

Part 6: Structured Prediction and Energy Minimization (1/2)

Sebastian Nowozin and Christoph H. Lampert

Colorado Springs, 25th June 2011



Prediction Problem

$$y^* = f(x) = \operatorname{argmax}_{y \in \mathcal{Y}} g(x, y)$$

- ▶ $g(x, y) = p(y|x)$, factor graphs/MRF/CRF,
- ▶ $g(x, y) = -E(y; x, w)$, factor graphs/MRF/CRF,
- ▶ $g(x, y) = \langle w, \psi(x, y) \rangle$, linear model (e.g. multiclass SVM),

→ difficulty: $\mathcal{Y}$ finite but very large

Prediction Problem

$$y^* = f(x) = \operatorname{argmax}_{y \in \mathcal{Y}} g(x, y)$$

- ▶ $g(x, y) = p(y|x)$, factor graphs/MRF/CRF,
- ▶ $g(x, y) = -E(y; x, w)$, factor graphs/MRF/CRF,
- ▶ $g(x, y) = \langle w, \psi(x, y) \rangle$, linear model (e.g. multiclass SVM),

→ difficulty: $\mathcal{Y}$ finite but very large

Prediction Problem (cont)

Definition (Optimization Problem)

Given $(g, \mathcal{Y}, \mathcal{G}, x)$, with *feasible set* $\mathcal{Y} \subseteq \mathcal{G}$ over *decision domain* $\mathcal{G}$, and given an input instance $x \in \mathcal{X}$ and an *objective function* $g : \mathcal{X} \times \mathcal{G} \rightarrow \mathbb{R}$, find the optimal value

$$\alpha = \sup_{y \in \mathcal{Y}} g(x, y),$$

and, if the supremum exists, find an *optimal solution* $y^* \in \mathcal{Y}$ such that $g(x, y^*) = \alpha$.

The feasible set

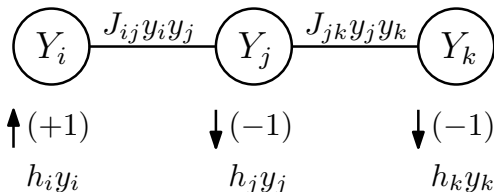
Ingredients

- ▶ Decision domain $\mathcal{G}$,
typically simple ($\mathcal{G} = \mathbb{R}^d$, $\mathcal{G} = 2^V$, etc.)
- ▶ Feasible set $\mathcal{Y} \subseteq \mathcal{G}$,
defining the problem-specific structure
- ▶ Objective function $g : \mathcal{X} \times \mathcal{G} \rightarrow \mathbb{R}$.

Terminology

- ▶ $\mathcal{Y} = \mathcal{G}$: *unconstrained* optimization problem,
- ▶ $\mathcal{G}$ finite: *discrete* optimization problem,
- ▶ $\mathcal{G} = 2^\Sigma$ for ground set Σ : *combinatorial* optimization problem,
- ▶ $\mathcal{Y} = \emptyset$: *infeasible* problem.

Example: Feasible Sets (cont)



- ▶ *Ising model* with external field
- ▶ Graph $G = (V, E)$
- ▶ “External field”: $h \in \mathbb{R}^V$
- ▶ Interaction matrix: $J \in \mathbb{R}^{V \times V}$
- ▶ Objective, defined on $y_i \in \{-1, 1\}$

$$g(y) = h_i y_i + h_j y_j + h_k y_k + \frac{1}{2} J_{ij} y_i y_j + \frac{1}{2} J_{jk} y_j y_k$$

Example: Feasible Sets (cont)

Ising model with external field

$$\mathcal{Y} = \mathcal{G} = \{-1, +1\}^V$$

$$g(y) = \frac{1}{2} \sum_{(i,j) \in E} J_{i,j} y_i y_j + \sum_{i \in V} h_i y_i$$

- ▶ Unconstrained
- ▶ Objective function contains quadratic terms

Example: Feasible Sets (cont)

$$\mathcal{G} = \{0, 1\}^{(V \times \{-1, +1\}) \cup (E \times \{-1, +1\} \times \{-1, +1\})},$$

$$\mathcal{Y} = \{y \in \mathcal{G} : \forall i \in V : y_{i,-1} + y_{i,+1} = 1,$$

$$\forall (i, j) \in E : y_{i,j,+1,+1} + y_{i,j,+1,-1} = y_{i,+1},$$

$$\forall (i, j) \in E : y_{i,j,-1,+1} + y_{i,j,-1,-1} = y_{i,-1}\},$$

$$\begin{aligned} g(y) = & \frac{1}{2} \sum_{(i,j) \in E} J_{i,j}(y_{i,j,+1,+1} + y_{i,j,-1,-1}) \\ & - \frac{1}{2} \sum_{(i,j) \in E} J_{i,j}(y_{i,j,+1,-1} + y_{i,j,-1,+1}) \\ & + \sum_{i \in V} h_i(y_{i,+1} - y_{i,-1}) \end{aligned}$$

- ▶ Constrained, more variables
- ▶ Objective function contains linear terms only

Evaluating f : what do we want?

$$f(x) = \operatorname{argmax}_{y \in \mathcal{Y}} g(x, y)$$

For evaluating $f(x)$ we want an algorithm that

1. is *general*: applicable to all instances of the problem,
2. is *optimal*: provides an optimal y^* ,
3. has good *worst-case complexity*: for all instances the runtime and space is acceptably bounded,
4. is *integral*: its solutions are restricted to $\mathcal{Y}$,
5. is *deterministic*: its results and runtime are reproducible and depend on the input data only.

Evaluating f : what do we want?

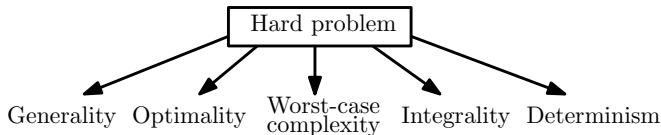
$$f(x) = \operatorname{argmax}_{y \in \mathcal{Y}} g(x, y)$$

For evaluating $f(x)$ we want an algorithm that

1. is *general*: applicable to all instances of the problem,
2. is *optimal*: provides an optimal y^* ,
3. has good *worst-case complexity*: for all instances the runtime and space is acceptably bounded,
4. is *integral*: its solutions are restricted to $\mathcal{Y}$,
5. is *deterministic*: its results and runtime are reproducible and depend on the input data only.

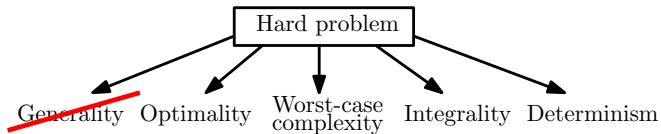
wanting all of them → impossible

Giving up some properties



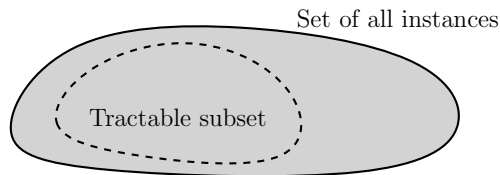
→ giving up one or more properties

- ▶ allows us to design algorithms satisfying the remaining properties
- ▶ might be sufficient for the task at hand



Giving up Generality

- Identify an interesting and tractable subset of instances



Example: MAP Inference in Markov Random Fields

Although NP-hard in general, it is tractable...

- ▶ with low tree-width (Lauritzen, Spiegelhalter, 1988)
- ▶ with binary states, pairwise submodular interactions (Boykov, Jolly, 2001)
- ▶ with binary states, pairwise interactions (only), planar graph structure (Globerson, Jaakkola, 2006)
- ▶ with submodular pairwise interactions (Schlesinger, 2006)
- ▶ with $\mathcal{P}^n$ -Potts higher order factors (Kohli, Kumar, Torr, 2007)
- ▶ with perfect graph structure (Jebara, 2009)

Binary Graph-Cuts

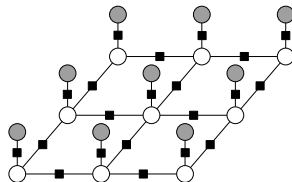
- ▶ Energy function: unary and pairwise

$$E(y; x, w) = \sum_{F \in \mathcal{F}_1} E_F(y_F; x, w_{t_F}) + \sum_{F \in \mathcal{F}_2} E_F(y_F; x, w_{t_F})$$

- Restriction 1 (wlog)

$$E_F(y_i; x, w_{t_F}) \geq 0$$

- Restriction 2
(regular/submodular/attractive)



Binary Graph-Cuts

- Energy function: unary and pairwise

$$E(y; x, w) = \sum_{F \in \mathcal{F}_1} E_F(y_F; x, w_{t_F}) + \sum_{F \in \mathcal{F}_2} E_F(y_F; x, w_{t_F})$$

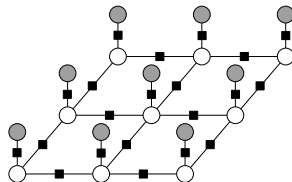
- Restriction 1 (wlog)

$$E_F(y_i; x, w_{t_F}) \geq 0$$

- Restriction 2
(regular/submodular/attractive)

$$E_F(y_i, y_j; x, w_{t_F}) = 0, \quad \text{if } y_i = y_j,$$

$$E_F(y_i, y_j; x, w_{t_F}) = E_F(y_j, y_i; x, w_{t_F}) \geq 0, \quad \text{otherwise.}$$

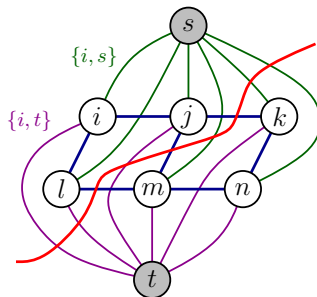


Binary Graph-Cuts (cont)

- ▶ Construct auxiliary undirected graph
- ▶ One node $\{i\}_{i \in V}$ per variable
- ▶ Two extra nodes: source s , sink t
- ▶ Edges

Edge	Graph cut weight
$\{i, j\}$	$E_F(y_i = 0, y_j = 1; \mathbf{x}, w_{t_F})$
$\{i, s\}$	$E_F(y_i = 1; \mathbf{x}, w_{t_F})$
$\{i, t\}$	$E_F(y_i = 0; \mathbf{x}, w_{t_F})$

- ▶ Find linear s - t -mincut
- ▶ Solution defines optimal binary labeling of the original energy minimization problem

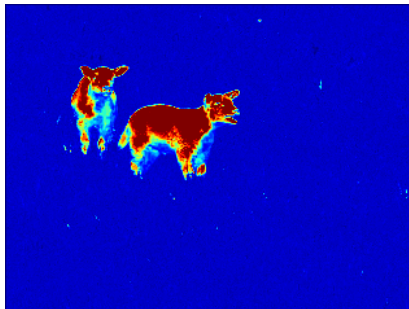


Example: Figure-Ground Segmentation



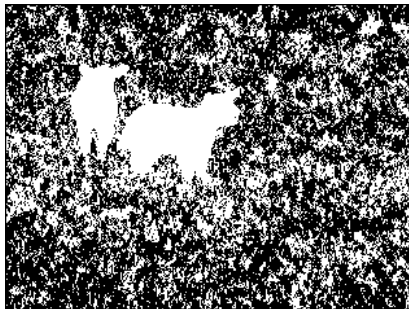
Input image (<http://pdphoto.org>)

Example: Figure-Ground Segmentation



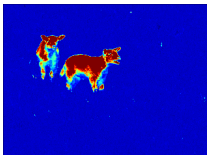
Color model log-odds

Example: Figure-Ground Segmentation



Independent decisions

Example: Figure-Ground Segmentation



$$g(x, y, w) = \sum_{i \in V} \log p(y_i | x_i) + w \sum_{(i,j) \in E} C(x_i, x_j) I(y_i \neq y_j)$$

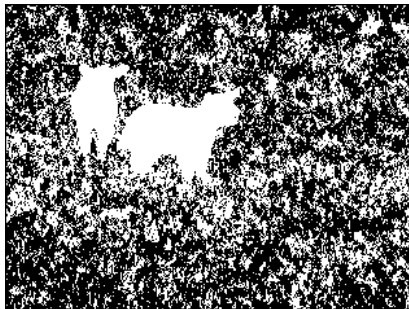
- Gradient strength

$$C(x_i, x_j) = \exp(\gamma \|x_i - x_j\|^2)$$

γ estimated from mean edge strength (Blake et al, 2004)

- $w \geq 0$ controls smoothing

Example: Figure-Ground Segmentation



$$w = 0$$

Example: Figure-Ground Segmentation



Small $w > 0$

Example: Figure-Ground Segmentation



Medium $w > 0$

Example: Figure-Ground Segmentation



Large $w > 0$

General Binary Case

- ▶ Is there a larger class of energies for which binary graph cuts are applicable?
- ▶ (Kolmogorov and Zabih, 2004), (Freedman and Drineas, 2005)

Theorem (Regular Binary Energies)

$$E(y; x, w) = \sum_{F \in \mathcal{F}_1} E_F(y_F; x, w_{t_F}) + \sum_{F \in \mathcal{F}_2} E_F(y_F; x, w_{t_F})$$

is a energy function of binary variables containing only unary and pairwise factors. The discrete energy minimization problem $\operatorname{argmin}_y E(y; x, w)$ is representable as a graph cut problem if and only if all pairwise energy functions E_F for $F \in \mathcal{F}_2$ with $F = \{i, j\}$ satisfy

$$E_{i,j}(0, 0) + E_{i,j}(1, 1) \leq E_{i,j}(0, 1) + E_{i,j}(1, 0).$$

General Binary Case

- ▶ Is there a larger class of energies for which binary graph cuts are applicable?
- ▶ (Kolmogorov and Zabih, 2004), (Freedman and Drineas, 2005)

Theorem (Regular Binary Energies)

$$E(y; x, w) = \sum_{F \in \mathcal{F}_1} E_F(y_F; x, w_{t_F}) + \sum_{F \in \mathcal{F}_2} E_F(y_F; x, w_{t_F})$$

is a energy function of binary variables containing only unary and pairwise factors. The discrete energy minimization problem $\operatorname{argmin}_y E(y; x, w)$ is representable as a graph cut problem if and only if all pairwise energy functions E_F for $F \in \mathcal{F}_2$ with $F = \{i, j\}$ satisfy

$$E_{i,j}(0, 0) + E_{i,j}(1, 1) \leq E_{i,j}(0, 1) + E_{i,j}(1, 0).$$

Example: Class-independent Object Hypotheses

- ▶ (Carreira and Sminchisescu, 2010)
PASCAL VOC 2009/2010 segmentation winner
- ▶ Generate class-independent object hypotheses
- ▶ Energy (almost) as before

$$g(x, y, w) = \sum_{i \in V} E_i(y_i) + w \sum_{(i,j) \in E} C(x_i, x_j) I(y_i \neq y_j)$$

- ▶ Fixed unaries

$$E_i(y_i) = \begin{cases} \infty & \text{if } i \in V_{fg} \text{ and } y_i = 0 \\ \infty & \text{if } i \in V_{bg} \text{ and } y_i = 1 \\ 0 & \text{otherwise} \end{cases}$$

- ▶ Test all $w \geq 0$ using *parametric max-flow*
(Picard and Queyranne, 1980), (Kolmogorov et al., 2007)

Example: Class-independent Object Hypotheses

- ▶ (Carreira and Sminchisescu, 2010)
PASCAL VOC 2009/2010 segmentation winner
- ▶ Generate class-independent object hypotheses
- ▶ Energy (almost) as before

$$g(x, y, w) = \sum_{i \in V} E_i(y_i) + w \sum_{(i,j) \in E} C(x_i, x_j) I(y_i \neq y_j)$$

- ▶ Fixed unaries

$$E_i(y_i) = \begin{cases} \infty & \text{if } i \in V_{fg} \text{ and } y_i = 0 \\ \infty & \text{if } i \in V_{bg} \text{ and } y_i = 1 \\ 0 & \text{otherwise} \end{cases}$$

- ▶ Test all $w \geq 0$ using *parametric max-flow*
(Picard and Queyranne, 1980), (Kolmogorov et al., 2007)

Example: Class-independent Object Hypotheses

- ▶ (Carreira and Sminchisescu, 2010)
PASCAL VOC 2009/2010 segmentation winner
- ▶ Generate class-independent object hypotheses
- ▶ Energy (almost) as before

$$g(x, y, w) = \sum_{i \in V} E_i(y_i) + w \sum_{(i,j) \in E} C(x_i, x_j) I(y_i \neq y_j)$$

- ▶ Fixed unaries

$$E_i(y_i) = \begin{cases} \infty & \text{if } i \in V_{fg} \text{ and } y_i = 0 \\ \infty & \text{if } i \in V_{bg} \text{ and } y_i = 1 \\ 0 & \text{otherwise} \end{cases}$$

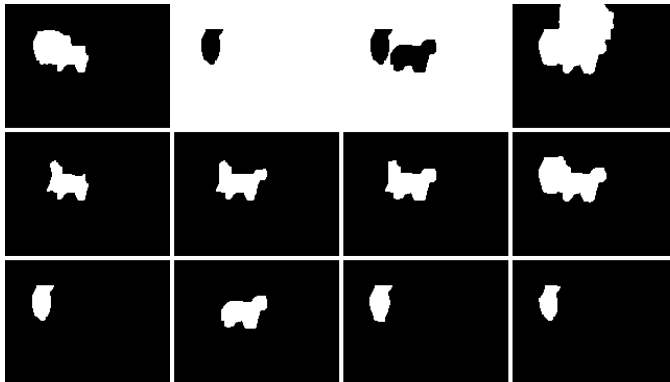
- ▶ Test all $w \geq 0$ using *parametric max-flow*
(Picard and Queyranne, 1980), (Kolmogorov et al., 2007)

Example: Class-independent Object Hypotheses (cont)

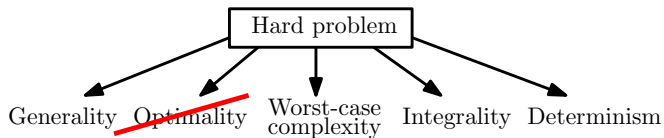


Input image (<http://pdphoto.org>)

Example: Class-independent Object Hypotheses (cont)



CPMC proposal segmentations (Carreira and Sminchisescu, 2010)



Giving up Optimality

Solving for y^* is hard, but is it necessary?

- ▶ pragmatic motivation: in many applications a close-to-optimal solution is good enough
- ▶ computational motivation: set of “good” solutions might be large and finding just one element can be easy

For machine learning models

- ▶ *modeling error*: we always use the wrong model
- ▶ *estimation error*: preference for y^* might be an artifact

Giving up Optimality

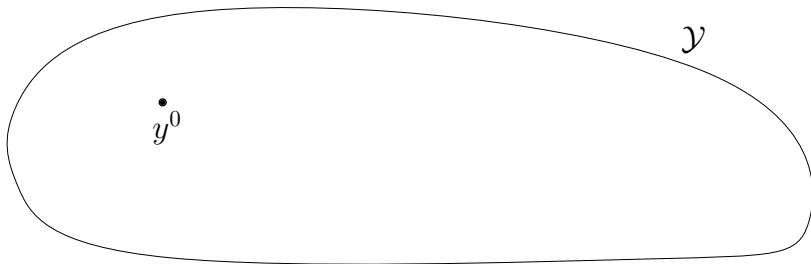
Solving for y^* is hard, but is it necessary?

- ▶ pragmatic motivation: in many applications a close-to-optimal solution is good enough
- ▶ computational motivation: set of “good” solutions might be large and finding just one element can be easy

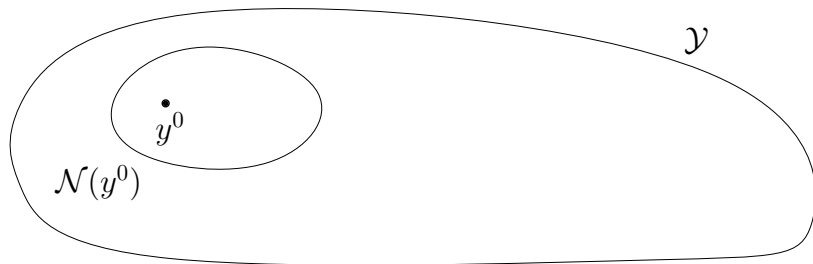
For machine learning models

- ▶ *modeling error*: we always use the wrong model
- ▶ *estimation error*: preference for y^* might be an artifact

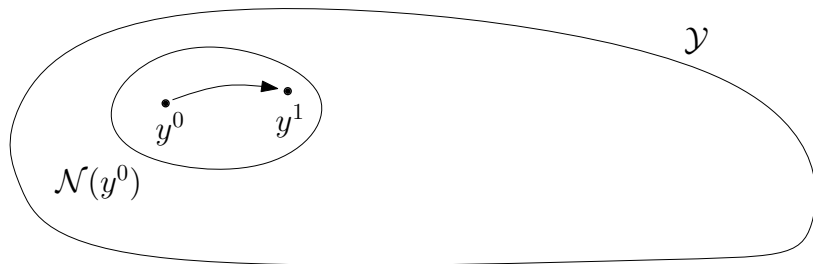
Local Search



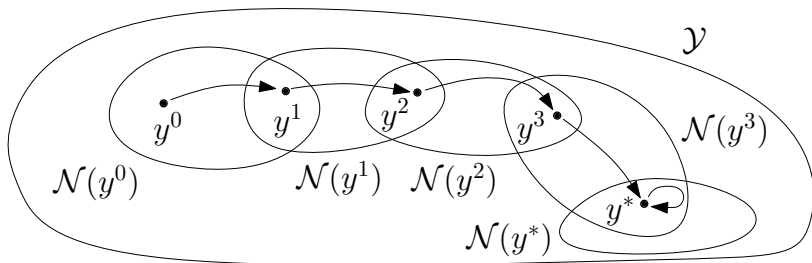
Local Search



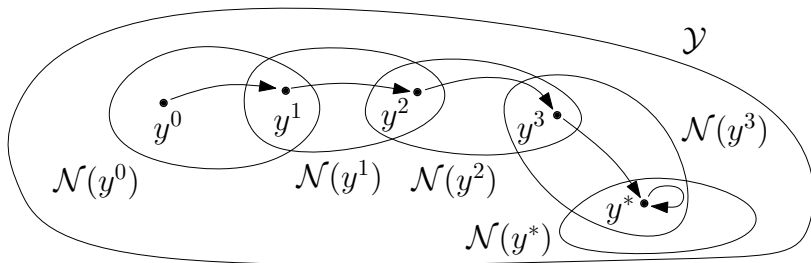
Local Search



Local Search



Local Search

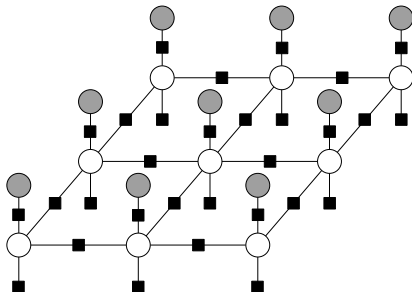


- ▶ $\mathcal{N}_t : \mathcal{Y} \rightarrow 2^{\mathcal{Y}}$, neighborhood system
- ▶ Optimization with respect to $\mathcal{N}_t(y)$ must be tractable:

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

Example: Iterated Conditional Modes (ICM)

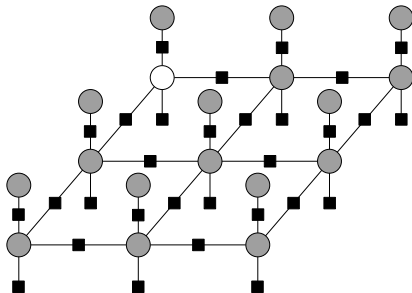
Iterated Conditional Modes (ICM), (Besag, 1986)



- ▶ $g(x, y) = \log p(y|x)$
- ▶ $y^* = \operatorname{argmax}_{y \in \mathcal{Y}} \log p(y|x)$
- ▶ Neighborhoods $\mathcal{N}_s(y) = \{(y_1, \dots, y_{s-1}, z_s, y_{s+1}, \dots, y_S) | z_s \in \mathcal{Y}_s\}$

Example: Iterated Conditional Modes (ICM)

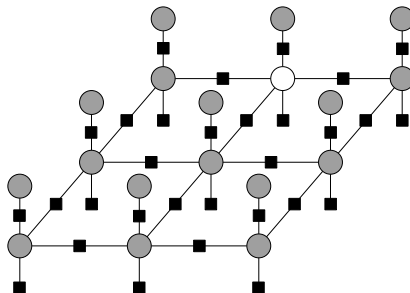
Iterated Conditional Modes (ICM), (Besag, 1986)



$$\blacktriangleright y^{t+1} = \underset{y_1 \in \mathcal{Y}_1}{\operatorname{argmax}} \log p(y_1, y_2^t, \dots, y_V^t | x)$$

Example: Iterated Conditional Modes (ICM)

Iterated Conditional Modes (ICM), (Besag, 1986)



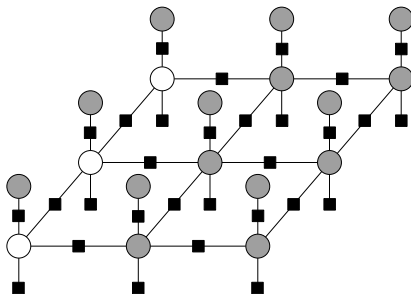
$$\blacktriangleright y^{t+1} = \underset{y_2 \in \mathcal{Y}_2}{\operatorname{argmax}} \log p(y_1^t, y_2, y_3^t, \dots, y_V^t | x)$$

Neighborhood Size

- ▶ ICM neighborhood $\mathcal{N}_t(y^t)$: all states reachable from y^t by changing a single variable (Besag, 1986)
- ▶ Neighborhood size: in general, larger is better (VLSN, Ahuja, 2000)
- ▶ Example: neighborhood along chains

Example: Block ICM

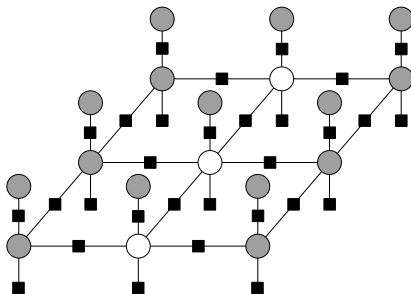
Block Iterated Conditional Modes (ICM)
(Kelm et al., 2006), (Kittler and Föglein, 1984)



$$\blacktriangleright y^{t+1} = \operatorname{argmax}_{y_{C_1} \in \mathcal{Y}_{C_1}} \log p(y_{C_1}, y_{V \setminus C_1}^t | x)$$

Example: Block ICM

Block Iterated Conditional Modes (ICM)
(Kelm et al., 2006), (Kittler and Föglein, 1984)



$$\blacktriangleright y^{t+1} = \underset{y_{C_2} \in \mathcal{Y}_{C_2}}{\operatorname{argmax}} \log p(y_{C_2}, y_{V \setminus C_2}^t | x)$$

Example: Multilabel Graph-Cut

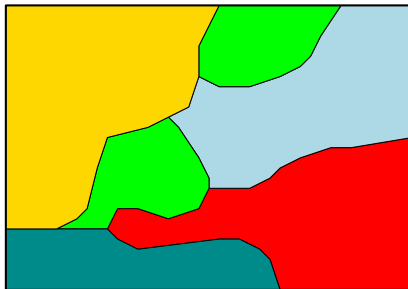
- ▶ Binary graph-cuts are not applicable to multilabel energy minimization problems
- ▶ (Boykov et al., 2001): two local search algorithms for multilabel problems
- ▶ Sequence of binary directed s - t -mincut problems
- ▶ Iteratively improve multilabel solution

α - β Swap Neighborhood

- ▶ Select two different labels α and β
- ▶ Fix all variables i for which $y_i \notin \{\alpha, \beta\}$
- ▶ Optimize over remaining i with $y_i \in \{\alpha, \beta\}$

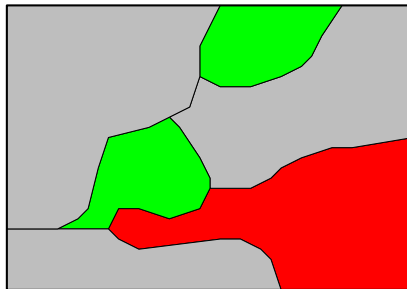
$$\begin{aligned}\mathcal{N}_{\alpha, \beta} : \mathcal{Y} \times \mathbb{N} \times \mathbb{N} &\rightarrow 2^{\mathcal{Y}}, \\ \mathcal{N}_{\alpha, \beta}(y, \alpha, \beta) &:= \{z \in \mathcal{Y} : z_i = y_i \text{ if } y_i \notin \{\alpha, \beta\}, \\ &\quad \text{otherwise } z_i \in \{\alpha, \beta\}\}.\end{aligned}$$

α - β -swap illustrated



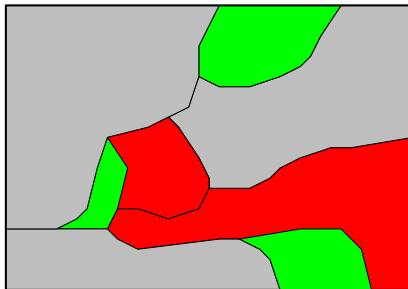
- ▶ 5-label problem
- ▶ α — β -swap

α - β -swap illustrated



- ▶ 5-label problem
- ▶ α — β -swap

α - β -swap illustrated



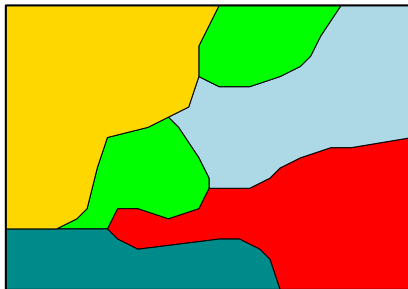
- ▶ 5-label problem
- ▶ α — β -swap

α - β -swap illustrated



- ▶ 5-label problem
- ▶ α — β -swap

α - β -swap illustrated



- ▶ 5-label problem
- ▶ α — β -swap

α - β -swap derivation

$$y^{t+1} = \operatorname{argmin}_{y \in \mathcal{N}_{\alpha, \beta}(y^t, \alpha, \beta)} E(y; x)$$

- ▶ Constant: drop out
- ▶ Unary: combine
- ▶ Pairwise: binary pairwise

α - β -swap derivation

$$y^{t+1} = \operatorname{argmin}_{y \in \mathcal{N}_{\alpha, \beta}(y^t, \alpha, \beta)} \sum_{i \in V} E_i(y_i; x) + \sum_{(i, j) \in E} E_{i, j}(y_i, y_j; x)$$

- ▶ Constant: drop out
- ▶ Unary: combine
- ▶ Pairwise: binary pairwise

α - β -swap derivation

$$\begin{aligned}
 y^{t+1} = \operatorname{argmin}_{y \in \mathcal{N}_{\alpha, \beta}(y^t, \alpha, \beta)} & \left[\sum_{\substack{i \in V, \\ y_i^t \notin \{\alpha, \beta\}}} E_i(y_i^t; x) + \sum_{\substack{i \in V, \\ y_i^t \in \{\alpha, \beta\}}} E_i(y_i; x) \right. \\
 & + \sum_{\substack{(i, j) \in E, \\ y_i^t \notin \{\alpha, \beta\}, y_j^t \notin \{\alpha, \beta\}}} E_{i, j}(y_i^t, y_j^t; x) + \sum_{\substack{(i, j) \in E, \\ y_i^t \in \{\alpha, \beta\}, y_j^t \notin \{\alpha, \beta\}}} E_{i, j}(y_i, y_j^t; x) \\
 & \left. + \sum_{\substack{(i, j) \in E, \\ y_i^t \notin \{\alpha, \beta\}, y_j^t \in \{\alpha, \beta\}}} E_{i, j}(y_i^t, y_j; x) + \sum_{\substack{(i, j) \in E, \\ y_i^t \in \{\alpha, \beta\}, y_j^t \in \{\alpha, \beta\}}} E_{i, j}(y_i, y_j; x) \right].
 \end{aligned}$$

- ▶ Constant: drop out
- ▶ Unary: combine
- ▶ Pairwise: binary pairwise

α - β -swap derivation

$$\begin{aligned}
 y^{t+1} = \operatorname{argmin}_{y \in \mathcal{N}_{\alpha, \beta}(y^t, \alpha, \beta)} & \left[\sum_{\substack{i \in V, \\ y_i^t \notin \{\alpha, \beta\}}} E_i(y_i^t; x) + \sum_{\substack{i \in V, \\ y_i^t \in \{\alpha, \beta\}}} E_i(y_i; x) \right. \\
 & + \sum_{\substack{(i,j) \in E, \\ y_i^t \notin \{\alpha, \beta\}, y_j^t \notin \{\alpha, \beta\}}} E_{i,j}(y_i^t, y_j^t; x) + \sum_{\substack{(i,j) \in E, \\ y_i^t \in \{\alpha, \beta\}, y_j^t \notin \{\alpha, \beta\}}} E_{i,j}(y_i, y_j^t; x) \\
 & \left. + \sum_{\substack{(i,j) \in E, \\ y_i^t \notin \{\alpha, \beta\}, y_j^t \in \{\alpha, \beta\}}} E_{i,j}(y_i^t, y_j; x) + \sum_{\substack{(i,j) \in E, \\ y_i^t \in \{\alpha, \beta\}, y_j^t \in \{\alpha, \beta\}}} E_{i,j}(y_i, y_j; x) \right].
 \end{aligned}$$

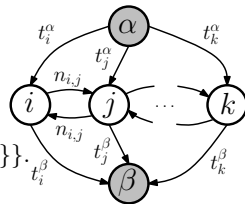
- **Constant:** drop out
- **Unary:** combine
- **Pairwise:** binary pairwise

α - β -swap graph construction

- Directed graph $G' = (V', \mathcal{E}')$

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

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



- Edge weights t_i^α , t_i^β , and $n_{i,j}$

$$n_{i,j} = E_{i,j}(\alpha, \beta; x)$$

$$t_i^\alpha = E_i(\alpha; x) + \sum_{\substack{(i,j) \in \mathcal{E}, \\ y_j \notin \{\alpha, \beta\}}} E_{i,j}(\alpha, y_j; x)$$

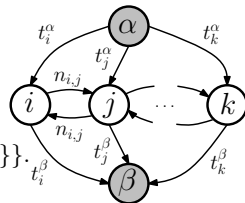
$$t_i^\beta = E_i(\beta; x) + \sum_{\substack{(i,j) \in \mathcal{E}, \\ y_j \notin \{\alpha, \beta\}}} E_{i,j}(\beta, y_j; x)$$

α - β -swap graph construction

- Directed graph $G' = (V', \mathcal{E}')$

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

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

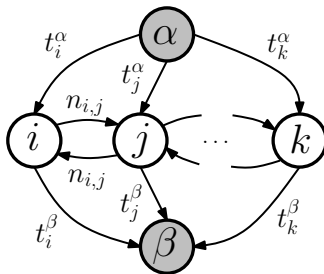


- Edge weights t_i^α , t_i^β , and $n_{i,j}$

$$n_{i,j} = E_{i,j}(\alpha, \beta; \mathbf{x})$$

$$t_i^\alpha = E_i(\alpha; \mathbf{x}) + \sum_{\substack{(i,j) \in \mathcal{E}, \\ y_j \notin \{\alpha, \beta\}}} E_{i,j}(\alpha, y_j; \mathbf{x})$$

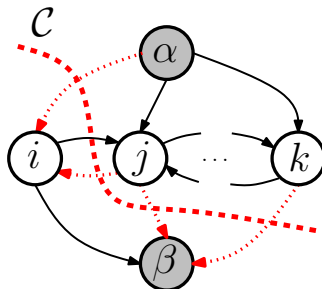
$$t_i^\beta = E_i(\beta; \mathbf{x}) + \sum_{\substack{(i,j) \in \mathcal{E}, \\ y_j \notin \{\alpha, \beta\}}} E_{i,j}(\beta, y_j; \mathbf{x})$$



$$E_{i,j}(y_i, y_j; \mathbf{x}) = 0 \Leftrightarrow y_i = y_j$$

$$E_{i,j}(y_i, y_j; x) = E_{i,j}(y_j, y_i; x) \geq 0$$

α - β -swap move

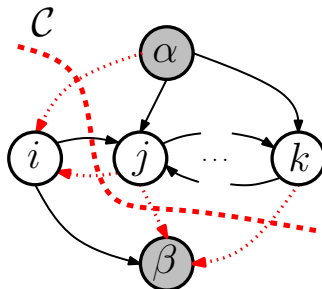


- ▶ Side of cut determines $y_i \in \{\alpha, \beta\}$
- ▶ Iterate all possible (α, β) combinations
- ▶ Semi-metric requirement on pairwise energies

$$E_{i,j}(y_i, y_j; x) = 0 \Leftrightarrow y_i = y_j$$

$$E_{i,j}(y_i, y_j; x) = E_{i,j}(y_j, y_i; x) \geq 0$$

α - β -swap move



- ▶ Side of cut determines $y_i \in \{\alpha, \beta\}$
- ▶ Iterate all possible (α, β) combinations
- ▶ Semi-metric requirement on pairwise energies

$$E_{i,j}(y_i, y_j; \mathbf{x}) = 0 \Leftrightarrow y_i = y_j$$

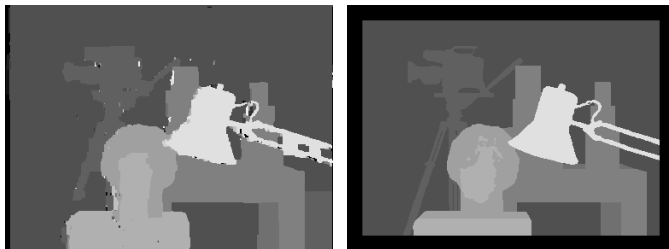
$$E_{i,j}(y_i, y_j; \mathbf{x}) = E_{i,j}(y_j, y_i; \mathbf{x}) \geq 0$$

Example: Stereo Disparity Estimation



- ▶ Infer depth from two images
- ▶ Discretized multi-label problem
- ▶ α -expansion solution close to optimal

Example: Stereo Disparity Estimation



- ▶ Infer depth from two images
- ▶ Discretized multi-label problem
- ▶ α -expansion solution close to optimal

Model Reduction

- ▶ Energy minimization problem: many decision to make jointly
- ▶ Model reduction
 1. Fix a subset of decisions
 2. Optimize the smaller remaining model
- ▶ Example: forcing $y_i = y_j$ for pairs (i, j)

Example: Superpixels in Labeling Problems



Input image: 500-by-375 pixels (187,500 decisions)

Example: Superpixels in Labeling Problems



Image with 149 superpixels (149 decisions)