November 19, 2021

Inter-University Research Institute Corporation /
Research Organization of Information and Systems
National Institute of Informatics

Boltzmann Machines

Data Mining 05 (データマイニング)

Mahito Sugiyama (杉山麿人)

Today's Outline

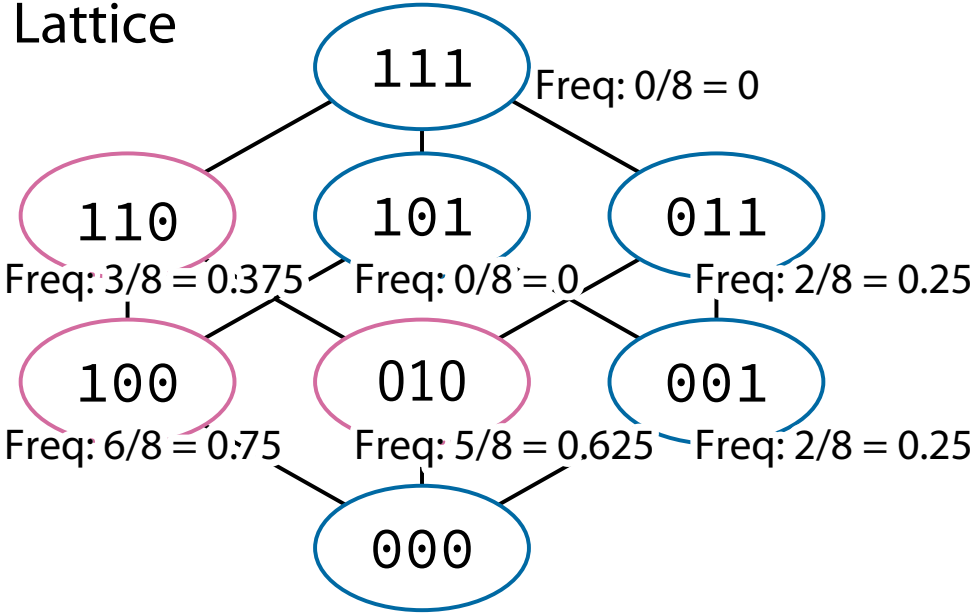
- Boltzmann Machines (Ising models):
  A fundamental probabilistic model of deep learning
  - Gibbs distribution
  - The learning equation
  - Gibbs sampling
- Relationship to the deep architecture
  - DBM (Deep Boltzmann Machines)

Learning Hierarchical Distribution (1/2)

Dataset

	Bread	Milk	Apple
ID 1	1	0	0
ID 2	1	1	0
ID 3	1	0	0
ID 4	0	1	1
ID 5	0	1	1
ID 6	1	1	0
ID 7	1	0	0
ID 8	1	1	0

Lattice

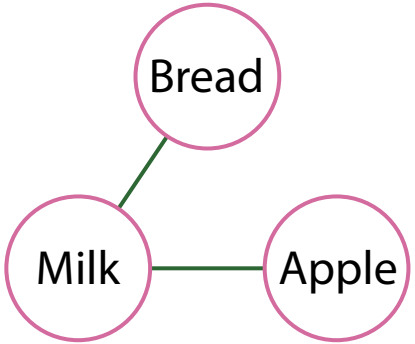


Learning Hierarchical Distribution (2/2)

MLE

$$\log(\text{prob.}) = -10.41 + 9.43[\text{Bread}] + 8.52[\text{Milk}] - 9.84[\text{Apple}] - 9.03[\text{Bread\&Milk}] + 9.43[\text{Milk\&Apple}]$$

Boltzmann machine



Bread	Milk	Apple	Prob. from data	Learned prob.
0	0	0	?	0.0000300109
1	0	0	0.375	0.3749599867
0	1	0	?	0.1499903954
0	0	1	?	0.0000000016
1	1	0	0.375	0.2250096042
1	0	1	?	0.0000200043
0	1	1	0.25	0.0999895960
1	1	1	?	0.1500004008

Boltzmann Machines

- A Boltzmann machine (BM) is represented as an undirected graph $G = (V, E)$ with $V = \{1, 2, \dots, n\}$ and $E \subseteq \{\{i, j\} \mid i, j \in V\}$
- The energy function $\Phi : \{0, 1\}^n \to \mathbb{R}$ of a BM G is defined as

$$\Phi(\boldsymbol{x}; \boldsymbol{\theta}) = -\sum_{i \in V} \theta_i x_i - \sum_{\{i,j\} \in E} \theta_{ij} x_i x_j$$

  - $\boldsymbol{x} = (x_1, x_2, \dots, x_n) \in \{0, 1\}^n$
  - $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_n, \theta_{12}, \theta_{13}, \dots, \theta_{n-1n})$ is a parameter vector for vertices (bias) $\theta_1, \dots, \theta_n$ and edges (weight) $\theta_{12}, \dots, \theta_{n-1n}$
  - $\theta_{ij} = 0$ if $\{i, j\} \notin E$

Gibbs Distribution

- Probability $p(\boldsymbol{x};\theta)$ is obtained for each $\boldsymbol{x} \in \{0,1\}^n$ as

$$p(\boldsymbol{x};\theta) = \frac{\exp(-\Phi(\boldsymbol{x},\theta))}{Z(\theta)}$$

- $Z(\theta)$ is a partition function such that

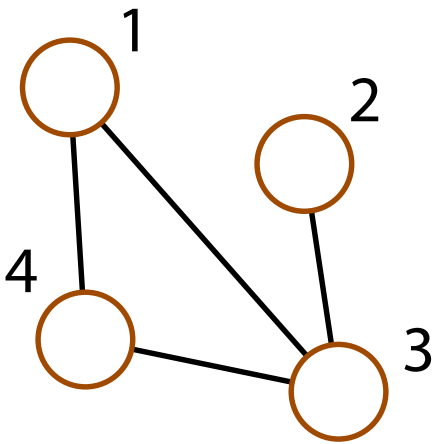
$$Z(\theta) = \sum_{\boldsymbol{x}\in\{0,1\}^n} \exp(-\Phi(\boldsymbol{x};\theta))$$

  to ensure the condition $\sum_{\boldsymbol{x}\in\{0,1\}^n} p(\boldsymbol{x}) = 1$

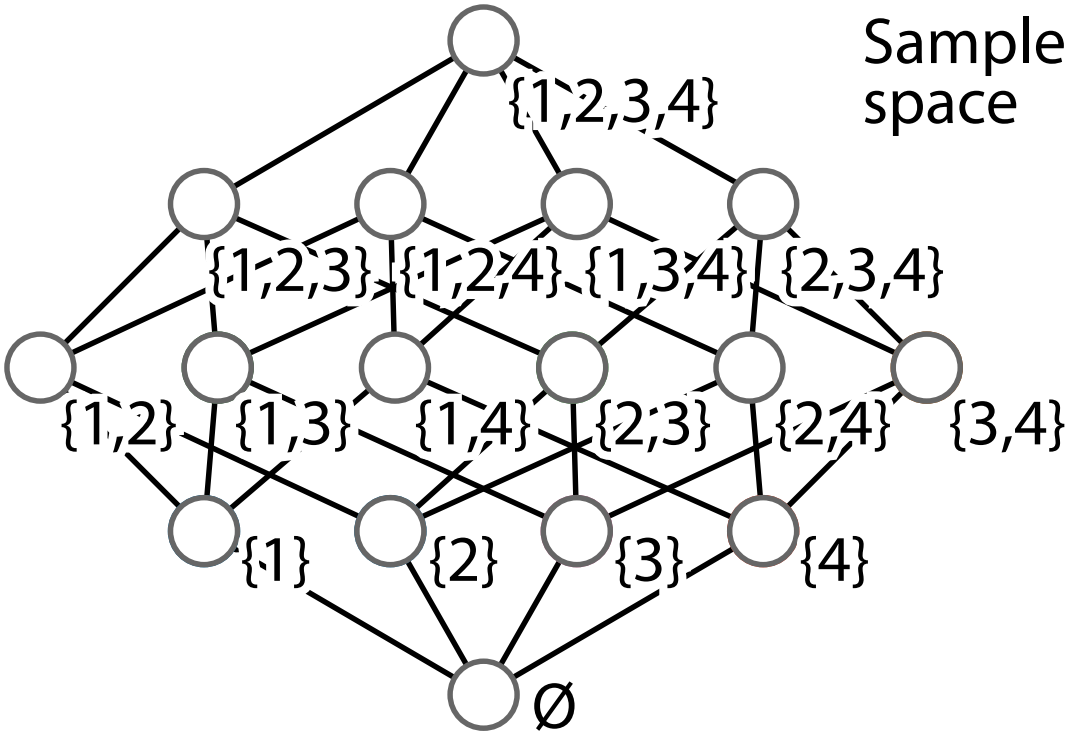
- The distribution P composed of $p(\boldsymbol{x})$ is called Gibbs distribution

Sample Space of BM

Boltzmann Machine (BM)

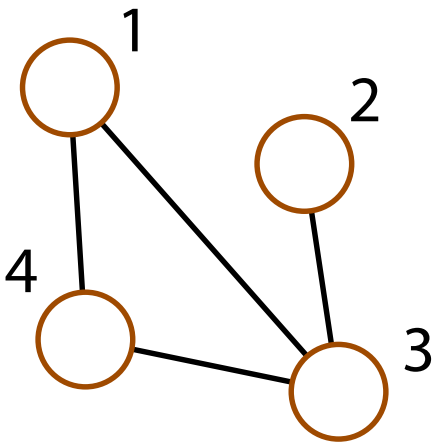


Sample space

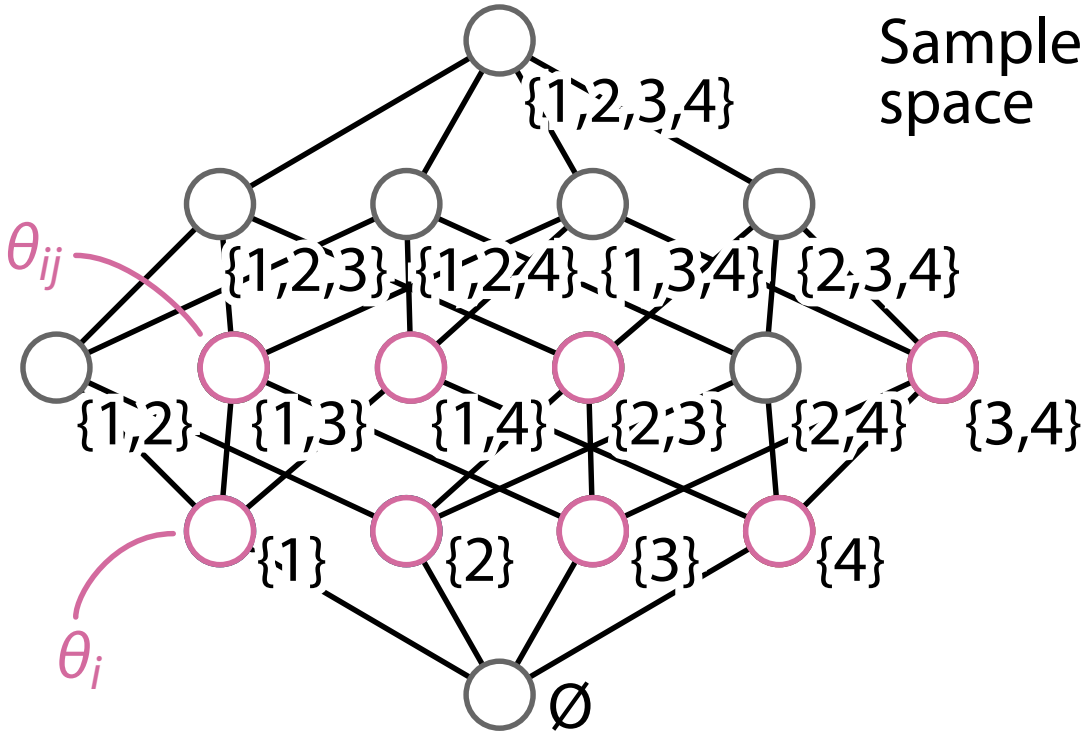


Parameters θ

Boltzmann Machine (BM)

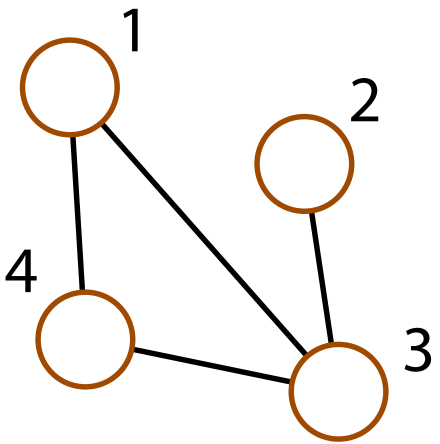


Sample space

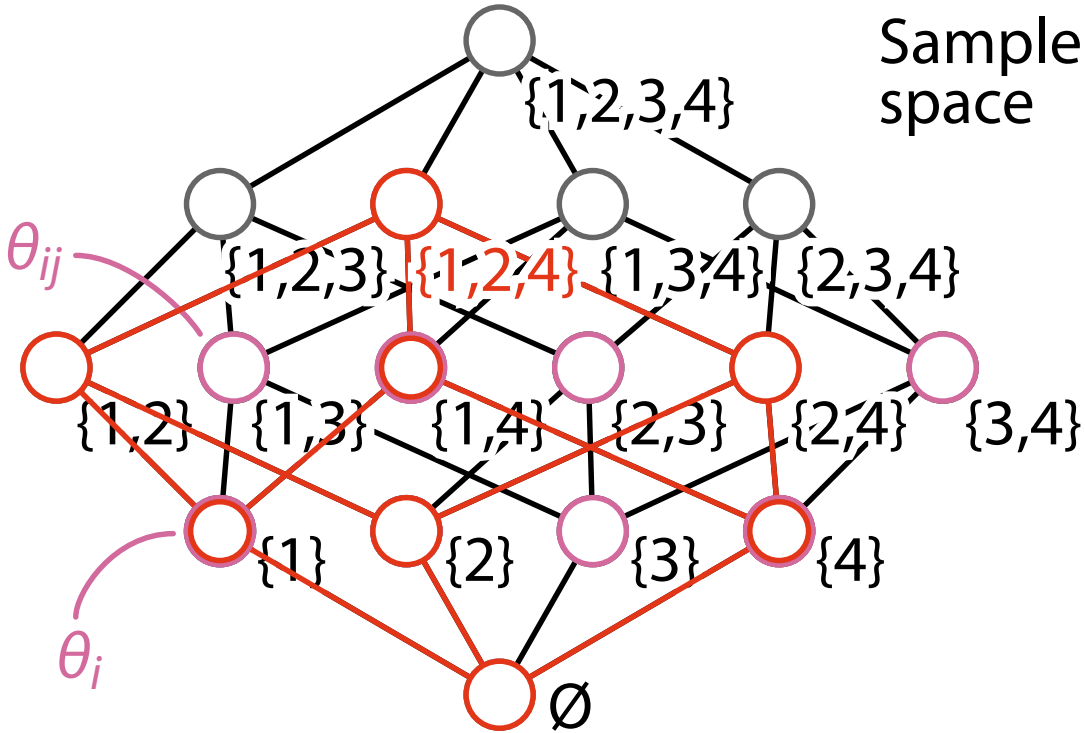


Probability Computation

Boltzmann Machine (BM)



Sample space



Learning of BM by MLE

- Given a dataset $D = \{\boldsymbol{x}_{(1)}, \boldsymbol{x}_{(2)}, \dots, \boldsymbol{x}_{(N)}\}$, the objective of learning Boltzmann machines is to maximize the (log-)likelihood (Maximum Likelihood Estimation; MLE)

  Find θ that maximizes $\prod_{k=1}^{N} p(\boldsymbol{x}_{(k)}; \theta) = p(\boldsymbol{x}_{(1)}; \theta) \cdot p(\boldsymbol{x}_{(2)}; \theta) \cdot \cdots \cdot p(\boldsymbol{x}_{(N)}; \theta)$

  - The probability of generating the given dataset by a BM

- The log-likelihood is usually used:

$$L_D(\theta) = \log \prod_{k=1}^{N} p(\boldsymbol{x}_{(k)}; \theta) = \sum_{k=1}^{N} \log p(\boldsymbol{x}_{(k)}; \theta)$$

Gradient of θ

- The gradient of $L_D(\theta)$ w.r.t. θ_i and θ_{ij} is obtained as

$$\frac{\partial L_D(\theta)}{\partial \theta_i} = |\{\boldsymbol{x}_{(k)} \in D \mid x_{(k)i} = 1\}| - N\,\eta_i,$$

$$\frac{\partial L_D(\theta)}{\partial \theta_{ij}} = \underbrace{|\{\boldsymbol{x}_{(k)} \in D \mid x_{(k)i} = x_{(k)j} = 1\}|}_{\text{data}} - \underbrace{N\,\eta_{ij}}_{\text{model}}$$

where

$$\begin{cases} \eta_i = \mathbf{E}_\theta[x_i] = \Pr(x_i = 1) = \sum_{\boldsymbol{x}} p(\boldsymbol{x};\theta)\mathbf{1}[x_i = 1] \\ \eta_{ij} = \mathbf{E}_\theta[x_i x_j] = \Pr(x_i = x_j = 1) = \sum_{\boldsymbol{x}} p(\boldsymbol{x};\theta)\mathbf{1}[x_i = x_j = 1] \end{cases}$$

Learning Equation of BM

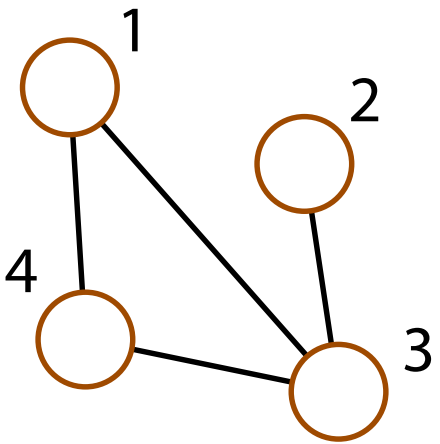
- $L_D(\theta)$ is maximized when the gradient is zero $\iff$ For all θ_i, θ_{ij}

$$\begin{cases} \dfrac{1}{N}|\{\boldsymbol{x}_{(k)} \in D \mid x_{(k)i} = 1\}| = \eta_i \\ \dfrac{1}{N}|\{\boldsymbol{x}_{(k)} \in D \mid x_{(k)i} = x_{(k)j} = 1\}| = \eta_{ij} \end{cases}$$

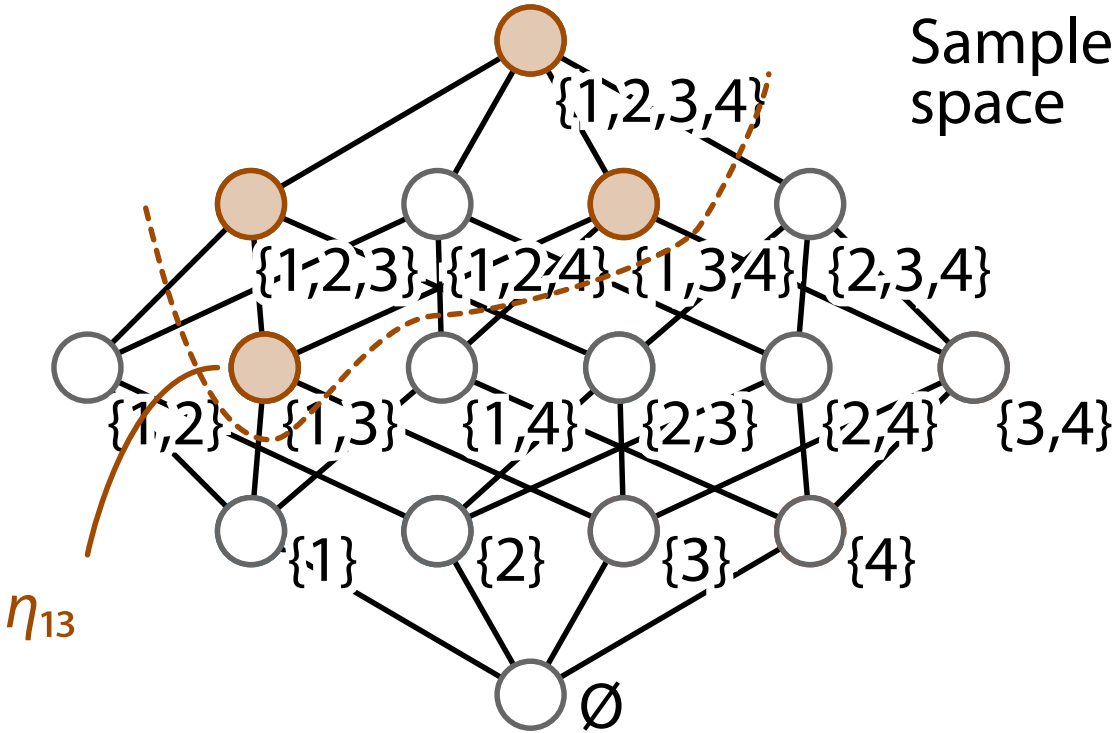
  - This is known as learning equation of BM
  - η coincides with the frequency used in itemset mining

Frequency η

Boltzmann Machine (BM)



Sample space



Empirical Frequency

- For the empirical distribution $\hat{P}$ with

  $$\hat{p}(\boldsymbol{x}) = \frac{1}{N}|\{\boldsymbol{x}_{(k)} \in D \mid \boldsymbol{x}_{(k)} = \boldsymbol{x}\}|,$$

  define

  $$\hat{\eta}_i = \frac{1}{N}|\{\boldsymbol{x}_{(k)} \in D \mid x_{(k)i} = 1\}|$$

  $$\hat{\eta}_{ij} = \frac{1}{N}|\{\boldsymbol{x}_{(k)} \in D \mid x_{(k)i} = x_{(k)j} = 1\}|$$

- The learning equation becomes

  $$\hat{\eta}_i = \eta_i, \quad \hat{\eta}_{ij} = \eta_{ij}$$

KL Divergence Minimization

- Given two distributions P, Q, the Kullback–Leibler (KL) divergence from P to Q:

$$D_{\mathrm{KL}}(P, Q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

- Maximizing the (log)likelihood is equivalent to minimizing the KL divergence: $\min_{P \in \mathcal{S}} D_{\mathrm{KL}}(\hat{P}, P)$
  - $\mathcal{S}$: the set of Gibbs distributions

Optimization: Gradient Ascent

Algorithm 1: Learning of BM by gradient ascent

Initialize θ with some values; $t \leftarrow 0$;
repeat
  foreach $i \in V$ **do**
    $\theta_i^{(t+1)} \leftarrow \theta_i^{(t)} + \varepsilon(\hat{\eta}_i - \eta_i)$;
  foreach $\{i, j\} \in E$ **do**
    $\theta_{ij}^{(t+1)} \leftarrow \theta_{ij}^{(t)} + \varepsilon(\hat{\eta}_{ij} - \eta_{ij})$;
  $t \leftarrow t + 1$
until $\theta^{(t)} = \theta^{(t+1)}$;

Combinatorial Explosion

- The serious problem of learning BMs: combinatorial explosion!!
- The time complexity of computation of η_i:

  $$\eta_i = \sum_{\boldsymbol{x} \in \{0,1\}^n} p(\boldsymbol{x}; \theta) \mathbf{1}[x_i = 1]$$

  is $O(2^n)$ and it is impossible to evaluate
  - This is required to get the gradient $\hat{\eta}(x) - \eta(x)$
- Solution: approximate it by Gibbs sampling

Gibbs Sampling (1/2)

- A Markov chain Monte Carlo (MCMC) algorithm
- We can generate samples from the current Gibbs distribution
  - n variables are dependent with each other
  - The partition function is not needed
- After obtaining enough sample $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \dots, \boldsymbol{s}_M\}$ by Gibbs sampling, η_i can be approximated as

$$\eta_i \approx \frac{1}{M} |\{\boldsymbol{s} \in S \mid s_i = 1\}|,$$

$$\eta_{ij} \approx \frac{1}{M} |\{\boldsymbol{s} \in S \mid s_i = s_j = 1\}|$$

Gibbs Sampling (2/2)

- For $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$, the conditional probability of the $i$the variable being x_i with fixing others is

$$
\begin{aligned}
p_i &= \frac{p(x_1, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_n)}{p(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n) + p(x_1, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n)} \\
&= \frac{\exp(\lambda_i x_i)}{1 + \exp(\lambda_i)}, \\
\lambda_i &= \theta_i + \sum_{j \neq i} \theta_{ij} x_j
\end{aligned}
$$

Algorithm 2: Gibbs Sampling

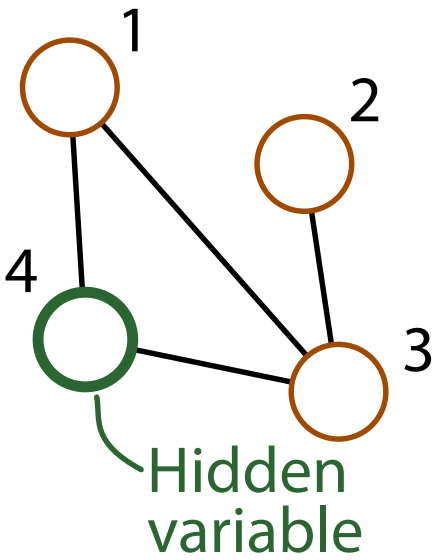
```
Initialize x with some values;
repeat
    foreach i ∈ {1, 2, ..., n} do
        if p_i ≥ random value u ∈ [0, 1] then
            x_i ← 1
        else
            x_i ← 0
    Output x and use it for the next initial vector
until getting enough sample;
```

Introducing Hidden Variables

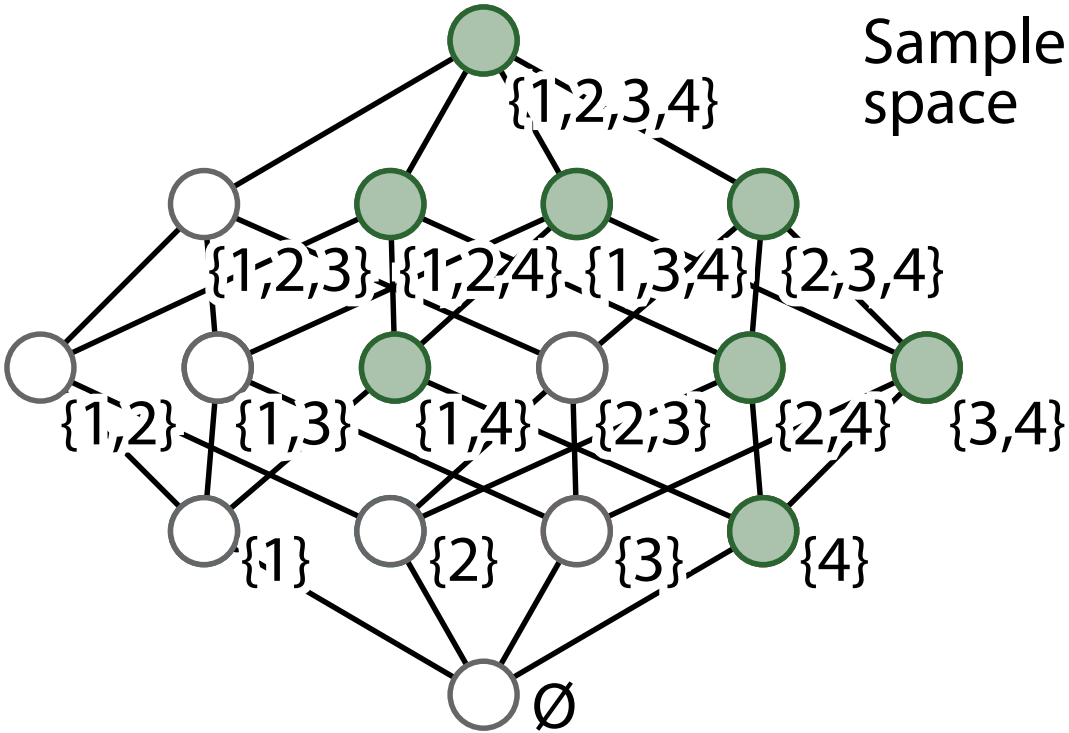
- To increase the representation power of BMs, we can introduce hidden variables
- When there are hidden nodes, the (log-)likelihood is maximized with respect to the distribution in which the hidden variables are marginalized out
- Let V and H be visible and hidden nodes

Outcome Space with Hidden Variable

Boltzmann Machine (BM)

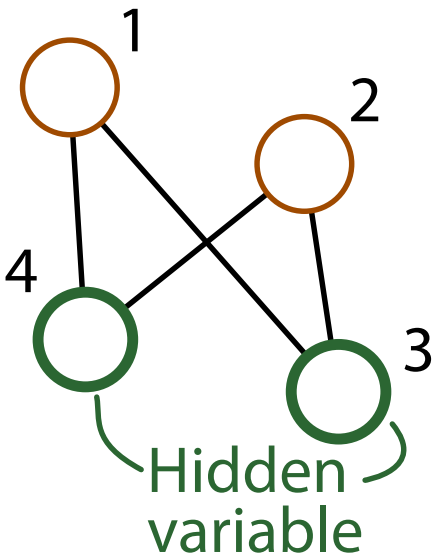


Sample space

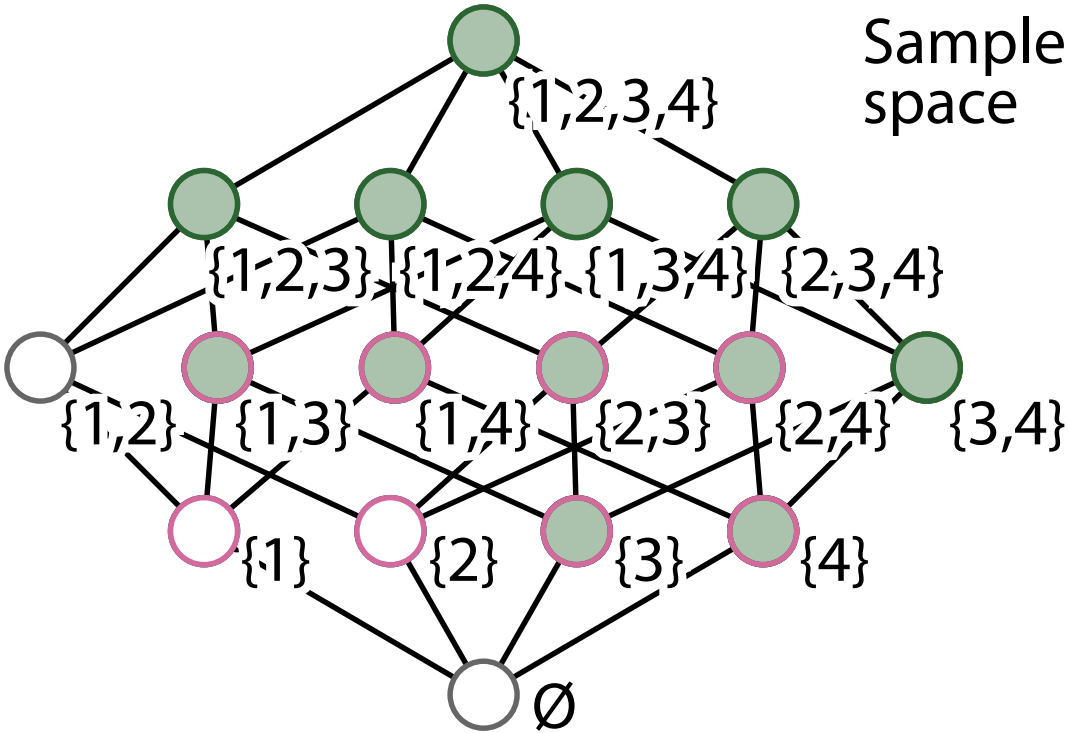


Restricted Boltzmann Machines (RBMs)

Boltzmann Machine (BM)



Sample space



Learning of RBMs

- Given a dataset $D = \{\boldsymbol{v}_{(1)}, \dots, \boldsymbol{v}_{(N)}\}$, learning equations in RBMs are

$$\frac{1}{N}\sum_{k=1}^{N} v_{(k)i} = \eta_i \quad \text{(visible, } i \in V\text{)}$$

$$\frac{1}{N}\sum_{k=1}^{N} \mathrm{sig}(\lambda_{(k)j}) = \eta_j \quad \text{(hidden, } j \in H\text{)}$$

$$\frac{1}{N}\sum_{k=1}^{N} v_{(k)i}\mathrm{sig}(\lambda_{(k)j}) = \eta_{ij}, \quad \text{(visible–hidden)}$$

$$\mathrm{sig}(\lambda_{(k)j}) = \frac{\exp(\lambda_{(k)j})}{1+\exp(\lambda_{(k)j})}, \quad \lambda_{(k)j} = \theta_j + \sum_i \theta_{ij} v_{(k)i}$$

Deep Boltzmann Machines (DBMs)