Table of Contents I

1. Inference Tasks
   - First example (MRF) and the challenges

2. Chains and Trees

3. Loopy Graphs
   - Computing Marginals and Partition Function
   - Computing the MAP Exactly for Binary Label
   - Computing the MAP Approximately for Multilabel
Outline

1. Inference Tasks
   - First example (MRF) and the challenges

2. Chains and Trees

3. Loopy Graphs
   - Computing Marginals and Partition Function
   - Computing the MAP Exactly for Binary Label
   - Computing the MAP Approximately for Multilabel
Marginal and MAP Queries

Given joint distribution \( P(Y, E) \), where
- \( Y \), query random variable(s), unknown
- \( E \), evidence random variable(s), observed i.e. \( E = e \).

Two types of queries:
- **Marginal** queries (a.k.a. probability queries)
  task is to compute \( P(Y|E = e) \)
- **MAP** queries (a.k.a. most probable explanation)
  task is to find \( y^* = \arg\max_{y \in \text{Val}(Y)} P(Y|E = e) \)
Marginal and MAP Inference (with/without Evidence)

(a) Directed graph (Bayesian network)  
(b) Undirected graph (Markov random field (MRF))  
(c) Factor graph (higher order)

MAP inference: \((x_1^*, x_2^*, x_3^*) = \arg\max_{x_1,x_2,x_3} P(x_1, x_2, x_3)\)

Marginal inference: \(P(x_i) = \sum_{x_j : j \neq i} P(x_1, x_2, x_3)\)

Max-Marginal inference: \((\arg\max_{x_1} P(x_1), \arg\max_{x_2} P(x_2), \arg\max_{x_3} P(x_3))\)
Outline

1. Inference Tasks
   - First example (MRF) and the challenges

2. Chains and Trees

3. Loopy Graphs
   - Computing Marginals and Partition Function
   - Computing the MAP Exactly for Binary Label
   - Computing the MAP Approximately for Multilabel
MRF: the First Example – Ising Model

- Statistical physics: ferromagnetism, Ernst Ising (1924, PhD Thesis)
- Atoms in a lattice: magnetic dipole moments (up/down)
- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, Labels $\mathcal{L} = \{+1, -1\} = \{\text{up, down}\}$
- All configurations (labelings): $\mathcal{Y} = \{+1, -1\}^\mathcal{V}$
Ising Model (Cont’d)

- All configurations (labelings): \( y = \{+1, -1\}^{|\mathcal{V}|} \)

- Energy
  \[ E(y) = \sum_{(i,j) \in \mathcal{E}} \theta(y_i, y_j) + \sum_{i \in \mathcal{V}} \mu(y_i) \]

- Probability
  \[ P(y) = \exp(-E(y))/Z, \text{ where } Z = \sum_{y \in \mathcal{Y}} P(y) \]

- In this example: \( E(y) = 12\alpha + 12\beta + 8\gamma + 8\eta \)

Parameters:

\[
\begin{array}{c|cc}
  y_j & +1 & -1 \\
  \hline
  y_i & +1 & \alpha & \beta \\
  -1 & \beta & \alpha \\
  -1 & \gamma & \eta \\
\end{array}
\]
In General

- **Ising model:**
  - A grid graph
  - A discrete distribution
  - The domain has $2^{16}$ configurations/labelings
  - 4 parameters ($\alpha, \beta, \gamma, \eta$)

![Graph](image-url)
In General (cont’d)

- **Labelings:** \( y = \mathcal{L}^{V} = \{1, \cdots, L\}^{V} \)
- **Parameters:** \( [\vec{\theta}, \vec{\mu}], |E| \cdot L^2 + |V| \cdot L \) long

![Graph](image)

<table>
<thead>
<tr>
<th>Binaries</th>
<th>Unaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y_i )</td>
<td>( y_j )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>( \theta_{ij}(1,2) )</td>
</tr>
<tr>
<td>( \cdots )</td>
<td>( \cdots )</td>
</tr>
<tr>
<td>( L )</td>
<td>( \theta_{ij}(L,1) )</td>
</tr>
</tbody>
</table>
Chao Chen  
Tutorial on Probabilistic Graphical Models: A Geometric and Topological View  
Part 2: Inference

In General (cont’d)

- **Labelings:** \( y = \mathcal{L}^{\mathcal{V}} = \{1, \cdots, L\}^{\mathcal{V}} \)
- **Parameters:** \([\vec{\theta}, \vec{\mu}], |\mathcal{E}| \cdot L^2 + |\mathcal{V}| \cdot L \text{ long}\)

![Graph]

- **Energy:** \( E(y) = \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) + \sum_{i \in \mathcal{V}} \mu_i(y_i) \)
- **Probability:** \( P(y) = \exp\left(-E(y)\right)/Z \)
- **Partition function:** \( Z = \sum_{y \in \mathcal{Y}} \exp\left(-E(y)\right) \)
- **Conditional independent:** \( \forall (i, j) \notin \mathcal{E}, V_i \perp V_j \mid \mathcal{V} \setminus \{V_i, V_j\} \)
Inference tasks

- **MAP** $\text{argmax}_{y \in Y} P(y)$
  Predict the most possible explanation

- Marginals $P(y_i = \ell)$
  Important in parameter learning (compute gradient)

Pic from Shotton et al. 2007
Challenges

- Computing the maximum a posteriori (MAP):
  \[ \arg\max_{y \in \mathcal{Y}} P(y) = \arg\min_{y \in \mathcal{Y}} E(y) \]

- NP-hard (reduction from maximal independent set problem)
Challenges

- Computing marginals: \( P(y_i = \ell) = \sum_{y : y_i = \ell} P(y) \)
- Computing the partition function: \( Z = \sum_{y \in Y} \exp(-E(y)) \)
  \#P-hard (counting \# of solutions of a NP-hard problem)
- \( E(y) = 0 \) if \( y \) is an independent set (with label 2)
- \( E(y) = \infty \) otherwise
- \( Z = \sum_y e^{-E(y)} = \# \) of independent sets
Inference Tasks
- First example (MRF) and the challenges

Chains and Trees

Loopy Graphs
- Computing Marginals and Partition Function
- Computing the MAP Exactly for Binary Label
- Computing the MAP Approximately for Multilabel
Simple Graphs

When the graph is a chain or a tree:

- Inference is easy:
  - dynamic programming
  - variable elimination
  - message passing

- Simplify the setting, drop unary terms:
  \[
  E(y) = \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) + \sum_{i \in \mathcal{V}} \mu_i(y_i)
  \]

- Start with the partition function

\[
Z = \sum_y \exp(-E(y)) = \sum_y \exp \left( - \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) \right)
\]
\[ Z = \sum_{y} \exp \left( - \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) \right) \]
\[ Z = \sum_y \exp \left( - \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) \right) \]

\[ Z = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} \sum_{y_4} \sum_{y_5} e^{-\theta_{12}(y_1, y_2)} e^{-\theta_{23}(y_2, y_3)} e^{-\theta_{34}(y_3, y_4)} e^{-\theta_{45}(y_4, y_5)} \]
\[ Z = \sum_y \exp \left( - \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) \right) \]
\[ Z = \sum_y \exp \left( - \sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) \right) \]

\[
Z = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} \sum_{y_4} \sum_{y_5} e^{-\theta_{12}(y_1, y_2)} e^{-\theta_{23}(y_2, y_3)} e^{-\theta_{34}(y_3, y_4)} e^{-\theta_{45}(y_4, y_5)}
\]

\[
= \sum_{y_1 \in \mathcal{L}} \sum_{y_2} e^{-\theta_{12}(y_1, y_2)} \sum_{y_3} e^{-\theta_{23}(y_2, y_3)} \sum_{y_4} e^{-\theta_{34}(y_3, y_4)} \sum_{y_5} e^{-\theta_{45}(y_4, y_5)}
\]

\[
= \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{-\theta_{12}(y_1, y_2)} e^{-\theta_{23}(y_2, y_3)} \sum_{y_4} e^{-\theta_{34}(y_3, y_4)} \sum_{y_5} e^{-\theta_{45}(y_4, y_5)}
\]
Inference Tasks
Chains and Trees
Loopy Graphs

\[ Z = \sum_y \exp \left( -\sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) \right) \]

\[ Z = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} \sum_{y_4} \sum_{y_5} e^{-\theta_{12}(y_1, y_2)} e^{-\theta_{23}(y_2, y_3)} e^{-\theta_{34}(y_3, y_4)} e^{-\theta_{45}(y_4, y_5)} \]

\[ = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{-\theta_{12}(y_1, y_2)} \sum_{y_4} e^{-\theta_{23}(y_2, y_3)} \sum_{y_4} e^{-\theta_{34}(y_3, y_4)} \sum_{y_5} e^{-\theta_{45}(y_4, y_5)} \]

\[ = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} e^{-\theta_{12}(y_1, y_2)} \sum_{y_3} e^{-\theta_{23}(y_2, y_3)} \sum_{y_4} e^{-\theta_{34}(y_3, y_4)} M_{5\to4}(y_4) \]

\[ = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} e^{-\theta_{12}(y_1, y_2)} \sum_{y_3} e^{-\theta_{23}(y_2, y_3)} M_{4\to3}(y_3) \]
\[ Z = \sum_y \exp \left( - \sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) \right) \]

\[ Z = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} \sum_{y_4} \sum_{y_5} e^{-\theta_{12}(y_1, y_2)} e^{-\theta_{23}(y_2, y_3)} e^{-\theta_{34}(y_3, y_4)} e^{-\theta_{45}(y_4, y_5)} \]

\[ = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{-\theta_{12}(y_1, y_2)} \sum_{y_3} e^{-\theta_{23}(y_2, y_3)} \sum_{y_4} e^{-\theta_{34}(y_3, y_4)} \sum_{y_5} e^{-\theta_{45}(y_4, y_5)} \]

\[ = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{-\theta_{12}(y_1, y_2)} \sum_{y_3} e^{-\theta_{23}(y_2, y_3)} \sum_{y_4} e^{-\theta_{34}(y_3, y_4)} M_{5 \rightarrow 4}(y_4) \]

\[ = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{-\theta_{12}(y_1, y_2)} \sum_{y_3} e^{-\theta_{23}(y_2, y_3)} M_{4 \rightarrow 3}(y_3) \]

\[ = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} e^{-\theta_{12}(y_1, y_2)} M_{3 \rightarrow 2}(y_2) \]
\[
Z = \sum_{y} \exp \left( - \sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) \right)
\]

\[
Z = \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} \sum_{y_4} \sum_{y_5} e^{(-\theta_{12}(y_1, y_2))} e^{(-\theta_{23}(y_2, y_3))} e^{(-\theta_{34}(y_3, y_4))} e^{(-\theta_{45}(y_4, y_5))}
\]

\[
= \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{(-\theta_{12}(y_1, y_2))} \sum_{y_4} e^{(-\theta_{23}(y_2, y_3))} \sum_{y_5} e^{(-\theta_{34}(y_3, y_4))} e^{(-\theta_{45}(y_4, y_5))}
\]

\[
= \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{(-\theta_{12}(y_1, y_2))} \sum_{y_4} e^{(-\theta_{23}(y_2, y_3))} M_{5 \rightarrow 4}(y_4)
\]

\[
= \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_3} e^{(-\theta_{12}(y_1, y_2))} M_{4 \rightarrow 3}(y_3)
\]

\[
= \sum_{y_1 \in \mathcal{L}} \sum_{y_2} e^{(-\theta_{12}(y_1, y_2))} M_{3 \rightarrow 2}(y_2)
\]

\[
= \sum_{y_1 \in \mathcal{L}} M_{2 \rightarrow 1}(y_1)
\]
Dynamic Programming

Compute the partition function

\[ Z = \sum_y \exp \left( - \sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) \right) \]

Recursion:

\[ M_{i+1 \rightarrow i}(y_i) = \sum_{y_{i+1}, \ldots, y_D} e^{[-\theta_{i,i+1}(y_i, y_{i+1}) - \cdots - \theta_{D-1,D}(y_{D-1}, y_D)]} \]

\[ = \sum_{y_{i+1}} e^{[-\theta_{i,i+1}(y_i, y_{i+1})]} M_{i+2 \rightarrow i+1}(y_{i+1}) \]
A Simple Chain (Marginals)

Marginals $P(y_i = \ell)$:

$$
\frac{1}{Z} \sum_{y : y_i = \ell} \exp \left( - \sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) \right)
$$
Inference Tasks
Chains and Trees
Loopy Graphs

A Simple Chain (Marginals)

Marginals \( P(y_i = \ell) \):

\[
\frac{1}{Z} \sum_{y:y_i = \ell} \exp \left( - \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) \right)
\]

\[
M_{4 \rightarrow 3}(y_3) = \sum_{y_4, y_5} e^{-\theta_{34}(y_3, y_4)} e^{-\theta_{45}(y_4, y_5)}
\]

\[
M_{2 \rightarrow 3}(y_3) = \sum_{y_1, y_2} e^{-\theta_{12}(y_1, y_2)} e^{-\theta_{23}(y_2, y_3)}
\]
Marginals $P(y_i = \ell)$:

\[
\frac{1}{Z} \sum_{y:y_i=\ell} \exp \left( - \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) \right)
\]

\[
M_{4 \rightarrow 3}(y_3) = \sum_{y_4, y_5} e^{(-\theta_{34}(y_3, y_4))} e^{(-\theta_{45}(y_4, y_5))}
\]

\[
M_{2 \rightarrow 3}(y_3) = \sum_{y_1, y_2} e^{(-\theta_{12}(y_1, y_2))} e^{(-\theta_{23}(y_2, y_3))}
\]

\[
P(y_3 = \ell) = \frac{1}{Z} \sum_{y_1 \in \mathcal{L}} \sum_{y_2} \sum_{y_4} \sum_{y_5} e^{(-\theta_{12}(y_1, y_2))} e^{(-\theta_{23}(y_2, y_3 = \ell))} e^{(-\theta_{34}(y_3 = \ell, y_4))} e^{(-\theta_{45}(y_4, y_5))}
\]

\[
= \frac{1}{Z} \left( \sum_{y_1, y_2} e^{(-\theta_{12}(y_1, y_2))} e^{(-\theta_{23}(y_2, y_3 = \ell))} \right) \cdot \left( \sum_{y_4, y_5} e^{(-\theta_{34}(y_3, y_4))} e^{(-\theta_{45}(y_4, y_5))} \right)
\]

\[
= \frac{1}{Z} M_{2 \rightarrow 3}(\ell) \cdot M_{4 \rightarrow 3}(\ell)
\]
A Simple Chain: Algorithm

\[ M_{i-1 \rightarrow i}(y_i) = \sum_{y_{i-1}} e^{-\theta_{i-1,i}(y_{i-1},y_i)} M_{i-2 \rightarrow i-1}(y_{i-1}) \]

\[ M_{i+1 \rightarrow i}(y_i) = \sum_{y_{i+1}} e^{-\theta_{i,i+1}(y_i,y_{i+1})} M_{i+2 \rightarrow i+1}(y_{i+1}) \]

\[ Z = \sum_{y_1} M_{2 \rightarrow 1}(y_1) \]

\[ P(y_i = \ell) = \frac{1}{Z} M_{i-1 \rightarrow i}(\ell) \cdot M_{i+1 \rightarrow i}(\ell) \]

Implementation: take log of everything
**MAP Computation**

Compute the MAP

\[
\arg\min_y \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j)
\]

Recursion:

\[
M_{i+1 \rightarrow i}(y_i) = \arg\min_{y_{i+1}, \ldots, y_D} [\theta_{i,i+1}(y_i, y_{i+1}) + \cdots + \theta_{D-1,D}(y_{D-1}, y_D)]
\]

\[
= \arg\min_{y_{i+1}} [\theta_{i,i+1}(y_i, y_{i+1}) + M_{i+2 \rightarrow i+1}(y_{i+1})]
\]

Complexity \(O(\mathcal{V}L^2)\)
A Tree: Partition Function/Marginals

- \( M_{v\rightarrow p}(y_p = \ell) = \sum_{y_v} e^{-\theta_{v,p}(y_v,y_p)} M_{c_1\rightarrow v}(y_v) M_{c_2\rightarrow v}(y_v) \)
- Message Scheduling:
  - Fix a root \( R \)
  - Compute messages from leaves to the root (blue)
  - Compute messages from the root to leaves (red)

- \( Z = \sum_{y_R} \prod_{u \in \text{Children}(R)} M_{u\rightarrow R}(y_R) \)

- \( P(y_v = \ell) = \frac{1}{Z} \prod_{u \in \mathcal{N}(v)} M_{u\rightarrow v}(\ell) \)
- Implementation: take log of everything
A Tree: MAP

- $MAP = \arg\min E(y)$
- $M_{v \to p}(y_p = \ell) = \min_{y_v} (\theta_{v,p}(y_v, y_p) + M_{c_1 \to v}(y_v) + M_{c_2 \to v}(y_v))$
- Message Scheduling: the same
- $E(MAP) = \min_{y_R} \sum_{u \in \text{Children}(R)} M_{u \to R}(y_R)$
- Recover $MAP$ via backtracking
Chains and Trees: Applications

- Video labeling
- Simple Chain:
  - nodes → frames
  - labels → different gestures

ChaLearn dataset,
http://chalearnlap.cvc.uab.es/dataset/23/description/
Chains and Trees: Neuron image segmentation

- Input: images (nm² per pixel),
- Output: partitions = neurons
- Challenges: large variation of shape and appearance

Mustafa, Chen and Metaxas MICCAI’14, MedIA’15
Chains and Trees: Neuron image segmentation

- hierarchical merging tree
  - nodes → regions of the image
  - labels → whether the region is dark = a subregion of a neuron; blue = a neuron region; red = union of several

(a)  
(b)  
(c)  
(d)  
(e)  
(f)
Chains and Trees: Human Pose Estimation

- Tree-structured model for articulated pose (Felzenszwalb and Huttenlocher, 2000), (Fischler and Elschlager, 1973)
- Body-part variables, states: discretized tuple \((x, y, s, \theta)\)
- \((x, y)\) position, \(s\) scale, and \(\theta\) rotation

Pic from Nowozin and Lampert 2011
Chains and Trees: Human Pose Estimation

Pic from Chen and Yuille NIPS 2014
Outline

1 Inference Tasks
   - First example (MRF) and the challenges

2 Chains and Trees

3 Loopy Graphs
   - Computing Marginals and Partition Function
   - Computing the MAP Exactly for Binary Label
   - Computing the MAP Approximately for Multilabel
Outline

1. Inference Tasks
   - First example (MRF) and the challenges

2. Chains and Trees

3. Loopy Graphs
   - Computing Marginals and Partition Function
   - Computing the MAP Exactly for Binary Label
   - Computing the MAP Approximately for Multilabel
Loopy Belief Propagation

- A natural extension of message passing on trees
- Every node sends messages to each neighbor at each iteration

Example: (blue: messages in iteration $t-1$, red: messages in $t$)
Loopy Belief Propagation

- A natural extension of message passing on trees
- Every node sends message to each neighbor at each iteration

Example: (blue: messages in iteration $t - 1$, red: messages in $t$)

- The message is identical to tree-message
  $$M_{v \rightarrow p}(y_p = \ell) = \sum_{y_v} e^{(-\theta_{v,p}(y_v,y_p))} \prod_{k \in nbd_v \setminus \{p\}} M_{k \rightarrow v}(y_v)$$
- Runs until messages converge.
- Compute partition function and marginals accordingly.
- (Limited) theoretical guarantees
Outline

1. Inference Tasks
   - First example (MRF) and the challenges

2. Chains and Trees

3. Loopy Graphs
   - Computing Marginals and Partition Function
   - Computing the MAP Exactly for Binary Label
   - Computing the MAP Approximately for Multilabel
Graphcut: Minimum Cut

The minimal cut problem

- A directed graph,
  - positive weights on all edges, $w(u,v) > 0$
  - one source (s) and a sink (t)
- A cut: a partition of vertices into two parts, $S \cup T = \mathcal{V}$, such that $s \in S$ and $t \in T$.
- The minimum cut: a cut with the minimal total weight,
  \[ \sum_{(u,v) \in \mathcal{E}, u \in S, v \in T} w(s,t) \]
- Max-flow min-cut theorem
- $O(|\mathcal{V}| |\mathcal{E}|)$ (Orlin STOC’13)
### Special case, binary labels (1,2)
- $\forall u \in \mathcal{V}$, $\mu_u(1) > 0$, $\mu_u(2) > 0$
- $\forall (u, v) \in \mathcal{E}$, $\theta_{uv}(1, 1) = \theta_{uv}(2, 2) = 0$, $\theta_{uv}(1, 2) = \theta_{uv}(2, 1) = w_{uv} > 0$

### Graph Construction
- Construct directed graph, $\mathcal{V} \cup \{s, t\}$
- An edge from $s$ to each original node (weight $\mu_u(2)$)
- An edge from each original node to $t$ (weight $\mu_u(1)$)
- A pair of edges for each original edge (both with weight $w_{uv}$)

### Min-cut = MAP:
- $u \in S$ if $y_u = 1$, cut weight = $E(y)$
Special case, binary labels (1,2)

- \( \forall u \in V, \mu_u(1) > 0, \mu_u(2) > 0 \)
- \( \forall (u, v) \in E, \theta_{uv}(1, 1) = \theta_{uv}(2, 2) = 0, \theta_{uv}(1, 2) = \theta_{uv}(2, 1) = w_{uv} > 0 \)

Construct directed graph, \( V \cup \{s, t\} \)

- An edge from \( s \) to each original node (weight \( \mu_u(2) \))
- An edge from each original node to \( t \) (weight \( \mu_u(1) \))
- A pair of edges for each original edge (both with weight \( w_{uv} \))

Min-cut = MAP: \( u \in S \) iff \( y_u = 1 \), cut weight = \( E(y) \)
Graphcut: min-cut for **exact** MAP computation

- More general, binary labels $(1,2)$ Regular
  \[ \forall (u, v) \in \mathcal{E}, \quad \theta_{uv}(1, 1) + \theta_{uv}(2, 2) \leq \theta_{uv}(1, 2) + \theta_{uv}(2, 1) \]
- Lemma: following modifications of parameters will not change
  \[
  \text{MAP} = \min_y E(y) = \min_y \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) + \sum_{i \in \mathcal{V}} \mu_i(y_i)
  \]

- **[M1]** $\mu_u'(\ast) = \mu_u(\ast) + c$; **[M2]** $\theta_{uv}'(\ast, \ast) = \theta_{uv}(\ast, \ast) + c$
- **[M3]** $\theta_{uv}'(\ast, 2) = \theta_{uv}(\ast, 2) + c$ and $\mu_v'(2) = \mu_v(2) - c$
- **[M4]** $\theta_{uv}'(2, \ast) = \theta_{uv}(2, \ast) + c$ and $\mu_u'(2) = \mu_u(2) - c$

\[
\begin{array}{c|cc}
\mu_u & \mu_v(1) & \mu_v(2) \\
\hline
\mu_u(1) & \theta_{uv}(1,1) & \theta_{uv}(1,2) \\
\mu_u(2) & \theta_{uv}(2,1) & \theta_{uv}(2,2) \\
\hline
\mu_u & & \\
\end{array}
\]
For each edge, modify parameters as follows:

<table>
<thead>
<tr>
<th>$\mu_{u}$</th>
<th>$\theta_{uv}$</th>
<th>$\mu_{v}$</th>
<th>$\theta_{uv}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mu_{u}(1)$</td>
<td>A</td>
<td>$\mu_{v}(1)$</td>
<td>A</td>
</tr>
<tr>
<td>$\mu_{u}(2)$</td>
<td>C</td>
<td>$\mu_{v}(2)$ - $C + D$</td>
<td>$\mu_{u}(2)$ - $A + C$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mu_{u}$</th>
<th>$\theta_{uv}$</th>
<th>$\mu_{v}$</th>
<th>$\theta_{uv}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mu_{u}(1)$</td>
<td>B+C-D</td>
<td>$\mu_{v}(1)$</td>
<td>0</td>
</tr>
<tr>
<td>$\mu_{u}(2)$</td>
<td>C</td>
<td>$\mu_{v}(2)$ - $C + D$</td>
<td>0</td>
</tr>
</tbody>
</table>

The modified parameters lead to a same MAP solution.

Nonzero entries in $\theta_{uv}$, $(B+C-A-D)/2$, is nonegative (regularity assumption).

Use $M_1$ to make all unaries, $\mu^\ast$, nonegative.

Solve using min-cut.
For each edge, modify parameters as follows

<table>
<thead>
<tr>
<th></th>
<th>μ_v(1)</th>
<th>μ_v(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>μ_u(1)</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>μ_u(2)</td>
<td>C</td>
<td>D</td>
</tr>
</tbody>
</table>

\[ θ_{uv} \]

\[ \mu_v - \mu_v(2) - C + D \]

The modified parameters lead to a same MAP solution

- Nonzero entries in \( θ_{uv} \), \((B+C-A-D)/2\), is nonnegative (regularity assumption)
- Use \( M1 \) to make all unaries, \( \mu_*(*) \), nonnegative
- Solve using min-cut.
Application: Binary Segmentation

Fig. 4.2 A natural image to be segmented. (Image source: http://pdphoto.org)

Fig. 4.3 Resulting foreground region.

Pic from Nowozin and Lampert 2011
Application: Binary Segmentation

$$MAP = \arg\min_y E(y) = \arg\min_y \omega \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) + \sum_{i \in \mathcal{V}} \mu_i(y_i)$$

- Contrast-sensitivity (Blake et al. 04):
  $$\theta_{ij}(\alpha, \beta) = const. \exp \left( \frac{-\|l_i - l_j\|^2}{const.} \right), \text{ if } \alpha \neq \beta$$
  $$= 0, \text{ if } \alpha = \beta$$

- $\mu_i$'s: unary terms coming from some probability distribution on colors (or more)
Inference Tasks
Chains and Trees
Loopy Graphs
Computing Marginals and Partition Function
Computing the MAP Exactly for Binary Label
Computing the MAP Approximately for Multilabel

\[ \text{MAP} = \arg\min_y E(y) = \arg\min_y \omega \sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) + \sum_{i \in V} \mu_i(y_i) \]

Fig. 4.4 Left: heatmap of unary potential values. Right: segmentation masks for large \( w \).

Fig. 4.5 Segmentation masks for medium and small \( w \).
Inference Tasks
Chains and Trees
Loopy Graphs
Computing Marginals and Partition Function
Computing the MAP Exactly for Binary Label
Computing the MAP Approximately for Multilabel

Application: GrabCut

Rother et al. SIGGRAPH 2004
Application: GrabCut

Rother et al. SIGGRAPH 2004
Inference Tasks
- First example (MRF) and the challenges

Chains and Trees

Loopy Graphs
- Computing Marginals and Partition Function
- Computing the MAP Exactly for Binary Label
- Computing the MAP Approximately for Multilabel
Multilabel Graphcut: approximate MAP computation

- $\mathcal{N}_t : \mathcal{Y} \rightarrow 2^\mathcal{Y}$, neighborhood system
- Optimization with respect to $\mathcal{N}_t(y)$ must be tractable:

$$y^{t+1} = \underset{y \in \mathcal{N}_t(y^t)}{\text{argmax}} \ g(x,y)$$

Pic from Nowozin and Lampert
Key Question: Neighborhood? Efficiently search through the neighborhood? balance between quality and efficiency

- $\text{Nbd} = \text{all configurations/labelings}$
- $\text{Nbd} = y$ itself
Key Question: Neighborhood? Efficiently search through the neighborhood?
balance between quality and efficiency

- $Nbd = \text{all configurations/labelings}$
- $Nbd = y$ itself

Two graphcut-based local search methods:
$\alpha$-$\beta$ swap and $\alpha$-expansion
Multilabel Graphcut: $\alpha$-$\beta$ swap

- Fix $\alpha$ and $\beta$, $N_t(y^t) :=$ all possible relabelings of $y^t$ in the $\alpha$- or $\beta$-labeled region using $\alpha$ or $\beta$.
- Each iteration, fix $\alpha$ and $\beta$, find the best in $N_t(y^t)$ as $y^{t+1}$.
- Continue until $y^{t+1} = y^t$. Stop at a local maximum.
Multilabel Graphcut: $\alpha-\beta$ swap, derivation

\[
y^{t+1} = \arg\min_{y \in \mathcal{N}_{\alpha, \beta}(y^t, \alpha, \beta)} \left[ \sum_{i \in V, \, y_i^t \notin \{\alpha, \beta\}} E_i(y_i^t; x) + \sum_{i \in V, \, y_i^t \in \{\alpha, \beta\}} E_i(y_i; x) \right. \\
+ \sum_{(i, j) \in E, \, y_i^t \notin \{\alpha, \beta\}, y_j^t \notin \{\alpha, \beta\}} E_{i,j}(y_i^t, y_j^t; x) \\
+ \sum_{(i, j) \in E, \, y_i^t \notin \{\alpha, \beta\}, y_j^t \in \{\alpha, \beta\}} E_{i,j}(y_i^t, y_j; x) \\
+ \sum_{(i, j) \in E, \, y_i^t \in \{\alpha, \beta\}, y_j^t \notin \{\alpha, \beta\}} E_{i,j}(y_i^t, y_j^t; x) + \left. \sum_{(i, j) \in E, \, y_i^t \in \{\alpha, \beta\}, y_j^t \in \{\alpha, \beta\}} E_{i,j}(y_i, y_j; x) \right].
\]

- **Constant**: drop out
- **Unary**: combine
- **Pairwise**: binary pairwise
Directed graph $G' = (V', E')$

$$V' = \{\alpha, \beta\} \cup \{i \in V : y_i \in \{\alpha, \beta\}\},$$

$$E' = \{(\alpha, i, t_i^\alpha) : \forall i \in V : y_i \in \{\alpha, \beta\}\} \cup \{(\beta, i, t_i^\beta) : \forall i \in V : y_i \in \{\alpha, \beta\}\} \cup \{(i, j, n_{i,j}) : \forall (i, j), (j, i) \in E : y_i, y_j \in \{\alpha, \beta\}\}.$$

Edge weights $t_i^\alpha$, $t_i^\beta$, and $n_{i,j}$

$$t_i^\alpha = \mu_i(\beta) + \sum_{(i, j) \in E, y_j \notin \{\alpha, \beta\}} \theta_{ij}(\beta, y_j)$$

$$t_i^\beta = \mu_i(\alpha) + \sum_{(i, j) \in E, y_j \notin \{\alpha, \beta\}} \theta_{ij}(\alpha, y_j)$$

$$n_{i,j} = \theta_{ij}(\alpha, \beta)$$

Semi-metric condition:

$$\theta_{ij}(y_i, y_j) = 0, \text{ if } y_i = y_j$$

$$\theta_{ij}(y_i, y_j) = \theta_{ij}(y_j, y_i) \geq 0, \text{ if } y_i \neq y_j$$
Multilabel Graphcut: $\alpha$ expansion

- An even bigger neighborhood (fix $\alpha$)
  $\mathcal{N}^t(y^t) := \text{all possible relabelings of } y^t \text{ such that each node is either switched to } \alpha \text{ or stay unchanged}$

- Each iteration can be solved by min-cut
- Converges to a strong local optimum
  $E(y^*) \leq 2cE(\text{MAP})$

$$c = \max_{(u,v) \in \mathcal{E}} \frac{\max_{\alpha \neq \beta} \theta_{uv}(\alpha, \beta)}{\min_{\alpha \neq \beta} \theta_{uv}(\alpha, \beta)}$$
Multilabel Graphcut: \( \alpha \) expansion

- What's the catch?
  - Semi-metric condition:
    \[
    \theta_{ij}(y_i, y_j) = 0, \text{ if } y_i = y_j \\
    \theta_{ij}(y_i, y_j) = \theta_{ij}(y_j, y_i) \geq 0, \text{ if } y_i \neq y_j
    \]
  - Triangle inequality:
    \[
    \theta_{ij}(\alpha, \beta) \leq \theta_{ij}(\alpha, \gamma) + \theta_{ij}(\gamma, \beta), \forall \alpha, \beta, \gamma
    \]

\[
\begin{array}{ccccccc}
    & & & & & & \theta_{ij} \\
    y_i & y_j & 1 & 2 & \cdots & L \\
    \hline
    1 & \theta_{ij}(1,1) & \theta_{ij}(1,2) & \cdots & \theta_{ij}(1,L) \\
    2 & \theta_{ij}(2,1) & \theta_{ij}(2,2) & \cdots & \theta_{ij}(2,L) \\
    \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
    L & \theta_{ij}(L,1) & \theta_{ij}(L,2) & \cdots & \theta_{ij}(L,L) \\
\end{array}
\]
How bad is it?

- Contrast-sensitive (Blake et al. 04):

\[
\theta_{ij}(\alpha, \beta) = \text{const.} \exp \left( -\frac{||l_i - l_j||^2}{\text{const.}} \right), \text{ if } \alpha \neq \beta
\]

\[
= 0, \text{ if } \alpha = \beta
\]

- All conditions satisfied

- 2-approximation of the optimal energy

\[
c = \max_{(u, v) \in E} \frac{\max_{\alpha \neq \beta} \theta_{uv}(\alpha, \beta)}{\min_{\alpha \neq \beta} \theta_{uv}(\alpha, \beta)} = 1
\]
Speedup: SuperPixels

Fig. 4.19 Input image: 500-by-375 pixels, for a total of 187,500 labeling decisions.

Fig. 4.20 The same image with 149 super-pixels and hence 149 decisions.

Pic from Nowozin and Lampert 2011
Other Methods:

- Junction Tree
- Tree-Reweighted Message Passing
- LP Relaxation
- Variational Inference
- ...

Next:
Advanced Inference Questions (Related to Topology and Geometry)
[Machine learning a probabilistic perspective, Kevin Murphy 2012] – comprehensive

[Structured learning and prediction in computer vision, Nowozin and Lampert 2011] – concise, intuitive, focus on computation, a good start

[An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, Boykov and Kolmogorov, PAMI 2004]

[What energy functions can be minimized via graph cuts? Kolmogorov and Zabin, PAMI 2004]

[Fast approximate energy minimization via graph cuts, Boykov et al. PAMI 2001]