ECCV 2006 tutorial on

Graph Cuts vs. Level Sets

part IV

Global vs. local optimisation algorithms

Yuri Boykov
University of Western Ontario

Daniel Cremers
University of Bonn

Vladimir Kolmogorov
University College London
Global vs. local minima
(Geodesic active contours)

• Geodesic active contours
  – Variational approach (e.g. level sets)
    • Gradient descent in the *space of contours*
    • Local minimum
    • Non-convex formulation?

  – Graph cuts (e.g. geo-cuts)
    • Same problem, global minimum
    • Convex formulation?

[Anonymous attendee of CVPR’05]:
*How is it possible?*
Global vs. local minima
(Geodesic active contours)

- Geodesic active contours
  - Variational approach (e.g. level sets)
    - Gradient descent in the *space of contours*
    - Local minimum
    - Non-convex formulation?
  - Graph cuts (e.g. geo-cuts)
    - Same problem, global minimum
    - Convex formulation?

[Anonymous attendee of CVPR’05]:
*How is it possible?*
Graph cuts

- Function $E(x)$ of discrete variables: convexity not defined
- Extend the space of solutions: $x_p \in \{0,1\} \Rightarrow x_p \in [0,1]$
  - Allow fractional segmentations
- Extend energy: linear programming (LP) relaxation
  - Now convex problem!

submodular function $\Rightarrow$ integer solution

Energy function with discrete variables

LP relaxation

tight
Graph cuts

- Function $E(x)$ of discrete variables: convexity not defined
- Extend the space of solutions: $x_p \in \{0,1\} \Rightarrow x_p \in [0,1]$  
  - Allow *fractional* segmentations
- Extend energy: *linear programming (LP) relaxation*  
  - Now convex problem!

non-submodular function $\Rightarrow$ fractional solution (in general)
Solving LP relaxation

- Too large for general purpose LP solvers (e.g. interior point methods)
- Solve dual problem instead of primal:
  - Formulate lower bound on the energy
  - Maximize this bound
  - When done, solves primal problem (LP relaxation)
- Two different ways to formulate lower bound
  - Part A: Via posiforms $\Rightarrow$ maxflow algorithm (for binary variables)
  - Part B: Via convex combination of trees $\Rightarrow$ tree-rewighted message passing
Notation and Preliminaries
Energy function

\[ E(x \mid \theta) = \theta_{\text{const}} + \sum_{p} \theta_p(x_p) + \sum_{p,q} \theta_{pq}(x_p, x_q) \]

- \( x_p \) are discrete variables (for example, \( x_p \in \{0, 1\} \))
- \( \theta_p(\cdot) \) are unary potentials
- \( \theta_{pq}(\cdot, \cdot) \) are pairwise potentials
LP relaxation

- [Schlesinger’76, Koster et al.’98, Chekuri et al.’00, Wainwright et al.’03]

- Introduce indicator variables $x_{p;i}$, $x_{pq;ij}$

$$\sum_{p,i} \theta_p(i) x_{p;i} + \sum_{p,q,i,j} \theta_{pq}(i, j) x_{pq;ij} \to \min$$

\[
\begin{align*}
\sum_j x_{pq;ij} &= x_{p;i} \\
\sum_i x_{p;i} &= 1 \\
x_{pq;ij} &\in \{0,1\} \quad \text{relaxation} \quad x_{pq;ij} \in [0,1]
\end{align*}
\]
Energy function - visualisation

\[
E(x | \theta) = \theta_{\text{const}} + \sum_{p} \theta_p(x_p) + \sum_{p,q} \theta_{pq}(x_p, x_q)
\]
Energy function - visualisation

\[ E(x \mid \theta) = \theta_{\text{const}} + \sum_{p} \theta_{p}(x_{p}) + \sum_{p,q} \theta_{pq}(x_{p}, x_{q}) \]

\[ \theta = \text{vector of all parameters} \]

label 0

label 1

node \( p \)

edge \((p, q)\)

node \( q \)
Reparameterisation

\begin{center}
\begin{tikzpicture}
    \node (A) at (0,0) {2};
    \node (B) at (2,0) {1};
    \node (C) at (4,0) {1};
    \node (D) at (0,2) {0};
    \node (E) at (2,2) {0};
    \node (F) at (4,2) {4};

    \draw (A) -- (B);
    \draw (B) -- (C);
    \draw (D) -- (E);
    \draw (E) -- (F);
    \draw[dashed] (A) -- (D);
    \draw[dashed] (B) -- (E);
    \draw[dashed] (C) -- (F);

    \draw (A) -- (2,0.5);
    \draw (B) -- (2,1.5);
    \draw (C) -- (2,-0.5);
    \draw (D) -- (2,0.5);
    \draw (E) -- (2,1.5);
    \draw (F) -- (2,-0.5);

    \node at (-0.5,0) {2};
    \node at (2.5,0) {1};
    \node at (4.5,0) {1};
    \node at (-0.5,2) {0};
    \node at (2.5,2) {0};
    \node at (4.5,2) {4};
    \node at (2,0) {4};
    \node at (2,2) {5};

    \end{tikzpicture}
\end{center}

$0 + 1$
Reparameterisation

- **Definition.** $\theta'$ is a reparameterisation of $\theta$ ($\theta' \equiv \theta$) if they define the same energy:

$$E(x | \theta') = E(x | \theta) \quad \text{for any } x$$

- Maxflow, BP and TRW perform reparameterisations
Part A: Lower bound via posiforms

(⇒ maxflow algorithm)
Lower bound via posiforms
[Hammer, Hansen, Simeone’84]

\[ E(x | \theta) = \theta_{\text{const}} + \sum_{p} \theta_{p}(x_{p}) + \sum_{p,q} \theta_{pq}(x_{p}, x_{q}) \]

\[ \text{maximize} \]

\[ \theta_{\text{const}} \quad - \quad \text{lower bound on the energy:} \]

\[ E(x | \theta) \geq \theta_{\text{const}} \quad \forall x \]
Outline of part A

• Posiform maximisation: algorithm?

• Binary variables, \textit{submodular} functions
  – Reduction to maxflow
  – Global minimum of the energy

• Binary variables, \textit{non-submodular} functions
  – Reduction to maxflow
    • More complicated graph
  – \textit{Part} of optimal solution
Posiform maximisation

Binary variables, submodular functions
**Submodularity and canonical form**

- **Definition:** $E$ is *submodular* if every pairwise term satisfies

\[ \theta_{pq}(0,0) + \theta_{pq}(1,1) \leq \theta_{pq}(0,1) + \theta_{pq}(1,0) \]

- Can be converted to “canonical form”: 

![Diagram showing a network with labeled nodes and edges.](image-url)
Overview of min cut/max flow
Min Cut problem

Directed weighted graph
Min Cut problem

Cut:

\[ S = \{ \text{source, node 1} \} \]
\[ T = \{ \text{sink, node 2, node 3} \} \]
Min Cut problem

Cut:
$S = \{\text{source, node 1}\}$
$T = \{\text{sink, node 2, node 3}\}$
$\text{Cost}(S,T) = 1 + 1 = 2$

• Task: Compute cut with minimum cost
Maxflow algorithm

value(flow) = 0
Maxflow algorithm

\[\text{value(flow)} = 0\]
Maxflow algorithm

\[ \text{value(flow)} = 0 \]
Maxflow algorithm

\[ \text{value(flow)} = 0 \]
Maxflow algorithm

![Graph with labeled edges and a flow value]

\[\text{value(flow)} = 1\]
Maxflow algorithm

\[
\text{value(flow)} = 1
\]
Maxflow algorithm

\[ \text{value(flow)} = 1 \]
Maxflow algorithm

\[ \text{value(flow)} = 1 \]
Maxflow algorithm

value(flow) = 2
Maxflow algorithm

\[\text{value(flow)} = 2\]
Maxflow algorithm

value(flow) = 2
Posiform maximisation

Binary variables, non-submodular functions

Reduction to maxflow
Maxflow algorithm
and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm and reparameterisation
Maxflow algorithm
and reparameterisation

\[
\text{value(flow)} = 0
\]
Maxflow algorithm and reparameterisation

\[
\text{value}(\text{flow}) = 0
\]
Maxflow algorithm and reparameterisation

value(flow) = 1
Maxflow algorithm and reparameterisation

\[
\text{value(flow)} = 1
\]
Maxflow algorithm and reparameterisation

\[ \text{value(flow)} = 2 \]
Maxflow algorithm and reparameterisation

\[ \text{value(flow)} = 2 \]
Maxflow algorithm and reparameterisation

\[ \text{value}(\text{flow}) = 2 \]

Minimum of the energy:

\[ x = (0, 1, 1) \]
Posiform maximisation

Binary variables,
non-submodular functions
Arbitrary functions of binary variables

$$E(x | \theta) = \theta_{\text{const}} + \sum_{p} \theta_p (x_p) + \sum_{p,q} \theta_{pq} (x_p, x_q)$$

- Can be solved via maxflow
  - Specially constructed graph

- Gives solution to LP relaxation: for each node
  $$x_p \in \{0, 1/2, 1\}$$
Arbitrary functions of binary variables

Part of optimal solution
[Hammer, Hansen, Simeone’84]
Graph construction - Main idea

\[ E(\{x_p\}) = \sum E_p(x_p) + \sum E_{pq}(x_p, x_q) + \sum \tilde{E}_{pq}(x_p, x_q) \]

- **Unary**
- **Pairwise submodular**
- **Pairwise non-submodular**

- Double # of variables: \( x_p \rightarrow x_p, x_{\overline{p}} \)
  - Ideally, \( x_{\overline{p}} = 1 - x_p \)

- Write \( E \) as a function of both old and new variables
  - New function is submodular!
Graph construction - Main idea

\[
E(\{x_p\}) = \sum E_p(x_p) + \sum E_{pq}(x_p, x_q) + \sum \tilde{E}_{pq}(x_p, x_q)
\]

\[
E(\{x_p\}, \{x_{\overline{p}}\}) = \sum \frac{E_p(x_p) + E_p(1-x_p)}{2} + \sum \frac{E_{pq}(x_p, x_p) + E_p(1-x_{\overline{p}}, 1-x_{\overline{q}})}{2} + \sum \frac{\tilde{E}_{pq}(x_p, 1-x_q) + \tilde{E}_p(1-x_{\overline{p}}, x_q)}{2}
\]

- Double # of variables: \( x_p \rightarrow x_p, x_{\overline{p}} \)
  - Ideally, \( x_{\overline{p}} = 1 - x_p \)

- Write \( E \) as a function of both old and new variables
  - New function is submodular!
Graph construction - Main idea

\[ E(\{x_p\}) = \sum E_p(x_p) \]

\[ E(\{x_p\}, \{x_{\bar{p}}\}) = \sum \frac{E_p(x_p) + E_p(1 - x_{\bar{p}})}{2} \]

- Double # of variables: \( x_p \rightarrow x_p, x_{\bar{p}} \)
  - Ideally, \( x_{\bar{p}} = 1 - x_p \)

- Write \( E \) as a function of both old and new variables
  - New function is submodular!
Graph construction - Main idea

\[ + \sum E_{pq}(x_p, x_q) \rightarrow + \sum \frac{E_{pq}(x_p, x_p) + E_p(1-x_p, 1-x_q)}{2} \]

- Double # of variables: \( x_p \rightarrow x_p, x_{\overline{p}} \)
  - Ideally, \( x_{\overline{p}} = 1 - x_p \)

- Write \( E \) as a function of both old and new variables
  - New function is submodular!
Graph construction - Main idea

\[ + \sum \tilde{E}_{pq}(x_p, x_q) \]

\[ + \sum \frac{\tilde{E}_{pq}(x_p, 1-x_q)}{2} + \tilde{E}_p(1-x_{\bar{p}}, x_q) \]

- Double # of variables: \( x_p \rightarrow x_p, x_{\bar{p}} \)
  - Ideally, \( x_{\bar{p}} = 1 - x_p \)

- Write \( E \) as a function of both old and new variables
  - New function is submodular!
Graph construction - Main idea

\[ E(\{x_p\}) = \sum E_p(x_p) \]
\[ + \sum E_{pq}(x_p, x_q) \]
\[ + \sum \tilde{E}_{pq}(x_p, x_q) \]

\[ E(\{x_p\}, \{x_{\bar{p}}\}) = \sum \frac{E_p(x_p) + E_p(1-x_{\bar{p}})}{2} \]
\[ + \sum \frac{E_{pq}(x_p, x_p) + E_p(1-x_{\bar{p}}, 1-x_{\bar{q}})}{2} \]
\[ + \sum \frac{\tilde{E}_{pq}(x_p, 1-x_{\bar{q}}) + \tilde{E}_p(1-x_{\bar{p}}, x_q)}{2} \]

- Minimise new function \( E(\{x_p\}, \{x_{\bar{p}}\}) \)
  - Without constraint \( x_{\bar{p}} = 1 - x_p \)
Graph construction

\( x_{\overline{p}} \)

\( x_{p} \)

\( \text{source} \)

\( \text{sink} \)
Graph construction
Graph construction
Graph construction

\[ \begin{array}{c}
\text{source} \\
\text{sink}
\end{array} \]
Assigning labels

- Assign labels based on minimum cut in auxiliary graph

\[ x_p = 1 \quad x_p = 0 \]
\[ x_{\overline{p}} = 0 \quad x_{\overline{p}} = 1 \]

- To maximize # of labeled nodes, choose a particular minimum cut

node \( p \) is unlabeled
Theorem [Hammer, Hansen, Simeone’84]. Labeling $x$ is part of optimal labeling $x^\ast$.

\[ x_p = 0 \quad \quad x_p = 1 \quad \quad x_p = ? \]
Part B: Lower bound via convex combination of trees

($\Rightarrow$ tree-reweighted message passing)
Convex combination of trees
[Wainwright, Jaakkola, Willsky ’02]

- Goal: compute minimum of the energy for $\theta$:
  $$\Phi(\theta) = \min_{x} E(x \mid \theta)$$

- In general, intractable!

- Obtaining lower bound:
  - Split $\theta$ into several components: $\theta = \theta^1 + \theta^2 + ...$
  - Compute minimum for each component:
    $$\Phi(\theta^i) = \min_{x} E(x \mid \theta^i)$$
  - Combine $\Phi(\theta^1), \Phi(\theta^2), ...$ to get a bound on $\Phi(\theta)$

- Use trees!
Convex combination of trees (cont’d)

\[ \theta \equiv \frac{1}{2} \theta^T + \frac{1}{2} \theta^{T'} \]

\[ \Phi(\theta) \geq \frac{1}{2} \Phi(\theta^T) + \frac{1}{2} \Phi(\theta^{T'}) \]

maximize lower bound on the energy
TRW algorithms

- Goal: find reparameterisation maximizing lower bound

- Apply sequence of different reparameterisation operations:
  - Node averaging
  - Ordinary BP on trees

- Order of operations?
  - Affects performance dramatically

- Algorithms:
  - [Wainwright et al. ’02]: parallel schedule
    - May not converge
  - [Kolmogorov’05]: specific sequential schedule
    - Lower bound does not decrease, convergence guarantees
    - Needs half the memory
Experimental results: stereo

- Global minima for some instances with TRW
  [Meltzer, Yanover, Weiss’05]
Parts A and B: Summary

• MAP estimation algorithms are based on LP relaxation
  – Maximize lower bound

• Two ways to formulate lower bound

• Via posiforms: leads to maxflow algorithm (for binary variables)
  – Polynomial time solution
  – Submodular functions: global minimum
  – Non-submodular functions: part of optimal solution

• Via convex combination of trees: leads to TRW algorithm
  – Convergence in the limit (for TRW-S)
  – Applicable to arbitrary energy function
Non-binary variables:
Other methods for solving LP

- No polynomial-time algorithm (except general purpose LP solvers)

- Iterative methods:
  - [Koval,Schlesinger’76]: *augmenting DAG algorithm*
  - [Kovalevsky,Koval’75, Flach’98] (unpublished): *max-sum diffusion*
    - See tech. report [Werner’05]
    - Not guaranteed to solve LP (only *arc consistent* solution) – same as TRW

- Special case: *submodular functions*
  - LP has integer optimal solution [Schlesinger,Flach’00]
  - Reduction to maxflow [Ishikawa’03, D.Schlesinger’05]
Continuous mincut/maxflow
Continuous mincut/maxflow

- Primal problem:
  \[
  \int_C g(C(s)) \, ds \rightarrow \min
  \]

subject to

\[
\begin{cases}
  s \text{ inside } C \\
  t \text{ outside } C
\end{cases}
\]

Alternatively:

\[
\int |\nabla u|_g \rightarrow \min
\]

subject to

\[
\begin{cases}
  u_p = 0, \ p \in s \\
  u_p = 1, \ p \in t
\end{cases}
\]

[total variation]

[Rudin,Osher,Fatemi’92]:
image restoration

[Amar,Belletini’94]:
definition for arbitrary metric
Continuous mincut/maxflow

- Primal problem:
  \[
  \int_C g(C(s)) \, ds \rightarrow \min
  \]
  subject to \[
  \begin{cases}
  s \text{ inside } C \\
  t \text{ outside } C
  \end{cases}
  \]
  Alternatively:
  \[
  \int \| \nabla u \|_g \rightarrow \min
  \]
  subject to \[
  \begin{cases}
  u_p = 0, \ p \in s \\
  u_p = 1, \ p \in t
  \end{cases}
  \]

- \(u_p \in [0,1]\) – fractional segmentations
- Convex problem
- Integer optimal solution
Continuous mincut/maxflow

• Dual problem:

\[
\int_s (\text{div } \vec{f}_p) \, da \rightarrow \text{max} \\
\text{subject to}
\]

\[|\vec{f}_p| \leq g \quad \text{(capacity constraint)}\]

\[\text{div } \vec{f}_p = 0 \quad \text{(flow conservation)}\]

for \( p \notin s,t \)
Reparameterisation

- Any flow with
  \[ \text{div } \vec{f}_p = 0 \quad \text{for } p \notin s, t \]
defines reparameterisation

(by the divergence theorem):

\[ E(C) \equiv \int_C g \, ds = \text{const} + \int_C (g - \vec{f} \cdot \vec{N}) \, ds \]

where

\[ \text{const} = \int_s (\text{div } \vec{f}) \, ds \]
Reparameterisation

\[ E(C) \equiv \int_C g \, ds = \text{const} + \int_C (g - \vec{f} \cdot \vec{N}) \, ds \]

lower bound on \( E(C) \)

\[ |\vec{f}| \leq g \Rightarrow \text{non-negative} \]
Suppose flow saturates cut $C^*$

\[
\vec{f}_p = g_p \vec{N} \quad \text{for } p \in C^*:
\]

$\Rightarrow C^* = \text{minimum cut}$

\[
E(C) \equiv \int_C g \ ds = \text{const} + \int_C (g - \vec{f} \cdot \vec{N}) \ ds
\]

zero for $C^*$
Global vs. local optimisation algorithms: Summary

- Geodesic active contours
  - Variational approach (e.g. level sets)
    - Gradient descent in the *space of contours*
    - Local minimum
    - Non-convex formulation

- Graph cuts (e.g. geo-cuts)
  - Extended space (fractional segmentations)
  - Convex formulation
  - Integer solution (for submodular functions)
Global vs. local optimisation algorithms: Summary

• Geodesic active contours
  – Variational approach (e.g. level sets)
    • Gradient descent in the \textit{space of contours}
    • Local minimum
    • Non-convex formulation

  – Graph cuts (e.g. geo-cuts)
    • Extended space (fractional segmentations)
    • Convex formulation
    • Integer solution (for submodular functions)
Other relaxations/extensions

- Energy $E(x)$ defined for integer configurations ($x_p \in \{0,1\}$)
- How to define for fractional configurations ($x_p \in [0,1]$)?
Other relaxations/extensions

- **LP relaxation** [Schlesinger’76, Koster et al.’98, Chekuri et al.’00, Wainwright et al.’03]
  - Defined for multi-valued variables
  - Convex
  - $E$ is submodular $\Rightarrow$ integer solution

- **Lovász extension** [Lovász’83]
  - Defined for binary variables
  - Always integer solution
  - $E$ is submodular $\Leftrightarrow$ extension is convex
  - “Submodularity” – discrete analogue of convexity

- **Sherali-Adams relaxation**, semi-definite relaxation, SOCP relaxation, ...
LP relaxation and Lovász extension

Submodular function: $E(0,0) + E(1,1) \leq E(0,1) + E(1,0)$
LP relaxation and Lovász extension

Non-submodular function: $E(0,0) + E(1,1) \geq E(0,1) + E(1,0)$