Watersheding hierarchies: recognition of hierarchical watersheds

Deise Santana Maia
Jean Cousty, Laurent Najman, Benjamin Perret

ESIEE Paris, Université Paris-Est, Laboratoire Gaspard-Monge

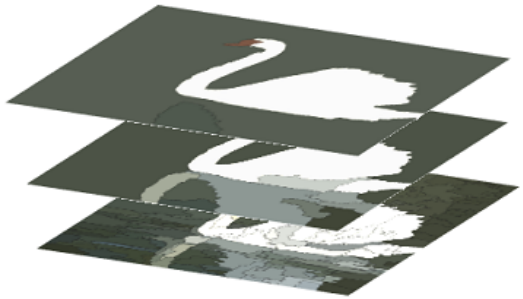
June 19, 2018

Hierarchical watersheds

- Sequences of nested partitions such that:
  - Each partition is a watershed of a filtering of the initial relief/map
  - Filtering according to regional attributes based, *e.g.*, on geometric, photometric, or learned, information
  - Good performance on natural images [P18]

Original image

Hierarchical watershed $\mathcal{H}$



Saliency map of $\mathcal{H}$ [1]

[1] *i.e.*, a characteristic function of $\mathcal{H}$, see previous presentation of Jean Cousty

[P18] B. Perret, J Cousty, S. Guimaraes, D. Maia. Evaluation of hierarchical watersheds. IEEE TIP. 2018.

Combination of hierarchies (of watersheds)

- Based on saliency maps combination [C17, M17]
  - with, *e.g.*, infimum, supremum, or linear combination

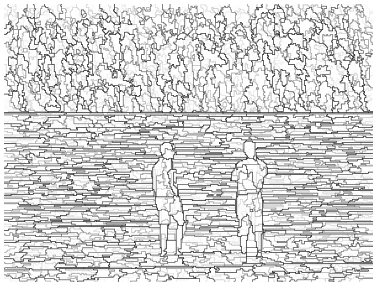
[C17] J. Cousty, L. Najman, Y. Kenmochi, S. Guimarães. Hierarchical segmentations with graphs: quasi-flat zones, minimum spanning trees, and saliency maps. JMIV. 2017

[M17] D. S. Maia, A. de A. Araujo, J. Cousty, L. Najman, B. Perret, H. Talbot. Evaluation of combinations of watershed hierarchies. ISMM. 2017

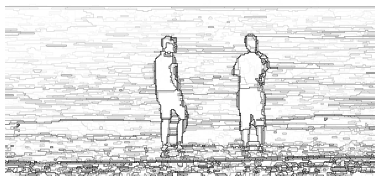
Combination of hierarchies (of watersheds)

- Based on saliency maps combination [C17, M17]
  - with, *e.g.*, infimum, supremum, or linear combination

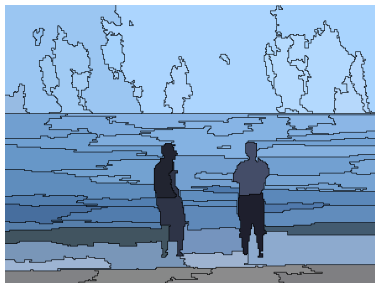
Original

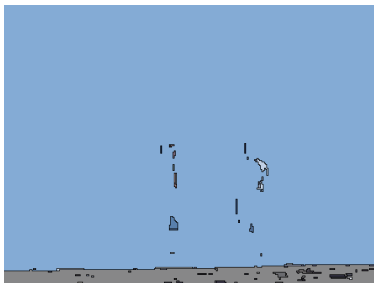


Area attribute



Dynamics attribute





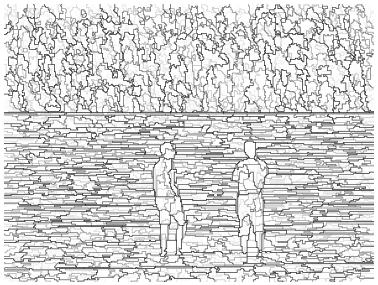
One level of each hierarchy with 75 levels

[C17] J. Cousty, L. Najman, Y. Kenmochi, S. Guimarães. Hierarchical segmentations with graphs: quasi-flat zones, minimum spanning trees, and saliency maps. JMIV. 2017

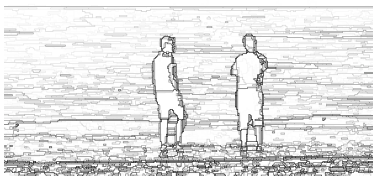
[M17] D. S. Maia, A. de A. Araujo, J. Cousty, L. Najman, B. Perret, H. Talbot. Evaluation of combinations of watershed hierarchies. ISMM. 2017

Combination of hierarchies (of watersheds)

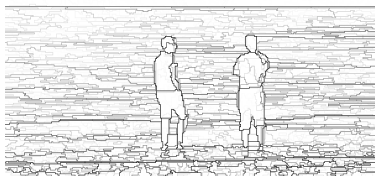
- Based on saliency maps combination [C17, M17]
  - with, *e.g.*, infimum, supremum, or linear combination



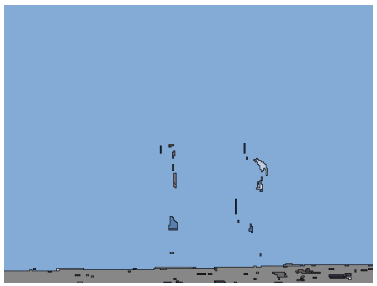
Area attribute

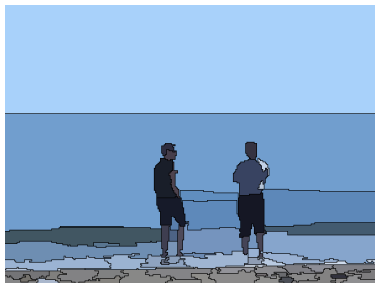


Dynamics attribute



Combination by average





One level of each hierarchy with 75 levels

[C17] J. Cousty, L. Najman, Y. Kenmochi, S. Guimarães. Hierarchical segmentations with graphs: quasi-flat zones, minimum spanning trees, and saliency maps. JMIV. 2017

[M17] D. S. Maia, A. de A. Araujo, J. Cousty, L. Najman, B. Perret, H. Talbot. Evaluation of combinations of watershed hierarchies. ISMM. 2017

Combinations of hierarchical watersheds

Theoretical problem

Does the combination of two hierarchical watersheds always result in a hierarchy which itself is a hierarchical watershed?

Combinations of hierarchical watersheds

Theoretical problem

Does the combination of two hierarchical watersheds always result in a hierarchy which itself is a hierarchical watershed?

- Is there a regional attribute A such that the combination of two hierarchical watersheds is the hierarchical watershed obtained when filtering the function with attribute A?
- Does the combination of hierarchical watersheds allow to go beyond hierarchical watersheds?

Recognition of hierarchical watersheds

Problem

Given a hierarchy $\mathcal{H}$ and a weighted graph (G, w):

- *Recognize if $\mathcal{H}$ is a hierarchical watershed of (G, w)*

Recognition of hierarchical watersheds

Problem

Given a hierarchy $\mathcal{H}$ and a weighted graph (G, w):

- *Recognize if $\mathcal{H}$ is a hierarchical watershed of (G, w)*

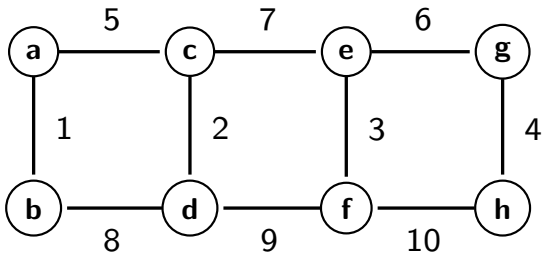
Contributions

- *Introduction of the* watersheding transform
  - *which transforms any hierarchy into a hierarchical watershed*
  - *whose fixed points are the hierarchical watersheds*
- *Characterization of hierarchical watersheds*
- *Quasi-linear algorithm to recognize the hierarchical watersheds*

Outlines

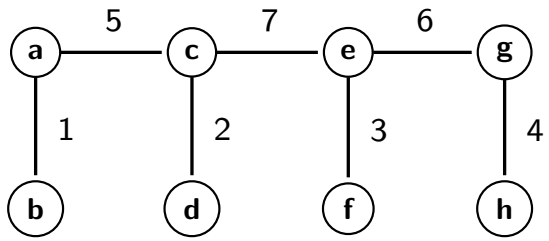
Graph settings

- (G, w) is an edge-weighted graph
  - which can be a pixel-adjacency graph
  - edge weight can represent a gradient of intensity



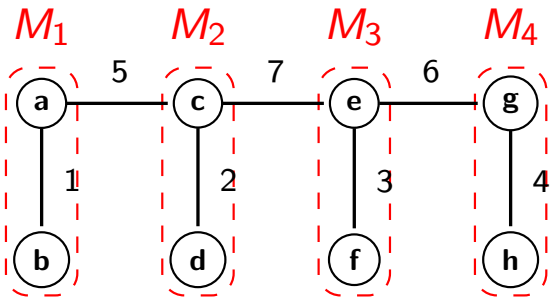
Graph settings

- (G, w) is an edge-weighted graph
  - which can be a pixel-adjacency graph
  - edge weight can represent a gradient of intensity
- For simplification, we consider that *G is a tree and that the edge-weights are pairwise distinct*



Graph settings

- (G, w) is an edge-weighted graph
  - which can be a pixel-adjacency graph
  - edge weight can represent a gradient of intensity
- For simplification, we consider that *G is a tree and that the edge-weights are pairwise distinct*
- $\mathcal{M} = (M_0, \ldots, M_\ell)$ is any sequence of the regional minima of w ranked by importance according to some given attribute

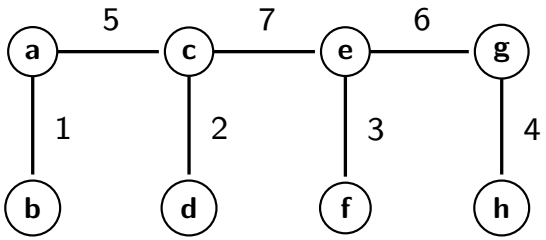


Hierarchical watersheds

- A *hierarchical watershed of* (G, w) *for* $\mathcal{M}$ is a hierarchy of partitions $(\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ such that, for any $i \in \{0, \ldots, \ell\}$:
  - $\mathbf{P}_i$ is the connected component partition of a minimum spanning forest rooted in the minima ranked after i

Hierarchical watersheds

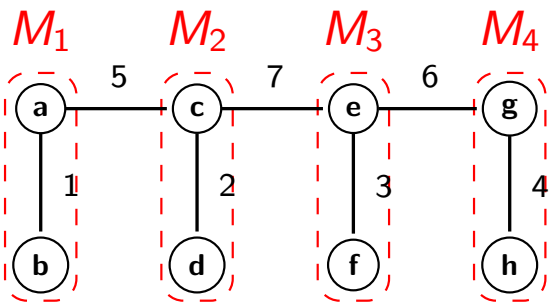
- A *hierarchical watershed of* (G, w) *for* $\mathcal{M}$ is a hierarchy of partitions $(\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ such that, for any $i \in \{0, \ldots, \ell\}$:
  - $\mathbf{P}_i$ is the connected component partition of a minimum spanning forest rooted in the minima ranked after i



(G, w)

Hierarchical watersheds

- A *hierarchical watershed of* (G, w) *for* $\mathcal{M}$ is a hierarchy of partitions $(\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ such that, for any $i \in \{0, \ldots, \ell\}$:
  - $\mathbf{P}_i$ is the connected component partition of a minimum spanning forest rooted in the minima ranked after i

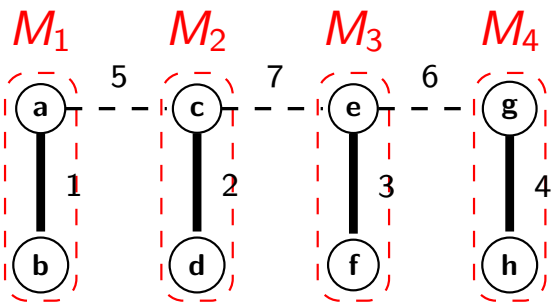


Minima of w

$$\mathcal{M} = (M_1, M_2, M_3, M_4)$$

Hierarchical watersheds

- A *hierarchical watershed of* (G, w) *for* $\mathcal{M}$ is a hierarchy of partitions $(\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ such that, for any $i \in \{0, \ldots, \ell\}$:
  - $\mathbf{P}_i$ is the connected component partition of a minimum spanning forest rooted in the minima ranked after i

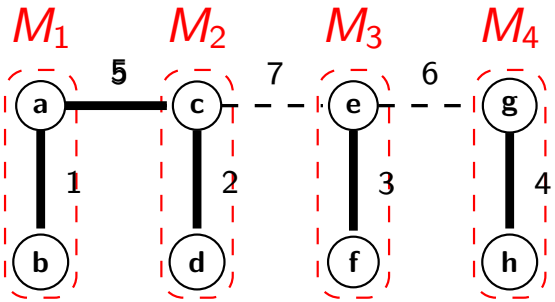


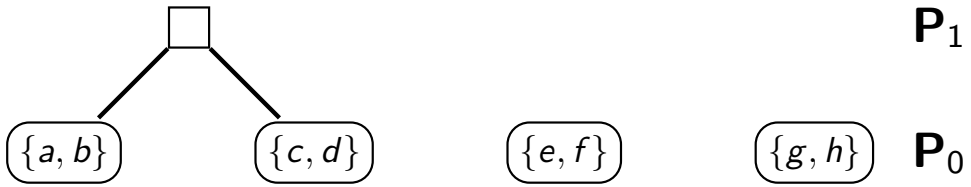
$\{a, b\}$ $\{c, d\}$ $\{e, f\}$ $\{g, h\}$ $\mathbf{P}_0$

$$\mathcal{M} = (M_1, M_2, M_3, M_4)$$

Hierarchical watersheds

- A *hierarchical watershed of* (G, w) *for* $\mathcal{M}$ is a hierarchy of partitions $(\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ such that, for any $i \in \{0, \ldots, \ell\}$:
  - $\mathbf{P}_i$ is the connected component partition of a minimum spanning forest rooted in the minima ranked after i

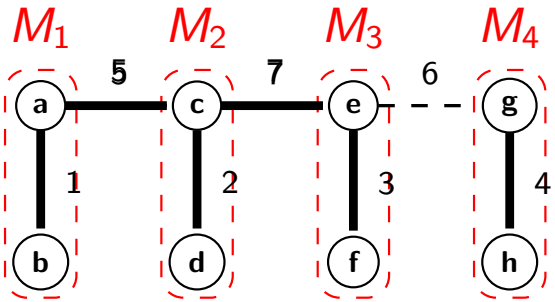


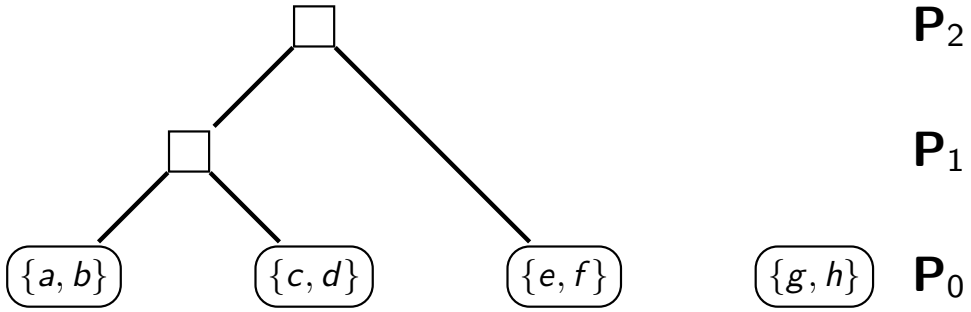


$$\mathcal{M} = (M_1, M_2, M_3, M_4)$$

Hierarchical watersheds

- A *hierarchical watershed of* (G, w) *for* $\mathcal{M}$ is a hierarchy of partitions $(\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ such that, for any $i \in \{0, \ldots, \ell\}$:
  - $\mathbf{P}_i$ is the connected component partition of a minimum spanning forest rooted in the minima ranked after i

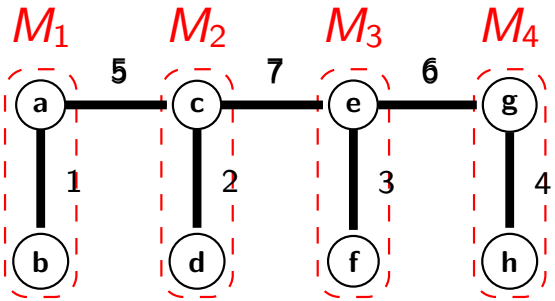


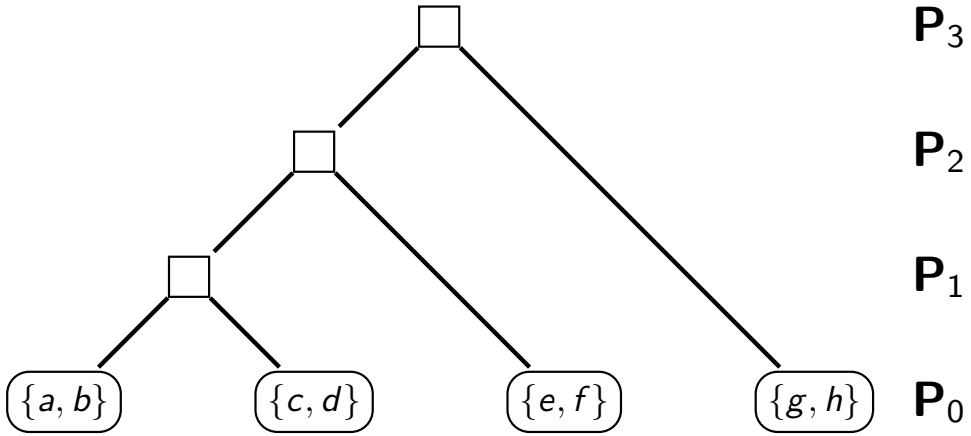


$$\mathcal{M} = (M_1, M_2, M_3, M_4)$$

Hierarchical watersheds

- A *hierarchical watershed of* (G, w) *for* $\mathcal{M}$ is a hierarchy of partitions $(\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ such that, for any $i \in \{0, \ldots, \ell\}$:
  - $\mathbf{P}_i$ is the connected component partition of a minimum spanning forest rooted in the minima ranked after i





$(\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3)$:
hierarchical watershed of (G, w) for $\mathcal{M}$

Hierarchical watersheds

Definition

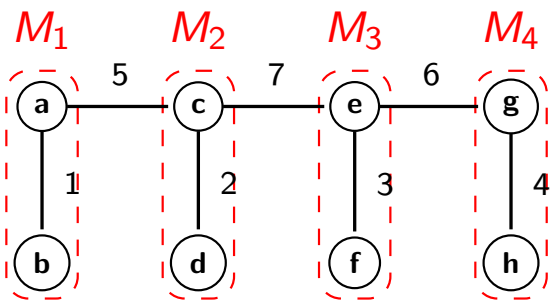
We say that $\mathcal{H}$ is a *hierarchical watershed of* (G, w) if there is a sequence $\mathcal{M}$ of minima such that $\mathcal{H}$ is the hierarchical watershed of (G, w) for $\mathcal{M}$

Hierarchical watersheds

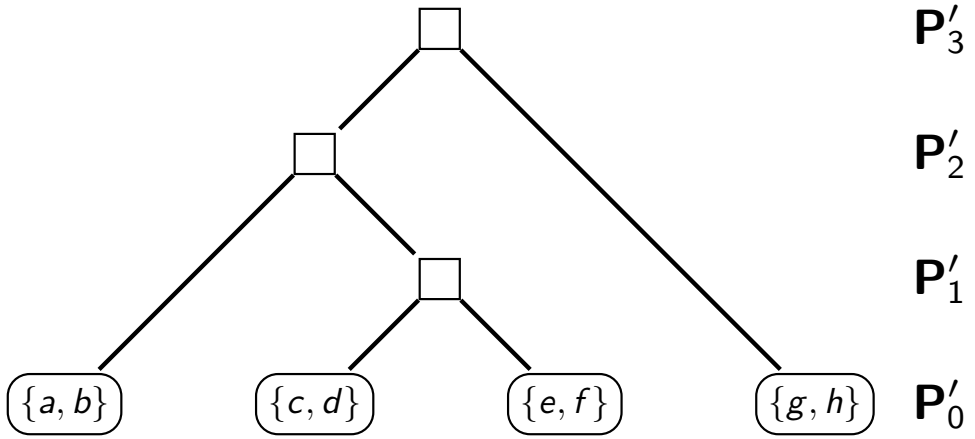
Definition

We say that $\mathcal{H}$ is a *hierarchical watershed of* (G, w) if there is a sequence $\mathcal{M}$ of minima such that $\mathcal{H}$ is the hierarchical watershed of (G, w) for $\mathcal{M}$

Remark: there are hierarchies which are not hierarchical watershed of a graph, e.g., $(\mathbf{P}'_0, \mathbf{P}'_1, \mathbf{P}'_2, \mathbf{P}'_3)$ *is* **not** *a hierarchical watershed of* (G, w)



(G, w)



Outlines

Hierarchical watershed recognition: general overview

Key idea for recognizing hierarchical watersheds:

- "Inverse" the hierarchical watershed algorithm of [C13, N13]

Given a hierarchy $\mathcal{H}$ and a weighted graph (G, w):

1. compute the *binary partition hierarchy (by altitude ordering)* of w
2. compute the saliency map s of $\mathcal{H}$
3. compute the *watershed-cut ordering* s_o from s
4. compute the *extinction estimation* ξ from s_o
5. compute the *persistence estimation* π from ξ
6. if $\pi = s$ then and only the $\mathcal{H}$ is a a hierarchical watershed

---

[C13] J. Cousty, L. Najman, B. Perret. Constructive links between some morphological hierarchies on edge-weighted graphs. ISMM 2013.

[N13] L. Najman, J. Cousty, B. Perret. Playing with Kruskal: algorithms for morphological trees in edge-weighted graphs.. ISMM 2013.

Hierarchical watershed recognition: general overview

Key idea for recognizing hierarchical watersheds:

- "Inverse" the hierarchical watershed algorithm of [C13, N13]

Given a hierarchy $\mathcal{H}$ and a weighted graph (G, w):

1. compute the *binary partition hierarchy (by altitude ordering)* of w
2. compute the saliency map s of $\mathcal{H}$
3. compute the *watershed-cut ordering* s_o from s
4. compute the *extinction estimation* ξ from s_o
5. compute the *persistence estimation* π from ξ
6. if $\pi = s$ then and only the $\mathcal{H}$ is a a hierarchical watershed

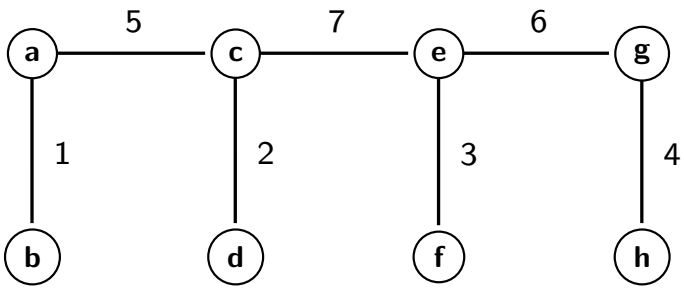
---

[C13] J. Cousty, L. Najman, B. Perret. Constructive links between some morphological hierarchies on edge-weighted graphs. ISMM 2013.

[N13] L. Najman, J. Cousty, B. Perret. Playing with Kruskal: algorithms for morphological trees in edge-weighted graphs.. ISMM 2013.

Binary partition hierarchy

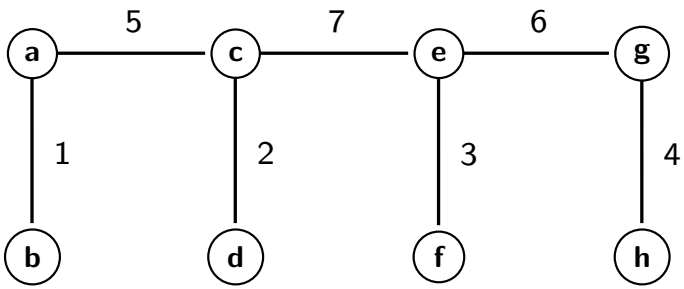
Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



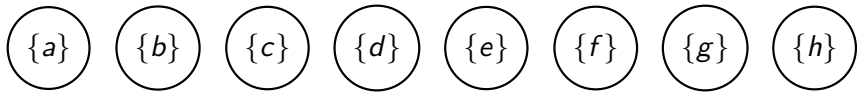
(G, w)

Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$

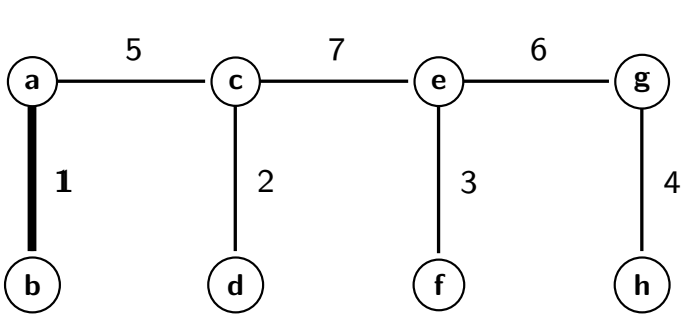


(G, w)

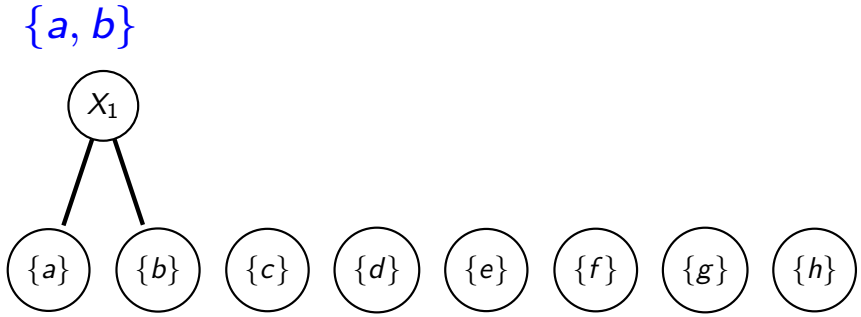


Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



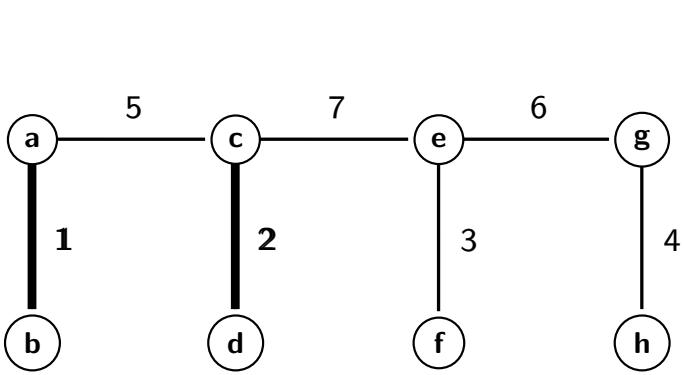
(G, w)



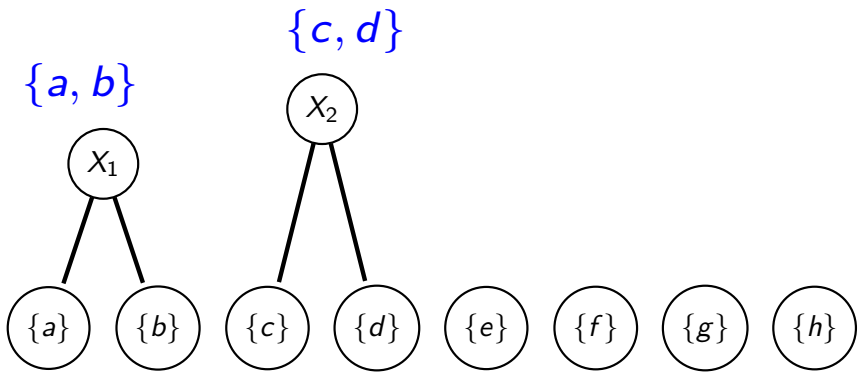
Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)

Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



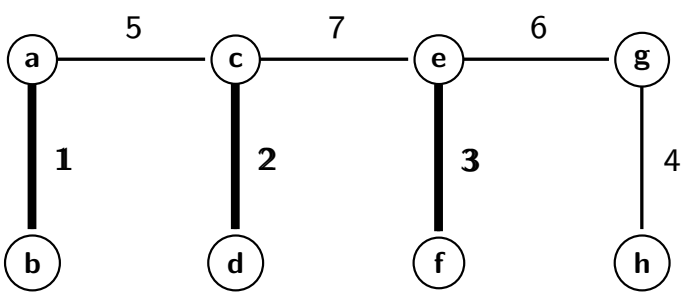
(G, w)



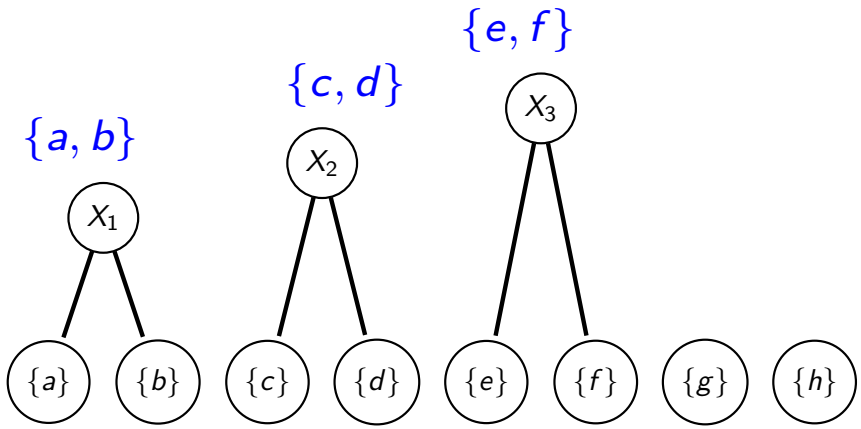
Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)

Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



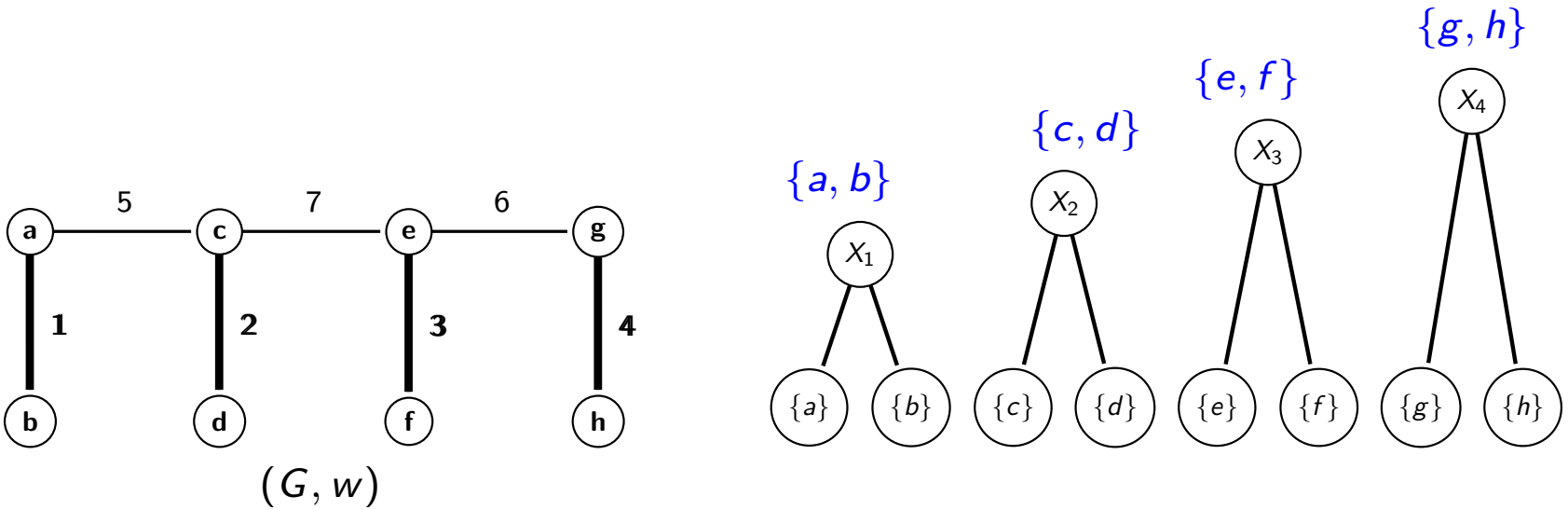
(G, w)



Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)

Binary partition hierarchy

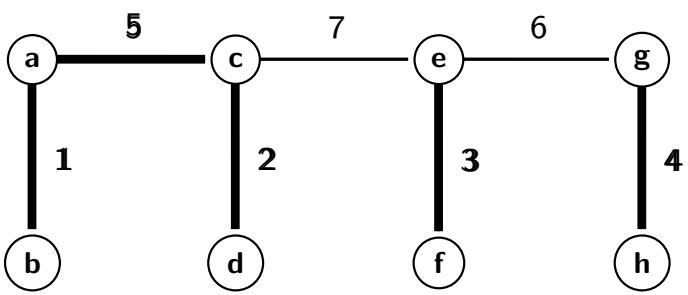
Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



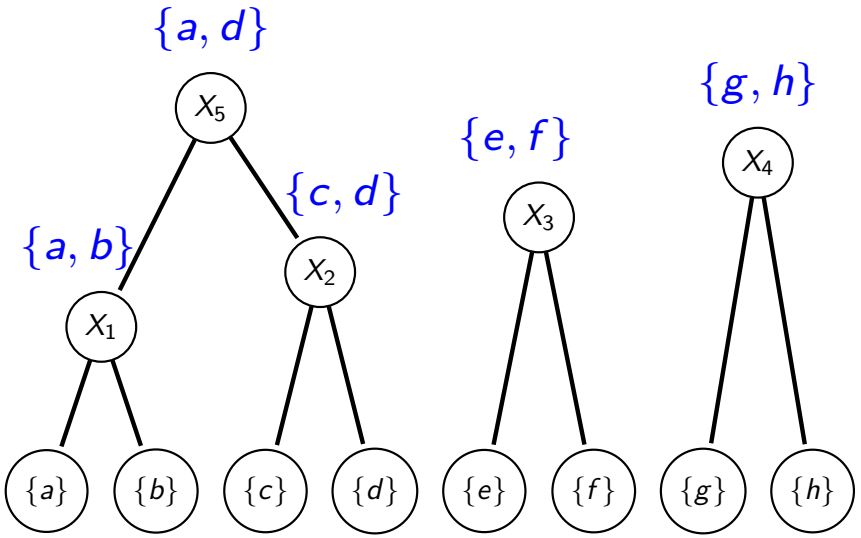
Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)

Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



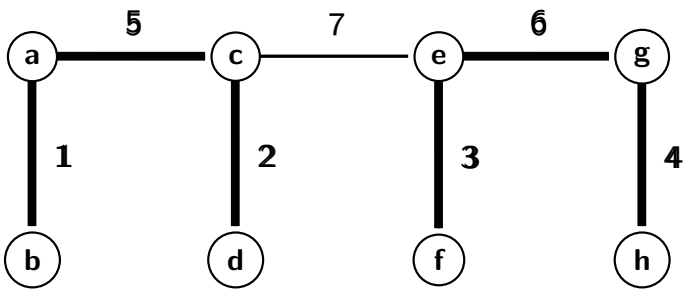
(G, w)



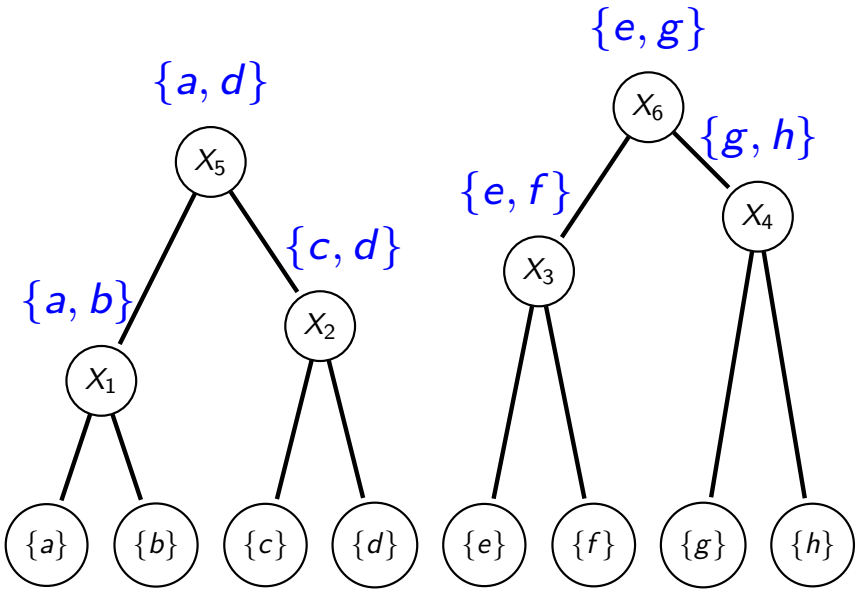
Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)

Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



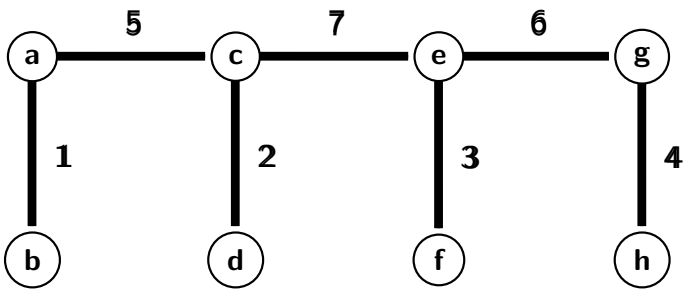
(G, w)



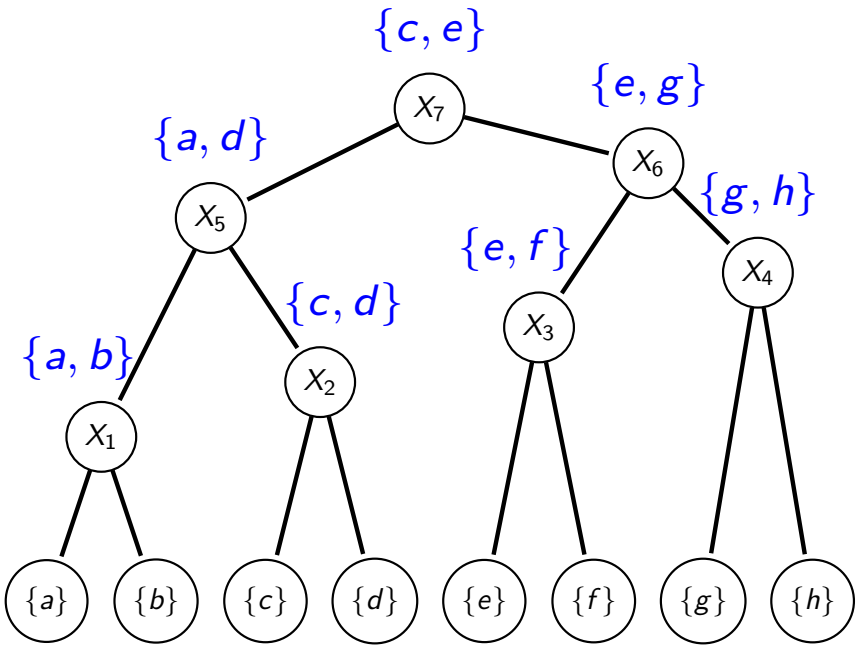
Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)

Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



(G, w)

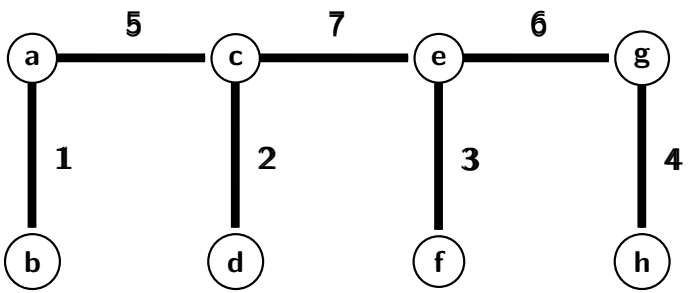


$\mathcal{B}$

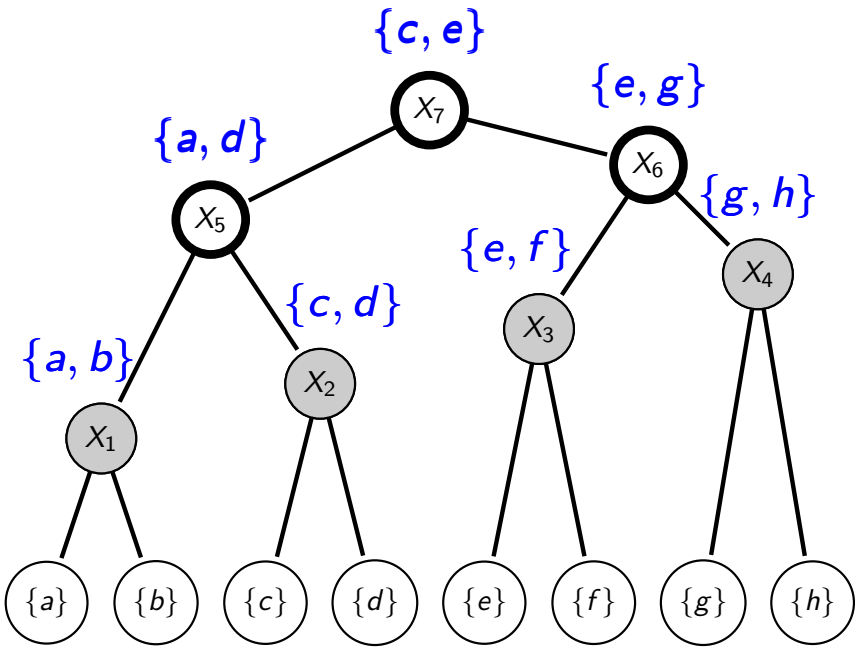
Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)

Binary partition hierarchy

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



(G, w)



Any non-leaf region R of $\mathcal{B}$ can be mapped to an edge u_R of G which is then called the building edge of R (shown in blue)
Minima (in grey) and watershed-cut edges (in bold)

Watershed-cut ordering

In a hierarchical watershed, only the ordering of watershed-cut edges are considered

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. $\Rightarrow$ watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$

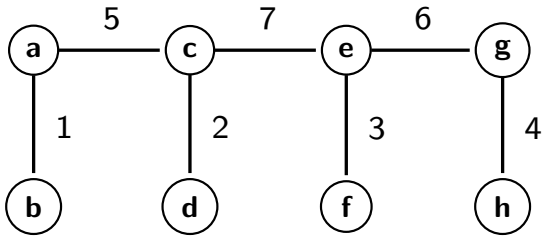
Definition

$$\eta_s(u) = \begin{cases} |\{v \in W \mid v \prec_s u\}| + 1 & u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where W is the set of watershed edges of w and where $v \prec u$ if $s(u) < s(v)$ or if $s(u) = s(v)$ and $w(u) < w(v)$

Watershed-cut ordering

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. $\Rightarrow$ watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



Weighted graph (G, w)

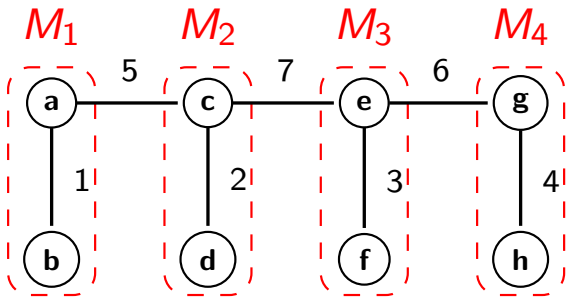
Definition

$$\eta_s(u) = \begin{cases} |\{v \in W \mid v \prec_s u\}| + 1 & u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where W is the set of watershed edges of w and where $v \prec u$ if $s(u) < s(v)$ or if $s(u) = s(v)$ and $w(u) < w(v)$

Watershed-cut ordering

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. $\Rightarrow$ watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



Minima of w (in red)

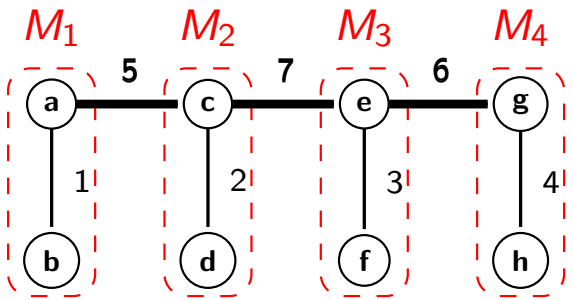
Definition

$$\eta_s(u) = \begin{cases} |\{v \in W \mid v \prec_s u\}| + 1 & u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where W is the set of watershed edges of w and where $v \prec u$ if $s(u) < s(v)$ or if $s(u) = s(v)$ and $w(u) < w(v)$

Watershed-cut ordering

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. $\Rightarrow$ watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



Minima of w (in red) and watershed-cut edges of w (in bold)

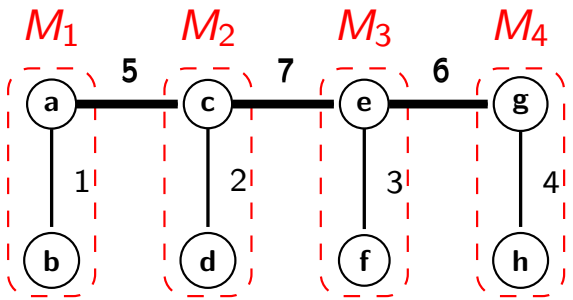
Definition

$$\eta_s(u) = \begin{cases} |\{v \in W \mid v \prec_s u\}| + 1 & u \in W \\ 0 & \text{otherwise} \end{cases}$$

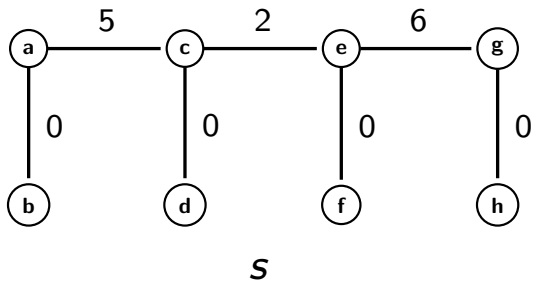
, where W is the set of watershed edges of w and where $v \prec u$ if $s(u) < s(v)$ or if $s(u) = s(v)$ and $w(u) < w(v)$

Watershed-cut ordering

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. $\Rightarrow$ watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



Minima of w (in red) and watershed-cut edges of w (in bold)



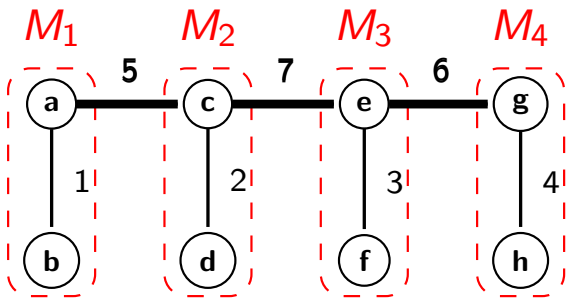
Definition

$$\eta_s(u) = \begin{cases} |\{v \in W \mid v \prec_s u\}| + 1 & u \in W \\ 0 & \text{otherwise} \end{cases}$$

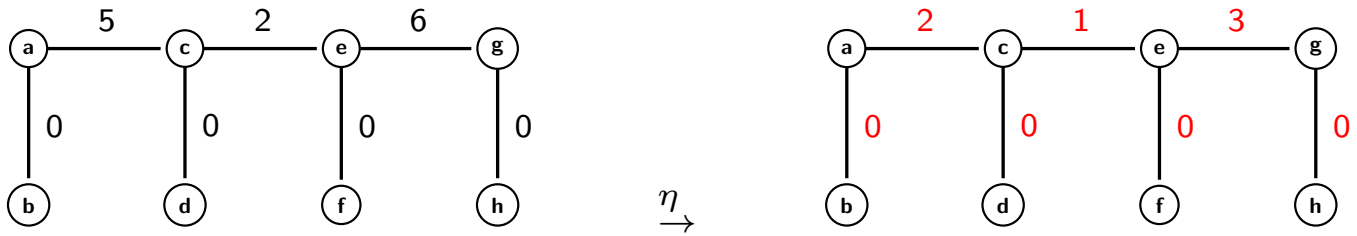
, where W is the set of watershed edges of w and where $v \prec u$ if $s(u) < s(v)$ or if $s(u) = s(v)$ and $w(u) < w(v)$

Watershed-cut ordering

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. $\Longrightarrow$ watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



Minima of w (in red) and watershed-cut edges of w (in bold)

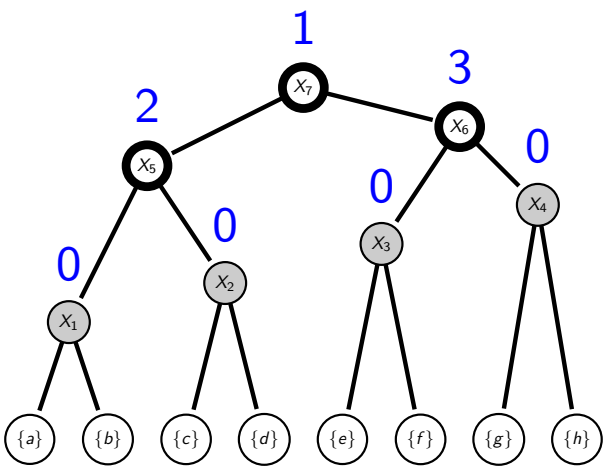


Definition

$$\eta_s(u) = \begin{cases} |\{v \in W \mid v \prec_s u\}| + 1 & u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where W is the set of watershed edges of w and where $v \prec u$ if $s(u) < s(v)$ or if $s(u) = s(v)$ and $w(u) < w(v)$

Extinction estimation



η_s

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. $\Rightarrow$ extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$

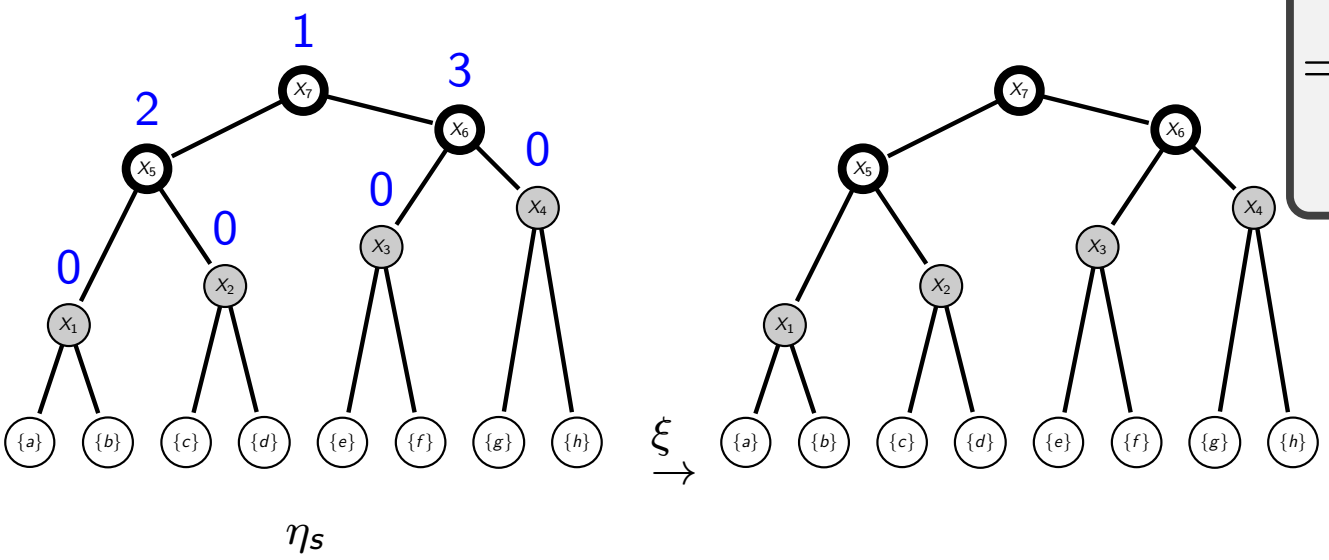
Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



η_s

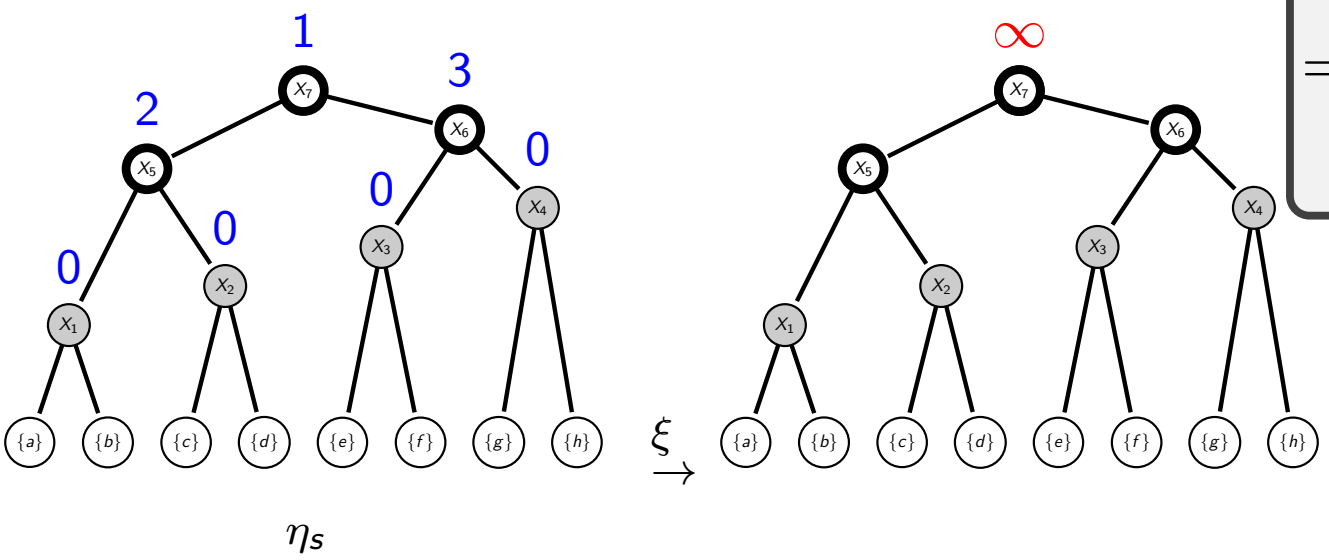
Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



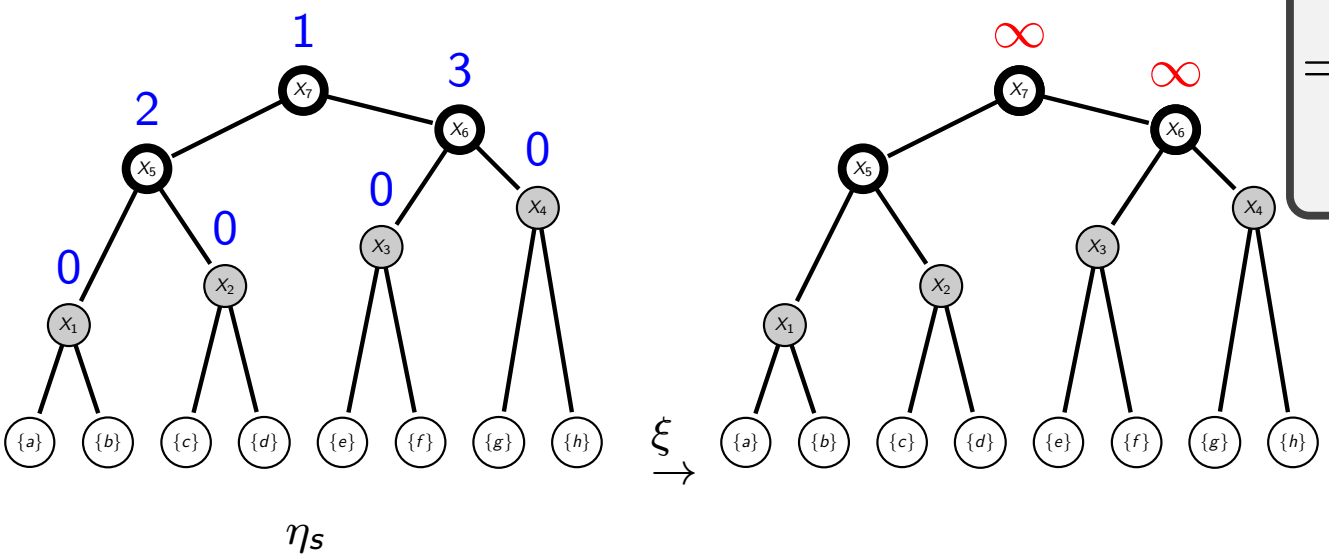
Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



η_s

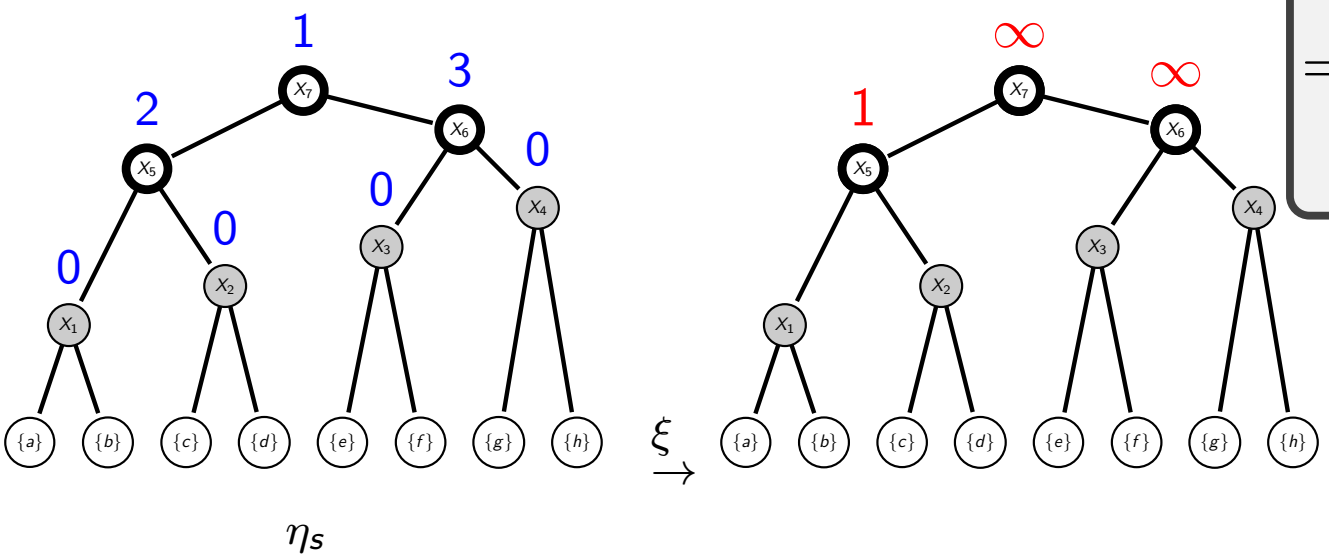
Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



η_s

Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

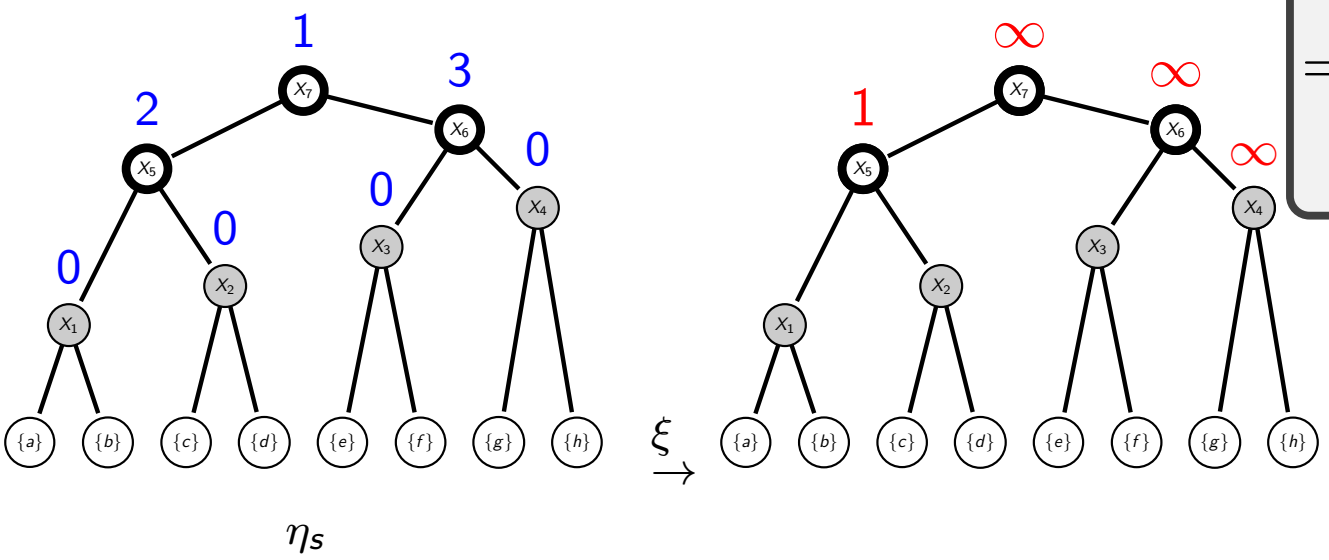
, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s

$\Rightarrow$ 4. extinction estimation ξ_s

5. persistence estimation π_s
6. verify if $\pi_s = s$



η_s

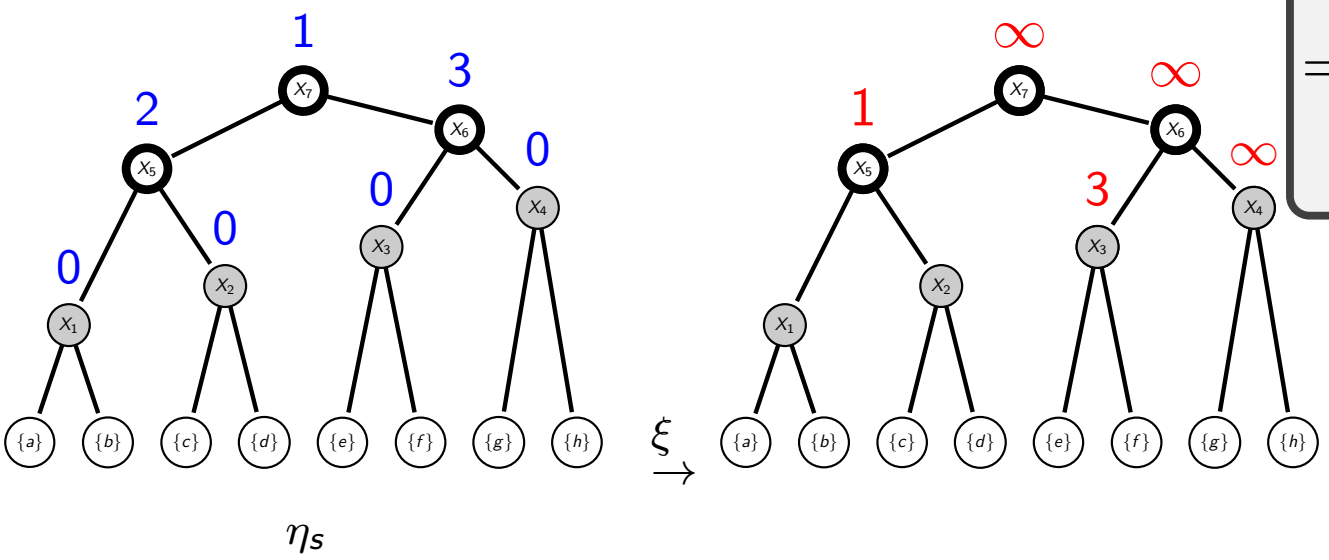
Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



η_s

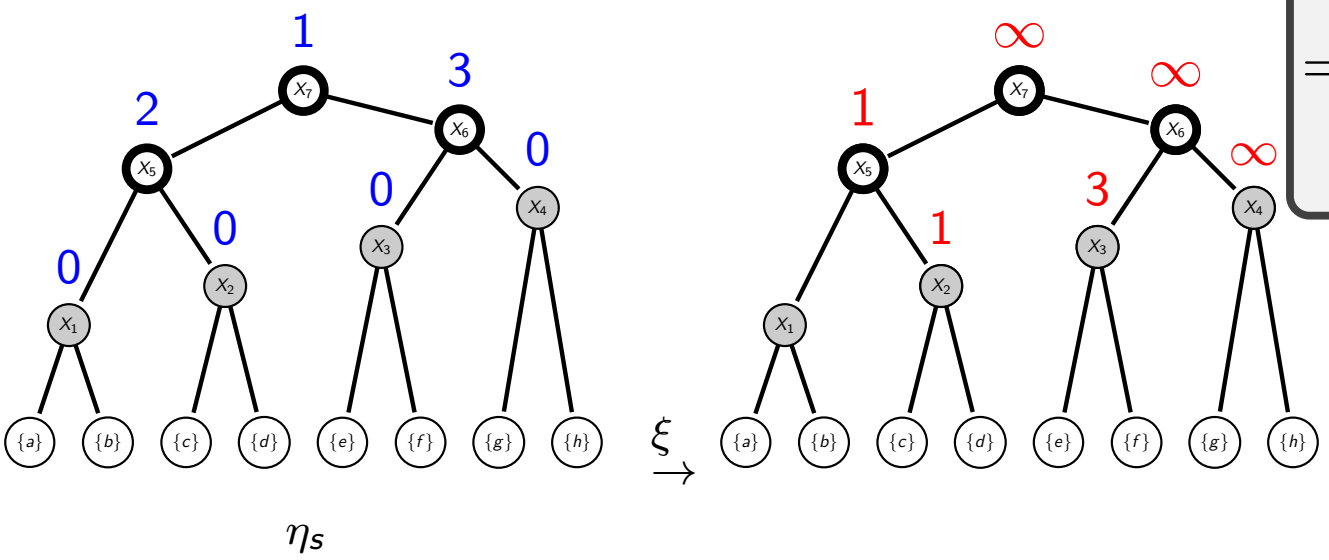
Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



η_s

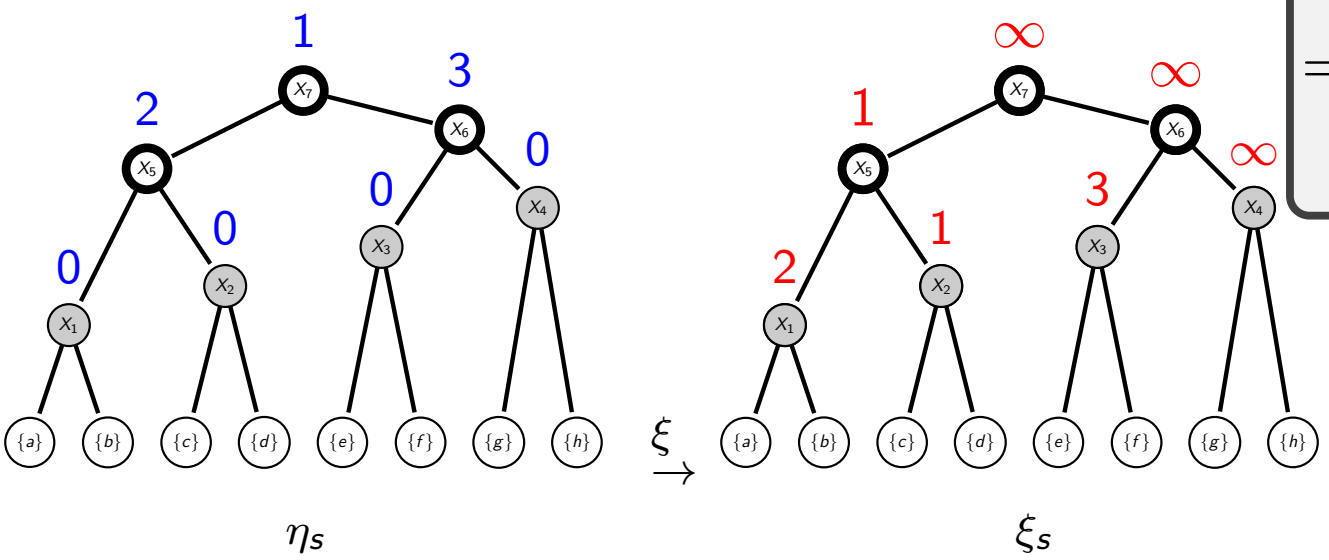
Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Extinction estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$



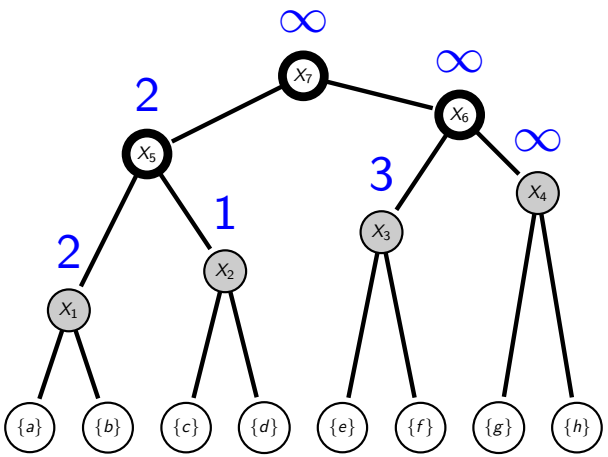
η_s $\qquad$ ξ_s

Definition

$$\xi_s(u) = \begin{cases} \infty & \text{if } R_u = V \\ \xi_s(parent(u)) & \bullet \text{ if the sibling of } R_u \text{ is a leaf or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) > \vee^{\eta_s}(sibling(u)) \text{ or} \\ & \bullet \text{ if } \vee^{\eta_s}(u) = \vee^{\eta_s}(sibling(u)) \text{ and} \\ & w(sibling(u)) < w(u) \\ \eta_s(parent(u)) & \text{otherwise} \end{cases}$$

, where $\vee^{\eta_s}(u) = max\{\eta_s(v) \mid v \in E, R_v \subseteq R_u\}$

Persistence estimation



ξ_s

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. $\Rightarrow$ persistence estimation π_s
6. verify if $\pi_s = s$

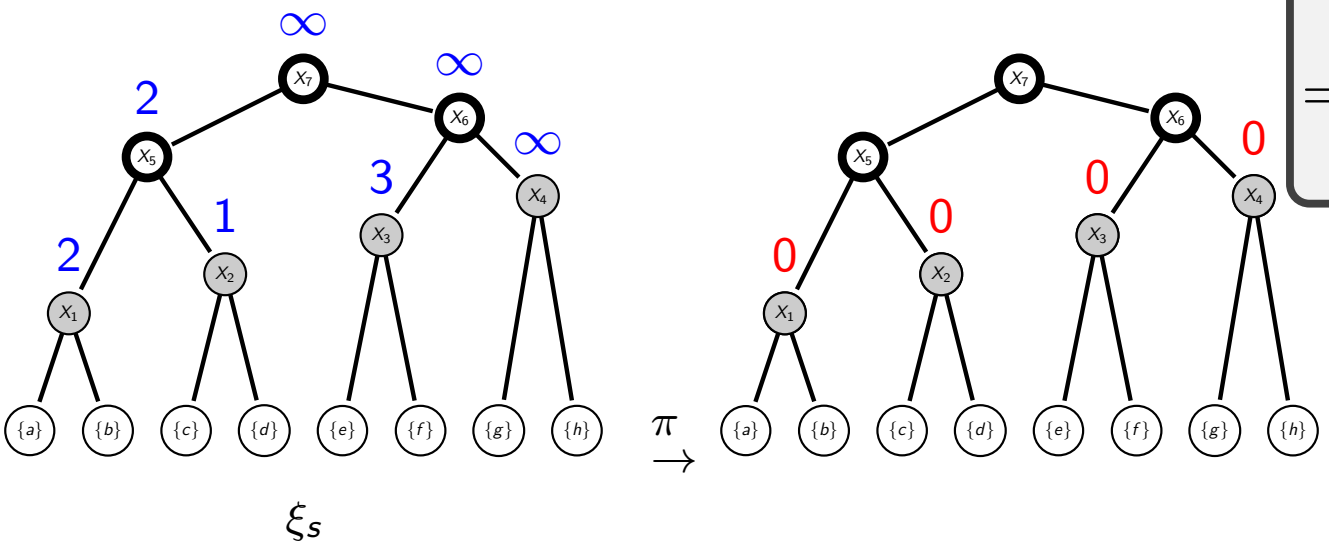
Definition

$$\pi_s(u) = \begin{cases} min\{\vee^{\xi_s}(v) \mid v \in E,\ parent(v) = u\} & \text{if } u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where $\vee^{\xi_s}(u) = max\{\xi_s(v) \mid v \in E, R_v \subseteq R_u\}$ and where W is the set of watershed edges of w.

Persistence estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. $\Longrightarrow$ persistence estimation π_s
6. verify if $\pi_s = s$



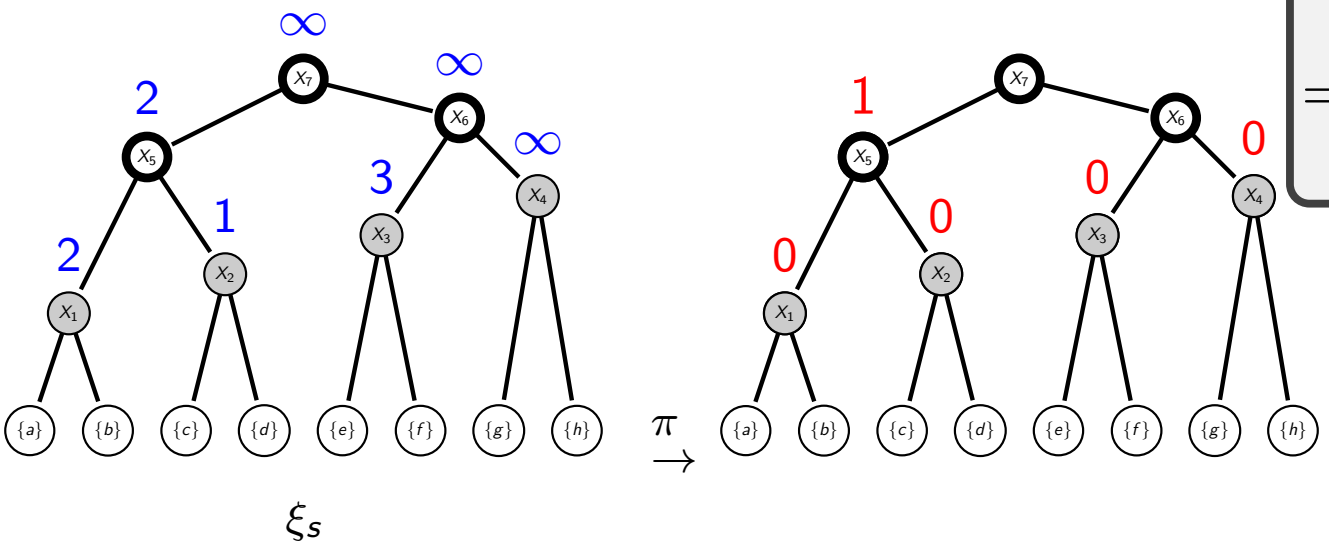
Definition

$$\pi_s(u) = \begin{cases} min\{\vee^{\xi_s}(v) \mid v \in E,\ parent(v) = u\} & \text{if } u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where $\vee^{\xi_s}(u) = max\{\xi_s(v) \mid v \in E, R_v \subseteq R_u\}$ and where W is the set of watershed edges of w.

Persistence estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. $\Longrightarrow$ persistence estimation π_s
6. verify if $\pi_s = s$



ξ_s

Definition

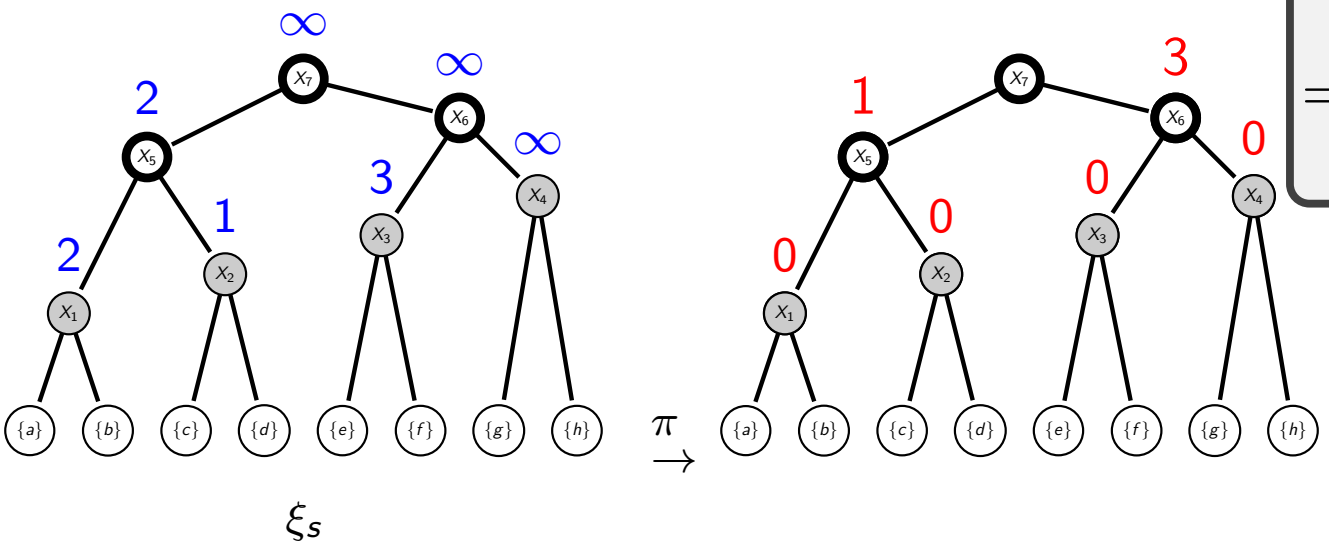
$$\pi_s(u) = \begin{cases} min\{\vee^{\xi_s}(v) \mid v \in E,\ parent(v) = u\} & \text{if } u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where $\vee^{\xi_s}(u) = max\{\xi_s(v) \mid v \in E, R_v \subseteq R_u\}$ and where W is the set of watershed edges of w.

Persistence estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s
5. persistence estimation π_s
6. verify if $\pi_s = s$

$\Longrightarrow$



Definition

$$\pi_s(u) = \begin{cases} min\{\vee^{\xi_s}(v) \mid v \in E,\ parent(v) = u\} & \text{if } u \in W \\ 0 & \text{otherwise} \end{cases}$$

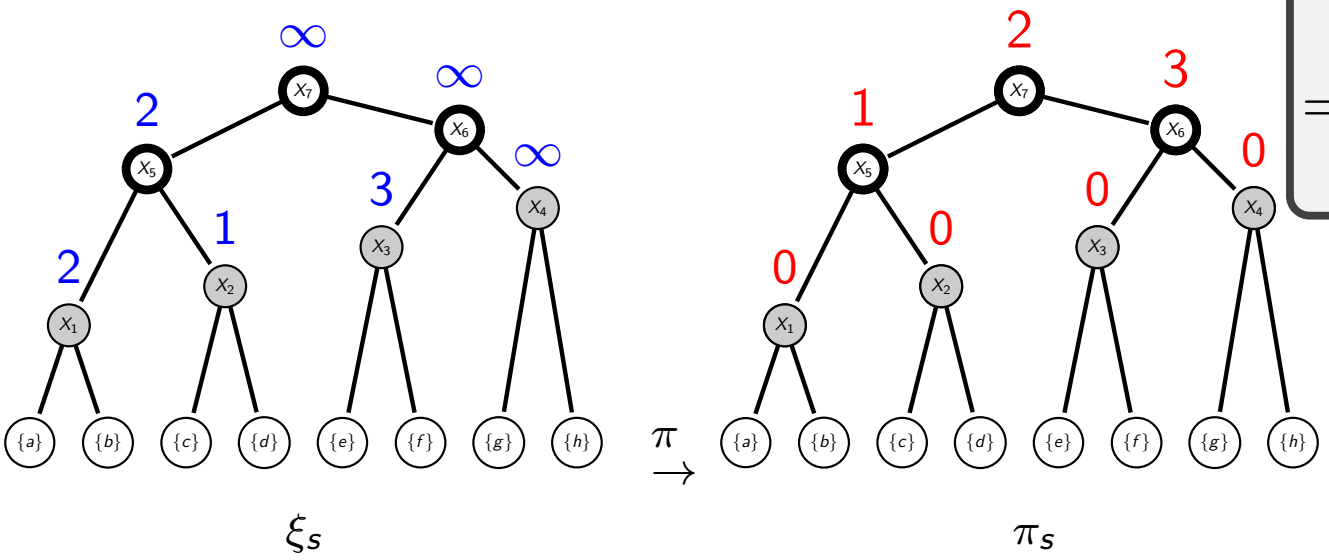
, where $\vee^{\xi_s}(u) = max\{\xi_s(v) \mid v \in E, R_v \subseteq R_u\}$ and where W is the set of watershed edges of w.

Persistence estimation

Recognition algorithm:
1. binary partition hierarchy $\mathcal{B}$
2. saliecy map s
3. watershed-cut ordering η_s
4. extinction estimation ξ_s

$\Longrightarrow$ 5. persistence estimation π_s

6. verify if $\pi_s = s$



ξ_s π_s

Definition

$$\pi_s(u) = \begin{cases} min\{\vee^{\xi_s}(v) \mid v \in E,\ parent(v) = u\} & \text{if } u \in W \\ 0 & \text{otherwise} \end{cases}$$

, where $\vee^{\xi_s}(u) = max\{\xi_s(v) \mid v \in E, R_v \subseteq R_u\}$ and where W is the set of watershed edges of w.

Watersheding

Let $\mathcal{H}$ be a hierarchy and let s be the saliency map of $\mathcal{H}$.

Definition

The binary partition hierarchy of π_s is called the *watersheding of* $\mathcal{H}$

Watersheding

Let $\mathcal{H}$ be a hierarchy and let s be the saliency map of $\mathcal{H}$.

Definition

The binary partition hierarchy of π_s is called the *watersheding of* $\mathcal{H}$

Theorem

The three following statements are equivalent:

- $\mathcal{H}$ *is a hierarchical watershed*
- $\pi_s = s$
- $\mathcal{H}$ *is the watersheding of* $\mathcal{H}$

Watersheding

Let $\mathcal{H}$ be a hierarchy and let s be the saliency map of $\mathcal{H}$.

Definition

The binary partition hierarchy of π_s is called the *watersheding of* $\mathcal{H}$

Theorem

The three following statements are equivalent:

- $\mathcal{H}$ *is a hierarchical watershed*
- $\pi_s = s$
- $\mathcal{H}$ *is the watersheding of* $\mathcal{H}$

Consequence: The proposed algorithm recognizes hierarchical watershed

Outlines

Conclusion and perspectives

Summary

- *Introduction of watersheding transform*
- *Problem of recognition of hierarchical watersheds solved by the watersheding transform*
- *Hierarchical watersheds are the fixed points of the watersheding transform*
- *Quasi-linear algorithm to the watersheding proposed*

Perspectives

- *Answer the question: does the combination of hierarchical watersheds always result in a hierarchical watershed?*
- *Extension to arbitrary weighted graphs*
- *Frequency study of hierarchical watersheds*
- *Practical interest of the watersheding transform*