

# Watershed cuts and Combinatorial Optimization

## LPE-Coupure et Optimisation combinatoire

Laurent Najman<sup>1</sup>,  
Jean Cousty<sup>1</sup>, Gilles Bertrand<sup>1</sup>, Michel Couprie<sup>1</sup>  
Camille Couprie<sup>1</sup>, Leo Grady<sup>2</sup>, Hugues Talbot<sup>1</sup>

<sup>1</sup>LIGM, UPE-MLV

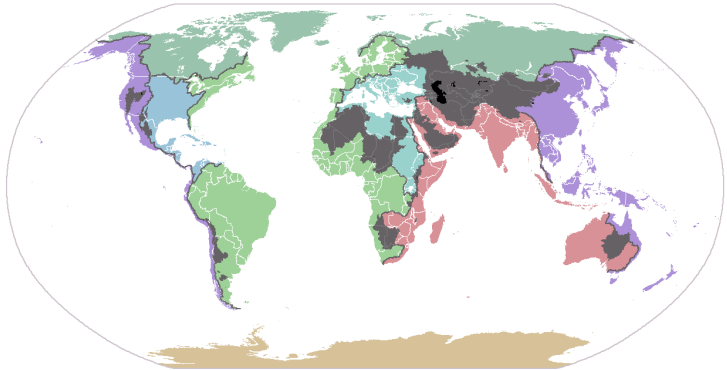
<sup>2</sup>Siemens Corporate Research

Master Course  
10 février 2014

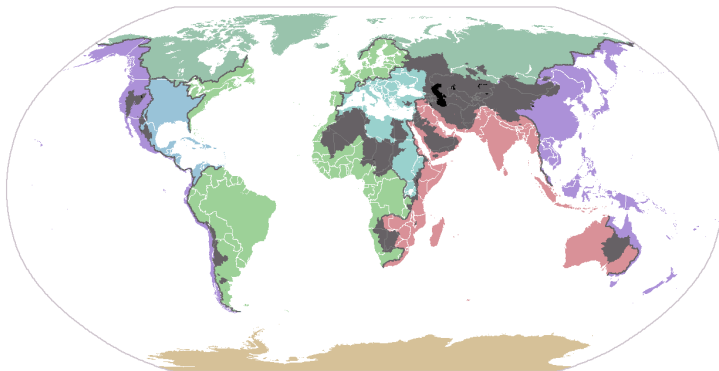
# Outline

- Introduction
- Watershed cuts
  - Definition and consistency
  - Relative minimum spanning forests : watershed optimality
- Power Watersheds
  - A unifying framework for combinatorial optimization
  - The powerwatershed algorithm
  - Qualitative and quantitative comparison
- Conclusion and perspectives

# Context



## Context



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

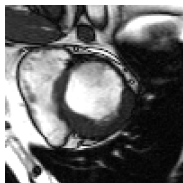


## Context

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

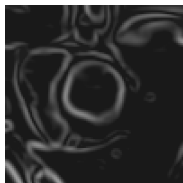
# Context

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



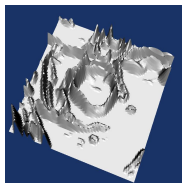
# Context

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



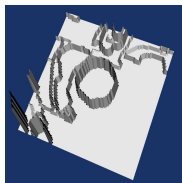
# Context

- 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

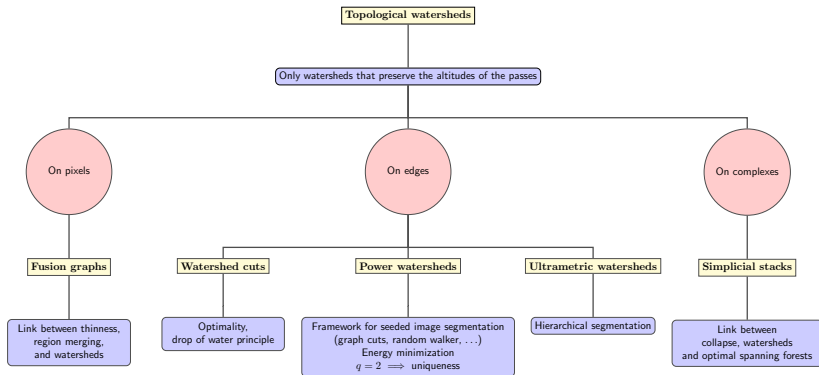


# Context

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



# The family of discrete watersheds



# In this talk

## Problem

- *Watersheds in edge-weighted graphs ?*



# In this talk

## Problem

- *Watersheds in edge-weighted graphs?*
- *Mathematical properties?*

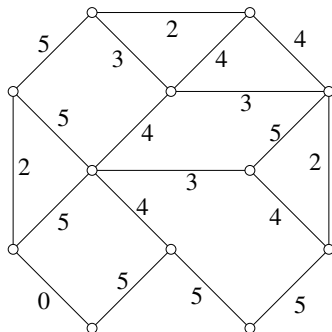
# In this talk

## Problem

- *Watersheds in edge-weighted graphs ?*
- *Mathematical properties ?*
- *Use of watersheds for optimization ?*

# Edge-weighted graph

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

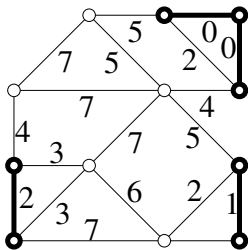


# Image and edge-weighted graph

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

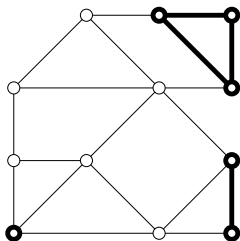


### Definition

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



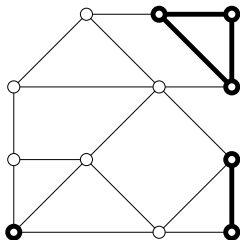
a subgraph  $X$

Definition (from Def. 12, (Ber05))

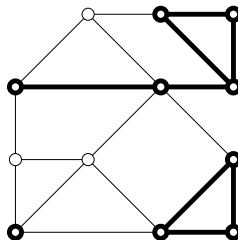
Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ .

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  $X$



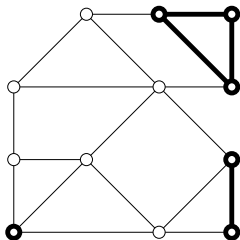
an extension  $Y$  of  $X$

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  $X$

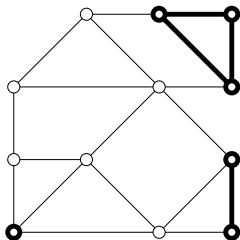
Definition (from Def. 12, (Ber05))

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

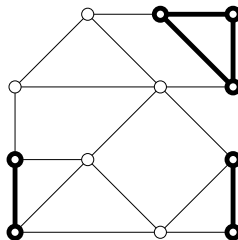
We say that  $Y$  is an **extension of  $X$  (in  $G$ )** if  $X \subseteq Y$  and if any component of  $Y$  contains exactly one component of  $X$ .



# Extension



a subgraph  $X$

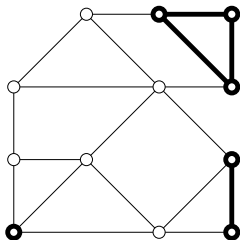


Definition (from Def. 12, (Ber05))

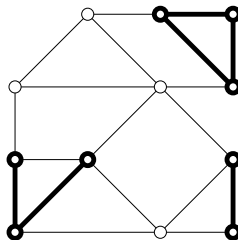
Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ .

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  $X$

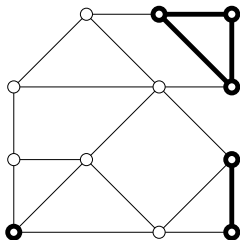


Definition (from Def. 12, (Ber05))

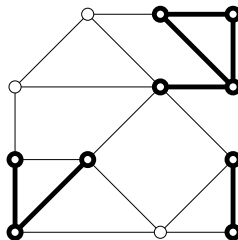
Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ .

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  $X$

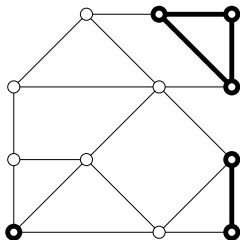


Definition (from Def. 12, (Ber05))

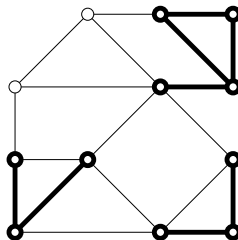
Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ .

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  $X$

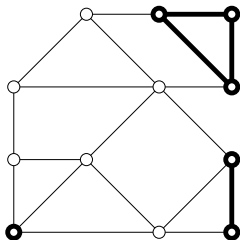


Definition (from Def. 12, (Ber05))

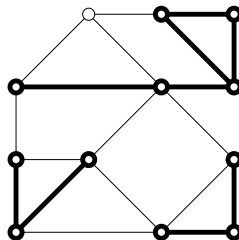
Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ .

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  $X$

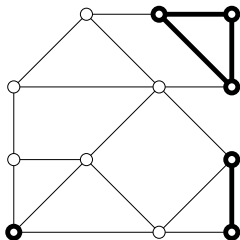


Definition (from Def. 12, (Ber05))

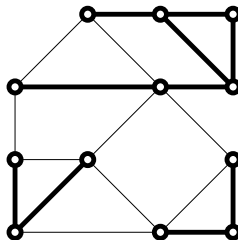
Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ .

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  $X$

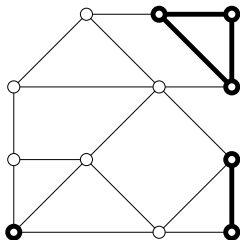


Definition (from Def. 12, (Ber05))

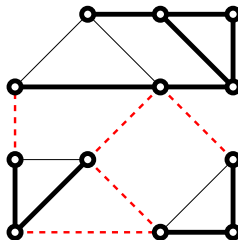
Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ .

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  $X$



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

## Definition

Let  $X$  be a subgraph of  $G$  and let  $S \subseteq E$  be an edge-set.

- 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, i.e., if  $T \subseteq S$  and  $\overline{T}$  is an extension of  $X$ , then we have  $T = S$ .



# Watershed : intuitive idea

*The church of Sorbier*



# Watershed cut

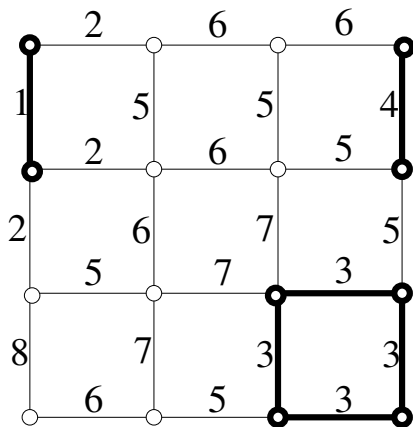
## Definition (drop of water principle)

Let  $S \subseteq E$  be an edge-set.

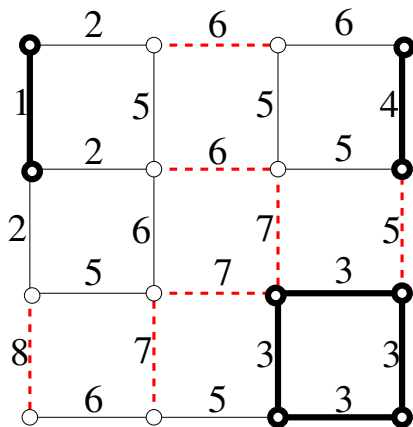
We say that  $S$  is a watershed cut of  $F$  if  $\bar{S}$  is an extension of  $M(F)$  and if for any  $u = \{x_0, y_0\} \in S$ , there exist  $\pi_1 = \langle x_0, \dots, x_n \rangle$  and  $\pi_2 = \langle y_0, \dots, y_m \rangle$  which are two descending paths in  $\bar{S}$  such that :

- $x_n$  and  $y_m$  are vertices of two distinct minima of  $F$  ; and
- $F(u) \geq F(\{x_0, x_1\})$  (resp.  $F(u) \geq F(\{y_0, y_1\})$ ), whenever  $\pi_1$  (resp.  $\pi_2$ ) is not trivial.

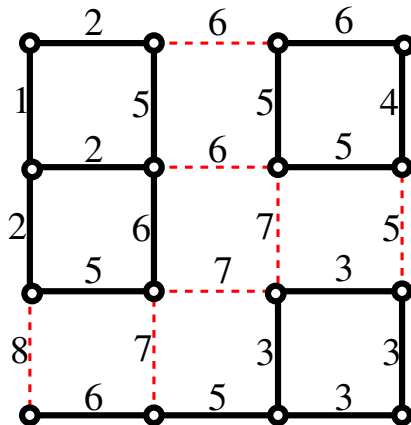
# Watershed cut : example



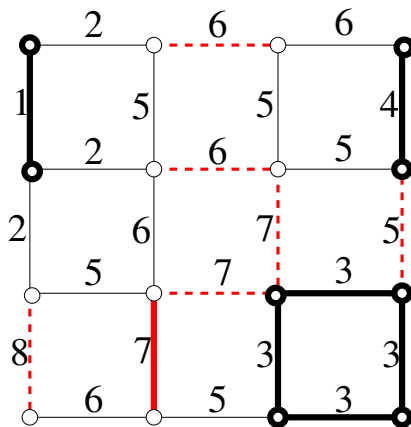
## Watershed cut : example



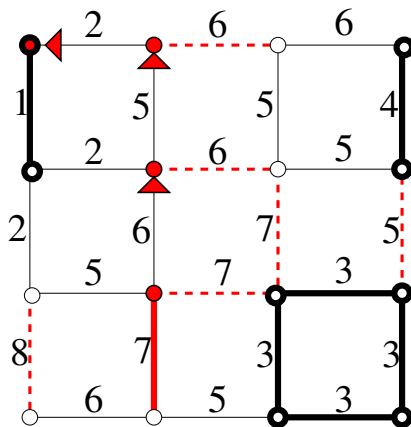
# Watershed cut : example



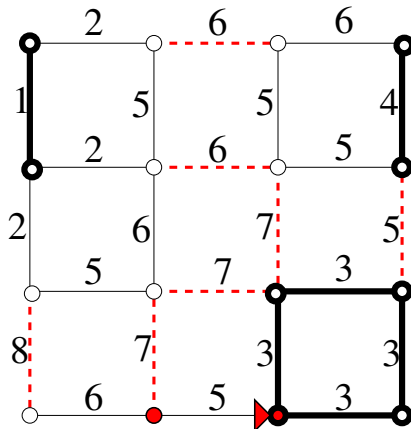
## Watershed cut : example



## Watershed cut : example



## Watershed cut : example





# Catchment basins by a steepest descent property

- The *altitude* of a vertex  $x$  of  $G$ , denoted by  $F(x)$ , is the minimal altitude of an edge which contains  $x$  :
  - $F(x) = \min\{F(u) \mid u \in E, x \in u\}$

# Catchment basins by a steepest descent property

- The *altitude* of a vertex  $x$  of  $G$ , denoted by  $F(x)$ , is the minimal altitude of an edge which contains  $x$  :
  - $F(x) = \min\{F(u) \mid u \in E, x \in u\}$
- Let  $\pi = \langle x_0, \dots, x_l \rangle$  be a path in  $G$ . The path  $\pi$  is *a path with steepest descent for  $F$*  if, for any  $i \in [1, l]$ ,  $F(\{x_{i-1}, x_i\}) = F(x_{i-1})$ .

# 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  $F$ .*

# 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  $F$ .*

## Contribution

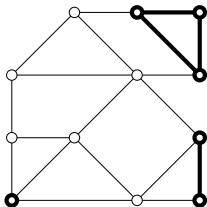
- As far as we know, in the literature about discrete watersheds, no similar property has ever been proved.

# Relative forest

- Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ . We say that  $Y$  is a *forest relative to  $X$*  if :
  - $Y$  is an extension of  $X$  ; and
  - for any extension  $Z \subseteq Y$  of  $X$ , we have  $Z = Y$  whenever  $V(Z) = V(Y)$ .

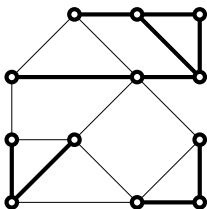
# Relative forest

- Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ . We say that  $Y$  is a *forest relative to  $X$*  if :
  - $Y$  is an extension of  $X$  ; and
  - for any extension  $Z \subseteq Y$  of  $X$ , we have  $Z = Y$  whenever  $V(Z) = V(Y)$ .



# Relative forest

- Let  $X$  and  $Y$  be two non-empty subgraphs of  $G$ . We say that  $Y$  is a *forest relative to  $X$*  if :
  - $Y$  is an extension of  $X$  ; and
  - for any extension  $Z \subseteq Y$  of  $X$ , we have  $Z = Y$  whenever  $V(Z) = V(Y)$ .





# Minimum spanning forest

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

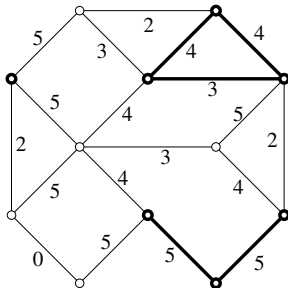
# Minimum spanning forest

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

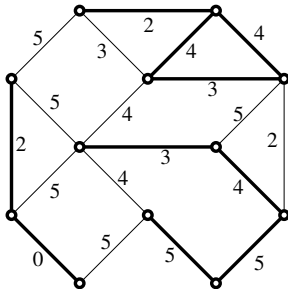
## Definition

*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  $X$ .*

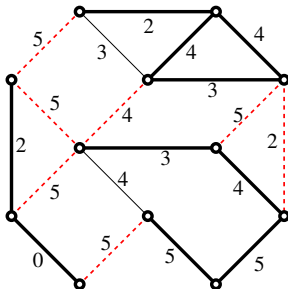
# Minimum spanning forest : example



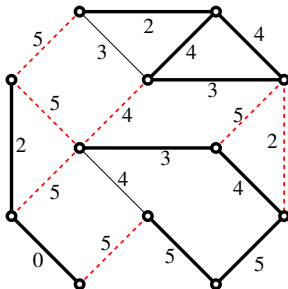
# Minimum spanning forest : example



# 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  $X$  ;



# 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  $F$ .*



# 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  $F$ .*

## Contribution

- As far as we know, this is the first result which establishes watershed optimality.

# 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 ?*

# A linear-time algorithm for watershed cuts

## Result

*We propose the Stream Algorithm.*

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

# A linear-time algorithm for watershed cuts

## Result

*We propose the Stream Algorithm.*

- *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.*

# A linear-time algorithm for watershed cuts

## Result

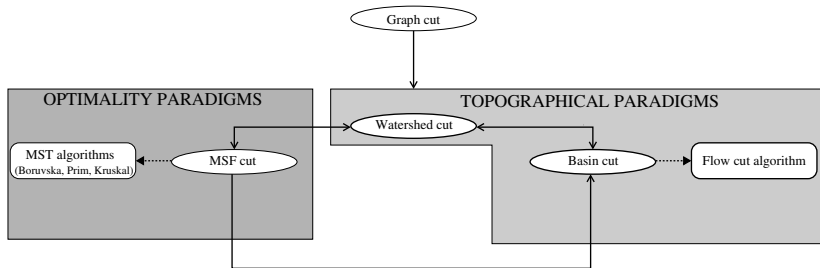
*We propose the Stream Algorithm.*

- *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.*

## Contribution

To the best of our knowledge, this is the first watershed algorithm satisfying such properties

# Conclusion on watershed cuts

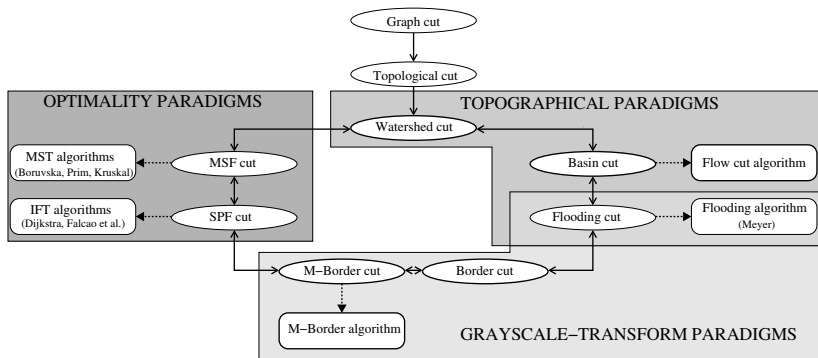




# Conclusion on watershed cuts

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

# Conclusion on watershed cuts



# Watershed and optimization : intuitive idea

# Watershed and optimization : intuitive idea

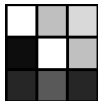
*The church of Sorbier*



## Edge-weighted graph, revisited

- An image seen as a graph  $G = (V, E)$

Image  $3 \times 3 \rightarrow$  Weighted graph  $3 \times 3$



# Edge-weighted graph, revisited

- An image seen as a graph  $G = (V, E)$

Image  $3 \times 3 \rightarrow$  Weighted graph  $3 \times 3$

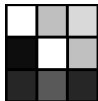


- Edges are weighted by a *similarity* measure  
 i.e. *inversely* proportional to the image gradient
  - $w_{ij} = F(\{x_i, x_j\}) = F(u) = \exp(-\beta(l(x_i) - l(x_j))^2)$ .

# Edge-weighted graph, revisited

- An image seen as a graph  $G = (V, E)$

Image  $3 \times 3 \rightarrow$  Weighted graph  $3 \times 3$



- Edges are weighted by a *similarity* measure  
 i.e. *inversely* proportional to the image gradient
  - $w_{ij} = F(\{x_i, x_j\}) = F(u) = \exp(-\beta(l(x_i) - l(x_j))^2)$ .



- Seed specification :

## New framework for image segmentation

- Given  $\left\{ \begin{array}{l} \text{two real positive numbers } p \text{ and } q \\ \text{seeds for the background } B, \\ \text{seeds for the foreground } F, \end{array} \right.$



## New framework for image segmentation

- Given  $\left\{ \begin{array}{l} \text{two real positive numbers } p \text{ and } q \\ \text{seeds for the background } B, \\ \text{seeds for the foreground } F, \end{array} \right.$
- Compute  $x$  verifying

$$\min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$$

Such that  $x(F) = 1, x(B) = 0$ .

## New framework for image segmentation

- Given  $\left\{ \begin{array}{l} \text{two real positive numbers } p \text{ and } q \\ \text{seeds for the background } B, \\ \text{seeds for the foreground } F, \end{array} \right.$
- Compute  $x$  verifying

$$\min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$$

Such that  $x(F) = 1, x(B) = 0$ .

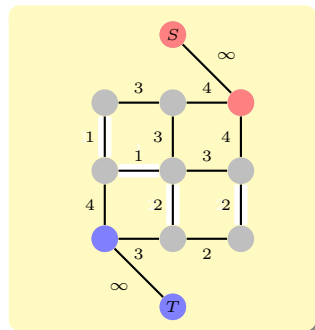
- Result : segmentation  $s$  defined  $\forall i$  by  $s_i = \begin{cases} 1 & \text{si } x_i \geq \frac{1}{2}, \\ 0 & \text{si } x_i < \frac{1}{2}. \end{cases}$

# Graph Cuts

- Problem : compute  $x$

$$x = \arg \min \sum_{e_{ij} \in E} w_{ij}^{p=1} |x_i - x_j|^{q=1}$$

- Min cut / Max flow duality
- Max Flow algorithm

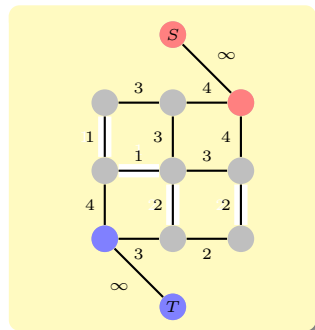


# Graph Cuts

- Problem : compute  $x$

$$x = \arg \min \sum_{e_{ij} \in E} w_{ij} |x_i - x_j|$$

- Min cut / Max flow duality
- Max Flow algorithm

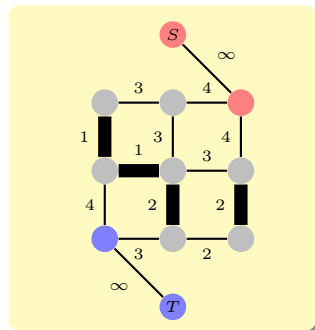


# Graph Cuts

- Problem : compute  $x$

$$x = \arg \min \sum_{e_{ij} \in E} w_{ij} |x_i - x_j|$$

- Min cut / Max flow duality
- Max Flow algorithm

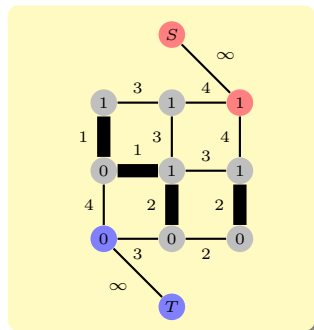


# Graph Cuts

- Problem : compute  $x$

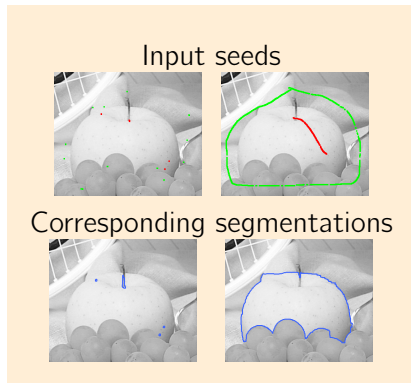
$$x = \arg \min \sum_{e_{ij} \in E} w_{ij} |x_i - x_j|$$

- Min cut / Max flow duality
- Max Flow algorithm



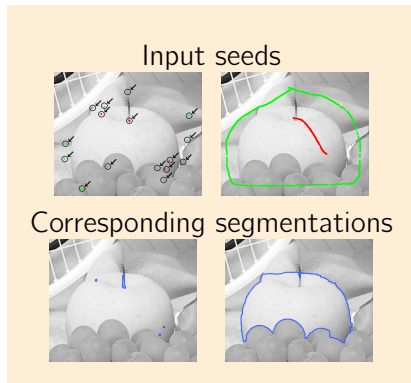
## Graph Cuts : example

- favor small boundaries
- robust to uncentered seed placement



## Graph Cuts : example

- favor small boundaries
- robust to uncentered seed placement

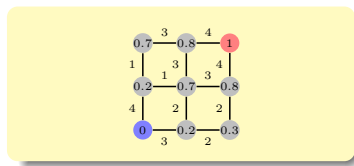




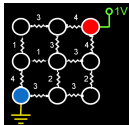
# Random Walker

- Combinatorial version of the Dirichlet problem

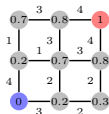
$$x = \arg \min \sum_{e_{ij} \in E} w_{ij} (x_i - x_j)^2 \quad \leftarrow \quad u = \arg \min \int_{\Omega} |\nabla u|^2 d\Omega$$



- Potentials analogy



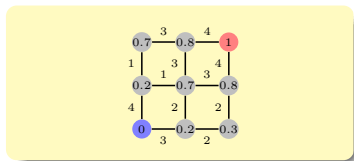
- Random walker analogy



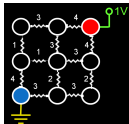
# Random Walker

- Combinatorial version of the Dirichlet problem

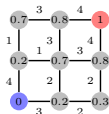
$$x = \arg \min \sum_{e_{ij} \in E} w_{ij}^{p=1} (x_i - x_j)^{q=2} \quad \leftarrow \quad u = \arg \min \int_{\Omega} |\nabla u|^2 d\Omega$$



- Potentials analogy



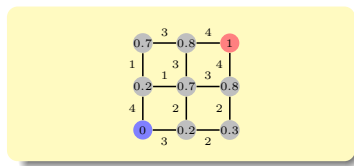
- Random walker analogy



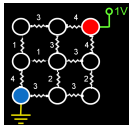
# Random Walker

- Combinatorial version of the Dirichlet problem

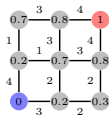
$$x = \arg \min \sum_{e_{ij} \in E} w_{ij} (x_i - x_j)^2 \quad \leftarrow \quad u = \arg \min \int_{\Omega} |\nabla u|^2 d\Omega$$



- Potentials analogy



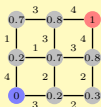
- Random walker analogy



# Random Walker

- Combinatorial version of the Dirichlet problem

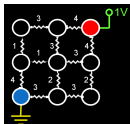
$$x = \arg \min \sum_{e_{ij} \in E} w_{ij} (x_i - x_j)^2 \quad \leftarrow \quad u = \arg \min \int_{\Omega} |\nabla u|^2 d\Omega$$



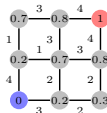
Strictly convex problem

$\Rightarrow$  unique optimal solution  $x^*$

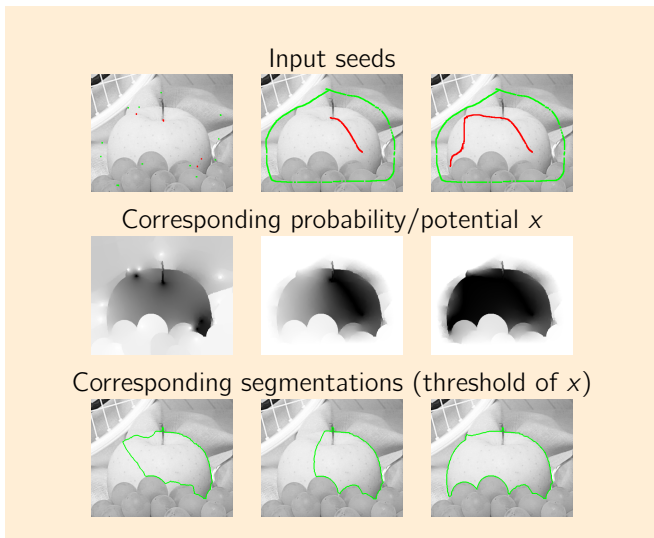
- Potentials analogy



- Random walker analogy



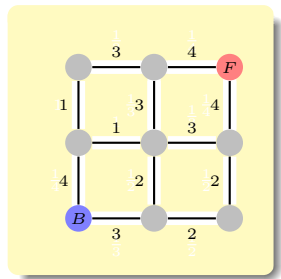
# Random Walker : example



# Shortest path forest

- take the inverse of the weights
- the shortest path starting from each node to reach a seed node is computed
- Dijkstra algorithm
- [Sinop et al. 07] :

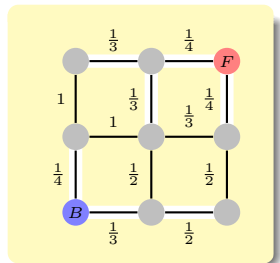
$$\lim_{p=q \rightarrow \infty} \bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^{p=q} (x_i - x_j)^q$$



# Shortest path forest

- take the inverse of the weights
- the shortest path starting from each node to reach a seed node is computed
- Dijkstra algorithm
- [Sinop et al. 07] :

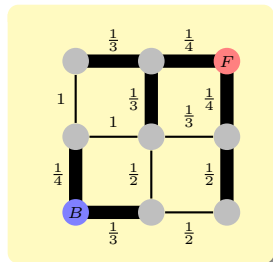
$$\lim_{p=q \rightarrow \infty} \bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^{p=q} (x_i - x_j)^q$$



# Shortest path forest

- take the inverse of the weights
- the shortest path starting from each node to reach a seed node is computed
- Dijkstra algorithm
- [Sinop et al. 07] :

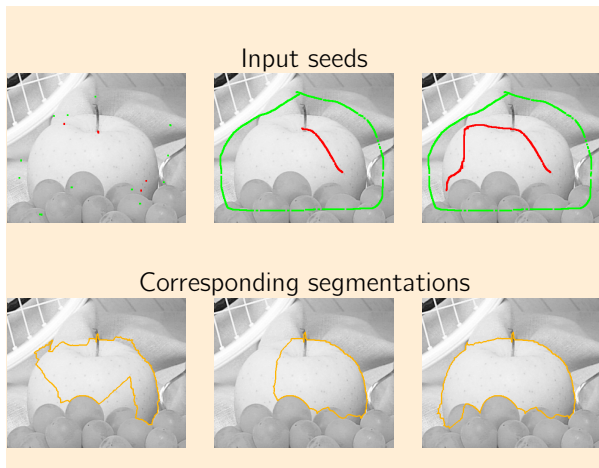
$$\lim_{p=q \rightarrow \infty} \bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^{p=q} (x_i - x_j)^q$$





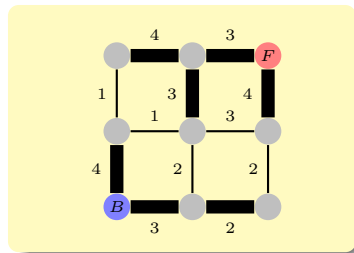
# Shortest path forest : example

- Very sensitive to the object centering relatively to the seeds



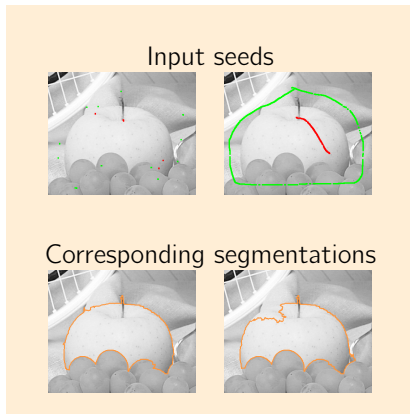
# Maximum Spanning Forest (MSF)

- maximize the sum of weights over the edges of a forest spanning the graph
- different labeled nodes have to belong to different trees
- Kruskal, Prim algorithms



# Maximum Spanning Forest (MSF) : example

- robust to small seeds
- leaking effect



# Algorithms deriving from values of $p$ et $q$

Recall the energy function :  $\bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$

| q \ p    | 0                      | finite                 | $\infty$                                   |
|----------|------------------------|------------------------|--|
| 1        | Reduction to seeds     | Graph cuts             | Max Spanning Forest<br>[Allène et al. 07]  |
| 2        | $\ell_2$ -norm Voronoi | Random walker          | Max Spanning Forest<br>[Couprie et al. 09] |
| $\infty$ | $\ell_1$ -norm Voronoi | $\ell_1$ -norm Voronoi | Shortest Path Forest<br>[Sinop et al. 07]  |

# Algorithms deriving from values of $p$ et $q$

Recall the energy function :  $\bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$

| q \ p    | 0                      | finite                 | $\infty$                                   |
|----------|------------------------|------------------------|--|
| 1        | Reduction to seeds     | Graph cuts             | Max Spanning Forest<br>[Allène et al. 07]  |
| 2        | $\ell_2$ -norm Voronoi | Random walker          | Max Spanning Forest<br>[Couprie et al. 09] |
| $\infty$ | $\ell_1$ -norm Voronoi | $\ell_1$ -norm Voronoi | Shortest Path Forest<br>[Sinop et al. 07]  |

# Algorithms deriving from values of $p$ et $q$

Recall the energy function :  $\bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$

| $q \backslash p$ | 0                      | finite                 | $\infty$                                   |
|------------------|------------------------|------------------------|--|
| 1                | Reduction to seeds     | Graph cuts             | Max Spanning Forest<br>[Allène et al. 07]  |
| 2                | $\ell_2$ -norm Voronoi | Random walker          | Max Spanning Forest<br>[Couprie et al. 09] |
| $\infty$         | $\ell_1$ -norm Voronoi | $\ell_1$ -norm Voronoi | Shortest Path Forest<br>[Sinop et al. 07]  |

# Algorithms deriving from values of $p$ et $q$

Recall the energy function :  $\bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$

| $q \backslash p$ | 0                      | finite                 | $\infty$                                   |
|------------------|------------------------|------------------------|--|
| 1                | Reduction to seeds     | Graph cuts             | Max Spanning Forest<br>[Allène et al. 07]  |
| 2                | $\ell_2$ -norm Voronoi | Random walker          | Max Spanning Forest<br>[Couprie et al. 09] |
| $\infty$         | $\ell_1$ -norm Voronoi | $\ell_1$ -norm Voronoi | Shortest Path Forest<br>[Sinop et al. 07]  |

# Algorithms deriving from values of $p$ et $q$

Recall the energy function :  $\bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$

| $q \backslash p$ | 0                      | finite                 | $\infty$                                   |
|------------------|------------------------|------------------------|--|
| 1                | Reduction to seeds     | Graph cuts             | Max Spanning Forest<br>[Allène et al. 07]  |
| 2                | $\ell_2$ -norm Voronoi | Random walker          | Max Spanning Forest<br>[Couprie et al. 09] |
| $\infty$         | $\ell_1$ -norm Voronoi | $\ell_1$ -norm Voronoi | Shortest Path Forest<br>[Sinop et al. 07]  |



# Algorithms deriving from values of $p$ et $q$

Recall the energy function :  $\bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$

| $q \backslash p$ | 0                      | finite                 | $\infty$                                   |
|------------------|------------------------|------------------------|--|
| 1                | Reduction to seeds     | Graph cuts             | Max Spanning Forest<br>[Allène et al. 07]  |
| 2                | $\ell_2$ -norm Voronoi | Random walker          | Max Spanning Forest<br>[Coupric et al. 09] |
| $\infty$         | $\ell_1$ -norm Voronoi | $\ell_1$ -norm Voronoi | Shortest Path Forest<br>[Sinop et al. 07]  |

# Algorithms deriving from values of $p$ et $q$

Recall the energy function :  $\bar{x}_{pq} = \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$

| $q \backslash p$ | 0                      | finite                 | $\infty$                                   |
|------------------|------------------------|------------------------|--|
| 1                | Reduction to seeds     | Graph cuts             | Max Spanning Forest<br>[Allène et al. 07]  |
| 2                | $\ell_2$ -norm Voronoi | Random walker          | Max Spanning Forest<br>[Couprie et al. 09] |
| $\infty$         | $\ell_1$ -norm Voronoi | $\ell_1$ -norm Voronoi | Shortest Path Forest<br>[Sinop et al. 07]  |

## Algorithm for the case $p = \infty$ , variable $q$

$$\bar{x}_q = \lim_{p \rightarrow \infty} x_{p,q}^*$$

Power watershed algorithm (outline)

Build an MSF outside of plateaus, and optimize on plateaus

$$\sum_{e_{ij} \in \text{plateau}} |x_i - x_j|^q$$

## Algorithm for the case $p = \infty$ , variable $q$

$$\bar{x}_q = \lim_{p \rightarrow \infty} x_{p,q}^*$$

### Power watershed algorithm (outline)

Build an MSF outside of plateaus, and optimize on plateaus

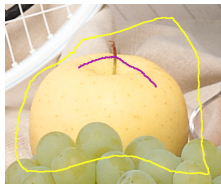
$$\sum_{e_{ij} \in \text{plateau}} |x_i - x_j|^q$$

### Theorem (Convergence)

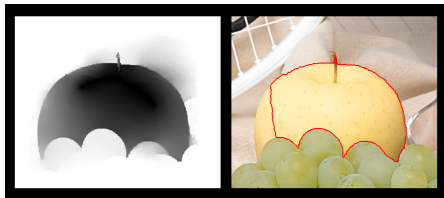
*If  $q > 1$ , the potential  $\bar{x}_{p,q}$  converges, as  $p \rightarrow \infty$ , towards the potential  $\bar{x}_q$  obtained by the Power Watershed algorithm.*

# Convergence of RW when $p \rightarrow \infty$ towards PW

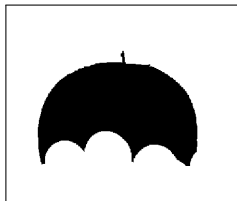
Input seeds



Random Walker  $p = 1 \dots 30$



PowerWatershed  $q = 2$



Random Walker  $p = 30$



# Theorems

## Theorem (MSF cut)

*The cut obtained by the Powerwatershed algorithm is a MSF cut.*

## Theorem (Watershed cut)

*The cut obtained by the Powerwatershed algorithm is a watershed cut of the graph morphologically reconstructed from the seeds.*

## Theorem (Uniqueness)

*When  $q > 1$ , the solution  $x^*$  to the minimization of*

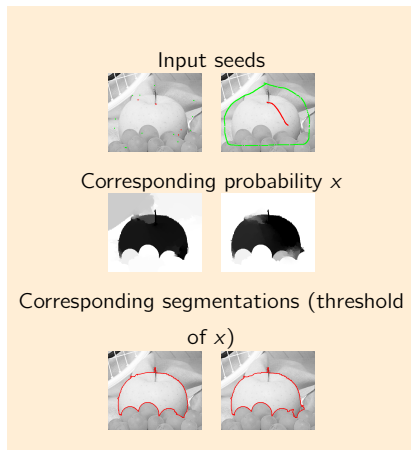
$$\lim_{p \rightarrow \infty} \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$$

*is unique.*

*(Thus, when  $q > 1$ , the solution  $\bar{x}$  of the Powerwatershed algorithm is unique.)*

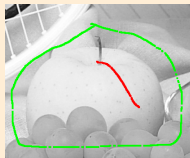
## Powerwatershed ( $q=2$ ) : example

- robust to small seeds size
- less leaking than with standard Maximum Spanning Forest



# Powerwatershed ( $q=2$ ) : example

Input seeds



Powerwatershed ( $q = 2$ )



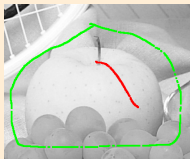
Prim algorithm for MSF



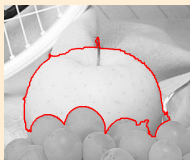


# Powerwatershed ( $q=2$ ) : example

Input seeds



Powerwatershed ( $q = 2$ )



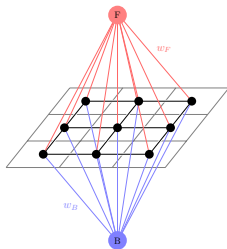
Prim algorithm for MSF



# Generality of the framework

- Possibility to add unary terms to the energy function

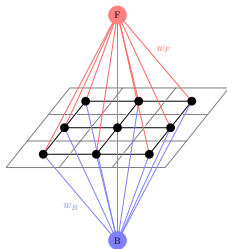
$$\min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$$



# Generality of the framework

- Possibility to add unary terms to the energy function

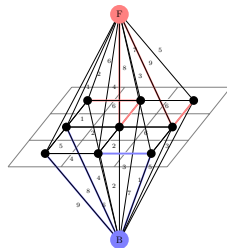
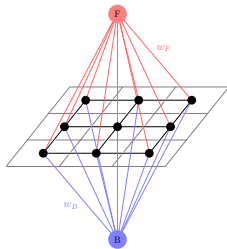
$$\min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q + \sum_{v_i} w_{F_i}^p |x_i - 1|^q + \sum_{v_i} w_{B_i}^p |x_i|^q$$



# Generality of the framework

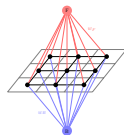
- Possibility to add unary terms to the energy function

$$\min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q + \sum_{v_i} w_{F_i}^p |x_i - 1|^q + \sum_{v_i} w_{B_i}^p |x_i|^q$$



# Generality of the framework

$$\min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q + \sum_{v_i} w_{F_i}^p |x_i - 1|^q + \sum_{v_i} w_{B_i}^p |x_i|^q$$



Image



Graph Cuts

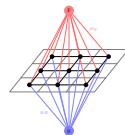


MaxSF



## Generality of the framework

$$\min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q + \sum_{v_i} w_{F_i}^p |x_i - 1|^q + \sum_{v_i} w_{B_i}^p |x_i|^q$$



Image



Graph Cuts



MaxSF



### Contribution

To the best of our knowledge, this is the first time that watershed is used in other applications than seeded segmentation

# Optimal multilabels segmentation

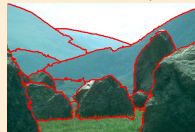
- More than 2-labels segmentation : NP-hard for Graph cuts
- Exact  $n \geq 2$  labels segmentation for the other algorithms :
- $n$  solutions  $x^1, x^2, \dots, x^n$  computed
- $x^k$  computed by enforcing  $\begin{cases} x^k(n^k) = 1 \\ x^k(n^q) = 0 \text{ for all } q \neq k. \end{cases}$
- Each node  $i$  is affected to the label for which  $x_i^k$  is maximum :

$$s_i = \arg \max_k x_i^k$$

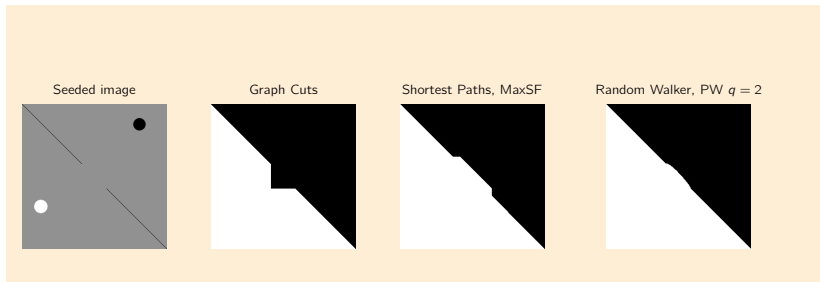
Input seeds



Segmentation by PowerWatershed ( $q = 2$ )

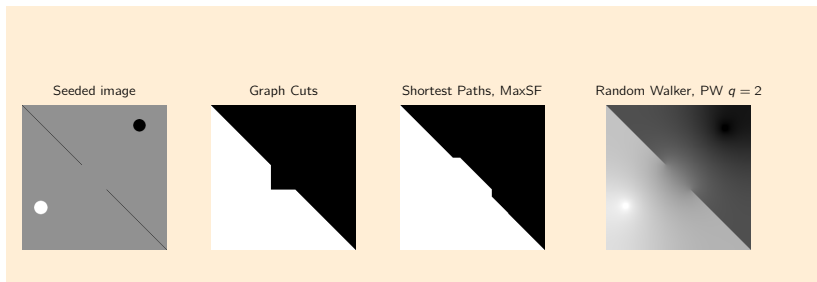


# Algorithms behavior on plateaus

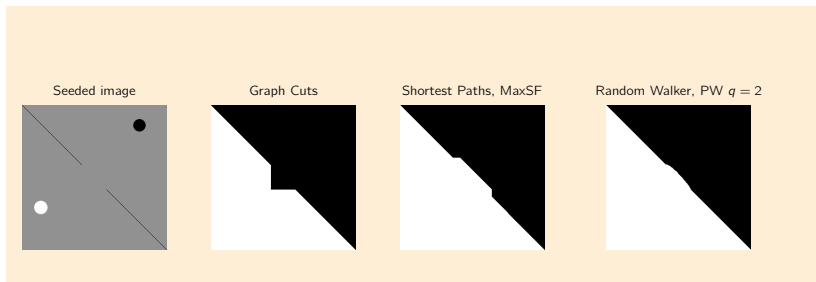




# Algorithms behavior on plateaus



# Algorithms behavior on plateaus

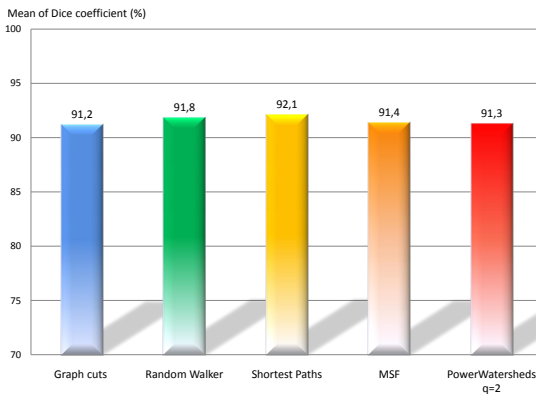


# Algorithms comparison

- Evaluation on Berkeley database
- Ground truths
- 2 sets of seeds to study robustness to seeds centering :
  - 1 well centered seeds
  - 2 less centered seeds

## Quantitative Results

Dice coeff. between ground truths and the algorithms results on Berkeley database with the centered seeds.

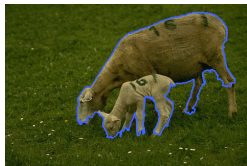


# Examples

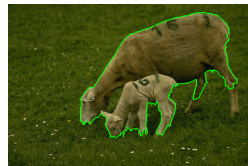
Input seeds



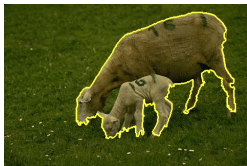
Graph Cuts



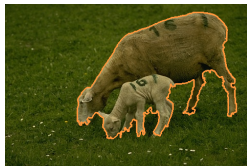
Random Walker



Shortest Paths



Max Spanning Forests

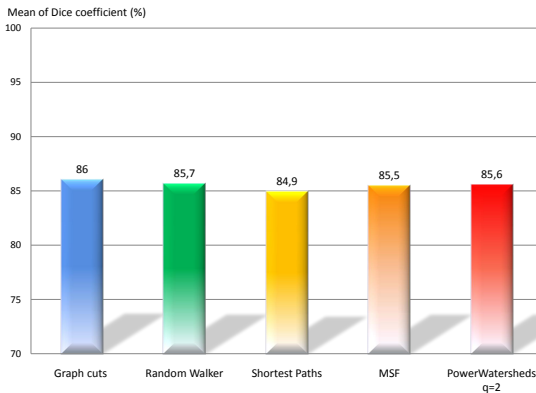


Power Watersheds  $q = 2$



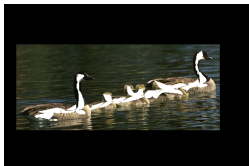
## Quantitative Results

Dice coeff. between ground truths and the algorithms results on Berkeley database with the less centered seeds.

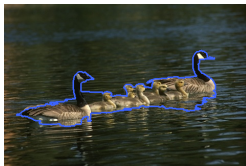


# Examples

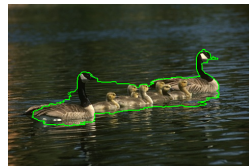
Input seeds



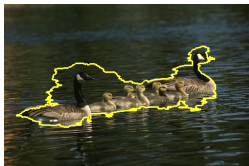
Graph Cuts



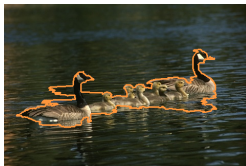
Random Walker



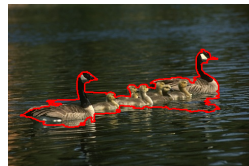
Shortest Paths



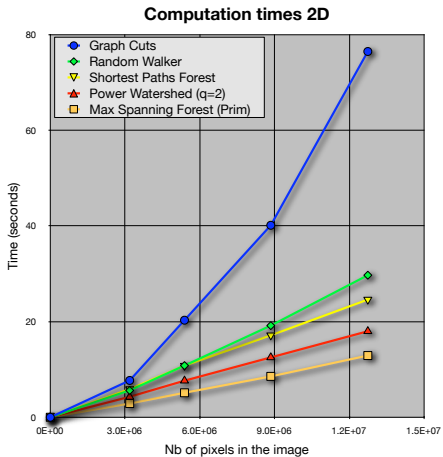
Max Spanning Forests



Power Watersheds  $q = 2$

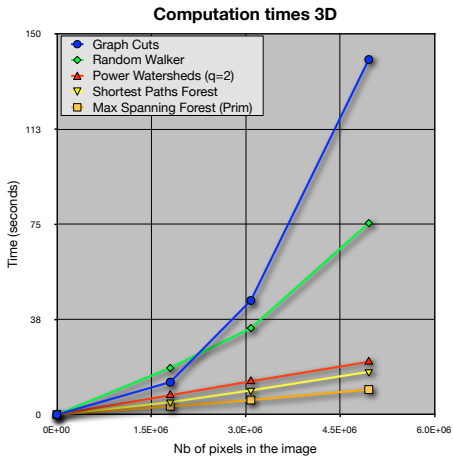


# Computation time 2D





# Computation time 3D



## Which algorithm to use ?

- Graph Cuts :
  - good fit for 2D image segmentation in 2 labels
  - too slow for 3D segmentation
- Shortest Paths : segmentation of well centered seeds around the object
- Random Walker :
  - efficient with uncentered seeds
  - defined behavior on plateaus
- Max SF :
  - better segmentations than SPF with uncentered seeds
  - fast → 3D segmentation
- Powerwatershed  $q = 2$  :
  - MaxSF properties
  - less sensitive to leaking than standard MaxSF

## Conclusion

- New framework unifying Graph Cuts, Random Walker, MSF and SPF.

## Conclusion

- New framework unifying Graph Cuts, Random Walker, MSF and SPF.
- New optimization algorithms family with variable  $q$ .

## Conclusion

- New framework unifying Graph Cuts, Random Walker, MSF and SPF.
- New optimization algorithms family with variable  $q$ .
- The  $q = 2$  algorithm shows segmentation improvement while retaining watershed speed.

## Conclusion

- New framework unifying Graph Cuts, Random Walker, MSF and SPF.
- New optimization algorithms family with variable  $q$ .
- The  $q = 2$  algorithm shows segmentation improvement while retaining watershed speed.
- Unary terms formulation makes powerwatersheds useful beyond segmentation.

## Conclusion

- New framework unifying Graph Cuts, Random Walker, MSF and SPF.
- New optimization algorithms family with variable  $q$ .
- The  $q = 2$  algorithm shows segmentation improvement while retaining watershed speed.
- Unary terms formulation makes powerwatersheds useful beyond segmentation.

### Contribution

The power watershed leads to a multilabel, scale and contrast invariant, unique global optimum obtained in practice in quasi-linear time

## Conclusion

- New framework unifying Graph Cuts, Random Walker, MSF and SPF.
- New optimization algorithms family with variable  $q$ .
- The  $q = 2$  algorithm shows segmentation improvement while retaining watershed speed.
- Unary terms formulation makes powerwatersheds useful beyond segmentation.

### Contribution

The power watershed leads to a multilabel, scale and contrast invariant, unique global optimum obtained in practice in quasi-linear time



# Non-convex diffusion using power watersheds

- Anisotropic diffusion [Perona-Malik 1990]

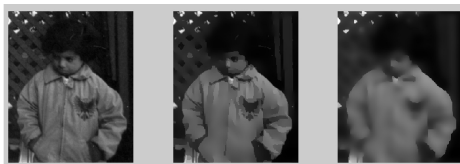


Image      100 iterations      200 iterations

Goals of this work :

- perform anisotropic diffusion using an  $\ell_0$  norm to avoid the blurring effect
- optimize a non convex energy using Power Watershed [Couprie-Grady-Najman-Talbot, ICIP 2010]

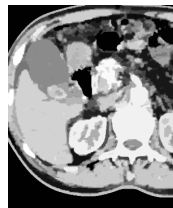
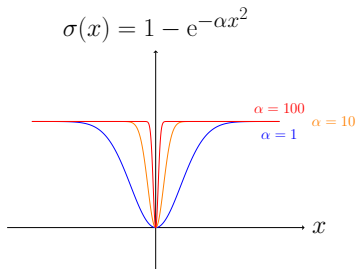
# Anisotropic diffusion and $\ell_0$ norm

$$x^* = \arg \min_x \underbrace{\sum_{e_{ij} \in E} \sigma(x_i - x_j)}_{\text{smoothness term}} + \lambda \underbrace{\sum_{v_i \in V} \sigma(x_i - f_i)}_{\text{data fidelity term}}$$

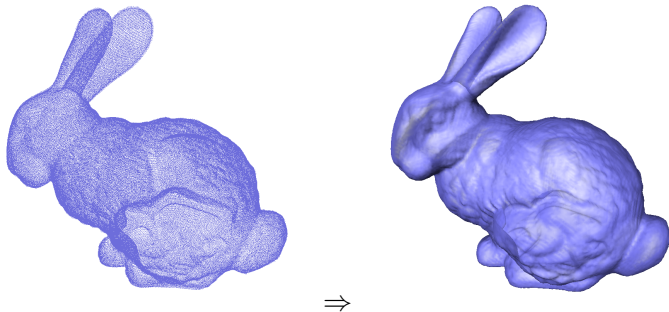
Leads to piecewise constant results

Original image

PW result



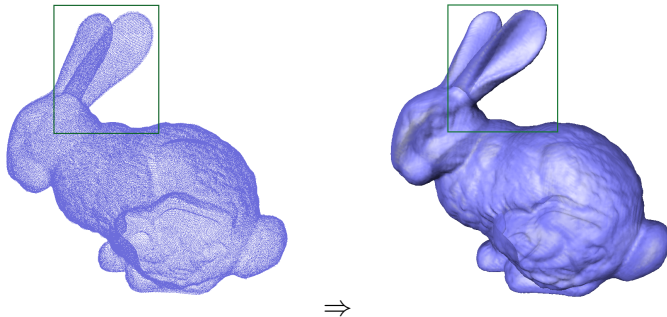
# Surface reconstruction from a noisy set of dots



- Goal : given a noisy set of dots, find an explicit surface fitting the dots.

Joint work with Xavier Bresson

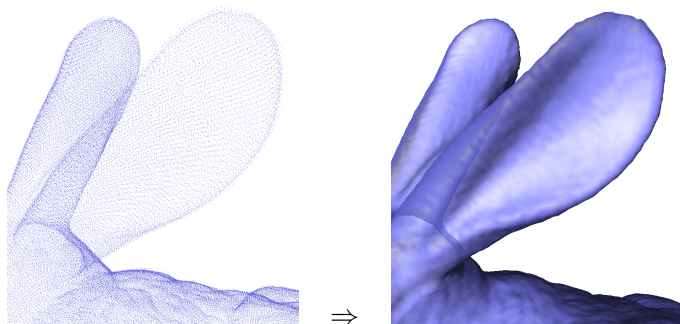
## Surface reconstruction from a noisy set of dots



- Goal : given a noisy set of dots, find an explicit surface fitting the dots.

Joint work with Xavier Bresson

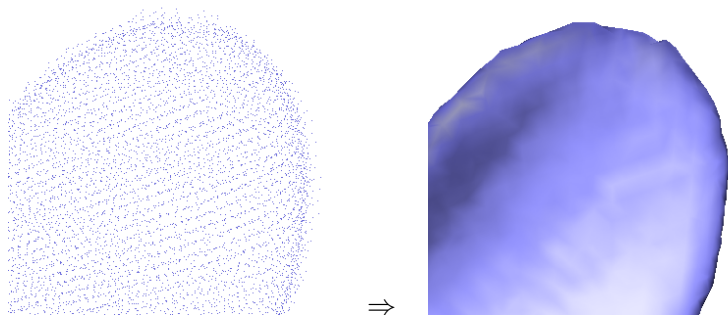
# Surface reconstruction from a noisy set of dots



- Goal : given a noisy set of dots, find an explicit surface fitting the dots.

Joint work with Xavier Bresson

# Surface reconstruction from a noisy set of dots



- Goal : given a noisy set of dots, find an explicit surface fitting the dots.

Joint work with Xavier Bresson

# How to solve this problem

- Graph : 3D grid
- Here  $x$  represents the object indicator to recover.

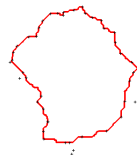
$$\bar{x} = \lim_{p \rightarrow \infty} \arg \min_x \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q$$

$$\text{s.t. } x(F) = 1, x(B) = 0$$

- weights : distance function from the set of dots to fit

Why PW are a good fit for this problem ?

numerous plateaus around the dots to fit  $\rightarrow$   
 smooth isosurface is obtained



Power  
 watershed  
 solution

# Perspectives



## Future work

- Study of the different energies possibly minimized in this framework







## Some papers on watershed cuts

### Bibliography on watershed cuts

-  Cousty, J., Bertrand, B., Najman, L. and Couprie, M. :  
Watershed cuts : minimum spanning forests and the drop of water principle.  
*IEEE Transactions on PAMI*, 31(8) :1362–1374, Aug. 2009.
-  Cousty, J., Bertrand, G., Najman, L. and Couprie, M. :  
Watershed cuts : thinnings, shortest-path forests and topological watersheds.  
*IEEE Transactions on PAMI*, 32(5) :925-939, May 2010

## Some papers on Power watersheds

### Bibliography on powerwatersheds

-  Couprie, C., Grady, L., Najman, L. and Talbot, H. :  
Power Watersheds : A Unifying Graph Based Optimization  
Framework.  
*IEEE Transactions on PAMI*, 33(7) :1384-1399, July 2011.
-  C. Couprie, X. Bresson, L. Najman, H. Talbot and L. Grady :  
Surface reconstruction using Power watersheds. In *Proc. of  
ISMM 2011*.
-  C. Couprie, L. Grady, L. Najman, and H. Talbot : Anisotropic  
diffusion using power watersheds. In *Proc. of ICIP 2010*.
-  Couprie, C., Grady, L., Najman, L. and Talbot, H. :  
Power watersheds : A new image segmentation framework  
extending graph cuts, random walker and optimal spanning  
forest.  
In *Proc. of ICCV*, pages 731–738, Sept. 2009.

## Questions



Source code available from

<http://sourceforge.net/projects/powerwatershed/>