| Optimization | P.H.S. Torr |
Max Flow

**Given:** a weighted directed graph with two distinguished nodes:
- source s,
- sink (destination) t

Interpret edge weights (all positive) as capacities

**Goal:** Find maximum flow from s to t

- Flow does not exceed capacity in any edge
- Flow at every vertex satisfies **equilibrium** [flow in equals flow out]

e.g. oil flowing through pipes, internet routing
LP formulation of maxflow problem

One variable per edge.
One inequality per edge, one equality per vertex.

maximize \[ x_{ts} \]
subject to the constraints
\[
\begin{align*}
x_{s1} & \leq 2 \\
x_{s2} & \leq 3 \\
x_{13} & \leq 3 \\
x_{14} & \leq 1 \\
x_{23} & \leq 1 \\
x_{24} & \leq 1 \\
x_{3t} & \leq 2 \\
x_{4t} & \leq 3
\end{align*}
\]

interpretation: \[ x_{ij} = \text{flow in edge } i-j \]
equilibrium constraints
\[
\begin{align*}
x_{ts} &= x_{s1} + x_{s2} \\
x_{s1} &= x_{13} + x_{14} \\
x_{s2} &= x_{23} + x_{24} \\
x_{13} + x_{23} &= x_{3t} \\
x_{14} + x_{24} &= x_{4t} \\
x_{3t} + x_{4t} &= x_{ts}
\end{align*}
\]
all \[ x_{ij} \geq 0 \]

Slide: Robert Sedgewick and Kevin Wayne
LP formulation of maxflow problem

One variable per edge.
One inequality per edge, one equality per vertex.

\[
\begin{align*}
\text{maximize} & & x_{ts} \\
\text{subject to the constraints} & & \\
& & x_{s1} \leq 2 \\
& & x_{s2} \leq 3 \\
& & x_{13} \leq 3 \\
& & x_{14} \leq 1 \\
& & x_{23} \leq 1 \\
& & x_{24} \leq 1 \\
& & x_{3t} \leq 2 \\
& & x_{4t} \leq 3 \\
\end{align*}
\]

interpretation: 
\( x_{ij} = \text{flow in edge } i-j \)

equilibrium constraints:
\[
\begin{align*}
x_{ts} &= x_{s1} + x_{s2} \\
x_{s1} &= x_{13} + x_{14} \\
x_{s2} &= x_{23} + x_{24} \\
x_{13} + x_{23} &= x_{3t} \\
x_{14} + x_{24} &= x_{4t} \\
x_{3t} + x_{4t} &= x_{ts} \\
\end{align*}
\]

all \( x_{ij} \geq 0 \)

capacity constraints

solution
\[
\begin{align*}
x_{s1} &= 2 \\
x_{s2} &= 2 \\
x_{13} &= 1 \\
x_{14} &= 1 \\
x_{23} &= 1 \\
x_{24} &= 1 \\
x_{3t} &= 2 \\
x_{4t} &= 2 \\
x_{ts} &= 4 \\
\end{align*}
\]

maxflow value

add dummy edge from \( t \) to \( s \)
Importance of Co-occurrence

Local features

Contextual features

Distance

Information

Slide courtesy A Torralba
Importance of Co-occurrence

We know there is a keyboard present in this scene even if we cannot see it clearly.

We know there is no keyboard present in this scene

... even if there is one indeed.
Need for Global Models

- Torralba and many others have shown the need for more global energy models of the scene to improve object recognition.

- Spatial models are typically handled by a Markov or Conditional Random Fields.
Toralba and many others have shown the need for more global energy models of the scene to improve object recognition.

Spatial models are typically handled by a Markov or Conditional Random Fields.
Markov Random Fields

Pairwise Markov Random Field: Model commonly used to represent images

Image Data Nodes ($d$)

Hidden Scene Nodes ($u$)

Sensor model

Prior model

Neighborhood $N_i$

$$p(u_i \mid u) = p(u_i \mid \{u_k\}), \quad u_k \in N_i$$
Example: Image Segmentation

\[ E(X) \]

\[ E: \{0, 1\}^n \rightarrow \mathbb{R} \]

\[ 0 \rightarrow fg \]

\[ 1 \rightarrow bg \]

\[ n = \text{number of pixels} \]

[Boykov and Jolly '01] [Blake et al. '04] [Rother, Kolmogorov and Blake '04]
Example: Image Segmentation

\[ E(X) = \sum c_i x_i \]

Pixel Colour

Unary Cost \( c_i \)

Dark (negative)  Bright (positive)

\[ E: \{0, 1\}^n \rightarrow \mathbb{R} \]

\[
\begin{align*}
0 & \rightarrow \text{fg} \\
1 & \rightarrow \text{bg}
\end{align*}
\]

\( n = \) number of pixels

[Boykov and Jolly `01] [Blake et al. `04] [Rother, Kolmogorov and Blake `04]
Example: Image Segmentation

\[ E(X) = \sum c_i x_i \]

Pixel Colour

 Unary Cost \( (c_i) \)

Dark (negative)   Bright (positive)

\[ E : \{0,1\}^n \rightarrow \mathbb{R} \]

\[ 0 \rightarrow \text{fg} \]

\[ 1 \rightarrow \text{bg} \]

\( n = \text{number of pixels} \)

\[ x^* = \arg \min E(x) \]

[Boykov and Jolly '01] [Blake et al. '04] [Rother, Kolmogorov and Blake '04]
Example: Image Segmentation

\[ E(X) = \sum c_i x_i \quad + \quad \sum d_{ij} |x_i - x_j| \]

Pixel Colour \quad \text{Smoothness Prior}

\[ E: \{0,1\}^n \rightarrow \mathbb{R} \]
\[ 0 \rightarrow \text{fg} \]
\[ 1 \rightarrow \text{bg} \]
\[ n = \text{number of pixels} \]

Discontinuity Cost \( (d_{ij}) \)

[Boykov and Jolly ‘01] [Blake et al. ‘04] [Rother, Kolmogorov and Blake ‘04]
Example: Image Segmentation

\[ E(X) = \sum c_i x_i + \sum d_{ij} x_i (1 - x_j) + d_{ij} x_j (1 - x_i) \]

- Pixel Colour
- Smoothness Prior

-E: \{0, 1\}^n \rightarrow R
- 0 \rightarrow fg
- 1 \rightarrow bg

n = number of pixels

[Boykov and Jolly ‘01] [Blake et al. ‘04] [Rother, Kolmogorov and Blake ‘04]
Example: Image Segmentation

\[ E(x) = \sum_i c_i x_i + \sum_{i,j} c_{ij} x_i(1-x_j) \]

E: \{0,1\}^n \rightarrow \mathbb{R}
0 \rightarrow fg
1 \rightarrow bg

\( n = \text{number of pixels} \)

Global Minimum (\( x^* \))

\[ x^* = \arg \min_x E(x) \]

How to minimize \( E(x) \)?
Outline of this Part

The st-mincut problem

Connection between st-mincut and energy minimization?

What problems can we solve using st-mincut?

st-mincut based Move algorithms

Recent Advances and Open Problems
Outline of this Part

The st-mincut problem

Connection between st-mincut and energy minimization?

What problems can we solve using st-mincut?

st-mincut based Move algorithms

Recent Advances and Open Problems
The st-Mincut Problem

Graph \((V, E, C)\)

Vertices \(V = \{v_1, v_2, \ldots, v_n\}\)

Edges \(E = \{(v_1, v_2), \ldots\}\)

Costs \(C = \{c_{(1, 2)}, \ldots\}\)
The st-mincut problem

Connection between st-mincut and energy minimization?

What problems can we solve using st-mincut?

st-mincut based Move algorithms

Recent Advances and Open Problems
St-mincut and Energy Minimization

Minimizing a Quadratic Pseudoboolean function $E(x)$

Functions of boolean variables

$$E: \{0,1\}^n \rightarrow \mathbb{R}$$

$E(x) = \sum_i c_i x_i + \sum_{i,j} c_{ij} x_i(1-x_j)$

Pseudoboolean?

Polynomial time st-mincut algorithms require non-negative edge weights

$f(x) = \sum_i c_i x_i + \sum_{i,j} c_{ij} x_i(1-x_j)$
Given a graph:

1. Any st-cut corresponds to an assignment of $x$
2. Construct a cost function $E(x)$ equal to the cost of the cut for an assignment of $x$
3. Use this to show that the min-cut is obtained

Later show how to construct a graph corresponding the the cost $f(x)$
Graph Cuts

Consider the case of two segments.
Question: Why (or when) is graph cut good for this problem?
Graph Construction

$E(a_1, a_2)$

Source (0)

Sink (1)

$a_1$  

$a_2$
Graph Construction

\[ E(a_1, a_2) = 2a_1 \]
Graph Construction

\[ E(a_1, a_2) = 2a_1 + 5\bar{a}_1 \]
Graph Construction

\[ E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 \]
Graph Construction

$E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2$
Graph Construction

\[ E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]
Graph Construction

\[ E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]
Graph Construction

$E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2$

Sink (1)  
Source (0)

Cost of cut = 11

$a_1 = 1 \quad a_2 = 1$

$E(1,1) = 11$
$E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2$

Graph Construction

$E(1,0) = 8$

st-mincut cost = 8

$a_1 = 1 \quad a_2 = 0$
Energy Function Reparameterization

Two functions $E_1$ and $E_2$ are reparameterizations if

$$E_1(x) = E_2(x) \text{ for all } x$$

For instance:

$E_1(a_1) = 1 + 2a_1 + 3\tilde{a}_1$

$E_2(a_1) = 3 + \tilde{a}_1$

<table>
<thead>
<tr>
<th>$a_1$</th>
<th>$\tilde{a}_1$</th>
<th>$1 + 2a_1 + 3\tilde{a}_1$</th>
<th>$3 + \tilde{a}_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

use $\tilde{a} = (1 - a)$
Flow and Reparametrization

\[ E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]
Flow and Reparametrization

\[ E(a_1, a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]

2a_1 + 5\bar{a}_1

= 2(a_1 + \bar{a}_1) + 3\bar{a}_1

= 2 + 3\bar{a}_1
Flow and Reparametrization

\[ E(a_1, a_2) = 2 + 3\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]

\[ 2a_1 + 5\bar{a}_1 = 2(a_1 + \bar{a}_1) + 3\bar{a}_1 = 2 + 3\bar{a}_1 \]
Flow and Reparametrization

\[ E(a_1, a_2) = 2 + 3\overline{a}_1 + 9a_2 + 4\overline{a}_2 + 2a_1\overline{a}_2 + \overline{a}_1a_2 \]

\[ 9a_2 + 4\overline{a}_2 \]
\[ = 4(a_2 + \overline{a}_2) + 5\overline{a}_2 \]
\[ = 4 + 5\overline{a}_2 \]
Flow and Reparametrization

\[ E(a_1, a_2) = 2 + 3\bar{a}_1 + 5a_2 + 4 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]

\[ 9a_2 + 4\bar{a}_2 = 4(a_2 + \bar{a}_2) + 5\bar{a}_2 = 4 + 5\bar{a}_2 \]
Flow and Reparametrization

\[ E(a_1, a_2) = 6 + 3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]
Flow and Reparametrization

\[ E(a_1, a_2) = 6 + 3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]
Flow and Reparametrization

\[ E(a_1, a_2) = 6 + 3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2 \]

\[
3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 \\
= 2(\bar{a}_1 + a_2 + a_1\bar{a}_2) + \bar{a}_1 + 3a_2 \\
= 2(1 + \bar{a}_1a_2) + \bar{a}_1 + 3a_2
\]

\[ F1 = \bar{a}_1 + a_2 + a_1\bar{a}_2 \]
\[ F2 = 1 + \bar{a}_1a_2 \]

<table>
<thead>
<tr>
<th>(a_1)</th>
<th>(a_2)</th>
<th>(F1)</th>
<th>(F2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Flow and Reparametrization

\[ E(a_1, a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1a_2 \]

\[ 3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 \]
\[ = 2(\bar{a}_1 + a_2 + a_1\bar{a}_2) + \bar{a}_1 + 3a_2 \]
\[ = 2(1 + \bar{a}_1a_2) + \bar{a}_1 + 3a_2 \]

\[ F1 = \bar{a}_1 + a_2 + a_1\bar{a}_2 \]
\[ F2 = 1 + \bar{a}_1a_2 \]

<table>
<thead>
<tr>
<th>(a_1)</th>
<th>(a_2)</th>
<th>(F1)</th>
<th>(F2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Flow and Reparametrization

\[ E(a_1, a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1a_2 \]

No more augmenting paths possible
Flow and Reparametrization

\[ E(a_1, a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1 a_2 \]

Total Flow

Inference of the optimal solution becomes trivial because the bound is tight
Flow and Reparametrization

\[ E(a_1, a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1a_2 \]

Residual Graph (positive coefficients)

Total Flow bound on the optimal solution

Inference of the optimal solution becomes trivial because the bound is tight

Source (0)

\[ \text{st-mincut cost} = 8 \]

Sink (1)

\[ a_1 = 1 \quad a_2 = 0 \]

\[ E(1,0) = 8 \]
Example: Image Segmentation

\[ E(x) = \sum_i c_i x_i + \sum_{i,j} c_{ij} x_i(1-x_j) \]

\[ E: \{0,1\}^n \rightarrow \mathbb{R} \]

\[ 0 \rightarrow fg \]

\[ 1 \rightarrow bg \]

Global Minimum (\( x^* \))

\[ x^* = \arg \min_x E(x) \]

How to minimize \( E(x) \)?
Graph *g;

For all pixels p

/* Add a node to the graph */
nodeID(p) = g->add_node();

/* Set cost of terminal edges */
set_weights(nodeID(p), fgCost(p), bgCost(p));

end

for all adjacent pixels p,q

add_weights(nodeID(p), nodeID(q), cost);
end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
Graph *g;

For all pixels p

/* Add a node to the graph */
nodeID(p) = g->add_node();

/* Set cost of terminal edges */
set_weights(nodeID(p), fgCost(p), bgCost(p));
end

for all adjacent pixels p,q

add_weights(nodeID(p), nodeID(q), cost);
end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
Graph *g;

For all pixels p

    /* Add a node to the graph */
    nodeID(p) = g->add_node();

    /* Set cost of terminal edges */
    set_weights(nodeID(p), fgCost(p), bgCost(p));

end

for all adjacent pixels p,q

    add_weights(nodeID(p), nodeID(q), cost(p,q));

end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
Graph *g;

For all pixels p

    /* Add a node to the graph */
    nodeID(p) = g->add_node();

    /* Set cost of terminal edges */
    set_weights(nodeID(p), fgCost(p), bgCost(p));

end

for all adjacent pixels p,q

    add_weights(nodeID(p), nodeID(q), cost(p,q));

end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)

\[ a_1 = \text{bg} \quad a_2 = \text{fg} \]
Image Segmentation in Video

$$E(x) \rightarrow \text{st-cut} \rightarrow x^*$$

- **Image**
- **Flow**
- **Global Optimum**
Map $f(x)$ onto network flow

Construct a network so that a cut corresponds to an assignment of $x_i$

\[
f(x) = \sum_{i=1}^{n} \{m_i(x_i) + \sum_{j \in \mathcal{N}(i)} \phi_i(x_i, x_j)\}
\]
For unary terms only:

\[ x_1 = 1, x_2 = 1 \]

\[ \rightarrow f(x) = m_1(1) + m_2(1) \]
Now, include pair wise term ...

Sub-modular constraint: flows must be positive. So, $B+C-A-D \geq 0$
\[
\begin{align*}
x_1 = 1, x_2 = 1 &\implies f(x) = m_1(1) + m_2(1) + D - A
\end{align*}
\]
\[ x_1 = 0, \quad x_2 = 1 \]

\[ \rightarrow f(x) = m_1(0) + m_2(1) + D - C + B + C - A - D \]

\[ = m_1(0) + m_2(1) + B - A \]
Summary: optimization using graph cuts

Stage 1: map the cost function $f(x)$ onto a flow network so that a cut of the network corresponds to the cost $f(x)$

Stage 2: compute the min-cut of the network using an augmented path algorithm
Applications

Optimization of binary image graph using graph-cuts:

1. Image cut-out and editing
2. Image quilting
3. Interactive Digital Photo-montage
1. Image cut-out by binary segmentation
Object - white, Background - green/grey

Each vertex corresponds to a pixel

Edges define a 4-neighbourhood grid graph

Assign a label to each vertex from \( L = \{\text{obj}, \text{bkg}\} \)
Graph $G = (V,E)$

Cost of a labelling $f : V \rightarrow L$

Object - white, Background - green/grey

Per Vertex Cost

Cost of label ‘obj’ low
Cost of label ‘bkg’ high
Graph $G = (V,E)$

Cost of a labelling $f : V \rightarrow L$

Cost of label ‘obj’ high    Cost of label ‘bkg’ low

Object - white, Background - green/grey

Per Vertex Cost

UNARY COST
Graph $G = (V,E)$

Cost of a labelling $f : V \to L$

Object - white, Background - green/grey

Cost of same label low

Cost of different labels high
Graph $G = (V,E)$

Cost of a labelling $f : V \rightarrow L$

Object - white, Background - green/grey

Cost of different labels low

Per Edge Cost

PAIRWISE COST
Graph $G = (V,E)$

Problem: Find the labelling with minimum cost $f^*$

Object - white, Background - green/grey
\[ f(x) = \sum_{i=1}^{n} \{ m_i(x_i) + \sum_{j \in \mathcal{N}(i)} \phi_i(x_i, x_j) \} \]

- \( x_i = 1 \) for foreground pixels, \( x_i = 0 \) for background
- \( m_i(x_i) \) is likelihood that pixel at \( i \) is foreground (if \( x_i = 1 \)), or background (if \( x_i = 0 \)), e.g. using colour histogram of seed regions
- \( \phi_i(x_i, x_j) \) penalizes a change of state:

\[
\phi_i(x_i, x_j) = \begin{cases} 
0 & \text{if } x_i = x_j \\
\gamma e^{-\beta (I_i - I_j)^2} & \text{if } x_i \neq x_j.
\end{cases}
\]
Application: foreground/background image segmentation

use seed pixels to learn colour distribution
Image editing …

Available in Microsoft Office …
Image Quilting
Example: Texture Synthesis

Goal of Texture Synthesis: create new samples of a given texture

Many applications: virtual environments, hole-filling, texturing surfaces
Input texture

Random placement of blocks

Neighboring blocks constrained by overlap

Minimal error boundary cut
Algorithm

• Pick size of block and size of overlap
• Synthesize blocks in raster order

• Search input texture for block that satisfies overlap constraints (above and left)
• Paste new block into resulting texture
  > use graph cuts to compute minimal error boundary cut

Efros & Freeman 2001, Kwatra et al. 2003
Minimal error boundary

overlapping blocks

vertical boundary

\[ \text{overlap error} \leq 2 \]

\[ \text{min. error boundary} \]
Interactive Digital Photomontage
Use graph-cuts to quilt images
Minimizing Energy Functions

General Energy Functions
• NP-hard to minimize
• Only approximate minimization possible

Easy energy functions
• Solvable in polynomial time
• Submodular $\sim O(n^6)$
Submodular Set Functions

Let $E = \{a_1, a_2, \ldots, a_n\}$ be a set.

Set function $f : 2^{\mid E \mid} \rightarrow \mathbb{R}$

$2^{\mid E \mid} = \# \text{subsets of } E$
Submodular Set Functions

Let \( E = \{a_1, a_2, \ldots, a_n\} \) be a set

Set function \( f : 2^{|E|} \rightarrow \mathbb{R} \) is submodular if

\[
f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \text{ for all } A, B \subseteq E
\]

\( 2^{|E|} = \# \text{subsets of } E \)

Important Property

Sum of two submodular functions is submodular
Minimizing Submodular Functions

Minimizing general submodular functions
- $O(n^5 Q + n^6)$ where $Q$ is function evaluation time
  [Orlin, IPCO 2007]

Symmetric submodular functions
- $E(x) = E(1 - x)$
- $O(n^3)$  [Queyranne 1998]

Quadratic pseudoboolean
- Can be transformed to st-mincut
- One node per variable  [ $O(n^3)$ complexity]
- Very low empirical running time
Submodular Pseudoboolean Functions

Function defined over boolean vectors $x = \{x_1, x_2, \ldots, x_n\}$

**Definition:**

• All functions for one boolean variable ($f: \{0,1\} \rightarrow \mathbb{R}$) are submodular.

• A function of two boolean variables ($f: \{0,1\}^2 \rightarrow \mathbb{R}$) is submodular if $f(0,1) + f(1,0) \geq f(0,0) + f(1,1)$

• A general pseudoboolean function $f: 2^n \rightarrow \mathbb{R}$ is submodular if all its projections $f^p$ are submodular i.e.

$$f^p(0,1) + f^p(1,0) \geq f^p(0,0) + f^p(1,1)$$
Quadratic Submodular Pseudoboolean Functions

\[ E(x) = \sum_i \theta_i (x_i) + \sum_{i,j} \theta_{ij} (x_i, x_j) \]

For all \( ij \)

\[ \theta_{ij}(0,1) + \theta_{ij}(1,0) \geq \theta_{ij}(0,0) + \theta_{ij}(1,1) \]
Quadratic Submodular Pseudoboolean Functions

\[ E(x) = \sum_i \theta_i(x_i) + \sum_{i,j} \theta_{ij}(x_i, x_j) \]

For all \( ij \)
\[ \theta_{ij}(0,1) + \theta_{ij}(1,0) \geq \theta_{ij}(0,0) + \theta_{ij}(1,1) \]

Equivalent (transformable)

\[ E(x) = \sum_i c_i x_i + \sum_{i,j} c_{ij} x_i(1-x_j) \]

i.e. All submodular QPBFs are st-mincut solvable
How are they equivalent?

\[ A = \theta_{ij}(0,0) \quad B = \theta_{ij}(0,1) \quad C = \theta_{ij}(1,0) \quad D = \theta_{ij}(1,1) \]

\[ \theta_{ij}(x_i, x_j) = \theta_{ij}(0,0) + (\theta_{ij}(1,1) - \theta_{ij}(0,0)) x_i + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) x_j + (\theta_{ij}(1,0) + \theta_{ij}(0,1) - \theta_{ij}(0,0) - \theta_{ij}(1,1)) (1-x_i) x_j \]

\[ B+C-A-D \geq 0 \] is true from the submodularity of \( \theta_{ij} \)
How are they equivalent?

\[ A = \theta_{ij}(0,0) \quad B = \theta_{ij}(0,1) \quad C = \theta_{ij}(1,0) \quad D = \theta_{ij}(1,1) \]

\[ \theta_{ij}(x_i, x_j) = \theta_{ij}(0,0) + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) x_i + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) x_j 
+ (\theta_{ij}(1,0) + \theta_{ij}(0,1) - \theta_{ij}(0,0) - \theta_{ij}(1,1)) (1-x_i) x_j \]

\[ B + C - A - D \geq 0 \] is true from the submodularity of \( \theta_{ij} \)
How are they equivalent?

\[
\begin{align*}
A &= \theta_{ij}(0,0) \\
B &= \theta_{ij}(0,1) \\
C &= \theta_{ij}(1,0) \\
D &= \theta_{ij}(1,1)
\end{align*}
\]

\[
\begin{array}{c|cc|c}
\hline
& x_j & \hline
\hline
x_i & 0 & 1 & \hline
\hline
0 & A & B & \hline
1 & C & D & \hline
\hline
\end{array}
\]

\[
\begin{array}{c|cc|c}
\hline
& x_j & \hline
\hline
0 & 0 & 0 & \hline
1 & C-A & C-A & \hline
\hline
\end{array}
\]

if \( x_1 = 1 \) add \( C-A \)

\[
\begin{array}{c|cc|c}
\hline
& x_j & \hline
\hline
0 & 0 & D-C & \hline
1 & 0 & D-C & \hline
\hline
\end{array}
\]

if \( x_2 = 1 \) add \( D-C \)

\[
\begin{array}{c|cc|c}
\hline
& x_j & \hline
\hline
0 & 0 & B+C-A-D & \hline
1 & 0 & 0 & \hline
\hline
\end{array}
\]

\[
\theta_{ij}(x_i, x_j) = \theta_{ij}(0,0) + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) x_i + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) x_j + (\theta_{ij}(1,0) + \theta_{ij}(0,1) - \theta_{ij}(0,0) - \theta_{ij}(1,1)) (1-x_i) x_j
\]

\[
B+C-A-D \geq 0 \text{ is true from the submodularity of } \theta_{ij}
\]
How are they equivalent?

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>1</td>
<td>C</td>
<td>D</td>
</tr>
</tbody>
</table>

- $A = \theta_{ij}(0,0)$
- $B = \theta_{ij}(0,1)$
- $C = \theta_{ij}(1,0)$
- $D = \theta_{ij}(1,1)$

$x_i \times x_j = A + C-A \times x_i + D-C \times x_j$

$\theta_{ij}(x_i, x_j) = \theta_{ij}(0,0) + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) \times x_i + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) \times x_j$

$B + C - A - D \geq 0$ is true from the submodularity of $\theta_{ij}$
How are they equivalent?

\[ A = \theta_{ij}(0,0) \quad B = \theta_{ij}(0,1) \quad C = \theta_{ij}(1,0) \quad D = \theta_{ij}(1,1) \]

\[
\begin{align*}
\theta_{ij}(x_i, x_j) &= \theta_{ij}(0,0) \\
&\quad + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) x_i \\
&\quad + (\theta_{ij}(1,0) - \theta_{ij}(0,0)) x_j \\
&\quad + (\theta_{ij}(1,0) + \theta_{ij}(0,1) - \theta_{ij}(0,0) - \theta_{ij}(1,1)) (1-x_i) x_j
\end{align*}
\]

\[ B + C - A - D \geq 0 \] is true from the submodularity of \( \theta_{ij} \)
Quadratic Submodular Pseudoboolean Functions

\[ E(x) = \sum_{i} \theta_{i}(x_{i}) + \sum_{i,j} \theta_{ij}(x_{i},x_{j}) \]

For all \( ij \)

\[ \theta_{ij}(0,1) + \theta_{ij}(1,0) \geq \theta_{ij}(0,0) + \theta_{ij}(1,1) \]

\( x \in \{0,1\}^{n} \)

Equivalent (transformable)
Minimizing Non-Submodular Functions

Commonly used method is to solve a relaxation of the problem

\[ E(x) = \sum_i \theta_i(x_i) + \sum_{i,j} \theta_{ij}(x_i, x_j) \]

\[ \theta_{ij}(0,1) + \theta_{ij}(1,0) \leq \theta_{ij}(0,0) + \theta_{ij}(1,1) \text{ for some } ij \]

Minimizing general non-submodular functions is NP-hard.

[Slide credit: Carsten Rother]
Multi-label to Pseudo-boolean

So what is the problem?

Multi-label Problem \( E_m(y_1, y_2, \ldots, y_n) \)  
Binary label Problem \( E_b(x_1, x_2, \ldots, x_m) \)

such that:

Let \( Y \) and \( X \) be the set of feasible solutions, then

1. For each binary solution \( x \in X \) with finite energy there exists exactly one multi-label solution \( y \in Y \)

\(-\) One-One encoding function \( T:X \rightarrow Y \)

2. \( \arg\min E_m(y) = T(\arg\min E_b(x)) \)
Multi-label to Pseudo-boolean

- **Popular encoding scheme**
  [Roy and Cox ’98, Ishikawa ’03, Schlesinger & Flach ’06]

\[
\begin{align*}
\{y_1 = 1\} & \leftrightarrow \{x_1 = 1, x_2 = 1, x_3 = 1\}, \\
\{y_1 = 2\} & \leftrightarrow \{x_1 = 0, x_2 = 1, x_3 = 1\}, \\
\{y_1 = 3\} & \leftrightarrow \{x_1 = 0, x_2 = 0, x_3 = 1\}, \\
\{y_1 = 4\} & \leftrightarrow \{x_1 = 0, x_2 = 0, x_3 = 0\}.
\end{align*}
\]
Multi-label to Pseudo-boolean

- **Popular encoding scheme**
  [Roy and Cox '98, Ishikawa '03, Schlesinger & Flach '06]

Ishikawa's result:

\[
E(y) = \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{ij}(y_i, y_j)
\]

\(y \in \text{Labels } L = \{l_1, l_2, \ldots, l_k\}\)

\(\theta_{ij}(y_i, y_j) = g(|y_i - y_j|)\)

Convex Function

\[g(|y_i - y_j|)\]
Multi-label to Pseudo-boolean

- Popular encoding scheme
  [Roy and Cox '98, Ishikawa '03, Schlesinger & Flach '06]

\[
\begin{align*}
\{y_1 = 1\} & \leftrightarrow \{x_1 = 1, x_2 = 1, x_3 = 1\}, \\
\{y_1 = 2\} & \leftrightarrow \{x_1 = 0, x_2 = 1, x_3 = 1\}, \\
\{y_1 = 3\} & \leftrightarrow \{x_1 = 0, x_2 = 0, x_3 = 1\}, \\
\{y_1 = 4\} & \leftrightarrow \{x_1 = 0, x_2 = 0, x_3 = 0\}.
\end{align*}
\]

Schlesinger & Flach '06:

\[
E(y) = \sum_i \theta_i(y_i) + \sum_{i,j} \theta_{ij}(y_i, y_j)
\]

\[
y \in \text{Labels } L = \{l_1, l_2, \ldots, l_k\}
\]

\[
\theta_{ij}(l_{i+1}, l_j) + \theta_{ij}(l_i, l_{j+1}) \geq \theta_{ij}(l_i, l_j) + \theta_{ij}(l_{i+1}, l_{j+1})
\]

Covers all Submodular multi-label functions

More general than Ishikawa
Multi-label to Pseudo-boolean

Problems

Applicability

• Only solves restricted class of energy functions
• Cannot handle Potts model potentials

Computational Cost

• Very high computational cost
• Problem size = |Variables| x |Labels|
• Gray level image denoising (1 Mpixel image)
  (~2.5 x 10^8 graph nodes)
Outline of the Tutorial

The st-mincut problem

Connection between st-mincut and energy minimization?

What problems can we solve using st-mincut?

st-mincut based Move algorithms

Recent Advances and Open Problems
Graph cuts for multi label

Binary Potts energy (Ising model)

Multi-label Potts energy

$s-t$ graph cuts
(Greig at.al. 1989)

Multi-way graph cuts
(BVZ 1998,2001)
Multi-way graph cuts
Multi-way graph cuts

Equivalent to minimization of the Potts energy of labeling $L$

$$E(L) = \sum_p -D_p(L_p) + \sum_{pq \in N} w_{pq} \cdot \delta_{L_p \neq L_q}$$
**a-expansion move**

Basic idea: break multi-way cut computation into a sequence of binary $s$-$t$ cuts

Iteratively, each label competes with the other labels for space in the image
St-mincut based Move algorithms

\[ E(x) = \sum_{i} \theta_{i}(x_{i}) + \sum_{i,j} \theta_{ij}(x_{i},x_{j}) \]

\[ x \in \text{Labels } L = \{ l_1, l_2, \ldots, l_k \} \]

Commonly used for solving non-submodular multi-label problems

Extremely efficient and produce good solutions

Not Exact: Produce local optima
Move Making Algorithms

Energy

Solution Space
Move Making Algorithms

- Current Solution
- Search Neighbourhood
- Optimal Move

Energy

Solution Space
Computing the Optimal Move

- **Key Property**: Move Space
  - Bigger move space
  - Better solutions
  - Finding the optimal move hard

- **Solution Space**
- **Current Solution**
- **Search Neighbourhood**
- **Optimal Move**
Moves using Graph Cuts

Expansion and Swap move algorithms
[Boykov Veksler and Zabih, PAMI 2001]

- Makes a series of changes to the solution (moves)
- Each move results in a solution with smaller energy

Space of Solutions: $L^n$

Move Space: $2^N$

Current Solution
Search Neighbourhood

$N$: Number of Variables
$L$: Number of Labels
Moves using Graph Cuts

Expansion and Swap move algorithms
[Boykov Veksler and Zabih, PAMI 2001]

- Makes a series of changes to the solution (moves)
- Each move results in a solution with smaller energy

Current Solution

Construct a move function

Minimize move function to get optimal move

How to minimize move functions?

Move to new solution
General Binary Moves

\[ x = t \, x^1 + (1 - t) \, x^2 \]

Minimize over move variables $t$ to get the optimal move

\[ E_m(t) = E(t \, x^1 + (1 - t) \, x^2) \]

Move energy is a submodular QPBF (Exact Minimization Possible)

Boykov, Veksler and Zabih, PAMI 2001
Swap Move

- Variables labeled $\alpha, \beta$ can swap their labels

[Boykov, Veksler, Zabih]
Swap Move

- Variables labeled $\alpha, \beta$ can swap their labels

Swap Sky, House

[Boykov, Veksler, Zabih]
Swap Move

- Variables labeled $\alpha, \beta$ can swap their labels

Move energy is submodular if:

- Unary Potentials: Arbitrary
- Pairwise potentials: Semimetric

$$\theta_{ij}(l_a, l_b) \geq 0 \quad \theta_{ij}(l_a, l_b) = 0 \quad a = b$$

Examples: Potts model, Truncated Convex

[Boykov, Veksler, Zabih]
Expansion Move

- Variables take label $\alpha$ or retain current label
Expansion Move

- Variables take label $\alpha$ or retain current label

Status: Expand House, Sky, Tree
a-expansion moves

In each a-expansion a given label “a” grabs space from other labels

For each move we choose expansion that gives the largest decrease in the energy: **binary optimization problem**
Expansion Move

- Variables take label $\alpha$ or retain current label

Move energy is submodular if:

- Unary Potentials: Arbitrary
- Pairwise potentials: Metric

\[ \theta_{ij}(l_a, l_b) + \theta_{ij}(l_b, l_c) \geq \theta_{ij}(l_a, l_c) \]

Examples: Potts model, Truncated linear

Cannot solve truncated quadratic

[Boykov, Veksler, Zabih]
General Binary Moves

Minimize over move variables $t$

\[ x = t x^1 + (1-t) x^2 \]

<table>
<thead>
<tr>
<th>Move Type</th>
<th>First Solution</th>
<th>Second Solution</th>
<th>Guarantee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expansion</td>
<td>Old solution</td>
<td>All alpha</td>
<td>Metric</td>
</tr>
<tr>
<td>Fusion</td>
<td>Any solution</td>
<td>Any solution</td>
<td>( \times )</td>
</tr>
</tbody>
</table>

Move functions can be non-submodular!!
Solving Continuous Problems using Fusion Move

\[ x = t \, x^1 + (1-t) \, x^2 \]

\( x^1, x^2 \) can be continuous

(Lempitsky et al. CVPR08, Woodford et al. CVPR08)
Range Moves

Move variables can be multi-label

\[ x = (t==1) x^1 + (t==2) x^2 + \ldots + (t==k) x^k \]

Optimal move found out by using the Ishikawa

Useful for minimizing energies with truncated convex pairwise potentials

\[ \theta_{ij} (y_i, y_j) = \min(|y_i - y_j|, T) \]
Move Algorithms for Solving Higher Order Energies

Higher order functions give rise to higher order move energies

Move energies for certain classes of higher order energies can be transformed to QPBFs.

\[ E(x) = \sum \theta_i(x_i) + \sum_{i,j} \theta_{ij}(x_i,x_j) + \sum_c \theta_c(x_c) \]

\[ x \in \text{Labels} \ L = \{l_1, l_2, \ldots, l_k\} \]

\[ \text{Clique} \ c \subseteq V \]

[Kohli, Kumar and Torr, CVPR07]  [Kohli, Ladicky and Torr, CVPR08]