Graphical Models
Discrete Inference and Learning

MVA
2022 – 2023

http://thoth.inrialpes.fr/~alahari/disinflearn
Lecturers

Karteek Alahari

Demian Wassermann

Email: <firstname>.lastname@inria.fr
Organization

• 7 lectures of 3 hours each
  – Today + 24/1, 31/1, 7/2, 28/2, 7/3, 14/3

• 13:45 – 17:00 (except today) with a short break or two

• Last lecture: 14th March

http://thoth.inrialpes.fr/~alahari/disinflearn
Requirements

• Solid understanding of mathematical models
  – Linear algebra
  – Integral transforms
  – Differential equations

• Ideally, a basic course in discrete optimization
Topics covered

• Basic concepts, Bayesian networks, Markov random fields
• Inference algorithms: belief propagation, tree-reweighted message passing, graph cuts, move-making algorithms, Parameter learning
• Deep learning in graphical models, graph neural networks, other recent advances
• Causality
Evaluation

• Projects

• In groups of at most 3 people

• Report and presentation – Date TBD

• Topics: your own or see list on 25/1

• Bonus points for excellent class participation
What you will learn?

• Fundamental methods

• Real-world applications

• Also, pointers to using these methods in your work
Your tasks

• Following the lectures and participating actively

• Reading the literature

• Doing well in the project
Graphical Models
Discrete Inference and Learning
Lecture 1

MVA
2022 – 2023

http://thoth.inrialpes.fr/~alahari/disinflearn

Slides based on material from Stephen Gould, Pushmeet Kohli, Nikos Komodakis, M. Pawan Kumar, Carsten Rother, Daphne Koller, Dhruv Batra
Graphical Models?
What this class is about?

• Making **global** predictions from **local** observations

  **Inference**

• Learning such models from large quantities of data

  **Learning**
Motivation

• Consider the example of medical diagnosis

- Predisposing factors
- Symptoms
- Test results
- Diseases
- Treatment outcomes

Slide inspired by PGM course, Daphne Koller
Motivation

• A very different example: image segmentation

 Millions of pixels
Colours / features

Pixel labels
{building, grass, cow, sky}

e.g., [He et al., 2004; Shotton et al., 2006; Gould et al., 2009]

Slide inspired by PGM course, Daphne Koller
Motivation

• What do these two problems have in common?
Motivation

• What do these two problems have in common?
  
  – Many variables
  
  – Uncertainty about the correct answer

Graphical Models (or Probabilistic Graphical Models) provide a framework to address these problems
(Probabilistic) Graphical Models

• First, it is a model: a declarative representation
• Can also define the model
  – with domain knowledge
  – from data

Slide inspired by PGM course, Daphne Koller
(Probabilistic) Graphical Models

• Why probabilistic?
• To model uncertainty
• Uncertainty due to:
  – Partial knowledge of state of the world
  – Noisy observations
  – Phenomena not observed by the model
  – Inherent stochasticity

Slide inspired by PGM course, Daphne Koller
(Probabilistic) Graphical Models

• Probability theory provides
  – Standalone representation with clear semantics
  – Reasoning patterns (conditioning, decision making)
  – Learning methods
(Probabilistic) Graphical Models

• Why graphical?
• Intersection of ideas from probability theory and computer science
  – To represent large number of variables

Predisposing factors
Symptoms
Test results

<table>
<thead>
<tr>
<th></th>
<th>Millions of pixels</th>
<th>Colours / features</th>
</tr>
</thead>
</table>

Random variables \( Y_1, Y_2, ..., Y_n \)

Goal: capture uncertainty through joint distribution \( P(Y_1, ..., Y_n) \)

Slide inspired by PGM course, Daphne Koller
(Probabilistic) Graphical Models
(Probabilistic) Graphical Model

• Examples

Bayesian network (directed graph)

Markov network (undirected graph)

Figure courtesy: D. Koller
(Probabilistic) Graphical Model

• Examples

Diagnosis network: Pradhan et al., UAI’94

Segmentation network (Courtesy D. Koller)
(Probabilistic) Graphical Model

• Intuitive & compact data structure

• Efficient reasoning through general-purpose algorithms

• Sparse parameterization
  – Through expert knowledge, or
  – Learning from data
(Probabilistic) Graphical Model

• Many many applications
  – Medical diagnosis
  – Fault diagnosis
  – Natural language processing
  – Traffic analysis
  – Social network models
  – Message decoding
  – Computer vision: segmentation, 3D, pose estimation
  – Speech recognition
  – Robot localization & mapping

Slide courtesy: PGM course, Daphne Koller
Image segmentation

Sturgess et al., 2009
Multi-sensor integration: Traffic

- Learn from historical data to make predictions

Slide courtesy: Eric Horvitz, MSR
Going global: Local ambiguity

- Text recognition

Smyth et al., 1994

Slide courtesy: Dhruv Batra
Going global: Local ambiguity

- Textual information extraction

e.g., Mrs. Green spoke today in New York. Green chairs the financial committee.

Slide courtesy: PGM course, Daphne Koller
Overview

• Representation
  – How do we store $P(Y_1, \ldots Y_n)$
  – Directed and undirected (model implications/assumptions)

• Inference
  – Answer questions with the model
  – Exact and approximate (marginal/most probable estimate)

• Learning
  – What model is right for data
  – Parameters and structure

Slide inspired by D. Batra, D. Koller ‘s courses
First, a recap of basics
Graphs

• Concepts
  – Definition of G
  – Vertices/Nodes
  – Edges
  – Directed vs Undirected
  – Neighbours vs Parent/Child
  – Degree vs In/Out degree
  – Walk vs Path vs Cycle
Graphs

A - B - C - D - E

A - B - D - C - E

Graphs
Special graphs

- Trees: undirected graph, no cycles
- Spanning tree: Same set of vertices, but subset of edges, connected and no cycles

Slide courtesy: D. Batra
Directed acyclic graphs (DAGs)
Joint distribution

• 3 variables
  – Intelligence (I)
  – Difficulty (D)
  – Grade (G)

• Independent parameters?

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>D</th>
<th>G</th>
<th>Prob.</th>
</tr>
</thead>
<tbody>
<tr>
<td>i⁰</td>
<td>d⁰</td>
<td>g¹</td>
<td>0.126</td>
<td></td>
</tr>
<tr>
<td>i⁰</td>
<td>d⁰</td>
<td>g²</td>
<td>0.168</td>
<td></td>
</tr>
<tr>
<td>i⁰</td>
<td>d⁰</td>
<td>g³</td>
<td>0.126</td>
<td></td>
</tr>
<tr>
<td>i⁰</td>
<td>d¹</td>
<td>g¹</td>
<td>0.009</td>
<td></td>
</tr>
<tr>
<td>i⁰</td>
<td>d¹</td>
<td>g²</td>
<td>0.045</td>
<td></td>
</tr>
<tr>
<td>i⁰</td>
<td>d¹</td>
<td>g³</td>
<td>0.126</td>
<td></td>
</tr>
<tr>
<td>i¹</td>
<td>d⁰</td>
<td>g¹</td>
<td>0.252</td>
<td></td>
</tr>
<tr>
<td>i¹</td>
<td>d⁰</td>
<td>g²</td>
<td>0.0224</td>
<td></td>
</tr>
<tr>
<td>i¹</td>
<td>d⁰</td>
<td>g³</td>
<td>0.0056</td>
<td></td>
</tr>
<tr>
<td>i¹</td>
<td>d¹</td>
<td>g¹</td>
<td>0.06</td>
<td></td>
</tr>
<tr>
<td>i¹</td>
<td>d¹</td>
<td>g²</td>
<td>0.036</td>
<td></td>
</tr>
<tr>
<td>i¹</td>
<td>d¹</td>
<td>g³</td>
<td>0.024</td>
<td></td>
</tr>
</tbody>
</table>

Example courtesy: PGM course, Daphne Koller
Conditioning

• Condition on $g^1$

<table>
<thead>
<tr>
<th>I</th>
<th>D</th>
<th>G</th>
<th>Prob.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i^0$</td>
<td>$d^0$</td>
<td>$g^1$</td>
<td>0.126</td>
</tr>
<tr>
<td>$i^0$</td>
<td>$d^0$</td>
<td>$g^2$</td>
<td>0.168</td>
</tr>
<tr>
<td>$i^0$</td>
<td>$d^0$</td>
<td>$g^3$</td>
<td>0.126</td>
</tr>
<tr>
<td>$i^0$</td>
<td>$d^1$</td>
<td>$g^1$</td>
<td>0.009</td>
</tr>
<tr>
<td>$i^0$</td>
<td>$d^1$</td>
<td>$g^2$</td>
<td>0.045</td>
</tr>
<tr>
<td>$i^0$</td>
<td>$d^1$</td>
<td>$g^3$</td>
<td>0.126</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^0$</td>
<td>$g^1$</td>
<td>0.252</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^0$</td>
<td>$g^2$</td>
<td>0.0224</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^0$</td>
<td>$g^3$</td>
<td>0.0056</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^1$</td>
<td>$g^1$</td>
<td>0.06</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^1$</td>
<td>$g^2$</td>
<td>0.036</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^1$</td>
<td>$g^3$</td>
<td>0.024</td>
</tr>
</tbody>
</table>

Example courtesy: PGM course, Daphne Koller
Conditioning

- $P(Y = y \mid X = x)$
- Informally,
  - What do you believe about $Y=y$ when I tell you $X=x$?

- $P($France wins a football tournament in 2023$)$?
- What if I tell you:
  - France almost won the world cup 2022
  - Hasn’t had catastrophic results since 😊
Conditioning: Reduction

- Condition on $g^1$

<table>
<thead>
<tr>
<th>I</th>
<th>D</th>
<th>G</th>
<th>Prob.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i^0$</td>
<td>$d^0$</td>
<td>$g^1$</td>
<td>0.126</td>
</tr>
<tr>
<td>$i^0$</td>
<td>$d^1$</td>
<td>$g^1$</td>
<td>0.009</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^0$</td>
<td>$g^1$</td>
<td>0.252</td>
</tr>
<tr>
<td>$i^1$</td>
<td>$d^1$</td>
<td>$g^1$</td>
<td>0.06</td>
</tr>
</tbody>
</table>

Example courtesy: PGM course, Daphne Koller
Conditioning: Renormalization

Unnormalized measure

Example courtesy: PGM course, Daphne Koller
Conditional probability distribution

- Example $P(G | I, D)$

<table>
<thead>
<tr>
<th></th>
<th>$g^1$</th>
<th>$g^2$</th>
<th>$g^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i^0, d^0$</td>
<td>0.3</td>
<td>0.4</td>
<td>0.3</td>
</tr>
<tr>
<td>$i^0, d^1$</td>
<td>0.05</td>
<td>0.25</td>
<td>0.7</td>
</tr>
<tr>
<td>$i^1, d^0$</td>
<td>0.9</td>
<td>0.08</td>
<td>0.02</td>
</tr>
<tr>
<td>$i^1, d^1$</td>
<td>0.5</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Example courtesy: PGM course, Daphne Koller
Conditional probability distribution

\[ p(x, y \mid Z = z) = \frac{p(x, y, z)}{p(z)} \]
Marginalization

**Example courtesy:** PGM course, Daphne Koller
Marginalization

• Events
  – \( P(A) = P(A \text{ and } B) + P(A \text{ and not } B) \)

• Random variables
  – \( P(X = x) = \sum_{y} P(X = x, Y = y) \)
Marginalization

\[ p(x, y) = \sum_{z \in \mathcal{Z}} p(x, y, z) \]

\[ p(x) = \sum_{y \in \mathcal{Y}} p(x, y) \]

Slide courtesy: Erik Sudderth
Factors

- A factor $\Phi(Y_1, \ldots, Y_k)$

  $\Phi: \text{Val}(Y_1, \ldots, Y_k) \rightarrow \mathbb{R}$

- Scope = $\{Y_1, \ldots, Y_k\}$
General factors

- Not necessarily for probabilities

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>φ</th>
</tr>
</thead>
<tbody>
<tr>
<td>a^0</td>
<td>b^0</td>
<td>30</td>
</tr>
<tr>
<td>a^0</td>
<td>b^1</td>
<td>5</td>
</tr>
<tr>
<td>a^1</td>
<td>b^0</td>
<td>1</td>
</tr>
<tr>
<td>a^1</td>
<td>b^1</td>
<td>10</td>
</tr>
</tbody>
</table>
Factor product

Example courtesy: PGM course, Daphne Koller
## Factor marginalization

<table>
<thead>
<tr>
<th></th>
<th>a(^1)</th>
<th>a(^2)</th>
<th>a(^3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>b(^1)</td>
<td>0.25</td>
<td>0.35</td>
<td>0.15</td>
</tr>
<tr>
<td>b(^2)</td>
<td>0.08</td>
<td>0.07</td>
<td>0.21</td>
</tr>
<tr>
<td>c(^1)</td>
<td>0</td>
<td>0</td>
<td>0.09</td>
</tr>
<tr>
<td>c(^2)</td>
<td>0.33</td>
<td>0.51</td>
<td>0.39</td>
</tr>
</tbody>
</table>

Example courtesy: PGM course, Daphne Koller
## Factor reduction

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$a^1$</td>
<td>$b^1$</td>
<td>$c^1$</td>
<td>0.25</td>
</tr>
<tr>
<td>$a^1$</td>
<td>$b^1$</td>
<td>$c^2$</td>
<td>0.35</td>
</tr>
<tr>
<td>$a^1$</td>
<td>$b^2$</td>
<td>$c^1$</td>
<td>0.08</td>
</tr>
<tr>
<td>$a^1$</td>
<td>$b^2$</td>
<td>$c^2$</td>
<td>0.16</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$b^1$</td>
<td>$c^1$</td>
<td>0.05</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$b^1$</td>
<td>$c^2$</td>
<td>0.07</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$b^2$</td>
<td>$c^1$</td>
<td>0.00</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$b^2$</td>
<td>$c^2$</td>
<td>0.00</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$b^1$</td>
<td>$c^1$</td>
<td>0.15</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$b^1$</td>
<td>$c^2$</td>
<td>0.21</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$b^2$</td>
<td>$c^1$</td>
<td>0.09</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$b^2$</td>
<td>$c^2$</td>
<td>0.18</td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$a^1$</td>
<td>$b^1$</td>
<td>$c^1$</td>
<td>0.25</td>
</tr>
<tr>
<td>$a^1$</td>
<td>$b^2$</td>
<td>$c^1$</td>
<td>0.08</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$b^1$</td>
<td>$c^1$</td>
<td>0.05</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$b^2$</td>
<td>$c^1$</td>
<td>0.00</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$b^1$</td>
<td>$c^1$</td>
<td>0.15</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$b^2$</td>
<td>$c^1$</td>
<td>0.09</td>
</tr>
</tbody>
</table>
Why factors?

• Building blocks for defining distributions in high-dimensional spaces

• Set of basic operations for manipulating these distributions
Bayesian Networks

• DAGs
  – nodes represent variables in the Bayesian sense
  – edges represent conditional dependencies

• Example
  – Suppose that we know the following:
    • The flu causes sinus inflammation
    • Allergies cause sinus inflammation
    • Sinus inflammation causes a runny nose
    • Sinus inflammation causes headaches
  – How are these connected?
Bayesian Networks

- Example
Bayesian Networks

• A general Bayes net
  – Set of random variables
  – DAG: encodes independence assumptions
  – Conditional probability trees
  – Joint distribution

\[
P(Y_1, \ldots, Y_n) = \prod_{i=1}^{n} P(Y_i \mid Pa_{Y_i})
\]
Bayesian Networks

• A general Bayes net
  – How many parameters?
    • Discrete variables $Y_1, \ldots, Y_n$
    • Graph: Defines parents of $Y_i$, i.e., $(Pa_{Y_i})$
    • CPTs: $P(Y_i \mid Pa_{Y_i})$

Slide courtesy: Dhruv Batra
Markov nets

• Set of random variables

• Undirected graph
  – Encodes independence assumptions

• Factors

Comparison to Bayesian Nets?
Pairwise MRFs

• Composed of pairwise factors
  – A function of two variables
  – Can also have unary terms

• Example
Markov Nets: Computing probabilities

- Can only compute ratio of probabilities directly

  - Need to normalize with a partition function
    - Hard! (sum over all possible assignments)

- In Bayesian Nets, can do by multiplying CPTs

Slide courtesy: Dhruv Batra
Markov nets $\longleftrightarrow$ Factorization

- Given an undirected graph $H$ over variables $Y=\{Y_1,\ldots,Y_n\}$
- A distribution $P$ factorizes over $H$ if there exist
  - Subsets of variables $S^i \subseteq Y$ s.t. $S^i$ are fully-connected in $H$
  - Non-negative potentials (factors) $\Phi_1(S^1),\ldots,$ $\Phi_m(S^m)$: clique potentials
  - Such that
    \[
    P(Y_1,\ldots,Y_n) = \frac{1}{Z} \prod_{i=1}^{m} \Phi_i(S^i)
    \]

Slide courtesy: Dhruv Batra
Conditional Markov Random Fields

• Also known as: Markov networks, undirected graphical models, MRFs
• Note: Not making a distinction between CRFs and MRFs
• $X \in \mathcal{X}$: observed random variables
• $Y = (Y_1, \ldots, Y_n) \in \mathcal{Y}$: output random variables
• $Y_c$ are subset of variables for clique $c \subseteq \{1, \ldots, n\}$
• Define a factored probability distribution

$$P(Y \mid X) = \frac{1}{Z(X)} \prod_c \psi_c(Y_c; X)$$

Partition function $= \sum_{Y \in \mathcal{Y}} \prod_c \psi_c(Y_c; X)$  

Exponential number of configurations!
MRFs / CRFs

• Several applications, e.g., computer vision

Interactive figure-ground segmentation [Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006]
Surface context [Hoiem et al., 2005]
Semantic labeling [He et al., 2004; Shotton et al., 2006; Gould et al., 2009]
Stereo matching [Kolmogorov and Zabih, 2001; Scharstein and Szeliski, 2002]
Image denoising [Felzenszwalb and Huttenlocher 2004]
MRFs / CRFs

• Several applications, e.g., computer vision

High-level vision problems

Object detection [Felzenszwalb et al., 2008]
Pose estimation [Akhter and Black, 2015; Ramakrishna et al., 2012]
Scene understanding [Fouhey et al., 2014; Ladicky et al., 2010; Xiao et al., 2013; Yao et al., 2012]
MRFs / CRFs

• Several applications, e.g., medical imaging
MRFs / CRFs

• Inherent in all these problems are graphical models
Maximum a posteriori (MAP) inference

\[ y^* = \arg \max_{y \in \mathcal{Y}} P(y \mid x) \]

\[ = \arg \max_{y \in \mathcal{Y}} \frac{1}{Z(x)} \prod_c \psi_c(Y_c; X) \]

\[ = \arg \max_{y \in \mathcal{Y}} \log \left( \frac{1}{Z(x)} \prod_c \psi_c(Y_c; X) \right) \]

\[ = \arg \max_{y \in \mathcal{Y}} \sum_c \log \psi_c(Y_c; X) - \log Z(X) \]

\[ = \arg \max_{y \in \mathcal{Y}} \sum_c \log \psi_c(Y_c; X) - E(Y; X) \]
Maximum a posteriori (MAP) inference

$$
\mathbf{y}^* = \arg\max_{\mathbf{y} \in \mathcal{Y}} P(\mathbf{y} \mid \mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \sum_c \log \Psi_c(\mathbf{Y}_c; \mathbf{X})
$$

$$
= \arg\min_{\mathbf{y} \in \mathcal{Y}} E(\mathbf{y}; \mathbf{x})
$$

MAP inference $\Leftrightarrow$ Energy minimization

The energy function is $E(\mathbf{Y}; \mathbf{X}) = \sum_c \psi_c(\mathbf{Y}_c; \mathbf{X})$

where $\psi_c(\cdot) = -\log \Psi_c(\cdot)$
Clique potentials

- Defines a mapping from an assignment of random variables to a real number
  \[ \psi_c : \mathcal{V}_c \times \mathcal{X} \rightarrow \mathbb{R} \]

- Encodes a preference for assignments to the random variables (lower is better)

- Parameterized as
  \[ \psi_c(y_c; x) = w_c^T \phi_c(y_c; x) \]
Clique potentials

- Arity

\[ E(y; x) = \sum_c \psi_c(y_c; x) \]

\[ = \sum_{i \in V} \psi_i^U(y_i; x) + \sum_{ij \in E} \psi_{ij}^P(y_i, y_j; x) + \sum_{c \in C} \psi_c^H(y_c; x). \]
Clique potentials

• Arity

4-connected, $\mathcal{N}_4$

8-connected, $\mathcal{N}_8$
Reason 1: Texture modelling

Training images

Test image

Test image (60% Noise)

Result MRF 4-connected (neighbours)

Result MRF 4-connected

Result MRF 9-connected (7 attractive; 2 repulsive)
Reason 2: Discretization artefacts

- Higher connectivity can model true Euclidean length

[Boykov et al. ’03; ’05]
Graphical representation

- Example

\[ E(y) = \psi(y_1, y_2) + \psi(y_2, y_3) + \psi(y_3, y_4) + \psi(y_4, y_1) \]

factor graph
Graphical representation

- Example

\[ E(y) = \sum_{i,j} \psi(y_i, y_j) \]
Graphical representation

• Example

\[ E(y) = \psi(y_1, y_2, y_3, y_4) \]
A Computer Vision Application

Binary Image Segmentation

How?

Cost function Models *our* knowledge about natural images

Optimize cost function to obtain the segmentation
A Computer Vision Application

Binary Image 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 $L = \{\text{obj,bkg}\}$
A Computer Vision Application

Binary Image Segmentation

Object - white, Background - green/grey

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

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

Per Vertex Cost

Graph $G = (V,E)$
Graph \( G = (V,E) \)

Cost of a labelling \( f : V \rightarrow L \)

Object - white, Background - green/grey

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

Per Vertex Cost

UNARY COST
A Computer Vision Application

Binary Image Segmentation

Graph $G = (V,E)$

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

Cost of same label low

Cost of different labels high

Object - white, Background - green/grey

Per Edge Cost
A Computer Vision Application

Binary Image Segmentation

Graph $G = (V,E)$

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

Object - white, Background - green/grey

Cost of same label high

Per Edge Cost

Cost of different labels low

PAIRWISE COST
A Computer Vision Application

Binary Image Segmentation

Object - white, Background - green/grey

Graph $G = (V,E)$

Problem: Find the labelling with minimum cost $f^*$
A Computer Vision Application

Binary Image Segmentation

Graph \( G = (V,E) \)

Problem: Find the labelling with minimum cost \( f^* \)
Another Computer Vision Application

Stereo Correspondence

Disparity Map

How?

Minimizing a cost function
Another Computer Vision Application

Stereo Correspondence

Graph $G = (V,E)$

Vertex corresponds to a pixel

Edges define grid graph

$L = \{\text{disparities}\}$
Another Computer Vision Application

Stereo Correspondence

Cost of labelling $f$:

Unary cost + Pairwise Cost

Find minimum cost $f^*$
The General Problem

Graph $G = (V, E)$

Discrete label set $L = \{1,2,\ldots,h\}$

Assign a label to each vertex $f: V \rightarrow L$

Cost of a labelling $Q(f)$

Unary Cost Pairwise Cost

Find $f^* = \text{arg min } Q(f)$
Overview

• Basics: problem formulation
  – Energy Function
  – MAP Estimation
  – Computing min-marginals
  – Reparameterization

• Solutions
  – Belief Propagation and related methods
  – Graph cuts
Remainder of today’s lecture

- Belief propagation
- TRW
- Graph cuts
Belief Propagation
A Computer Vision Application

Binary Image Segmentation

How?

Cost function Models our knowledge about natural images

Optimize cost function to obtain the segmentation
Another Computer Vision Application

Stereo Correspondence

Disparity Map

How?

Minimizing a cost function
The General Problem

Graph $G = (V, E)$

Discrete label set $L = \{1, 2, \ldots, h\}$

Assign a label to each vertex $f: V \rightarrow L$

Cost of a labelling $Q(f)$

Unary Cost  Pairwise Cost

Find $f^* = \arg \min Q(f)$
Overview

• Basics: problem formulation
  – Energy Function
  – MAP Estimation
  – Computing min-marginals
  – Reparameterization

• Solutions
  – Belief Propagation and related methods
  – Graph cuts
Energy Function

Random Variables $V = \{V_a, V_b, \ldots\}$

Labels $L = \{l_0, l_1, \ldots\}$  Data $D$

Labelling $f: \{a, b, \ldots\} \rightarrow \{0, 1, \ldots\}$
Energy Function

\[ Q(f) = \sum_a \theta_{a;f(a)} \]

Unary Potential

Easy to minimize

Neighbourhood
Energy Function

$E : (a,b) \in E$ iff $V_a$ and $V_b$ are neighbours

$E = \{ (a,b) , (b,c) , (c,d) \}$
Energy Function

$$Q(f) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$
Energy Function

\[ Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)} \]
Overview

• Basics: problem formulation
  – Energy Function
  – MAP Estimation
  – Computing min-marginals
  – Reparameterization

• Solutions
  – Belief Propagation and related methods
  – Graph cuts
\[ Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)} \]

\[ 2 + 1 + 2 + 1 + 3 + 1 + 3 = 13 \]
\[
Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}
\]

\[
5 + 1 + 4 + 0 + 6 + 4 + 7 = 27
\]
MAP Estimation

\[ Q(f; \theta) = \sum_a \theta_a f(a) + \sum_{(a,b)} \theta_{ab} f(a) f(b) \]

\[ f^* = \arg \min Q(f; \theta) \]

Equivalent to maximizing the associated probability
### MAP Estimation

16 possible labellings

<table>
<thead>
<tr>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
<th>f(d)</th>
<th>Q(f; (\theta))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>22</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20</td>
</tr>
</tbody>
</table>

### f* = \{1, 0, 0, 1\}

<table>
<thead>
<tr>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
<th>f(d)</th>
<th>Q(f; (\theta))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>23</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>16</td>
</tr>
</tbody>
</table>

q* = 13
Computational Complexity

Segmentation

$2^{|V|}$

$|V| = \text{number of pixels} \approx 153600$

Can we do better than brute-force?

MAP Estimation is NP-hard!!
MAP Inference / Energy Minimization

- Computing the assignment minimizing the energy in NP-hard in general
  \[
  \arg\min_{y \in Y} E(y; x) = \arg\max_{y \in Y} P(y | x)
  \]

- Exact inference is possible in some cases, e.g.,
  - Low treewidth graphs → message-passing
  - Submodular potentials → graph cuts

- Efficient approximate inference algorithms exist
  - Message passing on general graphs
  - Move-making algorithms
  - Relaxation algorithms
Overview

• Basics: problem formulation
  – Energy Function
  – MAP Estimation
  – Computing min-marginals
  – Reparameterization

• Solutions
  – Belief Propagation and related methods
  – Graph cuts
Min-Marginals

\[ f^* = \text{arg min } Q(f; \theta) \text{ such that } f(a) = i \]

Min-marginal \( q_{a;i} \)

Not a marginal (no summation)
# Min-Marginals

16 possible labellings

\[ q_{a;0} = 15 \]

<table>
<thead>
<tr>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
<th>f(d)</th>
<th>Q(f; \theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>22</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
<th>f(d)</th>
<th>Q(f; \theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>23</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>16</td>
</tr>
</tbody>
</table>
Min-Marginals

16 possible labellings

\[ q_{a;1} = 13 \]

<table>
<thead>
<tr>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
<th>f(d)</th>
<th>Q(f; \theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>22</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
<th>f(d)</th>
<th>Q(f; \theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>23</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>16</td>
</tr>
</tbody>
</table>
Min-Marginals and MAP

• Minimum min-marginal of any variable = energy of MAP labelling

\[ \min_i q_{a;i} \]

\[ \min_i \left( \min_f Q(f; \theta) \text{ such that } f(a) = i \right) \]

\[ V_a \text{ has to take one label} \]

\[ \min_f Q(f; \theta) \]
Summary

Energy Function

$$Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

MAP Estimation

$$f^* = \text{arg min } Q(f; \theta)$$

Min-marginals

$$q_{a;i} = \text{min } Q(f; \theta) \quad \text{s.t. } f(a) = i$$
Overview

• Basics: problem formulation
  – Energy Function
  – MAP Estimation
  – Computing min-marginals
  – Reparameterization

• Solutions
  – Belief Propagation and related methods
  – Graph cuts
Reparameterization

Add a constant to all \( \theta_{a;i} \)
Subtract that constant from all \( \theta_{b;k} \)

\[
Q(f; \theta') = Q(f; \theta)
\]
Reparameterization

\[ f(a) \quad f(b) \quad Q(f; \theta) \]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>10 - 3 + 3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>6 - 3 + 3</td>
</tr>
</tbody>
</table>

Add a constant to one \( \theta_{b;k} \)

Subtract that constant from \( \theta_{ab;ik} \) for all ‘i’

\[ Q(f; \theta') = Q(f; \theta) \]
Reparameterization

\( \theta' \) is a reparameterization of \( \theta \), iff

\[
Q(f; \theta') = Q(f; \theta), \text{ for all } f
\]

\[
\theta' \equiv \theta
\]

Equivalently

\[
\theta'_{a; i} = \theta_{a; i} + M_{ba; i}
\]

\[
\theta'_{b; k} = \theta_{b; k} + M_{ab; k}
\]

\[
\theta'_{ab; ik} = \theta_{ab; ik} - M_{ab; k} - M_{ba; i}
\]

Kolmogorov, PAMI, 2006
Recap

MAP Estimation

$$f^* = \operatorname{arg\,min} \, Q(f; \theta)$$

$$Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

Min-marginals

$$q_{a;i} = \min Q(f; \theta) \; \text{s.t.} \; f(a) = i$$

Reparameterization

$$Q(f; \theta') = Q(f; \theta), \text{ for all } f \quad \theta' \equiv \theta$$
Overview

• Basics: problem formulation
  – Energy Function
  – MAP Estimation
  – Computing min-marginals
  – Reparameterization

• Solutions
  – Belief Propagation and related methods
  – Graph cuts
Belief Propagation

• Remember, some MAP problems are easy

• Belief Propagation gives exact MAP for chains

• Exact MAP for trees

• Clever Reparameterization
Two Variables

Add a constant to one $\theta_{b;k}$

Subtract that constant from $\theta_{ab;ik}$ for all ‘$i$’

Choose the right constant

$$\theta'_{b;k} = q_{b;k}$$
Choose the right constant $\theta'_{b;k} = q_{b;k}$.

\[
M_{ab;0} = \min \begin{aligned}
\theta_{a;0} + \theta_{ab;00} &= 5 + 0 \\
\theta_{a;1} + \theta_{ab;10} &= 2 + 1
\end{aligned}
\]
Two Variables

\[ f(a) = 1 \]

\[ \theta'_{b;0} = q_{b;0} \]

Potentials along the red path add up to 0

Choose the *right* constant

\[ \theta'_{b;k} = q_{b;k} \]
Choose the \textit{right} constant \( \theta'_{b;k} = q_{b;k} \)
Choose the right constant \( \theta'_{b;k} = q_{b;k} \)
Choose the right constant $\theta'_{b;k} = q_{b;k}$.
Two Variables

\[ f(a) = 1 \]

\[ V_a \]
\[ V_b \]

\[ 2 \]
\[ -2 \]
\[ 5 \]
\[ -3 \]
\[ 5 \]

We get all the min-marginals of \( V_b \)

Choose the \textit{right} constant

\[ \theta'_{b;k} = q_{b;k} \]
Recap

We only need to know two sets of equations

General form of Reparameterization

\[
\begin{align*}
\theta'_{a;i} &= \theta_{a;i} + M_{ba;i} \\
\theta'_{b;k} &= \theta_{b;k} + M_{ab;k} \\
\theta'_{ab;ik} &= \theta_{ab;ik} - M_{ab;k} - M_{ba;i}
\end{align*}
\]

Reparameterization of (a,b) in Belief Propagation

\[
\begin{align*}
M_{ab;k} &= \min_i \{ \theta_{a;i} + \theta_{ab;ik} \} \\
M_{ba;i} &= 0
\end{align*}
\]
Three Variables

Reparameterize the edge \((a,b)\) as before
Reparameterize the edge \((a,b)\) as before.
Reparameterize the edge \((a,b)\) as before

Potentials along the red path add up to 0
Reparameterize the edge \((b,c)\) as before.

Potentials along the red path add up to 0.
Reparameterize the edge (b,c) as before

Potentials along the red path add up to 0
Three Variables

\[ f(a) = 1 \quad f(b) = 1 \]

Generalizes to any length chain

\[ f^*(c) = 0 \quad f^*(b) = 0 \quad f^*(a) = 1 \]
Three Variables

\[ f(a) = 1 \quad f(b) = 1 \]

\[ f^*(c) = 0 \quad f^*(b) = 0 \quad f^*(a) = 1 \]

Only Dynamic Programming
Why Dynamic Programming?

3 variables ≡ 2 variables + book-keeping

n variables ≡ (n-1) variables + book-keeping

Start from left, go to right

Reparameterize current edge (a,b)

\[ M_{ab;k} = \min_i \{ \theta_{a;i} + \theta_{ab;ik} \} \]

\[ \theta'_{b;k} = \theta_{b;k} + M_{ab;k} \]

\[ \theta'_{ab;ik} = \theta_{ab;ik} - M_{ab;k} \]

Repeat
**Why Dynamic Programming?**

**Messages**  **Message Passing**

**Why stop at dynamic programming?**

Start from left, go to right

Reparameterize current edge \((a,b)\)

\[
M_{ab;k} = \min_i \{ \theta_{a;i} + \theta_{ab;ik} \}
\]

\[
\theta'_{b;k} = \theta_{b;k} + M_{ab;k} \quad \theta'_{ab;ik} = \theta_{ab;ik} - M_{ab;k}
\]

Repeat
Reparameterize the edge (c,b) as before
Reparameterize the edge \((c,b)\) as before

\[
\theta'_{b,i} = q_{b;i}
\]
Reparameterize the edge \((b,a)\) as before

\[
\theta'_{a;i} = q_{a;i}
\]
Three Variables

Forward Pass ➔

Backward Pass ←

All min-marginals are computed
Chains

Reparameterize the edge (1,2)
Chains

Reparameterize the edge (2,3)
Chains

Reparameterize the edge (3, 4)
Reparameterize the edge \((n-1,n)\)

Min-marginals \(e_n(i)\) for all labels
Belief Propagation on Chains

Start from left, go to right

Reparameterize current edge \((a,b)\)

\[
M_{ab;k} = \min_i \{ \theta_a;i + \theta_{ab;ik} \}
\]

\[
\theta'_{b;k} = \theta_{b;k} + M_{ab;k} \quad \theta'_{ab;ik} = \theta_{ab;ik} - M_{ab;k}
\]

Repeat till the end of the chain

Start from right, go to left

Repeat till the end of the chain
Belief Propagation on Chains

- Generalizes to chains of any length
- A way of computing reparam constants

- Forward Pass - Start to End
  - MAP estimate
  - Min-marginals of final variable

- Backward Pass - End to start
  - All other min-marginals
Computational Complexity

Number of reparameterization constants = \( (n-1)h \)

Complexity for each constant = \( O(h) \)

Total complexity = \( O(nh^2) \)

Better than brute-force \( O(h^n) \)
Reparameterize the edge (4,2)
Reparameterize the edge \((5,2)\)
Reparameterize the edge (6,3)
Reparameterize the edge (7,3)
Reparameterize the edge (2,1)
Reparameterize the edge (3,1)

Min-marginals $e_1(i)$ for all labels
Start from leaves and move towards root

Pick the minimum of min-marginals

Backtrack to find the best labeling $x$
Outline

• Preliminaries
  – s-t Flow
  – s-t Cut
  – Flows vs. Cuts

• Maximum Flow
  • Algorithms
  • Energy minimization with max flow/min cut
s-t Flow

Function flow: $A \rightarrow R$

Flow of arc $\leq$ arc capacity

Flow is non-negative

For all vertex except $s, t$

Incoming flow = Outgoing flow
s-t Flow

Function flow: $A \rightarrow R$

flow(a) $\leq c(a)$

Flow is non-negative

For all vertex except s,t

Incoming flow = Outgoing flow
s-t Flow

Function flow: $A \rightarrow R$

$\text{flow}(a) \leq c(a)$

$\text{flow}(a) \geq 0$

For all vertex except $s,t$

Incoming flow = Outgoing flow
s-t Flow

Function flow: A \rightarrow R

flow(a) \leq c(a)

flow(a) \geq 0

For all v \in V \setminus \{s,t\}

Incoming flow

= Outgoing flow
s-t Flow

Function flow: $A \rightarrow R$

flow(a) ≤ c(a)

flow(a) ≥ 0

For all $v \in V \setminus \{s,t\}$

$$\sum_{(u,v) \in A} \text{flow}((u,v))$$

= Outgoing flow
s-t Flow

Function flow: $A \rightarrow R$

$\text{flow}(a) \leq c(a)$

$\text{flow}(a) \geq 0$

For all $v \in V \setminus \{s,t\}$

$\sum_{(u,v) \in A} \text{flow}((u,v)) = \sum_{(v,u) \in A} \text{flow}((v,u))$
s-t Flow

Function flow: $A \Rightarrow R$

flow(a) ≤ c(a)

flow(a) ≥ 0

For all $v \in V \setminus \{s,t\}$

$E_{flow}(v) = 0$
s-t Flow

Function flow: $A \rightarrow R$

flow($a$) $\leq$ c($a$)

flow($a$) $\geq$ 0

For all $v \in V \setminus \{s,t\}$

$E_{\text{flow}}(v) = 0$
**s-t Flow**

Function flow: $A \rightarrow R$

$\text{flow}(a) \leq c(a)$

$\text{flow}(a) \geq 0$

For all $v \in V \setminus \{s,t\}$

$E_{\text{flow}}(v) = 0$
s-t Flow

Function flow: $A \rightarrow R$

$\text{flow}(a) \leq c(a)$

$\text{flow}(a) \geq 0$

For all $v \in V \setminus \{s,t\}$

$E_{\text{flow}}(v) = 0$

✓
Value of s-t Flow

Outgoing flow of s
- Incoming flow of s

Graph with nodes labeled s, v₁, v₂, v₃, v₄, and t.

Values:
- s to v₁: 1
- s to v₂: 8
- v₁ to v₂: 6
- v₂ to v₃: 2
- v₂ to v₄: 5
- v₃ to t: 7
- v₄ to t: 3
Value of s-t Flow

\[
\begin{align*}
\text{Value} &= 1 \\
-\sum_{(u,s) \in A} \text{flow}((u,s)) - \sum_{(s,v) \in A} \text{flow}((s,v)) - E_{\text{flow}}(s) + E_{\text{flow}}(t)
\end{align*}
\]
Outline

• Preliminaries
  – Functions and Excess Functions
  – s-t Flow
  – s-t Cut
  – Flows vs. Cuts

• Maximum Flow
  • Algorithms
  • Energy minimization with max flow/min cut
Let $U$ be a subset of $V$

$D = (V, A)$

$C$ is a set of arcs such that

- $(u, v) \in A$
- $u \in U$
- $v \in V \setminus U$

$C$ is a cut in the digraph $D$
What is $C$?

$D = (V, A)$

$\{ (v_1, v_2), (v_1, v_4) \}$ ?

$\{ (v_1, v_4), (v_3, v_2) \}$ ?

$\{ (v_1, v_4) \}$ ?
What is C?

\[ D = (V, A) \]

\{ (v_1, v_2), (v_1, v_4), (v_3, v_2) \} ?

\{ (v_4, v_3) \} ?

\{ (v_1, v_4), (v_3, v_2) \} ?
What is $C$?

$D = (V, A)$

$\{(v_1, v_2), (v_1, v_4), (v_3, v_2)\}$ ?

$\{(v_3, v_2)\}$ ?

$\{(v_1, v_4), (v_3, v_2)\}$ ?
Cut

\[ D = (V, A) \]

\[ C = \text{out-arcs}(U) \]
Capacity of Cut

Sum of capacity of all arcs in C
Capacity of Cut

\[ \sum_{a \in C} c(a) \]
Capacity of Cut

\[ U \]

\[ V \backslash U \]

\[ v_1 \quad 3 \quad v_2 \]
\[ v_3 \quad 3 \quad v_4 \]
\[ 10 \quad 3 \quad 2 \quad 2 \quad 5 \quad 3 \]
Capacity of Cut
A source vertex “s”

A sink vertex “t”

C is a cut such that
- \( s \in U \)
- \( t \in V \setminus U \)

C is an s-t cut
Capacity of s-t Cut

\[ \sum_{a \in C} c(a) \]
Capacity of s-t Cut

The diagram shows a network with vertices connected by edges. The capacity of the s-t cut is 5 units.
Capacity of s-t Cut

The capacity of the s-t cut in the graph is 17.
Outline

• Preliminaries
  – s-t Flow
  – s-t Cut
  – Flows vs. Cuts

• Maximum Flow
  • Algorithms
  • Energy minimization with max flow/min cut
Outline

• Preliminaries

• **Maximum Flow**
  – Residual Graph
  – Max-Flow Min-Cut Theorem

• Algorithms

• Energy minimization with max flow/min cut
Maximum Flow Problem

Find the flow with the maximum value !!

\[ \sum_{(s,v) \in A} \text{flow}((s,v)) - \sum_{(u,s) \in A} \text{flow}((u,s)) \]

First suggestion to solve this problem !!
Passing Flow through s-t Paths

Find an s-t path where \( \text{flow}(a) < c(a) \) for all arcs

Pass maximum allowable flow through the arcs
Passing Flow through s-t Paths

Find an s-t path where \( \text{flow}(a) < c(a) \) for all arcs
Passing Flow through s-t Paths

Find an s-t path where flow(a) < c(a) for all arcs

Pass maximum allowable flow through the arcs
Passing Flow through s-t Paths

Find an s-t path where \( \text{flow}(a) < c(a) \) for all arcs

No more paths. Stop.

Will this give us maximum flow? NO !!!
Passing Flow through s-t Paths

Find an s-t path where flow(a) < c(a) for all arcs

Pass maximum allowable flow through the arcs
Passing Flow through s-t Paths

Find an s-t path where flow(a) < c(a) for all arcs.

No more paths. Stop.

Another method?

Incorrect Answer!!
Outline

• Preliminaries

• Maximum Flow
  – Residual Graph
  – Max-Flow Min-Cut Theorem

• Algorithms

• Energy minimization with max flow/min cut
Residual Graph

Arcs where $\text{flow}(a) < c(a)$
Residual Graph

Including arcs to $s$ and from $t$ is not necessary
Inverse of arcs where $\text{flow}(a) > 0$
Maximum Flow using Residual Graphs

Start with zero flow.
Find an s-t path in the residual graph.
Maximum Flow using Residual Graphs

For inverse arcs in path, subtract flow $K$. 

Diagram:

- Graph with nodes $s$, $v_1$, $v_2$, and $t$.
- Edges with capacities:
  - $s$ to $v_1$: 2
  - $v_1$ to $v_2$: 3
  - $v_2$ to $t$: 1
  - $s$ to $t$: 4

On the right, the residual graph is shown with the same nodes and edges, but with the inverse arcs marked in red and the flow $K$ indicated. The residual graph shows the updated capacities after a flow has been pushed through the graph.
Maximum Flow using Residual Graphs

Choose maximum allowable value of $K$.
For forward arcs in path, add flow $K$. 
Maximum Flow using Residual Graphs

Update the residual graph.
Maximum Flow using Residual Graphs

Find an s-t path in the residual graph.
Choose maximum allowable value of K.
Add K to \( (s,v_2) \) and \( (v_1,t) \). Subtract K from \( (v_1,v_2) \).
Maximum Flow using Residual Graphs

Update the residual graph.
Find an s-t path in the residual graph.
Maximum Flow using Residual Graphs

No more s-t paths. Stop.
Maximum Flow using Residual Graphs

Correct Answer.
Outline

• Preliminaries

• Maximum Flow
  • Residual Graph
  • Max-Flow Min-Cut Theorem

• Algorithms

• Energy minimization with max flow/min cut
## History of Maxflow Algorithms

### Augmenting Path and Push-Relabel

<table>
<thead>
<tr>
<th>year</th>
<th>discoverer(s)</th>
<th>bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1951</td>
<td>Dantzig</td>
<td>$O(n^2mU)$</td>
</tr>
<tr>
<td>1955</td>
<td>Ford &amp; Fulkerson</td>
<td>$O(m^2U)$</td>
</tr>
<tr>
<td>1970</td>
<td>Dinitz</td>
<td>$O(n^2m)$</td>
</tr>
<tr>
<td>1972</td>
<td>Edmonds &amp; Karp</td>
<td>$O(m^2 \log U)$</td>
</tr>
<tr>
<td>1973</td>
<td>Dinitz</td>
<td>$O(nm \log U)$</td>
</tr>
<tr>
<td>1974</td>
<td>Karzanov</td>
<td>$O(n^3)$</td>
</tr>
<tr>
<td>1977</td>
<td>Cherkassky</td>
<td>$O(n^2m^{1/2})$</td>
</tr>
<tr>
<td>1980</td>
<td>Galil &amp; Naamad</td>
<td>$O(nm \log^2 n)$</td>
</tr>
<tr>
<td>1983</td>
<td>Sleator &amp; Tarjan</td>
<td>$O(nm \log n)$</td>
</tr>
<tr>
<td>1986</td>
<td>Goldberg &amp; Tarjan</td>
<td>$O(nm \log (n^2/m))$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja &amp; Orlin</td>
<td>$O(nm + n^2 \log U)$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja et al.</td>
<td>$O(nm \log (n \sqrt{\log U/m}))$</td>
</tr>
<tr>
<td>1989</td>
<td>Cheriyan &amp; Hagerup</td>
<td>$E(nm + n^2 \log^2 n)$</td>
</tr>
<tr>
<td>1990</td>
<td>Cheriyan et al.</td>
<td>$O(n^3/ \log n)$</td>
</tr>
<tr>
<td>1990</td>
<td>Alon</td>
<td>$O(nm + n^{8/3} \log n)$</td>
</tr>
<tr>
<td>1992</td>
<td>King et al.</td>
<td>$O(nm + n^{2+\epsilon})$</td>
</tr>
<tr>
<td>1993</td>
<td>Phillips &amp; Westbrook</td>
<td>$O(nm (\log_{m/n} n + \log^{2+\epsilon} n))$</td>
</tr>
<tr>
<td>1994</td>
<td>King et al.</td>
<td>$O(nm \log_{m/(n \log n)} n)$</td>
</tr>
<tr>
<td>1997</td>
<td>Goldberg &amp; Rao</td>
<td>$O(m^{3/2} \log (n^2/m) \log U)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$O(n^{2/3}m \log (n^2/m) \log U)$</td>
</tr>
</tbody>
</table>

### Notes
- **n**: #nodes
- **m**: #edges
- **U**: maximum edge weight

Algorithms assume non-negative edge weights.

[Slide credit: Andrew Goldberg]
# History of Maxflow Algorithms

## Augmenting Path and Push-Relabel

<table>
<thead>
<tr>
<th>year</th>
<th>discoverer(s)</th>
<th>bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1951</td>
<td>Dantzig</td>
<td>$O(n^2 m U)$</td>
</tr>
<tr>
<td>1955</td>
<td>Ford &amp; Fulkerson</td>
<td>$O(m^2 U)$</td>
</tr>
<tr>
<td>1970</td>
<td>Dinitz</td>
<td>$O(n^2 m)$</td>
</tr>
<tr>
<td>1972</td>
<td>Edmonds &amp; Karp</td>
<td>$O(m^2 \log U)$</td>
</tr>
<tr>
<td>1973</td>
<td>Dinitz</td>
<td>$O(n m \log U)$</td>
</tr>
<tr>
<td>1974</td>
<td>Karzanov</td>
<td>$O(n^3)$</td>
</tr>
<tr>
<td>1977</td>
<td>Cherkassky</td>
<td>$O(n^2 m^{1/2})$</td>
</tr>
<tr>
<td>1980</td>
<td>Galil &amp; Naamad</td>
<td>$O(n m \log^2 n)$</td>
</tr>
<tr>
<td>1983</td>
<td>Sleator &amp; Tarjan</td>
<td>$O(n m \log n)$</td>
</tr>
<tr>
<td>1986</td>
<td>Goldberg &amp; Tarjan</td>
<td>$O(n m \log(n^2/m))$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja &amp; Orlin</td>
<td>$O(n m + n^2 \log U)$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja et al.</td>
<td>$O(n m \log(n \sqrt{\log U/m}))$</td>
</tr>
<tr>
<td>1989</td>
<td>Cheriyan &amp; Hagerup</td>
<td>$E(n m + n^2 \log^2 n)$</td>
</tr>
<tr>
<td>1990</td>
<td>Cheriyan et al.</td>
<td>$O(n^3 / \log n)$</td>
</tr>
<tr>
<td>1990</td>
<td>Alon</td>
<td>$O(n m + n^{6/3} \log n)$</td>
</tr>
<tr>
<td>1992</td>
<td>King et al.</td>
<td>$O(n m + n^{2+\epsilon})$</td>
</tr>
<tr>
<td>1993</td>
<td>Phillips &amp; Westbrook</td>
<td>$O(n m (\log_{m/n} n + \log^{2+\epsilon} n))$</td>
</tr>
<tr>
<td>1994</td>
<td>King et al.</td>
<td>$O(n m \log_m (n \log n) n)$</td>
</tr>
<tr>
<td>1997</td>
<td>Goldberg &amp; Rao</td>
<td>$O(m^{3/2} \log(n^2/m) \log U)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$O(n^{2/3} m \log(n^2/m) \log U)$</td>
</tr>
</tbody>
</table>

- **n**: #nodes
- **m**: #edges
- **U**: maximum edge weight

Algorithms assume non-negative edge weights

[Slide credit: Andrew Goldberg]
Augmenting Path based Algorithms

Ford Fulkerson: Choose any augmenting path
Augmenting Path based Algorithms

Ford Fulkerson: Choose any augmenting path
Augmenting Path based Algorithms

Ford Fulkerson: Choose any augmenting path
Augmenting Path based Algorithms

Ford Fulkerson: Choose any augmenting path
Augmenting Path based Algorithms

Ford Fulkerson: Choose any augmenting path

We will have to perform 2000 augmentations!

Worst case complexity: $O(m \times \text{Total Flow})$
(Pseudo-polynomial bound: depends on flow)
Augmenting Path based Algorithms

Dinitz: Choose **shortest** augmenting path

Worst case complexity: $O(m n^2)$
Maxflow in Computer Vision

- Specialized algorithms for vision problems
  - Grid graphs
  - Low connectivity ($m \sim O(n)$)
- Dual search tree augmenting path algorithm
  [Boykov and Kolmogorov PAMI 2004]
  - Finds approximate shortest augmenting paths efficiently
  - High worst-case time complexity
  - Empirically outperforms other algorithms on vision problems
  - Efficient code available on the web
    e.g., http://pub.ist.ac.at/~vnk/software.html