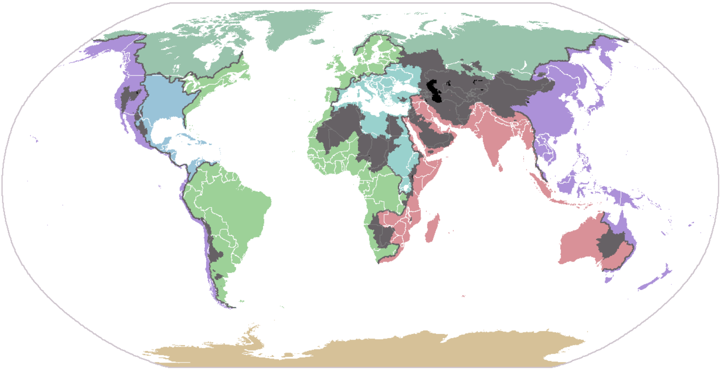
Watershed cuts

Jean Cousty, Gilles Bertrand,
Laurent Najman and Michel Couprie

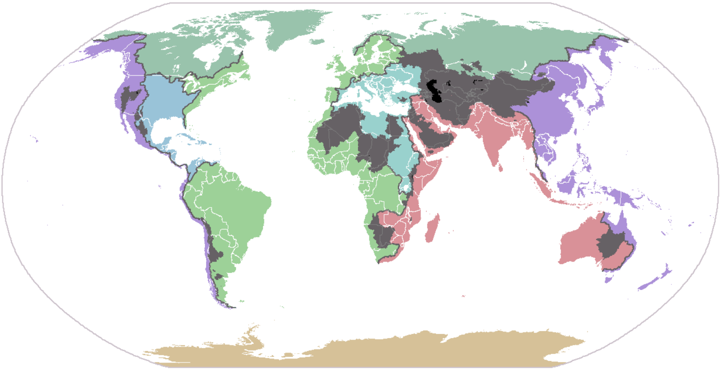
Université Paris-Est,
LABINFO-IGM, UMR CNRS 8049, A2SI-ESIEE, France

ISMM 2007
October 13th

Context



Context



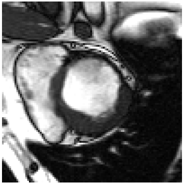
- For topographic purposes, the watershed has been studied since the 19th century (Maxwell, Jordan, . . .)

Context

- One hundred years later (1978), it was introduced by Digabel and Lantuéjoul for image segmentation

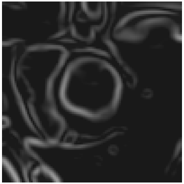
Context

- One hundred years later (1978), it was introduced by Digabel and Lantuéjoul for image segmentation



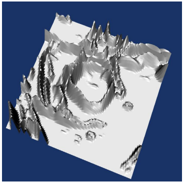
Context

- One hundred years later (1978), it was introduced by Digabel and Lantuéjoul for image segmentation



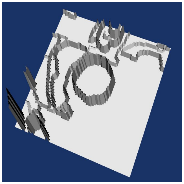
Context

- One hundred years later (1978), it was introduced by Digabel and Lantuéjoul for image segmentation



Context

- One hundred years later (1978), it was introduced by Digabel and Lantuéjoul for image segmentation



Context

- One hundred years later (1978), it was introduced by Digabel and Lantuéjoul for image segmentation

Context

- One hundred years later (1978), it was introduced by Digabel and Lantuéjoul for image segmentation

Hypothesis

- Most existing approaches consider a grayscale image as a vertex-weighted graph

Problem

Problem

- *Watersheds in edge-weighted graphs?*

Problem

Problem

- *Watersheds in edge-weighted graphs?*
- *What mathematical properties?*

Problem

Problem

- *Watersheds in edge-weighted graphs?*
- *What mathematical properties?*
- *How to efficiently compute them?*

1. Watershed cuts: definition and consistency
2. Relative minimum spanning forests: watershed optimality
3. Algorithm

Edge-weighted graph

- Let $G = (V, E)$ be a graph.
- Let F be a map from E to $\mathbb{Z}$.

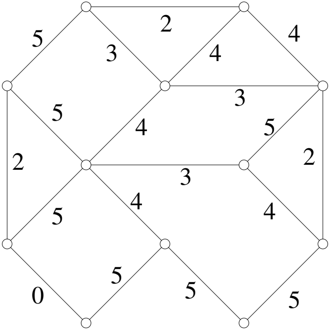
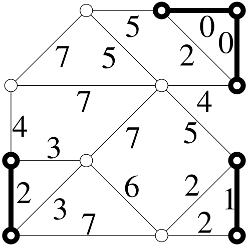


Image and edge-weighted graph

For applications to image analysis

- V is the set of *pixels*
- E corresponds to an *adjacency relation* on V, (*e.g.*, 4- or 8-adjacency in 2D)
- The altitude of u, an edge between two pixels x and y, represents the *dissimilarity between x and y*
    - $F(u) = |I(x) - I(y)|$.

Regional minima

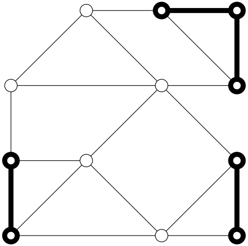


Definition

A subgraph X of G is a minimum of F (at altitude k) *if:*

- *X is connected; and*
- *k is the altitude of any edge of X; and*
- *the altitude of any edge adjacent to X is strictly greater than k*

Extension



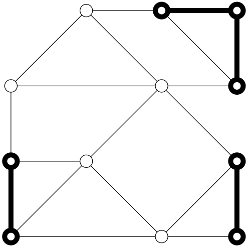
a subgraph X

Definition (from Def. 12, (Ber05))

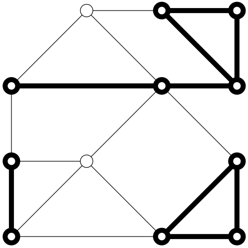
Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an extension of X (in G) if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Extension



a subgraph X



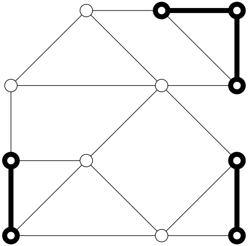
an *extension* Y of X

Definition (from Def. 12, (Ber05))

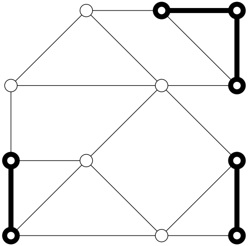
Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an* extension of X (in G) *if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Extension



a subgraph X



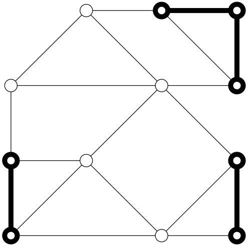
an *extension* Y of X

Definition (from Def. 12, (Ber05))

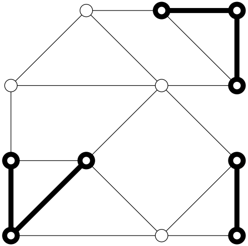
Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an* extension of X (in G) *if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Extension



a subgraph X



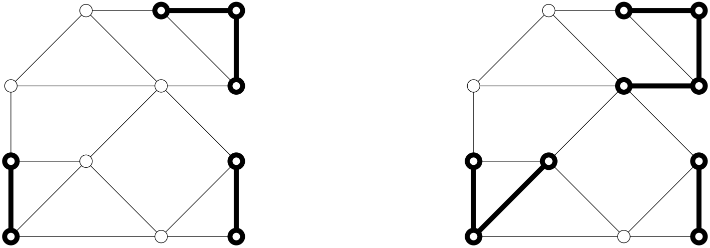
an *extension* Y of X

Definition (from Def. 12, (Ber05))

Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an* extension of X (in G) *if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Extension



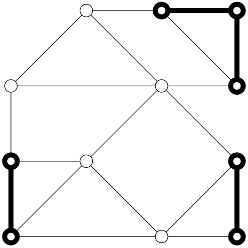
a subgraph X — an *extension* Y of X

Definition (from Def. 12, (Ber05))

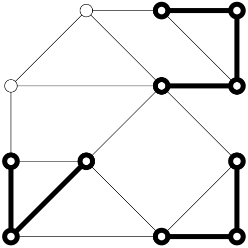
Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an* extension of X (in G) *if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Extension



a subgraph X



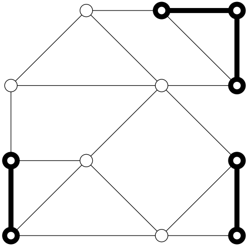
an *extension* Y of X

Definition (from Def. 12, (Ber05))

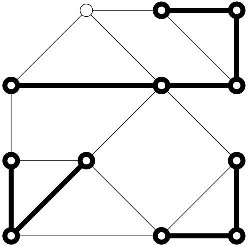
Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an extension of X (in G) if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Extension



a subgraph X



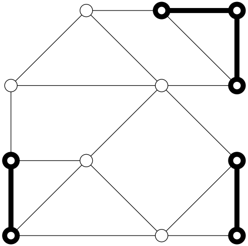
an *extension* Y of X

Definition (from Def. 12, (Ber05))

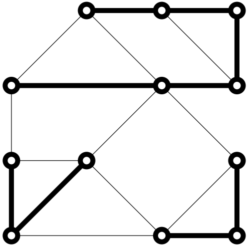
Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an* extension of X (in G) *if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Extension



a subgraph X



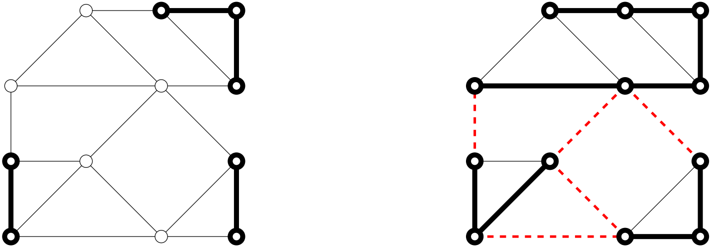
an *extension* Y of X

Definition (from Def. 12, (Ber05))

Let X and Y be two non-empty subgraphs of G.

- *We say that Y is an extension of X (in G) if $X \subseteq Y$ and if any component of Y contains exactly one component of X.*

Graph cut



a subgraph X — a *(graph) cut* S for X

Definition (Graph cut)

Let X be a subgraph of G and $S \subseteq E$, a set of edges.

- *We say that S is a (graph) cut for X if $\overline{S}$ is an extension of X and if S is minimal for this property*

Watershed cut

- *The church of Sorbier*

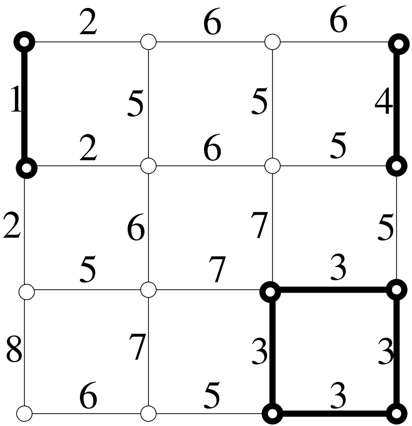
  (a topographic intuition)

Definition (drop of water principle)

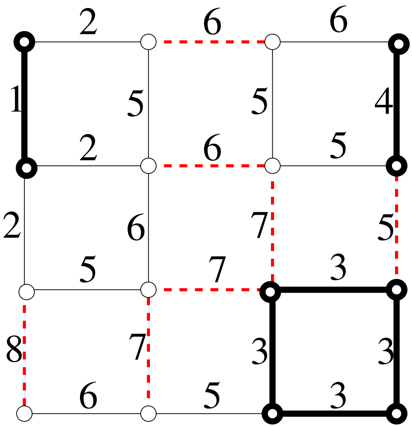
The set $S \subseteq E$ is a watershed cut of F if $\overline{S}$ is an extension of $M(F)$ and if for any $u = \{x_0, y_0\} \in S$, there exist $\langle x_0, \ldots, x_n \rangle$ and $\langle y_0, \ldots, y_m \rangle$, two descending paths in $\overline{S}$ such that:

1. *x_n and y_m are vertices of two distinct minima of F; and*
2. *$F(u) \geq F(\{x_0, x_1\})$ if $n > 0$ and $F(u) \geq F(\{y_0, y_1\})$ if $m > 0$*

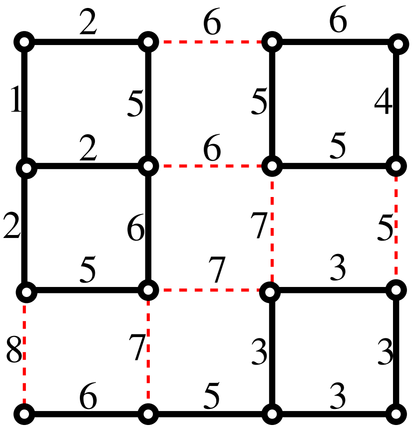
Watershed cut: example



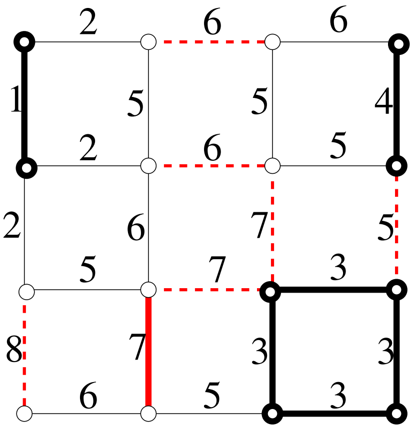
Watershed cut: example



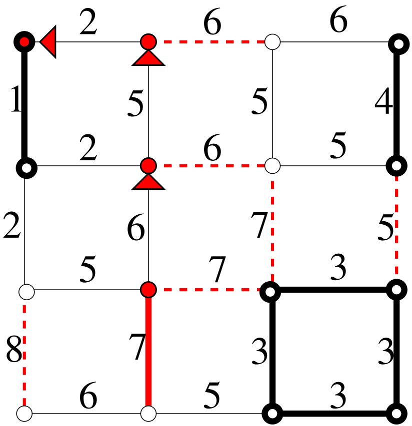
Watershed cut: example



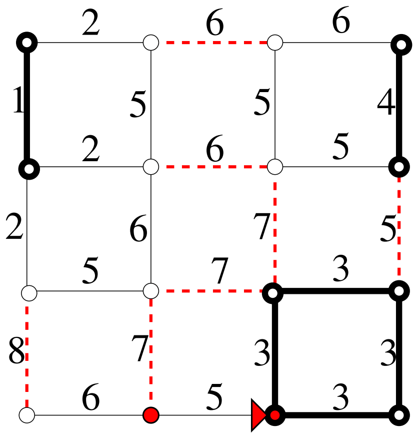
Watershed cut: example



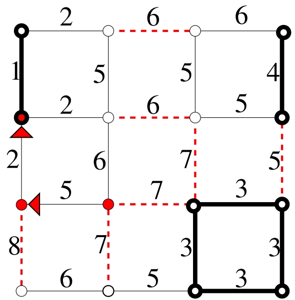
Watershed cut: example



Watershed cut: example



Steepest descent



Definition

Let $\pi = \langle x_0, \ldots, x_l \rangle$ be a path in G.

- *The path π is a* path with steepest descent for *F if:*
  $\forall i \in [1, l]$, $F(\{x_{i-1}, x_i\}) = \min_{\{x_{i-1}, y\} \in E} F(\{x_{i-1}, y\})$

Catchment basins by a steepest descent property

Definition

Let S be a cut for $M(F)$, the minima of F.
We say that S is a basin cut of F *if, from each point of V to $M(F)$, there exists, in the graph induced by $\overline{S}$, a path with steepest descent for F.*

Catchment basins by a steepest descent property

Theorem (consistency)

An edge-set $S \subseteq E$ is a basin cut of F if and only if *S is a watershed cut of F.*

Relative forests

In 1994, F. Meyer shows the links between flooding from markers and minimum spanning forest

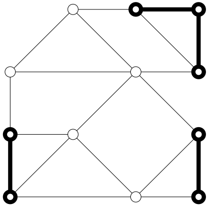
Relative forests

In 1994, F. Meyer shows the links between flooding from markers and minimum spanning forest

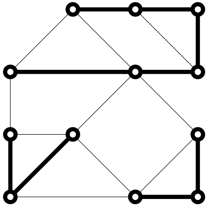
Problem

- *What about watershed cuts ?*

Relative forest



a subgraph X



a *forest Y relative to X*

Definition

Let X and Y be two non-empty subgraphs of G.
We say that Y is a forest relative to X *if:*

1. *Y is an extension of X; and*
2. *any cycle of Y is also a cycle of X*

Minimum spanning forest

- The *weight of a forest* Y is the sum of its edge weights *i.e.*, $\sum_{u \in E(Y)} F(u)$.

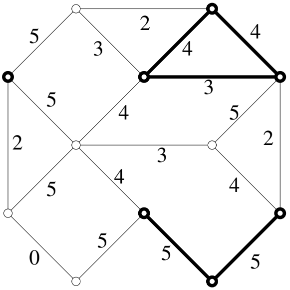
Minimum spanning forest

- The *weight of a forest* Y is the sum of its edge weights *i.e.*, $\sum_{u \in E(Y)} F(u)$.

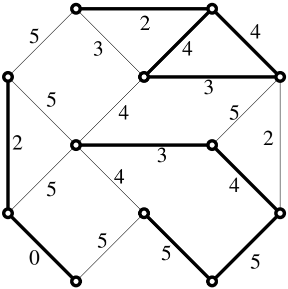
Definition

We say that Y is a minimum spanning forest (MSF) relative to X *if* Y *is a spanning forest relative to* X *and if the weight of* Y *is less than or equal to the weight of any other spanning forest relative to* X.

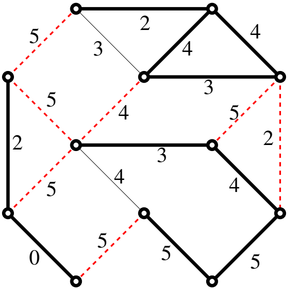
Minimum spanning forest: example



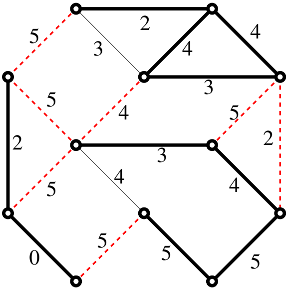
Minimum spanning forest: example



Minimum spanning forest: example

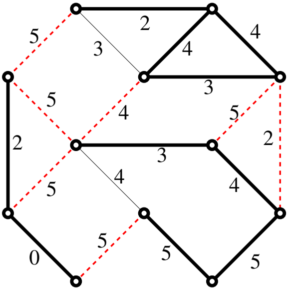


Minimum spanning forest: example



- If Y is a MSF relative to X, there exists a unique cut S for Y and this cut is also a cut for X;

Minimum spanning forest: example



- If Y is a MSF relative to X, there exists a unique cut S for Y and this cut is also a cut for X;
- In this case, we say that S is a *MSF cut for X*.

Watershed optimality

Theorem

An edge-set $S \subseteq E$ is a MSF cut for the minima of F if and only if *S is a watershed cut of F.*

Minimum spanning tree

- Computing a MSF $\Leftrightarrow$ computing a minimum spanning tree

Minimum spanning tree

- Computing a MSF $\Leftrightarrow$ computing a minimum spanning tree
- Best algorithm [CHAZEL00]: quasi-linear time

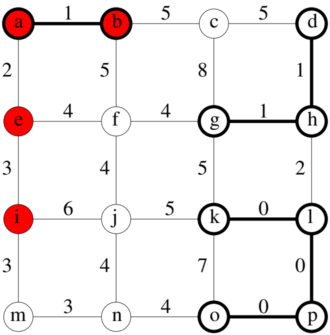
Minimum spanning tree

- Computing a MSF $\Leftrightarrow$ computing a minimum spanning tree
- Best algorithm [CHAZEL00]: quasi-linear time

Problem

Can we reach a better complexity for computing watershed cuts?

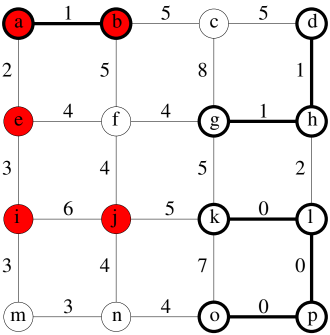
Streams



Definition

We say that a vertex-set $L \subseteq V$ is a stream *if, for any two points x and y of L, there exists, in L, either a path from x to y or from y to x, with steepest descent for F*

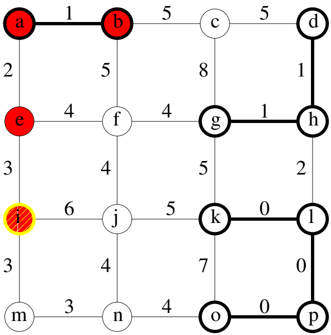
Streams



Definition

We say that a vertex-set $L \subseteq V$ is a stream *if, for any two points x and y of L, there exists, in L, either a path from x to y or from y to x, with steepest descent for F*

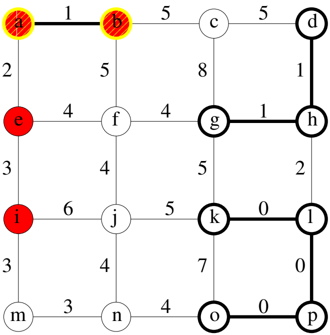
Streams



Definition

We say that a point x of a stream L is a top of L if for any point y in L, there exists, from x to y, a path in L, which is a path with steepest descent for F

Streams



Definition

We say that a point x of a stream L is a bottom of L *if for any point y in L, there exists, from y to x, a path in L, which is a path with steepest descent for F*

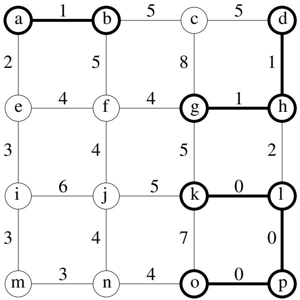
Streams

Problem

How to extract a stream ?

Streams concatenation

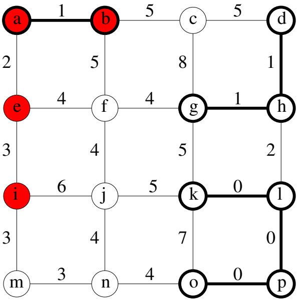
Stream concatenation



Definition

Let L_1 and L_2 be two disjoint streams ($L_1 \cap L_2 = \emptyset$).
We say that L_1 is under L_2 *(written $L_1 \prec L_2$), if there exist a top x of L_1, a bottom y of L_2, such that x and y are adjacent and $\langle x, y \rangle$ is a path with steepest descent for F*

Stream concatenation

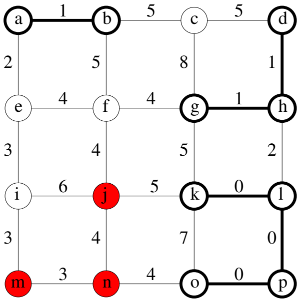


A stream L_1

Definition

Let L_1 and L_2 be two disjoint streams ($L_1 \cap L_2 = \emptyset$).
We say that L_1 is under L_2 *(written $L_1 \prec L_2$), if there exist a top x of L_1, a bottom y of L_2, such that x and y are adjacent and $\langle x, y \rangle$ is a path with steepest descent for F*

Stream concatenation

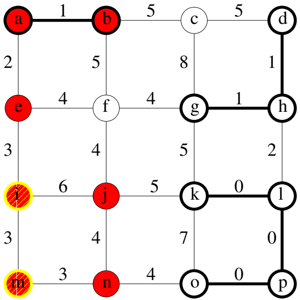


A stream L_2

Definition

Let L_1 and L_2 be two disjoint streams ($L_1 \cap L_2 = \emptyset$).
We say that L_1 is under L_2 *(written $L_1 \prec L_2$), if there exist a top x of L_1, a bottom y of L_2, such that x and y are adjacent and $\langle x, y \rangle$ is a path with steepest descent for F*

Stream concatenation

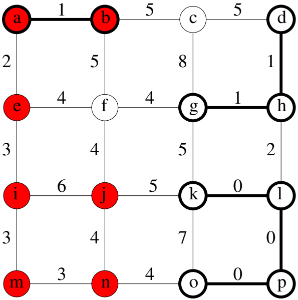


$L_1 \prec L_2$

Definition

Let L_1 and L_2 be two disjoint streams ($L_1 \cap L_2 = \emptyset$).
We say that L_1 is under L_2 *(written $L_1 \prec L_2$), if there exist a top x of L_1, a bottom y of L_2, such that x and y are adjacent and $\langle x, y \rangle$ is a path with steepest descent for F*

Stream concatenation

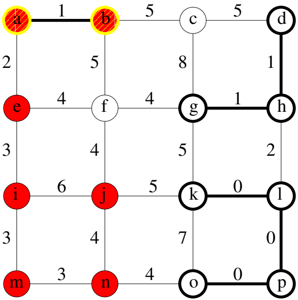


$L = L_1 \cup L_2$ is also a stream

Property

Let L_1 and L_2 be two disjoint streams ($L_1 \cap L_2 = \emptyset$).
If $L_1 \prec L_2$, then $L_1 \cup L_2$ is a stream

Stream concatenation

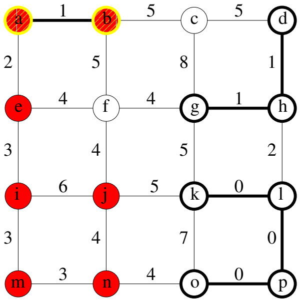


The bottoms of L

Definition

A stream L is an $\prec$-stream if there is no stream under L

Stream concatenation



L is also an $\prec$-stream

Definition

A stream L is an $\prec$-stream *if there is no stream under L*

Streams and minima

- The proposed algorithm is based on $\prec$-streams extraction

Streams and minima

Property

A stream L is an $\prec$-stream if and only if *L contains the vertex set of a minimum of F*

Problem

From the streams of F, how can we partition the vertex set of G?

Flow familly

- Let $\mathcal{L} = \{L_1, \ldots, L_n\}$ be a set of n $\prec$-streams. We say that $\mathcal{L}$ is a *flow family* if:
  - $\cup\{L_i \mid i \in \{1, \ldots, n\}\} = V$; and
  - for any two distinct L_1 and L_2 in $\mathcal{L}$, if $L_1 \cap L_2 \neq \emptyset$, then $L_1 \cap L_2$ is the vertex set of a minimum of F.

Flow familly

- Let $\mathcal{L} = \{L_1, \ldots, L_n\}$ be a set of n $\prec$-streams. We say that $\mathcal{L}$ is a *flow family* if:
  - $\cup\{L_i \mid i \in \{1, \ldots, n\}\} = V$; and
  - for any two distinct L_1 and L_2 in $\mathcal{L}$, if $L_1 \cap L_2 \neq \emptyset$, then $L_1 \cap L_2$ is the vertex set of a minimum of F.
- We remark that a flow family induces a unique graph cut for the minima of F.

Streams and watershed cut

Theorem

An edge set $S \subseteq E$ is a watershed cut of F if and only if *S is a cut induced by a flow family.*

Linear-time algorithm

Algorithm 1: LPE par flux

Data: (V, E, F): an edge-weighted graph;

Result: ψ: a flow mapping of F.

foreach $x \in V$ **do** $\psi(x) := NO\_LABEL$;
$nb\_labs := 0$; /* the number of minima already found */
foreach $x \in V$ *such that* $\psi(x) = NO\_LABEL$ **do**
    $[L, lab] := \texttt{stream}(V, E, F, \psi, x)$;
    if $lab = \emptyset$ **then** /* L is an $\prec$-stream */
        $nb\_labs{+}{+}$;
        foreach $y \in L$ **do** $\psi(y) := nb\_labs$;
    else
        foreach $y \in L$ **do** $\psi(y) := lab$;

Linear-time algorithm

Function `Stream(` V, E, F, ψ, x `)`

Data: (V, E, F): an edge-weighted graph; ψ: a labeling of V; x: a point in V.

Result: $[L, lab]$ where L is a stream such that x is a top of L, and *lab* is either a label of an $\prec$-stream under L or $\emptyset$.

$L := \{x\}$;
$L' := \{x\}$; /* the set of non-explored bottoms of L */
while *there exists* $y \in L'$ **do**
    $L' := L' \setminus \{y\}$;
    $breadth\_first := TRUE$;
    while *(breadth_first) and (there exists* $\{y, z\} \in E$ *such that* $z \notin L$ *and* $F(\{y, z\}) = F(y)$*)* **do**
        if $\psi(z) \neq NO\_LABEL$ **then**
            /* there is an $\prec$-stream under L already labelled */
            return $[L, \psi(z)]$;
        else if $F(z) < F(y)$ **then**
            $L := L \cup \{z\}$; /* z is now the only bottom of L */
            $L' := \{z\}$; /* hence, switch to depth-first exploration */
            $breadth\_first := FALSE$;
        else
            $L := L \cup \{z\}$; /* $F(z) = F(y)$, thus z is also a bottom of L */
            $L' := L' \cup \{z\}$; /* continue breadth-first exploration */
return $[L, \emptyset]$;

Algorithm

Result

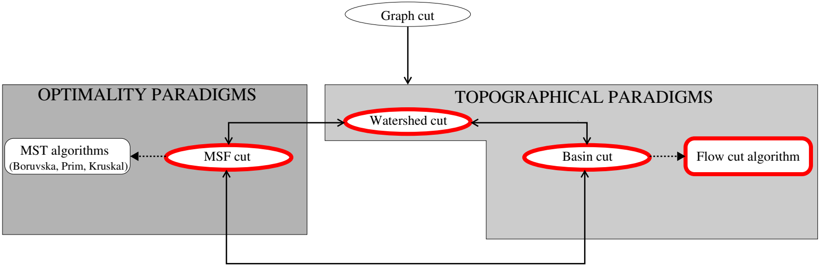
- *Stream Algorithm runs in linear time whatever the range of the input map*
  - *No need to sort*
  - *No need to use a hierarchical queue*

Algorithm

Result

- *Stream Algorithm runs in linear time whatever the range of the input map*
  - *No need to sort*
  - *No need to use a hierarchical queue*
- *Furthermore, Stream Algorithm does not need to compute the minima as a pre-processing step.*

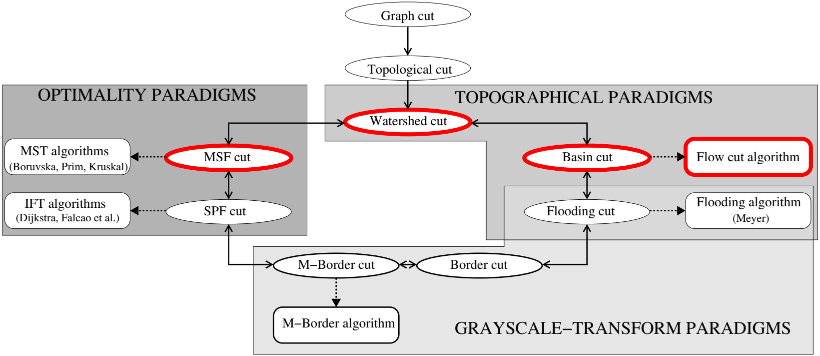
Conclusion


Graph cut
OPTIMALITY PARADIGMS
TOPOGRAPHICAL PARADIGMS
Watershed cut
MST algorithms
(Boruvska, Prim, Kruskal)
MSF cut
Basin cut
Flow cut algorithm

Conclusion

- In fact, there is more to say on watershed cuts . . .

Conclusion


Graph cut
Topological cut
OPTIMALITY PARADIGMS
TOPOGRAPHICAL PARADIGMS
Watershed cut
MST algorithms
(Boruvska, Prim, Kruskal)
MSF cut
Basin cut
Flow cut algorithm
IFT algorithms
(Dijkstra, Falcao et al.)
SPF cut
Flooding cut
Flooding algorithm
(Meyer)
M−Border cut
Border cut
M−Border algorithm
GRAYSCALE−TRANSFORM PARADIGMS

Perspectives

- Hierarchical segmentations
  - Saliency of watershed contours
  - Incremental MSF

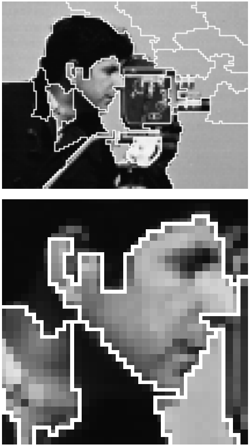
Perspectives

- Hierarchical segmentations
  - Saliency of watershed contours
  - Incremental MSF
- Topological properties of watershed cuts (simplicial, cubical complexes)

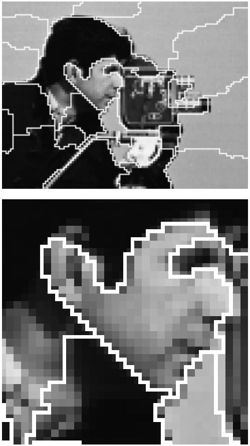
Perspectives

- Hierarchical segmentations
  - Saliency of watershed contours
  - Incremental MSF
- Topological properties of watershed cuts (simplicial, cubical complexes)
- Minimum spanning tree by watersheds

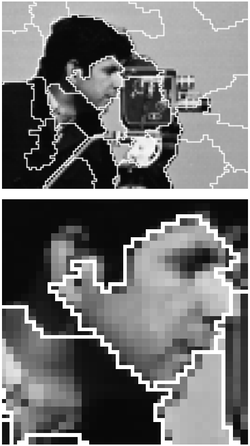
To finish : illustration



Edges

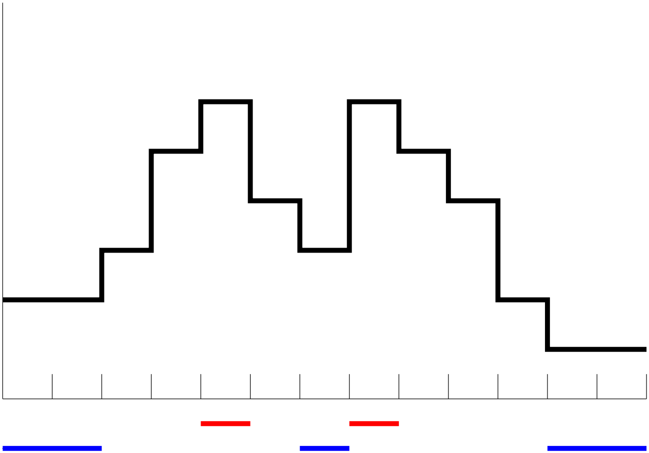


Vertices+Deriche

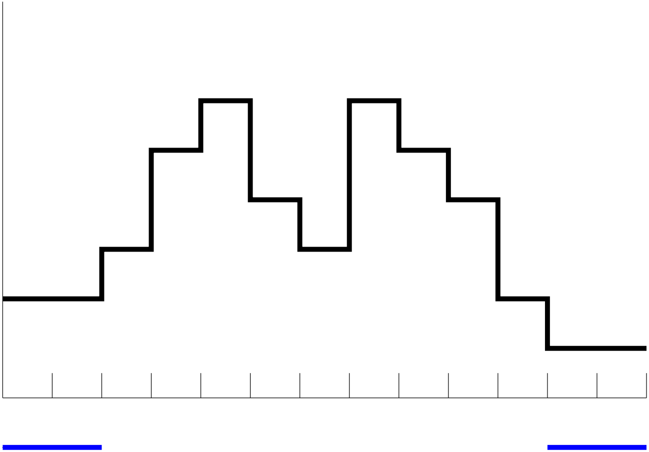


Vertices+morpho. grad.

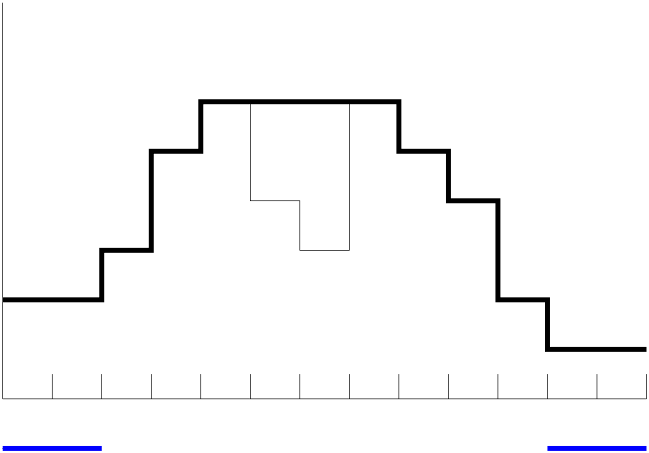
The plateau problem



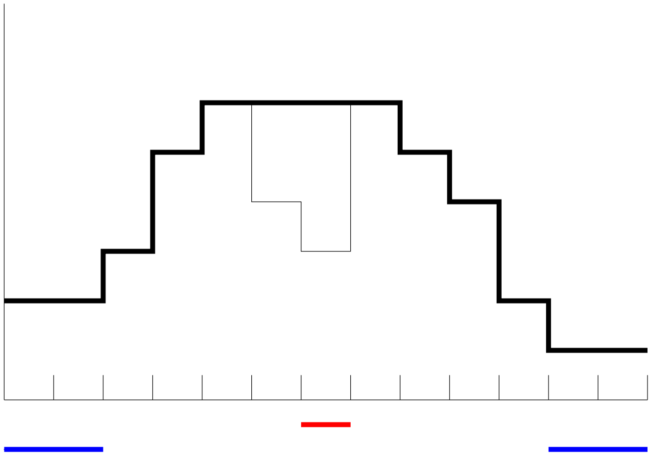
The plateau problem



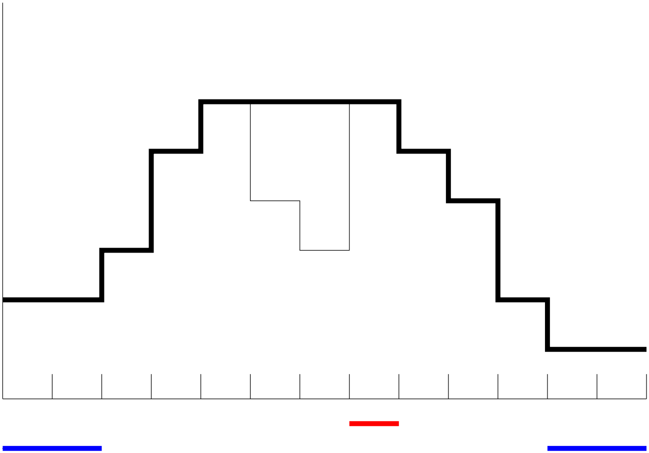
The plateau problem



The plateau problem



The plateau problem



The plateau problem