Component-trees: Structural and spectral extensions

Nicolas Passat

Université de Reims Champagne-Ardenne, Reims, France
CReSTIC, EA 3804

Noisy-le-Grand, 25 March 2019

Content

Content

Sets and partitions

Set and cardinality

A set Ω is composed of elements x

$\rightarrow$ Many tasks consist of gathering elements x within Ω

In other words, we aim at building partitions of Ω

High cardinalities

Problem: Ω can be “big”, i.e. $|\Omega| \gg 1$
Especially true in imaging

- Photography: 10^5–10^6 pixels
- Remote sensing: 10^6–10^8 pixels
- Medical imaging: 10^7–10^9 voxels

$\rightarrow$ Combinatorial explosion of the space of partitions

We need a way to reduce these huge spaces, without losing relevant information. . .

Here come the hierarchies. . .

What is a hierarchy?

In general, a hierarchy is a tree structure (rooted, connected, acyclic graph)

- Nodes = subsets of Ω
  - Root = Ω
  - Leaves = "minimal" subsets of Ω
- Parent-child relation = inclusion between subsets
- Brother nodes $\rightarrow$ non-overlapping

Cutting trees

A cut in a tree = subset of nodes without parent-child relations

- Cut $\sim$ partition (either total or partial) $\sim$ segmentation

$\rightarrow$ Trees are a good tools for defining / choosing / computing partitions

Usefulness of hierarchies in imaging

What can be done with a tree modeling an image

- Filtering
- Segmentation
- Simplification
- Clustering
- Visualization
- ad lib.

$\rightarrow$ Anything that can be interpreted as a partitioning problem

Different kinds of hierarchies

Many trees (non exhaustive...)

- Quadtrees, octrees
- (Binary) partition trees
- Component-trees (max-trees, min-trees)
- α-trees
- Trees of shapes
- Watershed trees

All trees are not the same

- Partial vs. total partitions
- Intrinsic vs. extrinsic
- Information lossless or not

Among all of them, we focus (mainly) on the component-tree

Content

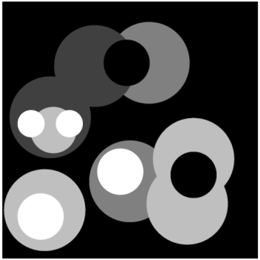
Informal definition

A first definition

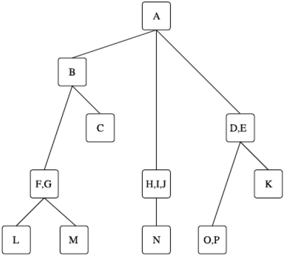
The component-tree of a grey level image is the tree modelling the inclusion of all the binary connected components obtained at all the threshold sets of the image.

Important terms

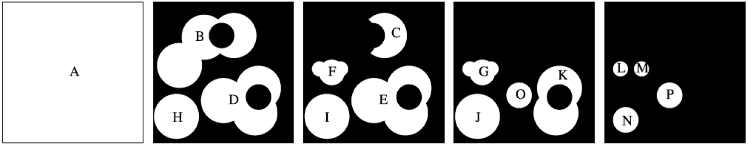
- "grey level" $\rightarrow$ total order on values
- "connected components" $\rightarrow$ topological structure on points
- "modelling [...] all the threshold sets" $\rightarrow$ lossless data structure



(a) I



(b) $\mathfrak{T}$



(c) $\lambda_0(I)$ (d) $\lambda_1(I)$ (e) $\lambda_2(I)$ (f) $\lambda_3(I)$ (g) $\lambda_4(I)$

Image

Image

$$\left| \begin{array}{cccc} I & : & \Omega & \rightarrow & V \\ & & x & \mapsto & I(x) = v \end{array} \right.$$

where

- Ω: a (discrete, finite) set of points
- V: a set of values

Additional information

We need a structure for the space

- an adjacency (irreflexive, symmetric) relation $\frown$
  $\rightarrow (\Omega, \frown)$ is an non-directed graph

and a structure for the values

- a total order relation $\leqslant$
  $\rightarrow (V, \leqslant)$ is a totally ordered set

Basic tools on graphs

$(\Omega, \frown)$ is an non-directed graph

From adjacency to connectedness

- $X \subseteq \Omega$: a subset of points
- $\leftrightarrow$: the reflexive–transitive closure of $\frown$ on X

This equivalence relation $\leftrightarrow$ is the connectedness relation on X

Connected components

- $X/\leftrightarrow$: the set of equivalence classes of X, noted $\mathcal{C}[X]$

Remark: we assume $\mathcal{C}[\Omega] = \{\Omega\}$ (i.e. Ω is connected)

Image decomposition

Thresholding

$$\left|\begin{array}{lllll} \lambda_v & : & V^\Omega & \to & 2^\Omega \\ & & I & \mapsto & \{x \in \Omega \mid I(x) \geqslant v\} \end{array}\right.$$

Cylinder functions

$$\left|\begin{array}{lllll} C_{(X,v)} & : & \Omega & \to & V \\ & & x & \mapsto & \begin{cases} v & \text{if } x \in X \\ \perp & \text{otherwise} \end{cases} \end{array}\right.$$

Image (de)composition

$$I = \bigvee_{v \in V}^{\leq} \; \bigvee_{X \in \mathcal{C}[\lambda_v(I)]}^{\leq} C_{(X,v)}$$

And now, the definition of a component-tree

Image

- Image $I : \Omega \to V$
- Support of the image: $(\Omega, \frown)$
- Value space of the image $(V, \leqslant)$

Component-tree

- Nodes of the tree

$$\Psi = \bigcup_{v \in V} \mathcal{C}[\lambda_v(I)]$$

- $\Psi \in 2^\Omega$ is (partially) ordered by $\subseteq$

The component-tree $\mathfrak{T}$ is the Hasse diagram of the partially ordered set $(\Psi, \subseteq)$

Advantages and limitations

Nice properties of component-trees

- Image lossless model
- Parameter-free data structure
- (Nearly) deterministic
- Low size: $|\Psi| \leq |\Omega|$
- Low cost computation: $\mathcal{O}(|\Omega| \log |\Omega|)$
- Low cost processing: many (quasi-)linear strategies

Meta-parameters

- Total order relation: either $\leqslant$ or $\geqslant$
- Adjacency (irreflexive, symmetric) relation $\frown$

Advantages and limitations

Range of validity of component-trees

- Grey-level images
- Structures of interest = either local maxima or local minima

Limitations

- $\leqslant$ is total (grey-levels)
- $\frown$ is symmetric ($\cap \; \Rightarrow \subseteq$)

Question

What happens for component-trees if we try to go beyond these two limitations?

- Component-graphs: a spectral extension of component-trees
- Asymmetric hierarchies: a structural extension of component-trees

Content

Relaxing ordering constraints

What happens if we relax the ordering constraints?

Total ordering $\rightarrow$ partial ordering
Impact on:

- intersection / inclusion relations ($\cap \not\Rightarrow \subseteq$)
- hierarchical structure (tree $\rightarrow$ directed acyclic graph)

The "component-tree" is no longer a tree...

This may modify our way of thinking for

- passing from the initial image to the hierarchy
- manipulating the hierarchy
- passing from a subset of the hierarchy to the final image

We need to study the structural properties of these new data structures

From component-trees to component-graphs

First idea: keeping the same paradigm?

Component-tree = Hasse diagram of $(\Psi, \subseteq)$

$\rightarrow$ Do the same for component-graphs?

Indeed, Ψ is well defined (since thresholding is well-defined)

$$\left| \begin{array}{cccc} \lambda_v & : & V^\Omega & \rightarrow & 2^\Omega \\ & & I & \mapsto & \{x \in \Omega \mid I(x) \geqslant v\} \end{array} \right.$$

But...

When $\leqslant$ is total we have

$$\subseteq \Rightarrow \leqslant$$

whereas when $\leqslant$ is partial, this is not always true

$\rightarrow$ We need a more informative relation on the connected components

From component-trees to component-graphs

Reminder: connected components

$$\Psi = \bigcup_{v \in V} \mathcal{C}[\lambda_v(I)]$$

$$X \in \Psi \to X \subseteq \Omega$$

Idea: explicitly embedding spectral information in connected components

Valued connected components

Θ : set of all the valued connected components of I

$$\Theta = \bigcup_{v \in V} \mathcal{C}[\lambda_v(I)] \times \{v\}$$

$$K = (X, v) \in \Theta \to (X \subseteq \Omega) \wedge (v \in V)$$

Choosing an order on Θ

Which order on Θ?

Now, $K \in \Theta$ carries information on

- Ω ($\rightarrow \subseteq$)
- V ($\rightarrow \leqslant$)

so that we can build a mixed order from $\subseteq$ and $\leqslant$

Two possibilites of order

- Strong order: $(X_1, v_1) \trianglelefteq_s (X_2, v_2) \Leftrightarrow X_1 \subseteq X_2 \wedge v_2 \leqslant v_1$
- Weak order: $(X_1, v_1) \trianglelefteq_w (X_2, v_2) \Leftrightarrow (X_1 \subset X_2) \vee (X_1 = X_2 \wedge v_2 \leqslant v_1)$

Strong implies weak

$$K \trianglelefteq_s K' \Rightarrow K \trianglelefteq_w K'$$

The counterpart is false, in general

Component-graph: definition

Reminder: component-tree

Component-tree $\mathfrak{T}$ = Hasse diagram of $(\Psi, \subseteq)$

Two kinds of component-graphs

The component-graph $\mathfrak{G}$ is Hasse diagram of

- $(\Theta, \trianglelefteq_s) \rightarrow$ strong component-graph
- $(\Theta, \trianglelefteq_w) \rightarrow$ weak component-graph

Three kinds of (sub)sets of nodes

- Θ
- $\dot{\Theta} = \bigcup_{X \in \Psi} \{X\} \times \nabla^{\leqslant}\{v \mid X \in \mathcal{C}[\lambda_v(I)]\}$
- $\ddot{\Theta} = \bigcap \left\{\Theta' \subseteq \Theta \mid I = \bigvee^{\leq}_{K \in \Theta'} C_K\right\}$

Basic properties of component-graphs

Inclusion of subsets of nodes

$$\ddot{\Theta} \subseteq \dot{\Theta} \subseteq \Theta$$

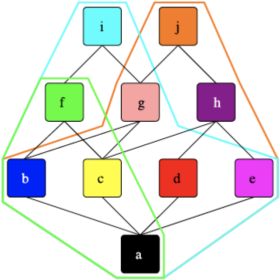
Equality of roots

$$\overset{\trianglelefteq}{\bigvee} \Theta = \overset{\trianglelefteq}{\bigvee} \dot{\Theta} = \overset{\trianglelefteq}{\bigvee} \ddot{\Theta} = (\Omega, \perp)$$

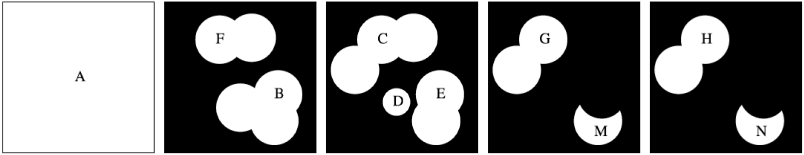
Equality of leaves

$$\overset{\trianglelefteq}{\bigtriangleup} \Theta = \overset{\trianglelefteq}{\bigtriangleup} \dot{\Theta} = \overset{\trianglelefteq}{\bigtriangleup} \ddot{\Theta}$$

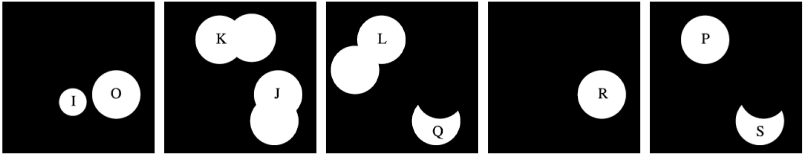
(a) I



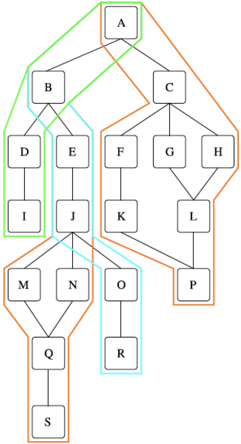
(b) $(V, \prec)$



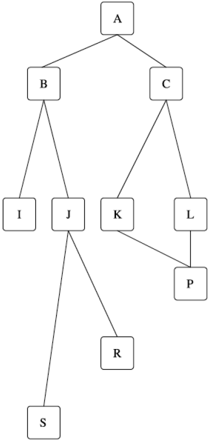
(c) $\lambda_a(I)$ (d) $\lambda_b(I)$ (e) $\lambda_c(I)$ (f) $\lambda_d(I)$ (g) $\lambda_e(I)$



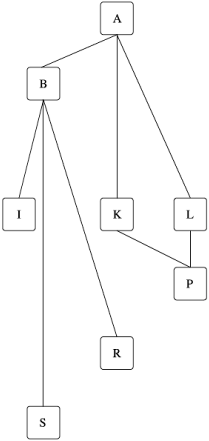
(h) $\lambda_f(I)$ (i) $\lambda_g(I)$ (j) $\lambda_h(I)$ (k) $\lambda_i(I)$ (l) $\lambda_j(I)$



(m) $\mathfrak{G}$



(n) $\dot{\mathfrak{G}}$



(o) $\ddot{\mathfrak{G}}$

Component-trees vs. component-graphs

Component-graphs extend component-trees

If $\leqslant$ is a total order (i.e. if I is a grey-level image), then

- $\dot{\mathfrak{G}}_s = \ddot{\mathfrak{G}}_s$
- $\mathfrak{T}$ is isomorphic to $\dot{\mathfrak{G}}_s = \ddot{\mathfrak{G}}_s = \dot{\mathfrak{G}}_w = \ddot{\mathfrak{G}}_w$
- $\mathfrak{T}$ is isomorphic to $\mathfrak{G}_s = \mathfrak{G}_w$ (modulo an equivalence relation on Θ induced by $\subseteq$)

In other words

The component-graphs are relevant (spectral) extensions of the component-tree

Component-graphs: spot the difference!

Links bteween $\mathfrak{G}$ and $(V, \prec)$

$K = (X, v) \in \Delta^{\trianglelefteq}\Theta$ a leaf of $\mathfrak{G}$

$$\left|\begin{array}{cccc} \sigma & : & K^{\uparrow} & \to & v^{\downarrow} \\ & & (Y, w) & \mapsto & w \end{array}\right.$$

is a bijection between $K^{\uparrow}$ and $v^{\downarrow}$

But...

The case of weak component-graphs

$\sigma^{-1} : v^{\downarrow} \to K^{\uparrow}$ induces a homomorphism from $(v^{\downarrow}, \geqslant)$ to $(K^{\uparrow}, \trianglelefteq_w)$

For any $K_1 = \sigma^{-1}(v_1)$, $K_2 = \sigma^{-1}(v_2)$

$$v_1 \geqslant v_2 \Rightarrow K_1 \trianglelefteq_w K_2$$

Component-graphs: spot the difference!

Links bteween $\mathfrak{G}$ and $(V, \prec)$

$K = (X, v) \in \Delta^{\trianglelefteq}\Theta$ a leaf of $\mathfrak{G}$

$$\left| \begin{array}{ccccc} \sigma & : & K^{\uparrow} & \rightarrow & v^{\downarrow} \\ & & (Y, w) & \mapsto & w \end{array} \right.$$

is a bijection between $K^{\uparrow}$ and $v^{\downarrow}$

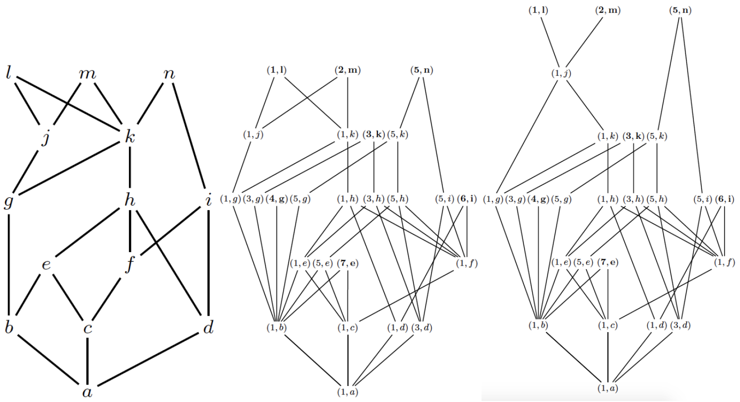
whereas...

The case of strong component-graphs

$\sigma^{-1} : v^{\downarrow} \rightarrow K^{\uparrow}$ induces an isomorphism between $(v^{\downarrow}, \geqslant)$ and $(K^{\uparrow}, \trianglelefteq_w)$

For any $K_1 = \sigma^{-1}(v_1)$, $K_2 = \sigma^{-1}(v_2)$

$$v_1 \geqslant v_2 \Leftrightarrow K_1 \trianglelefteq_s K_2$$


l m n
j k
g h i
e f
b c d
a
(1, l) (2, m) (5, n)
(1, j) (1, k) (3, k) (5, k)
(1, g) (3, g) (4, g) (5, g) (1, h) (3, h) (5, h) (5, i) (6, i)
(1, e) (5, e) (7, e) (1, f)
(1, b) (1, c) (1, d) (3, d)
(1, a)
(1, l) (2, m) (5, n)
(1, j)
(1, k) (3, k) (5, k)
(1, g) (3, g) (4, g) (5, g) (1, h) (3, h) (5, h) (5, i) (6, i)
(1, e) (5, e) (7, e) (1, f)
(1, b) (1, c) (1, d) (3, d)
(1, a)

Component-graphs: spot the difference!

In other words, strong component-graphs are "more regular" wrt the spectral information that weak component-graphs
With exceptions. . .

Case of $\ddot{\mathfrak{G}}$

$$\ddot{\mathfrak{G}}_s = \ddot{\mathfrak{G}}_w$$

in other words, if $K, K' \in \ddot{\Theta}$ then

$$K \mathrel{\ddot{\blacktriangleleft}_s} K' \Leftrightarrow K \mathrel{\ddot{\blacktriangleleft}_w} K'$$

Case of $\dot{\mathfrak{G}}$ for lattices

If $(V, \leqslant)$ is a lattice, then

$$\dot{\mathfrak{G}}_s = \dot{\mathfrak{G}}_w$$

in other words, if $K, K' \in \dot{\Theta}$ then

$$K \mathrel{\dot{\blacktriangleleft}_s} K' \Leftrightarrow K \mathrel{\dot{\blacktriangleleft}_w} K'$$

Back to the component-tree/-graph image processing paradigm

Space of images $\longleftrightarrow$ Space of component-trees/-graphs

Lossless image model: component-tree

$$I = \bigvee_{X \in \Psi}^{\leq} C_{(X, v_X)}$$

with $v_X = \arg\max\{v \in V \mid X \in \mathcal{C}[\lambda_v(I)]\}$ (implicit!)

Lossless image model: component-graph

$$I = \bigvee_{K \in \Theta}^{\leq} C_K = \bigvee_{v \in V}^{\leq} \bigvee_{X \in \mathcal{C}[\lambda_v(I)]}^{\leq} C_{(X, v)}$$

Back to the component-tree/-graph image processing paradigm

Standard antiextensive filtering framework

$$\begin{tikzcd} I \arrow[r, "(i)"] \arrow[d] & \mathfrak{G}[\Theta] \arrow[d, "(ii)"] \\ \widehat{I} \leq I & \widehat{\mathfrak{G}}[\widehat{\Theta} \subseteq \Theta] \arrow[l, "(iii)"'] \end{tikzcd}$$

Three steps

(i) Component-tree/-graph construction

(ii) Component-tree/-graph "pruning"

(iii) Sub-image reconstruction

Step (i): component-graph construction

Reminder on component-trees

- Space complexity: $\mathcal{O}(|\Psi|)$ with $|\Psi| \leq |\Omega|$
- Time complexity (construction): $\mathcal{O}(|\Omega| \log |\Omega|)$

Various efficient algorithms for building component-trees

Complexity of component-graphs

- Space complexity
  - $\mathfrak{G}$, $\dot{\mathfrak{G}}$: $\mathcal{O}(|\Omega|.|V|)$
  - $\ddot{\mathfrak{G}}$: $\mathcal{O}(|\Omega|)$
- Time complexity for construction $\geq$ space complexity...

So, we have to be careful when building component-graphs!

Step (i): component-graph construction

Reminder

For each leaf $K = (X, v) \in \Theta$, there is an isomorphism between $(v^{\downarrow}, \geqslant)$ and $(K^{\uparrow}, \trianglelefteq_w)$

Main ideas (for the strong component-graph)

Between the root and each leaf, we have a **copy** of the Hasse diagram of (a part of) $(V, \leqslant)$

$\rightarrow$ $\mathfrak{G}$ is highly similar to $(V, \leqslant)$

$\rightarrow$ $\mathfrak{G}$ is high redundant

The structure of the component-graph is fully described by the "bifurcations" between these parts of $(V, \leqslant)$

$\rightarrow$ $\mathfrak{G}$ can be fully modeled by analyzing its topological structure

Component-graph construction (and storage)

Strategy based on these paradigms:

- flooding + connectedness analysis
- time complexity: $\mathcal{O}(|\Omega|^{\alpha})$ with $2 \leq \alpha \leq 3$

Step (ii): component-tree/-graph "pruning"

Attribute-based node selection

All the standard strategies remain valid in case of

- monotonic attributes (acyclic graph)
- non-monotonic attributes: min, max, direct policies are still tractable

with still linear time costs

Optimal cut computation

Tree-based techniques ("divide and conquer") are no longer valid, but many graph-based strategies still are

- min-cut / max-flow
- random walkers
- optimal path finding
- minimum spanning tree
- etc.

Step (iii): sub-image reconstruction

Reminder

$$I = \bigvee_{K \in \Theta}^{\leq} C_K = \bigvee_{v \in V}^{\leq} \bigvee_{X \in \mathcal{C}[\lambda_v(I)]}^{\leq} C_{(X,v)}$$

Binary image reconstruction from $\widehat{\Theta} \subseteq \Theta$

$$I = \bigcup_{(X,v) \in \widehat{\Theta}} X$$

Non-binary image reconstruction from $\widehat{\Theta} \subseteq \Theta$

$$I = \bigvee_{K \in \widehat{\Theta}}^{\leq} C_K$$

$\rightarrow$ Ill-posed, because for any $(X_1, v_1), (X_2, v_2) \in \Theta$ we can have $X_1 \cap X_2 \neq \emptyset$ whereas v_1 and v_2 are non-comparable

Step (iii): sub-image reconstruction

Ad-hoc strategies for non-binary image reconstruction

$$I = \bigvee_{K \in \widehat{\Theta}}^{\leq} C_K$$

- multivoke image reconstruction: switch from $I : \Omega \to V$ to $I : \Omega \to 2^V$
- $\vee$ or $\wedge$ reconstructions (possibly non-deterministic)
- arbitrary choice between candidate values in ill-posed areas
- adding / removal of nodes until eliminating ill-posed areas
- etc.

$\rightarrow$ In general, the choice of a relevant strategy is application-dependent

Content

Relaxing adjacency constraints

What happens if we relax the adjacency constraints?

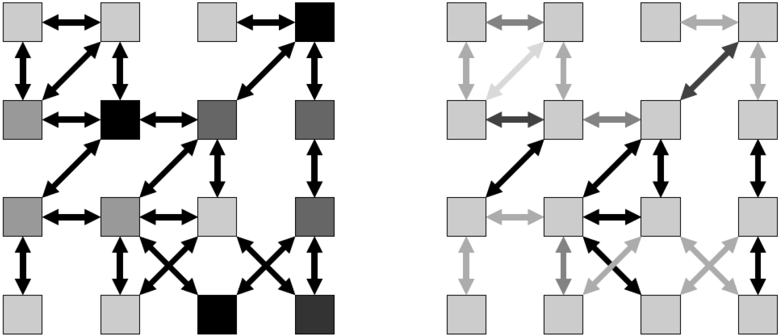
Adjacency (irreflexive, symmetric) $\rightarrow$ Directed adjacency (irreflexive)
Impact on

- connectedness ($\rightarrow$ directed connectedness)
- intersection / inclusion relations ($\cap \not\Rightarrow \subseteq$)
- hierarchy structure (tree $\rightarrow$ directed acyclic graph)

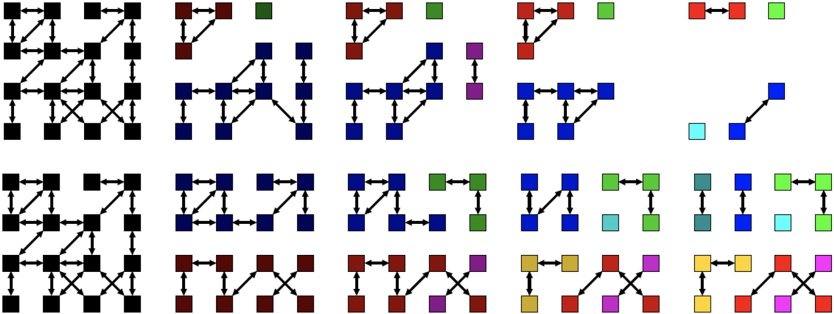
(Once again) the "component-tree" is no longer a tree...

But maybe, not so far from being a tree...

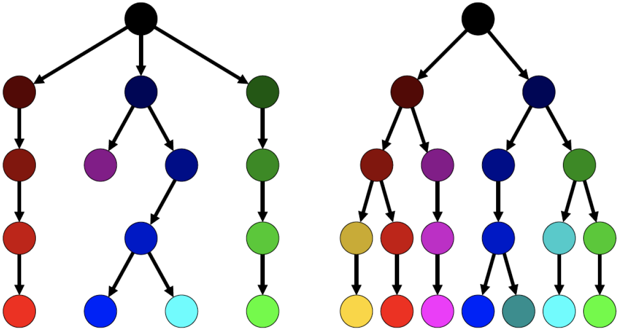
Back to symmetric adjacencies



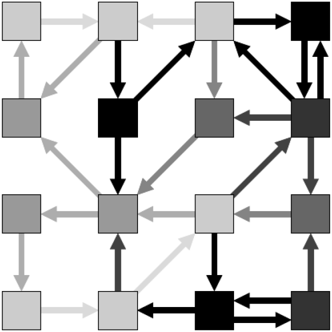
Back to symmetric adjacencies



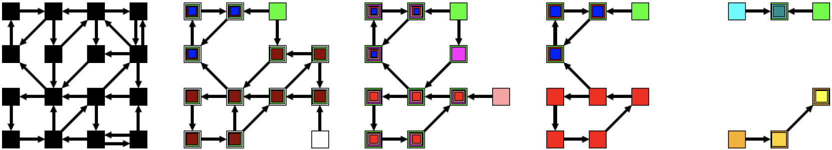
Back to symmetric adjacencies



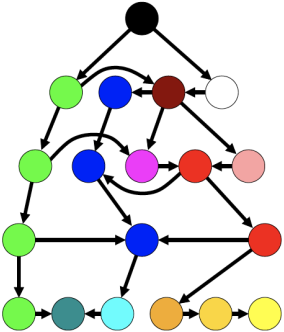
Now, non-symmetric adjacencies



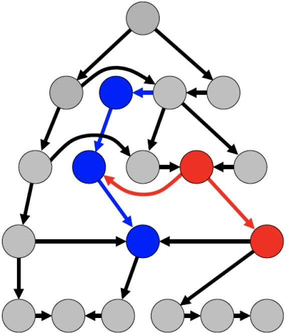
Now, non-symmetric adjacencies

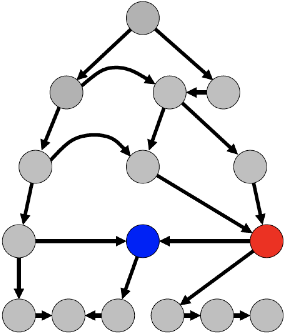


Directed hierarchies

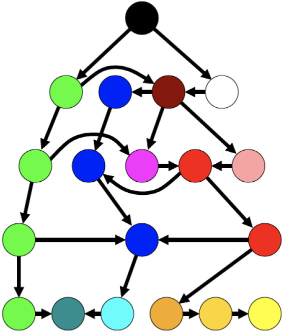


Directed hierarchies

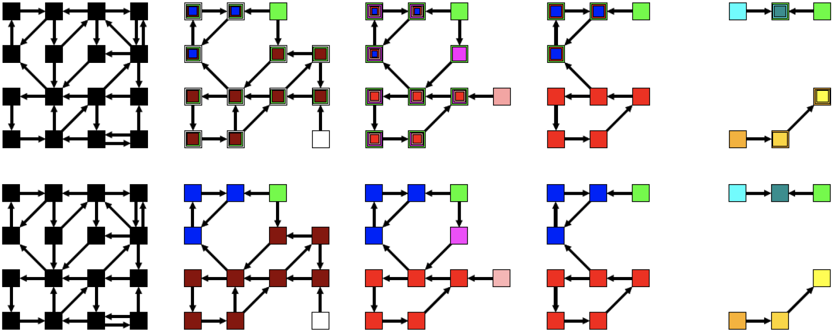




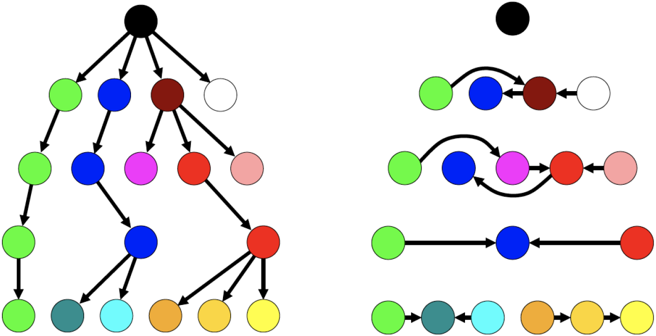
Directed hierarchies



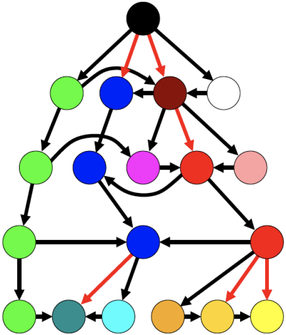
Directed hierarchies

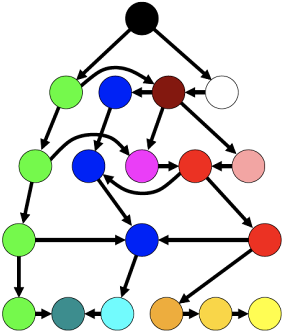


Directed hierarchies



Directed hierarchies





Content

Acknowledgements

Collaborators on these works (since 2008... a nice example of slow science)

- Benoît Naegel (Université de Strasbourg)
- Benjamin Perret (ESIEE Paris)
- Jean Cousty (ESIEE Paris)
- Camille Kurtz (Université Paris Descartes)
- Hugues Talbot (CentraleSupelec)
- Laurent Najman (ESIEE Paris)
- Éloïse Grossiord (IUCT Toulouse)
- Olena Tankyevych (Université Paris-Est Créteil)

Literature

Main papers published on these topics

- N. Passat, B. Naegel. Component-trees and multivalued images: Structural Properties. *Journal of Mathematical Imaging and Vision*, 49(1):37–50 (2014). hal-01821264
- C. Kurtz, B. Naegel, N. Passat. Connected filtering based on multivalued component-trees. *IEEE Transactions on Image Processing*, 23(12):5152–5164 (2014). hal-01694358
- B. Perret, J. Cousty, O. Tankyevych, H. Talbot, N. Passat. Directed connected operators: Asymmetric hierarchies for image filtering and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(6):1162–1176 (2015). hal-00869727
- N. Passat, B. Naegel, C. Kurtz. Component-graph construction. *Journal of Mathematical Imaging and Vision*, in Press. hal-01821264
- É. Grossiord, B. Naegel, H. Talbot, L. Najman, N. Passat. Shape-based analysis on component-graphs for multivalued image processing. *Mathematical Morphology – Theory and Applications*, in Press. hal-01695384

Thank you for your attention!