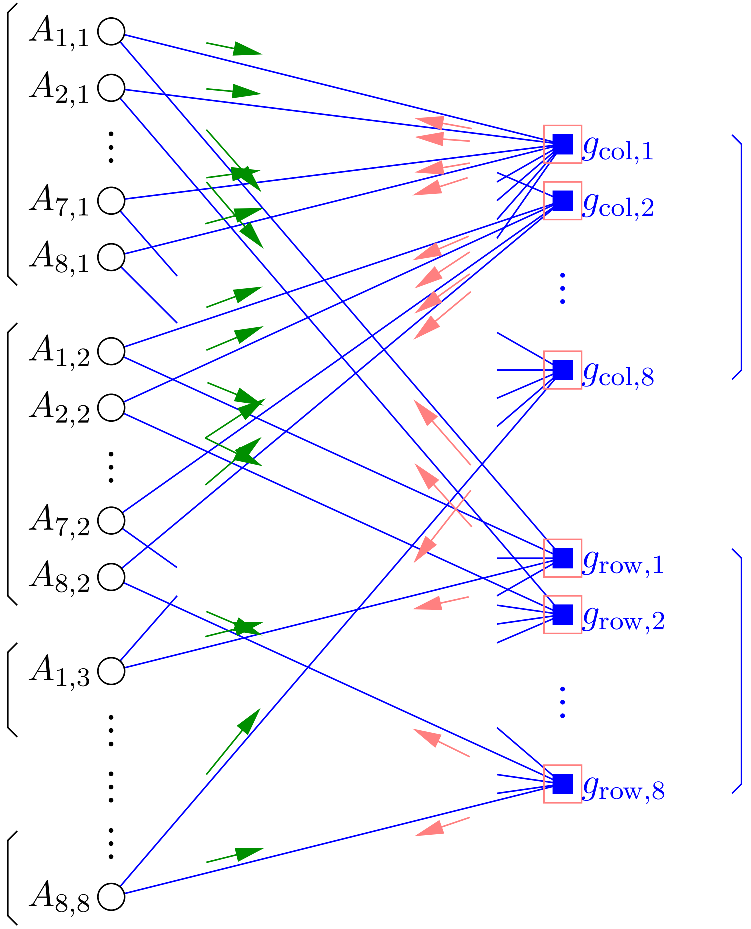
Total sum (partition function):

$$Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1}, \ldots, a_{8,8})$$

Use of loopy belief propagation for approximating Z?

LABS hp

Towards a Graphical Model



Global function:

$$g(a_{1,1},\ldots,a_{8,8}) = \prod_j g_{\text{col},j}(a_{1,j},\ldots,a_{8,j}) \times \prod_i g_{\text{row},i}(a_{i,1},\ldots,a_{i,8})$$

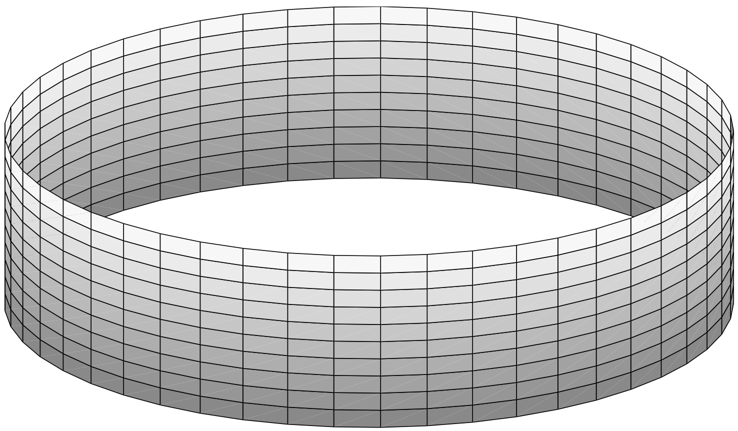
Total sum (partition function):

$$Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1},\ldots,a_{8,8})$$

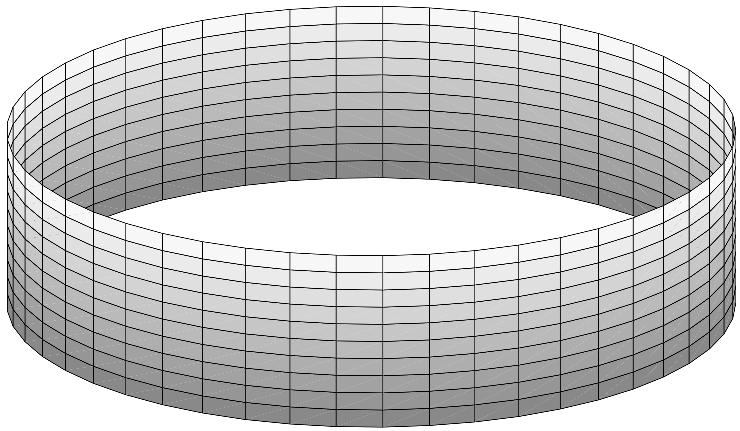
Use of loopy belief propagation for approximating Z?

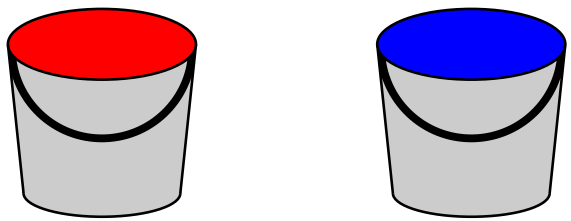
Some considerations on counting algorithms

Coloring the Surfaces of a Closed Strip

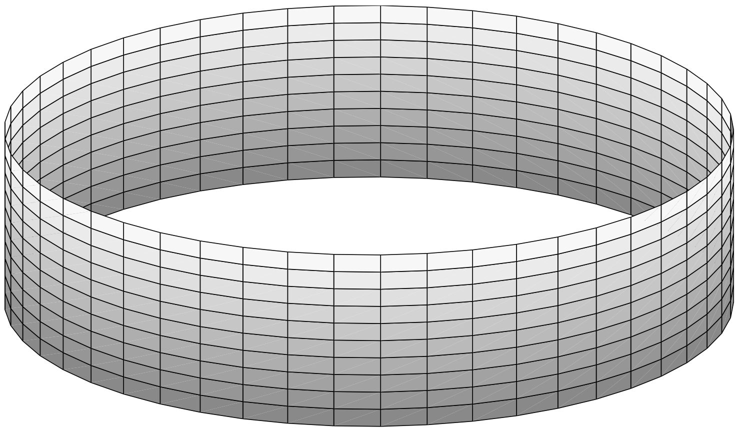


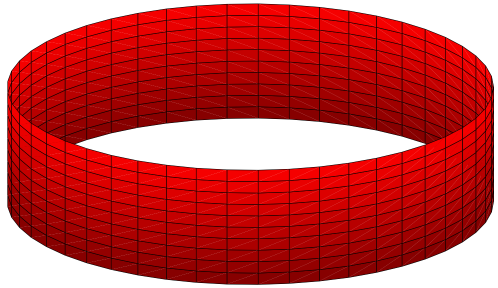
Coloring the Surfaces of a Closed Strip

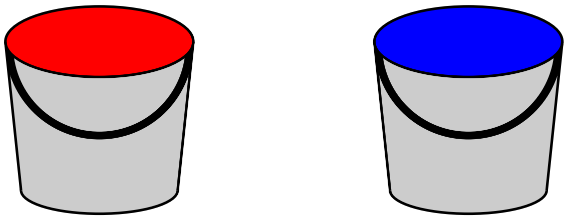




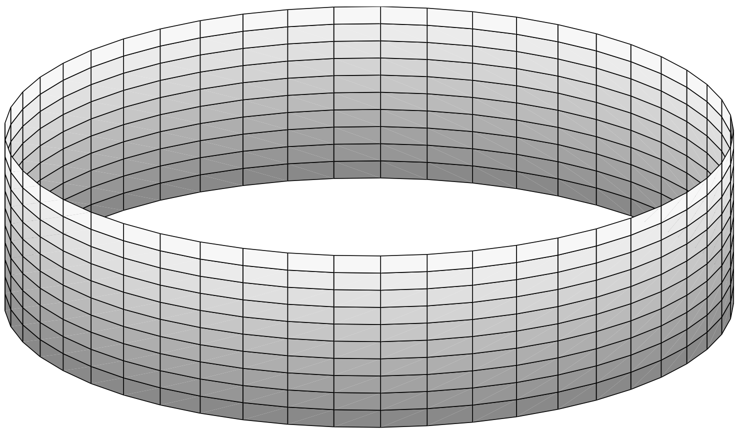
Coloring the Surfaces of a Closed Strip

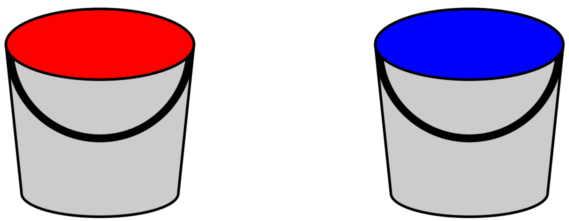


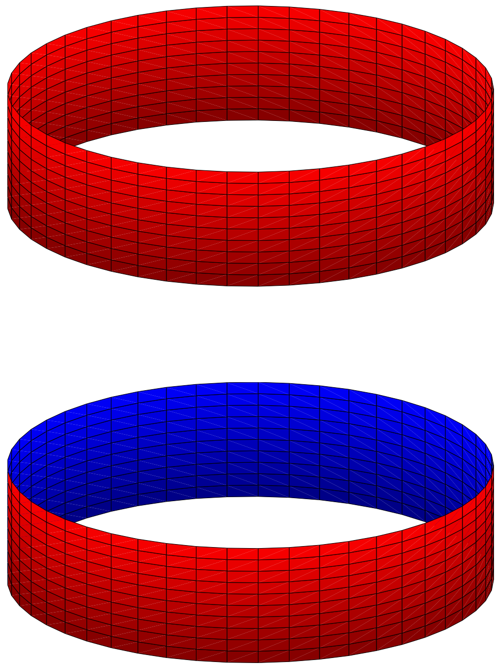




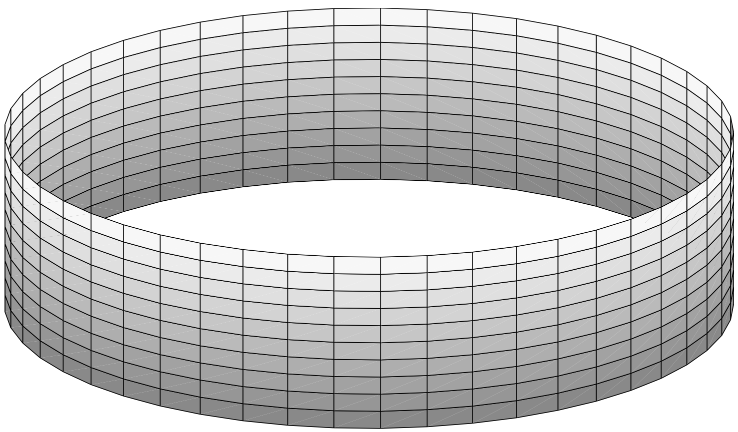
Coloring the Surfaces of a Closed Strip

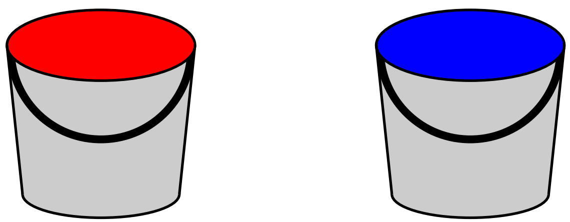


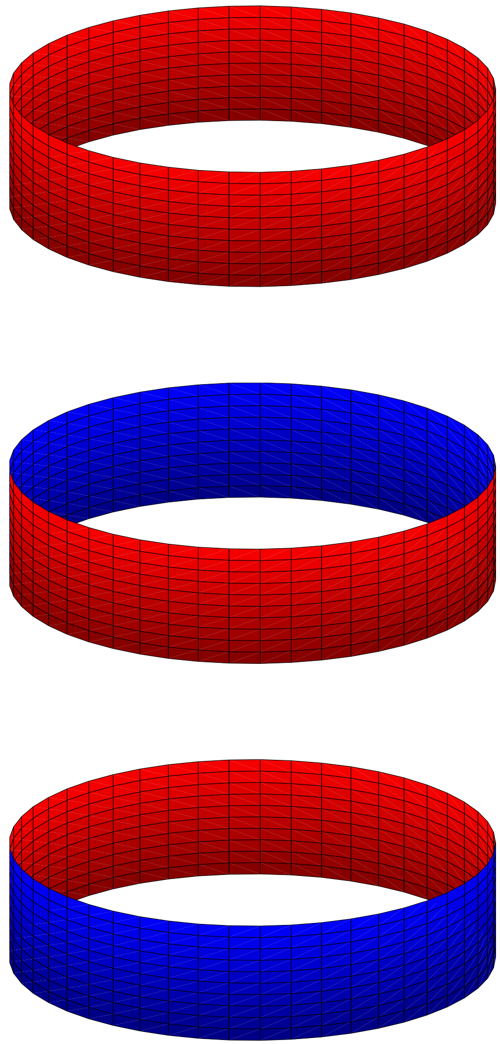




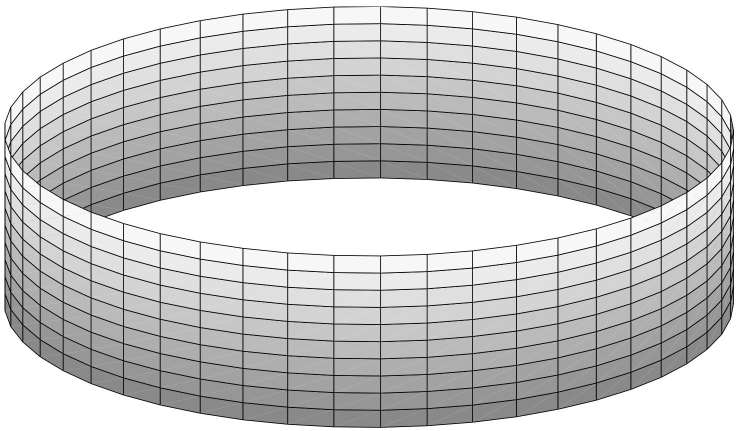
Coloring the Surfaces of a Closed Strip

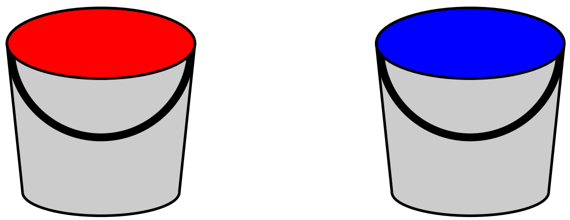


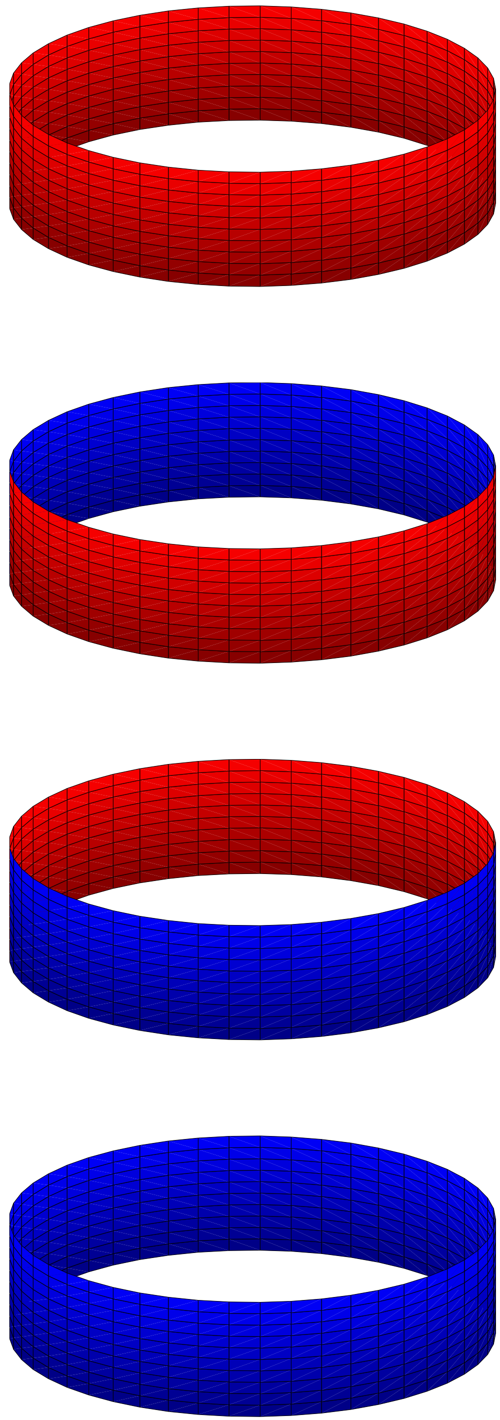




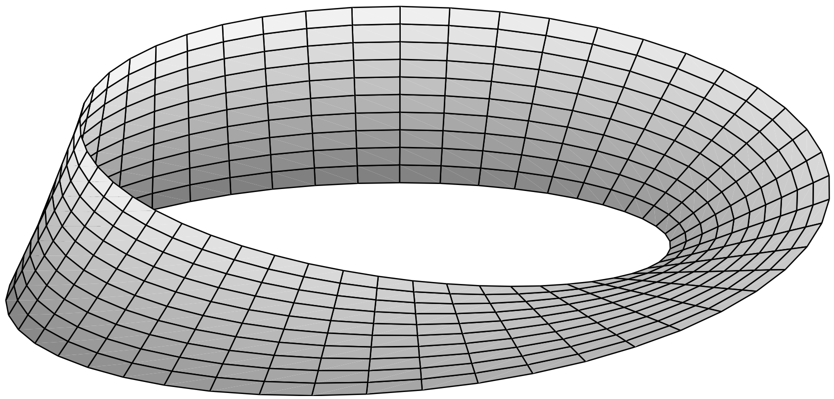
Coloring the Surfaces of a Closed Strip



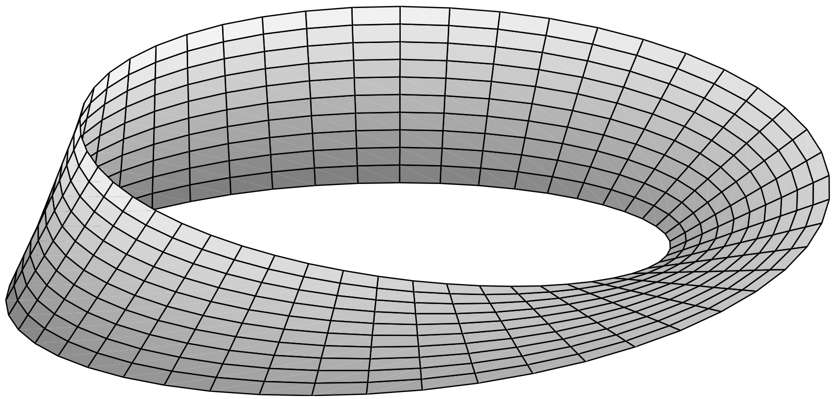


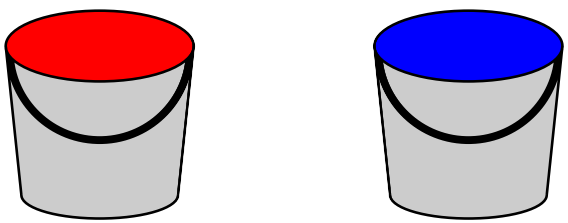


Coloring the Surfaces of a Closed Strip

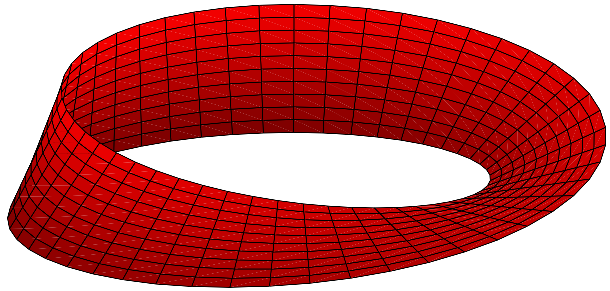


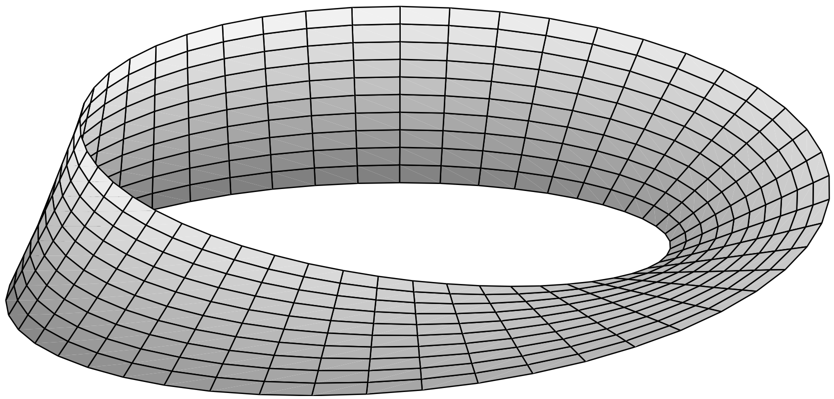
Coloring the Surfaces of a Closed Strip

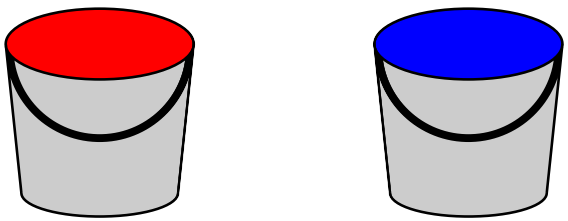




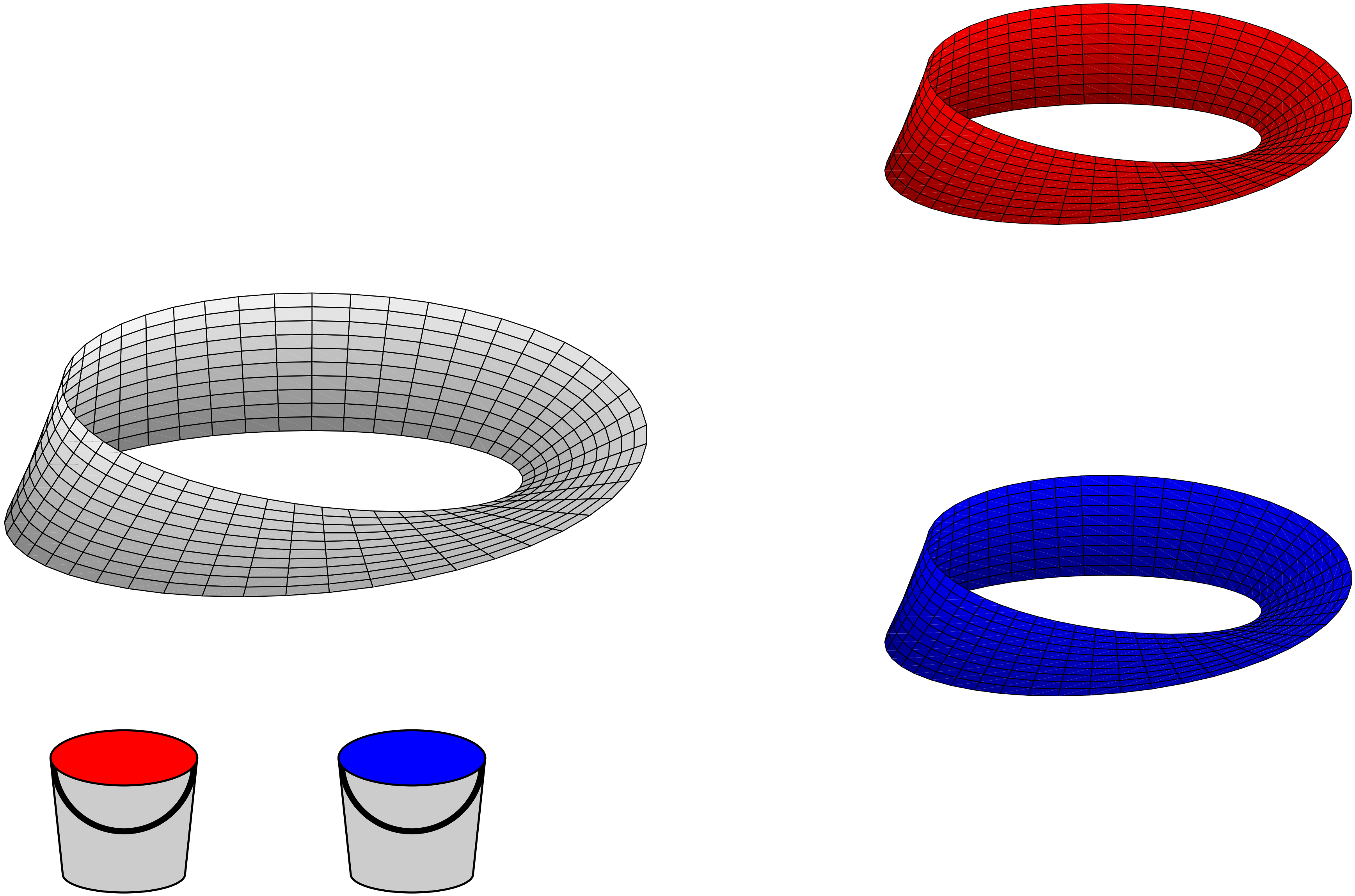
Coloring the Surfaces of a Closed Strip

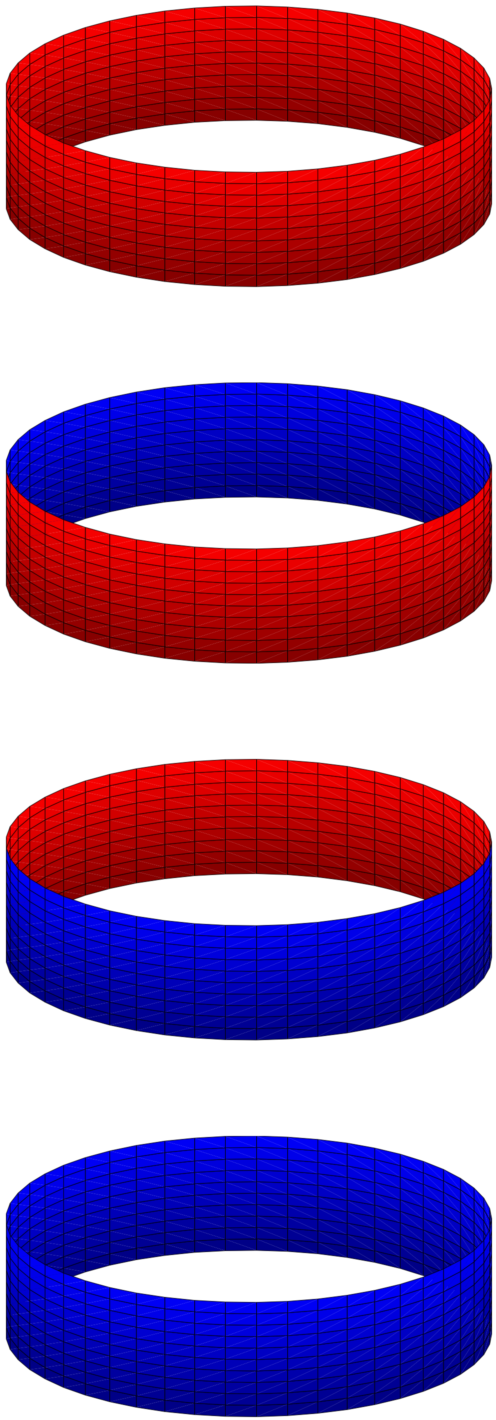


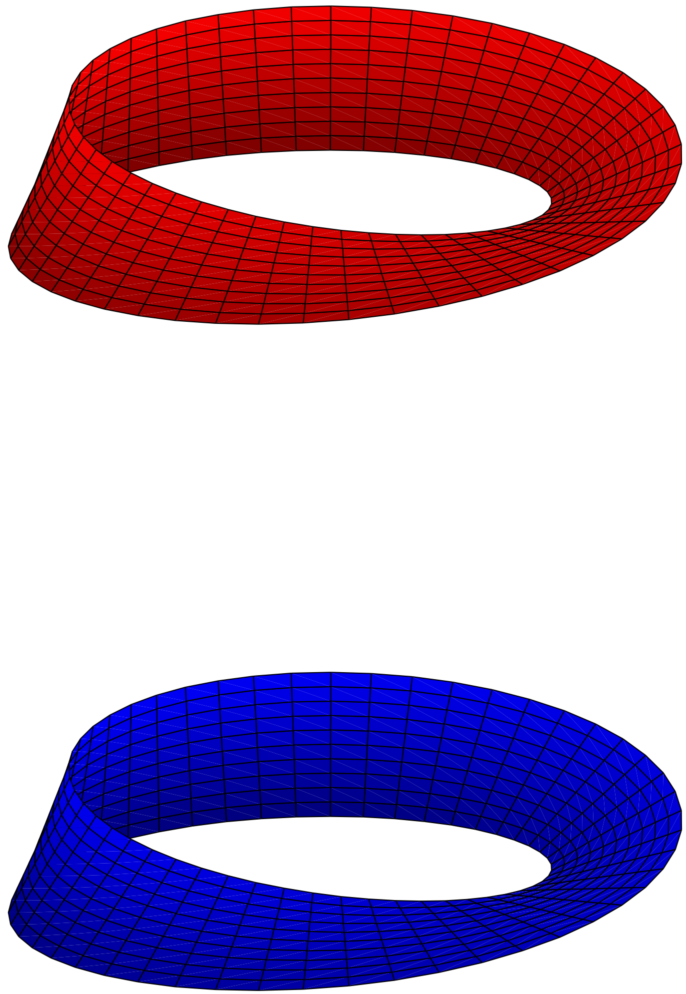


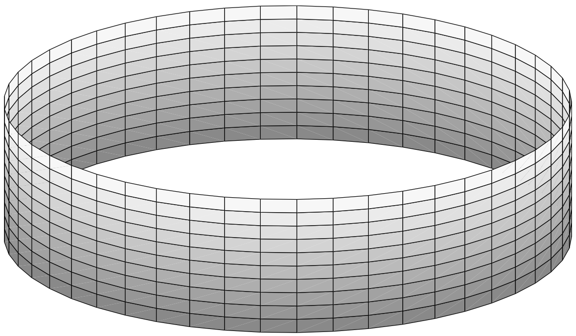


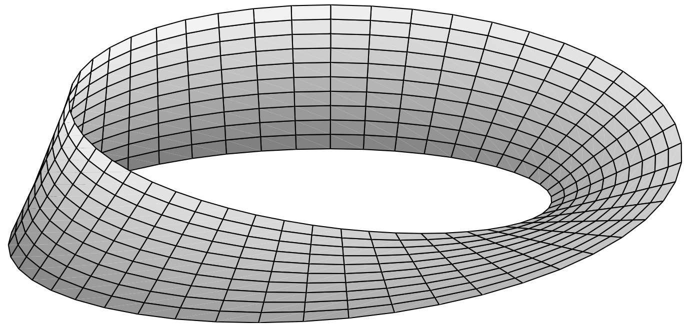
Coloring the Surfaces of a Closed Strip

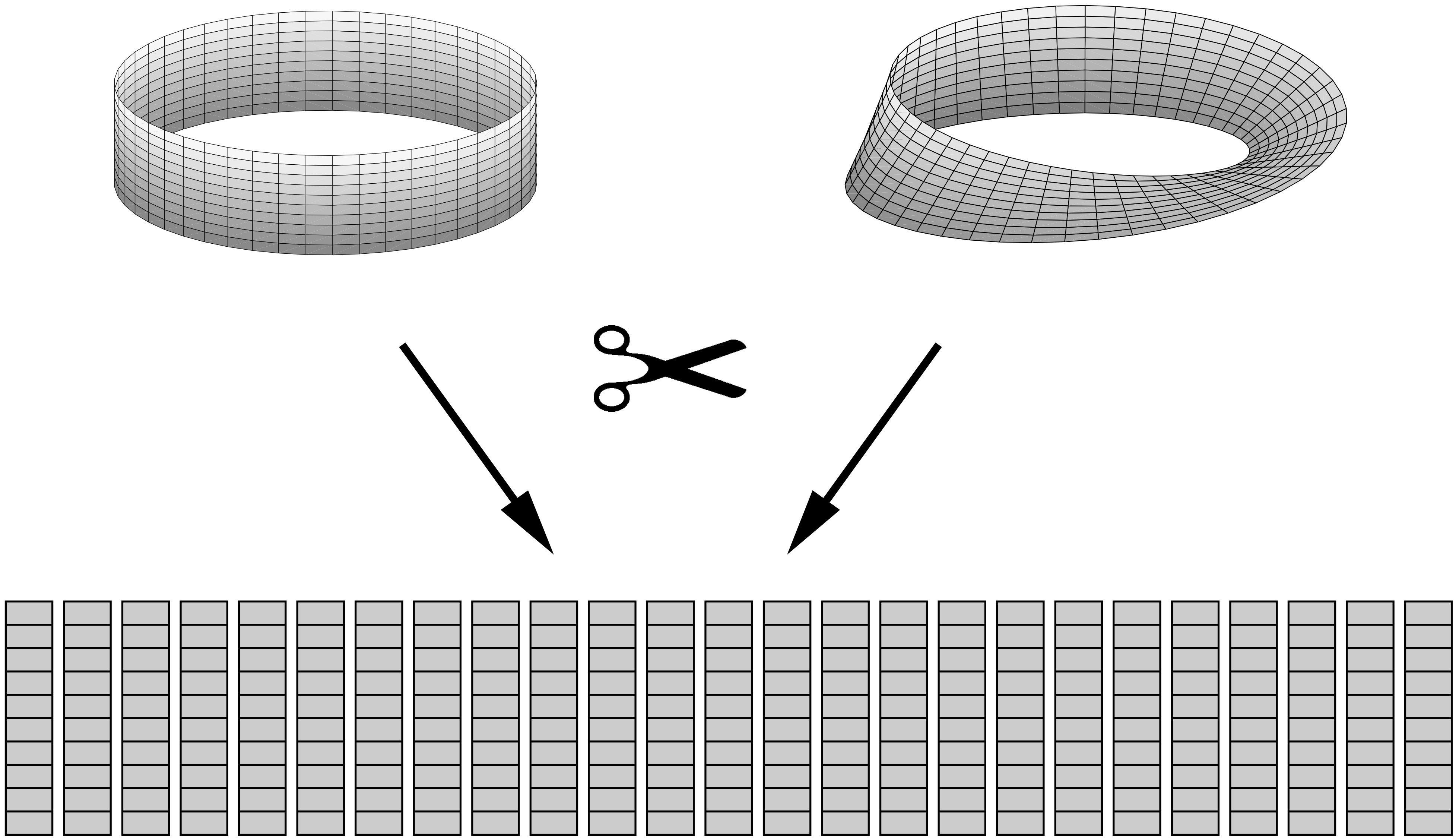




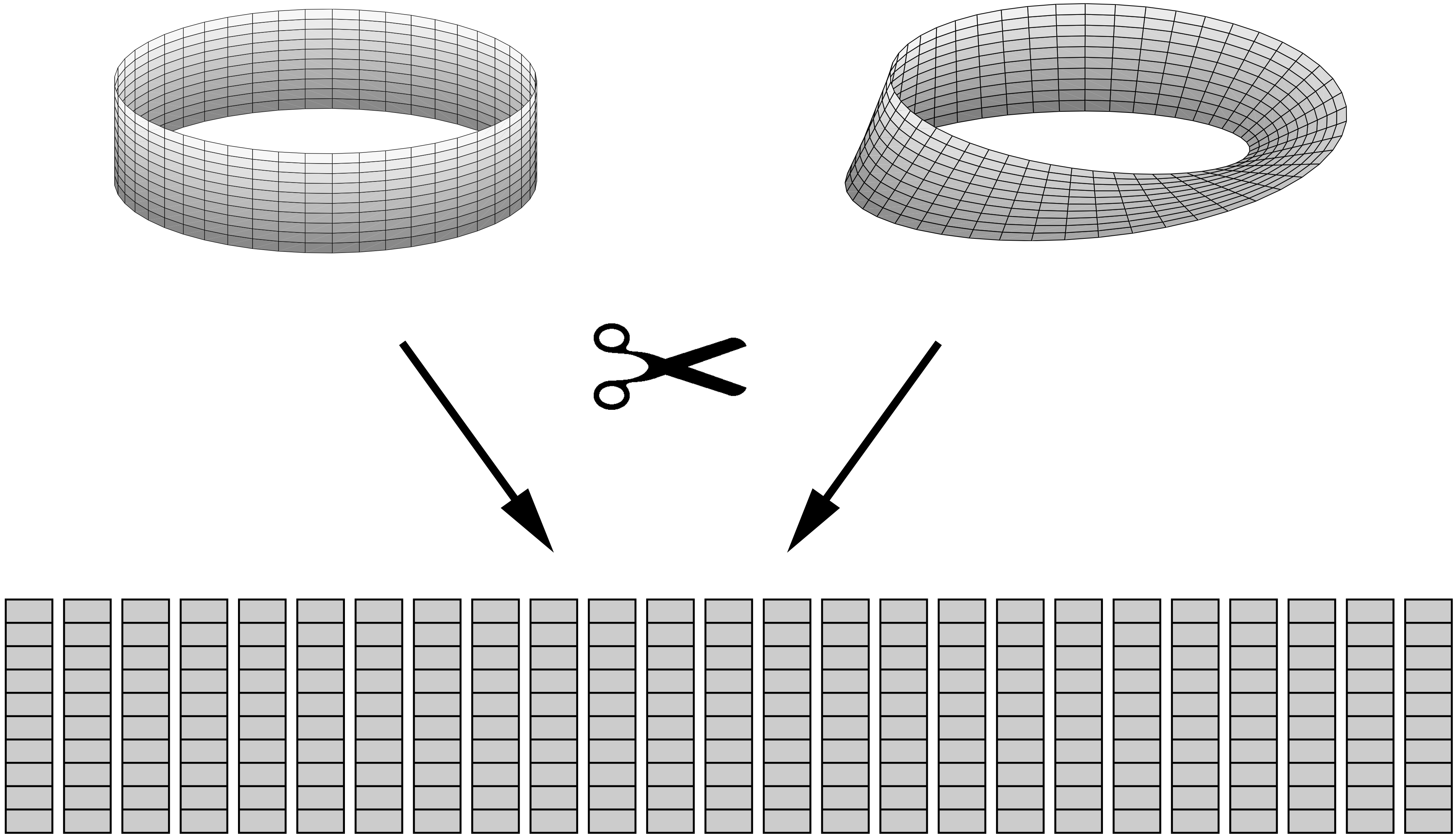


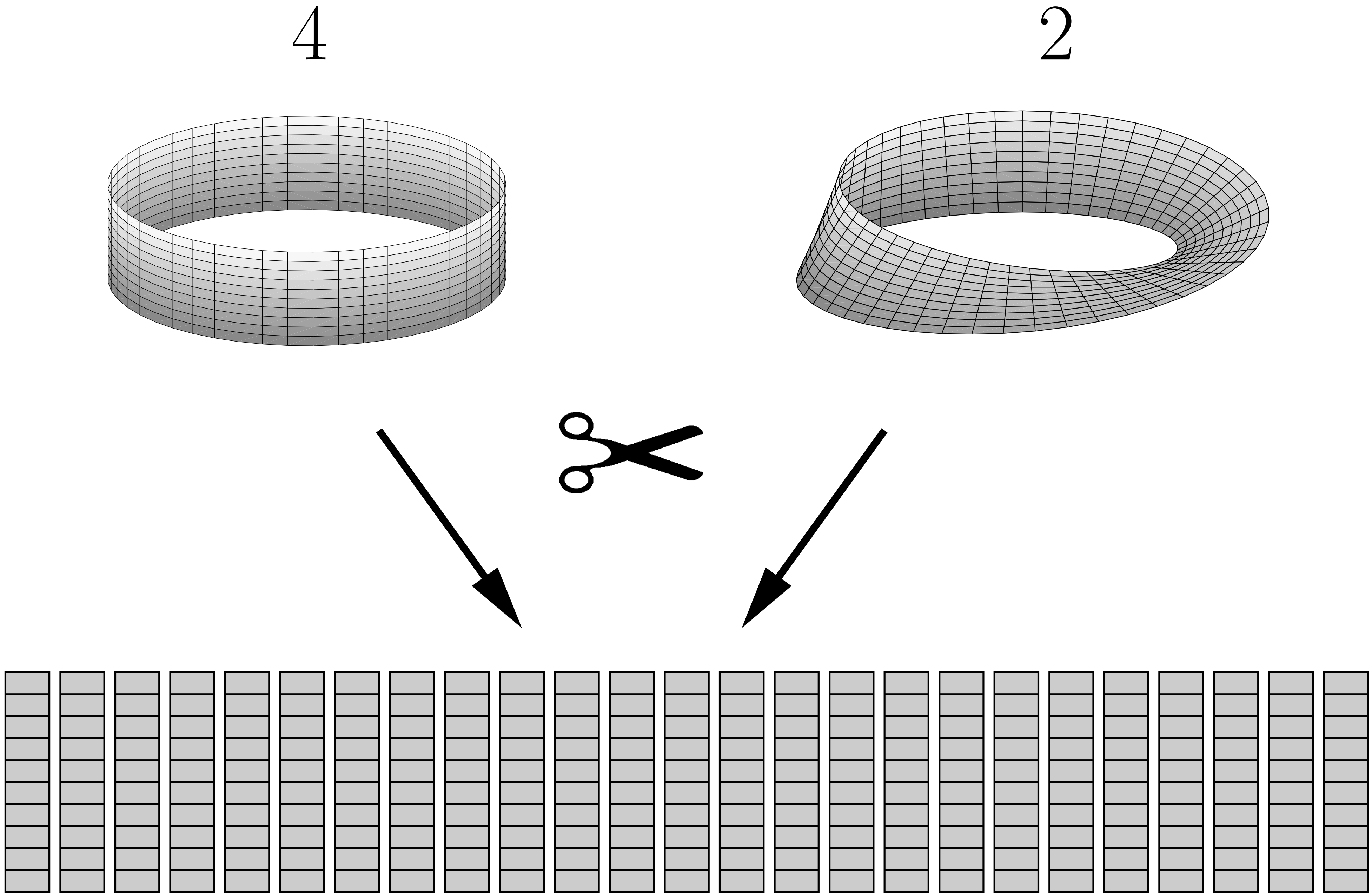




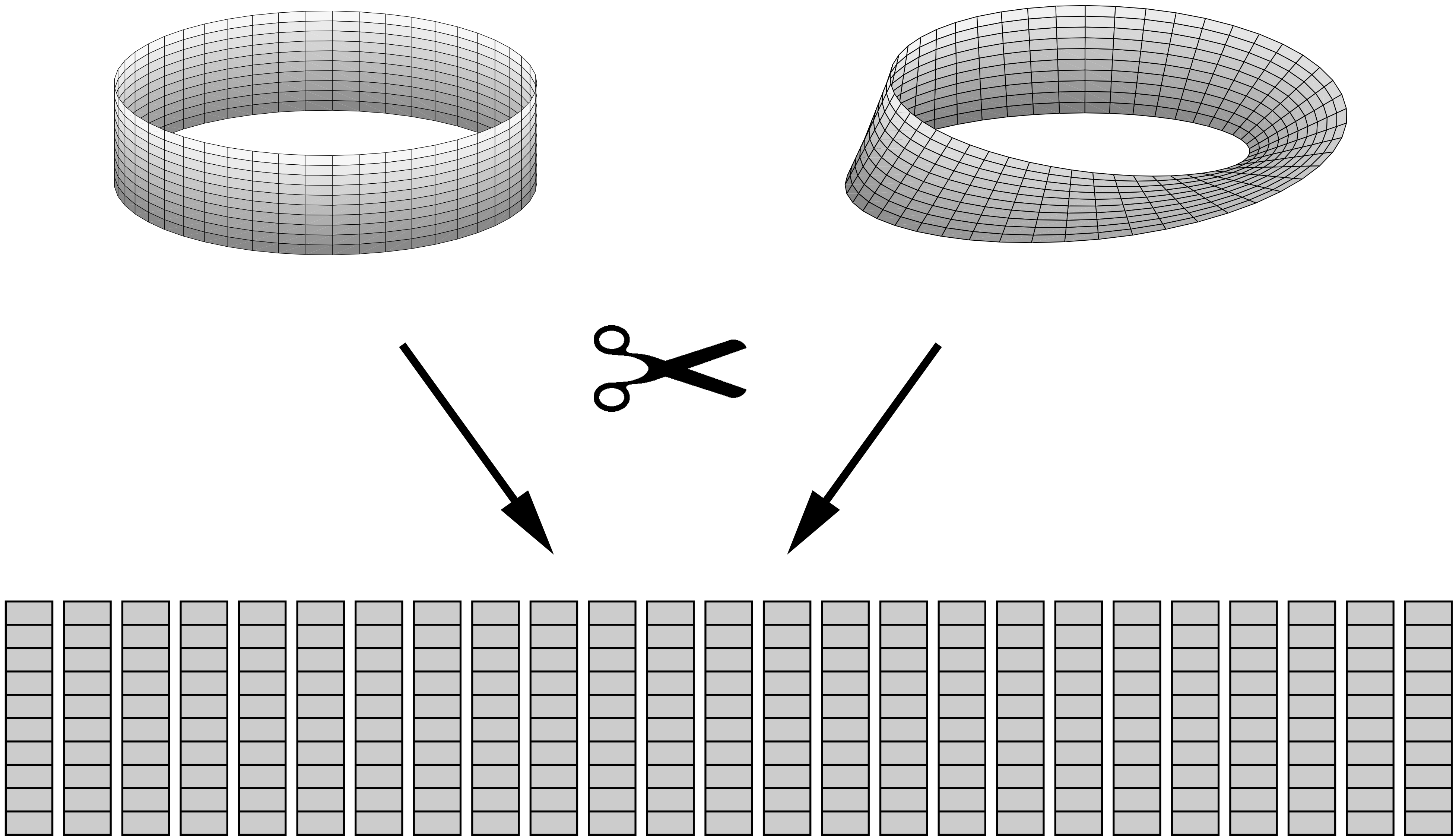


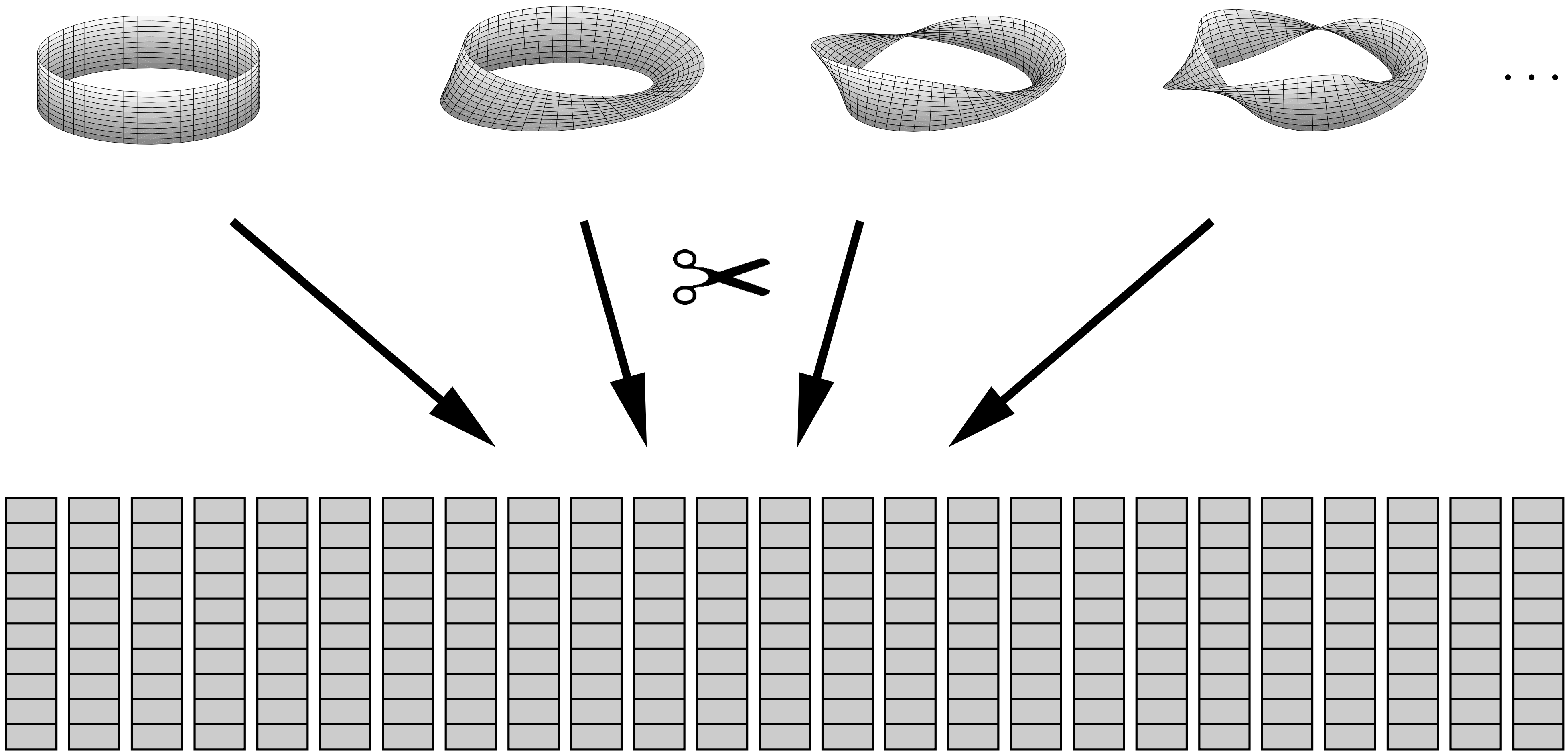
4

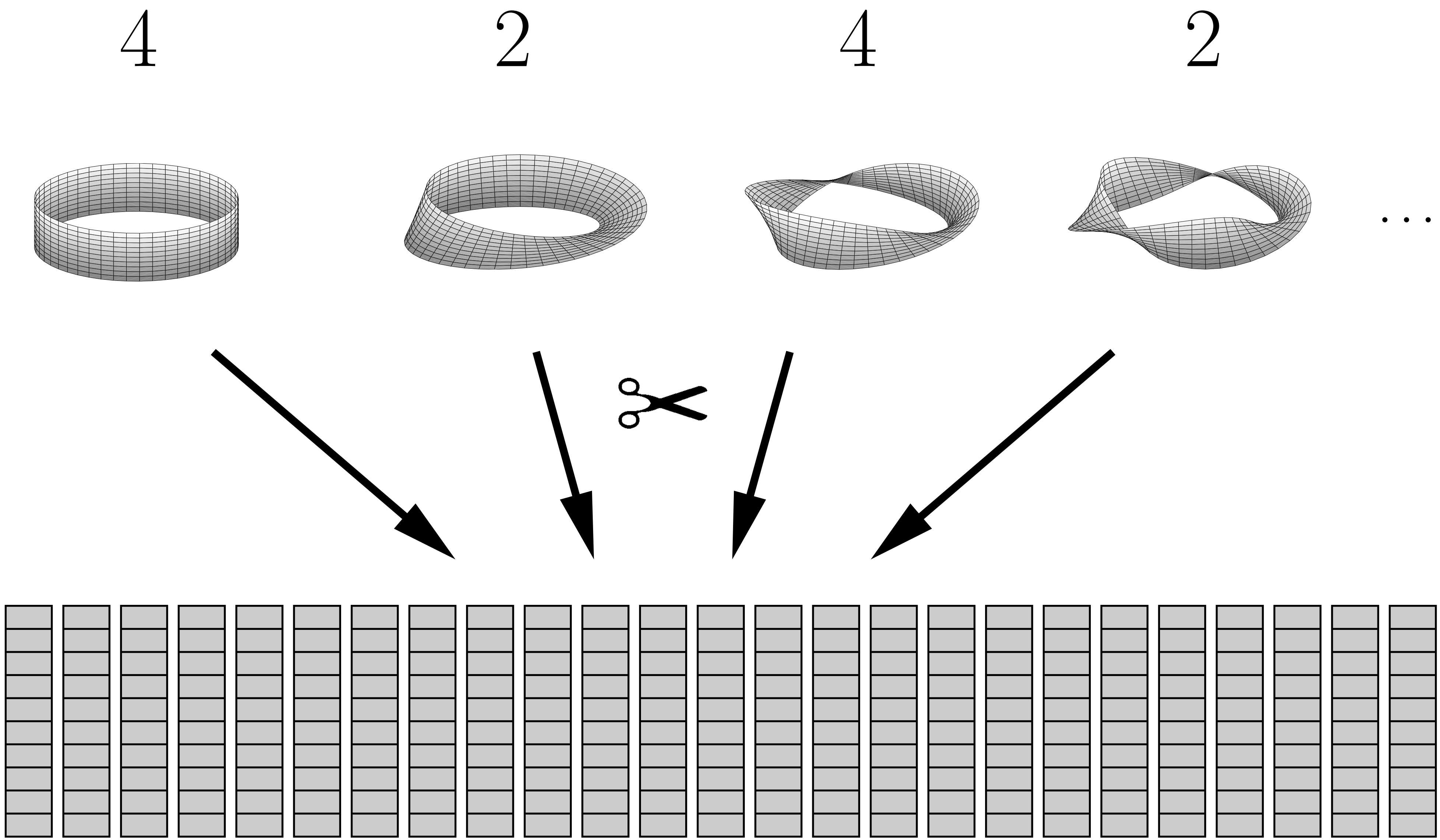



4
2

4 $\qquad \frac{4+2}{2} = 3 \qquad$ 2






4
2
4
2
...

The permanent of a matrix

Determinant vs. Permanent of a Matrix

Consider the matrix $\boldsymbol{\theta} = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{32} & \theta_{33} \end{pmatrix}$.

Determinant vs. Permanent of a Matrix

Consider the matrix $\boldsymbol{\theta} = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{32} & \theta_{33} \end{pmatrix}$.

The determinant of $\boldsymbol{\theta}$:

$$\begin{aligned} \det(\boldsymbol{\theta}) = &+\theta_{11}\theta_{22}\theta_{33} + \theta_{12}\theta_{23}\theta_{31} + \theta_{13}\theta_{21}\theta_{32} \\ &- \theta_{11}\theta_{23}\theta_{32} - \theta_{12}\theta_{21}\theta_{33} - \theta_{13}\theta_{22}\theta_{31}. \end{aligned}$$

Determinant vs. Permanent of a Matrix

Consider the matrix $\boldsymbol{\theta} = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{32} & \theta_{33} \end{pmatrix}$.

The determinant of $\boldsymbol{\theta}$:

$$\begin{aligned}\det(\boldsymbol{\theta}) = &+\theta_{11}\theta_{22}\theta_{33} + \theta_{12}\theta_{23}\theta_{31} + \theta_{13}\theta_{21}\theta_{32} \\ &- \theta_{11}\theta_{23}\theta_{32} - \theta_{12}\theta_{21}\theta_{33} - \theta_{13}\theta_{22}\theta_{31}.\end{aligned}$$

The permanent of $\boldsymbol{\theta}$:

$$\begin{aligned}\mathrm{perm}(\boldsymbol{\theta}) = &+\theta_{11}\theta_{22}\theta_{33} + \theta_{12}\theta_{23}\theta_{31} + \theta_{13}\theta_{21}\theta_{32} \\ &+ \theta_{11}\theta_{23}\theta_{32} + \theta_{12}\theta_{21}\theta_{33} + \theta_{13}\theta_{22}\theta_{31}.\end{aligned}$$

Determinant vs. Permanent of a Matrix

The determinant of an $n \times n$-matrix $\boldsymbol{\theta}$

$$\det(\boldsymbol{\theta}) = \sum_{\sigma} \operatorname{sgn}(\sigma) \prod_{i \in [n]} \theta_{i,\sigma(i)}.$$

where the sum is over all $n!$ permutations of the set $[n] \triangleq \{1, \ldots, n\}$.

Determinant vs. Permanent of a Matrix

The determinant of an $n \times n$-matrix $\boldsymbol{\theta}$

$$\det(\boldsymbol{\theta}) = \sum_{\sigma} \operatorname{sgn}(\sigma) \prod_{i \in [n]} \theta_{i,\sigma(i)}.$$

where the sum is over all $n!$ permutations of the set $[n] \triangleq \{1, \ldots, n\}$.

The permanent of an $n \times n$-matrix $\boldsymbol{\theta}$:

$$\operatorname{perm}(\boldsymbol{\theta}) = \sum_{\sigma} \prod_{i \in [n]} \theta_{i,\sigma(i)}.$$

Determinant vs. Permanent of a Matrix

The determinant of an $n \times n$-matrix $\boldsymbol{\theta}$

$$\det(\boldsymbol{\theta}) = \sum_{\sigma} \operatorname{sgn}(\sigma) \prod_{i \in [n]} \theta_{i,\sigma(i)}.$$

where the sum is over all $n!$ permutations of the set $[n] \triangleq \{1, \ldots, n\}$.

The permanent of an $n \times n$-matrix $\boldsymbol{\theta}$:

$$\operatorname{perm}(\boldsymbol{\theta}) = \sum_{\sigma} \prod_{i \in [n]} \theta_{i,\sigma(i)}.$$

The permanent turns up in a variety of context, especially in combinatorial problems, statistical physics (partition function), ...

Exactly Computing the Permanent

Exactly Computing the Permanent

- Brute-force computation:

$$O(n \cdot n!) = O\left(n^{3/2} \cdot (n/e)^n\right) \quad \text{arithmetic operations.}$$

Exactly Computing the Permanent

- Brute-force computation:

$$O(n \cdot n!) = O\big(n^{3/2} \cdot (n/e)^n\big) \quad \text{arithmetic operations.}$$

- Ryser's algorithm:

$$\Theta(n \cdot 2^n) \quad \text{arithmetic operations.}$$

Exactly Computing the Permanent

- Brute-force computation:

$$O(n \cdot n!) = O\big(n^{3/2} \cdot (n/e)^n\big) \quad \text{arithmetic operations.}$$

- Ryser's algorithm:

$$\Theta(n \cdot 2^n) \quad \text{arithmetic operations.}$$

- Complexity class [Valiant, 1979]:

#P ("sharp P" or "number P"),

where #P is the set of the counting problems associated with the decision problems in the set NP. (Note that even the computation of the permanent of zero-one matrices is #P-complete.)

Estimating the Permanent

More efficient algorithms are possible if one does not want to compute the permanent of a matrix exactly.

Estimating the Permanent

More efficient algorithms are possible if one does not want to compute the permanent of a matrix exactly.

- For a matrix that contains positive and negative entries:

  $\rightarrow$ "constructive and destructive interference of terms in the summation."

Estimating the Permanent

More efficient algorithms are possible if one does not want to compute the permanent of a matrix exactly.

- For a matrix that contains positive and negative entries:

  $\rightarrow$ "constructive and destructive interference of terms in the summation."

- For a matrix that contains only non-negative entries:

  $\rightarrow$ "constructive interference of terms in the summation."

Estimating the Permanent

FROM NOW ON: we focus on the case where all entries of the matrix are non-negative, i.e.

$$\theta_{ij} \geq 0 \quad \forall i, j.$$

Estimating the Permanent

FROM NOW ON: we focus on the case where all entries of the matrix are non-negative, i.e.

$$\theta_{ij} \geq 0 \quad \forall i, j.$$

- Markov chain Monte Carlo based methods: [Broder, 1986], ...

Estimating the Permanent

FROM NOW ON: we focus on the case where all entries of the matrix are non-negative, i.e.

$$\theta_{ij} \geq 0 \quad \forall i, j.$$

- Markov chain Monte Carlo based methods: [Broder, 1986], ...
- Godsil-Gutman formula based methods: [Karmarkar et al., 1993], [Barvinok, 1997ff.], [Chien, Rasmussen, Sinclair, 2004], ...

Estimating the Permanent

FROM NOW ON: we focus on the case where all entries of the matrix are non-negative, i.e.

$$\theta_{ij} \geq 0 \quad \forall i, j.$$

- Markov chain Monte Carlo based methods: [Broder, 1986], ...
- Godsil-Gutman formula based methods: [Karmarkar et al., 1993], [Barvinok, 1997ff.], [Chien, Rasmussen, Sinclair, 2004], ...
- Fully polynomial-time randomized approximation schemes (FPRAS): [Jerrum, Sinclair, Vigoda, 2004], ...

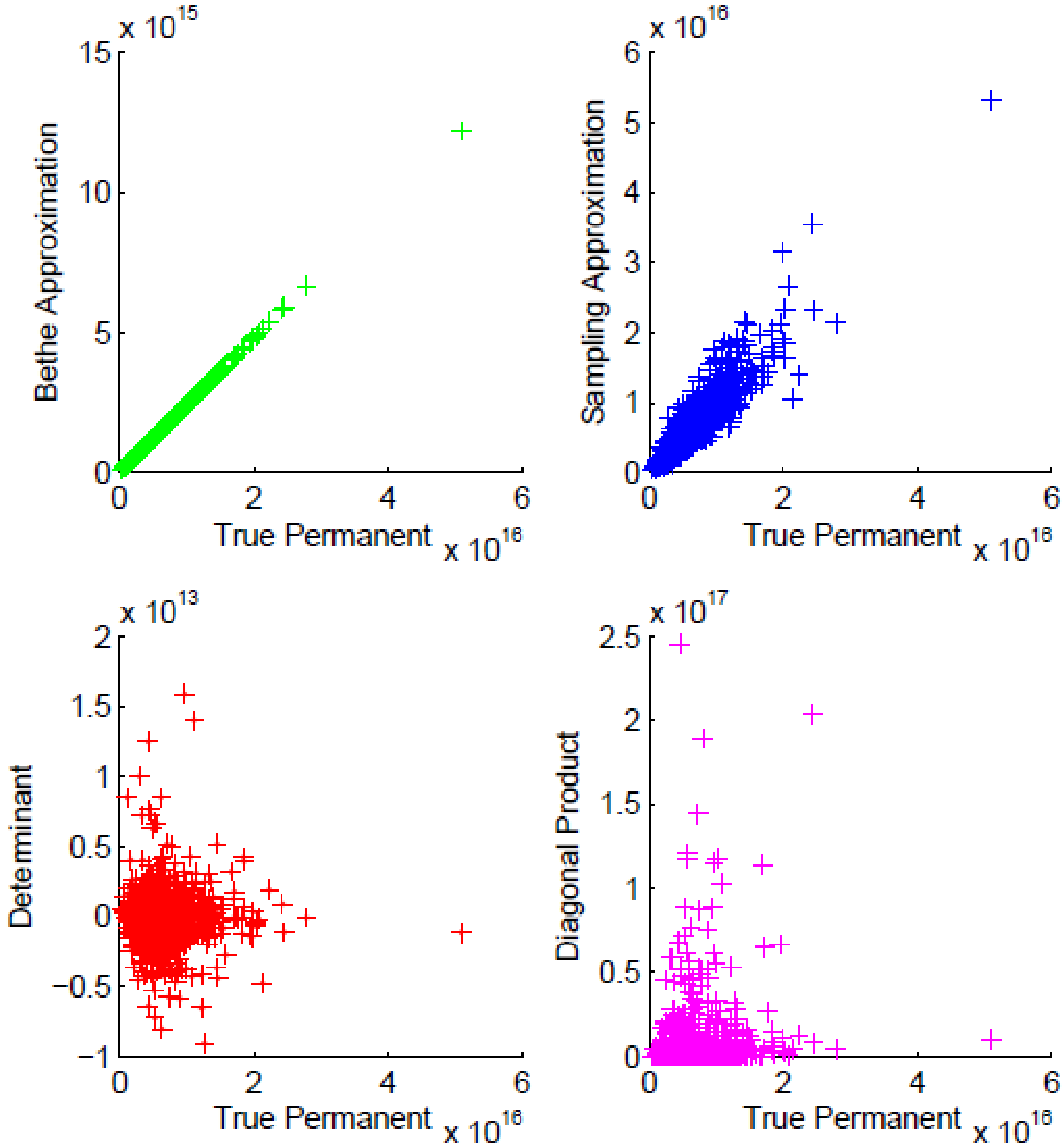
Estimating the Permanent

FROM NOW ON: we focus on the case where all entries of the matrix are non-negative, i.e.

$$\theta_{ij} \geq 0 \quad \forall i, j.$$

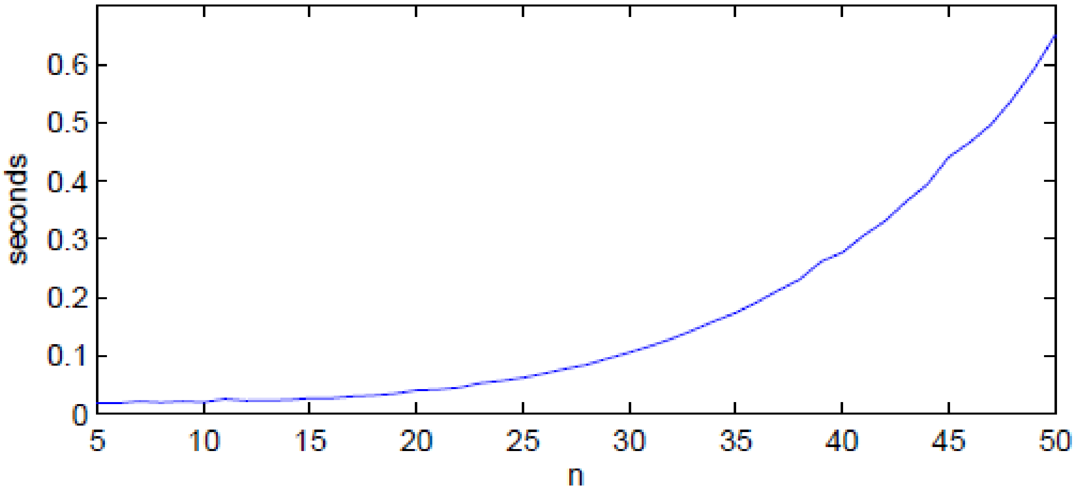
- Markov chain Monte Carlo based methods: [Broder, 1986], ...
- Godsil-Gutman formula based methods: [Karmarkar et al., 1993], [Barvinok, 1997ff.], [Chien, Rasmussen, Sinclair, 2004], ...
- Fully polynomial-time randomized approximation schemes (FPRAS): [Jerrum, Sinclair, Vigoda, 2004], ...
- Bethe-approximation-based / sum-product-algorithm-based methods: [Chertkov et al., 2008], [Huang and Jebara, 2009], ...

Estimating the Permanent of a Matrix

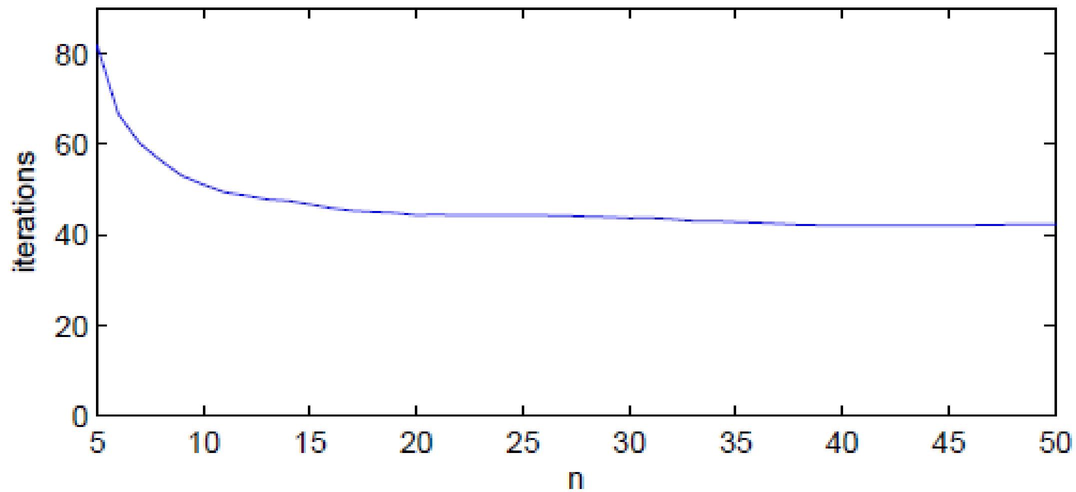


From [Huang/Jebara, 2009].

Estimating the Permanent of a Matrix



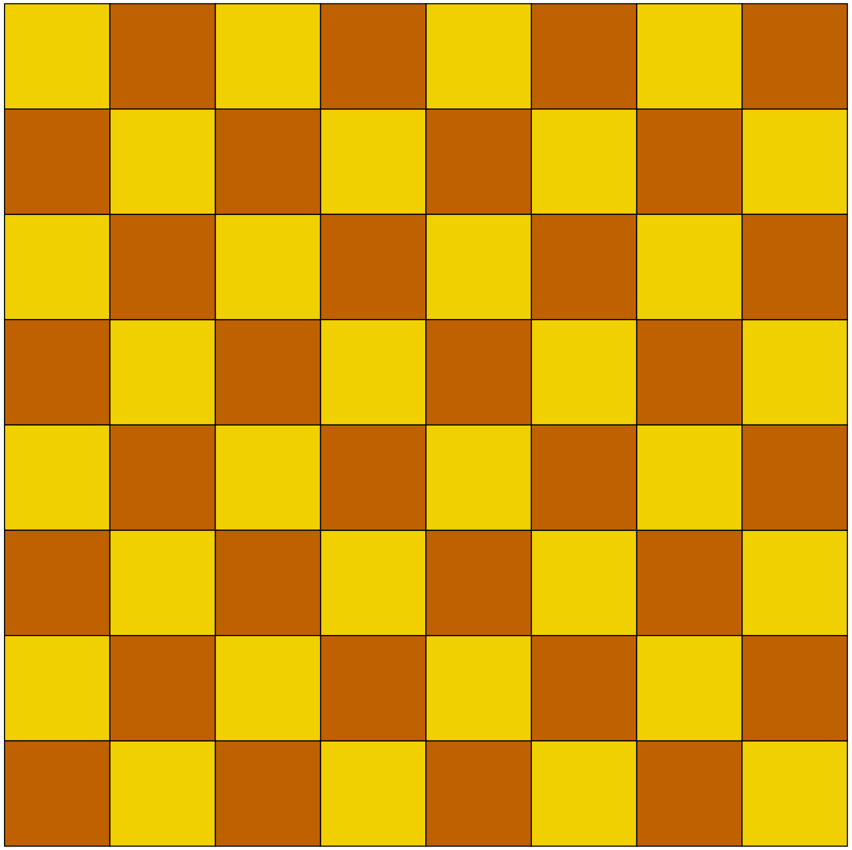
(a) Running time



(b) Iterations

From [Huang/Jebara, 2009].

Valid Rook Configurations and Permanents

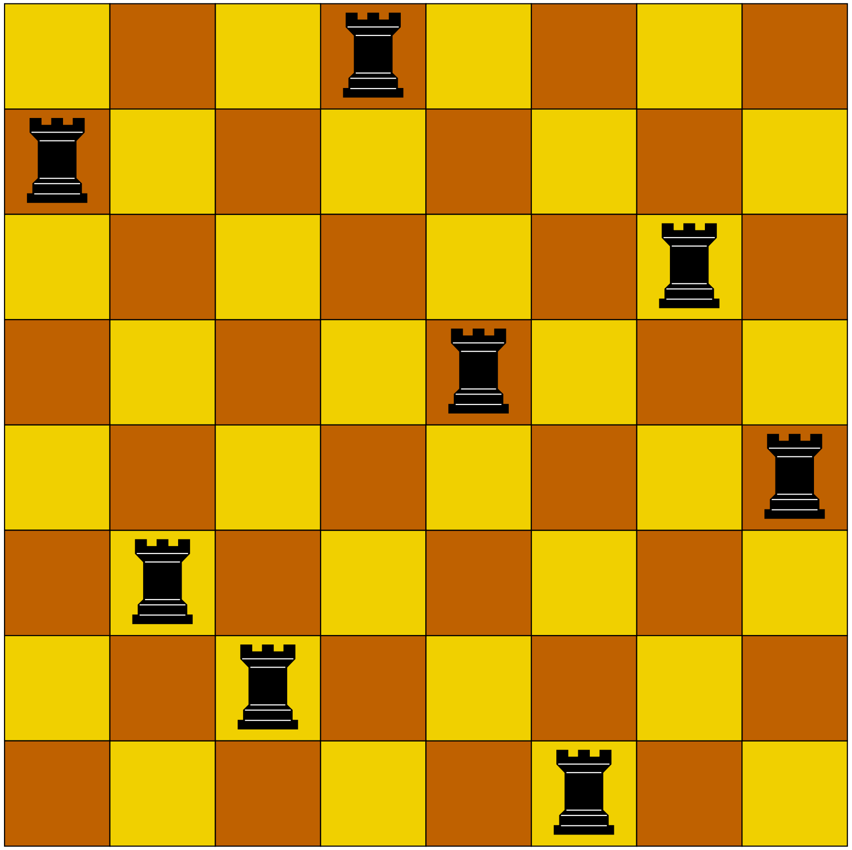


Number of valid rook configurations

$$= \operatorname{perm}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Valid Rook Configurations and Permanents

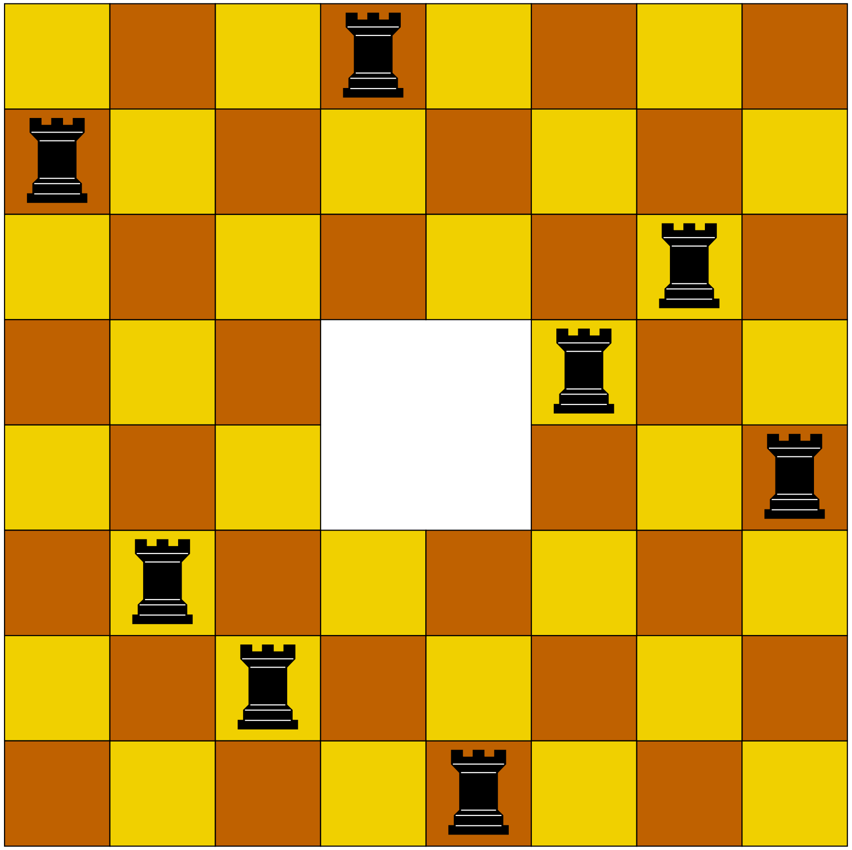


Number of valid rook configurations

$$= \operatorname{perm}\begin{pmatrix} 1 & 1 & 1 & {\color{red}1} & 1 & 1 & 1 & 1 \\ {\color{red}1} & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & {\color{red}1} & 1 \\ 1 & 1 & 1 & 1 & {\color{red}1} & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & {\color{red}1} \\ 1 & {\color{red}1} & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & {\color{red}1} & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & {\color{red}1} & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} {\color{red}\prod_{i \in [8]} \theta_{i,\sigma(i)}}$$

Valid Rook Configurations and Permanents

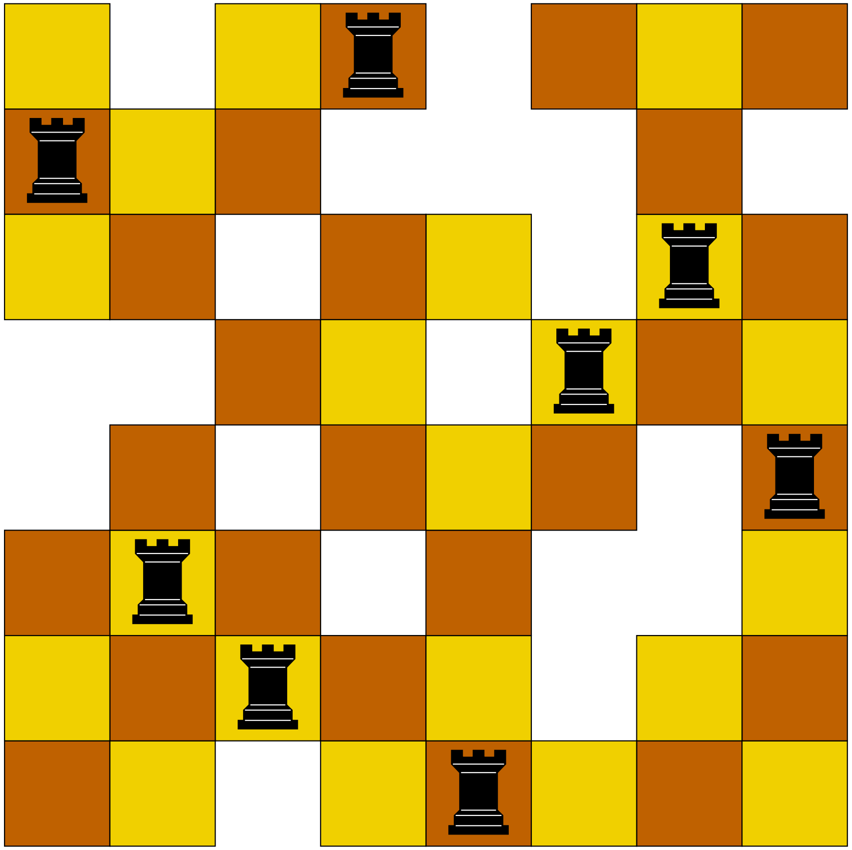


Number of valid rook configurations

$$= \mathrm{perm}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Valid Rook Configurations and Permanents

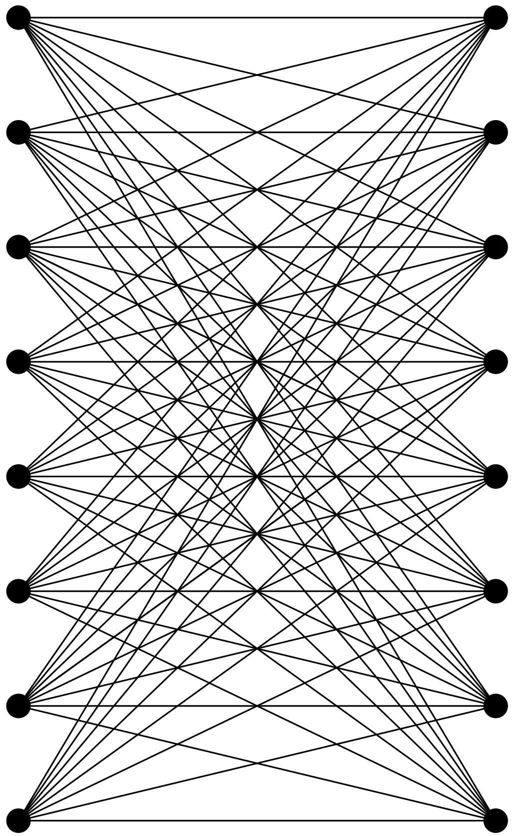


Number of valid rook configurations

$$= \operatorname{perm} \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Perfect Matchings and Permanents

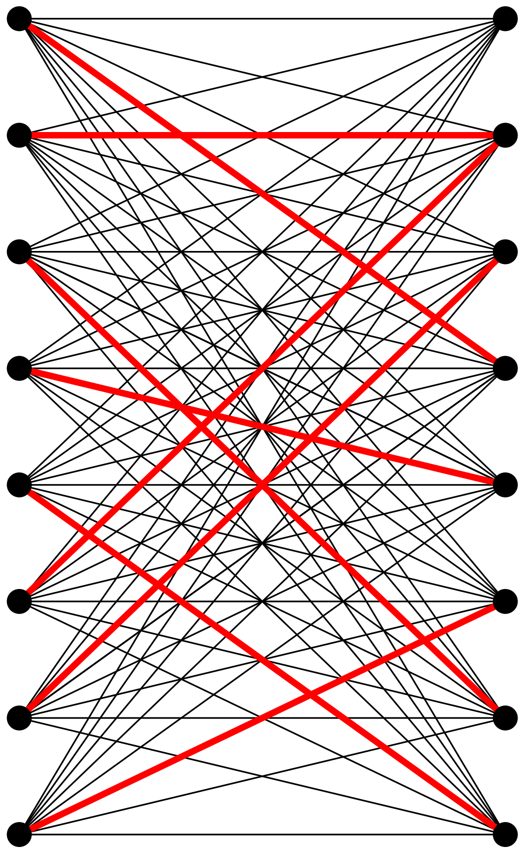


Number of valid perfect matchings

$$= \mathrm{perm}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Perfect Matchings and Permanents

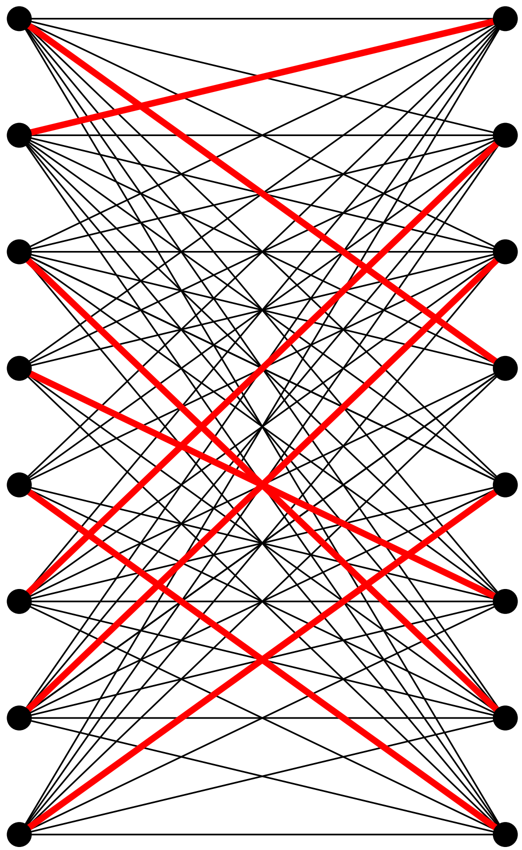


Number of valid perfect matchings

$$= \operatorname{perm} \begin{pmatrix} 1 & 1 & 1 & \mathbf{1} & 1 & 1 & 1 & 1 \\ \mathbf{1} & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 & 1 & \mathbf{1} & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & \mathbf{1} \\ 1 & \mathbf{1} & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & \mathbf{1} & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & \mathbf{1} & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Perfect Matchings and Permanents

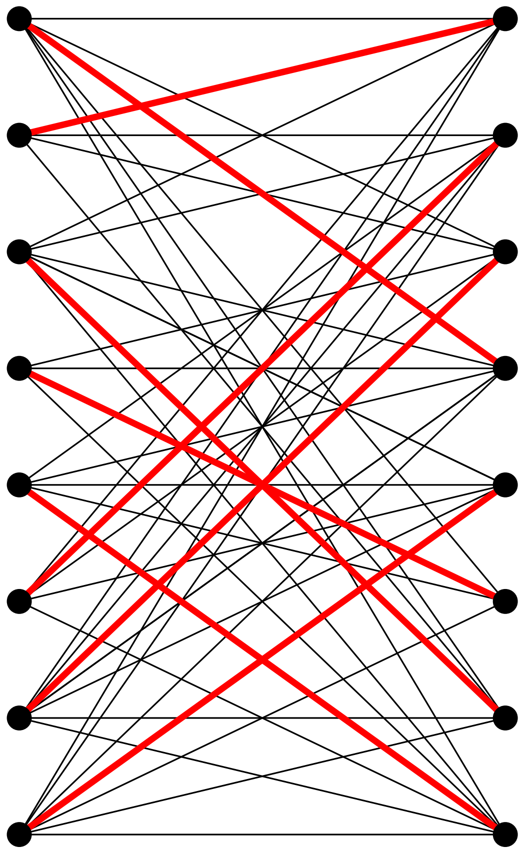


Number of valid perfect matchings

$$= \operatorname{perm}\begin{pmatrix} 1 & 1 & 1 & \color{red}{\mathbf{1}} & 1 & 1 & 1 & 1 \\ \color{red}{\mathbf{1}} & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & \color{red}{\mathbf{1}} & 1 \\ 1 & 1 & 1 & 0 & 0 & \color{red}{\mathbf{1}} & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & \color{red}{\mathbf{1}} \\ 1 & \color{red}{\mathbf{1}} & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & \color{red}{\mathbf{1}} & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & \color{red}{\mathbf{1}} & 1 & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Perfect Matchings and Permanents

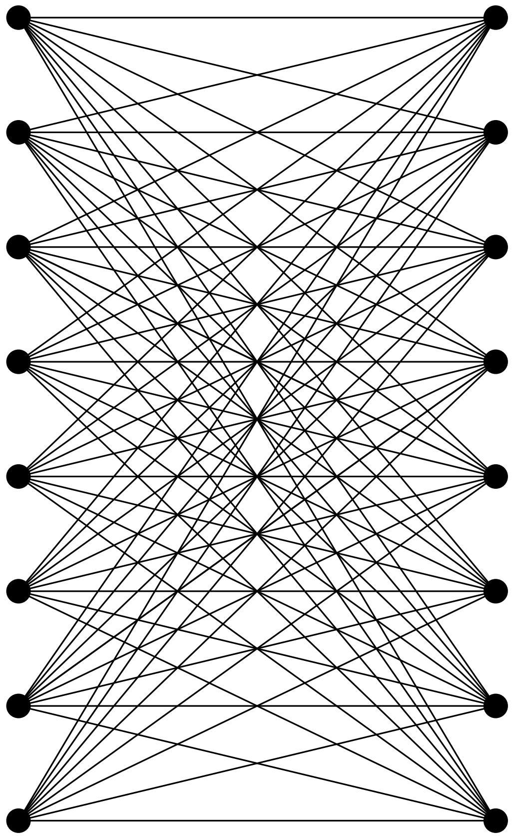


Number of valid perfect matchings

$$= \operatorname{perm}\begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Perfect Matchings and Permanents

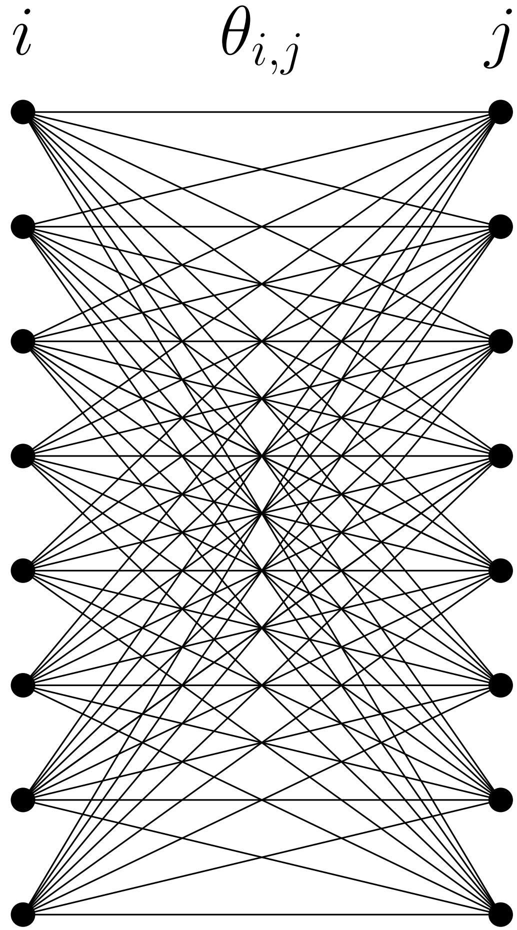


Number of valid perfect matchings

$$= \operatorname{perm}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Perfect Matchings and Permanents

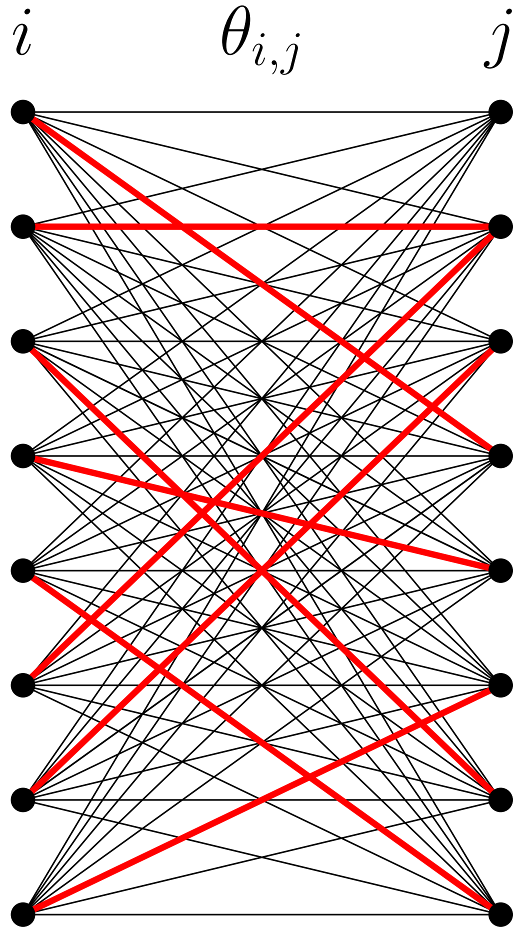


Total sum of weighted perf. matchings

$$= \operatorname{perm}\begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} & \theta_{14} & \theta_{15} & \theta_{16} & \theta_{17} & \theta_{18} \\ \theta_{21} & \theta_{22} & \theta_{23} & \theta_{24} & \theta_{25} & \theta_{26} & \theta_{27} & \theta_{28} \\ \theta_{31} & \theta_{32} & \theta_{33} & \theta_{34} & \theta_{35} & \theta_{36} & \theta_{37} & \theta_{38} \\ \theta_{41} & \theta_{42} & \theta_{43} & \theta_{44} & \theta_{45} & \theta_{46} & \theta_{47} & \theta_{48} \\ \theta_{51} & \theta_{52} & \theta_{53} & \theta_{54} & \theta_{55} & \theta_{56} & \theta_{57} & \theta_{58} \\ \theta_{61} & \theta_{62} & \theta_{63} & \theta_{64} & \theta_{65} & \theta_{66} & \theta_{67} & \theta_{68} \\ \theta_{71} & \theta_{72} & \theta_{73} & \theta_{74} & \theta_{75} & \theta_{76} & \theta_{77} & \theta_{78} \\ \theta_{81} & \theta_{82} & \theta_{83} & \theta_{84} & \theta_{85} & \theta_{86} & \theta_{87} & \theta_{88} \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Perfect Matchings and Permanents

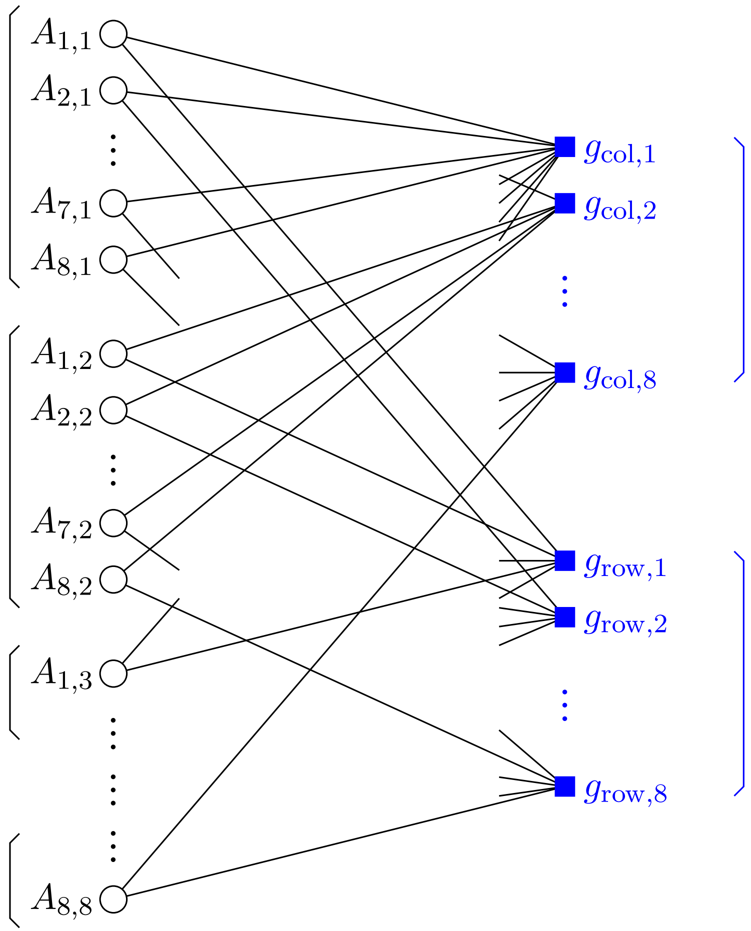


Total sum of weighted perf. matchings

$$= \operatorname{perm}\begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} & \theta_{14} & \theta_{15} & \theta_{16} & \theta_{17} & \theta_{18} \\ \theta_{21} & \theta_{22} & \theta_{23} & \theta_{24} & \theta_{25} & \theta_{26} & \theta_{27} & \theta_{28} \\ \theta_{31} & \theta_{32} & \theta_{33} & \theta_{34} & \theta_{35} & \theta_{36} & \theta_{37} & \theta_{38} \\ \theta_{41} & \theta_{42} & \theta_{43} & \theta_{44} & \theta_{45} & \theta_{46} & \theta_{47} & \theta_{48} \\ \theta_{51} & \theta_{52} & \theta_{53} & \theta_{54} & \theta_{55} & \theta_{56} & \theta_{57} & \theta_{58} \\ \theta_{61} & \theta_{62} & \theta_{63} & \theta_{64} & \theta_{65} & \theta_{66} & \theta_{67} & \theta_{68} \\ \theta_{71} & \theta_{72} & \theta_{73} & \theta_{74} & \theta_{75} & \theta_{76} & \theta_{77} & \theta_{78} \\ \theta_{81} & \theta_{82} & \theta_{83} & \theta_{84} & \theta_{85} & \theta_{86} & \theta_{87} & \theta_{88} \end{pmatrix}$$

$$= \sum_{\sigma} \prod_{i \in [8]} \theta_{i,\sigma(i)}$$

Graphical Model for Permanent



(function nodes are suitably defined based on $\boldsymbol{\theta}$)

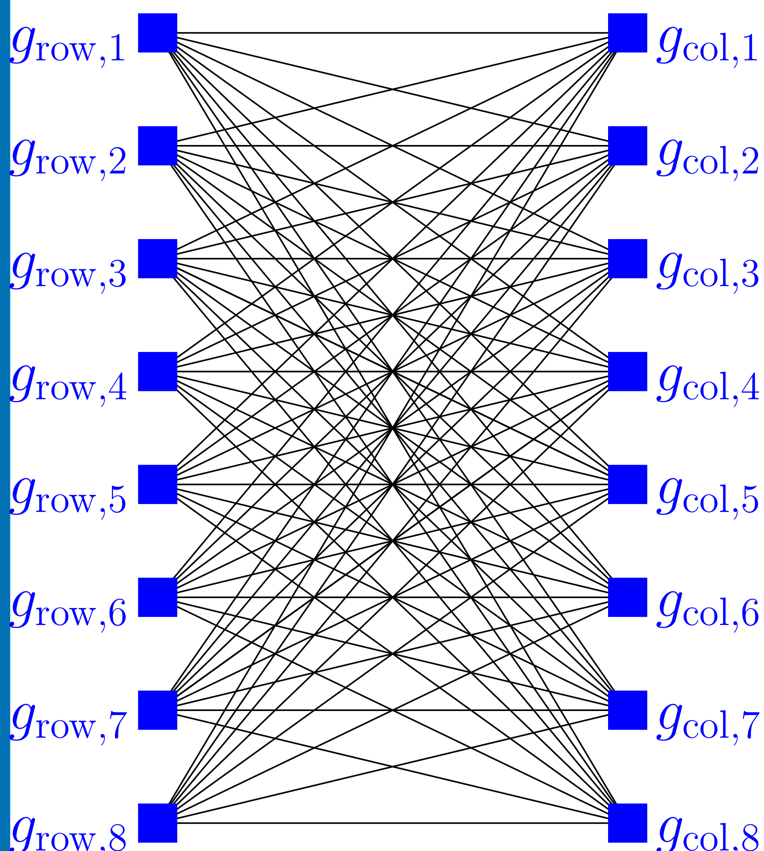
Global function:

$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\text{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\text{row},i}(a_{i,1}, \ldots, a_{i,8})$$

Permanent:

$$\text{perm}(\boldsymbol{\theta}) = Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1}, \ldots, a_{8,8})$$

Graphical Model for Permanent



(function nodes are suitably defined based on $\boldsymbol{\theta}$)

(variable nodes have been omitted)

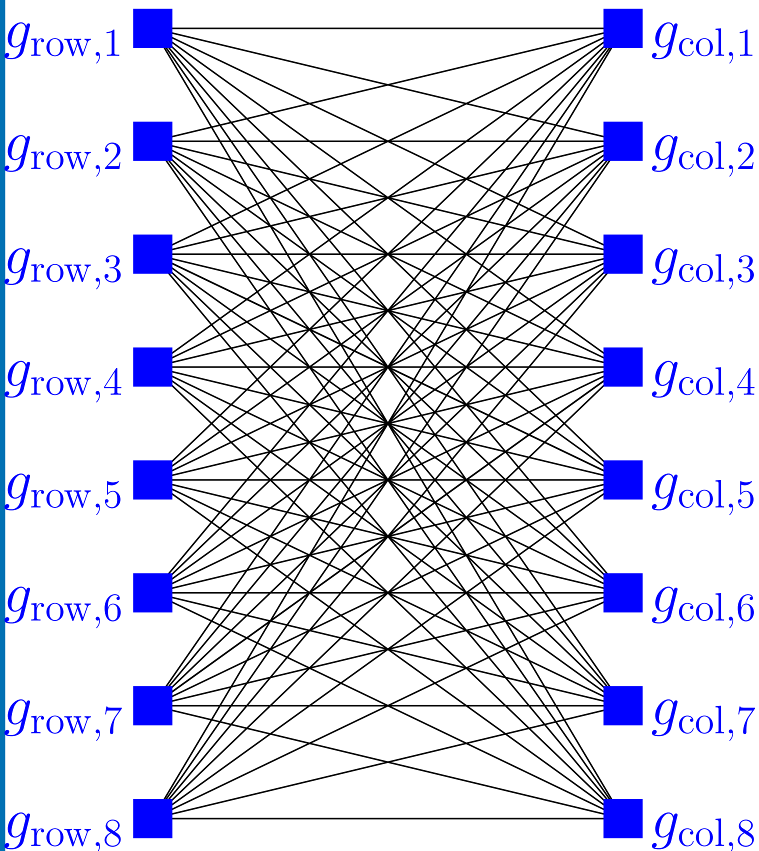
Global function:

$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\text{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\text{row},i}(a_{i,1}, \ldots, a_{i,8})$$

Permanent:

$$\text{perm}(\boldsymbol{\theta}) = Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1}, \ldots, a_{8,8})$$

Graphical Model for Permanent

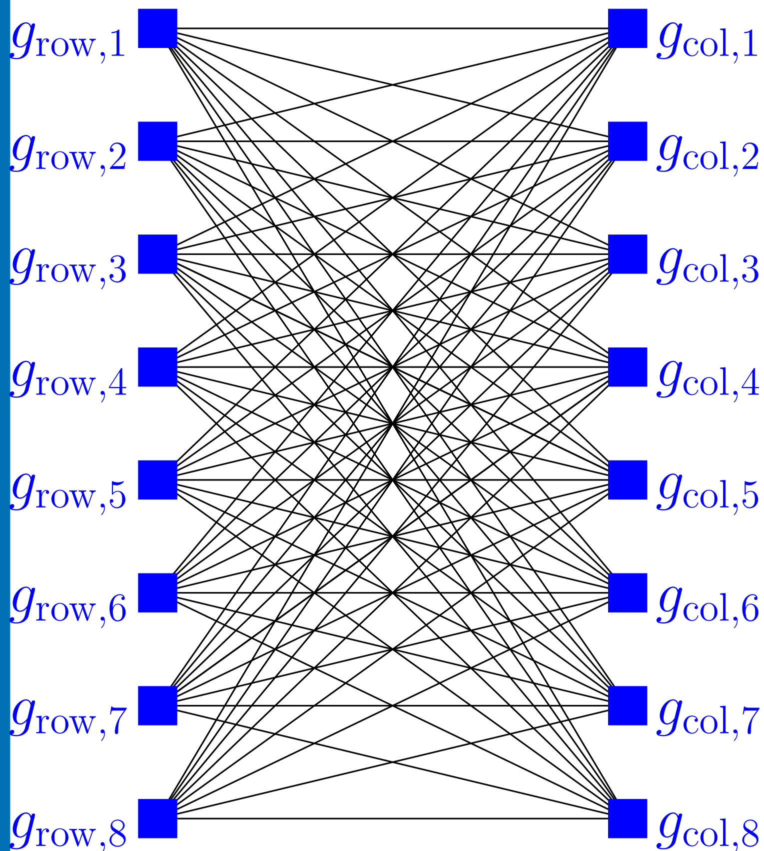


(function nodes are suitably defined based on $\boldsymbol{\theta}$)

(variable nodes have been omitted)

- Many short cycles.
- The vertex degrees are high.

Graphical Model for Permanent



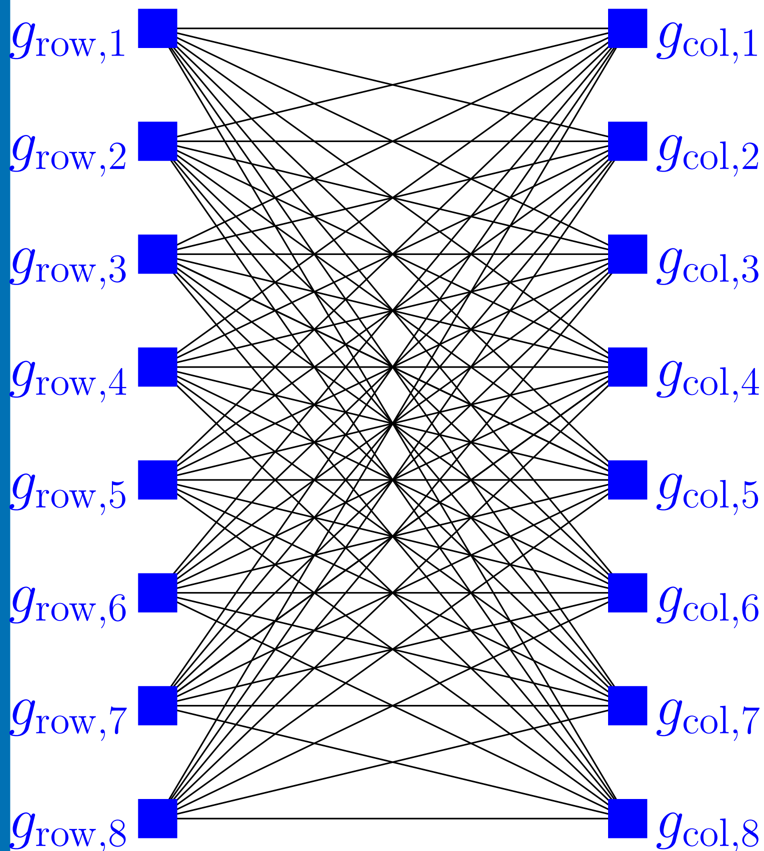
(function nodes are suitably defined based on $\boldsymbol{\theta}$)

(variable nodes have been omitted)

- ↯ Many short cycles.
- ↯ The vertex degrees are high.

Both facts might suggest that the application of the sum-product algorithm to this factor graph is rather problematic.

Graphical Model for Permanent



(function nodes are suitably defined based on $\boldsymbol{\theta}$)

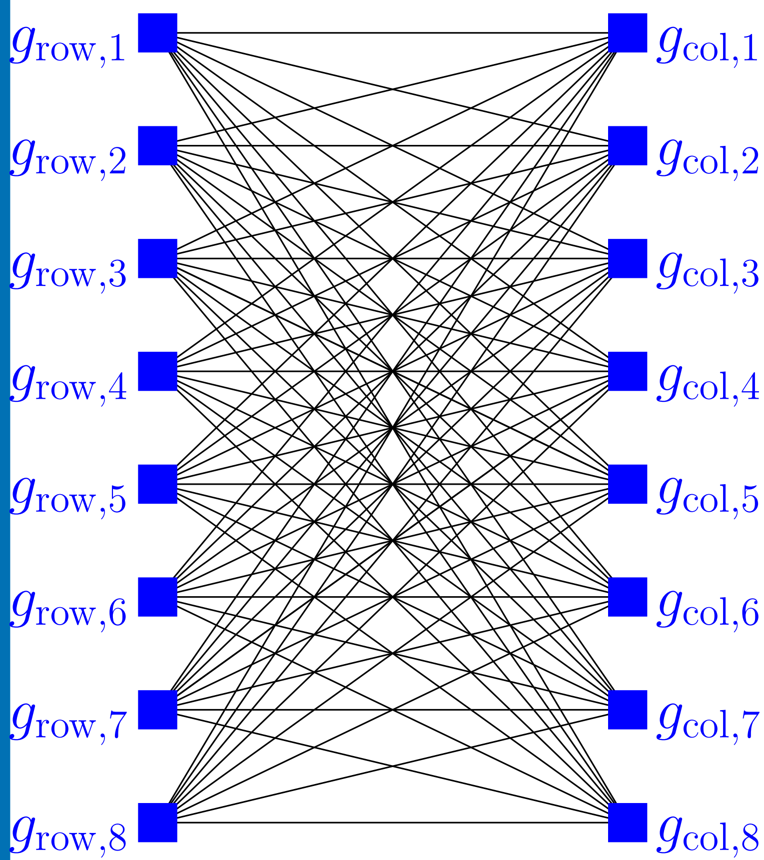
(variable nodes have been omitted)

- ↯ Many short cycles.
- ↯ The vertex degrees are high.

Both facts might suggest that the application of the sum-product algorithm to this factor graph is rather problematic.

However, luckily this is not the case.

Graphical Model for Permanent



(function nodes are suitably defined based on $\boldsymbol{\theta}$)

(variable nodes have been omitted)

- ↯ Many short cycles.
- ↯ The vertex degrees are high.

Both facts might suggest that the application of the sum-product algorithm to this factor graph is rather problematic.

However, luckily this is not the case.

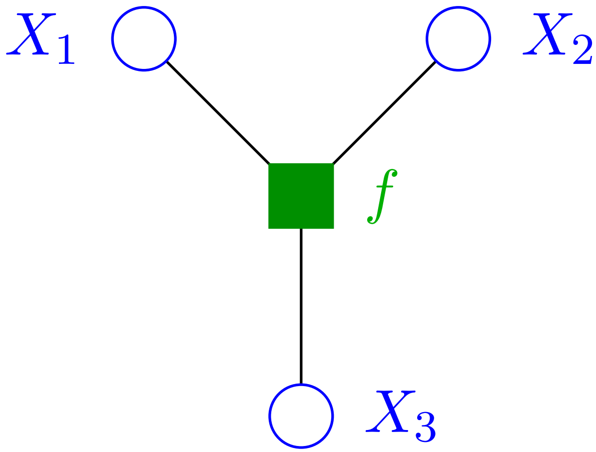
For an SPA suitability assessment, the overall cycle structure and the types of functions nodes are at least as important.

Factor graphs and the sum-product algorithm

Factor Graphs

A factor graph can be used to represent a multivariate function:

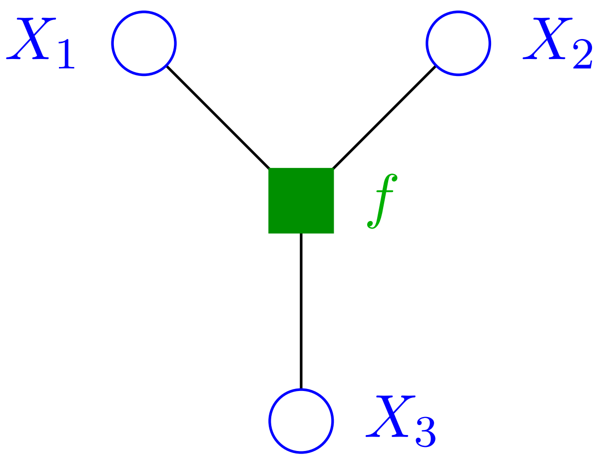
$$f(x_1, x_2, x_3)$$



Factor Graphs

A factor graph can be used to represent a multivariate function:

$$f(x_1, x_2, x_3)$$

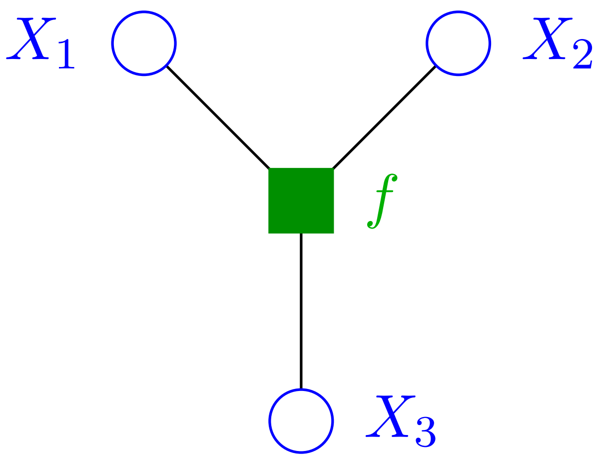


- Variable nodes: for each variable we draw a variable node (empty circles).

Factor Graphs

A factor graph can be used to represent a multivariate function:

$$f(x_1, x_2, x_3)$$

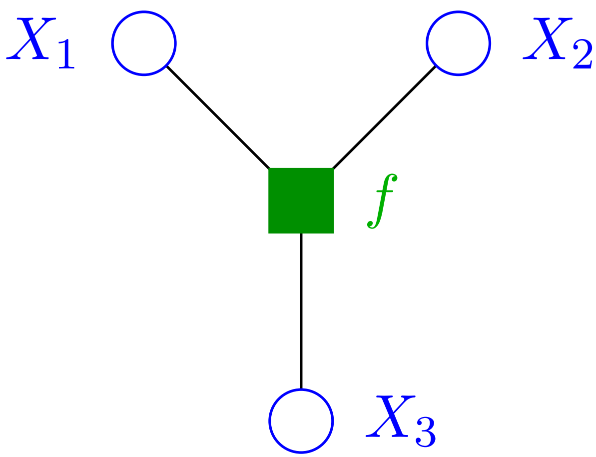


- Variable nodes: for each variable we draw a variable node (empty circles).
- Function nodes: for each function we draw a function node (filled squares).

Factor Graphs

A factor graph can be used to represent a multivariate function:

$$f(x_1, x_2, x_3)$$

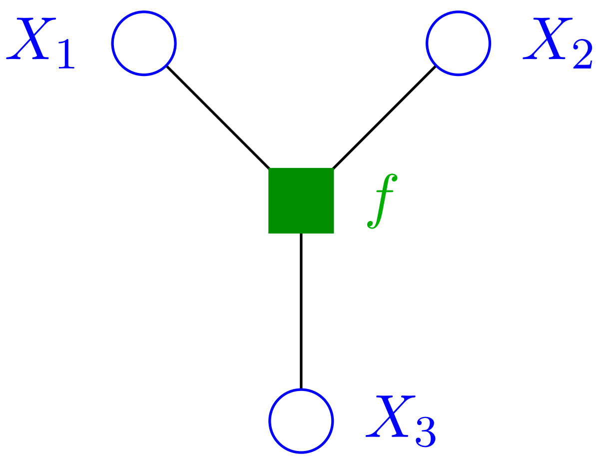


- Variable nodes: for each variable we draw a variable node (empty circles).
- Function nodes: for each function we draw a function node (filled squares).
- Edges: there is an edge between a variable node and a function node if the corresponding variable is an argument of the corresponding function.

Factor Graphs

A factor graph can be used to represent a multivariate function:

$$f(x_1, x_2, x_3)$$



- Variable nodes: for each variable we draw a variable node (empty circles).
- Function nodes: for each function we draw a function node (filled squares).
- Edges: there is an edge between a variable node and a function node if the corresponding variable is an argument of the corresponding function.
- Bipartite graph: the resulting graph is a bipartite graph, i.e. there are only edges between vertices of different types.

Factor Graphs

General references for factor graphs are:

- F. R. Kschischang, B. J. Frey and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," IEEE Trans. on Inform. Theory, IT–47, Feb. 2001.
- H.-A. Loeliger, "An introduction to factor graphs," IEEE Signal Processing Magazine, Jan. 2004.

Factor Graphs

We assume that we know more about the internal structure of the function $f(x_1, x_2, x_3)$, e.g.

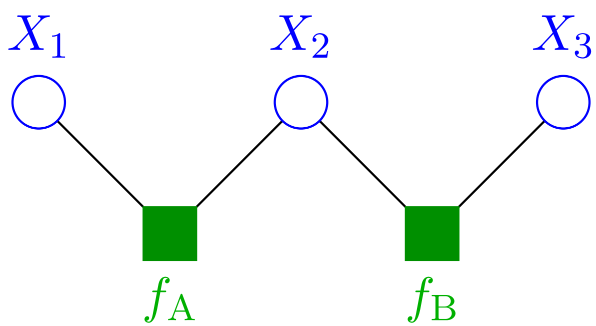
$$f(x_1, x_2, x_3) = f_{\mathrm{A}}(x_1, x_2) \cdot f_{\mathrm{B}}(x_2, x_3).$$

Factor Graphs

We assume that we know more about the internal structure of the function $f(x_1, x_2, x_3)$, e.g.

$$f(x_1, x_2, x_3) = f_\mathrm{A}(x_1, x_2) \cdot f_\mathrm{B}(x_2, x_3).$$

Then we can take advantage of this fact and the factor graph represents this structure.

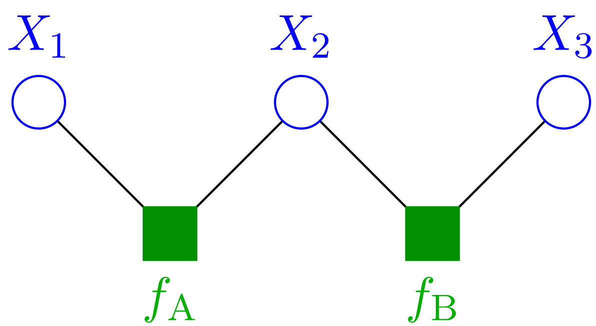


Factor Graphs

We assume that we know more about the internal structure of the function $f(x_1, x_2, x_3)$, e.g.

$$f(x_1, x_2, x_3) = f_{\mathrm{A}}(x_1, x_2) \cdot f_{\mathrm{B}}(x_2, x_3).$$

Then we can take advantage of this fact and the factor graph represents this structure.



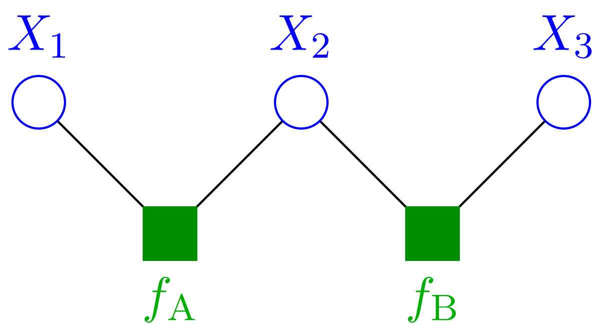
- $f(.,.,.)$ is called the global function.

Factor Graphs

We assume that we know more about the internal structure of the function $f(x_1, x_2, x_3)$, e.g.

$$f(x_1, x_2, x_3) = f_{\mathrm{A}}(x_1, x_2) \cdot f_{\mathrm{B}}(x_2, x_3).$$

Then we can take advantage of this fact and the factor graph represents this structure.



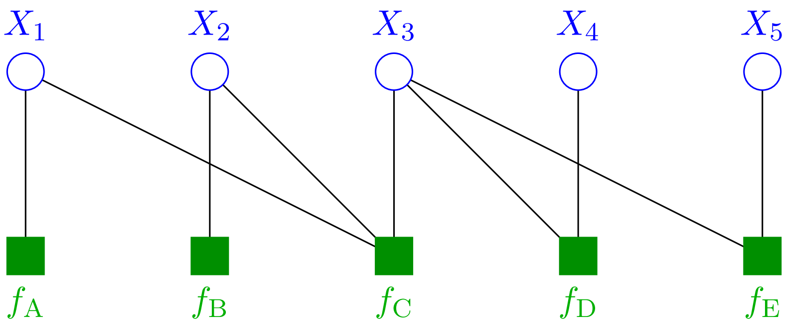
- $f(.,.,.)$ is called the global function.
- $f_{\mathrm{A}}(.,.)$ and $f_{\mathrm{B}}(.,.)$ are called local functions.

Factor Graphs

$$f(x_1, x_2, x_3, x_4, x_5)$$
$$= f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5)$$

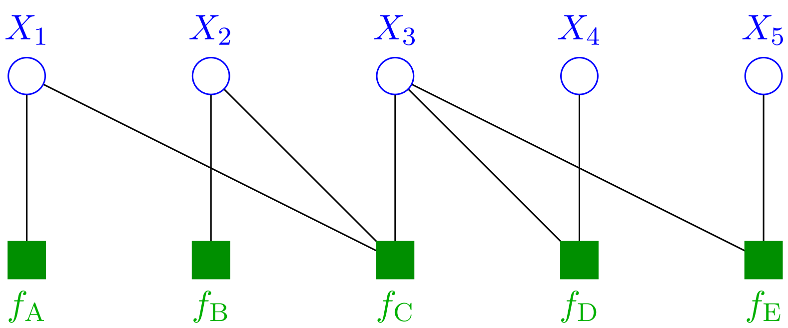
Factor Graphs

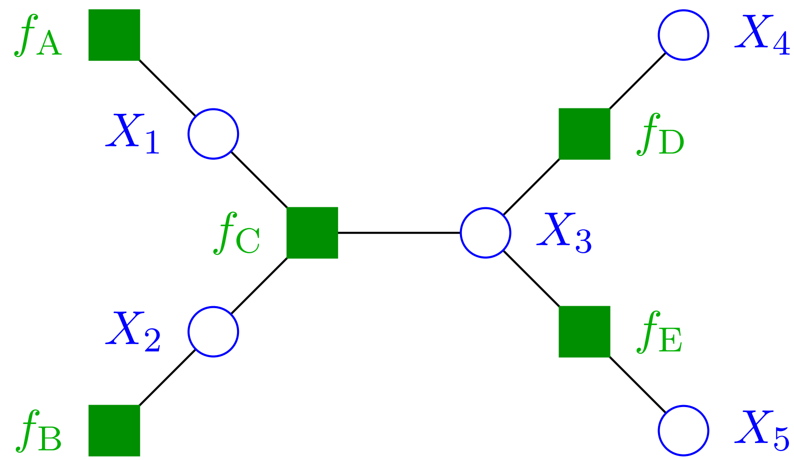
$$f(x_1, x_2, x_3, x_4, x_5)$$
$$= f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5)$$



Factor Graphs

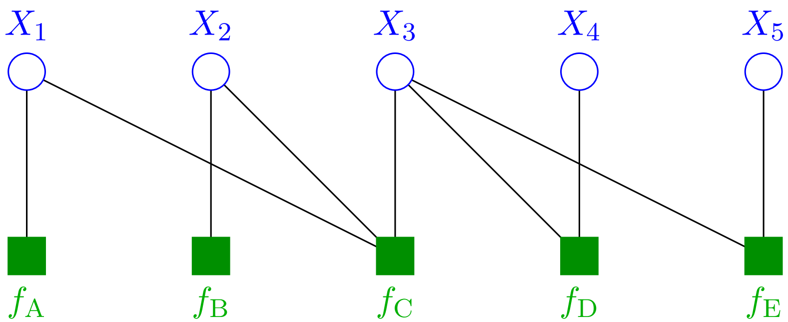
$$f(x_1, x_2, x_3, x_4, x_5) \\ = f_\mathrm{A}(x_1) \cdot f_\mathrm{B}(x_2) \cdot f_\mathrm{C}(x_1, x_2, x_3) \cdot f_\mathrm{D}(x_3, x_4) \cdot f_\mathrm{E}(x_3, x_5)$$

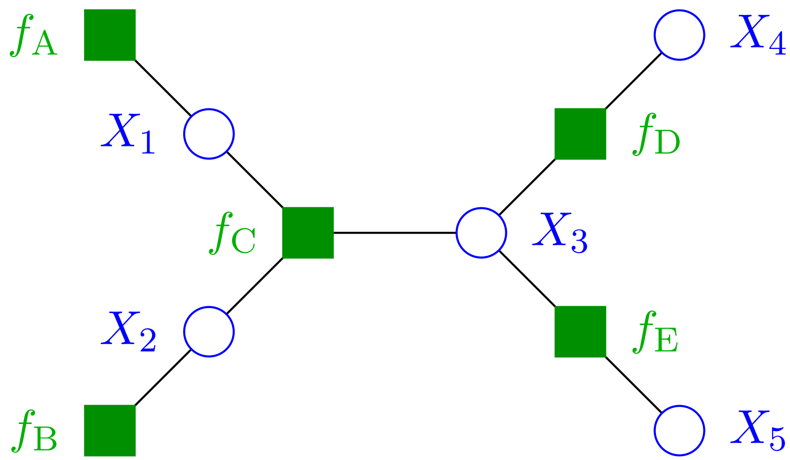




Factor Graphs

$$f(x_1, x_2, x_3, x_4, x_5)$$
$$= f_\mathrm{A}(x_1) \cdot f_\mathrm{B}(x_2) \cdot f_\mathrm{C}(x_1, x_2, x_3) \cdot f_\mathrm{D}(x_3, x_4) \cdot f_\mathrm{E}(x_3, x_5)$$

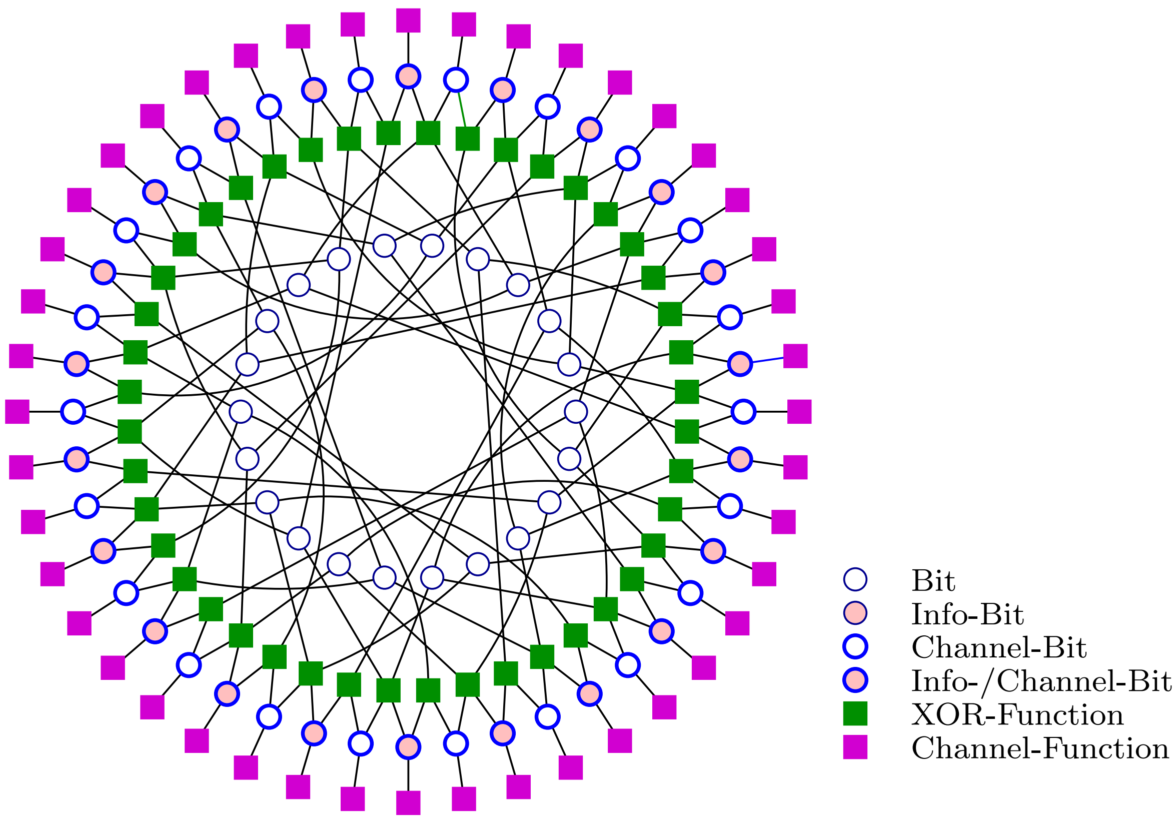




Note: One and the same function can be represented by graphs with different structures: some are more pleasing than others.

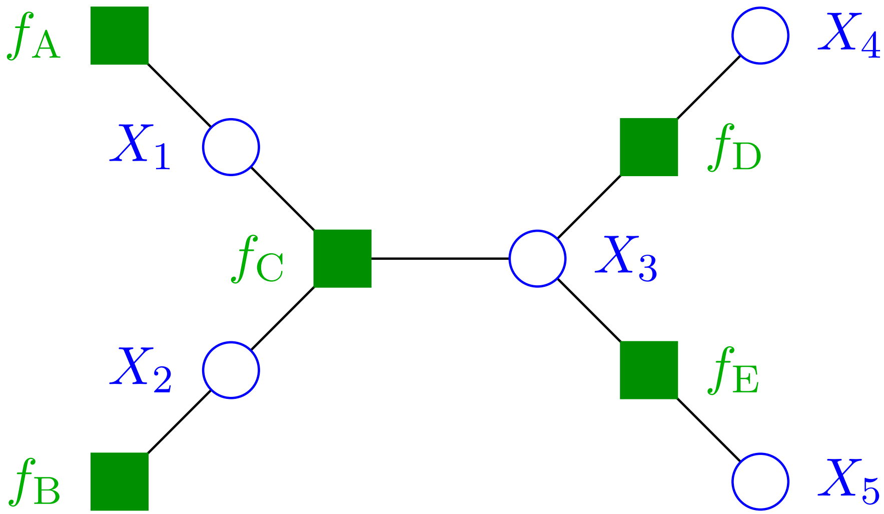
Factor Graph of an LDPC Code

In the context of channel coding, we usually take a factor graph to represent the factorization of the joint pmf/pdf of all occuring variables, i.e. uncoded symbols, coded symbols, and received symbols. Here it is shown when using a quasi-cyclic repeat-accumulate LDPC code (binary $[44, 22, 8]$ linear code).



The Sum-Product Algorithm

Let us consider again the following factor graph (which is a tree).



The global function is

$$
\begin{aligned}
& f(x_1, x_2, x_3, x_4, x_5) \\
&\qquad = f_\mathrm{A}(x_1) \cdot f_\mathrm{B}(x_2) \cdot f_\mathrm{C}(x_1, x_2, x_3) \cdot f_\mathrm{D}(x_3, x_4) \ \cdot f_\mathrm{E}(x_3, x_5).
\end{aligned}
$$

The Sum-Product Algorithm

The global function is

$$f(x_1, x_2, x_3, x_4, x_5) = f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5).$$

The Sum-Product Algorithm

The global function is

$$f(x_1, x_2, x_3, x_4, x_5) = f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5).$$

Very often one wants to calculate marginal functions. E.g.

$$\begin{aligned}\eta_{X_1}(x_1) &= \sum_{x_2, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5) \\ &= \sum_{x_2, x_3, x_4, x_5} f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5).\end{aligned}$$

The Sum-Product Algorithm

The global function is

$$f(x_1, x_2, x_3, x_4, x_5) = f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5).$$

Very often one wants to calculate marginal functions. E.g.

$$\begin{aligned}
\eta_{X_1}(x_1) &= \sum_{x_2, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5) \\
&= \sum_{x_2, x_3, x_4, x_5} f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5).
\end{aligned}$$

$$\begin{aligned}
\eta_{X_3}(x_3) &= \sum_{x_1, x_2, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5) \\
&= \sum_{x_1, x_2, x_4, x_5} f_{\mathrm{A}}(x_1) \cdot f_{\mathrm{B}}(x_2) \cdot f_{\mathrm{C}}(x_1, x_2, x_3) \cdot f_{\mathrm{D}}(x_3, x_4) \cdot f_{\mathrm{E}}(x_3, x_5).
\end{aligned}$$

etc.

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{}}_{} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{}}_{} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{}}_{}}_{}}_{}$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)} \; \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \; \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}} \; \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}} \; \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}}}$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}\ \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3)\ \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}}\ \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}}\ \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}}$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{}}_{} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{}}_{}$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}\ \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3)\ \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}}\ \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)}\ \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}}$$

The Sum-Product Algorithm

$$
\begin{aligned}
\eta_{X_1}(x_1) &= \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)\\
&= \underbrace{f_{\mathrm{A}}(x_1)}\;\underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3)\;\underbrace{\underbrace{f_{\mathrm{B}}(x_2)}}\;\underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot\underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)}\;\underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot\underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3\to f_{\mathrm{C}}}}(x_3)}}
\end{aligned}
$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)} \quad \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \quad \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}} \quad \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \quad \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3\to f_{\mathrm{C}}}}(x_3)}}$$

The Sum-Product Algorithm

$$
\begin{aligned}
\eta_{X_1}(x_1) &= \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5) \\
&= \underbrace{f_{\mathrm{A}}(x_1)} \quad \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \quad \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \quad \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3\to f_{\mathrm{C}}}}(x_3)}}
\end{aligned}
$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3\to f_{\mathrm{C}}}}(x_3)}}_{\mu_{f_{\mathrm{C}}\to X_1}(x_1)}$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{\mu_{f_{\mathrm{A}}\to X_1}(x_1)} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3}\to f_{\mathrm{C}}}(x_3)}}_{\mu_{f_{\mathrm{C}}\to X_1}(x_1)}$$

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{\mu_{f_{\mathrm{A}}\to X_1}(x_1)} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3\to f_{\mathrm{C}}}}(x_3)}}_{\mu_{f_{\mathrm{C}}\to X_1}(x_1)}$$

The objects $\mu_{X_i\to f_j}(x_i)$ and $\mu_{f_j\to X_i}(x_i)$:

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{\mu_{f_{\mathrm{A}}\to X_1}(x_1)} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3\to f_{\mathrm{C}}}}(x_3)}}_{\mu_{f_{\mathrm{C}}\to X_1}(x_1)}$$

The objects $\mu_{X_i\to f_j}(x_i)$ and $\mu_{f_j\to X_i}(x_i)$:

- They are called messages.

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{\mu_{f_{\mathrm{A}}\to X_1}(x_1)} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3}\to f_{\mathrm{C}}}(x_3)}}_{\mu_{f_{\mathrm{C}}\to X_1}(x_1)}$$

The objects $\mu_{X_i\to f_j}(x_i)$ and $\mu_{f_j\to X_i}(x_i)$:

- They are called messages.
- They can be associated with the edge between the vertex X_i and the vertex f_j.

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{\mu_{f_{\mathrm{A}}\to X_1}(x_1)} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3)\; \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3}\to f_{\mathrm{C}}}(x_3)}}_{\mu_{f_{\mathrm{C}}\to X_1}(x_1)}$$

The objects $\mu_{X_i\to f_j}(x_i)$ and $\mu_{f_j\to X_i}(x_i)$:

- They are called messages.
- They can be associated with the edge between the vertex X_i and the vertex f_j.
- They are functions of x_i, i.e. their domain is the alphabet of X_i.

The Sum-Product Algorithm

$$\eta_{X_1}(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_{\mathrm{A}}(x_1)\cdot f_{\mathrm{B}}(x_2)\cdot f_{\mathrm{C}}(x_1,x_2,x_3)\cdot f_{\mathrm{D}}(x_3,x_4)\cdot f_{\mathrm{E}}(x_3,x_5)$$

$$= \underbrace{f_{\mathrm{A}}(x_1)}_{\mu_{f_{\mathrm{A}}\to X_1}(x_1)} \underbrace{\sum_{x_2}\sum_{x_3} f_{\mathrm{C}}(x_1,x_2,x_3) \underbrace{\underbrace{f_{\mathrm{B}}(x_2)}_{\mu_{f_{\mathrm{B}}\to X_2}(x_2)}}_{\mu_{X_2\to f_{\mathrm{C}}}(x_2)} \underbrace{\underbrace{\sum_{x_4} f_{\mathrm{D}}(x_3,x_4)\cdot \underbrace{1}_{\mu_{X_4\to f_{\mathrm{D}}}(x_4)}}_{\mu_{f_{\mathrm{D}}\to X_3}(x_3)} \underbrace{\sum_{x_5} f_{\mathrm{E}}(x_3,x_5)\cdot \underbrace{1}_{\mu_{X_5\to f_{\mathrm{E}}}(x_5)}}_{\mu_{f_{\mathrm{E}}\to X_3}(x_3)}}_{\mu_{f_{X_3\to f_{\mathrm{C}}}}(x_3)}}_{\mu_{f_{\mathrm{C}}\to X_1}(x_1)}$$

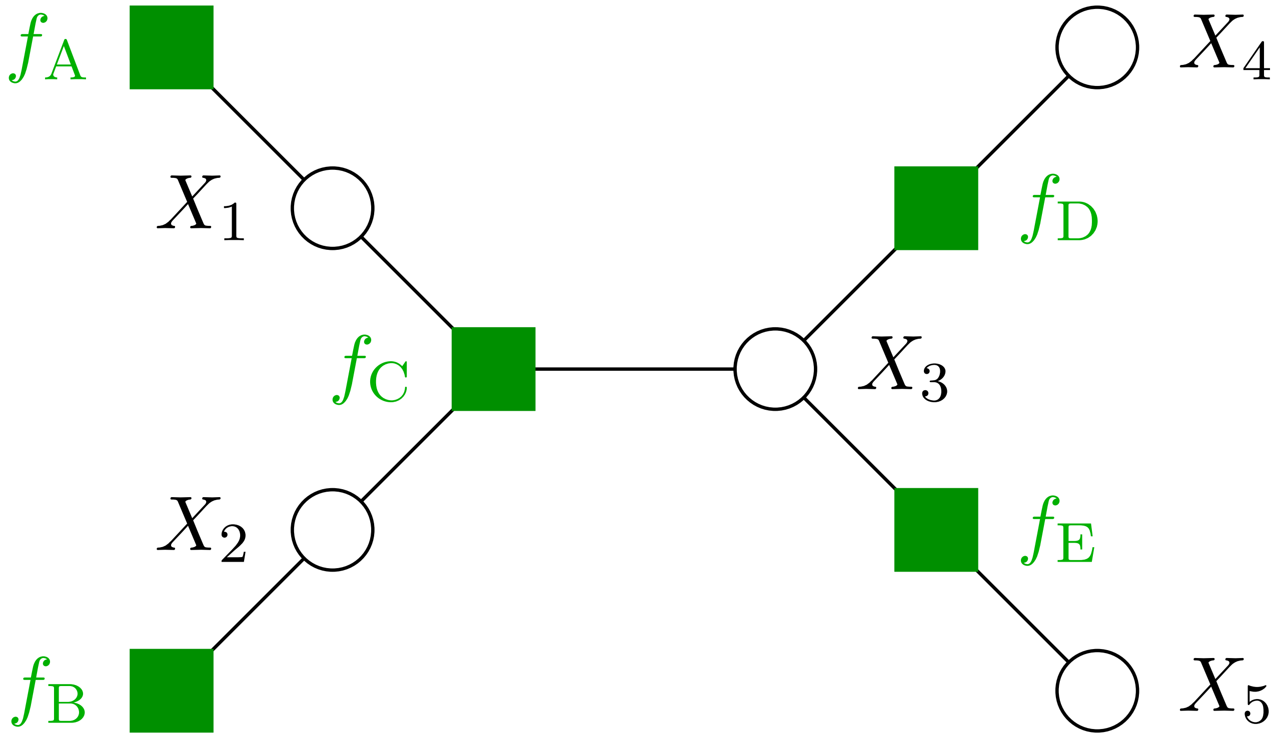
The objects $\mu_{X_i\to f_j}(x_i)$ and $\mu_{f_j\to X_i}(x_i)$:

- They are called messages.
- They can be associated with the edge between the vertex X_i and the vertex f_j.
- They are functions of x_i, i.e. their domain is the alphabet of X_i.

Note: similar manipulations can be performed for calculating $\eta_{X_2}(x_2)$, $\eta_{X_3}(x_3)$, $\eta_{X_4}(x_4)$, $\eta_{X_5}(x_5)$.

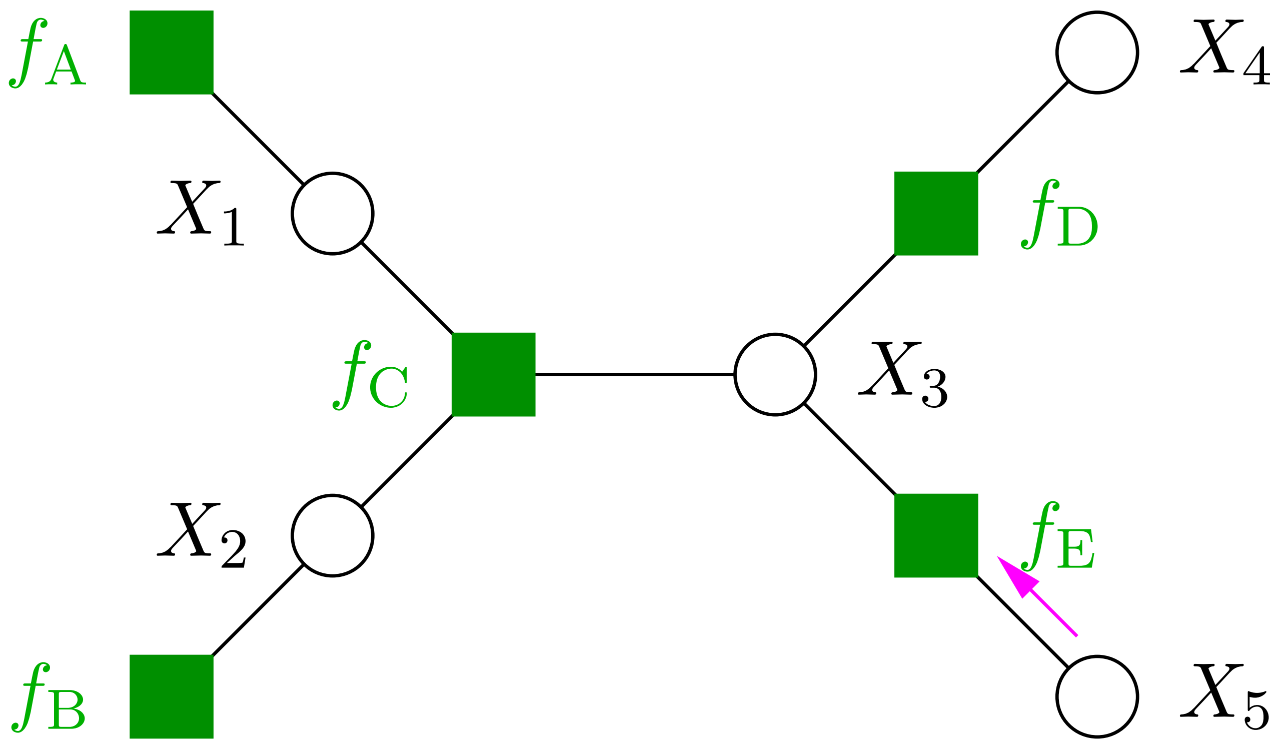
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



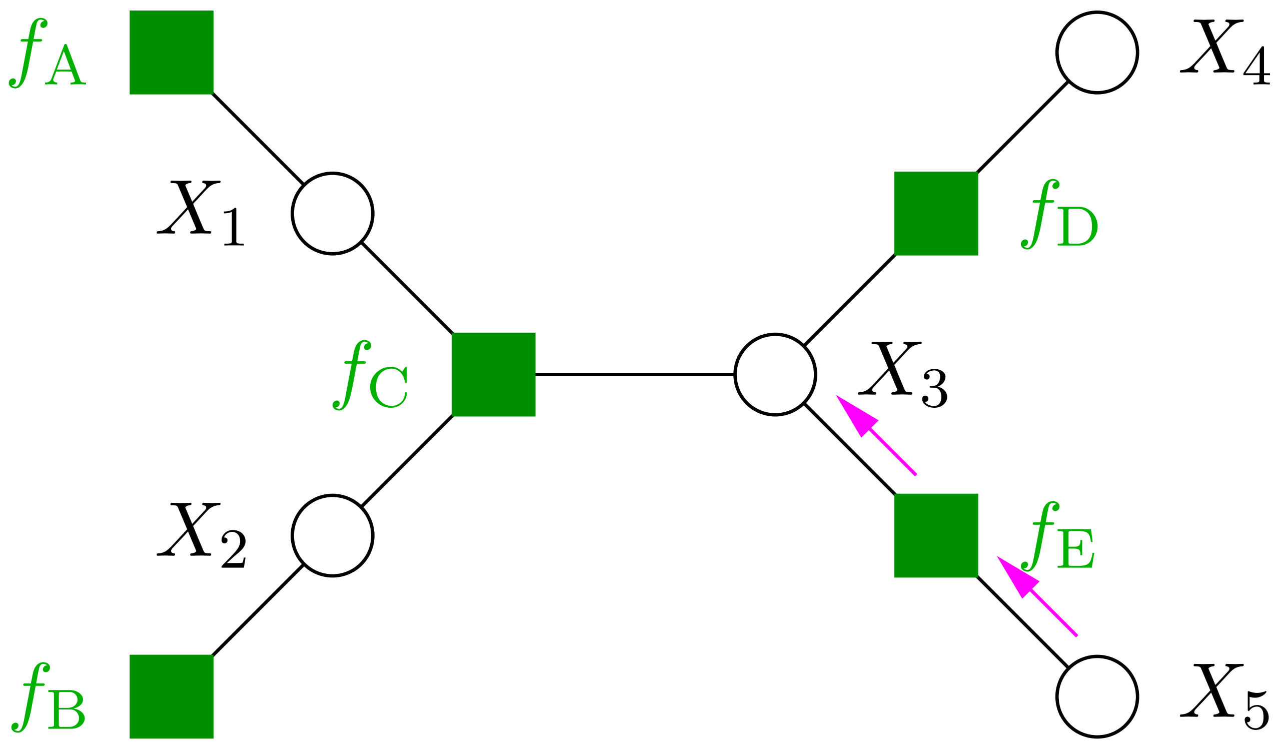
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



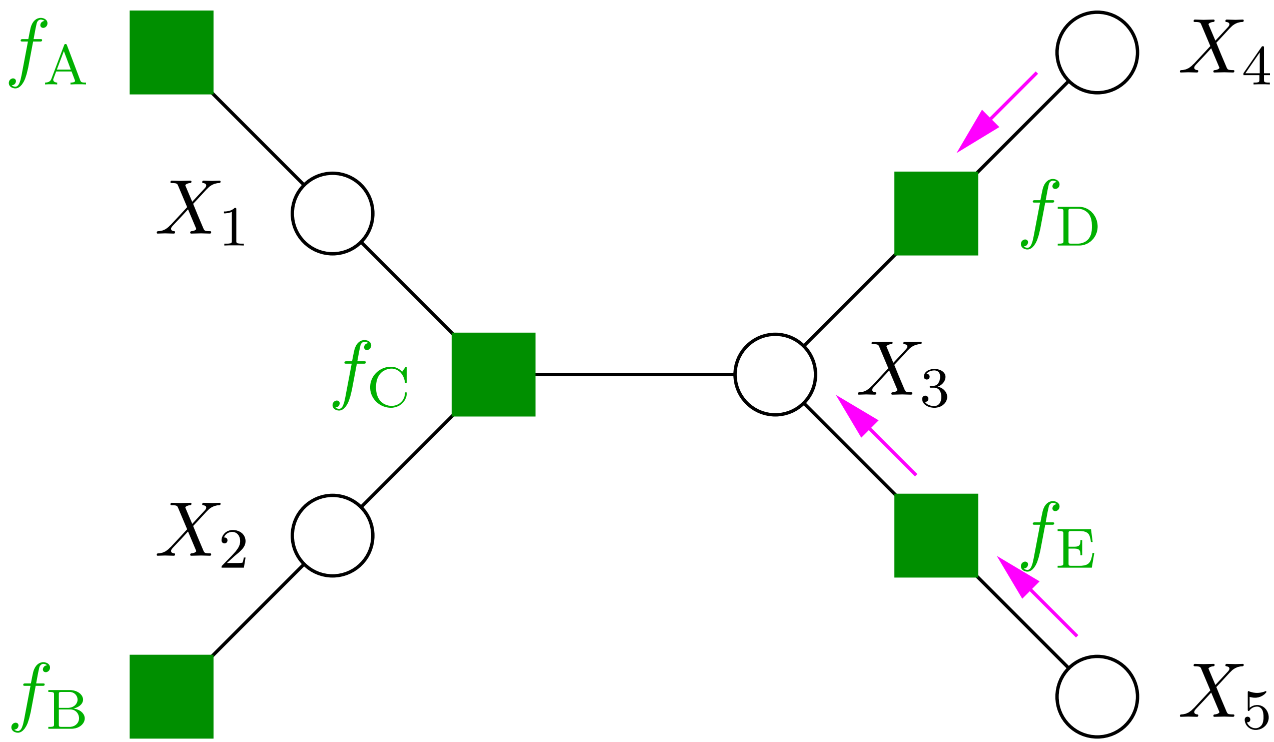
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



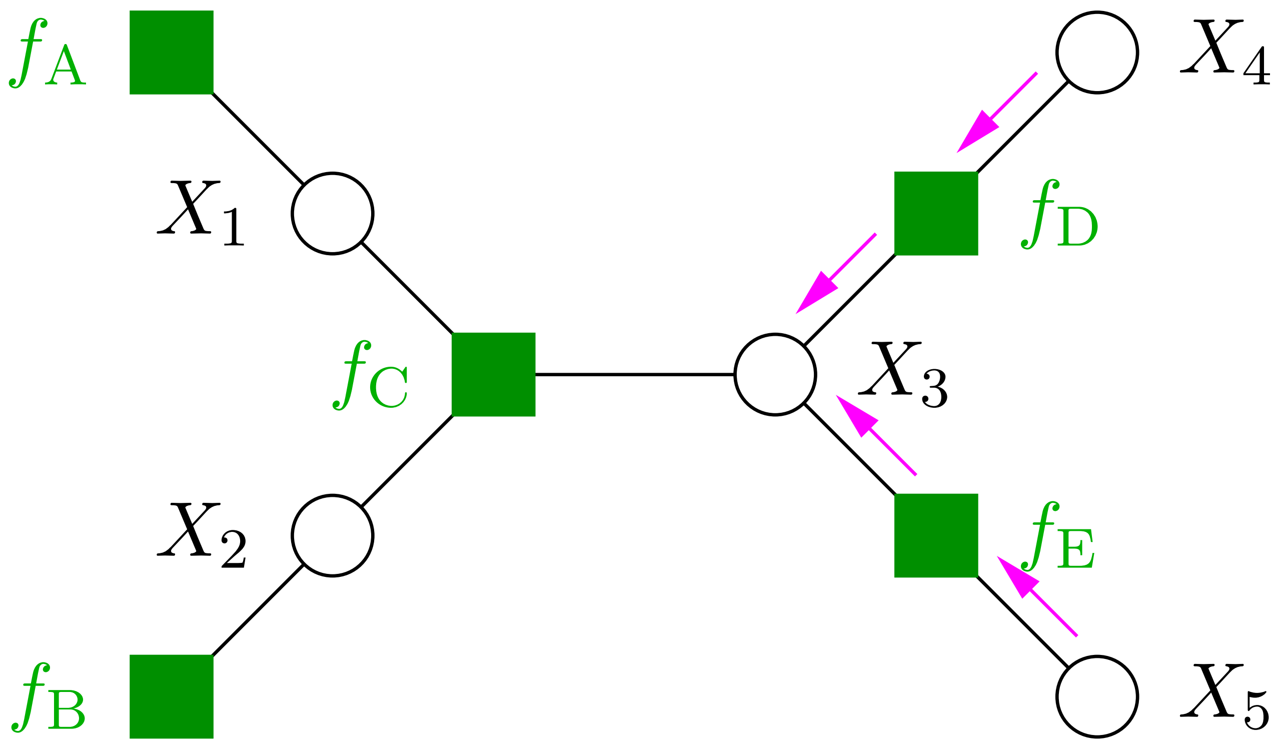
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



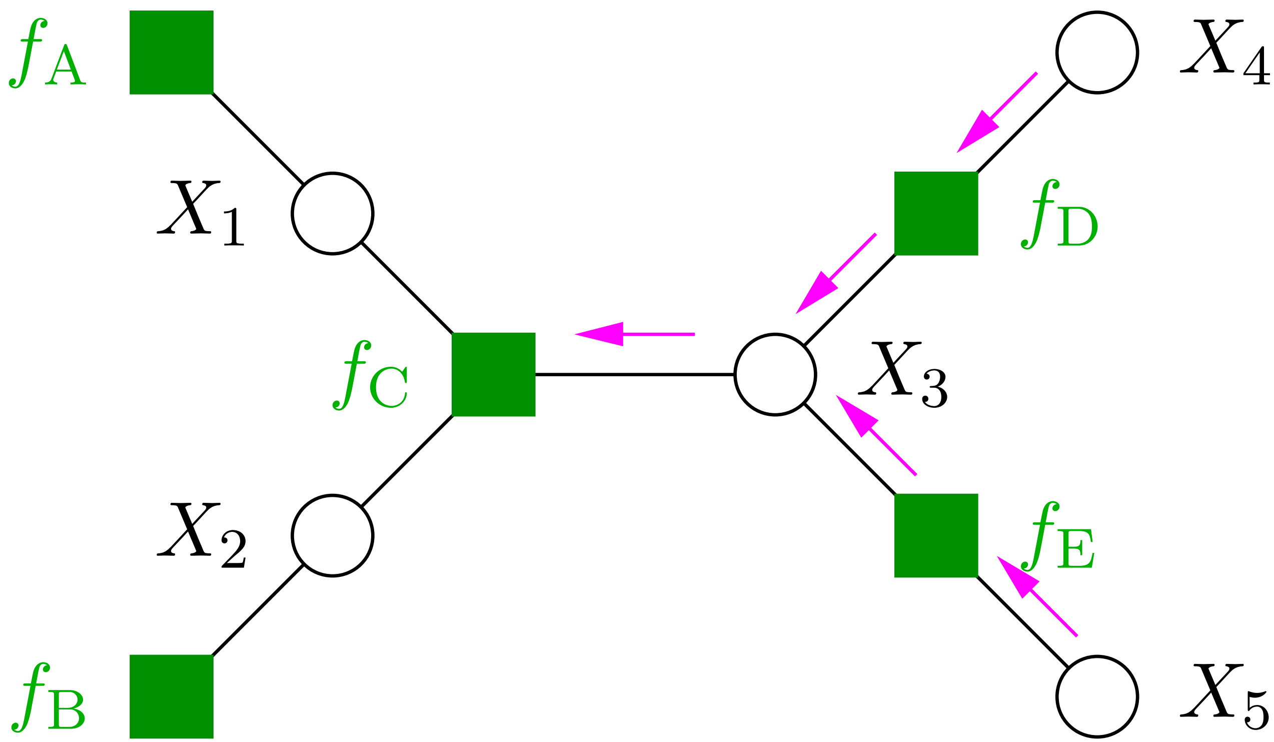
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



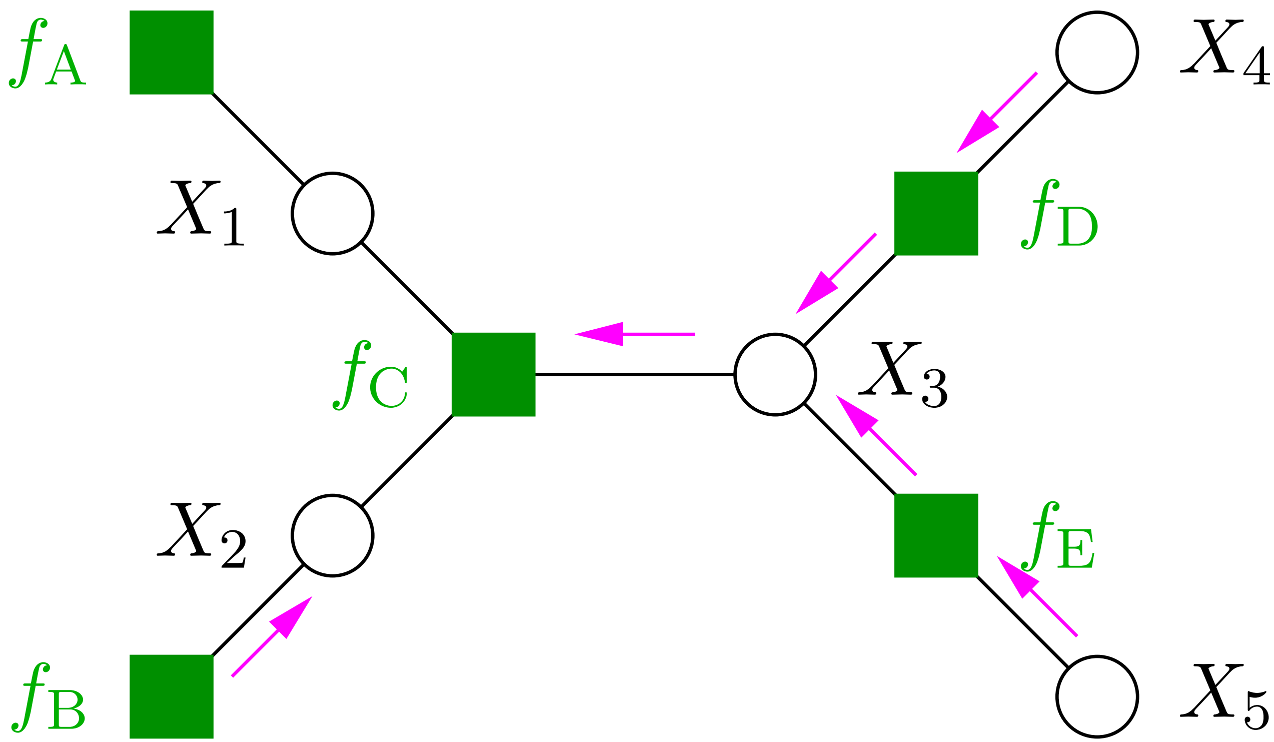
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



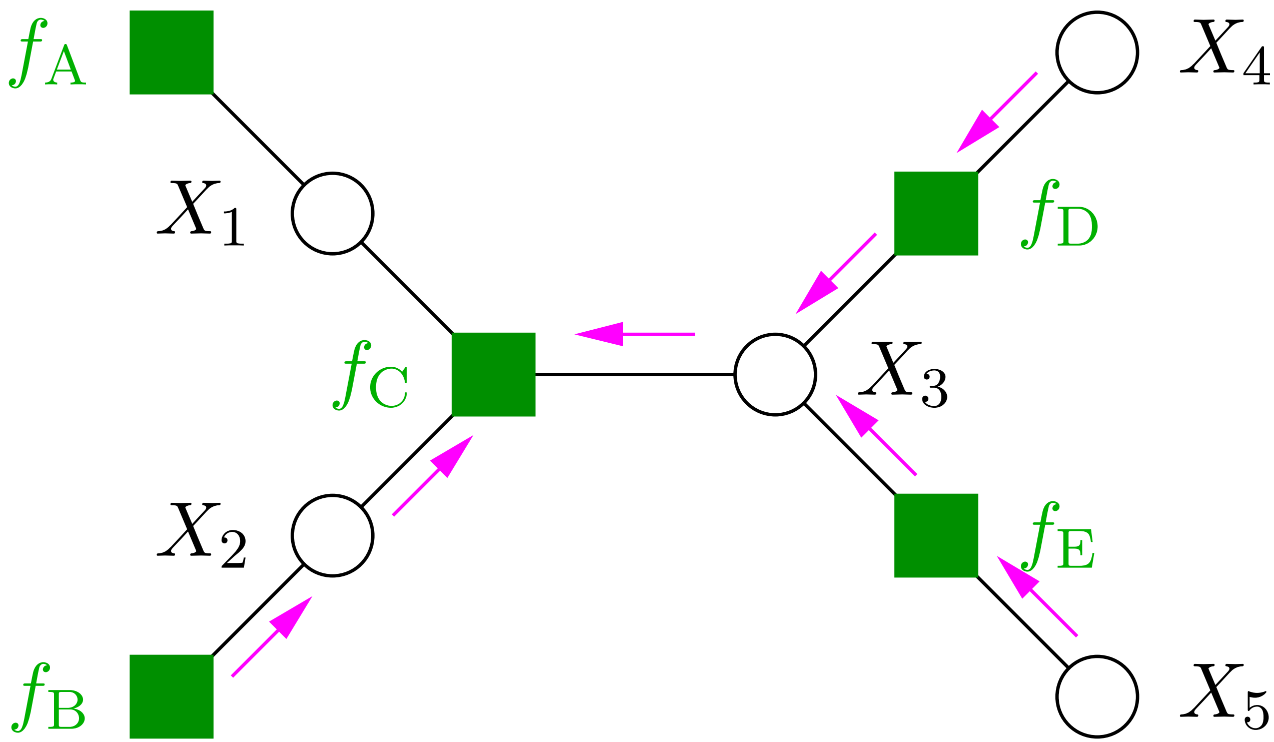
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



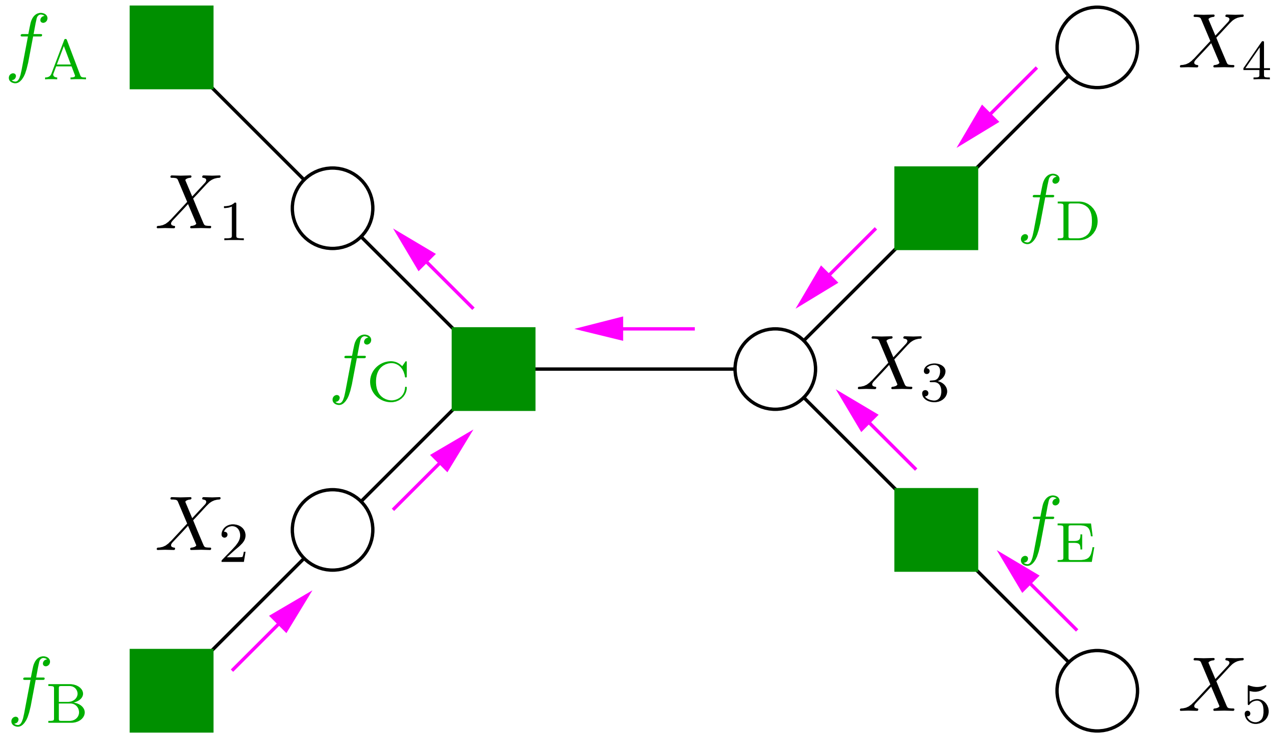
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



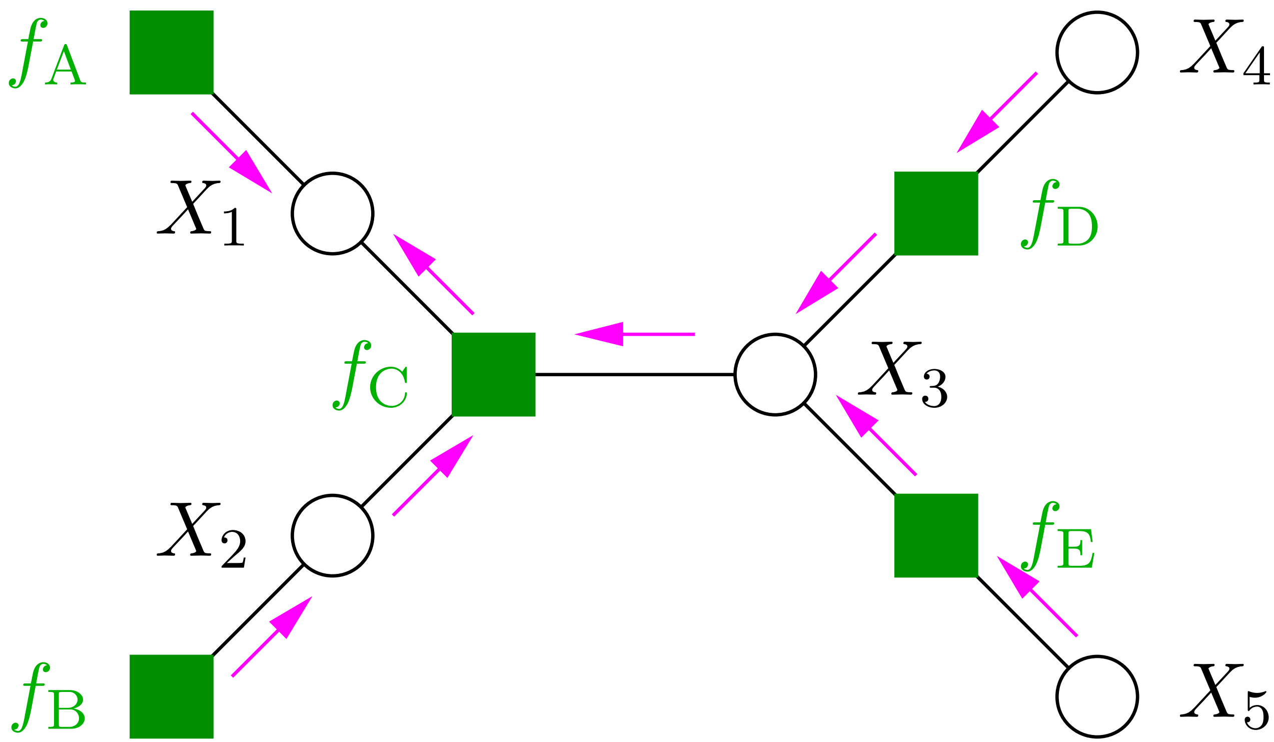
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



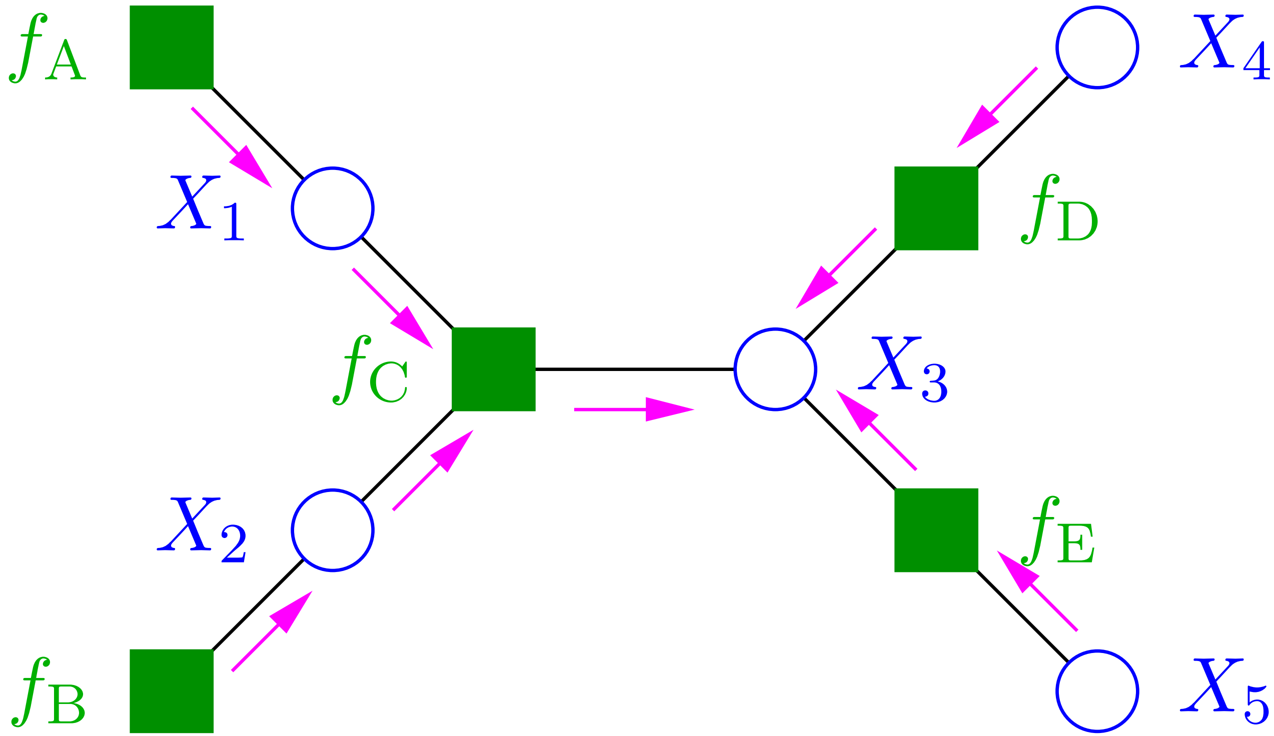
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_1}(x_1)$.



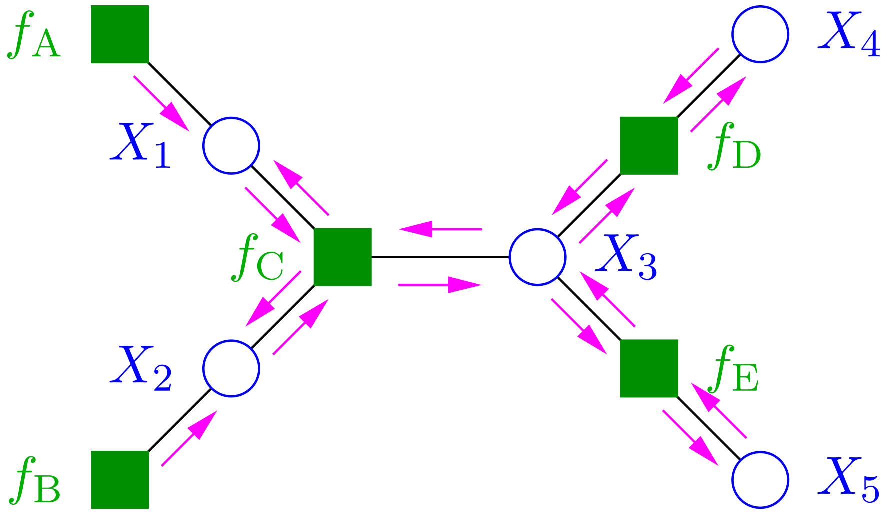
The Sum-Product Algorithm

Messages necessary for calculating $\eta_{X_3}(x_3)$.



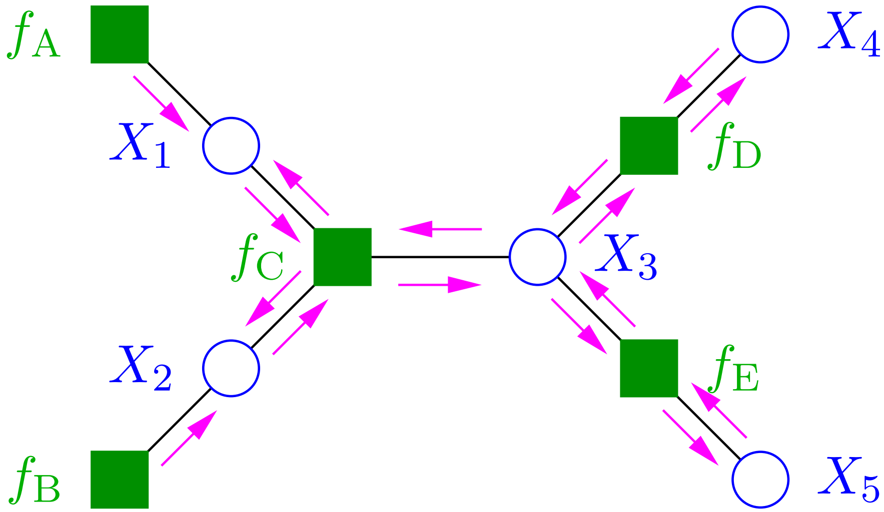
The Sum-Product Algorithm

The figure shows the messages that are necessary for calculating $\eta_{X_1}(x_1)$, $\eta_{X_2}(x_2)$, $\eta_{X_3}(x_3)$, $\eta_{X_4}(x_4)$, and $\eta_{X_5}(x_5)$.



The Sum-Product Algorithm

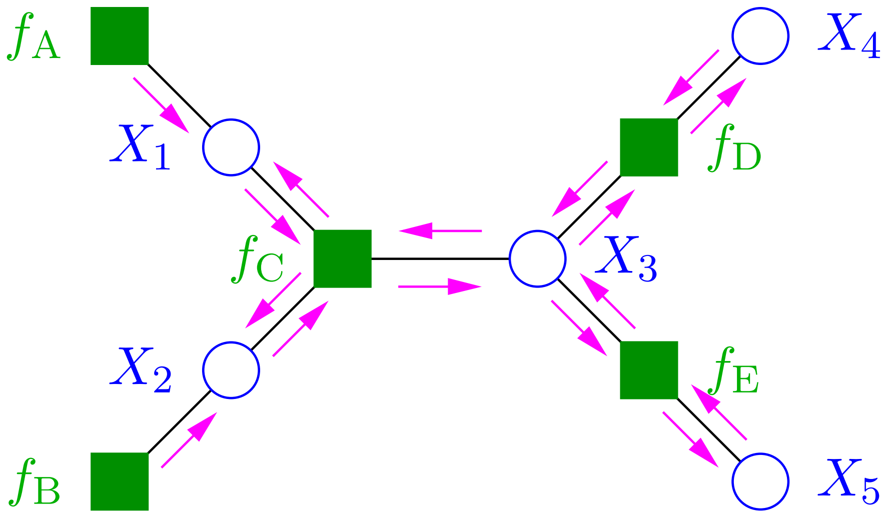
The figure shows the messages that are necessary for calculating $\eta_{X_1}(x_1)$, $\eta_{X_2}(x_2)$, $\eta_{X_3}(x_3)$, $\eta_{X_4}(x_4)$, and $\eta_{X_5}(x_5)$.



- Edges: Messages are sent along edges.

The Sum-Product Algorithm

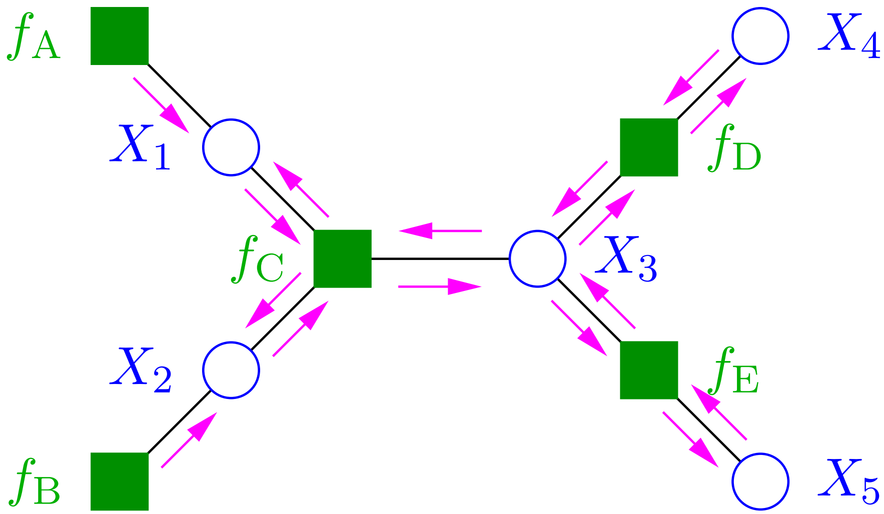
The figure shows the messages that are necessary for calculating $\eta_{X_1}(x_1)$, $\eta_{X_2}(x_2)$, $\eta_{X_3}(x_3)$, $\eta_{X_4}(x_4)$, and $\eta_{X_5}(x_5)$.



- Edges: Messages are sent along edges.
- Processing: Taking products and doing summations is done in the vertices.

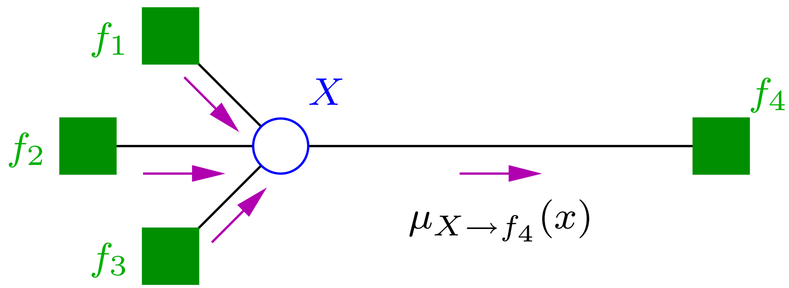
The Sum-Product Algorithm

The figure shows the messages that are necessary for calculating $\eta_{X_1}(x_1)$, $\eta_{X_2}(x_2)$, $\eta_{X_3}(x_3)$, $\eta_{X_4}(x_4)$, and $\eta_{X_5}(x_5)$.



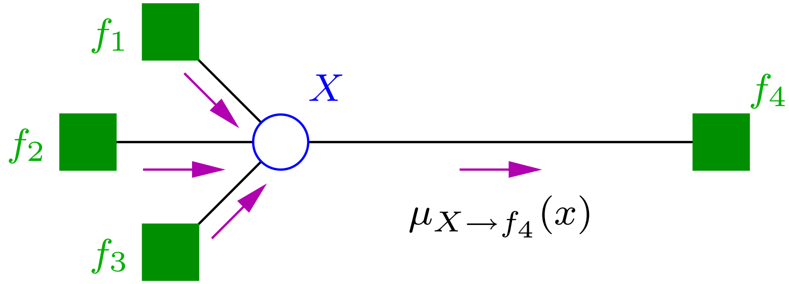
- Edges: Messages are sent along edges.
- Processing: Taking products and doing summations is done in the vertices.
- Reuse of messages: We see that messages can be "reused" in the sense that many partial calculations are the same; so it suffices to perform them only once.

The Sum-Product Algorithm

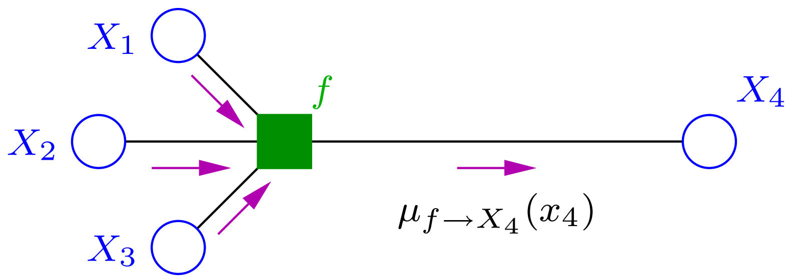


$$\mu_{X\to f_4}(x) = \mu_{f_1\to X}(x)\cdot\mu_{f_2\to X}(x)\cdot\mu_{f_3\to X}(x)$$

The Sum-Product Algorithm

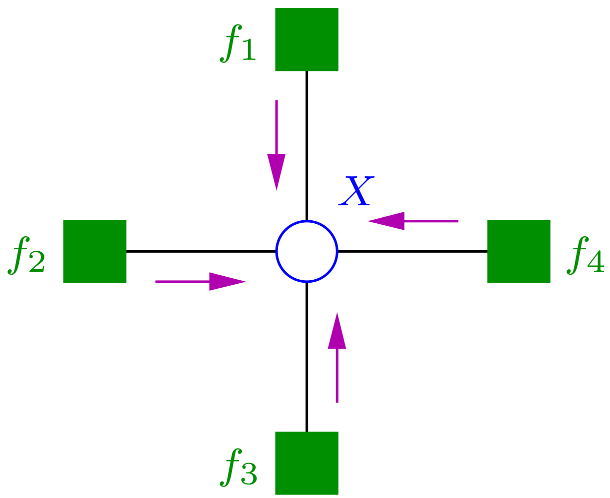


$$\mu_{X\to f_4}(x) = \mu_{f_1\to X}(x)\cdot\mu_{f_2\to X}(x)\cdot\mu_{f_3\to X}(x)$$



$$\mu_{f\to X_4}(x_4) = \sum_{x_1}\sum_{x_2}\sum_{x_3} f(x_1,x_2,x_3,x_4)\cdot\mu_{X_1\to f}(x_1)\cdot\mu_{X_2\to f}(x_2)\cdot\mu_{X_3\to f}(x_3)$$

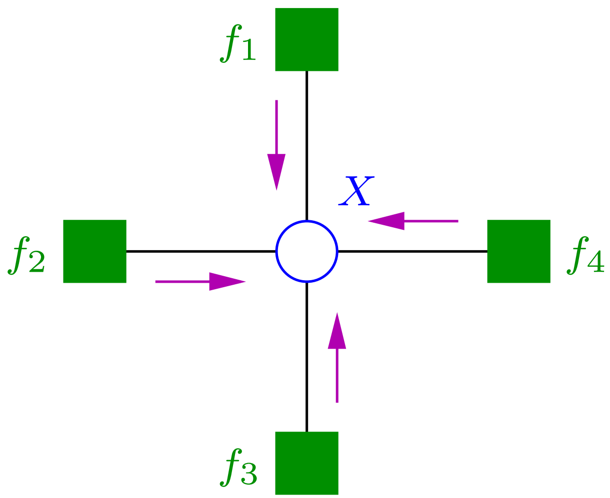
The Sum-Product Algorithm



Computation of marginal at variable node:

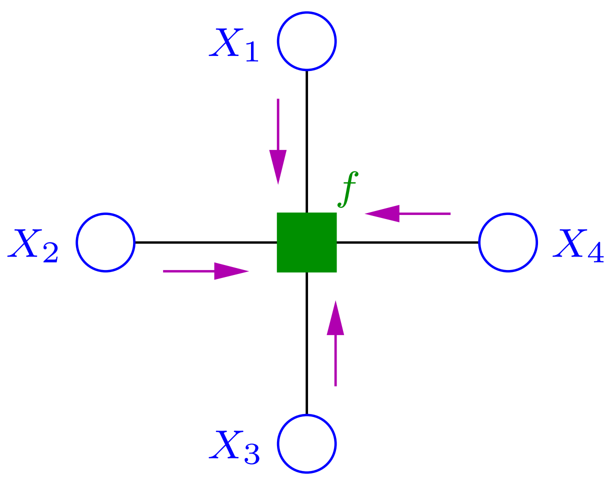
$$\eta_X(x) = \mu_{f_1 \to X}(x) \cdot \mu_{f_2 \to X}(x) \cdot \mu_{f_3 \to X}(x) \cdot \mu_{f_4 \to X}(x)$$

The Sum-Product Algorithm



Computation of marginal at variable node:

$$\eta_X(x) = \mu_{f_1 \to X}(x) \cdot \mu_{f_2 \to X}(x) \cdot \mu_{f_3 \to X}(x) \cdot \mu_{f_4 \to X}(x)$$



Computation of marginal at function node:

$$\eta_f(x_1, x_2, x_3, x_4) = f(x_1, x_2, x_3, x_4) \cdot \mu_{X_1 \to f}(x_1) \cdot \mu_{X_2 \to f}(x_2) \cdot \mu_{X_3 \to f}(x_3) \cdot \mu_{X_4 \to f}(x_4)$$

The Sum-Product Algorithm

- Factor graph without loops: in this case it is obvious what messages have to be calculated when.

$\Rightarrow$ Mode of operation 1

The Sum-Product Algorithm

- Factor graph without loops: in this case it is obvious what messages have to be calculated when.

  $\Rightarrow$ Mode of operation 1

- Factor graph with loops: one has to decide what update schedule to take.

  $\Rightarrow$ Mode of operation 2

Comments on the Sum-Product Algorithm

- If the factor graph has no loops then it is obvious what messages have to be calculated when.
- If the factor graphs has loops one has to decide what update schedule to take.
- Depending on the underlying semi-ring one gets different versions of the summary-product algorithm.
  - For $\langle \mathbb{R}, +, \cdot \rangle$ one gets the sum-product algorithm. (This is the case discussed above.)
  - For $\langle \mathbb{R}^{+}, \max, \cdot \rangle$ one gets the max-product algorithm.
  - For $\langle \mathbb{R}, \min, + \rangle$ one gets the min-sum algorithm.
  - etc.

Partition function (total sum)

Partition Function

$$Z = \sum_{x_1, x_2, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5)$$

Partition Function

$$Z = \sum_{x_1, x_2, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5)$$

Recall:

$$\eta_{X_1}(x_1) = \sum_{x_2, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5)$$

$$\eta_{X_2}(x_2) = \sum_{x_1, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5)$$

$$\vdots \qquad\qquad \vdots$$

Partition Function

$$Z = \sum_{x_1,x_2,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

Recall:

$$\eta_{X_1}(x_1) = \sum_{x_2,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

$$\eta_{X_2}(x_2) = \sum_{x_1,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

$$\vdots \qquad \vdots$$

Define:

$$Z_{X_1} = \sum_{x_1} \eta_{X_1}(x_1)$$

$$Z_{X_2} = \sum_{x_2} \eta_{X_2}(x_2)$$

$$\vdots \qquad \vdots$$

Partition Function

$$Z = \sum_{x_1,x_2,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

Recall:

$$\eta_{X_1}(x_1) = \sum_{x_2,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

$$\eta_{X_2}(x_2) = \sum_{x_1,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

$$\vdots \qquad \vdots$$

Define:

$$Z_{X_1} = \sum_{x_1} \eta_{X_1}(x_1) = Z$$

$$Z_{X_2} = \sum_{x_2} \eta_{X_2}(x_2) = Z$$

$$\vdots \qquad \vdots$$

Partition Function

$$Z = \sum_{x_1, x_2, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5)$$

Partition Function

$$Z = \sum_{x_1,x_2,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

Recall:

$$\begin{aligned} &\vdots && \vdots \\ \eta_{f_C}(x_1, x_2, x_3) &= \sum_{x_4,x_5} f(x_1, x_2, x_3, x_4, x_5) \\ &\vdots && \vdots \end{aligned}$$

Partition Function

$$Z = \sum_{x_1,x_2,x_3,x_4,x_5} f(x_1, x_2, x_3, x_4, x_5)$$

Recall:

$$\begin{aligned} &\vdots && \vdots \\ \eta_{f_C}(x_1, x_2, x_3) &= \sum_{x_4,x_5} f(x_1, x_2, x_3, x_4, x_5) \\ &\vdots && \vdots \end{aligned}$$

Define:

$$\begin{aligned} &\vdots && \vdots \\ Z_{f_C} &= \sum_{x_1,x_2,x_3} \eta_{f_C}(x_1, x_2, x_3) \\ &\vdots && \vdots \end{aligned}$$

Partition Function

$$Z = \sum_{x_1, x_2, x_3, x_4, x_5} f(x_1, x_2, x_3, x_4, x_5)$$

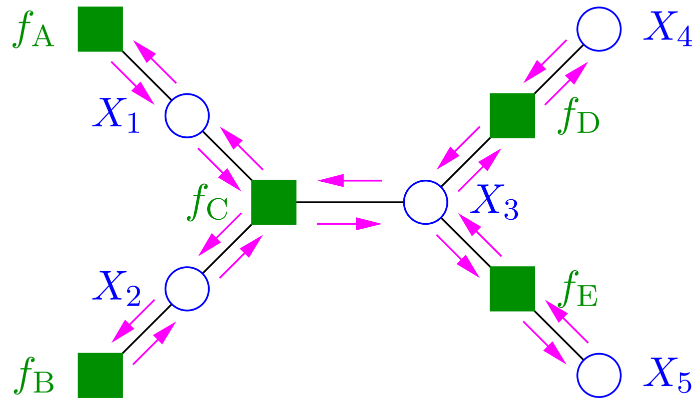
Recall:

$$\begin{aligned} &\vdots && \vdots \\ \eta_{f_C}(x_1, x_2, x_3) &= \sum_{x_4, x_5} f(x_1, x_2, x_3, x_4, x_5) \\ &\vdots && \vdots \end{aligned}$$

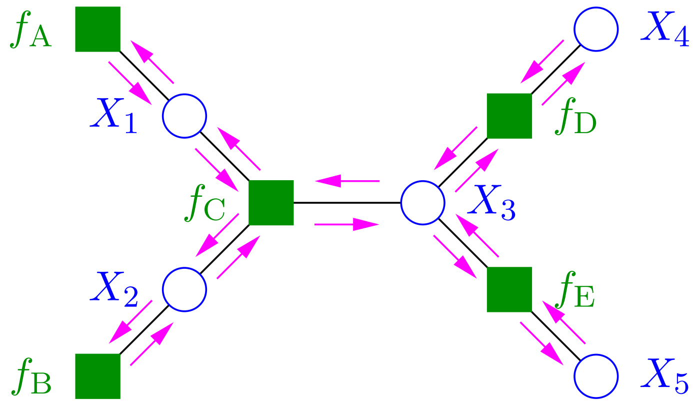
Define:

$$\begin{aligned} &\vdots && \vdots \\ Z_{f_C} &= \sum_{x_1, x_2, x_3} \eta_{f_C}(x_1, x_2, x_3) = Z \\ &\vdots && \vdots \end{aligned}$$

Partition Function

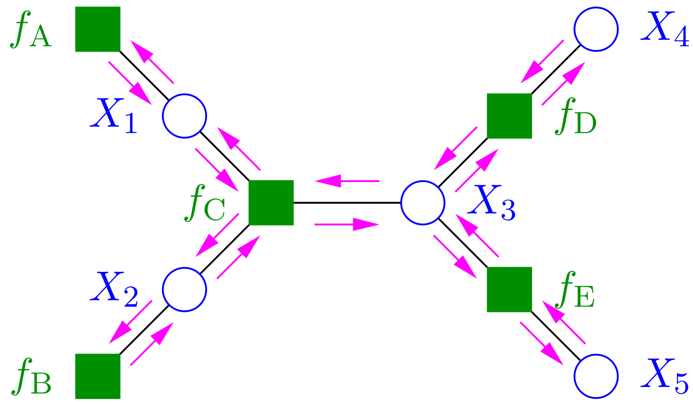


Partition Function



$$Z = Z_{X_1} = Z_{X_2} = Z_{X_3} = Z_{X_4} = Z_{X_5} = Z_{f_A} = Z_{f_B} = Z_{f_C} = Z_{f_D} = Z_{f_E}$$

Partition Function



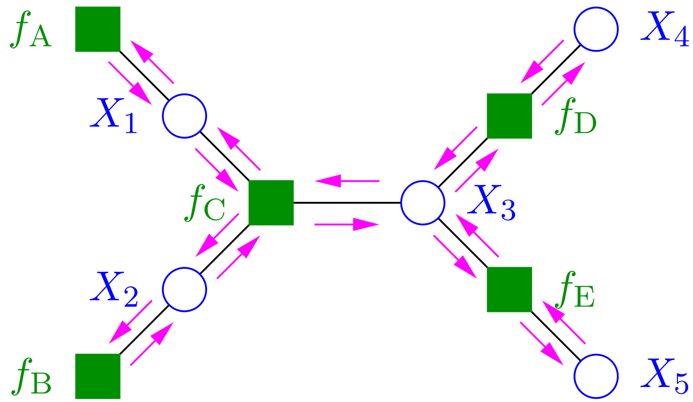
$$Z = Z_{X_1} = Z_{X_2} = Z_{X_3} = Z_{X_4} = Z_{X_5} = Z_{f_{\mathrm{A}}} = Z_{f_{\mathrm{B}}} = Z_{f_{\mathrm{C}}} = Z_{f_{\mathrm{D}}} = Z_{f_{\mathrm{E}}}$$

Claim:

$$Z = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot Z_{f_{\mathrm{D}}} \cdot Z_{f_{\mathrm{E}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

(Note: exponents in denominator equal variable node degrees.)

Partition Function



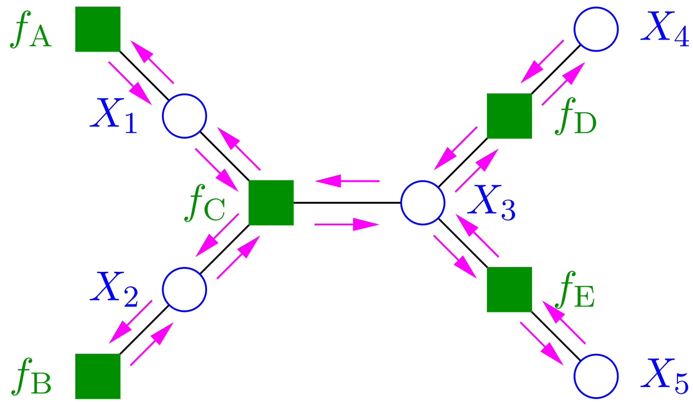
$$Z = Z_{X_1} = Z_{X_2} = Z_{X_3} = Z_{X_4} = Z_{X_5} = Z_{f_{\mathrm{A}}} = Z_{f_{\mathrm{B}}} = Z_{f_{\mathrm{C}}} = Z_{f_{\mathrm{D}}} = Z_{f_{\mathrm{E}}}$$

Claim:

$$Z = \frac{Z^{\#\text{vertices}}}{Z^{\#\text{edges}}} = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot Z_{f_{\mathrm{D}}} \cdot Z_{f_{\mathrm{E}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

(Note: exponents in denominator equal variable node degrees.)

Partition Function



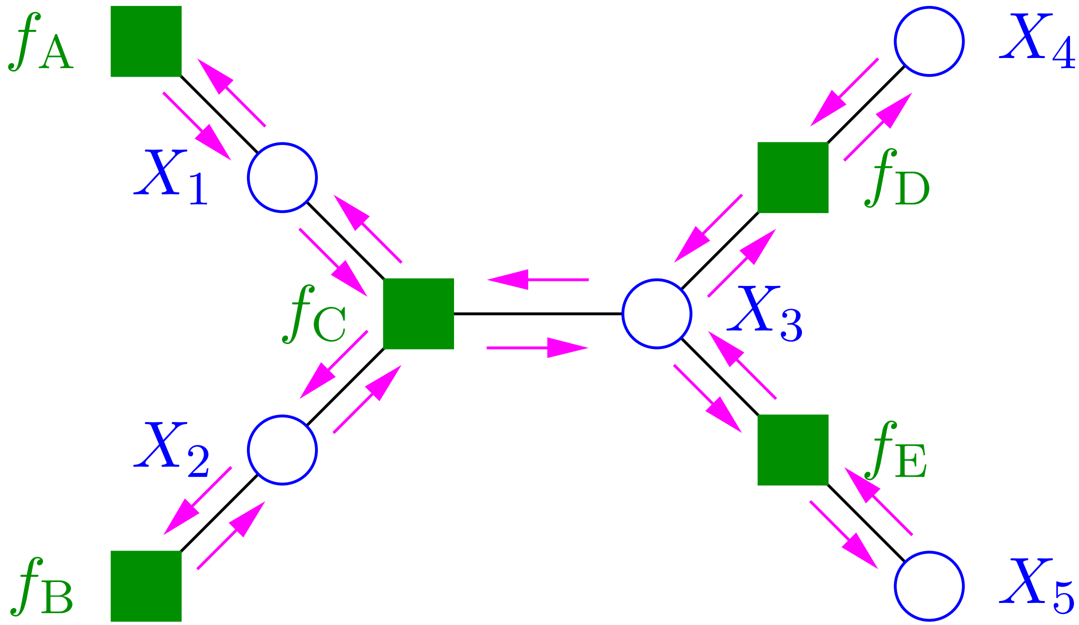
$$Z = Z_{X_1} = Z_{X_2} = Z_{X_3} = Z_{X_4} = Z_{X_5} = Z_{f_{\mathrm{A}}} = Z_{f_{\mathrm{B}}} = Z_{f_{\mathrm{C}}} = Z_{f_{\mathrm{D}}} = Z_{f_{\mathrm{E}}}$$

Claim:

$$Z = \frac{Z^{\#\text{vertices}}}{Z^{\#\text{edges}}} = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot Z_{f_{\mathrm{D}}} \cdot Z_{f_{\mathrm{E}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

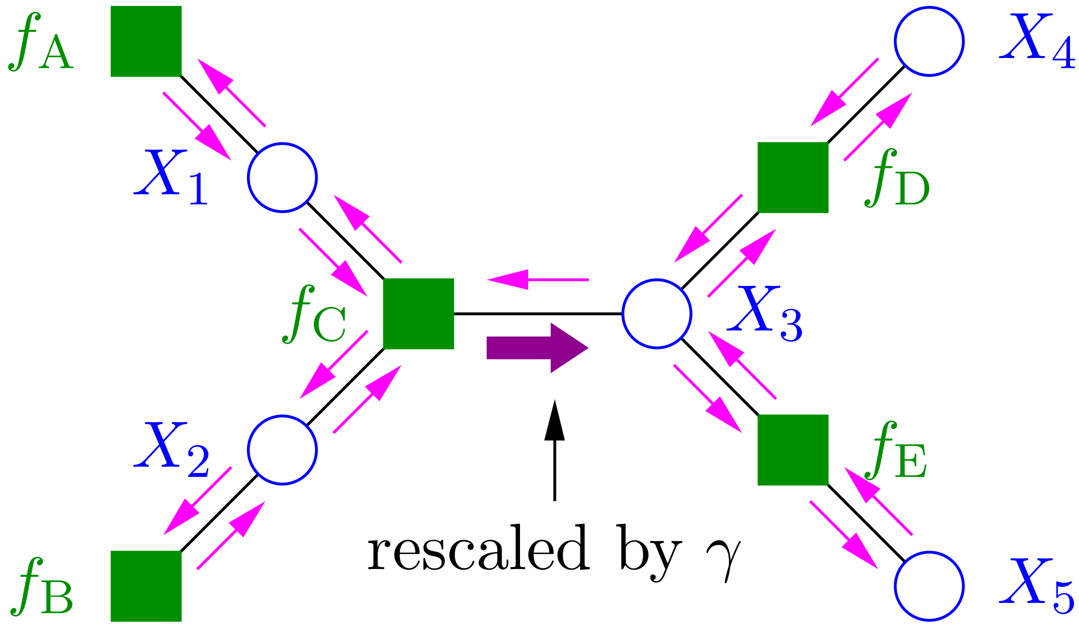
(Here we used the fact that for a graph with one component and no cycles it holds that $\#\text{vertices} = \#\text{edges} + 1$.)

Partition Function



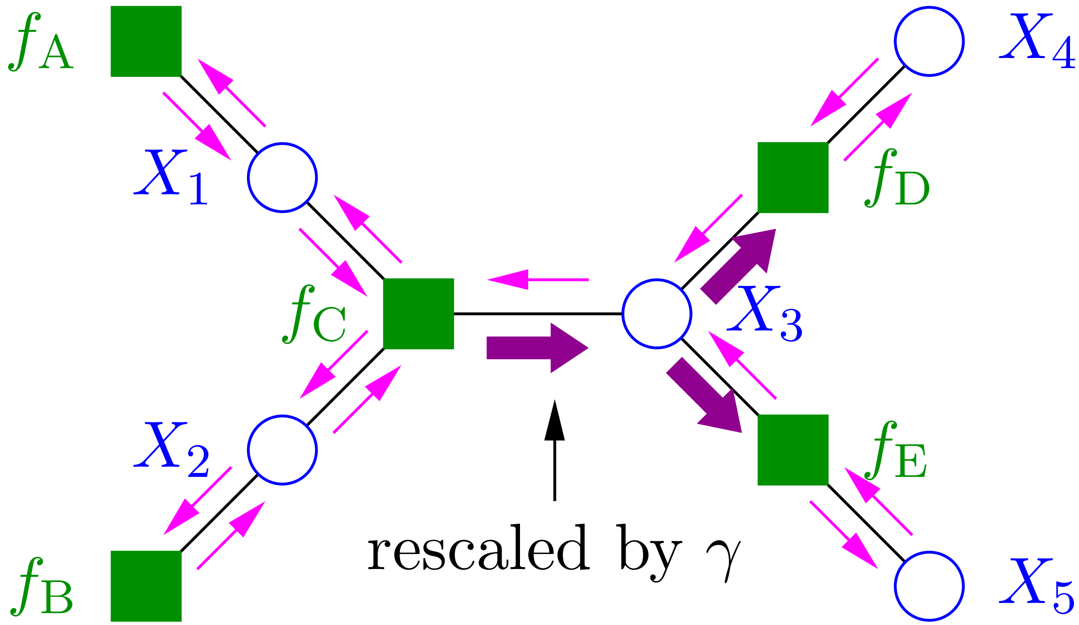
$$Z = \frac{Z_{f_A} \cdot Z_{f_B} \cdot Z_{f_C} \cdot Z_{f_D} \cdot Z_{f_E} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

Partition Function



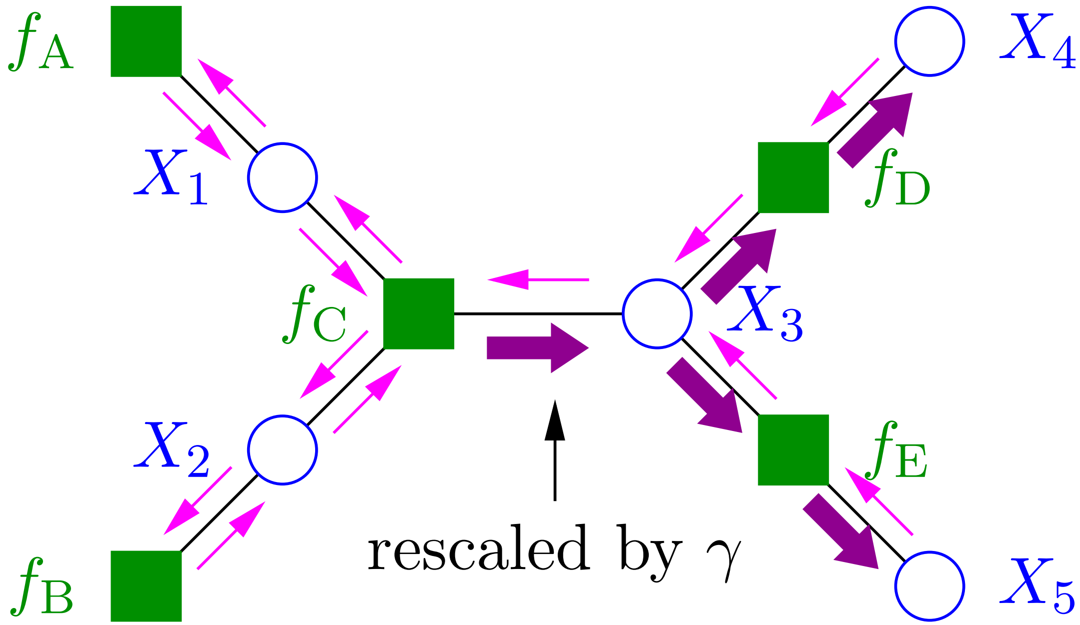
$$Z = \frac{Z_{f_A} \cdot Z_{f_B} \cdot Z_{f_C} \cdot Z_{f_D} \cdot Z_{f_E} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

Partition Function



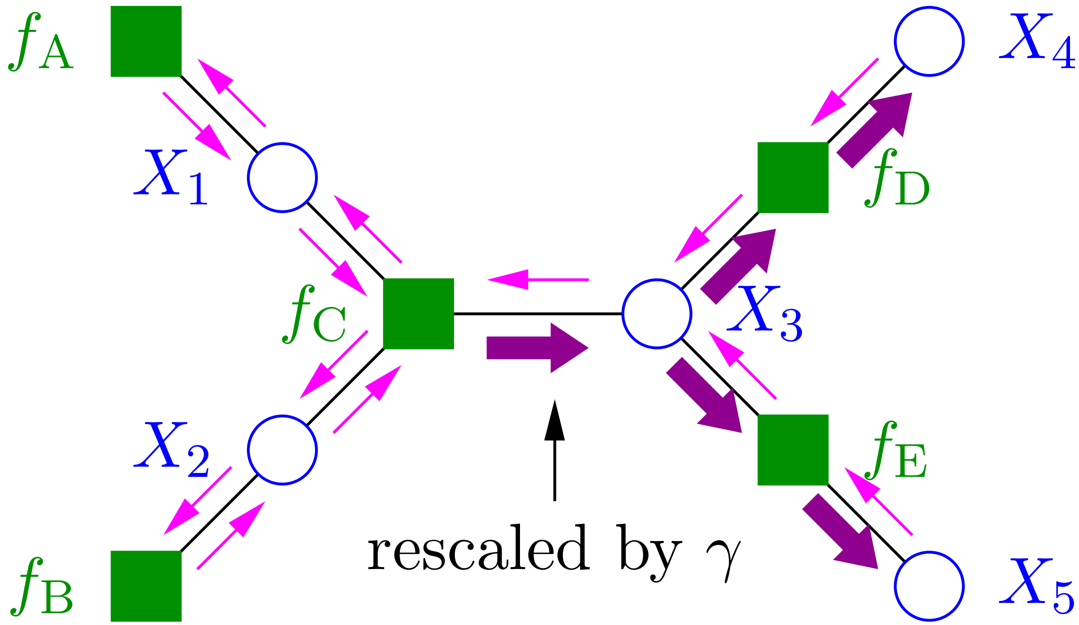
$$Z = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot Z_{f_{\mathrm{D}}} \cdot Z_{f_{\mathrm{E}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

Partition Function



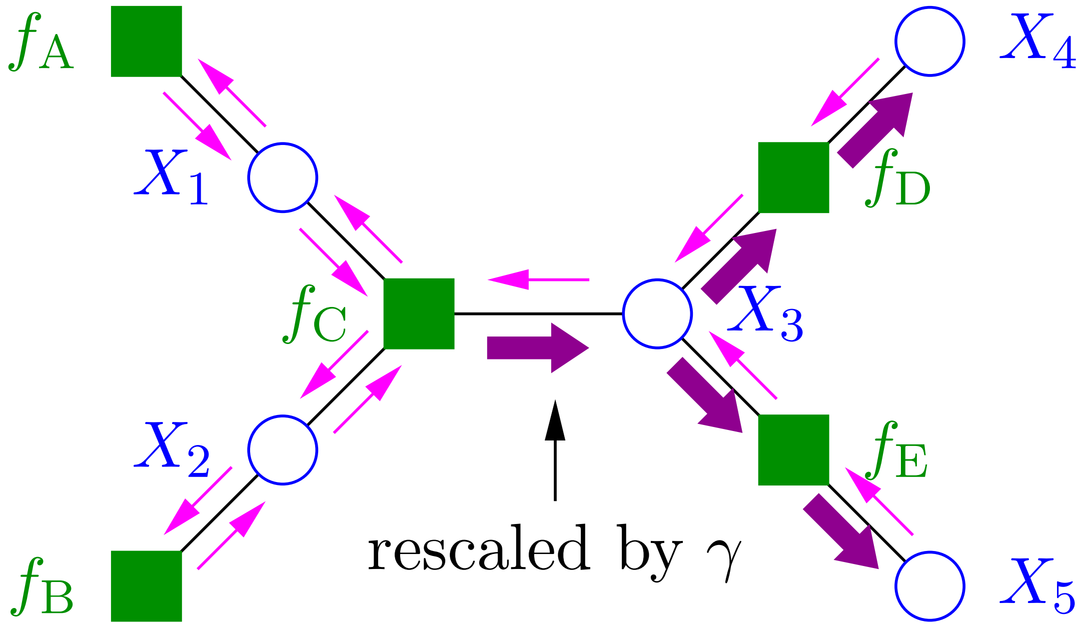
$$Z = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot Z_{f_{\mathrm{D}}} \cdot Z_{f_{\mathrm{E}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

Partition Function



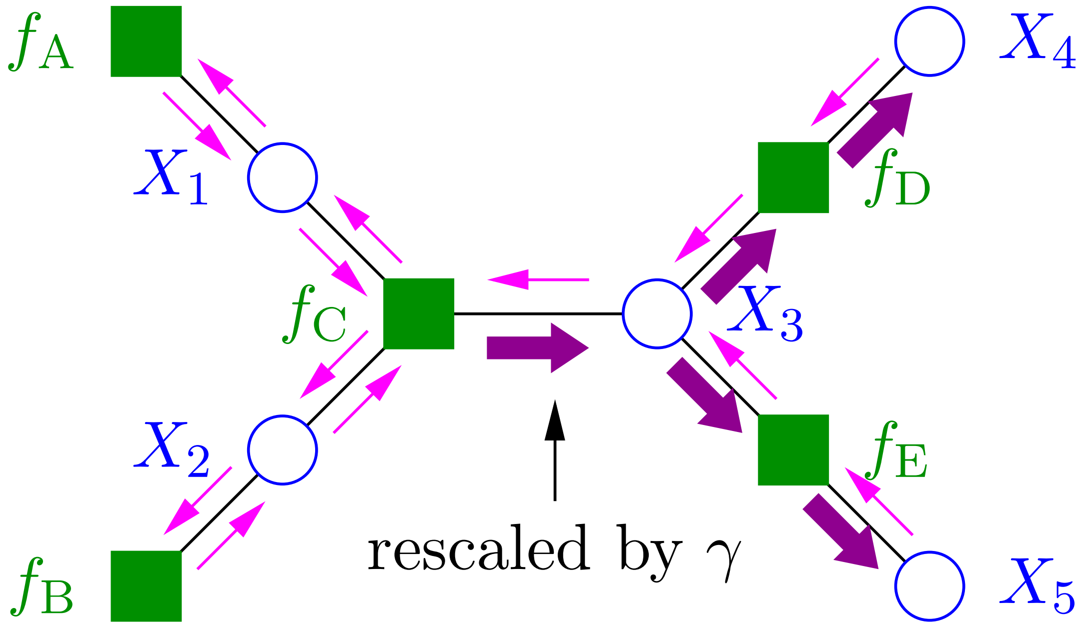
$$Z = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot \boxed{\hat{Z}_{f_{\mathrm{D}}}} \cdot \boxed{\hat{Z}_{f_{\mathrm{E}}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot \boxed{\hat{Z}_{X_3}} \cdot \boxed{\hat{Z}_{X_4}} \cdot \boxed{\hat{Z}_{X_5}}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot \boxed{\hat{Z}_{X_3}^3} \cdot \boxed{\hat{Z}_{X_4}^1} \cdot \boxed{\hat{Z}_{X_5}^1}}$$

Partition Function



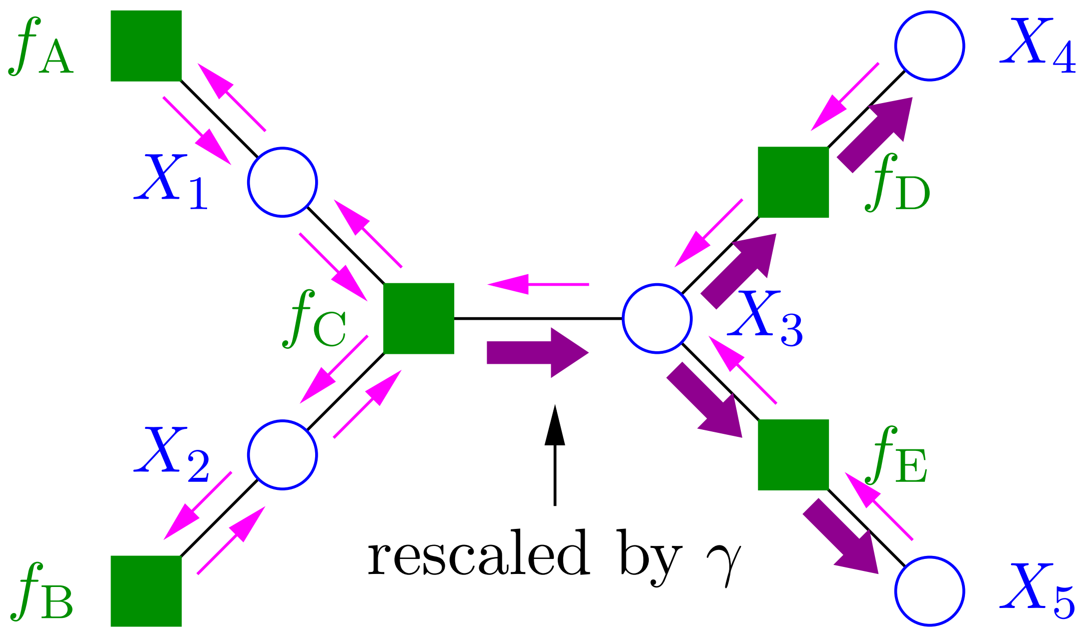
$$Z = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot \boxed{\gamma Z_{f_{\mathrm{D}}}} \cdot \boxed{\gamma Z_{f_{\mathrm{E}}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot \boxed{\gamma Z_{X_3}} \cdot \boxed{\gamma Z_{X_4}} \cdot \boxed{\gamma Z_{X_5}}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot \boxed{\gamma^3 Z_{X_3}^3} \cdot \boxed{\gamma^1 Z_{X_4}^1} \cdot \boxed{\gamma^1 Z_{X_5}^1}}$$

Partition Function



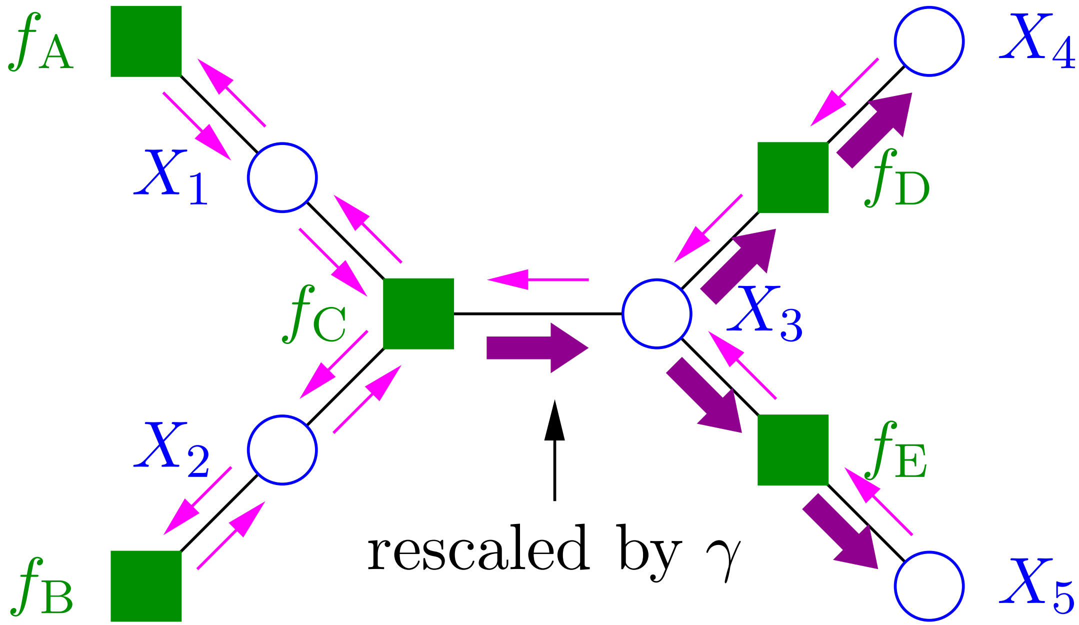
$$\color{red}{Z} = \frac{\gamma^5}{\gamma^5} \cdot \frac{Z_{f_A} \cdot Z_{f_B} \cdot Z_{f_C} \cdot Z_{f_D} \cdot Z_{f_E} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

Partition Function



$$Z = \frac{Z_{f_{\mathrm{A}}} \cdot Z_{f_{\mathrm{B}}} \cdot Z_{f_{\mathrm{C}}} \cdot Z_{f_{\mathrm{D}}} \cdot Z_{f_{\mathrm{E}}} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

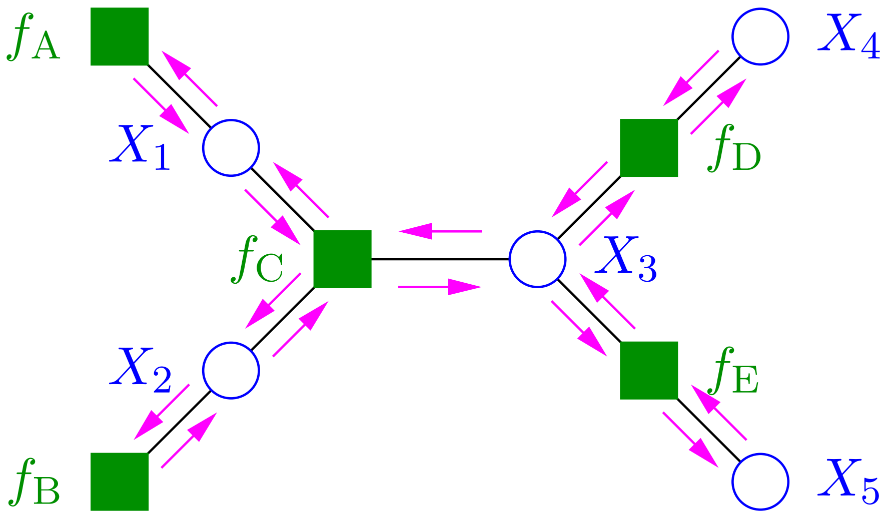
Partition Function



$$Z = \frac{Z_{f_A} \cdot Z_{f_B} \cdot Z_{f_C} \cdot Z_{f_D} \cdot Z_{f_E} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

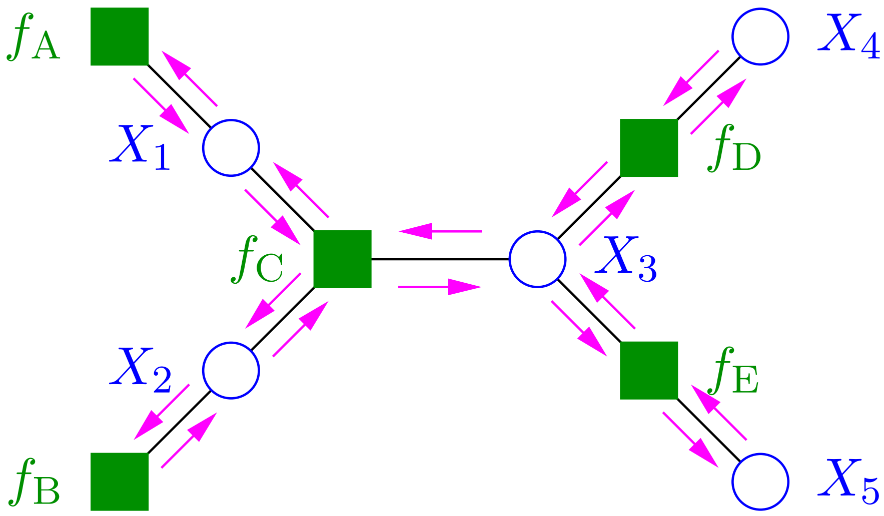
Remarkable: this expression is invariant to rescaling of *function-node-to-variable-node* messages!

Partition Function



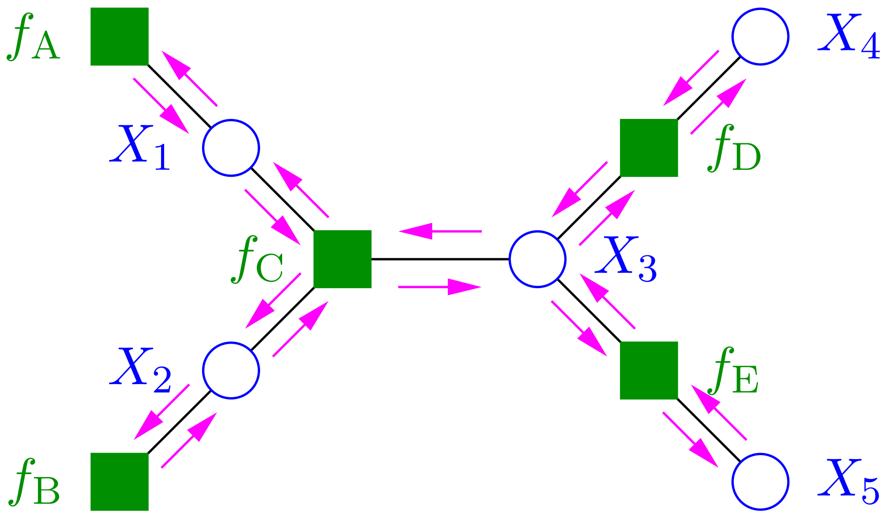
$$Z = \frac{Z_{f_A} \cdot Z_{f_B} \cdot Z_{f_C} \cdot Z_{f_D} \cdot Z_{f_E} \cdot Z_{X_1} \cdot Z_{X_2} \cdot Z_{X_3} \cdot Z_{X_4} \cdot Z_{X_5}}{Z_{X_1}^2 \cdot Z_{X_2}^2 \cdot Z_{X_3}^3 \cdot Z_{X_4}^1 \cdot Z_{X_5}^1}$$

Partition Function



$$Z = \frac{\prod_f Z_f \cdot \prod_X Z_X}{\prod_X Z_X^{\deg(\mathrm{X})}}$$

Partition Function

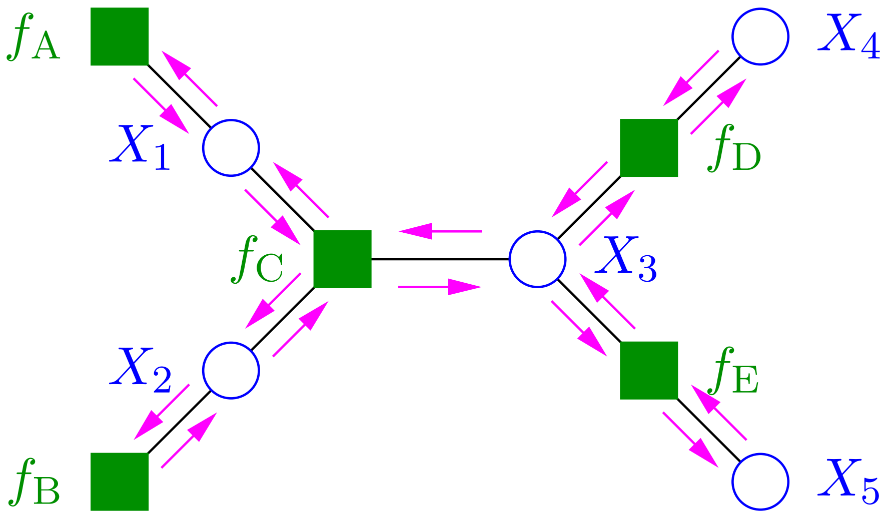


$$\color{red}{Z} = \frac{\prod_f Z_f \cdot \prod_X Z_X}{\prod_X Z_X^{\deg(X)}}$$

Bethe approximation:

Use the above type of expression also when factor graph has cycles.

Partition Function



$$\color{red}{Z} = \frac{\prod_f Z_f \cdot \prod_X Z_X}{\prod_X Z_X^{\deg(X)}}$$

Bethe approximation:

Use the above type of expression also when factor graph has cycles.

$$\rightarrow Z'_{\text{Bethe}}$$

Bethe Partition Function

Bethe Partition Function

- Basically, we can evaluate the expresion for Z'_{Bethe} at any iteration of the SPA.

Bethe Partition Function

- Basically, we can evaluate the expresion for Z'_{Bethe} at any iteration of the SPA.
- Factor graph without cycles:

  We have $Z'_{\mathrm{Bethe}} = Z$ only at a fixed point of the SPA.

Bethe Partition Function

- Basically, we can evaluate the expresion for Z'_{Bethe} at any iteration of the SPA.

- Factor graph without cycles:

  We have $Z'_{\mathrm{Bethe}} = Z$ only at a fixed point of the SPA.

- Factor graph with cycles:

  Therefore, we call Z'_{Bethe} a (local) Bethe partition function only if we are at a fixed point of the SPA.

Bethe Partition Function

- Basically, we can evaluate the expresion for Z'_{Bethe} at any iteration of the SPA.

- Factor graph without cycles:

  We have $Z'_{\mathrm{Bethe}} = Z$ only at a fixed point of the SPA.

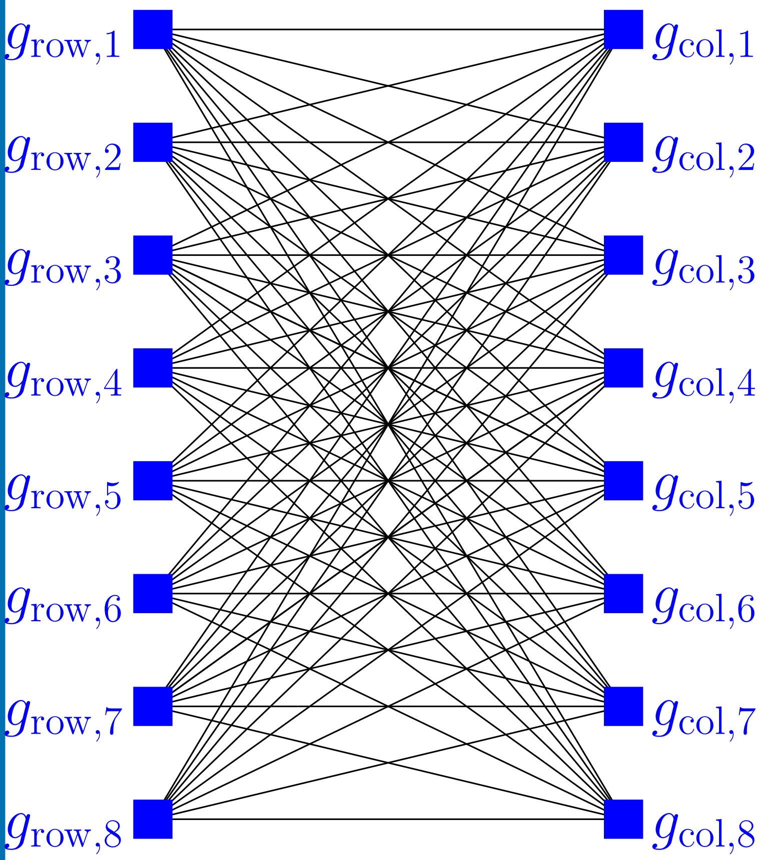
- Factor graph with cycles:

  Therefore, we call Z'_{Bethe} a (local) Bethe partition function only if we are at a fixed point of the SPA.

- Factor graph with cycles: the SPA can have multiple fixed points. We define the Bethe partition function to be

$$Z_{\mathrm{Bethe}} \triangleq \max_{\text{fixed points of SPA}} Z'_{\mathrm{Bethe}}.$$

Graphical Model for Permanent



(function nodes are suitably defined based on $\boldsymbol{\theta}$)

(variable nodes have been omitted)

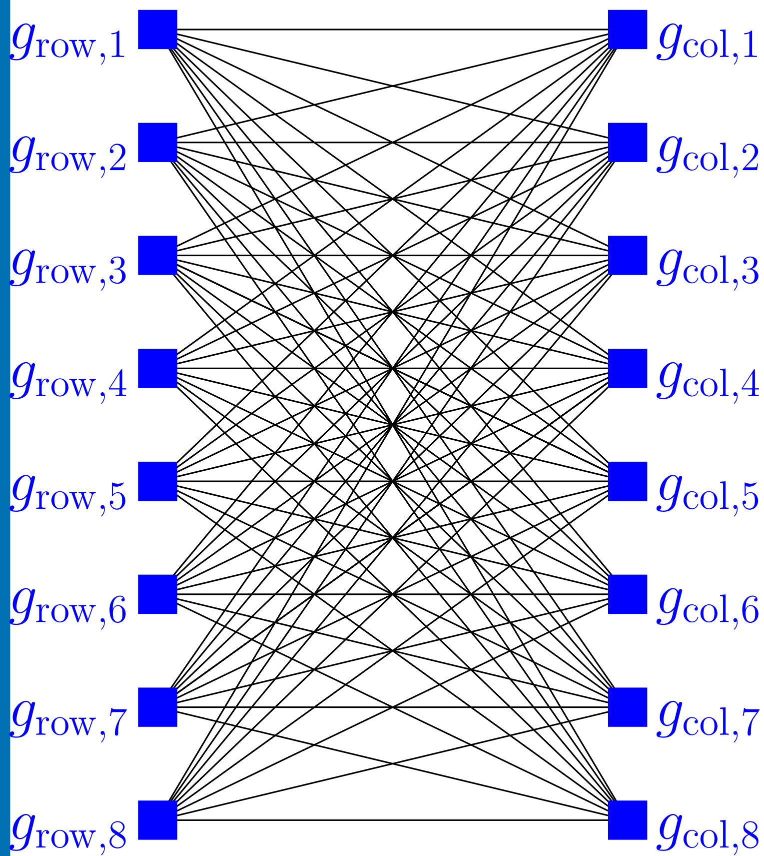
Global function:

$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\text{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\text{row},i}(a_{i,1}, \ldots, x_{i,8})$$

Permanent:

$$\text{perm}(\boldsymbol{\theta}) = Z = \sum_{a_{1,1},\ldots,a_{8,8}} g(a_{1,1}, \ldots, a_{8,8})$$

Graphical Model for Permanent



(function nodes are suitably defined based on $\boldsymbol{\theta}$)

(variable nodes have been omitted)

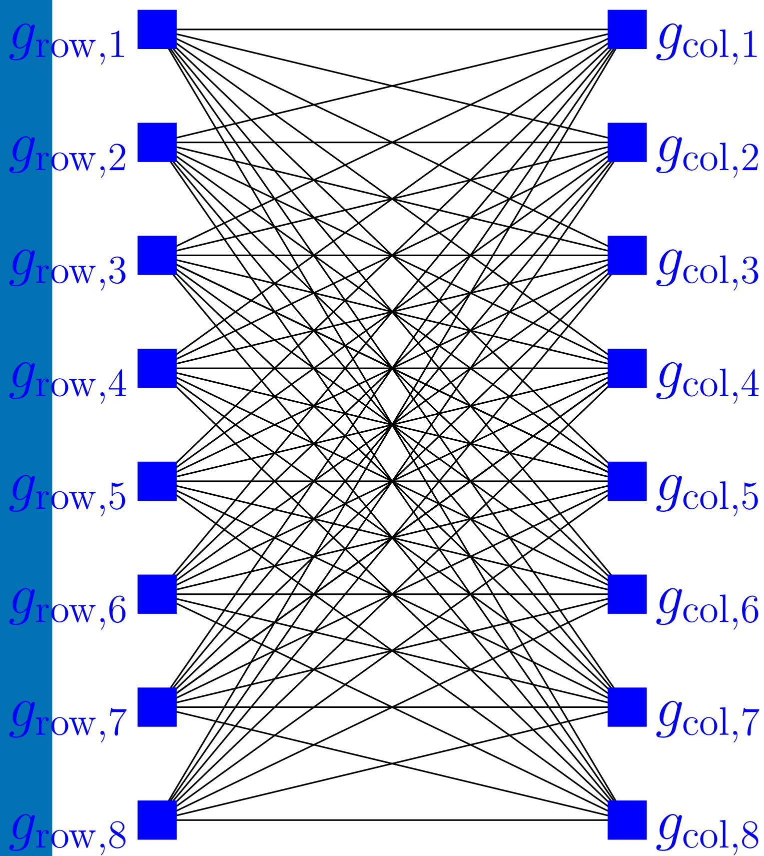
Global function:

$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\text{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\text{row},i}(a_{i,1}, \ldots, x_{i,8})$$

Bethe Permanent:

$$\text{perm}_{\text{B}}(\boldsymbol{\theta}) \triangleq Z_{\text{Bethe}}$$

Graphical Model for Permanent



Global function:

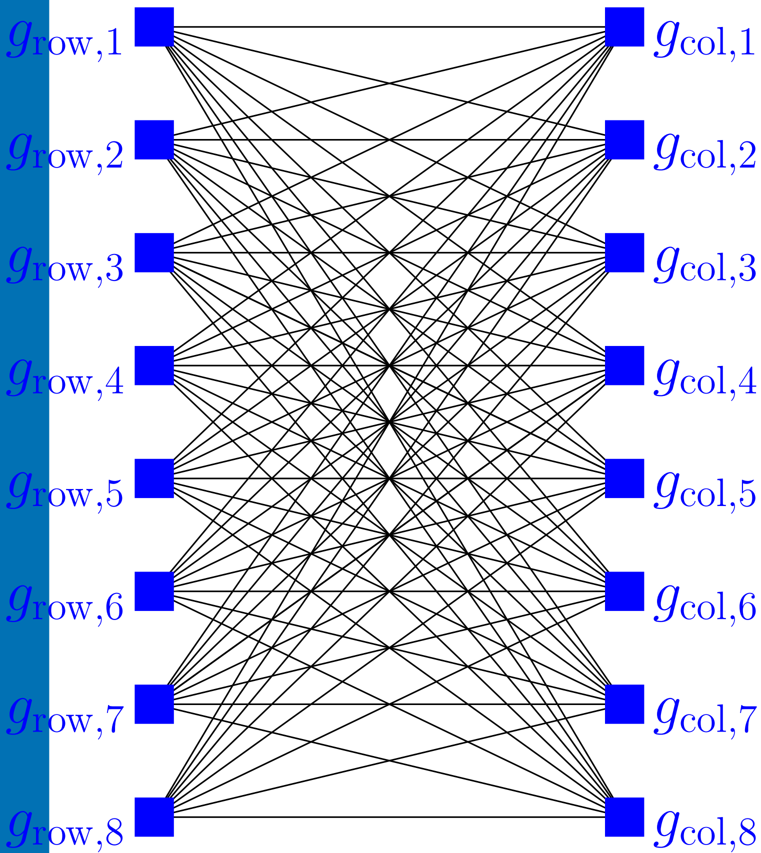
$$g(a_{1,1}, \ldots, a_{8,8}) = \prod_j g_{\mathrm{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \prod_i g_{\mathrm{row},i}(a_{i,1}, \ldots, x_{i,8})$$

Bethe Permanent:

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \triangleq Z_{\mathrm{Bethe}}$$

However, the SPA is a locally operating algorithm and so has its limitations in the conclusions that it can reach.

Graphical Model for Permanent



Global function:

$$\begin{aligned} g(a_{1,1}, \ldots, a_{8,8}) &= \prod_j g_{\mathrm{col},j}(a_{1,j}, \ldots, a_{8,j}) \times \\ &\quad \prod_i g_{\mathrm{row},i}(a_{i,1}, \ldots, x_{i,8}) \end{aligned}$$

Bethe Permanent:

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \triangleq Z_{\mathrm{Bethe}}$$

This locality of the SPA turns out to be well-captured by so-called finite graph covers, especially at fixed points of the SPA.

A combinatorial interpretation of the Bethe permanent

Combinatorial Characterization of the Permanent

Combinatorial Characterization of the Permanent

Consider the matrix

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_{1,1} & \theta_{1,2} \\ \theta_{2,1} & \theta_{2,2} \end{pmatrix} \qquad \text{with} \qquad \operatorname{perm}(\boldsymbol{\theta}) = \theta_{1,1}\theta_{2,2} + \theta_{2,1}\theta_{1,2}.$$

Combinatorial Characterization of the Permanent

Consider the matrix

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_{1,1} & \theta_{1,2} \\ \theta_{2,1} & \theta_{2,2} \end{pmatrix} \qquad \text{with} \qquad \mathrm{perm}(\boldsymbol{\theta}) = \theta_{1,1}\theta_{2,2} + \theta_{2,1}\theta_{1,2}.$$

In particular,

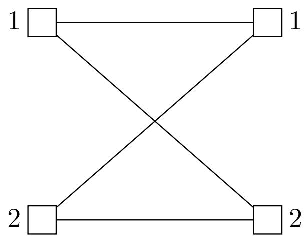
$$\boldsymbol{\theta} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \qquad \text{with} \qquad \mathrm{perm}(\boldsymbol{\theta}) = 1 \cdot 1 + 1 \cdot 1 = 2.$$

Combinatorial Characterization of the Permanent

Recall that the permanent of a zero/one matrix like

$$\boldsymbol{\theta} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

equals the number of perfect matchings in the following bipartite graph:

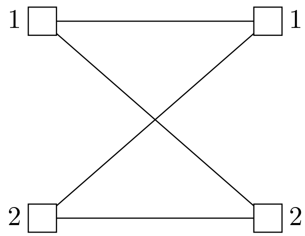


Combinatorial Characterization of the Permanent

Recall that the permanent of a zero/one matrix like

$$\boldsymbol{\theta} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

equals the number of perfect matchings in the following bipartite graph:



Namely,

A Combinatorial Interpretation of the Bethe Permanent

- Consider the non-negative matrix $\boldsymbol{\theta}$ of size $n \times n$.
- Let $\mathcal{P}_{M \times M}$ be the set of all permutation matrices of size $M \times M$.
- For every positive integer M, we define Ψ_M be the set

$$\Psi_M \triangleq \left\{ \mathbf{P} = \left\{ \mathbf{P}^{(i,j)} \right\}_{(i,j) \in [n]^2} \,\middle|\, \mathbf{P}^{(i,j)} \in \mathcal{P}_{M \times M} \right\}.$$

- For $\mathbf{P} \in \Psi_M$ we define the $\mathbf{P}$-lifting of $\boldsymbol{\theta}$ to be the following $(nM) \times (nM)$ matrix

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_{1,1} & \cdots & \theta_{1,n} \\ \vdots & & \vdots \\ \theta_{n,1} & \cdots & \theta_{n,n} \end{pmatrix} \quad \underset{\text{of } \boldsymbol{\theta}}{\xrightarrow{\mathbf{P}\text{-lifting}}} \quad \boldsymbol{\theta}^{\uparrow \mathbf{P}} \triangleq \begin{pmatrix} \theta_{1,1}\mathbf{P}^{(1,1)} & \cdots & \theta_{1,n}\mathbf{P}^{(1,n)} \\ \vdots & & \vdots \\ \theta_{n,1}\mathbf{P}^{(n,1)} & \cdots & \theta_{n,n}\mathbf{P}^{(n,n)} \end{pmatrix}.$$

Degree-M Bethe Permanent

Definition: For any positive integer M, we define the degree-M Bethe permanent of $\boldsymbol{\theta}$ to be

$$\mathrm{perm}_{\mathrm{B},M}(\boldsymbol{\theta}) \triangleq \sqrt[M]{\left\langle \mathrm{perm}\left(\boldsymbol{\theta}^{\uparrow \mathbf{P}}\right) \right\rangle_{\mathbf{P} \in \Psi_M}}.$$

Theorem:

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) = \limsup_{M \to \infty} \ \mathrm{perm}_{\mathrm{B},M}(\boldsymbol{\theta}).$$

Special Case: Deg.-M Bethe Permanent for $n = 2$

We want to obtain some appreciation why the Bethe permanent of $\boldsymbol{\theta}$ is close to the permanent of $\boldsymbol{\theta}$, and where the differences are.

Special Case: Deg.-M Bethe Permanent for $n = 2$

We want to obtain some appreciation why the Bethe permanent of $\boldsymbol{\theta}$ is close to the permanent of $\boldsymbol{\theta}$, and where the differences are.

As before, consider the matrix

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_{1,1} & \theta_{1,2} \\ \theta_{2,1} & \theta_{2,2} \end{pmatrix} \qquad \text{with} \qquad \mathrm{perm}(\boldsymbol{\theta}) = \theta_{1,1}\theta_{2,2} + \theta_{2,1}\theta_{1,2}.$$

Special Case: Deg.-M Bethe Permanent for $n = 2$

We want to obtain some appreciation why the Bethe permanent of $\boldsymbol{\theta}$ is close to the permanent of $\boldsymbol{\theta}$, and where the differences are.

As before, consider the matrix

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_{1,1} & \theta_{1,2} \\ \theta_{2,1} & \theta_{2,2} \end{pmatrix} \qquad \text{with} \qquad \operatorname{perm}(\boldsymbol{\theta}) = \theta_{1,1}\theta_{2,2} + \theta_{2,1}\theta_{1,2}.$$

In particular,

$$\boldsymbol{\theta} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \qquad \text{with} \qquad \operatorname{perm}(\boldsymbol{\theta}) = 1 \cdot 1 + 1 \cdot 1 = 2.$$

Special Case: Deg.-M Bethe Permanent for $n = 2$

For this $\boldsymbol{\theta}$, a $\mathbf{P}$-lifting looks like

$$\boldsymbol{\theta}^{\uparrow \mathbf{P}} = \begin{pmatrix} 1 \cdot \mathbf{P}_{1,1} & 1 \cdot \mathbf{P}_{1,2} \\ 1 \cdot \mathbf{P}_{2,1} & 1 \cdot \mathbf{P}_{2,2} \end{pmatrix} = \begin{pmatrix} \mathbf{P}_{1,1} & \mathbf{P}_{1,2} \\ \mathbf{P}_{2,1} & \mathbf{P}_{2,2} \end{pmatrix}.$$

Special Case: Deg.-M Bethe Permanent for $n = 2$

For this $\boldsymbol{\theta}$, a $\mathbf{P}$-lifting looks like

$$\boldsymbol{\theta}^{\uparrow \mathbf{P}} = \begin{pmatrix} 1 \cdot \mathbf{P}_{1,1} & 1 \cdot \mathbf{P}_{1,2} \\ 1 \cdot \mathbf{P}_{2,1} & 1 \cdot \mathbf{P}_{2,2} \end{pmatrix} = \begin{pmatrix} \mathbf{P}_{1,1} & \mathbf{P}_{1,2} \\ \mathbf{P}_{2,1} & \mathbf{P}_{2,2} \end{pmatrix}.$$

Applying some row and column permutations, we obtain

$$\operatorname{perm}\left(\boldsymbol{\theta}^{\uparrow \mathbf{P}}\right) = \operatorname{perm}\begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{P}_{2,1}^{-1}\mathbf{P}_{2,2}\mathbf{P}_{1,2}^{-1}\mathbf{P}_{1,1} \end{pmatrix}.$$

Special Case: Deg.-M Bethe Permanent for $n = 2$

For this $\boldsymbol{\theta}$, a $\mathbf{P}$-lifting looks like

$$\boldsymbol{\theta}^{\uparrow \mathbf{P}} = \begin{pmatrix} 1 \cdot \mathbf{P}_{1,1} & 1 \cdot \mathbf{P}_{1,2} \\ 1 \cdot \mathbf{P}_{2,1} & 1 \cdot \mathbf{P}_{2,2} \end{pmatrix} = \begin{pmatrix} \mathbf{P}_{1,1} & \mathbf{P}_{1,2} \\ \mathbf{P}_{2,1} & \mathbf{P}_{2,2} \end{pmatrix}.$$

Applying some row and column permutations, we obtain

$$\mathrm{perm}\left(\boldsymbol{\theta}^{\uparrow \mathbf{P}}\right) = \mathrm{perm}\begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{P}_{2,1}^{-1}\mathbf{P}_{2,2}\mathbf{P}_{1,2}^{-1}\mathbf{P}_{1,1} \end{pmatrix}.$$

Therefore,

$$\mathrm{perm}_{\mathrm{B},M}(\boldsymbol{\theta}) \triangleq \sqrt[M]{\left\langle \mathrm{perm}\begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{P}'_{2,2} \end{pmatrix} \right\rangle_{\mathbf{P}'_{2,2} \in \mathcal{P}_{M \times M}}}.$$

Special Case: Degree-2 Bethe Permanent for $n = 2$

For $M = 2$ we have

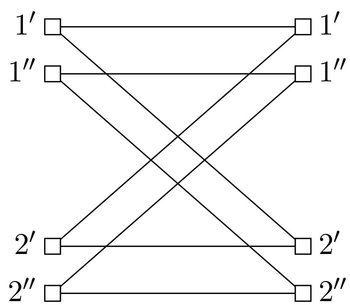
$$\mathrm{perm}_{\mathrm{B},2}(\boldsymbol{\theta}) \triangleq \sqrt[2]{\left\langle \mathrm{perm}\begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{P}'_{2,2} \end{pmatrix} \right\rangle_{\mathbf{P}'_{2,2} \in \mathcal{P}_{2\times 2}}}$$

Special Case: Degree-2 Bethe Permanent for $n = 2$

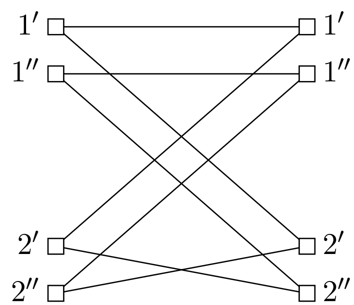
For $M = 2$ we have

$$\mathrm{perm}_{\mathrm{B},2}(\boldsymbol{\theta}) \triangleq \sqrt[2]{\left\langle \mathrm{perm}\begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{P}'_{2,2} \end{pmatrix} \right\rangle_{\mathbf{P}'_{2,2} \in \mathcal{P}_{2\times 2}}}$$

corresponds to computing the average number of perfect matchings in the following 2-covers (and taking the 2nd root):



4



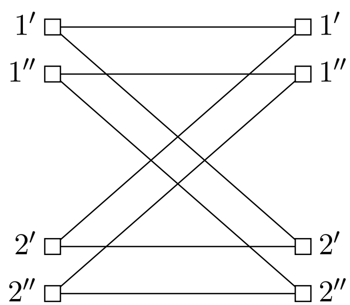
2

Special Case:
Degree-2 Bethe Permanent for $n = 2$

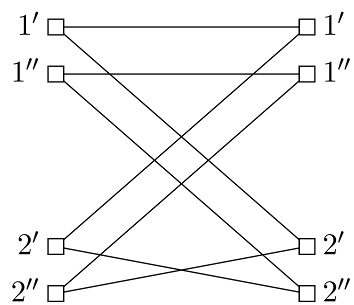
For $M = 2$ we have

$$\mathrm{perm}_{\mathrm{B},2}(\boldsymbol{\theta}) = \sqrt[2]{\frac{1}{2!} \cdot (4 + 2)}$$

corresponds to computing the average number of perfect matchings in the following 2-covers (and taking the 2nd root):



4



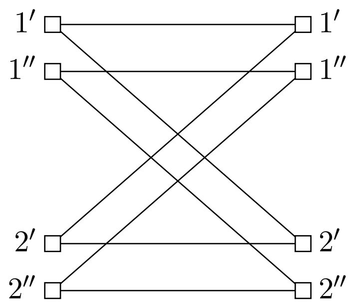
2

Special Case:
Degree-2 Bethe Permanent for $n = 2$

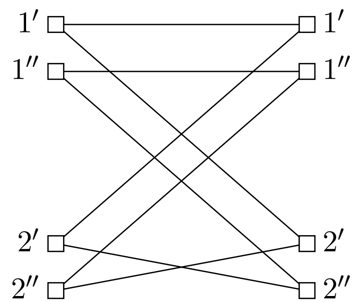
For $M = 2$ we have

$$
\begin{aligned}
\mathrm{perm}_{\mathrm{B},2}(\boldsymbol{\theta}) &= \sqrt[2]{\frac{1}{2!} \cdot (4+2)} \\
&= \sqrt[3]{\frac{1}{2!} \cdot 6} = \sqrt[2]{3} \approx 1.732
\end{aligned}
$$

corresponds to computing the average number of perfect matchings in the following 2-covers (and taking the 2nd root):



4



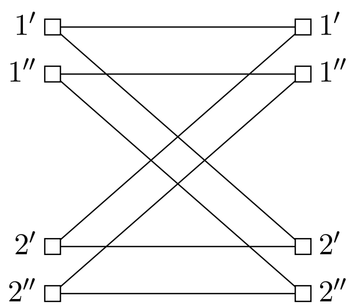
2

Special Case: Degree-2 Bethe Permanent for $n = 2$

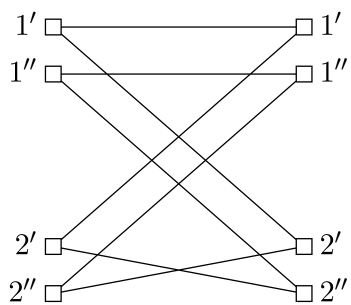
For $M = 2$ we have

$$\mathrm{perm}_{\mathrm{B},2}(\boldsymbol{\theta}) = \sqrt[2]{\frac{1}{2!} \cdot (4+2)}$$
$$= \sqrt[3]{\frac{1}{2!} \cdot 6} = \sqrt[2]{3} \approx 1.732 \; < \; \sqrt[2]{4} = 2 = \mathrm{perm}(\boldsymbol{\theta})$$

corresponds to computing the average number of perfect matchings in the following 2-covers (and taking the 2nd root):



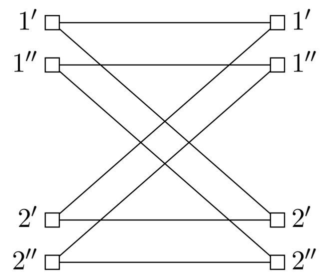
4



2

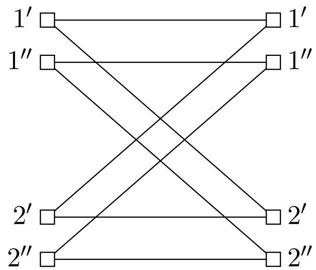
Special Case: Degree-2 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph

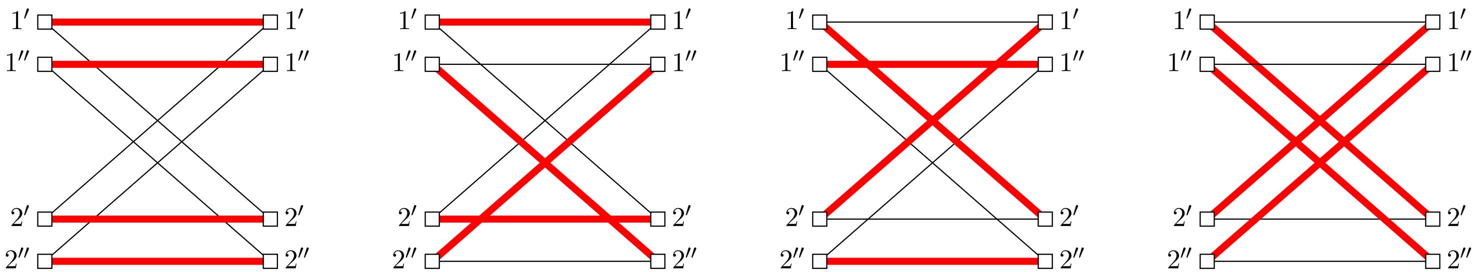


Special Case: Degree-2 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph

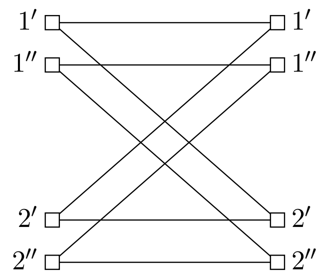


For this graph, the perfect matchings are

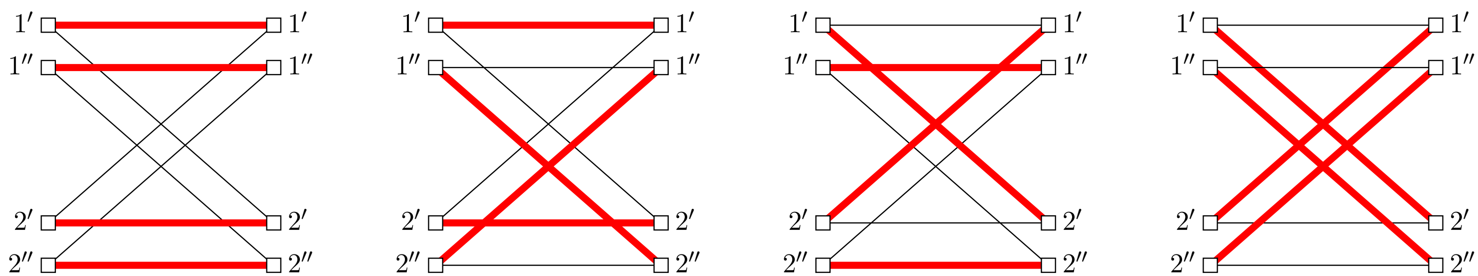


Special Case: Degree-2 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph



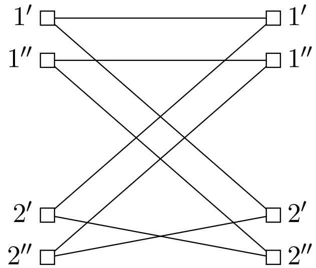
For this graph, the perfect matchings are



Because this double cover consists of two ***independent copies*** *of the base graph, the number of perfect matchings is* $2^2 = 4$.

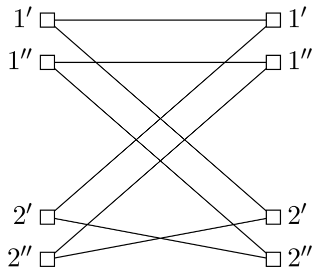
Special Case: Degree-2 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph

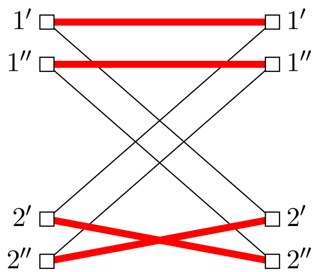


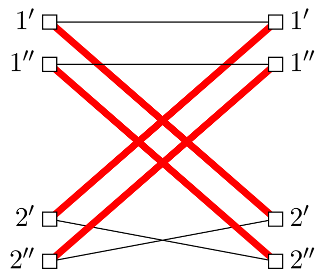
Special Case: Degree-2 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph



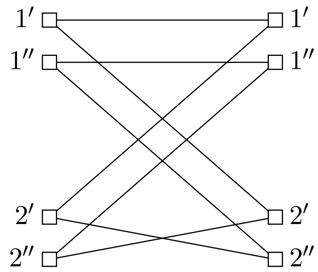
For this graph, the perfect matchings are



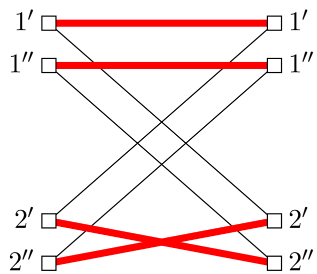


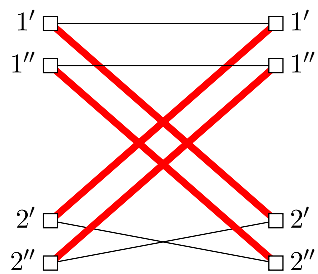
Special Case:
Degree-2 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph



For this graph, the perfect matchings are





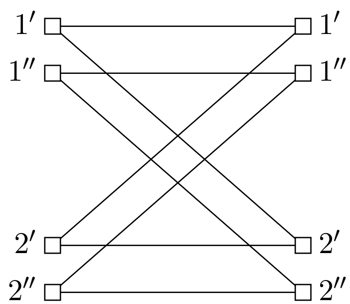
The ***coupling of the cycles*** *causes this graph to have fewer than 2^2 perfect matchings!*

Special Case:
Degree-2 Bethe Permanent for $n = 2$

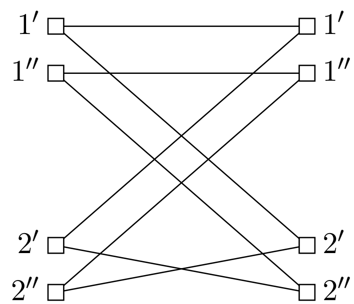
On the other hand, for $M = 2$ we have

$$\begin{aligned}\mathrm{perm}_{\mathrm{B},2}(\boldsymbol{\theta}) &= \sqrt[2]{\frac{1}{2!} \cdot (4+2)} \\ &= \sqrt[3]{\frac{1}{2!} \cdot 6} = \sqrt[2]{3} \approx 1.732 \; < \; \sqrt[2]{4} = 2 = \mathrm{perm}(\boldsymbol{\theta})\end{aligned}$$

corresponds to computing the average number of perfect matchings in the following 2-covers (and taking the 2nd root):



4



2

Special Case: Degree-3 Bethe Permanent for $n = 2$

On the other hand, for $M = 3$ we have

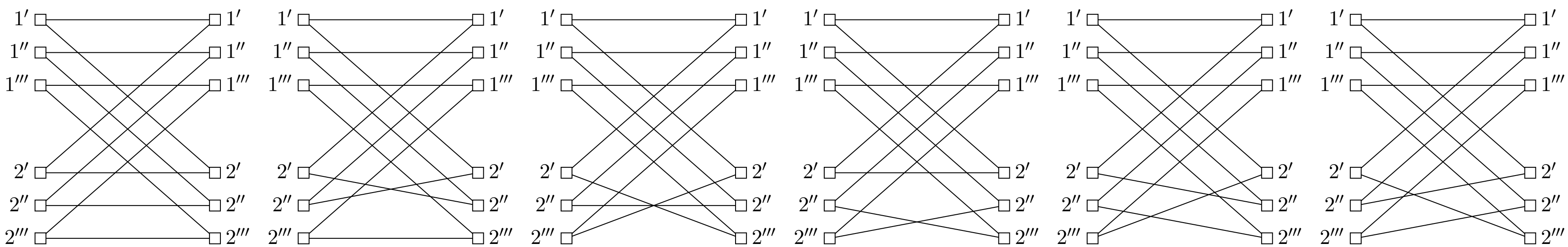
$$\mathrm{perm}_{\mathrm{B},3}(\boldsymbol{\theta}) \triangleq \sqrt[3]{\left\langle \mathrm{perm}\begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{P}'_{2,2} \end{pmatrix} \right\rangle_{\mathbf{P}'_{2,2} \in \mathcal{P}_{3\times 3}}}$$

Special Case: Degree-3 Bethe Permanent for $n = 2$

On the other hand, for $M = 3$ we have

$$\mathrm{perm}_{\mathrm{B},3}(\boldsymbol{\theta}) \triangleq \sqrt[3]{\left\langle \mathrm{perm}\begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{P}'_{2,2} \end{pmatrix} \right\rangle_{\mathbf{P}'_{2,2} \in \mathcal{P}_{3\times 3}}}$$

corresponds to computing the average number of perfect matchings in the following 3-covers (and taking the 3rd root):



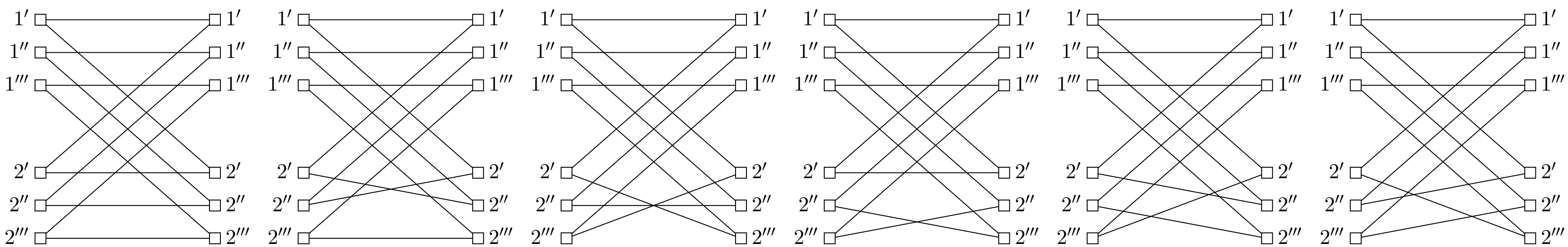
8 4 4 4 2 2

Special Case:
Degree-3 Bethe Permanent for $n = 2$

On the other hand, for $M = 3$ we have

$$\mathrm{perm}_{\mathrm{B},3}(\boldsymbol{\theta}) = \sqrt[3]{\frac{1}{3!} \cdot (8 + 4 + 4 + 4 + 2 + 2)}$$

corresponds to computing the average number of perfect matchings in the following 3-covers (and taking the 3rd root):



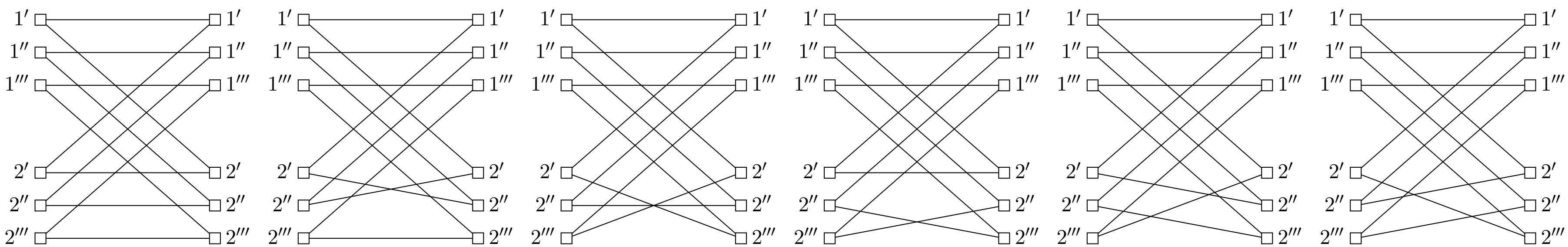
8 4 4 4 2 2

Special Case: Degree-3 Bethe Permanent for $n = 2$

On the other hand, for $M = 3$ we have

$$\begin{aligned}\mathrm{perm}_{\mathrm{B},3}(\boldsymbol{\theta}) &= \sqrt[3]{\frac{1}{3!} \cdot (8+4+4+4+2+2)} \\ &= \sqrt[3]{\frac{1}{3!} \cdot 24} = \sqrt[3]{4} \approx 1.587\end{aligned}$$

corresponds to computing the average number of perfect matchings in the following 3-covers (and taking the 3rd root):



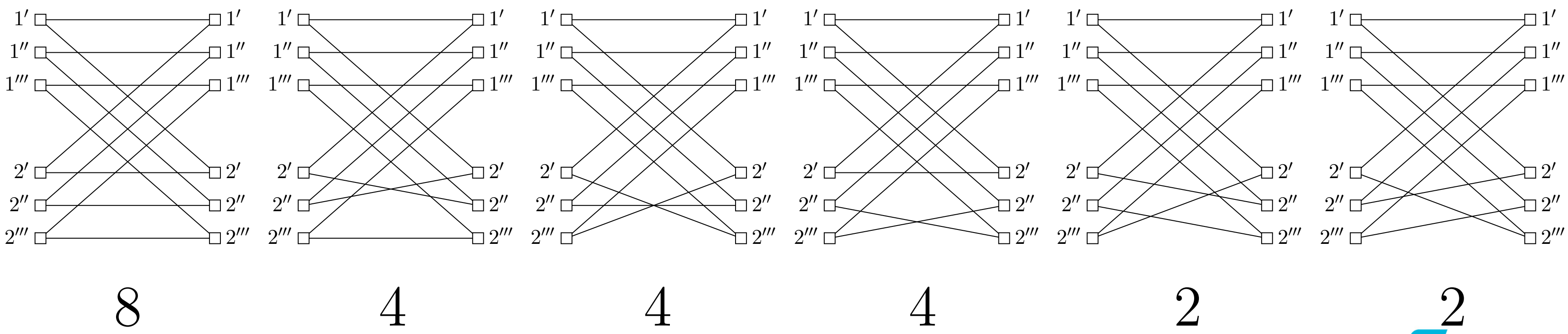
8 4 4 4 2 2

Special Case: Degree-3 Bethe Permanent for $n = 2$

On the other hand, for $M = 3$ we have

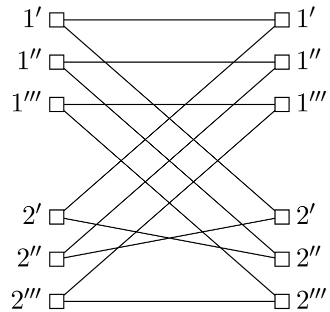
$$\begin{aligned}\mathrm{perm}_{\mathrm{B},3}(\boldsymbol{\theta}) &= \sqrt[3]{\frac{1}{3!}\cdot(8+4+4+4+2+2)} \\ &= \sqrt[3]{\frac{1}{3!}\cdot 24} = \sqrt[3]{4} \approx 1.587 \; < \; \sqrt[3]{8} = 2 = \mathrm{perm}(\boldsymbol{\theta})\end{aligned}$$

corresponds to computing the average number of perfect matchings in the following 3-covers (and taking the 3rd root):



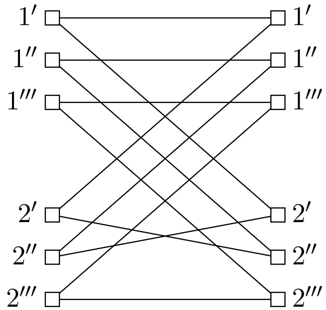
Special Case: Degree-3 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph

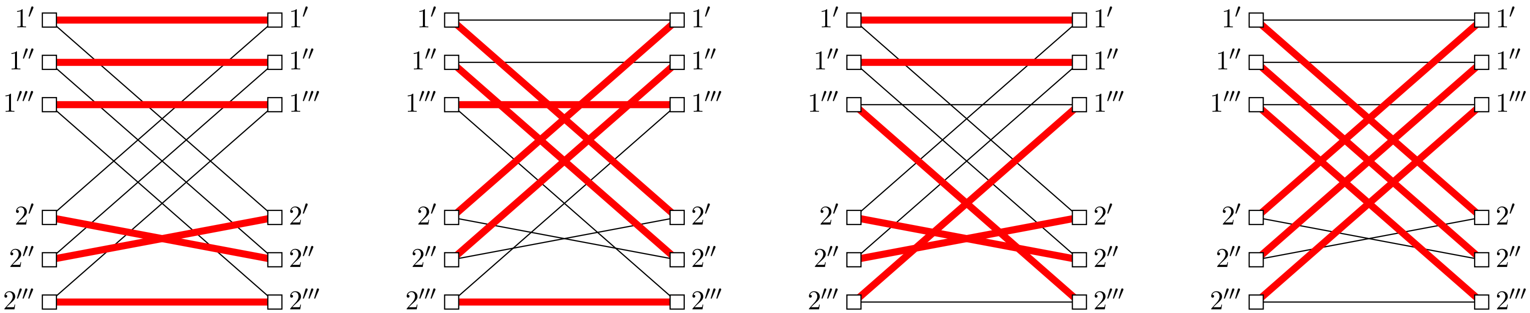


Special Case: Degree-3 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph

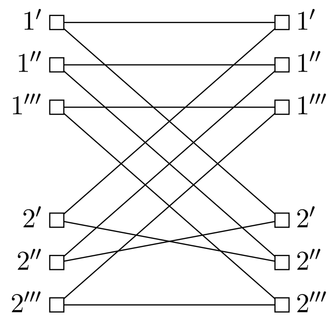


For this graph, the perfect matchings are

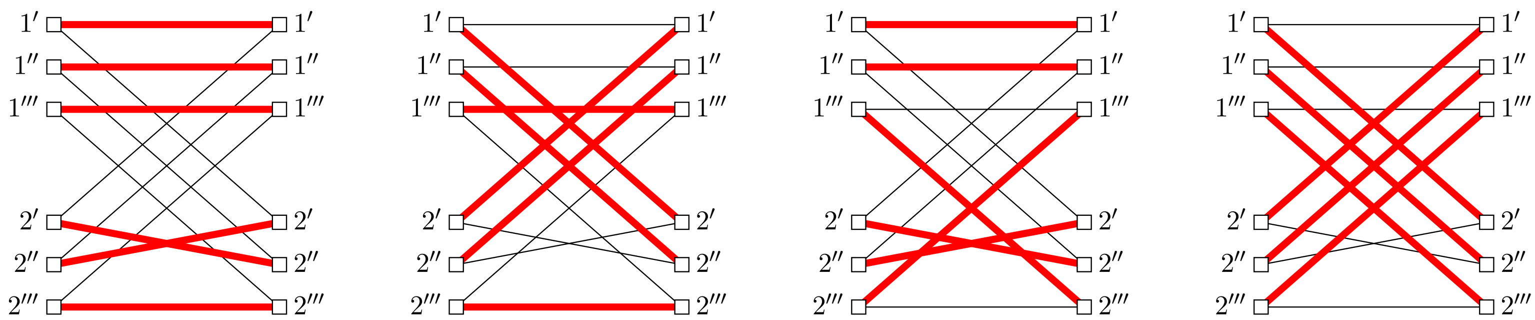


Special Case: Degree-3 Bethe Permanent for $n = 2$

Let us have a closer look at the perfect matchings in the graph



For this graph, the perfect matchings are



The ***coupling of the cycles*** *causes this graph to have fewer than* 2^3 *perfect matchings!*

Special Case: Deg.-M Bethe Permanent for $n = 2$

For general M we obtain

$$\mathrm{perm}_{\mathrm{B},M}(\boldsymbol{\theta}) = \sqrt[M]{\zeta_{S_M}} = \sqrt[M]{M+1},$$

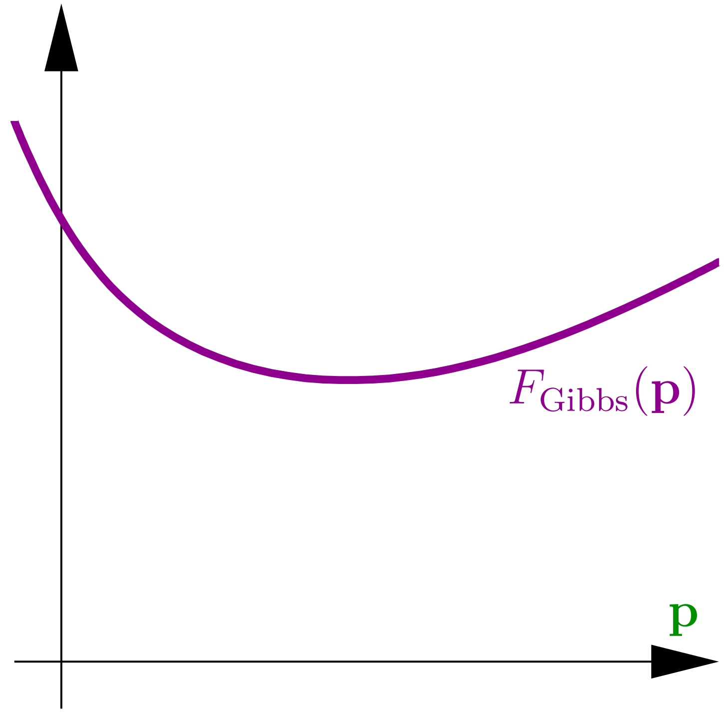
ζ_{S_M}: cycle index of the symmetric group over M elements.

The Gibbs free energy function

Gibbs Free Energy Function

The Gibbs free energy function

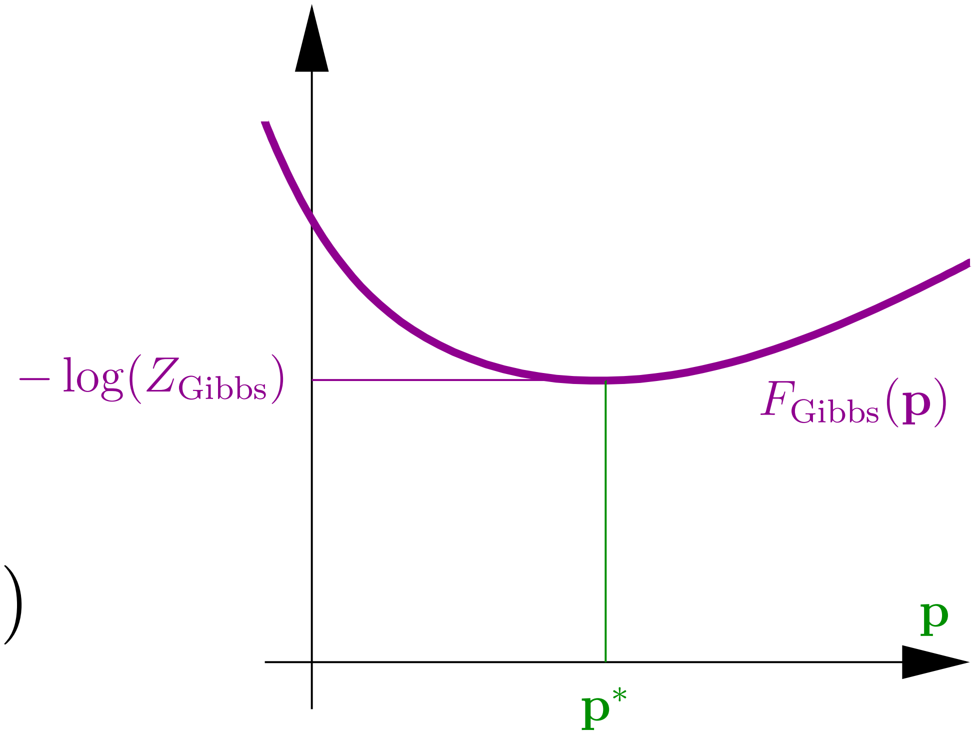
$$F_{\mathrm{Gibbs}}(\mathbf{p}) \triangleq -\sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log\big(g(\mathbf{a})\big) + \sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log(p_{\mathbf{a}}).$$



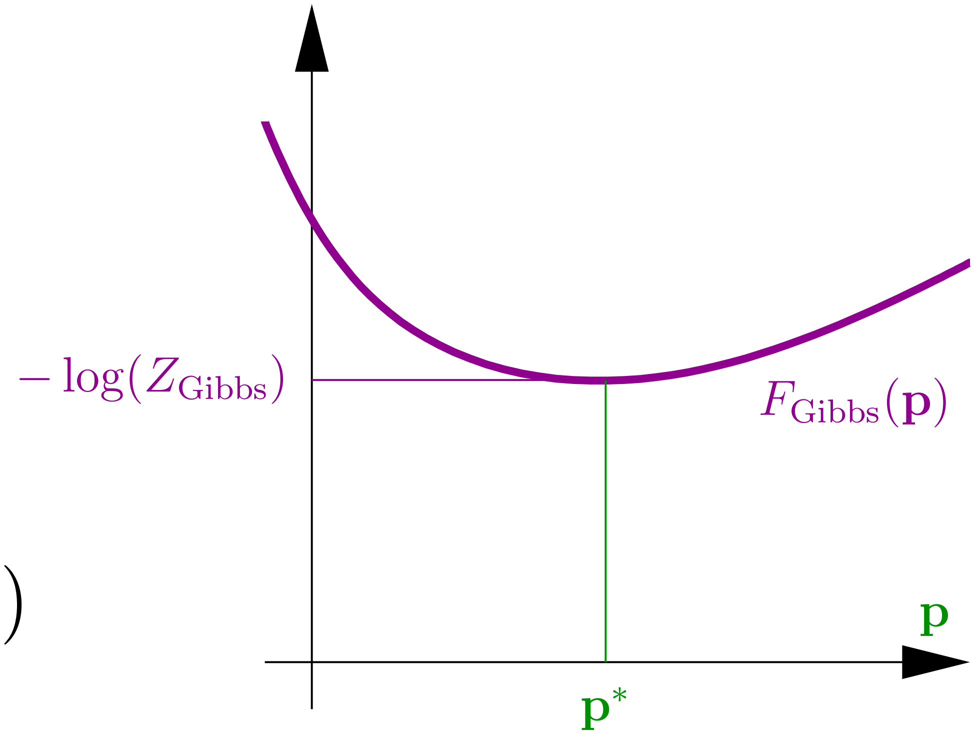
Gibbs Free Energy Function

The Gibbs free energy function

$$F_{\mathrm{Gibbs}}(\mathbf{p}) \triangleq -\sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log\big(g(\mathbf{a})\big) + \sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log(p_{\mathbf{a}}).$$



Gibbs Free Energy Function



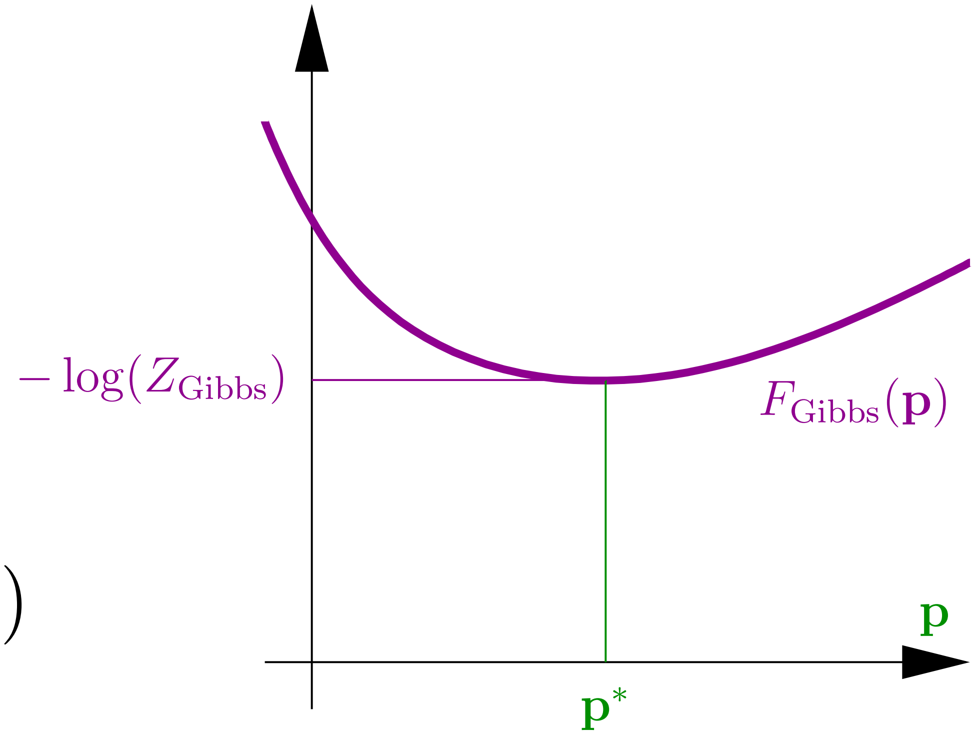
The Gibbs free energy function

$$F_{\mathrm{Gibbs}}(\mathbf{p}) \triangleq -\sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log\left(g(\mathbf{a})\right) + \sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log(p_{\mathbf{a}}).$$

is defined such that its minimal value is related to the partition function:

$$Z = \exp\left(-\min_{\mathbf{p}} \ F_{\mathrm{Gibbs}}(\mathbf{p})\right).$$

Gibbs Free Energy Function



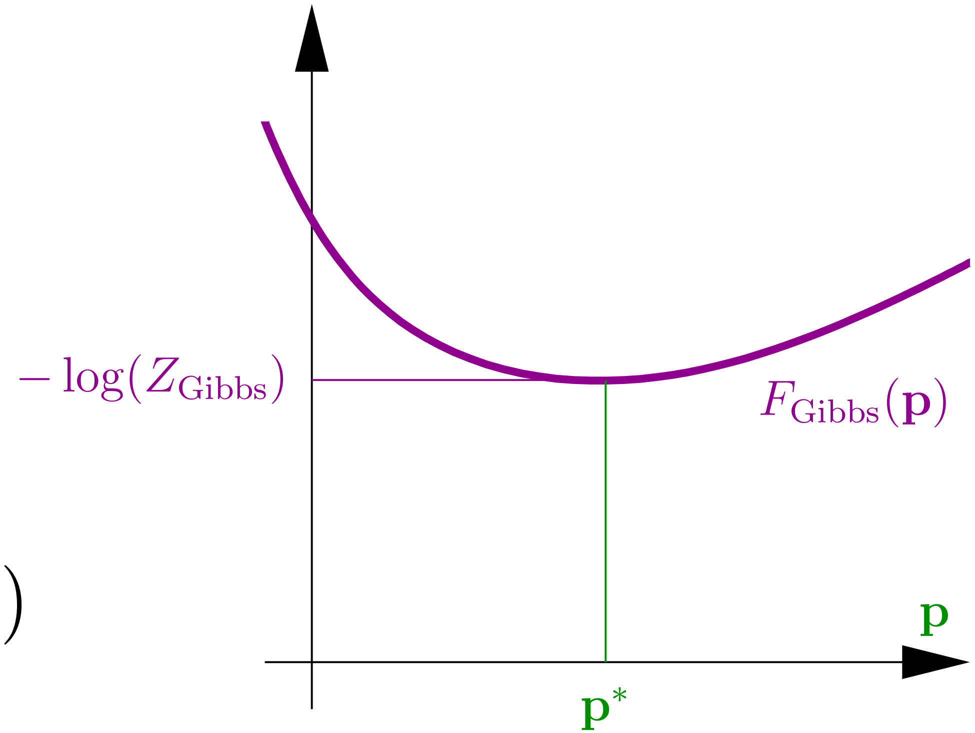
The Gibbs free energy function

$$F_{\mathrm{Gibbs}}(\mathbf{p}) \triangleq -\sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log\left(g(\mathbf{a})\right) + \sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log(p_{\mathbf{a}}).$$

is defined such that its minimal value is related to the partition function:

$$\mathrm{perm}(\boldsymbol{\theta}) = Z = \exp\left(-\min_{\mathbf{p}} \; F_{\mathrm{Gibbs}}(\mathbf{p})\right).$$

Gibbs Free Energy Function



The Gibbs free energy function

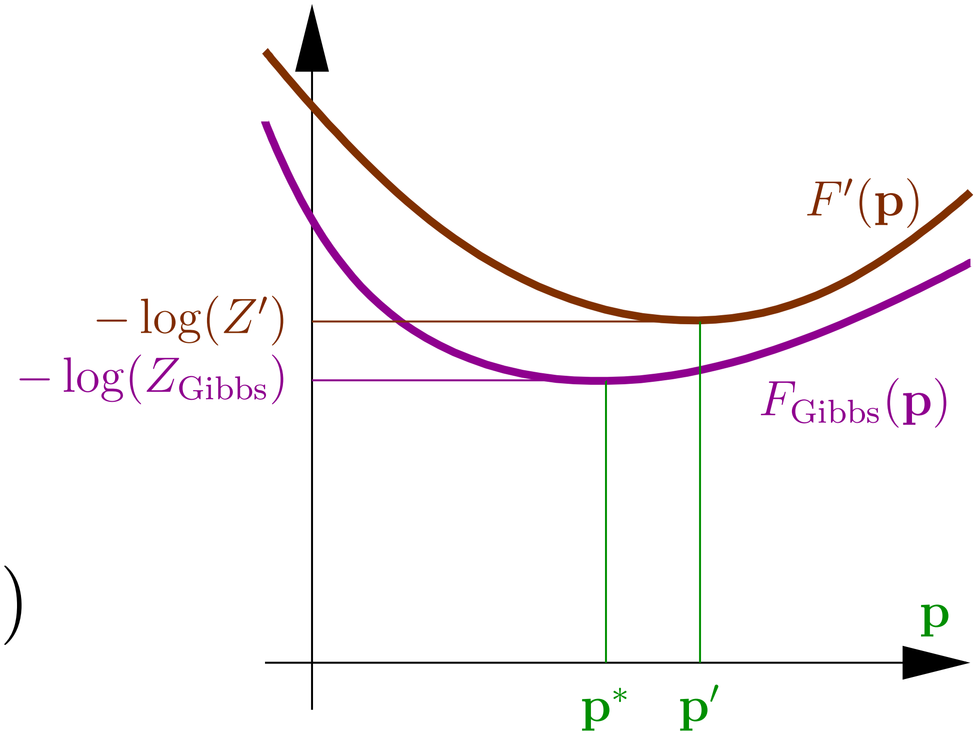
$$F_{\text{Gibbs}}(\mathbf{p}) \triangleq -\sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log\big(g(\mathbf{a})\big) + \sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log(p_{\mathbf{a}}).$$

is defined such that its minimal value is related to the partition function:

$$\operatorname{perm}(\boldsymbol{\theta}) = Z = \exp\left(-\min_{\mathbf{p}} \; F_{\text{Gibbs}}(\mathbf{p})\right).$$

Nice, but it does not yield any computational savings by itself.

Gibbs Free Energy Function



The Gibbs free energy function

$$F_{\mathrm{Gibbs}}(\mathbf{p}) \triangleq -\sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log\left(g(\mathbf{a})\right) + \sum_{\mathbf{a}} p_{\mathbf{a}} \cdot \log(p_{\mathbf{a}}).$$

is defined such that its minimal value is related to the partition function:

$$\mathrm{perm}(\boldsymbol{\theta}) = Z = \exp\left(-\min_{\mathbf{p}} F_{\mathrm{Gibbs}}(\mathbf{p})\right).$$

But it suggests other optimization schemes.

The Bethe approximation

Bethe Approximation

The Bethe approximation to the Gibbs free energy function yields such an alternative optimization scheme.

Bethe Approximation

The Bethe approximation to the Gibbs free energy function yields such an alternative optimization scheme.

This approximation is interesting because of the following theorem:

Theorem (Yedidia/Freeman/Weiss, 2000):
Fixed points of the sum-product algorithm (SPA) correspond to stationary points of the Bethe free energy function.

Bethe Approximation

The Bethe approximation to the Gibbs free energy function yields such an alternative optimization scheme.

This approximation is interesting because of the following theorem:

Theorem (Yedidia/Freeman/Weiss, 2000):
Fixed points of the sum-product algorithm (SPA) correspond to stationary points of the Bethe free energy function.

Definition: We define the Bethe permanent of $\boldsymbol{\theta}$ to be

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) = Z_{\mathrm{Bethe}} = \exp\left(- \min_{\boldsymbol{\beta}} \; F_{\mathrm{Bethe}}(\boldsymbol{\beta}) \right) .$$

Bethe Approximation

However, in general, this approach of replacing the Gibbs free energy by the Bethe free energy comes with very few guarantees:

Bethe Approximation

However, in general, this approach of replacing the Gibbs free energy by the Bethe free energy comes with very few guarantees:

- The Bethe free energy function might have multiple local minima.

Bethe Approximation

However, in general, this approach of replacing the Gibbs free energy by the Bethe free energy comes with very few guarantees:

- The Bethe free energy function might have multiple local minima.
- It is unclear how close the (global) minimum of the Bethe free energy is to the minimum of the Gibbs free energy.

Bethe Approximation

However, in general, this approach of replacing the Gibbs free energy by the Bethe free energy comes with very few guarantees:

- The Bethe free energy function might have multiple local minima.
- It is unclear how close the (global) minimum of the Bethe free energy is to the minimum of the Gibbs free energy.
- It is unclear if the sum-product algorithm converges (even to a local minimum of the Bethe free energy).

Bethe Approximation

Luckily, in the case of the permanent approximation problem, the above-mentioned normal factor graph $\mathsf{N}(\boldsymbol{\theta})$ is such that the Bethe free energy function is very well behaved. In particular, one can show that:

Bethe Approximation

Luckily, in the case of the permanent approximation problem, the above-mentioned normal factor graph $\mathsf{N}(\boldsymbol{\theta})$ is such that the Bethe free energy function is very well behaved. In particular, one can show that:

- The Bethe free energy function (for a suitable parametrization) is convex and therefore has no local minima [V., 2010, 2011].

Bethe Approximation

Luckily, in the case of the permanent approximation problem, the above-mentioned normal factor graph $\mathsf{N}(\boldsymbol{\theta})$ is such that the Bethe free energy function is very well behaved. In particular, one can show that:

- The Bethe free energy function (for a suitable parametrization) is convex and therefore has no local minima [V., 2010, 2011].
- The minimum of the Bethe free energy is quite close to the minimum of the Gibbs free energy. *(More details later.)*

Bethe Approximation

Luckily, in the case of the permanent approximation problem, the above-mentioned normal factor graph $\mathsf{N}(\boldsymbol{\theta})$ is such that the Bethe free energy function is very well behaved. In particular, one can show that:

- The Bethe free energy function (for a suitable parametrization) is convex and therefore has no local minima [V., 2010, 2011].
- The minimum of the Bethe free energy is quite close to the minimum of the Gibbs free energy. *(More details later.)*
- The sum-product algorithm converges to the minimum of the Bethe free energy. *(More details later.)*

Relationship between Permanent and Bethe Permanent

Theorem (Gurvits, 2011) $\quad$ **Conjecture (Gurvits, 2011)**

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \;\underset{\text{Theorem (Gurvits, 2011)}}{\leq}\; \mathrm{perm}(\boldsymbol{\theta}) \;\underset{\text{Conjecture (Gurvits, 2011)}}{\leq}\; \sqrt{2}^{\,n} \cdot \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta})$$

Relationship between Permanent and Bethe Permanent

$$\underset{\textbf{Theorem (Gurvits, 2011)}}{} \qquad \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \overset{\downarrow}{\leq} \mathrm{perm}(\boldsymbol{\theta}) \overset{\downarrow}{\leq} \sqrt{2}^{\,n} \cdot \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta})$$

Theorem (Gurvits, 2011) **Conjecture (Gurvits, 2011)**

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \quad \overset{\downarrow}{\leq} \quad \mathrm{perm}(\boldsymbol{\theta}) \quad \overset{\downarrow}{\leq} \quad \sqrt{2}^{\,n} \cdot \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta})$$

This can be rewritten as follows:

Theorem **Conjecture**

$$\frac{1}{n} \log \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \;\overset{\downarrow}{\leq}\; \frac{1}{n} \log \mathrm{perm}(\boldsymbol{\theta}) \;\overset{\downarrow}{\leq}\; \frac{1}{n} \log \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) + \log(\sqrt{2})$$

Relationship between Permanent and Bethe Permanent

Problem: find large classes of random matrices such that w.h.p.

Theorem (Gurvits, 2011)

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \overset{\downarrow}{\leq} \mathrm{perm}(\boldsymbol{\theta}) \leq O(\sqrt{n}) \cdot \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}).$$

Relationship between Permanent and Bethe Permanent

Problem: find large classes of random matrices such that w.h.p.

Theorem (Gurvits, 2011)

$$\mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \overset{\downarrow}{\leq} \mathrm{perm}(\boldsymbol{\theta}) \leq O(\sqrt{n}) \cdot \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}).$$

This can be rewritten as follows:

Theorem

$$\frac{1}{n} \log \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) \overset{\downarrow}{\leq} \frac{1}{n} \log \mathrm{perm}(\boldsymbol{\theta}) \leq \frac{1}{n} \log \mathrm{perm}_{\mathrm{B}}(\boldsymbol{\theta}) + O\left(\frac{1}{n} \log(n)\right)$$

Sum-Product Algorithm Convergence

Theorem: Modulo some minor technical conditions on the initial messages, the sum-product algorithm converges to the (global) minimum of the Bethe free energy function [V., 2010, 2011].

Sum-Product Algorithm Convergence

Theorem: Modulo some minor technical conditions on the initial messages, the sum-product algorithm converges to the (global) minimum of the Bethe free energy function [V., 2010, 2011].

Comment: the first part of the proof of the above theorem is very similar to the SPA convergence proof in

Bayati and Nair, "*A rigorous proof of the cavity method for counting matchings*," Allerton 2006.

Note that they consider matchings, not perfect matchings. (Although the perfect matching case can be seen as a limiting case of the matching setup, the convergence proof of the SPA is incomplete for that case.)

Conclusions

Conclusions

Conclusions

- Loopy belief propagagion is no silver bullet.

Conclusions

- Loopy belief propagagion is no silver bullet.
- However, there are interesting setups where it works very well.

Conclusions

- Loopy belief propagagion is no silver bullet.
- However, there are interesting setups where it works very well.
- Complexity of the permanent estimation based on the SPA is remarkably low. (Hard to be beaten by any standard convex optimization algorithm that minimizes the Bethe free energy.)

Conclusions

- Loopy belief propagagion is no silver bullet.
- However, there are interesting setups where it works very well.
- Complexity of the permanent estimation based on the SPA is remarkably low. (Hard to be beaten by any standard convex optimization algorithm that minimizes the Bethe free energy.)
- If the Bethe approximation does not work well, one can try better approximations, e.g., the Kikuchi approximation.

Conclusions

- Loopy belief propagagion is no silver bullet.
- However, there are interesting setups where it works very well.
- Complexity of the permanent estimation based on the SPA is remarkably low. (Hard to be beaten by any standard convex optimization algorithm that minimizes the Bethe free energy.)
- If the Bethe approximation does not work well, one can try better approximations, e.g., the Kikuchi approximation.
  Note: One can also give a comb. interpr. of the Kikuchi part. func.

Conclusions

- Loopy belief propagagion is no silver bullet.
- However, there are interesting setups where it works very well.
- Complexity of the permanent estimation based on the SPA is remarkably low. (Hard to be beaten by any standard convex optimization algorithm that minimizes the Bethe free energy.)
- If the Bethe approximation does not work well, one can try better approximations, e.g., the Kikuchi approximation.
  Note: One can also give a comb. interpr. of the Kikuchi part. func.
- Inspired by the approaches mentioned in this talk, Ryuhei Mori recently showed that many replica method computations can be simplified and made quite a bit more intuitive.


Thank you!

References

- P. O. Vontobel, "Counting in graph covers: a combinatorial characterization of the Bethe entropy function," submitted to IEEE Trans. Inf. Theory, Nov. 2010, arxiv: 1012.0065.
- P. O. Vontobel, "The Bethe permanent of a non-negative matrix," Proc. 48th Allerton Conf. on Communications, Control, and Computing, Allerton House, Monticello, IL, USA, Sep. 29 – Oct. 1, 2010.
- P. O. Vontobel, "A combinatorial characterization of the Bethe and the Kikuchi partition functions," Proc. Inf. Theory Appl. Workshop, UC San Diego, La Jolla, CA, USA, Feb. 6–11, 2011.
- R. Mori, "Connection between annealed free energy and belief propagation on random factor graph ensembles," Proc. ISIT 2011, St. Petersburg, Russia, Jul. 31 – Aug. 5, arxiv: 1102.3132.

LABS hp