



Minimum Cut / Maximum Flow



Min Cut / Max Flow Energy Minimisation



COMPSCI 773 S1 T
VISION GUIDED CONTROL
A/P Georgy Gimel'farb

Dissimilarity Minimisation

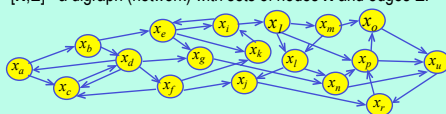
- 3-D surface by minimising energy (dissimilarity) of stereo images:
 - Combinatorial optimisation on graphs specifying relationships between neighbouring pairs of disparities and image signals
 - Generally, an **NP-hard** problem (the **exponential** complexity)
 - Energy (dissimilarity) accumulates weights of nodes and edges
 - Approximate iterative **polynomial-time** solution
 - Maximum flow / minimum cut** algorithms applied to special graphs
 - Solution is provably within a fixed factor of the global minimum
 - General **maximum flow** problem for a network, or a directed graph (digraph) G with two special nodes: a source, s , and a sink, t

COMPSCI 773 1

Basic Notation



- $G=[N,E]$ - a digraph (network) with sets of nodes N and edges E :



$N = \{x_a, x_b, x_c, \dots\}; E = \{(x_a, x_b), (x_a, x_c), \dots\} \subseteq N^2$

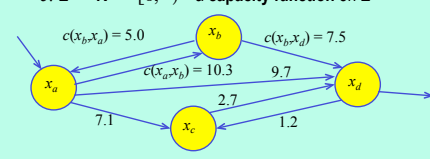
- Chain:** a sequence x_1, \dots, x_n such that $(x_i, x_{i+1}) \in E$
- Path:** a sequence x_1, \dots, x_n such that either $(x_i, x_{i+1}) \in E$ or $(x_{i+1}, x_i) \in E$
- Set of the subsequent nodes "after x ": $A(x) = \{y \in N \mid (x, y) \in E\}$
- Set of the preceding nodes "before x ": $B(x) = \{y \in N \mid (y, x) \in E\}$

COMPSCI 773 2



Flows in Networks

- $c(x,y) \geq 0$ - a *non-negative capacity* of $(x,y) \in E$
- $c: E \rightarrow \mathbb{R}^0 = [0, \infty)$ - a **capacity function** on E



- s, t - the two distinguished nodes (*source, sink*)
 - Edges then can be considered as "water pipes"...

COMPSCI 773 3

Flows in Networks

Static flow of value v from s to t in $[N, E]$ is a function $f: E \rightarrow \mathbb{R}^0$ satisfying linear conditions:



- The flow through every edge does not exceed the edge capacity

$$\forall_{(x,y) \in E} f(x,y) \leq c(x,y)$$

$$\sum_{y \in A(x)} f(x,y) - \sum_{y \in B(x)} f(y,x) = \begin{cases} v & x = s \\ 0 & x \neq s, t \\ -v & x = t \end{cases}$$

- Every node except s and t has equal inflow and outflow

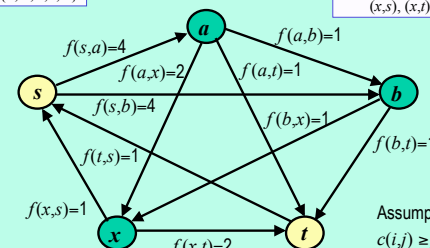
COMPSCI 773 4

Flow of Value 3


$N = \{a, b, s, t, x\}$

$E = \{(a,b), (a,t), (a,x), (b,t), (b,x), (s,a), (s,b), (t,s), (x,s), (x,t)\}$



Assumption: $c(i,j) \geq f(i,j)$

COMPSCI 773 5



Minimum Cut / Maximum Flow

Static Max Flow Problem

- Maximise the flow v subject to the flow constraints:

$$\left\{ \begin{array}{l} \max v : \forall_{(x,y) \in E} f(x,y) \leq c(x,y) \\ \sum_{y \in A(x)} f(x,y) - \sum_{y \in B(x)} f(y,x) = \begin{cases} v & x = s \\ 0 & x \neq s,t; \quad x \in N \\ -v & x = t \end{cases} \end{array} \right.$$
- A cut C of the network $[N; E]$ is a **set of edges** such that their removal separates the source s from the sink t
 - The cut breaks every chain of nodes from the source to the sink
- The **capacity of the cut C** is the total capacity of its edges, i.e. the sum of their capacities

COMPSCI 773 6

Cuts and Capacities

Example: the set of edges $C = \{(s,y), (x,y), (x,t)\}$ is a cut separating s and t

Capacity of the cut $c(C) = c(s,y) + c(x,y) + c(x,t) = 3 + 1 + 3 = 7$

Flow through the cut $f(C) = f(s,y) + f(x,y) + f(x,t) = 2 + 0 + 2 = 4$

COMPSCI 773 7

Flow vs. Capacity of the Cut

Lemma 1 [Ford,Fulkerson;1956]:
 Let a flow f from the source s to the sink t in a network $[N;E]$ have value v
 Let C be a cut that separates s from t
 Then the difference between the **forward flow** $f_{s \rightarrow t}(C)$ from s to t through C and the **reverse flow** $f_{t \rightarrow s}(C)$ from t to s through C is equal to v and is not greater than the capacity of the cut:

$$v = f_{s \rightarrow t}(C) - f_{t \rightarrow s}(C) \leq c(C)$$

COMPSCI 773 8

Meaning of Lemma 1

The equality in Lemma 1:
 the value v of a flow from the source s to the sink t is equal to the **net flow** across any cut separating s and t

The inequality in Lemma 1:
 the net flow across any cut separating s and t does not exceed the capacity of the cut

Thus, the net flow from s to t is bounded by the capacities of the cuts separating s and t

COMPSCI 773 9

Maximal Flow / Minimum Cut

Max-flow min-cut theorem [Ford,Fulkerson;1956]: For any network the maximum flow value from s to t is equal to the minimum cut capacity of all cuts separating s and t

Corollary 1: A flow is **maximum** if and only if (iff) there is no **flow augmenting path** with respect to f

- A path from s to t is a **flow augmenting path** w.r.t. a flow f if $f < c$ on forward edges of the path and $f > 0$ on reverse edges of the path
- Fundamental importance of the corollary:** to increase the value of a flow, improvements are of a very restricted kind!

COMPSCI 773 10

Maximal Flow / Minimum Cut

- An edge (x,y) is **saturated** w.r.t. a flow f if $f(x,y) = c(x,y)$ and is **flowless** w.r.t. f if $f(x,y) = 0$

Corollary 2: A cut C is **minimum** iff every maximum flow f saturates all forward edges of the cut whereas all reverse edges of the cut are flowless w.r.t. f

- Meaning of Corollary 2:** there are no flow augmenting paths w.r.t. the maximum flow
- The case of many sources and sinks with unrestricted flows is equivalent to a single source, single sink case

COMPSCI 773 11

Minimum Cut / Maximum Flow

Maximal Flow / Minimum Cut

- **Union (\cup)** of two cuts: the set of edges between the union of all the source-side nodes from each cut and all the other nodes in N
- **Intersection (\cap)** of two cuts: the set of edges between the intersection of the source-side nodes in these cuts and all the other nodes in N

Corollary 3: If $C1$ and $C2$ are minimum cuts, then the union $C1 \cup C2$ and intersection $C1 \cap C2$ are also minimum cuts

COMPSCI 773 12

Ford–Fulkerson Labelling Algorithm

- Proof of the **max-flow / min-cut theorem** provides, under mild restrictions on the capacity function, a simple efficient algorithm for constructing a maximal flow and minimal cut in a network
- **Initialization:** the zero flow
- **Sequence of “labellings”** (*Routine A*), each of which
 - either results in a flow of higher value (*Routine B*) or
 - terminates with the conclusion that the present flow is maximal (to ensure termination: integer capacities!)

COMPSCI 773 13

Informal Algorithm Description

- Main idea of labelling (routine A): use a system of labels to find paths between the source and the sink with **unsaturated** edges
 - Labelling begins from the source (getting the label 0)
 - Let a node x_i be already labelled
 1. A subsequent node x_j is not labelled if the edge (x_i, x_j) is saturated; otherwise ($f(x_i, x_j) < c(x_i, x_j)$) it is labelled with $+i$, that is, x_j^{+i}
 2. A preceding node x_j is not labelled if the flow $f(x_j, x_i) = 0$; otherwise ($f(x_j, x_i) > 0$) it is labelled with $-i$, that is, x_j^{-i}
 - Therefore, the network flow can be increased by increasing flow through edges ending with (+)-nodes and decreasing it through edges ending in (-)-nodes

COMPSCI 773 14

Informal Algorithm Description

- If the sink is labelled, then there exists a **flow augmenting path** between the source and the sink such that all its nodes are labelled with the indices of their preceding nodes
 - Because such a path contains only unsaturated edges, all the flows via its edges can be changed by a value

$$h = \min_{\substack{(x_i, x_u) \in \text{path} \\ (x_i^+, x_u) \in \text{path}}} \{c(x_i, x_u^{+i}) - f(x_i, x_u^{+i}), f(x_i^-, x_u)\} > 0$$
 - The flow via an edge is increased by h if the edge is oriented from s to t (from the source to the sink) and decreased by h otherwise

COMPSCI 773 15

Ford–Fulkerson Labelling Algorithm

- Given an integral flow f , labels are assigned to nodes of the network
 - Nodes can be unlabelled (**UN**), labelled unscanned (**LUN**), and labelled scanned (**LSN**)
 - A label has one of the forms (x^+, ϵ) or (x^-, ϵ) , where $x \in N$ and ϵ is a positive integer or infinity (∞)
- Routine A: Labelling**
 - Initially all nodes are unlabelled (**UN**):
 - The source node is **LUN** ($-, \epsilon(s) = \infty$)
 - Other nodes are **UN**

COMPSCI 773 16

Ford–Fulkerson Labelling Algorithm

Routine A: Labelling (cont.)

- For every **LUN** x having the label $(x^\pm, \epsilon(x))$:
 - (1) Convert all **UN** y “after x ” (i.e. in $A(x)$) such that $f(x, y) < c(x, y)$ into **LUN** with the labels $(x^+, \epsilon(y) = \min[\epsilon(x), c(x, y) - f(x, y)])$, and
 - (2) Convert all **UN** y “before x ” (i.e. in $B(x)$) such that $f(y, x) > 0$ into **LUN** with the labels $(x^-, \epsilon(y) = \min[\epsilon(x), f(y, x)])$
 - (3) Such x is now **LSN**
- If the sink t is **LUN**, go to **Routine B**; otherwise (t is **UN**) - stop

COMPSCI 773 17

Minimum Cut / Maximum Flow

Ford-Fulkerson Labelling Algorithm

Routine B: Flow change (the sink has been labelled $(y^-, \varepsilon(t))$):

- If t is labelled $(y^-, \varepsilon(t))$, replace $f(y,t)$ with $f(y,t) + \varepsilon(t)$
- If t is labelled $(y^-, \varepsilon(t))$, replace $f(t,y)$ with $f(t,y) - \varepsilon(t)$
- In either case,

if node y is labelled $(x^+, \varepsilon(t))$, replace $f(x,y)$ with $f(x,y) + \varepsilon(t)$
 if node y is labelled $(x^-, \varepsilon(t))$, replace $f(y,x)$ with $f(y,x) - \varepsilon(t)$
 and go on to node x

- Stop the flow change when the source s is reached, discard the old labels, and go back to **Routine A**

COMPSCI 773 18

Ford-Fulkerson Labelling Algorithm

Labelling searches for a flow augmenting path from s to t :

If **Routine A** ends and the sink is not labelled, the flow is maximum and the set of edges from UN to L*N nodes is a minimum cut

• **Example:**

Routine A for s :
 Node x : $f(s,x) = c(s,x) = 1$
 Node y : $f(s,y) = 0 < c(s,y) = 3$

Step 1
 $(-\infty)$ $(s^+, 3)$

Step 2
 $(s^+, \min\{\infty, 3 - 0\}) = 3$
 $\min\{\varepsilon(s), c(s,y) - f(s,y)\}$

COMPSCI 773 19

Ford-Fulkerson Labelling Algorithm

Node x : $f(y,x) = 0 < c(y,x) = 1$
 Node t : $f(y,t) = c(y,t) = 1$
 $\min\{\varepsilon(y), c(y,x) - f(y,x)\}$
 $(y^+, \min\{3, 1 - 0\}) = 1$

Step 3

Node x : $f(x,y) = c(x,y) = 1$
 Node t : $f(x,t) = 0 < c(x,t) = 3$

Step 4

COMPSCI 773 20

Ford-Fulkerson Labelling Algorithm

New flow value 2

A flow augmenting path is located by **backtracking** from the sink t according to directions given in labels along which a flow change of $\varepsilon(t) = 1$ can be made

COMPSCI 773 21

Ford-Fulkerson Labelling Algorithm

New flow value 3

New labelling:
non-sink stop

COMPSCI 773 22

Polynomial-Time Max-Flow

Maximum flow in the n -node, m -edge network (graph):

- Ford-Fulkerson (finding augmenting paths; 1956): $O(nm^2)$
- Dinic (shortest augmenting paths in 1 step; 1970): $O(n^2m)$
 - Graphs: dense - $O(n^3)$; sparse - $O(nm \log n)$
- Goldberg-Tarjan (pushing a pre-flow; 1985): $O(nm \log(n^2/m))$
 - Karzanov's **pre-flow**: the flow in and out of nodes may not be equal (the difference at node j is called **the excess at j**)
 - **Aggressive-passive mode**: push as much as possible into the graph, then trim the excess to 0; no flow until the end

COMPSCI 773 23

Minimum Cut / Maximum Flow

Goldberg–Tarjan Push-Label Algorithm

Relabelling eventually makes 0 the excess at each node

- **Excess** $e(x)$ at x is $e(x) = \sum_{y \in B(x)} f(y,x) - \sum_{y \in A(x)} f(x,y)$
 - The node x is **active** if $e(x) > 0$
 - The source and sink are never active
- **Residual capacity** of an edge (x,y) :
 - $r(x,y) = c(x,y) - f(x,y) + f(y,x)$
 - **Residual network** (graph): $RG = \{(x,y) : r(x,y) > 0\}$
- **Distance function** $d: N \rightarrow \mathbb{R}$ for the nodes:
 - (1) $d(t) = 0$;
 - (2) if $(x,y) \in E$ and $c(x,y) > 0$ then $d(x) \leq d(y) + 1$

COMPSCI 773 24

Goldberg–Tarjan Push-Label Algorithm

- **Initialisation:**
 - $d(s) = |N|$ (the number of nodes in a network)
 - $d(t) = 0$
 - $d(x) = 1$ for all $x \neq s, t$
 - $f(s,x) = c(s,x)$ for every edge $(s,x) \in E$
- **Processing** while an active node ($e(x) > 0$) exists:
 - Select an active node x and
 - Try to push more pre-flow towards the sink

COMPSCI 773 25

Goldberg–Tarjan Push-Label Algorithm

- **Processing (cont.):**
 - If $(x,y) \in E$, $d(x) = d(y) + 1$, $r(x,y) > 0$ or $(y,x) \in E$, $d(x) = d(y) + 1$, $r(x,y) > 1$, then push $\min\{e(x), r(x,y)\}$ from x to y and change f accordingly (pushing as much as the excess at the node and the residual capacity of or from the edge (x,y) allows)
 - If nothing can be pushed from x , relabel x by replacing $d(x)$ with $\min\{d(y) + 1 : (x,y) \in A(x) \text{ and } r(x,y) > 0\}$
- Once processing is finished, the pre-flow is a **max flow**

COMPSCI 773 26

Goldberg–Tarjan Push-Label Algorithm

Iter.	x	y	s,x	s,y	y,x	x,t	y,t	Active
1	1	1	4	6	0	0	0	x,y
2	1	1	4	6	0	4	5	y
3	1	2	4	6	0	4	5	y
4	1	2	4	6	1	4	5	x
5	3	2	4	6	1	4	5	x
6	3	2	4	6	0	4	5	y
7	3	4	4	6	0	4	5	y
8	3	4	4	6	1	4	5	x
9	5	4	4	6	1	4	5	x
10	5	4	3	6	1	4	5	

COMPSCI 773 27

D.M.Greig, B.T.Porteous, A.H.Sehulst: Exact Maximum A Posteriori Estimation for Binary Images. *Journal of the Royal Statistical Society, Ser. B, Vol.51 (2), pp.271-279, 1989*

Energy Minimization via Graph Cuts

- [D. M. Greig e.a., 1989]: Denoising binary images
 - Noisy signals $\mathbf{y}=(y_i; i = 1, \dots, n)$ are conditionally independent, given a noiseless image $\mathbf{x}=(x_i; i = 1, \dots, n)$:

$$P(\mathbf{y} | \mathbf{x}) = \prod_{i=1}^n p(y_i | x_i) = \prod_{i=1}^n p(y_i | 10)^{1-x_i} p(y_i | 11)^{x_i}$$

$$= \prod_{i=1}^n p(y_i | 10) \prod_{i=1}^n \left(\frac{p(y_i | 11)}{p(y_i | 10)} \right)^{x_i}$$
 - Prior image model:** a 2nd-order Markov random field (MRF):

$$P(\mathbf{x}) \propto \exp\left(\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \beta_{ij} [x_i x_j + (1-x_i)(1-x_j)]\right)$$
 where $\beta_{ii} = 0$; $\beta_{ij} \neq 0$ (**neighbours** if the strict inequality); i.e. the model of more probable identical neighbours ($x_i = x_j$)

COMPSCI 773 28

Negative Likelihood as Energy

- Log-likelihood (apart from an additive constant):

$$L(\mathbf{x} | \mathbf{y}) \propto \ln P(\mathbf{y} | \mathbf{x}) P(\mathbf{x})$$



$$= \sum_{i=1}^n \lambda_i x_i + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \beta_{ij} [x_i x_j + (1-x_i)(1-x_j)]$$
 where $\lambda_i = \log\{p(y_i | 1) / p(y_i | 0)\}$
- Bayesian **maximum a priori** (MAP) image estimate:

$$\mathbf{x}^* = \arg \max_{\mathbf{x}} L(\mathbf{x} | \mathbf{y}) = \arg \min_{\mathbf{x}} [-L(\mathbf{x} | \mathbf{y})]$$
 - Pixel-wise simulated annealing, ICM do not approach the MAP even after hundreds or thousands of iterations...







COMPSCI 773 29




Minimum Cut / Maximum Flow

D.M.Greig, B.T.Porteous, A.H.Seheult: Exact Maximum A Posteriori Estimation for Binary Images. *Journal of the Royal Statistical Society, Ser. B, Vol.51 (2), pp.271-279, 1989*

Noiseless image x  Noisy image y 

MRF prior: $\beta_{ij} = \beta$

MAP: $\beta_{ij} = 0.3$  Stochastic annealing  ICM Iterated conditional modes  MAP: $\beta_{ij} = 0.7$  Stochastic annealing  ICM Iterated conditional modes 

MAP: $\beta_{ij} = 1.1$  Stochastic annealing  ICM Iterated conditional modes 

COMPSI 773 30

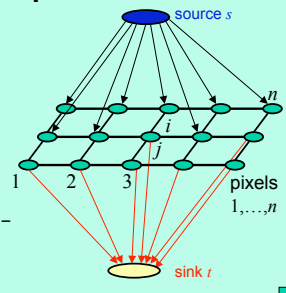
Network Representation

T-links (terminal links):

- Directed edge (s, i) with the capacity $c_{si} = \lambda_i$ if $\lambda_i > 0$
- Directed edge (i, t) with the capacity $c_{it} = -\lambda_i$ if $\lambda_i \leq 0$

N-links (neighbouring links):

- Undirected edge (i, j) between two internal nodes – neighbours i and j with the capacity $c_{ij} = \beta_{ij} > 0$



COMPSI 773 31

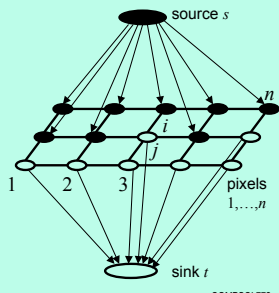
Network Representation

$\lambda_i = \log \{p(y_i|1)\} / p(y_i|0)$:

- $\lambda_i > 0$ for $y_i = 1$ ●
- and
- $\lambda_i < 0$ for $y_i = 0$ ○

Example:

$$p(y_i|x_i) = \begin{cases} 0.8 & y_i = x_i \\ 0.2 & y_i \neq x_i \end{cases} \Rightarrow$$

$$\lambda_i = \begin{cases} \log_2 4 = 2 & y_i = 1 \\ -\log_2 4 = -2 & y_i = 0 \end{cases}$$


COMPSI 773 32

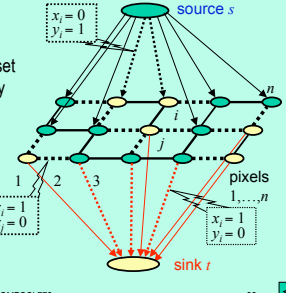
Energy Minimisation via Graph Cut

$B = \{s\} \cup \{i: x_i = 1\}$ and

$W = \{i: x_i = 0\} \cup \{t\}$ – a 2-set partition of the nodes for any binary image x

Cut – a set of edges (i, j) such that $i \in B$ and $j \in W$

Capacity of the cut:

$$C(\mathbf{x}) = \sum_{\substack{(i,j) \in E \\ i \in B, j \in W}} c_{ij}$$


COMPSI 773 33

Energy Minimisation via Graph Cut

- Capacity of the cut:

$$C(\mathbf{x}) = \sum_{i=1}^n x_i \max\{0, -\lambda_i\} + \sum_{i=1}^n (1-x_i) \max\{0, \lambda_i\} + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \beta_{ij} (x_i - x_j)^2$$
- Differs from $[-L(\mathbf{x}|y)]$ by a term that is independent of \mathbf{x}
- Maximizing this likelihood is equivalent to minimizing the capacity of the cut, i.e. to finding **the minimum cut**, or **the maximum flow** through the network


COMPSI 773 34

Energy Minimisation via Graph Cut

- [Greig e. a., 1989] Accelerated Ford-Fulkerson algorithm:
 - Partitioning the image into $2^K \times 2^K$ connected sub-images
 - Calculate the MAP estimate for each sub-image separately
 - Amalgamate the sub-images to form a set of $2^{K-1} \times 2^{K-1}$ larger sub-images
 - Form the MAP estimate for each of them
 - Continue until the MAP estimate of the complete image
- Simulated annealing: does not necessarily produce a good approximation of an MAP estimate and becomes bogged down by local maxima resulting in under-smooth solutions

COMPSI 773 35

Minimum Cut / Maximum Flow




Energies Minimised via Graph Cuts

Theorem [Friedman,Drineas,2005; generally, it is a folklore of the combinatorial optimisation: Papadimitriou, Steiglitz, 1986]:

Let $E(x_1, \dots, x_n) = \sum_{i,j} \beta_{ij} x_i x_j + L$
 where $x_i \in \{0, 1\}$ and L is linear in x_i plus constants (i.e. $L = \sum_i \lambda_i x_i + c$)
 Then E can be minimised via graph cut techniques if and only if $\beta_{ij} \leq 0$ for all $i, j \in \{1, 2, \dots, n\}$

COMPSCI 773 36



Energies Minimised via Graph Cuts


Proof of "if" part ("only if": see [Papadimitriou, Steiglitz, 1986]):

- Energy is rewritten as $E = \sum_{i,j=1, \dots, n} \alpha_{ij} x_i (1 - x_j) + \Lambda$ where $\alpha_{ij} = -\beta_{ij}$ and the linear term Λ is altered
- Minimal E over the binary $x_i \Rightarrow$ a min cut in a complete graph with n nodes and edge weights $w_{ij} = \alpha_{ij}$
 - The cut separates the nodes with $x_i = 0$ from those with $x_j = 1$ (because only $x_i = 1$ and $x_j = 0$ adds α_{ij} to the energy)

Polynomial-time min cut if and only if $w_{ij} \geq 0 \Rightarrow \beta_{ij} \leq 0$

- For the altered linear term $\Lambda = \sum_i \gamma_i x_i + \sigma$
 - An edge (s, t) with the weight $w_{st} = \gamma_s$ if $\gamma_s \geq 0$ and an edge (i, t) with $w_{it} = |\gamma_t|$ if $\gamma_t < 0$; therefore, **all weights are non-negative**

COMPSCI 773 37



Energies Minimised via Graph Cuts

Energy terms depending on signals and pairs of signals (class F^2):

$$E(x_1, \dots, x_n) = \sum_i E_i(x_i) + \sum_{i,j} E_{ij}(x_i, x_j); x_i \in \{0, 1\}$$


$$E_{ij}(x_i, x_j) = E_{ij}^{00}(1 - x_i)(1 - x_j) + E_{ij}^{01}(1 - x_i)x_j + E_{ij}^{10}x_i(1 - x_j) + E_{ij}^{11}x_i x_j$$

$$E(x_1, \dots, x_n) = \sum_{i,j} (E_{ij}^{00} + E_{ij}^{11} - E_{ij}^{01} - E_{ij}^{10}) x_i x_j + L$$

Such energy function can be minimised via graph cuts if and only if:

$$\underbrace{E_{ij}^{00} + E_{ij}^{11} - E_{ij}^{01} - E_{ij}^{10}}_{\text{regularity condition}} \leq 0 \quad \forall i, j$$

COMPSCI 773 38



Large Moves via Min-Cut/Max-Flow


[Boykov,Veksler,Zabih,2001] Approximate energy minimisation by replacing pixel-wise optimising moves with large moves

- Convergence to a solution being provably within a known factor of the global energy minimum
- Energy function minimised w.r.t. a labelling $\mathbf{x} = (x_1, \dots, x_n)$:

$$E(x_1, \dots, x_n) = \sum_{i=1}^n V_i(x_i) + \sum_{(i,j) \in \mathbf{E}} V_{ij}(x_i, x_j); x_i \in \mathbf{L}; i = 1, \dots, n$$

where $\mathbf{L} = \{1, \dots, L\}$ - an arbitrary finite set of labels
 $\mathbf{N} \subset \{1, \dots, n\}^2$ - a set of neighbouring (interacting) pixel pairs
 $V_i: \mathbf{L} \rightarrow \mathbf{R}$ - a pixel-wise potential function (energy of labels)
 $V_{ij}: \mathbf{L}^2 \rightarrow \mathbf{R}$ - a pair-wise potential function (energy of label pairs)


COMPSCI 773 39



Conditions and Optimal Moves

- Arbitrary pixel-wise energies V_i
- Semimetric or metric pair-wise energies** V_{ij}
 - Semimetric:** $\forall \alpha, \beta \in \mathbf{L} V_{ij}(\alpha, \alpha) = 0; V_{ij}(\alpha, \beta) = V_{ij}(\beta, \alpha) \geq 0$
 - Metric:** also the triangle inequality $V_{ij}(\alpha, \beta) \leq V_{ij}(\alpha, \gamma) + V_{ij}(\gamma, \beta)$
- Each labelling \mathbf{x} partitions the pixel set $R = \{1, \dots, n\}$ into L subsets $R_\lambda = \{i \mid i \in R; x_i = \lambda \in \mathbf{L}\}$
 - Conditionally optimal large moves change each partition $\mathbf{P} = \{R_\lambda; \lambda \in \mathbf{L}\}$ to approach a certain vicinity of the global minimum of the partition energy
 - α - β -swap (with the semimetric V_{ij})
 - α -expansion (with the metric V_{ij})

COMPSCI 773 40



α, β -swap and α -expansion

α, β -swap for an arbitrary pair of labels $\alpha, \beta \in \mathbf{L}$ is a move from a partition \mathbf{P} for a current labelling \mathbf{x} to a new partition \mathbf{P}' for a new labelling \mathbf{x}' such that $R_\lambda = R'_\lambda$ for any label $\lambda \neq \alpha, \beta$

- Only the labels α and β in their current region $R_{\alpha\beta} = R_\alpha \cup R_\beta$ whereas all other labels in $R \neq R_{\alpha\beta}$ remain fixed.
- In the general case, after an α - β -swap some pixels change their labels from α to β and some others -- from β to α .

α -expansion of an arbitrary label α is a move from a partition \mathbf{P} for a current labelling \mathbf{x} to a new labelling \mathbf{x}' such that $R_\alpha \subset R'_\alpha$ and $R \setminus R'_\alpha = \bigcup_{\lambda \in \mathbf{L}, \lambda \neq \alpha} R'_\lambda \subset R \setminus R_\alpha = \bigcup_{\lambda \in \mathbf{L}, \lambda \neq \alpha} R_\lambda$

- After this move any subset of pixels can change their labels to α

COMPSCI 773 41

Energy Minimisation Algorithms

Swap algorithm for semimetric interaction potentials

- Initialization:** An arbitrary labelling \mathbf{x}
- Iterative minimization:** For every pair of labels $(\alpha, \beta) \in L^2$ taken in a fixed or random order:
 - Find $\mathbf{x}^* = \arg \min_{\text{one } \alpha\text{-}\beta\text{-swap of } \mathbf{x}} E(\mathbf{x})$ with a min-cut/max-flow technique
 - If $E(\mathbf{x}^*) < E(\mathbf{x})$, then accept the lower-energy labelling: $\mathbf{x} \leftarrow \mathbf{x}^*$
- Stopping rule:** If a new labelling has been accepted for at least one pair of labels at Step 2.1, continue the minimisation process by returning to Step 2. Otherwise terminate the process and output the final labelling \mathbf{x}

COMPSCI 773 42

Energy Minimisation Algorithms

Expansion algorithm for metric interaction potentials

- Initialization:** An arbitrary labelling \mathbf{x}
- Iterative minimization:** For every label $\alpha \in L$ taken in a fixed or random order:
 - Find $\mathbf{x}^* = \arg \min_{\text{one } \alpha\text{-expansion of } \mathbf{x}} E(\mathbf{x})$ with a min-cut/max-flow technique
 - If $E(\mathbf{x}^*) < E(\mathbf{x})$, then accept the lower-energy labelling: $\mathbf{x} \leftarrow \mathbf{x}^*$
- Stopping rule:** If a new labelling has been accepted for at least one label at Step 2.1, continue the minimisation process by returning to Step 2. Otherwise terminate the process and output the final labelling \mathbf{x}

COMPSCI 773 43

Optimal Move: Swap Algorithm

N_i – the set of neighbours of the node i

Graph $G_{\alpha\beta} = [N_{\alpha\beta}; E_{\alpha\beta}]$ for a set of pixels with the labels α and β

- $N_{\alpha\beta}$ - two terminals, α and β , and all pixels in $R_{\alpha\beta}$
- Each pixel $i \in R_{\alpha\beta}$ is connected to the terminals by edges (t -links) $t_{\alpha,i}$ and $t_{\beta,i}$:

$$\text{weight } t_{\alpha,i} \Leftarrow V_i(\alpha) + \sum_{j \in N_i \cap R_{\alpha\beta}} V_{ij}(\alpha, x_j)$$

$$\text{weight } t_{\beta,i} \Leftarrow V_i(\beta) + \sum_{j \in N_i \cap R_{\alpha\beta}} V_{ij}(\beta, x_j)$$
- Each neighbour pair $(i,j) \in R_{\alpha\beta}$ is connected by an edge (n -link) $e_{i,j}$:

$$\text{weight } e_{i,j} \Leftarrow V_{ij}(\alpha, \beta); (i,j) \in E_{\alpha\beta} \cap R_{\alpha\beta}^2$$

COMPSCI 773 44

Optimal Move: Swap Algorithm

A cut C on $G_{\alpha\beta}$ must contain exactly one t -link for any pixel $i \in R_{\alpha\beta}$. Otherwise: either there would be a path between the terminals if both the links are included, or a proper subset of C would become a cut if both the links are excluded.

Therefore, any cut C provides a natural labelling \mathbf{x}_C : every pixel $i \in R_{\alpha\beta}$ is labelled with α or β if the cut C separates i from the terminal α or β , respectively, and the other pixels keep their initial labels:

$$\forall i \in R \quad x_{C,i} = \begin{cases} \alpha & \text{if } i \in R_{\alpha\beta} \text{ and } t_{\alpha,i} \in C \\ \beta & \text{if } i \in R_{\alpha\beta} \text{ and } t_{\beta,i} \in C \\ x_i & \text{if } i \notin R_{\alpha\beta} \end{cases}$$

COMPSCI 773 45

Optimal Move: Swap Algorithm

Each labelling \mathbf{x}_C corresponding to a cut C on $G_{\alpha\beta}$ is one α - β -swap from the initial \mathbf{x}

- Any n -link $e_{i,j}$ is included in a cut C only if the pixels i and j are linked to different terminals under the cut

Theorem (BVZ.2001): The capacity $\alpha(C)$ of the cut C is the energy function $E(\mathbf{x}_C)$ plus a constant

Corollary (BVZ.2001): The lowest energy labelling within a single α - β -swap move from a current labelling \mathbf{x} corresponds to the minimum cut labelling

COMPSCI 773 46

Optimal Move: Expansion Algorithm

Graph $G_\alpha = [N_\alpha; E_\alpha]$ for a set of pixels with the labels α and $\bar{\alpha}$

- N_α - two terminals, α and $\bar{\alpha}$, all pixels $i \in R$, and auxiliary nodes $a_{i,j}$ for each pair (i,j) of the nodes with the labels $x_i \neq x_j$
- Edge weights:

Edge	Weight	Condition
$t_{\alpha,i}$	∞	$i \in R_\alpha$
$t_{\bar{\alpha},i}$	$V_i(x_i)$	$i \notin R_\alpha$
$t_{\alpha,i}$	$V_i(\alpha)$	$i \in R_\alpha$
$e_{\alpha,a_{i,j}}$	$V_{ij}(x_i, x_j)$	$(i,j) \in N; x_i \neq x_j$
$e_{\bar{\alpha},a_{i,j}}$	$V_{ij}(x_i, \alpha)$	$(i,j) \in N; x_i \neq x_j$
$e_{i,j}$	$V_{ij}(\alpha, \alpha)$	$(i,j) \in N; x_i = x_j$

COMPSCI 773 47

Minimum Cut / Maximum Flow

Optimal Move: Expansion Algorithm

Any cut C on G_α must include exactly one t -link for any $i \in R$

This provides a natural labelling:

$$\forall i \in R \quad x_{C,i} = \begin{cases} \alpha & \text{if } t_{\alpha,i} \in C \\ x_i & \text{if } t_{\bar{\alpha},i} \in C \end{cases}$$

An n -link $e_{i,j}$ is in C if $i, j \in R$ are connected to different terminals

The edge triplet $E_{i,j}$ for $i, j \in R$ such that $x_i \neq x_j$ has the unique minimum cut due to the metric properties of the potentials:

- if $t_{\alpha,i}, t_{\alpha,j} \in C$ then $C \cap E_{i,j} = \emptyset$
- if $t_{\alpha,i}, t_{\bar{\alpha},j} \in C$ then $C \cap E_{i,j} = t_{\bar{\alpha},i}$
- if $t_{\bar{\alpha},i}, t_{\alpha,j} \in C$ then $C \cap E_{i,j} = e_{i,j}$
- if $t_{\bar{\alpha},i}, t_{\bar{\alpha},j} \in C$ then $C \cap E_{i,j} = t_{\alpha,j}$

$x_{C,j} = \alpha, x_{C,i} = \alpha \quad x_{C,j} = x_i, x_{C,i} = x_j \quad x_{C,j} = x_i, x_{C,i} = \alpha$

COMPSCI 773 48

Optimality of Large Moves

- No proven optimality properties for the swap move algorithm
- Local minimum within a fixed factor of the global minimum for the expansion move algorithm
- Theorem**[Boykov, Veksler, Zabih, 2001]:
Let x^* and x° be the labellings for a local energy when the expansion moves are allowed and the global energy minimum, respectively. Then $E(x^*) \leq 2\gamma E(x^\circ)$ where

$$\gamma = \max_{(i,j) \in N} \left(\frac{\max_{\alpha \neq \beta \in L} V_{i,j}(\alpha, \beta)}{\min_{\alpha \neq \beta \in L} V_{i,j}(\alpha, \beta)} \right)$$

COMPSCI 773 49

Pictures from: <http://bj.middlebury.edu/~schar/stereo/web/>

Stereo pair True DM

Disparity map (DM)

Graph cut stereo Dynamic programming stereo SSD stereo (window 21 x 21)

Signed errors

COMPSCI 773 50