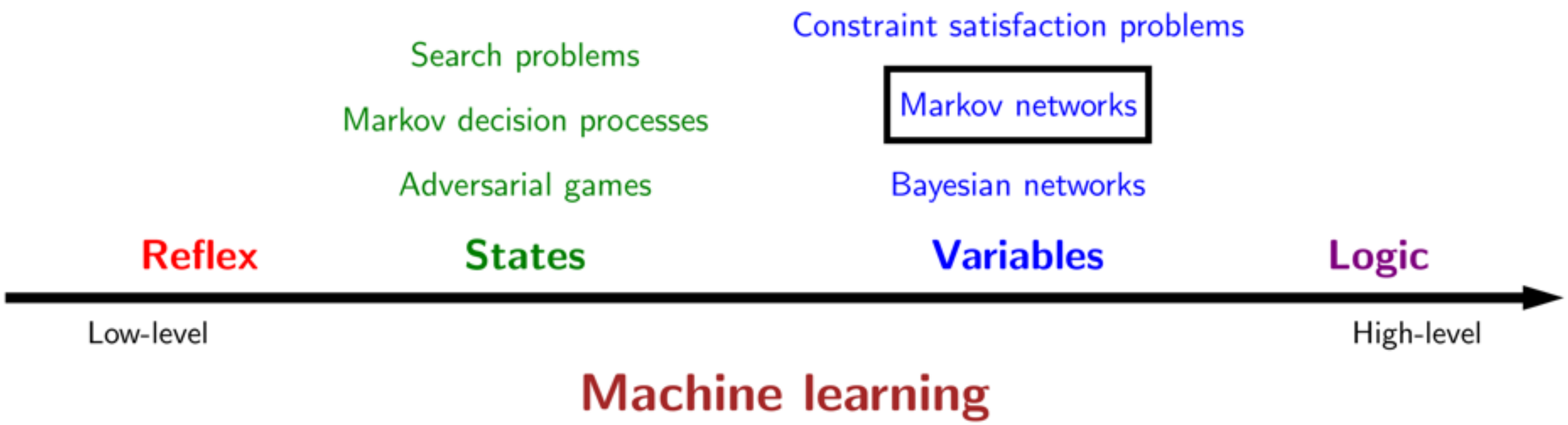
人工智能原理与技术
8.马尔可夫网络

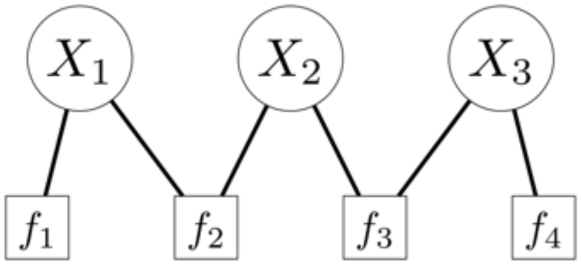
王翔

中国科学技术大学
数据科学实验室LDS

Course Plan
课程安排



Review: factor graphs
回顾：因子图



Definition: factor graph

Variables:

$X = (X_1, \dots, X_n)$, where $X_i \in \text{Domain}_i$

Factors:

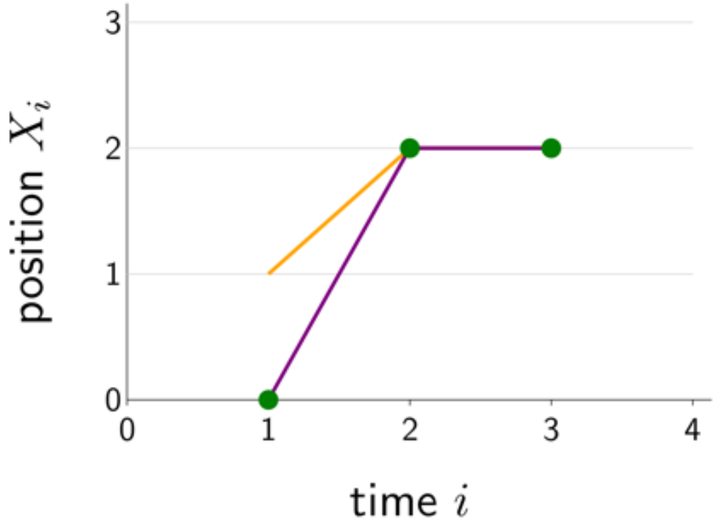
$f_1, \dots, f_m$, with each $f_j(X) \geq 0$

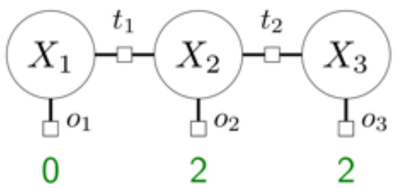
Definition: assignment weight

Each **assignment** $x = (x_1, \dots, x_n)$ has a **weight**:

$$\text{Weight}(x) = \prod_{j=1}^{m} f_j(x)$$

Example: object tracking
示例：对象跟踪





x_1	$o_1(x_1)$
0	2
1	1
2	0

x_2	$o_2(x_2)$
0	0
1	1
2	2

x_3	$o_3(x_3)$
0	0
1	1
2	2

$\lvert x_i - x_{i+1} \rvert$	$t_i(x_i, x_{i+1})$
0	2
1	1
2	0

Maximum weight assignment
最大权重分配

CSP objective: find the maximum weight assignment

$$\max_x \text{Weight}(x)$$

x_1	x_2	x_3	Weight(x)
0	1	1	4
0	1	2	4
1	1	1	4
1	1	2	4
1	2	1	2
1	2	2	8

Maximum weight assignment: $\{x_1 : 1, x_2 : 2, x_3 : 2\}$ (weight 8)

But this doesn't represent all the other possible assignments...

- How likely they are? (**uncertainty**)

Definition
定义

Definition: Markov network

A Markov network is a factor graph which defines a joint distribution over random variables $X = (X_1, \ldots, X_n)$: 定义一组随机变量的**联合分布**

$$\mathbb{P}(X = x) = \frac{\text{Weight}(x)}{Z}$$

where $Z = \sum_{x'} \text{Weight}(x')$ is the normalization constant.

x_1	x_2	x_3	Weight(x)	$\mathbb{P}(X = x)$
0	1	1	4	0.15
0	1	2	4	0.15
1	1	1	4	0.15
1	1	2	4	0.15
1	2	1	2	0.08
1	2	2	8	0.31

$$Z = 4 + 4 + 4 + 4 + 2 + 8 = 26$$

Represents uncertainty!

Marginal probabilities
边际概率

Example question: where was the object at time step 2 (X_2)?

Definition: Marginal probability

The marginal probability of $X_i = v$ is given by:

$$\mathbb{P}(X_i = v) = \sum_{x : x_i = v} \mathbb{P}(X = x)$$

Object tracking example:

x_1	x_2	x_3	Weight(x)	$\mathbb{P}(X = x)$
0	1	1	4	0.15
0	1	2	4	0.15
1	1	1	4	0.15
1	1	2	4	0.15
1	2	1	2	0.08
1	2	2	8	0.31

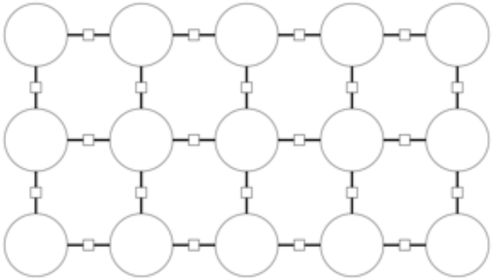
$$\mathbb{P}(X_2 = 1) = 0.15 + 0.15 + 0.15 + 0.15 = 0.62$$
$$\mathbb{P}(X_2 = 2) = 0.08 + 0.31 = 0.38$$

Note: different than max weight assignment!

最大边际概率与最大权重分配对应取值不一定相同!

Application: Ising model
应用：Ising模型

Ising model: classic model from statistical physics to model ferromagnetism



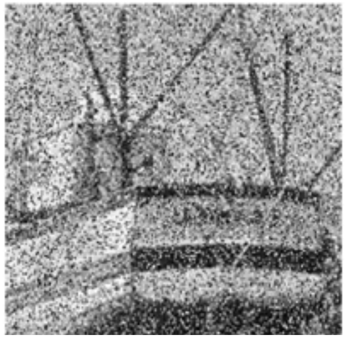
$X_i \in \{-1, +1\}$: atomic spin of site i

$f_{ij}(x_i, x_j) = \exp(\beta x_i x_j)$ wants same spin

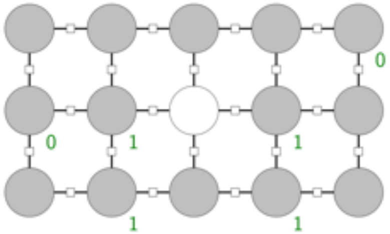
Samples as β increases:

Figure 2 from Perez (1998)

Application: image denoising
应用：图像去噪



Example: image denoising



- $X_i \in \{0, 1\}$ is pixel value in location i
- Subset of pixels are observed

$o_i(x_i) = [x_i = \text{observed value at } i]$

- Neighboring pixels more likely to be same than different

$t_{ij}(x_i, x_j) = [x_i = x_j] + 1$

Summary
小结

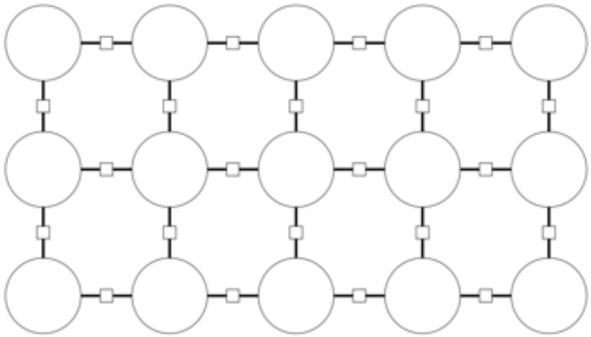
$\boxed{\text{Markov networks} = \text{factor graphs} + \text{probability}}$

马尔可夫网络 = 因子图 + 概率

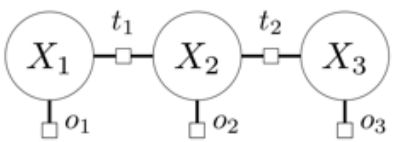
- Normalize weights to get probablity distribution
- Can compute marginal probabilities to focus on variables

CSPs	Markov networks
variables	random variables
weights	probabilities
maximum weight assignment	marginal probabilities

Markov networks: Gibbs sampling
马尔可夫网络：吉布斯抽样



Review: Markov networks
回顾：马尔可夫网络



Definition: Markov network

A Markov network is a factor graph which defines a joint distribution over random variables $X = (X_1, \ldots, X_n)$:

$$\mathbb{P}(X = x) = \frac{\text{Weight}(x)}{Z}$$

where $Z = \sum_{x'} \text{Weight}(x')$ is the normalization constant.

Objective: compute marginal probabilities $\mathbb{P}(X_i = v) = \sum_{x : x_i = v} \mathbb{P}(X = x)$

x_1	x_2	x_3	Weight(x)	$\mathbb{P}(X = x)$
0	1	1	4	0.15
0	1	2	4	0.15
1	1	1	4	0.15
1	1	2	4	0.15
1	2	1	2	0.08
1	2	2	8	0.31

$$Z = 4 + 4 + 4 + 4 + 2 + 8 = 26$$

$$\mathbb{P}(X_2 = 1) = 0.15 + 0.15 + 0.15 + 0.15 = 0.62$$
$$\mathbb{P}(X_2 = 2) = 0.08 + 0.31 = 0.38$$

Gibbs sampling
吉布斯抽样

Algorithm: Gibbs sampling

Initialize x to a random complete assignment

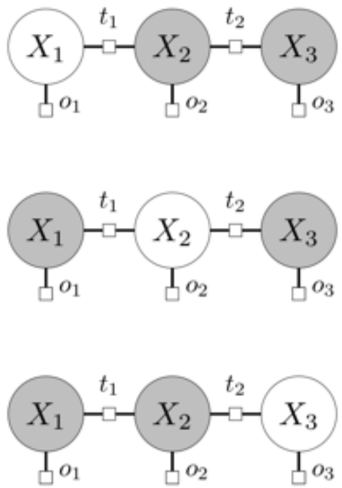
Loop through $i = 1, \ldots, n$ until convergence:

  Set $x_i = v$ with prob. $\mathbb{P}(X_i = v \mid X_{-i} = x_{-i})$

    (X_{-i} denotes all variables except X_i)

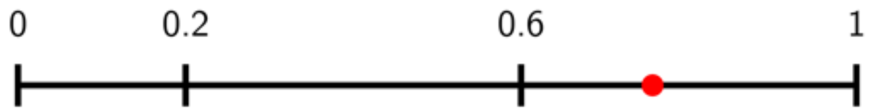
  Increment $\text{count}_i(x_i)$

Estimate $\hat{\mathbb{P}}(X_i = x_i) = \frac{\text{count}_i(x_i)}{\sum_v \text{count}_i(v)}$



Example: sampling one variable

$\text{Weight}(x \cup \{X_2 : 0\}) = 1$	prob. 0.2
$\text{Weight}(x \cup \{X_2 : 1\}) = 2$	prob. 0.4
$\text{Weight}(x \cup \{X_2 : 2\}) = 2$	prob. 0.4



- Now we present Gibbs sampling, a simple algorithm for approximately computing marginal probabilities. The algorithm follows the template of local search, where we change one variable at a time, but unlike Iterated Conditional Modes (ICM), Gibbs sampling is a randomized algorithm.
- Gibbs sampling proceeds by going through each variable X_i, considering all the possible assignments of X_i with some $v \in \text{Domain}_i$, and setting $X_i = v$ with probability equal to the conditional probability of $X_i = v$ given everything else.
- To perform this step, we can rewrite this expression using laws of probability: $\mathbb{P}(X_i = v \mid X_{-i} = x_{-i}) = \frac{\text{Weight}(x \cup \{X_i : v\})}{Z\mathbb{P}(X_{-i} = x_{-i}}$, where the denominator is a new normalization constant. We don't need to compute it directly. Instead, we first compute the weight of $x \cup \{X_i : v\}$ for each v, and then normalize to get a distribution. Finally we sample a v according to that distribution.
- Along the way, for each variable X_i that we're interested in tracking, we keep a counter $\text{count}_i(v)$ of how many times we've seen $X_i = v$. These counts can be normalized at any time to produce an estimate $\hat{\mathbb{P}}(X_i = x_i)$ of the marginal probability.

Gibbs sampling
吉布斯抽样

Examples: [vote] [csp] [pair] [chain] [track] [alarm] [med] [dep] [delay] [mln] [lsat] [new]
[Background] [Documentation]

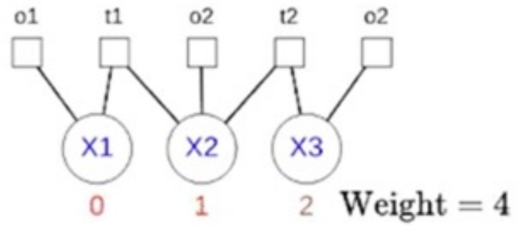
```
// Object tracking example

// X1,X2,X3 are unknown object positions
variable('X1', [0, 1, 2])
variable('X2', [0, 1, 2])
variable('X3', [0, 1, 2])

// Transitions: adjacent positions nearby
// Observations: positions, sensor readings nearby
function nearby(a, b) {
  if (a == b) return 2
  if (Math.abs(a-b) == 1) return 1
  return 0
}
function observe(a) {
  return function(b) {return nearby(a, b)}
}
factor('o1', 'X1', observe(0))
factor('t1', 'X1 X2', nearby)
factor('o2', 'X2', observe(2))
factor('t2', 'X2 X3', nearby)
factor('o2', 'X3', observe(2))

query('X2'); gibbsSampling({steps:1000})
```

Query: $\mathbb{P}(\mathrm{X2})$
Algorithm: **Gibbs sampling**



Sampling variable X3 given everything else:

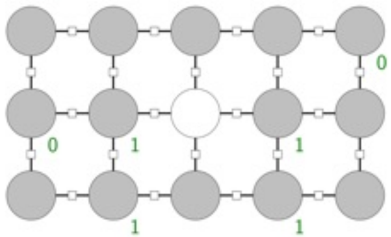
X3:?	t2	o2	Weight	$\mathbb{P}(\mathrm{X3}=?)$
0	1	0	0	0
1	2	1	2	0.5
2	1	2	2	0.5

Choose X3:2
Estimate of query based on 6000 samples:

X2	count	$\hat{\mathbb{P}}(\mathrm{X2})$
1	3671	0.61
2	2329	0.39

Application: image denoising
应用：图像去噪

Example: image denoising



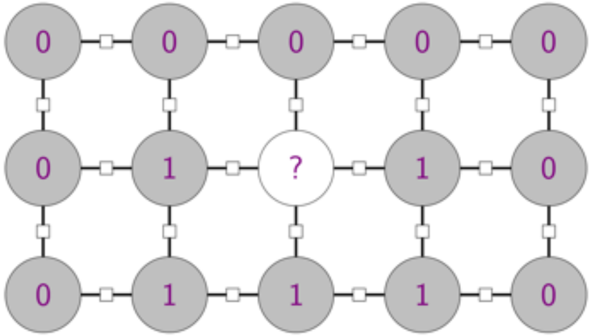
- $X_i \in \{0,1\}$ is pixel value in location i
- Subset of pixels are observed

$o_i(x_i) = [x_i = \text{observed value at } i]$

- Neighboring pixels more likely to be same than different

$t_{ij}(x_i, x_j) = [x_i = x_j] + 1$

Gibbs sampling for image denoising
吉布斯采样用于图像去噪

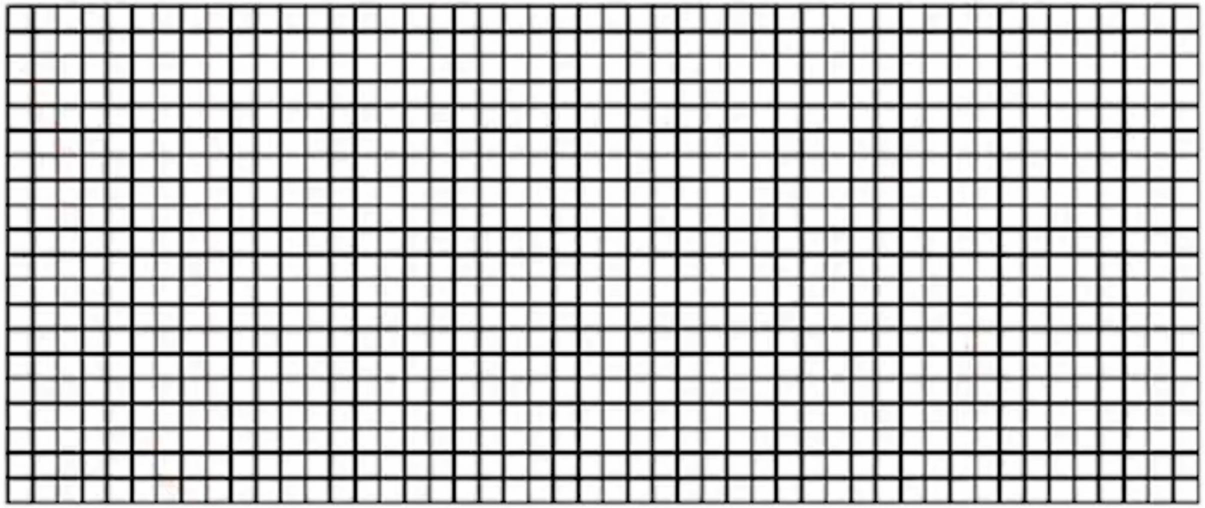


$$t_{ij}(x_i, x_j) = [x_i = x_j] + 1$$

Scan through image and update each pixel given rest:

v	weight	$\mathbb{P}(X_i = v \mid X_{-i} = x_{-i})$
0	$2 \cdot 1 \cdot 1 \cdot 1$	0.2
1	$1 \cdot 2 \cdot 2 \cdot 2$	0.8

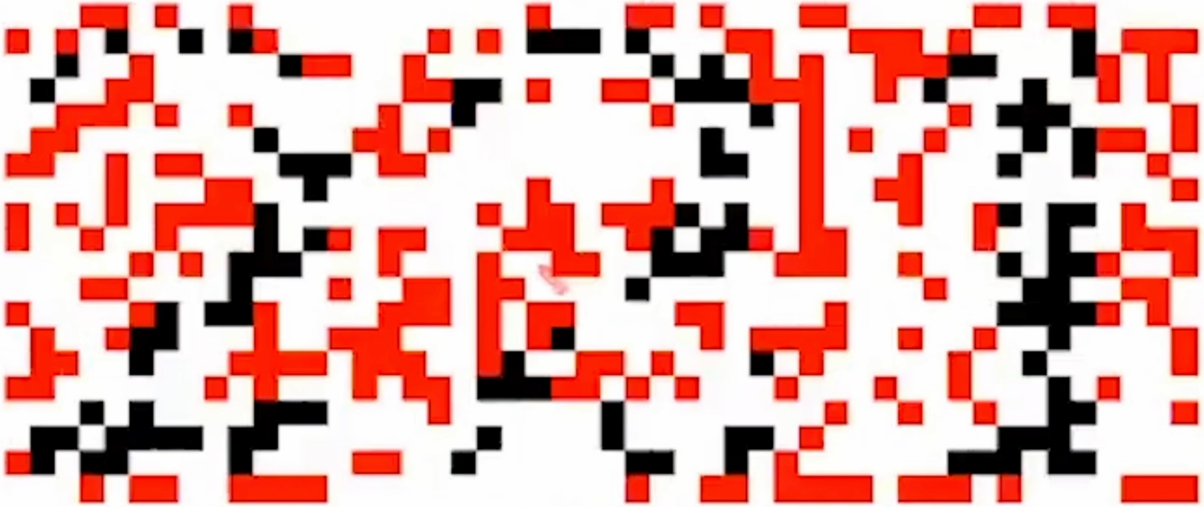
Image denoising demo
图像去噪演示



1 iterations

```
// Press ctrl-enter to save settings
setImage(0)              // What true image to use (either 0 or 1)
missingFrac = 0.6        // Fraction of missing pixels
coherenceFactor = 2      // Transition factors: prefer equal neighbors
initRandom = false       // Initialize sampler randomly
icm = false              // Use ICM, not Gibbs sampling
showMarginals = false    // Show marginals (averages), not samples
sampleTime = 100         // Milliseconds to wait between samples
```

Image denoising demo
图像去噪演示



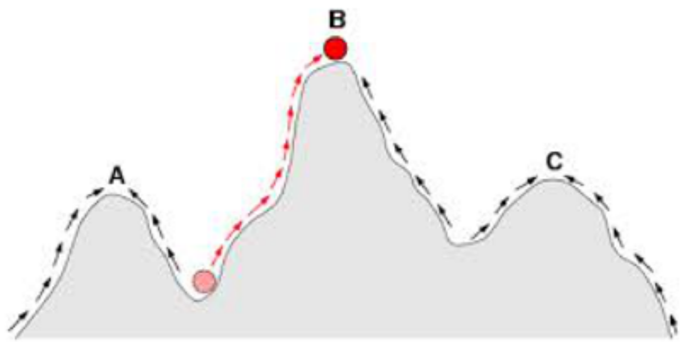
1 iterations

```
// Press ctrl-enter to save settings
setImage(0)               // What true image to use (either 0 or 1)
missingFrac = 0.6         // Fraction of missing pixels
coherenceFactor = 2       // Transition factors: prefer equal neighbors
initRandom = false        // Initialize sampler randomly
icm = false               // Use ICM, not Gibbs sampling
showMarginals = true      // Show marginals (averages), not samples
sampleTime = 100          // Milliseconds to wait between samples
```

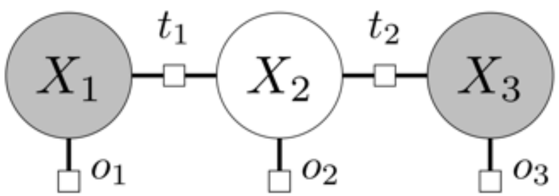
Search versus sampling
搜索与抽样

	Iterated Conditional Modes	Gibbs sampling	
最大权重分配	maximum weight assignment	marginal probabilities	边际概率
选择最优值	choose best value	sample a value	采样值
收敛局部最优	converges to local optimum	marginals converge to correct answer*	收敛全局最优

*under technical conditions (sufficient condition: all weights positive), but could take exponential time



Summary
小结



- Objective: compute marginal probabilities $\mathbb{P}(X_i = x_i)$
- Gibbs sampling: sample one variable at a time, count visitations
- More generally: Markov chain Monte Carlo (MCMC) powerful toolkit of randomized procedures