Optimization on Graphs and image generation

Camille Couprie

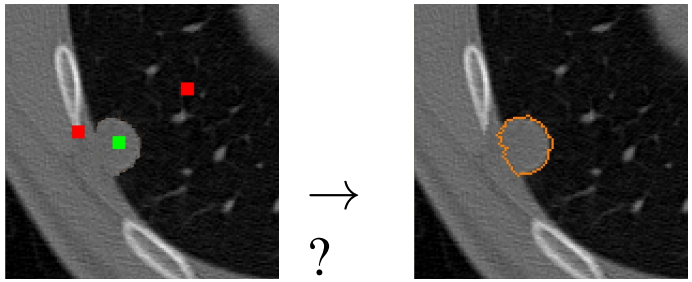
Facebook AI Research, Paris, France

September 6, 2018

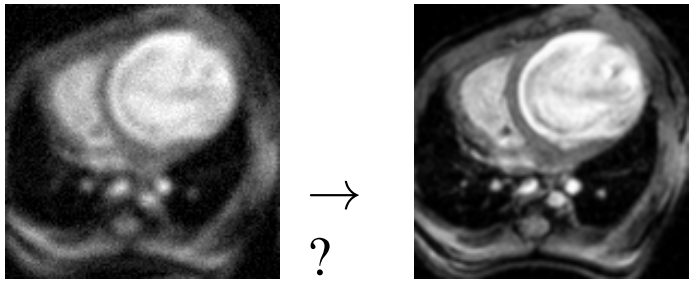
Contexte de recherche

$$x^* = \arg\min_{x \in \mathbb{R}^n} \underbrace{\sum_{e_{ij} \in E} R(x_i, x_j)}_{\text{Régularisation}} + \underbrace{\sum_{e_{ij} \in E} D(x_i, x_j)}_{\text{Fidelité aux données}}$$

Segmentation d'images



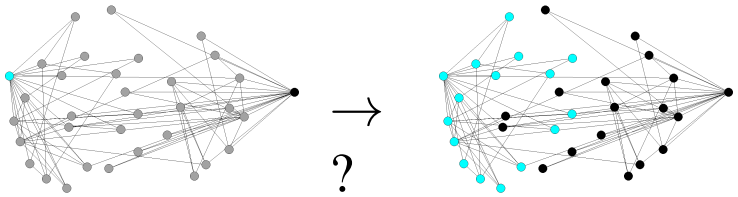
Restauration d'images



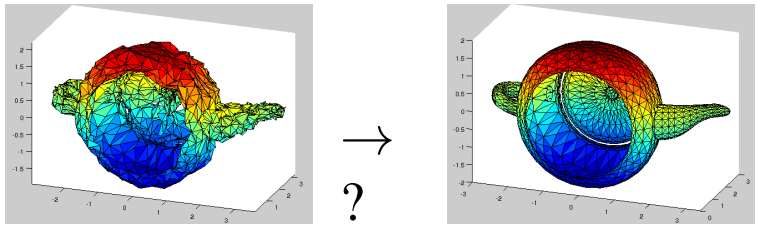
Contexte de recherche

$$x^* = \arg\min_{x \in \mathbb{R}^n} \underbrace{\sum_{e_{ij} \in E} R(x_i, x_j)}_{\text{Régularisation}} + \underbrace{\sum_{e_{ij} \in E} D(x_i, x_j)}_{\text{Fidelité aux données}}$$

Classification



Filtrage de maillages



Outline

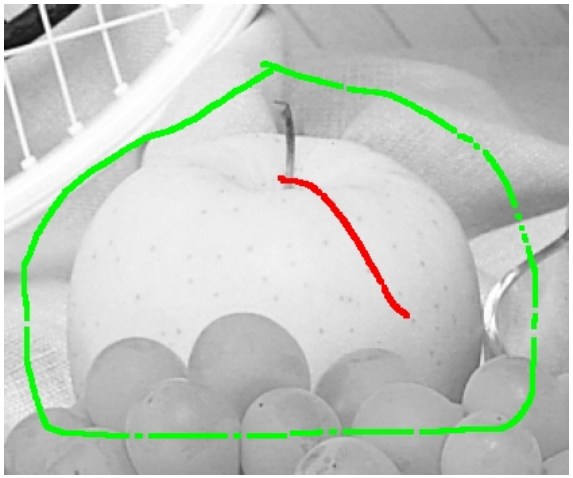
- **I - Standard graph-based methods**
- **II - Unifying optimization Framework**
  1. A new graph-based optimization framework
  2. Image segmentation
  3. Image filtering (nonconvex optimization)
  4. Surface reconstruction
- **III - Biological applications**
- **IV - Image generation using adversarial networks**
  1. Sharp video forecasting
  2. Creative image generation

Outline

- **I - Standard graph-based methods**
- **II - Unifying optimization Framework**
  1. A new graph-based optimization framework
  2. Image segmentation
  3. Image filtering (nonconvex optimization)
  4. Surface reconstruction
- **III - Biological applications**
- **IV - Image generation using adversarial networks**
  1. Sharp video forecasting
  2. Creative image generation

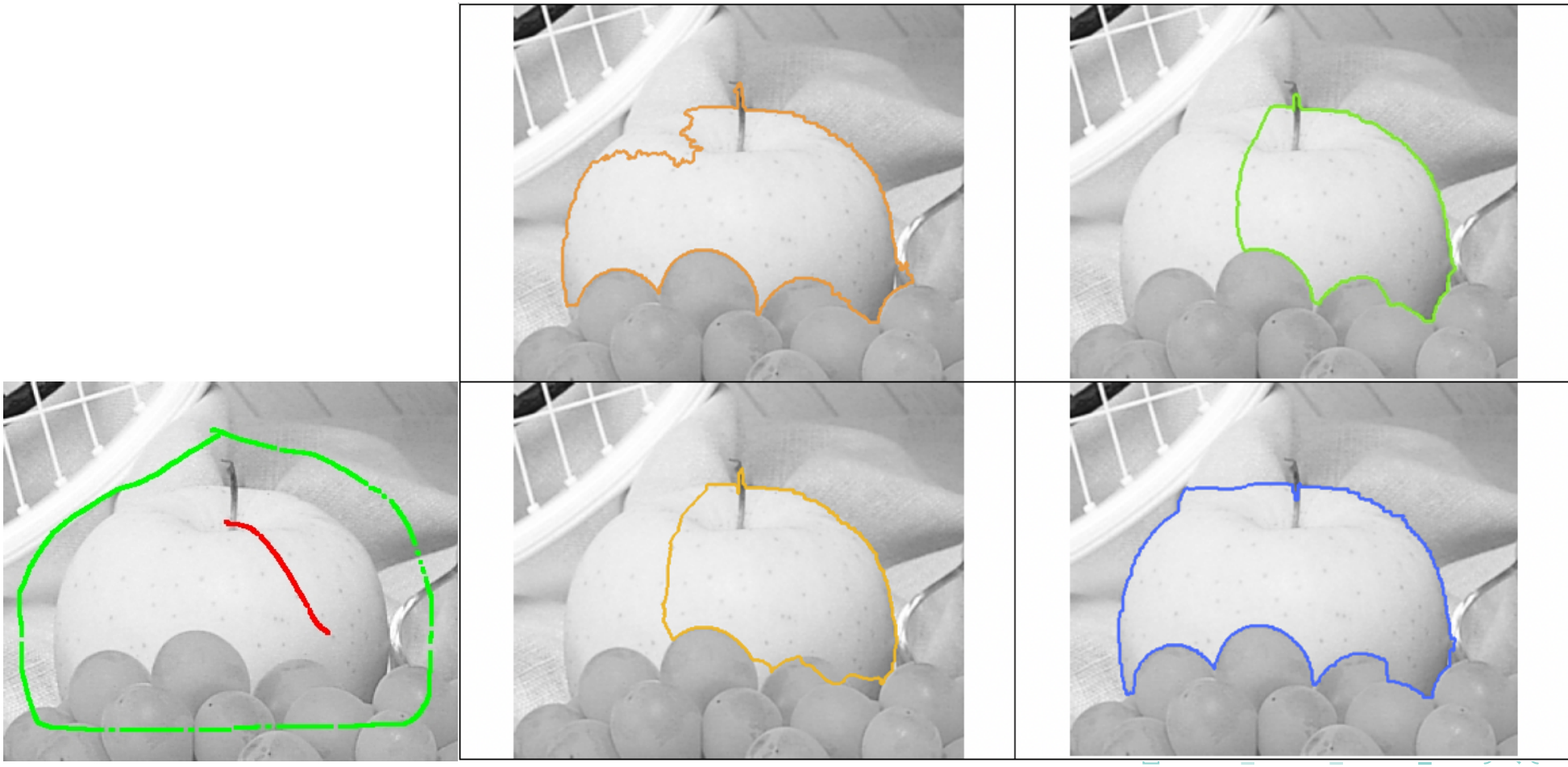
Foreground/Background segmentation of an image

Question 1: suppose you need to perform F/B segmentation of an image like this one, given only the image and some F/B seeds. How would you do?



Foreground/Background segmentation of an image

Question 1: suppose you need to perform F/B segmentation of an image like this one, given only the image and some F/B seeds. How would you do?



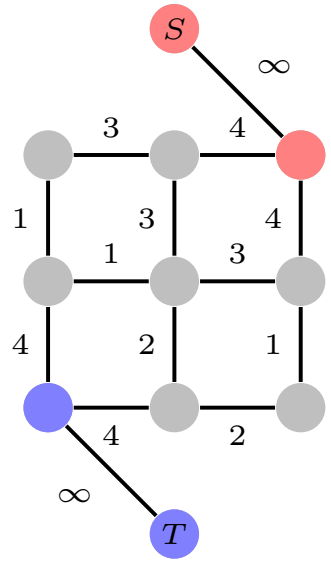
Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

Advantages

- Energy formulation → extends to a large class of problems

[Ford & Fulkerson 60s, Boykov-Joly 1998]



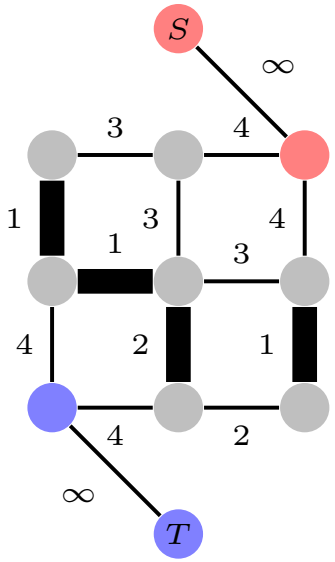
Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

Advantages

- Energy formulation → extends to a large class of problems

[Ford & Fulkerson 60s, Boykov-Joly 1998]



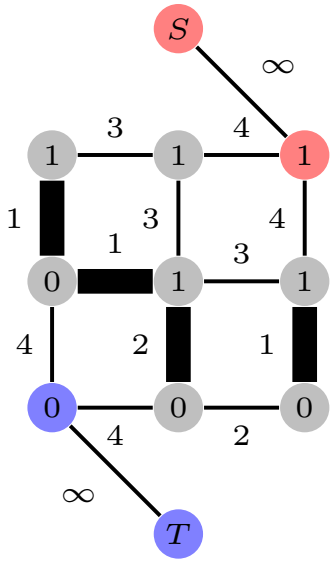
Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

Advantages

- Energy formulation → extends to a large class of problems

[Ford & Fulkerson 60s, Boykov-Joly 1998]



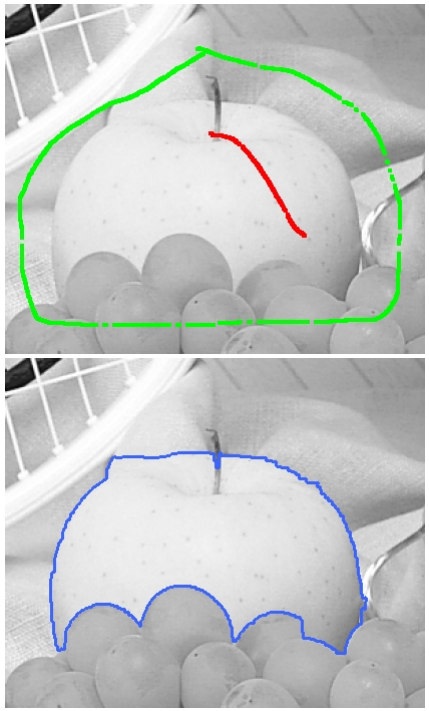
Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- Robust to markers placement

[Ford & Fulkerson 60s, Boykov-Joly 1998]



Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

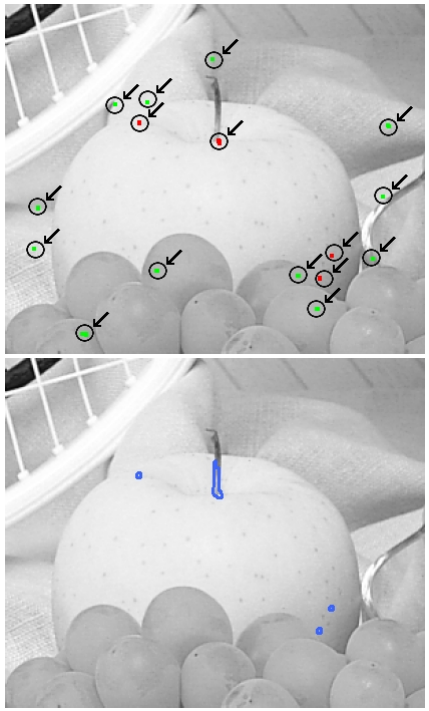
Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- Robust to markers placement

Drawbacks

- Bias toward small contours

[Ford & Fulkerson 60s, Boykov-Joly 1998]



Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

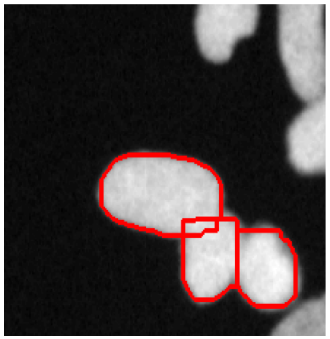
Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- Robust to markers placement

Drawbacks

- Bias toward small contours
- Block artifacts

[Ford & Fulkerson 60s, Boykov-Joly 1998]



Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

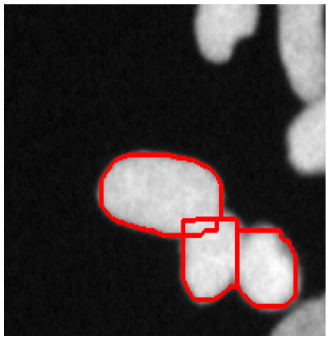
Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- Robust to markers placement

Drawbacks

- Bias toward small contours
- Block artifacts
- Super-linear complexity

[Ford & Fulkerson 60s, Boykov-Joly 1998]



Some graph-based segmentation tools: Graph Cuts

- Graph cuts / Max flow

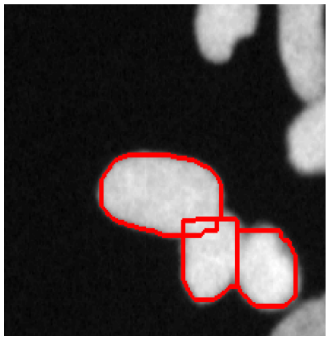
Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- Robust to markers placement

Drawbacks

- Bias toward small contours
- Block artifacts
- Super-linear complexity
- Limited to binary (2 labels) segmentation

[Ford & Fulkerson 60s, Boykov-Joly 1998]



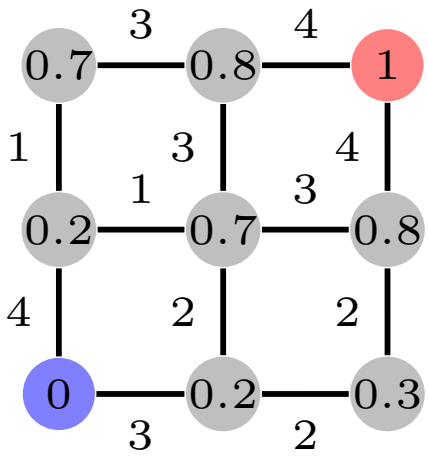
Question 2: A faster strategy:

How would you modify the graph cut energy function to solve the problem faster?

Some graph-based segmentation tools: Random Walker

- Combinatorial Dirichlet problem. Seeded segmentation [Grady 2006]
- Resolution of system of linear equations.

Advantages

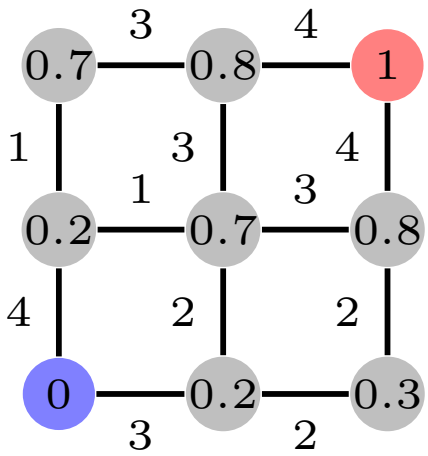


Some graph-based segmentation tools: Random Walker

- Combinatorial Dirichlet problem. Seeded segmentation [Grady 2006]
- Resolution of system of linear equations.

Advantages

- Energy formulation → extends to a large class of problems

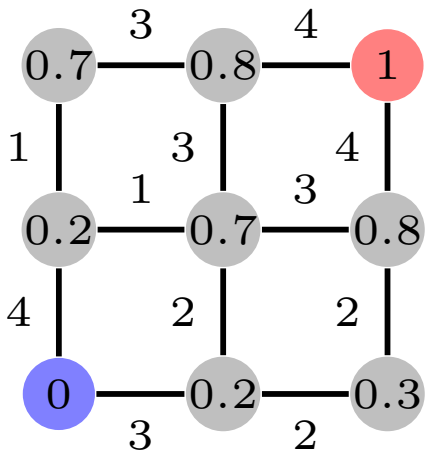


Some graph-based segmentation tools: Random Walker

- Combinatorial Dirichlet problem. Seeded segmentation [Grady 2006]
- Resolution of system of linear equations.

Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- No blocking artefacts



Some graph-based segmentation tools: Random Walker

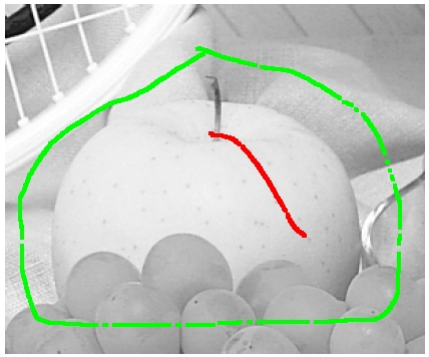
- Combinatorial Dirichlet problem. Seeded segmentation [Grady 2006]
- Resolution of system of linear equations.

Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- No blocking artefacts

Drawbacks

- Requires a more centered markers placement



Some graph-based segmentation tools: Random Walker

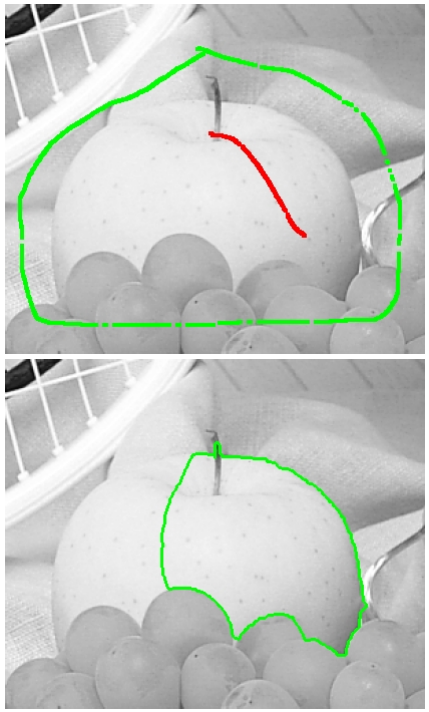
- Combinatorial Dirichlet problem. Seeded segmentation [Grady 2006]
- Resolution of system of linear equations.

Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- No blocking artefacts

Drawbacks

- Requires a more centered markers placement



Some graph-based segmentation tools: Random Walker

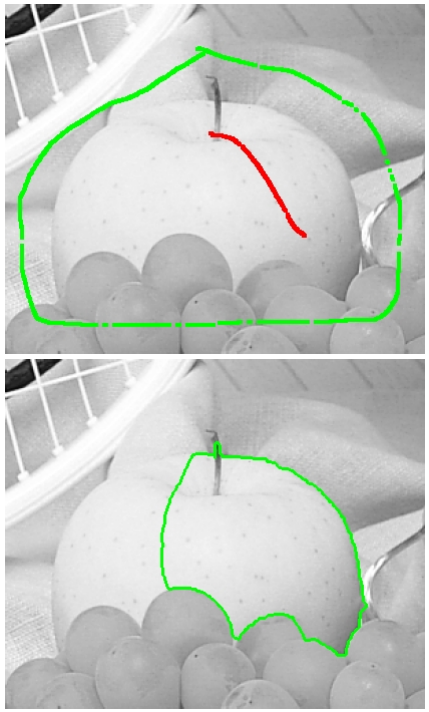
- Combinatorial Dirichlet problem. Seeded segmentation [Grady 2006]
- Resolution of system of linear equations.

Advantages

- Energy formulation $\rightarrow$ extends to a large class of problems
- No blocking artefacts

Drawbacks

- Requires a more centered markers placement
- Super-linear complexity

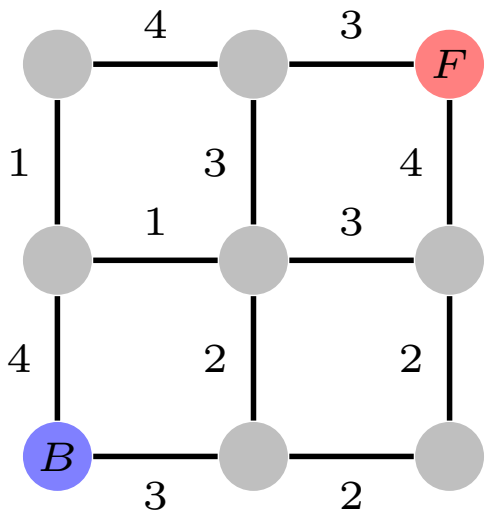


Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?

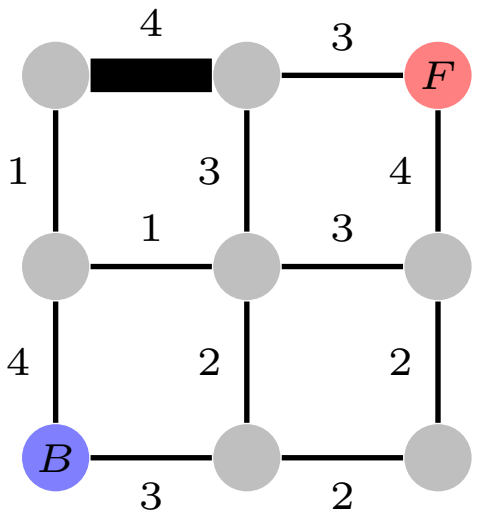
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



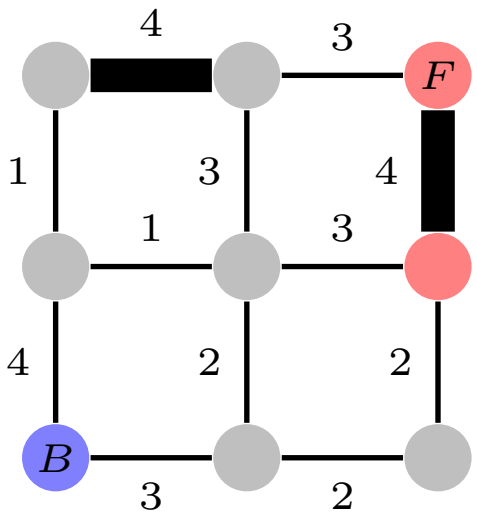
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



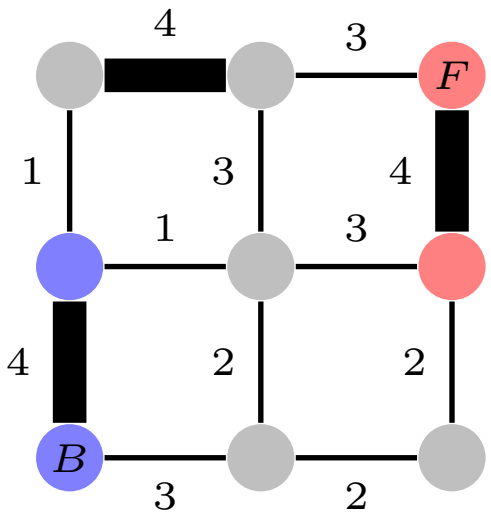
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



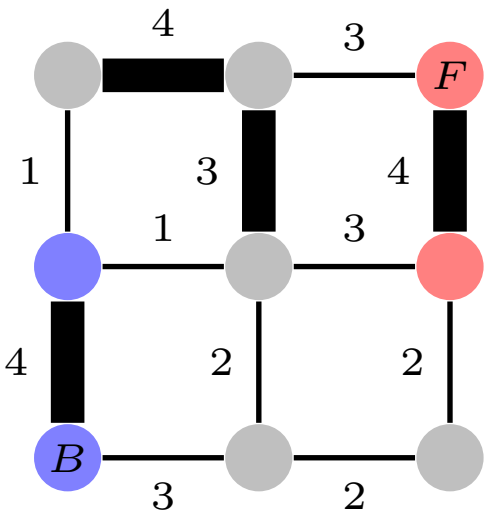
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



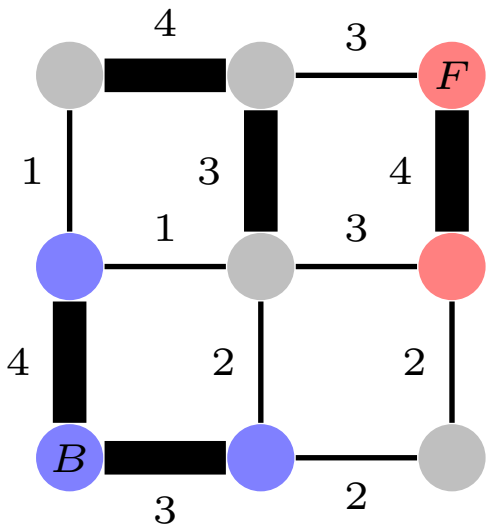
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



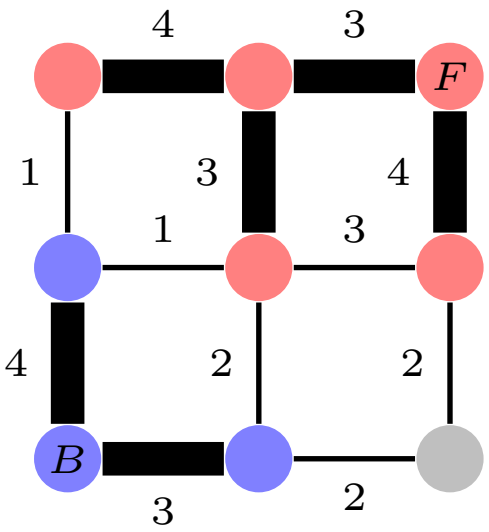
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



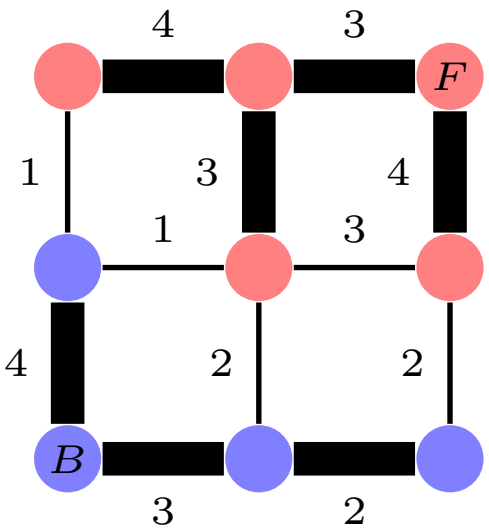
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



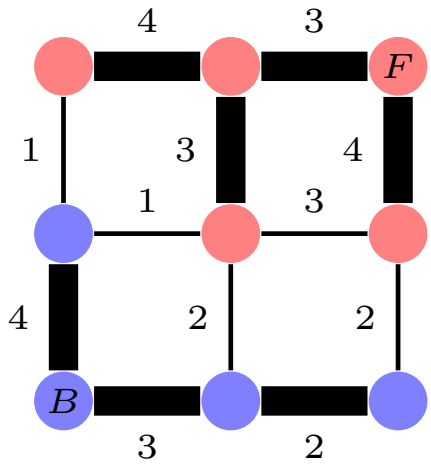
Question 3: Can you find even faster strategies:

For instance using famous algorithms from graph theory?



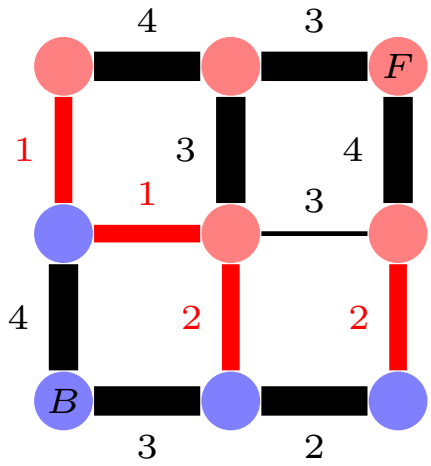
Watershed and Maximum Spanning Forest equivalence

- MSF: set of trees
  - spanning all nodes
  - not connecting different seeds
  - such that the total sum of their weights is maximum.
- If seeds are the maxima of the weight function, every MSF cut on the weight function is a watershed cut [Cousty *et al* 07, the drop of water principle]



Watershed and Maximum Spanning Forest equivalence

- MSF: set of trees
  - spanning all nodes
  - not connecting different seeds
  - such that the total sum of their weights is maximum.
- If seeds are the maxima of the weight function, every MSF cut on the weight function is a watershed cut [Cousty *et al* 07, the drop of water principle]



Some graph-based segmentation tools

- Watershed [Beucher-Lantuéjoul 1979, Vincent-Soille 1991]

Advantages

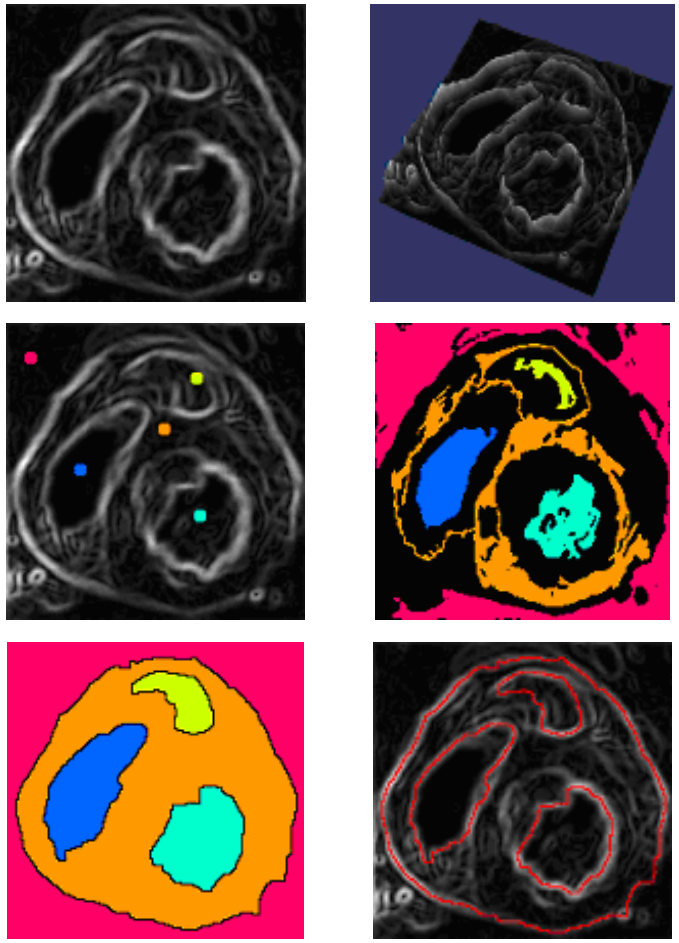
- Fast

Some graph-based segmentation tools

- Watershed [Beucher-Lantuéjoul 1979, Vincent-Soille 1991]

Advantages

- Fast
- Multilabel

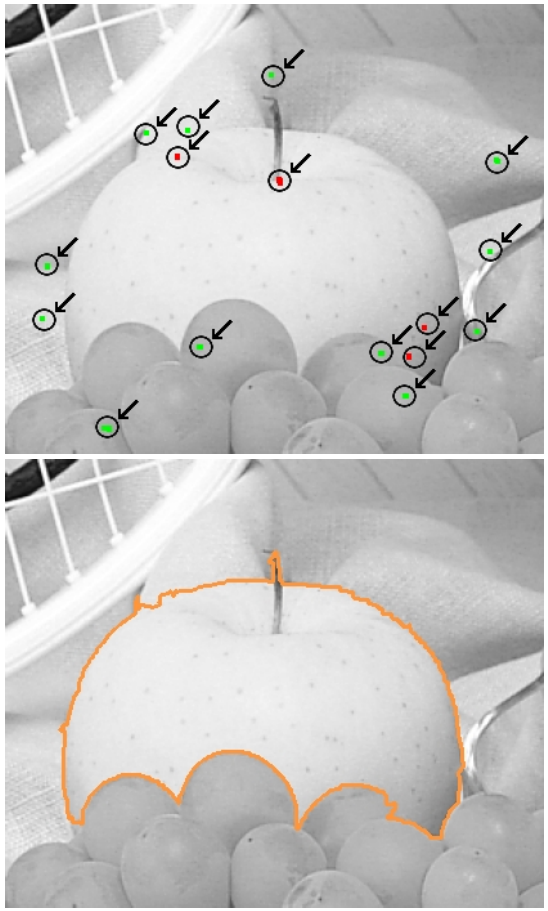


Some graph-based segmentation tools

- Watershed [Beucher-Lantuéjoul 1979, Vincent-Soille 1991]

Advantages

- Fast
- Multilabel
- Robust to markers size



Some graph-based segmentation tools

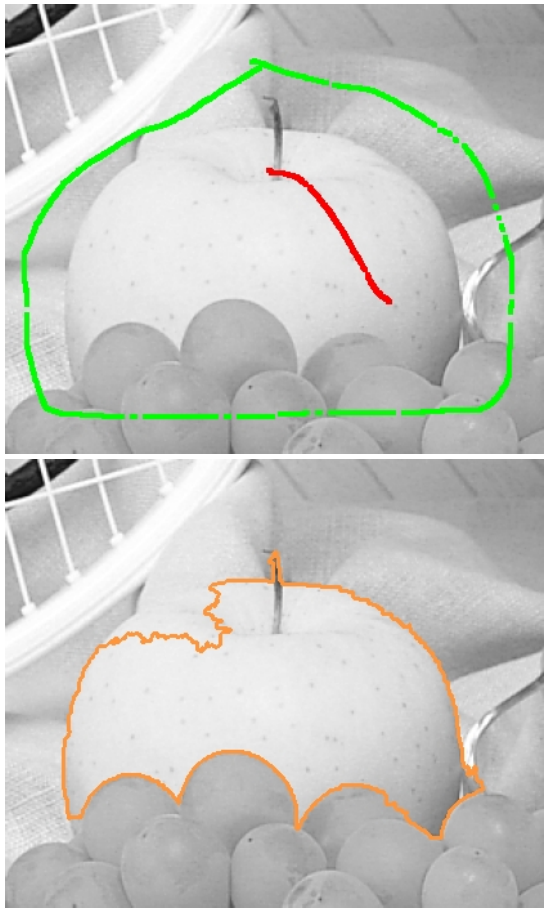
- Watershed [Beucher-Lantuéjoul 1979, Vincent-Soille 1991]

Advantages

- Fast
- Multilabel
- Robust to markers size

Drawbacks

- Leaking effect



Some graph-based segmentation tools

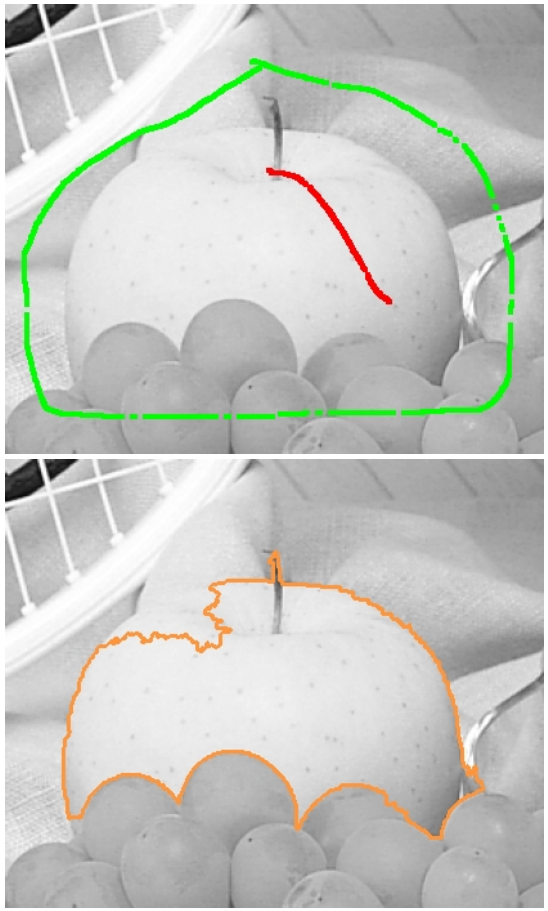
- Watershed [Beucher-Lantuéjoul 1979, Vincent-Soille 1991]

Advantages

- Fast
- Multilabel
- Robust to markers size

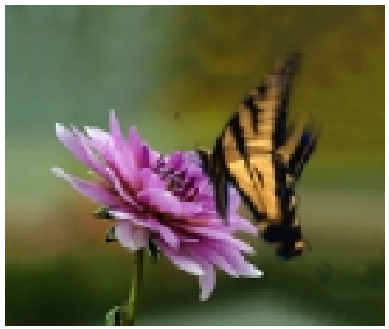
Drawbacks

- Leaking effect
- Non unique solution (difficult to get a non algorithmically dependent result)



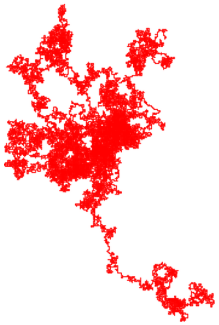
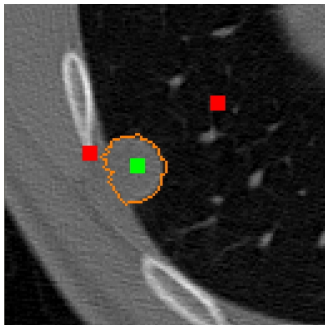
What does all those algorithms have in common ?

Graph cuts

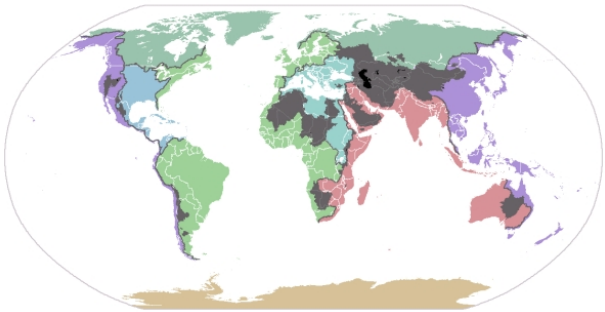



Shortest paths


Random walker

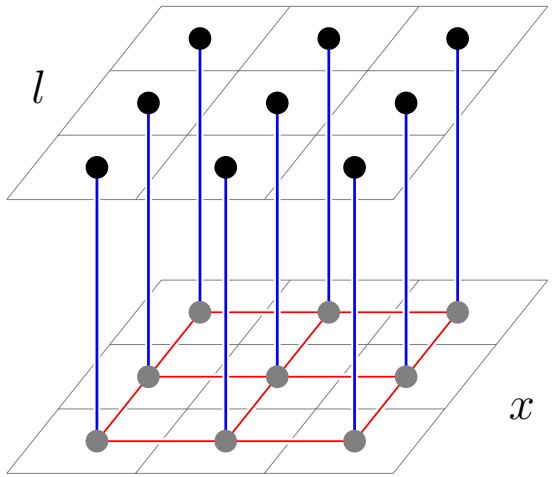



Watersheds

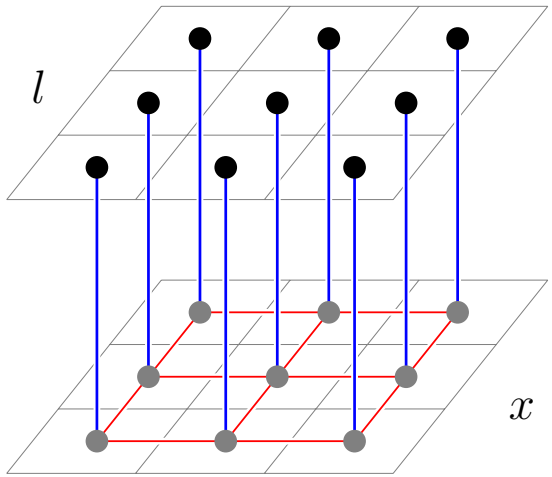
Previously established links

$$\arg\min_{x} \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^{q}|x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i\in V} w_i{}^{q}|x_i - l_i|^q}_{\text{Data term}}$$



Previously established links

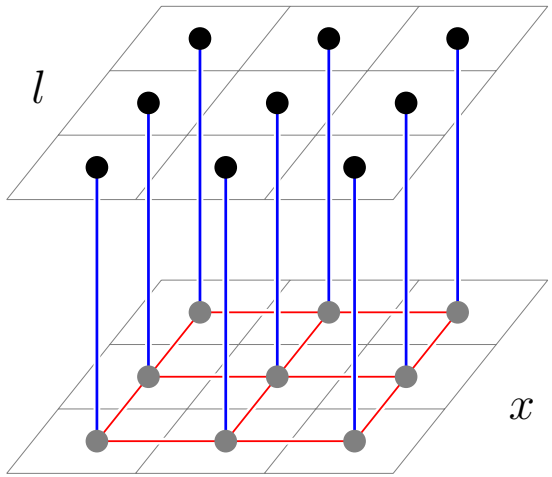
$$\arg\min_{x} \underbrace{\sum_{e_{ij}\in E} w_{ij}\ |x_i - x_j|}_{\text{Smoothness term}} + \underbrace{\sum_{v_i\in V} w_i\ |x_i - l_i|}_{\text{Data term}}$$



$q = 1$: Graph cuts [Boykov-Joly 2001 (only for 2 labels l)]

Previously established links

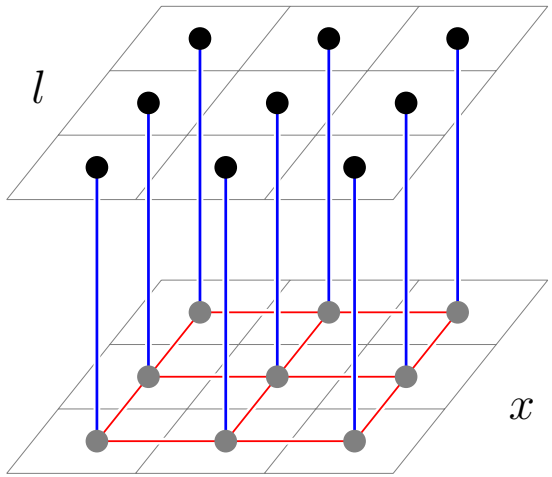
$$\arg\min_x \underbrace{\sum_{e_{ij}\in E} {w_{ij}}^2 |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} {w_i}^2 |x_i - l_i|^2}_{\text{Data term}}$$



$q = 2$: Random walker [Grady 2006]

Previously established links

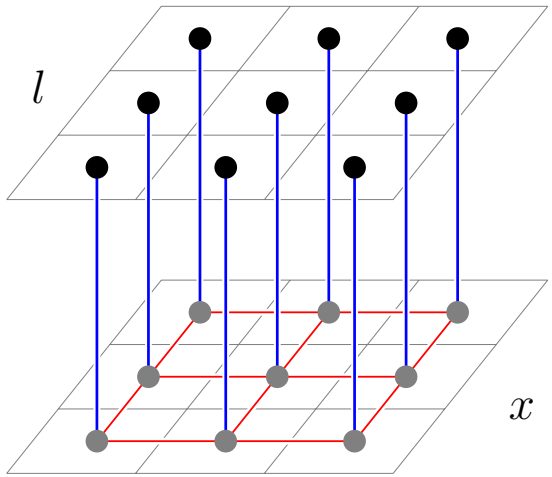
$$\lim_{q\to\infty} \arg\min_{x} \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^{q}|x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i\in V} w_i{}^{q}|x_i - l_i|^q}_{\text{Data term}}$$



$q \to \infty$: Shortest paths [Sinop *et al* 2007]

Previously established links

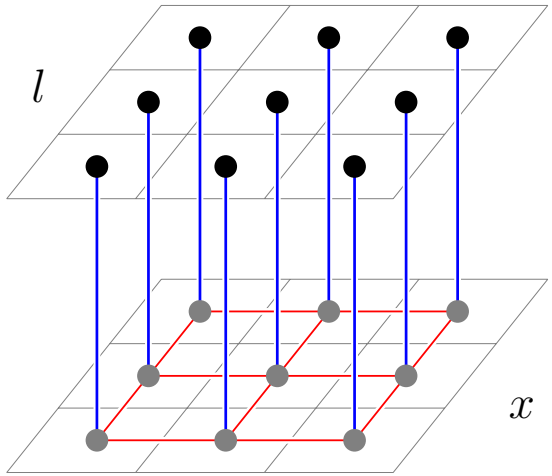
$$\lim_{p\to\infty} \arg\min_{x} \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^{p}|x_i - x_j|}_{\text{Smoothness term}} + \underbrace{\sum_{v_i\in V} w_i{}^{p}|x_i - l_i|}_{\text{Data term}}$$



$p \to \infty$: MSF (Watershed) [Allène et al. 2007]

Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$



Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$

q \ p	0	finite	∞
1	Reduction to seeds	Graph cuts	Max Spanning Forest (watershed) [Allène et al. 07]
2	ℓ_2-norm Voronoi	Random walker	
∞	ℓ_1-norm Voronoi	ℓ_1-norm Voronoi	Shortest Path [Sinop et al. 07]

Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$

q \ p	0	finite	∞
1	Reduction to seeds	Graph cuts	Max Spanning Forest (watershed) [Allène et al. 07]
2	ℓ_2-norm Voronoi	Random walker	
∞	ℓ_1-norm Voronoi	ℓ_1-norm Voronoi	Shortest Path [Sinop et al. 07]

Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$

q \ p	0	finite	∞
1	Reduction to seeds	Graph cuts	Max Spanning Forest (watershed) [Allène et al. 07]
2	ℓ_2-norm Voronoi	Random walker	
∞	ℓ_1-norm Voronoi	ℓ_1-norm Voronoi	Shortest Path [Sinop et al. 07]

Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$

q \ p	0	finite	∞
1	Reduction to seeds	Graph cuts	Max Spanning Forest (watershed) [Allène et al. 07]
2	ℓ_2-norm Voronoi	Random walker	
∞	ℓ_1-norm Voronoi	ℓ_1-norm Voronoi	Shortest Path [Sinop et al. 07]

Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$

q \ p	0	finite	∞
1	Reduction to seeds	Graph cuts	Max Spanning Forest (watershed) [Allène et al. 07]
2	ℓ_2-norm Voronoi	Random walker	
∞	ℓ_1-norm Voronoi	ℓ_1-norm Voronoi	Shortest Path [Sinop et al. 07]

Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$

q \ p	0	finite	∞
1	Reduction to seeds	Graph cuts	Max Spanning Forest (watershed) [Allène et al. 07]
2	ℓ_2-norm Voronoi	Random walker	
∞	ℓ_1-norm Voronoi	ℓ_1-norm Voronoi	Shortest Path [Sinop et al. 07]

[Couprie-Grady-Najman-Talbot, ICCV 2009, PAMI 2011]

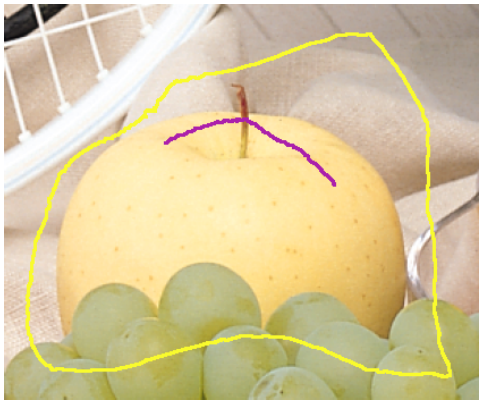
Power watershed framework

$$x^*_{p,q} = \arg\min_x \underbrace{\sum_{e_{ij}\in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\sum_{v_i \in V} w_i{}^p |x_i - l_i|^q}_{\text{Data term}}$$

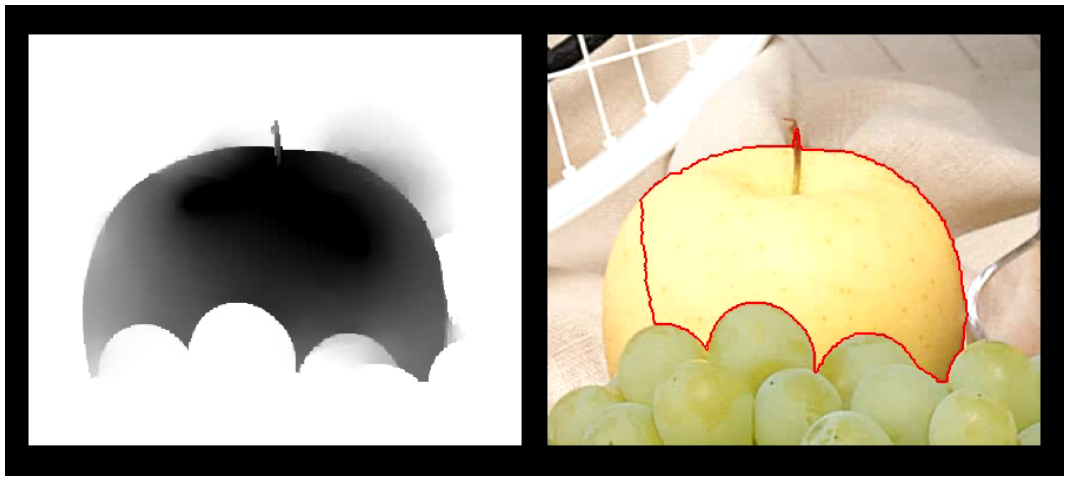
$$\bar{x} = \lim_{p\to\infty} x^*_{p,q}$$

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



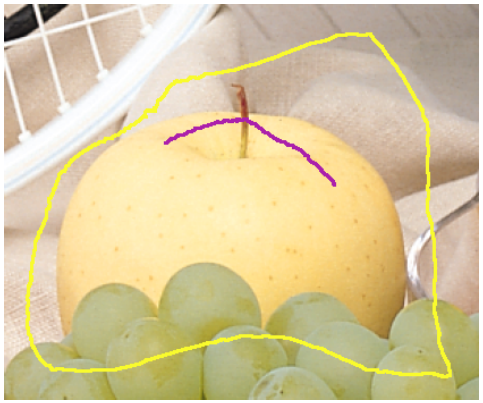
$$x^*_1 = \arg\min_x \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^1 |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



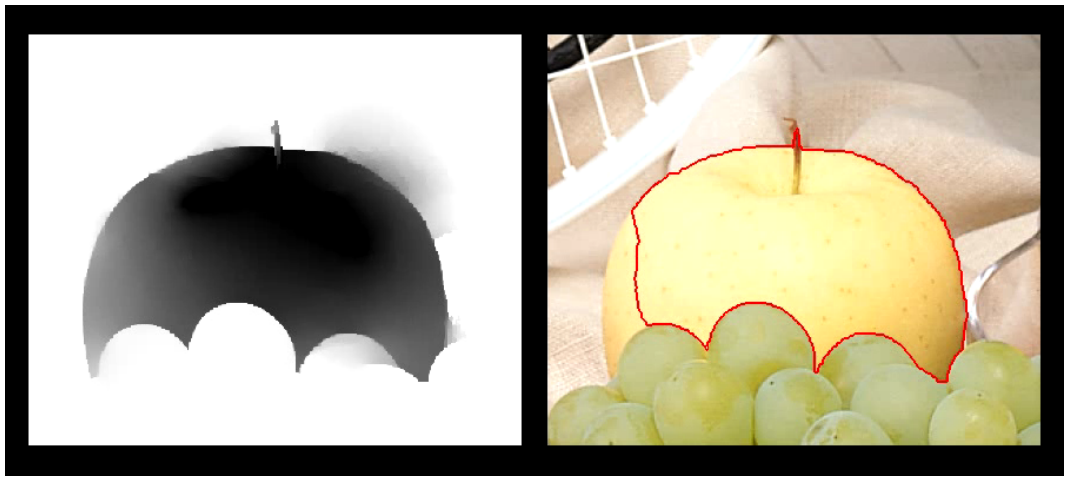
solution x^*_1 cut: threshold of x^*_1

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



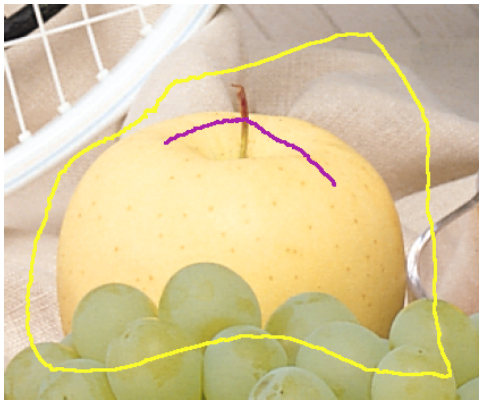
$$x^*_2 = \arg\min_x \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^2 |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



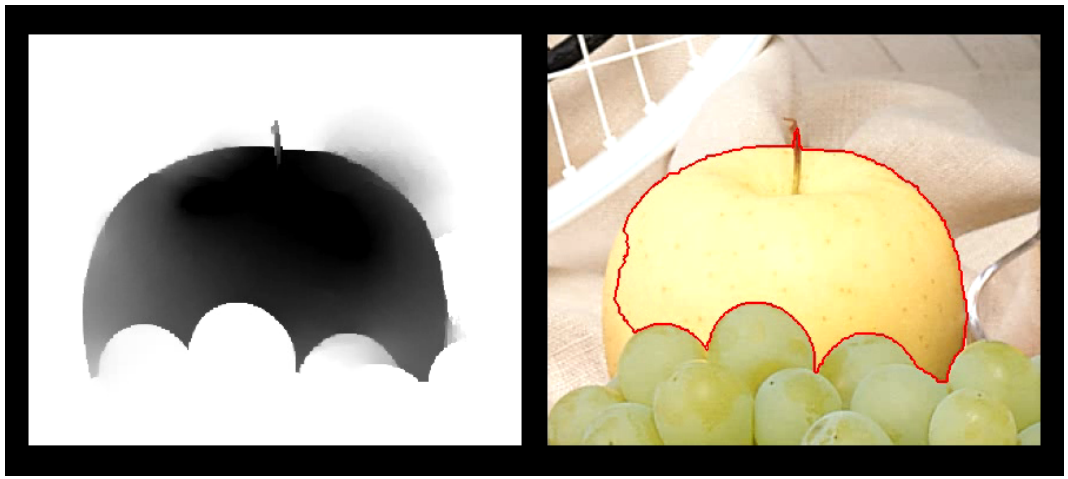
solution x^*_2 cut: threshold of x^*_2

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



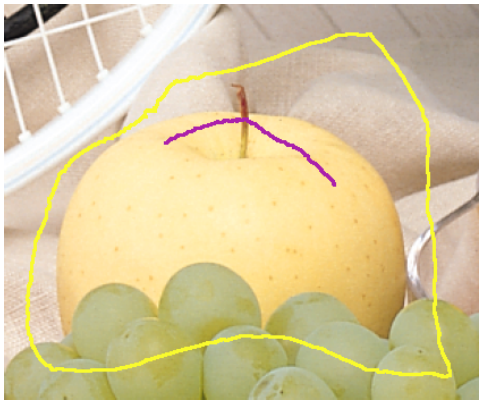
$$x^*_3 = \arg\min_x \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^3 |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



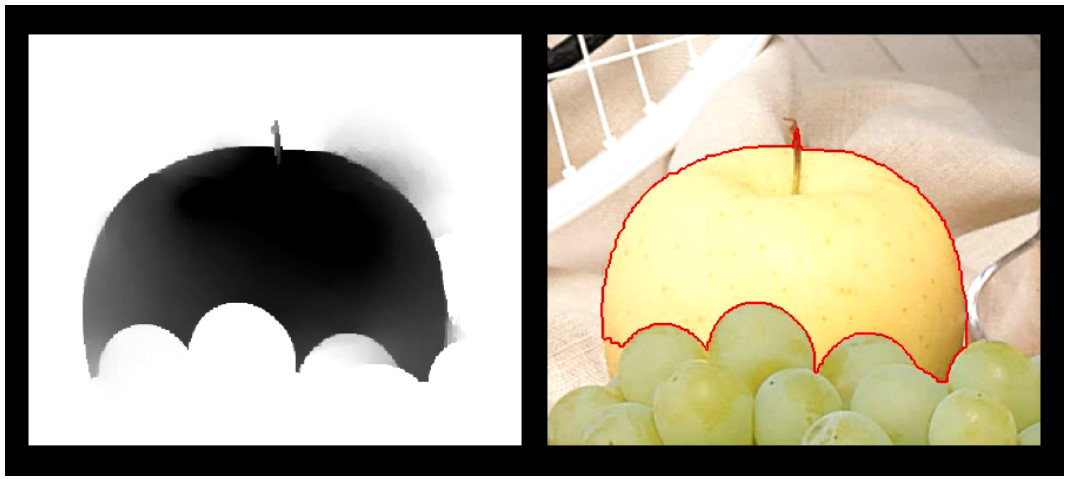
solution x^*_3 cut: threshold of x^*_3

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



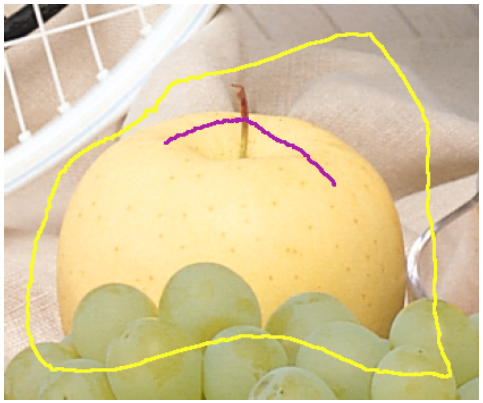
$$x^*{}_4 = \arg\min_x \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^4 |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



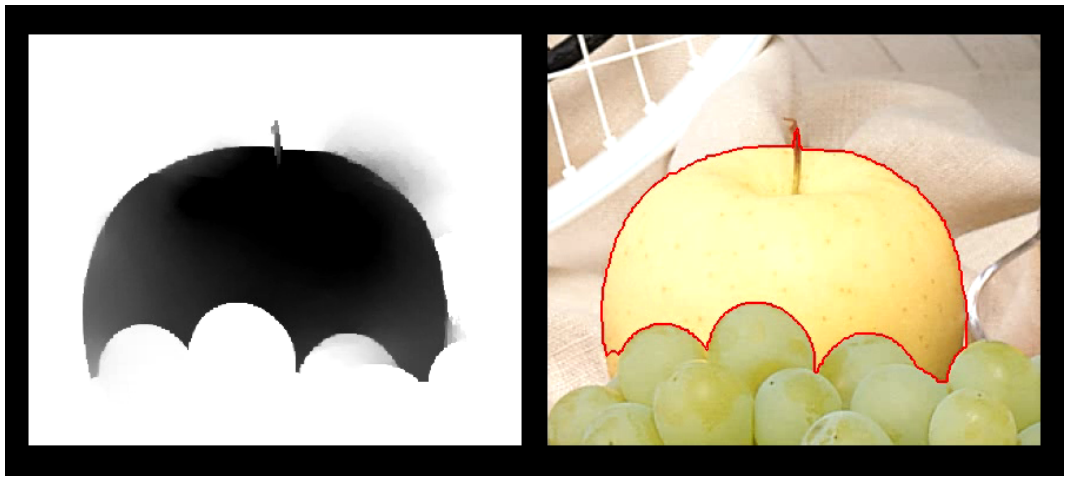
solution $x^*{}_4$ cut: threshold of $x^*{}_4$

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



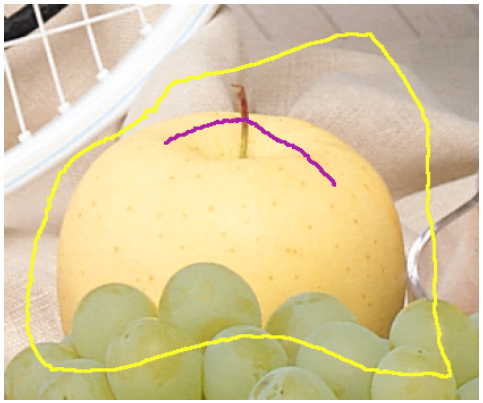
$$x^*_6 = \arg\min_x \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^6 |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



solution x^*_6 cut: threshold of x^*_6

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds

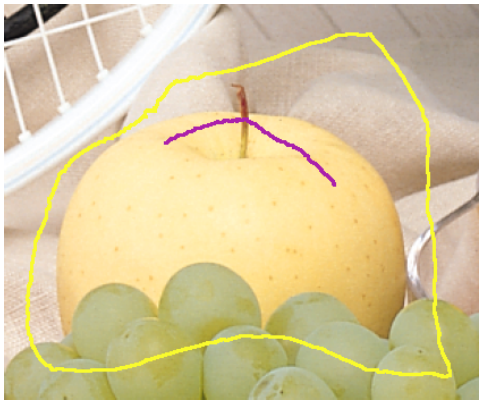


$$x^*_9 = \arg\min_x \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^9 |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$

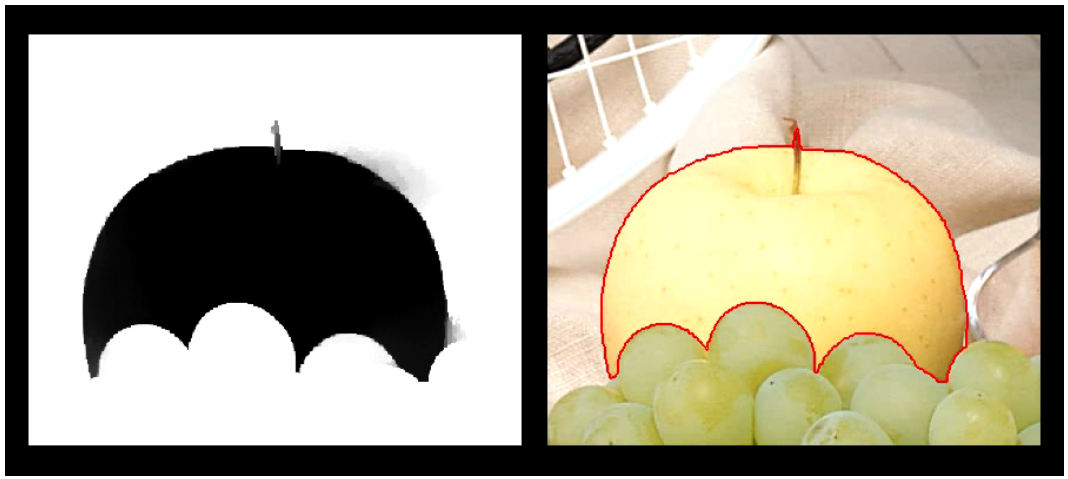
solution x^*_9 cut: threshold of x^*_9

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



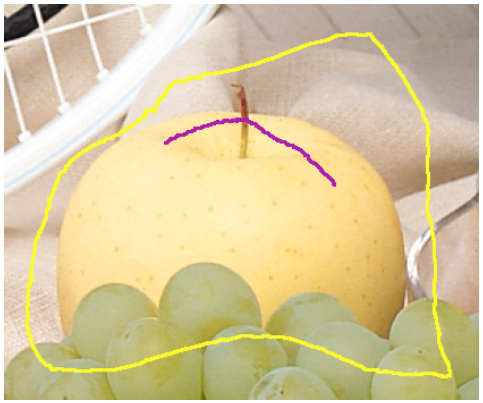
$$x_{13}{}^{*} = \arg\min_{x} \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^{13} |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



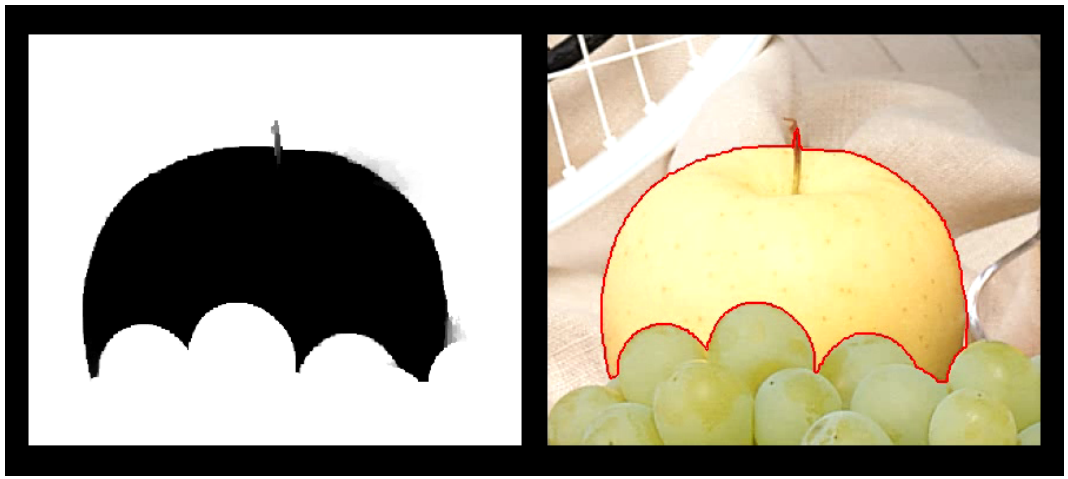
solution $x_{13}{}^{*}$ cut: threshold of $x_{13}{}^{*}$

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



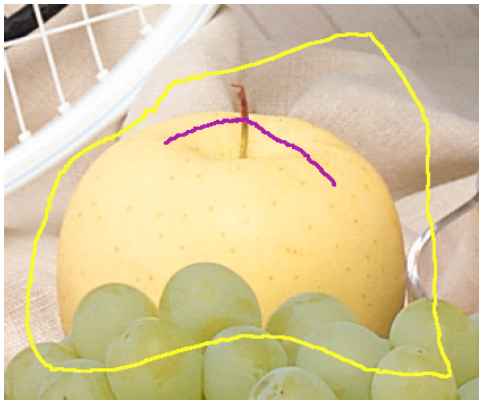
$$x_{18}{}^{*} = \arg\min_{x} \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^{18} |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



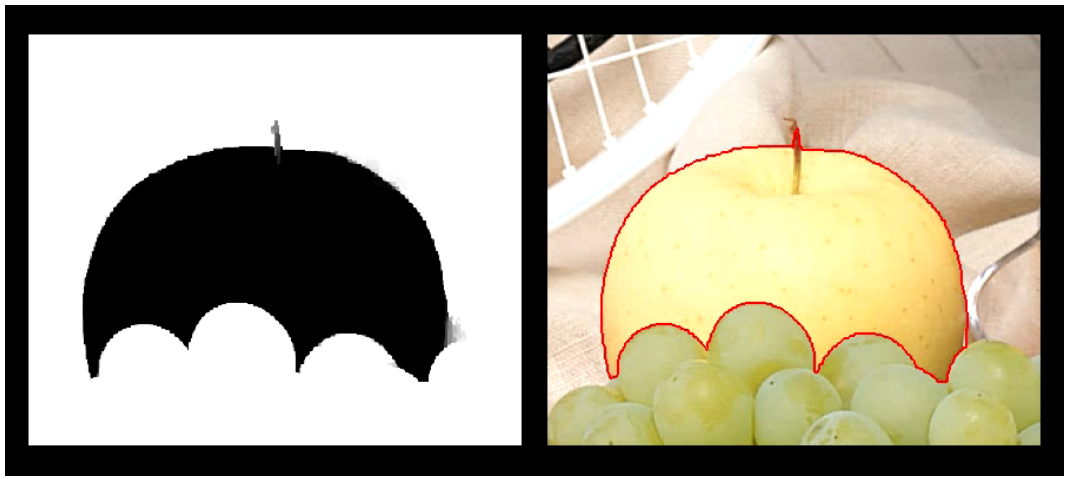
solution $x_{18}{}^{*}$ cut: threshold of $x_{18}{}^{*}$

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



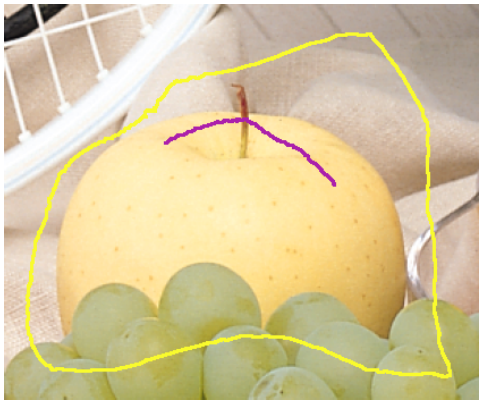
$$x_{24}{}^{*} = \arg\min_{x} \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^{24} |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



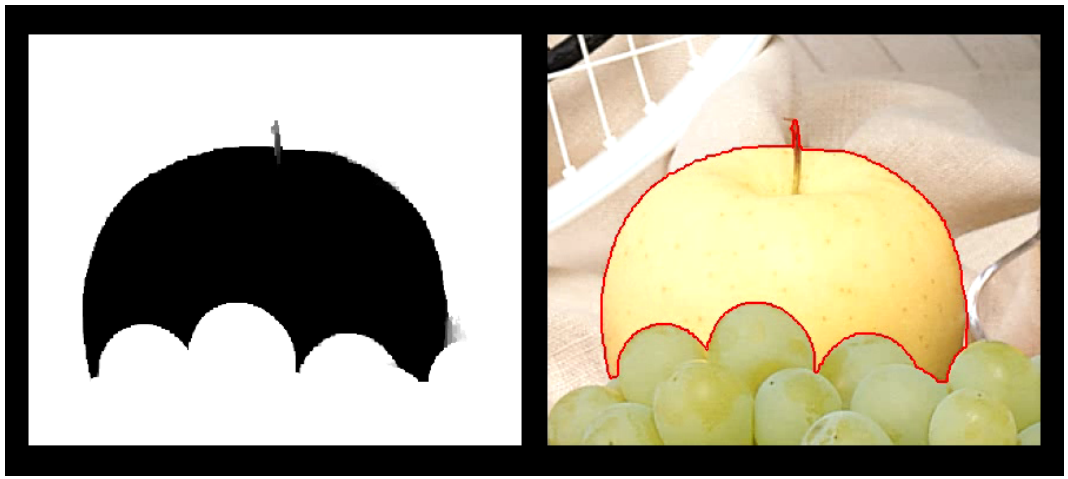
solution $x_{24}{}^{*}$ cut: threshold of $x_{24}{}^{*}$

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



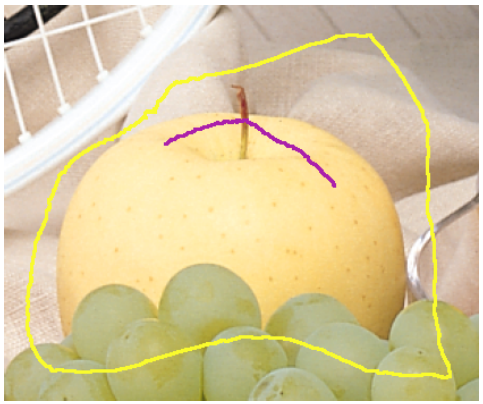
$$x_{30}{}^{*} = \arg\min_{x} \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^{30} |x_i - x_j|^2}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



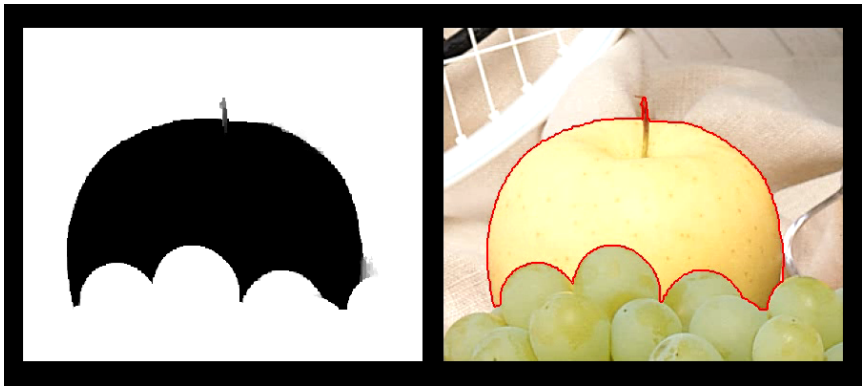
solution $x_{30}{}^{*}$ cut: threshold of $x_{30}{}^{*}$

Convergence of RW when $p \to \infty$ toward MSF cut

Input seeds



$$x_p{}^* = \arg\min_x \underbrace{\sum_{e_{ij} \in E} w_{ij}{}^p |x_i - x_j|^q}_{\text{Smoothness term}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}$$



$\bar{x} = \lim_{p \to \infty} x_p^*$ cut: threshold of $\bar{x}$

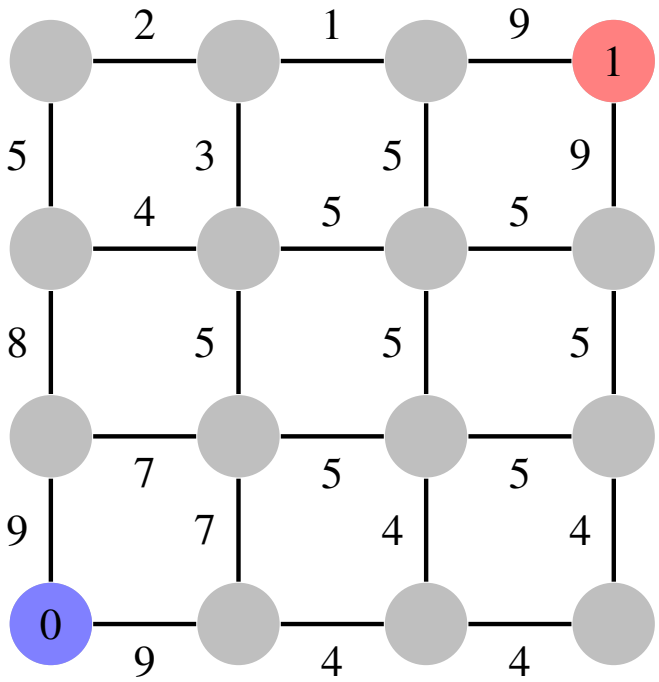
Theorems

When $p \to \infty$,

- the obtained cut is an MSF cut.
- when $q > 1$, the solution $\bar{x}$ is unique.

Power watershed algorithm

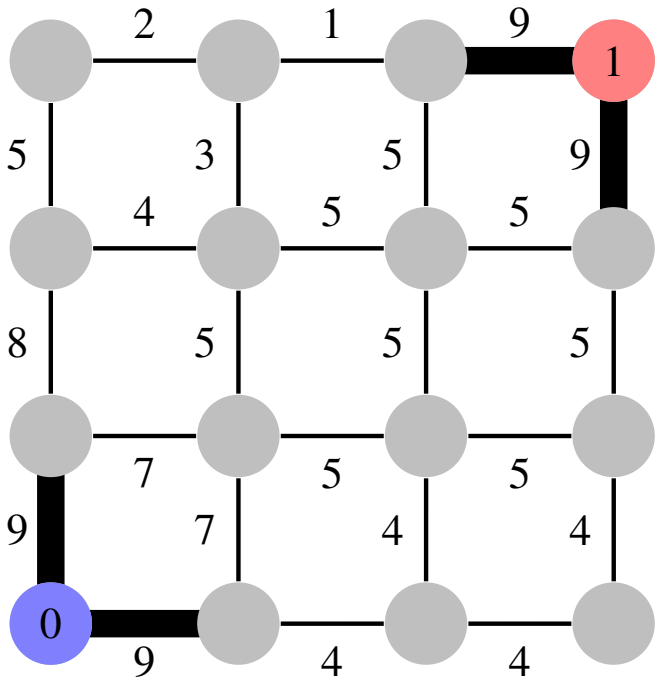
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

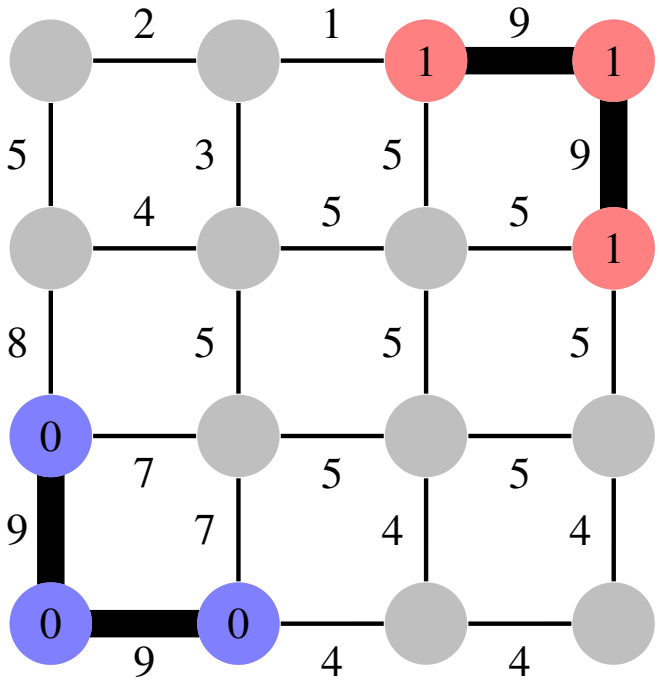
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

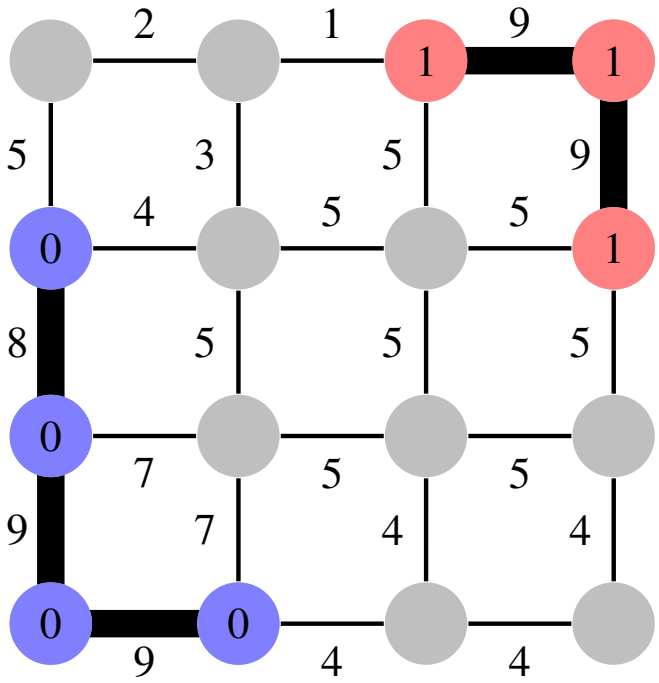
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

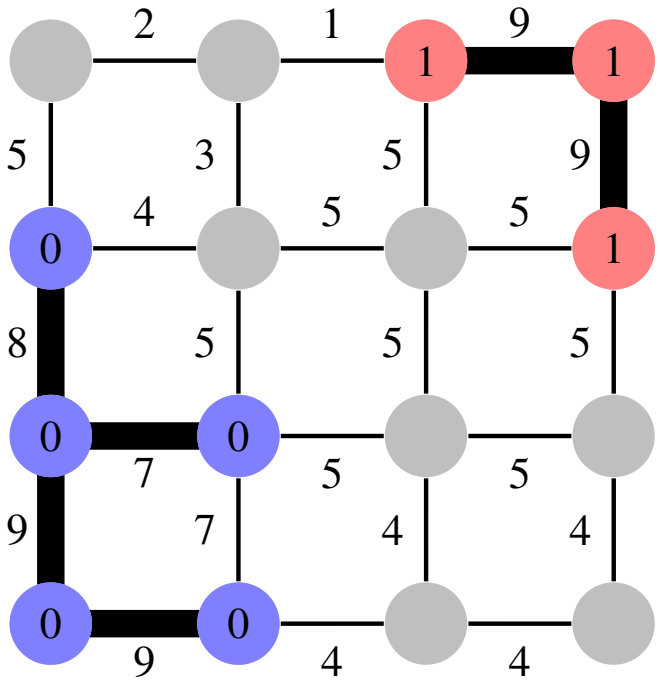
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

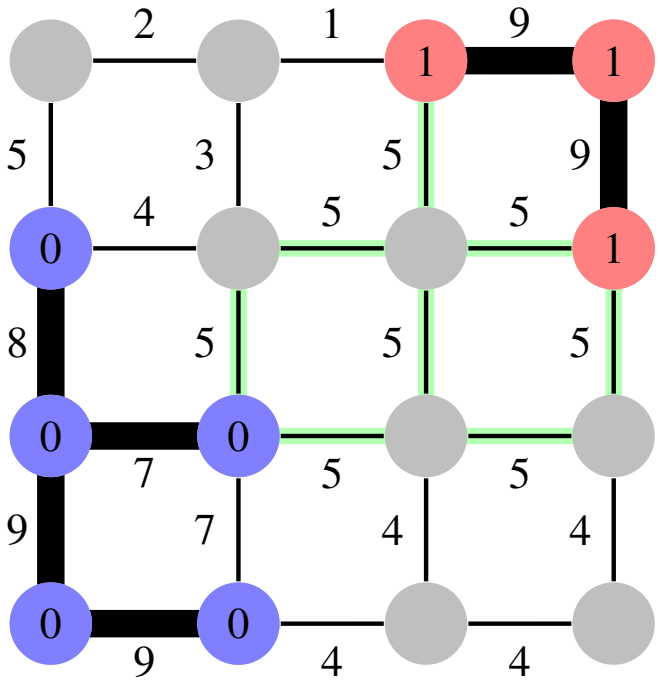
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

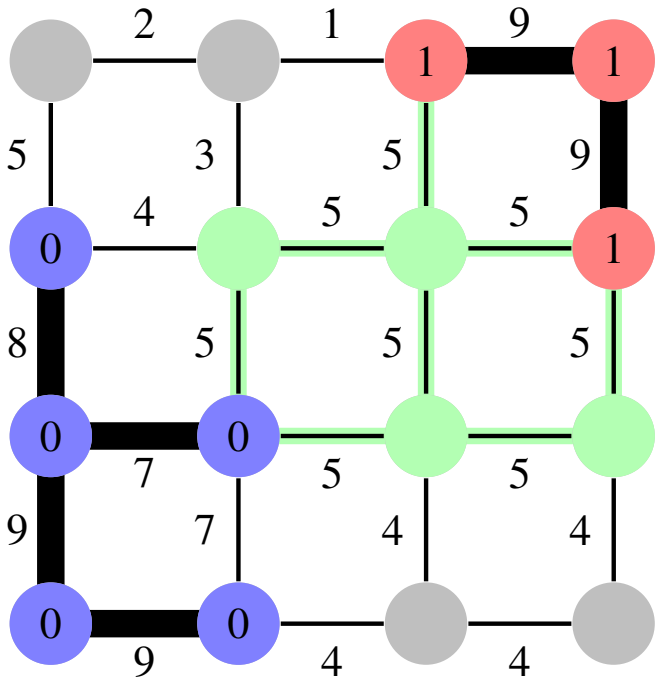
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

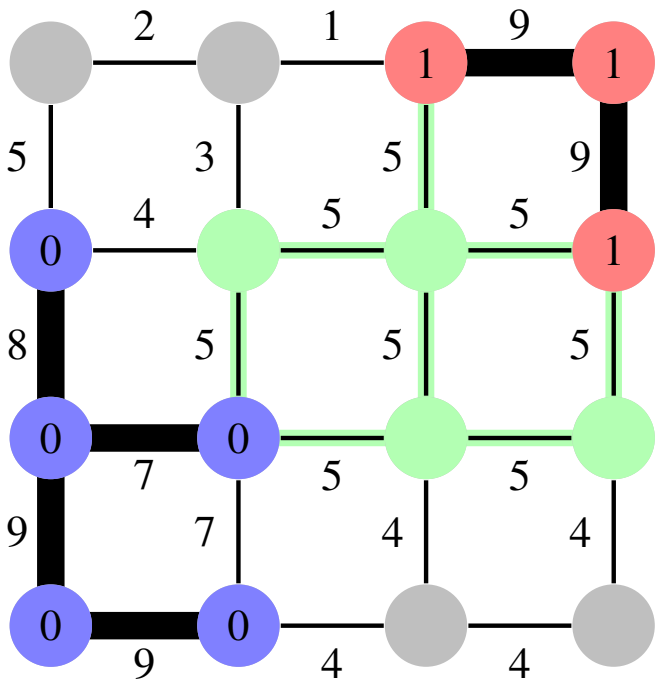
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

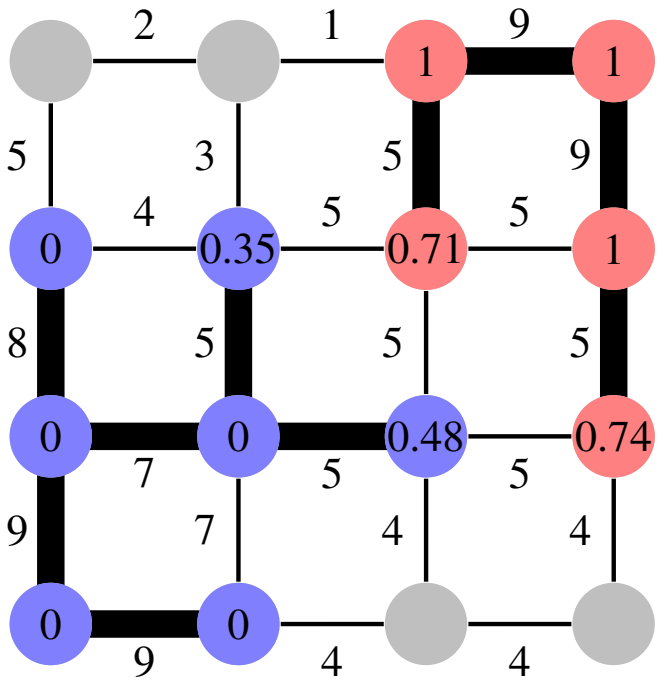
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



$$\bar{x} = \arg\min_x \sum_{e_{ij} \in \text{plateau}} |x_i - x_j|^q$$

Power watershed algorithm

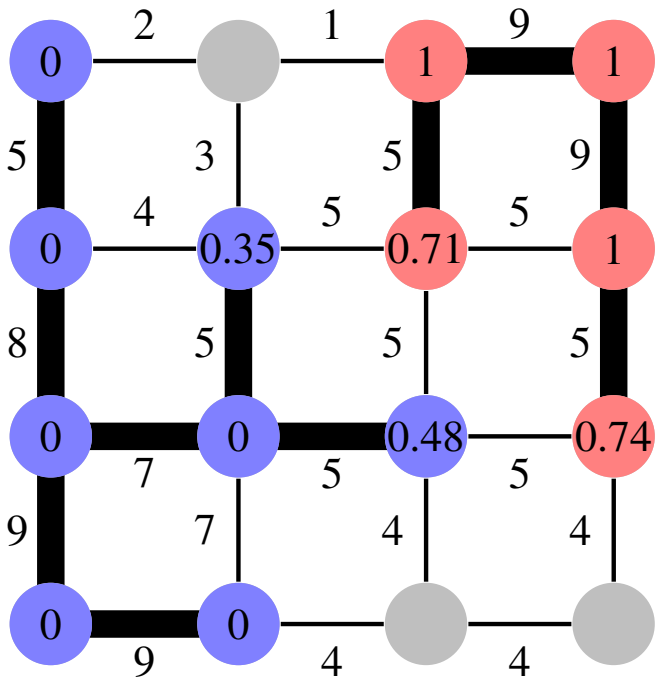
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

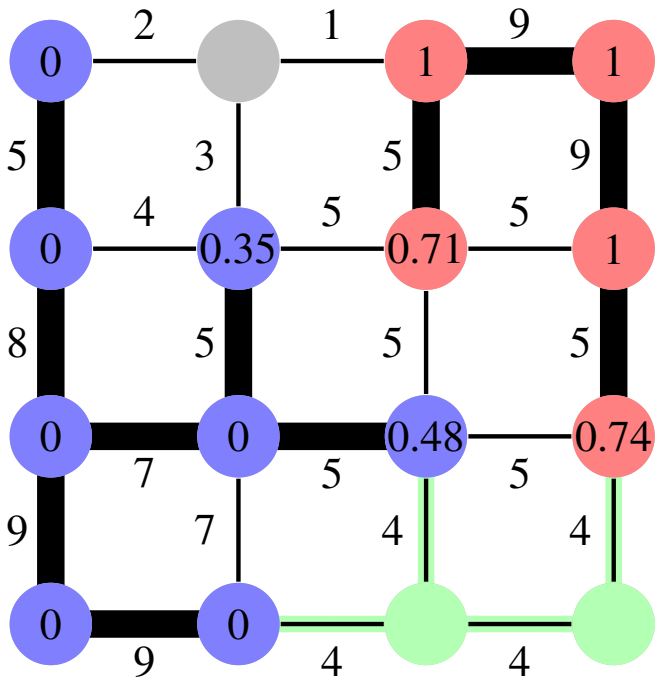
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

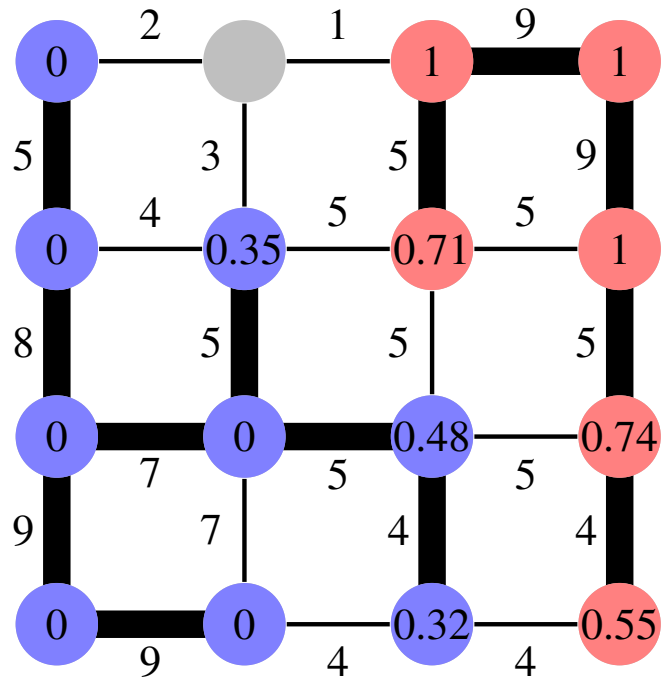
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

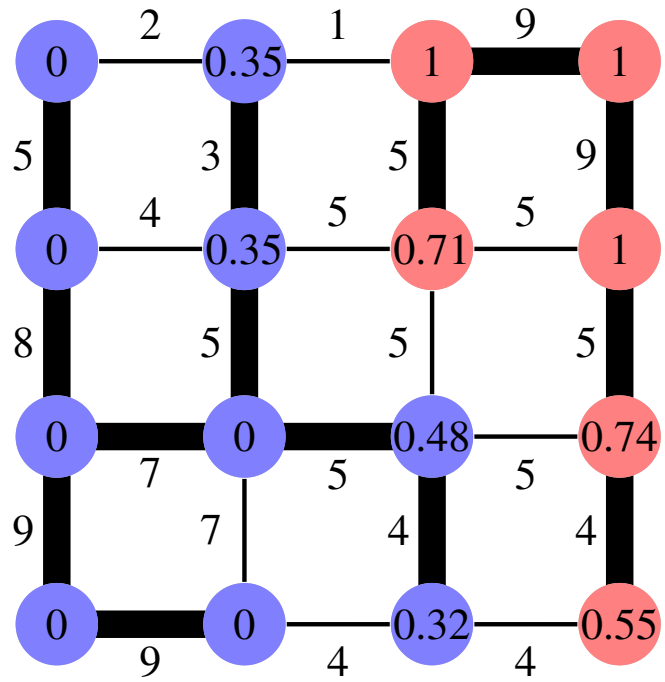
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

Power watershed algorithm

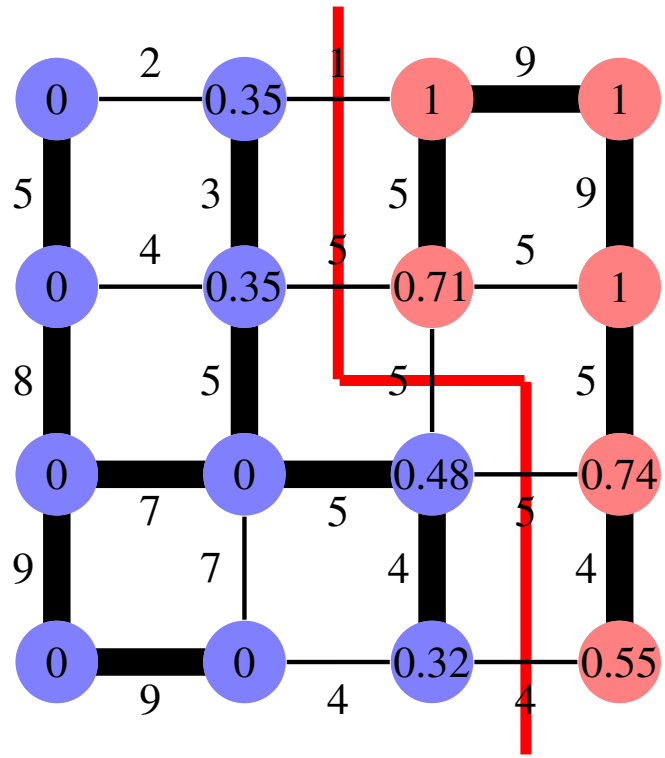
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.



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

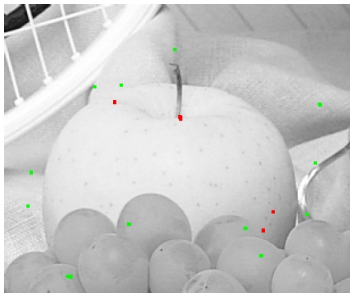
Power watershed algorithm

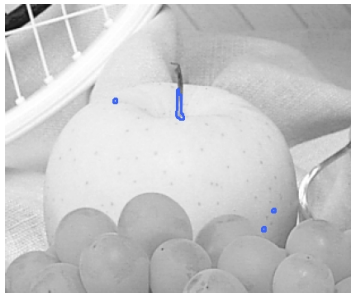
1. Choose an edge with maximal weight $e_{\max}$. Let S the set of edges connected to $e_{\max}$ with the same weight as $e_{\max}$.
2. If S does not contain vertices that have different labels, merge the nodes of S into one node, otherwise minimize $E_{1,q}$ on S.
3. Repeat steps 1 and 2 until all vertices are labeled.

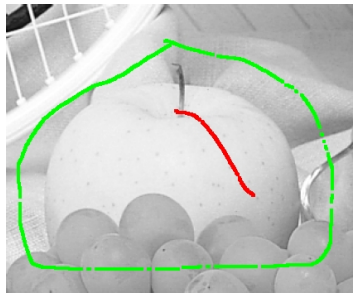


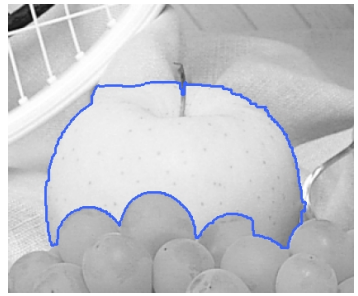
$$\bar{x} = \arg\min_x \lim_{p\to\infty} \sum_{e_{ij}\in E} w_{ij}^p |x_i - x_j|^q$$

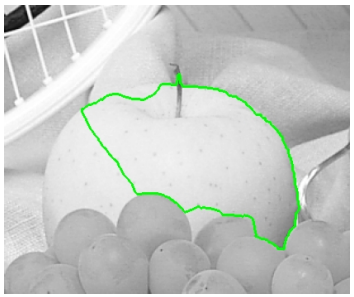
Comparison of results

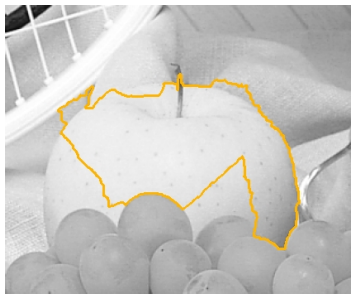

Input seeds

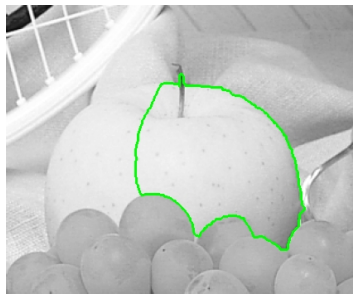

GraphCut

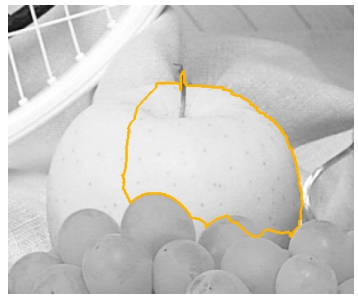

Input seeds

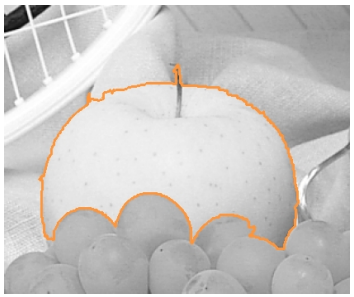

GraphCut

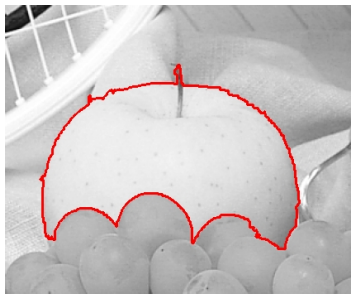

RandWalk

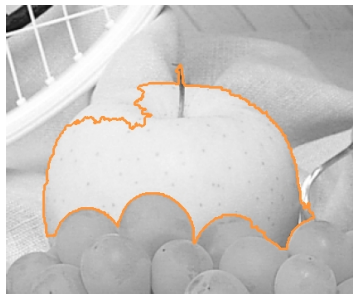

ShtPath

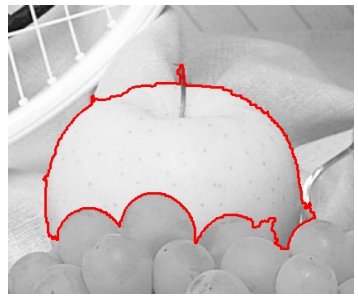

RandWalk


ShtPath

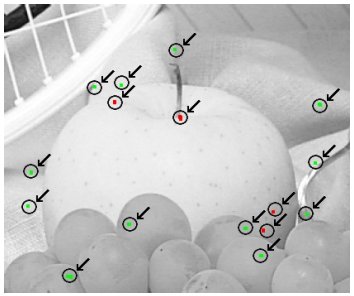

MaxSF

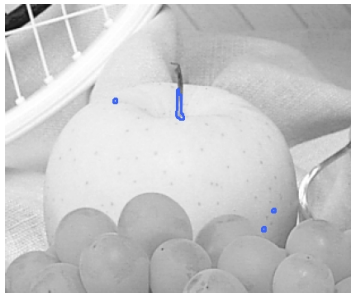

PW $q = 2$

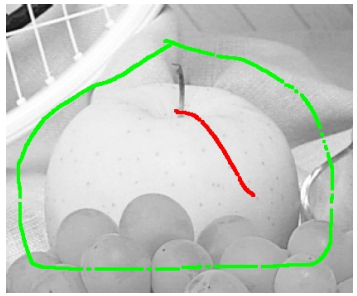

MaxSF

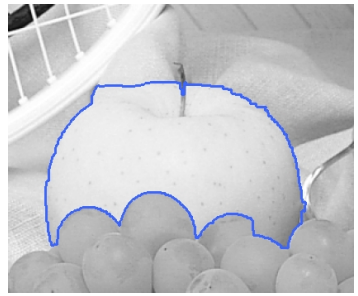

PW $q = 2$

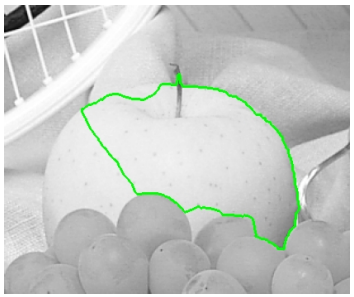
Comparison of results

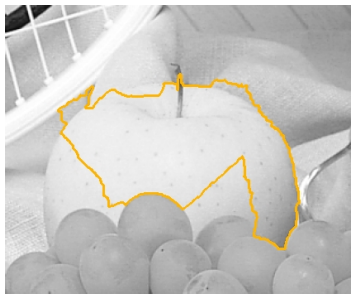

Input seeds

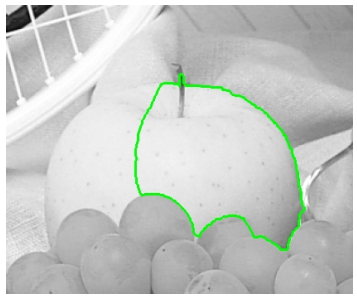

GraphCut

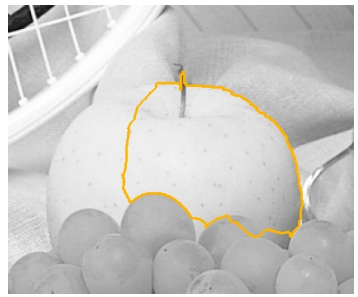

Input seeds

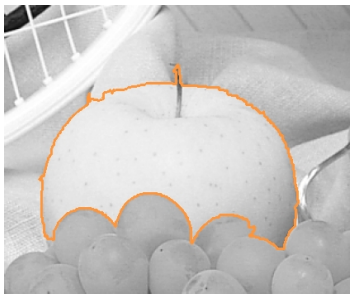

GraphCut

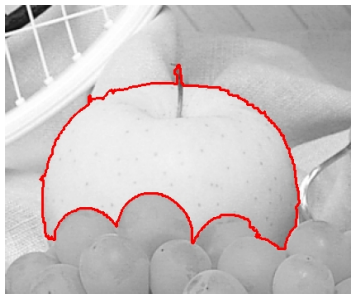

RandWalk

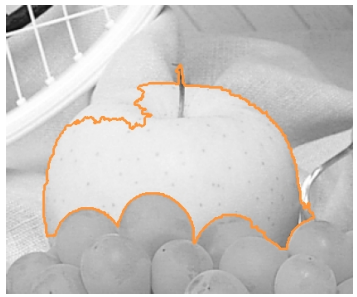

ShtPath

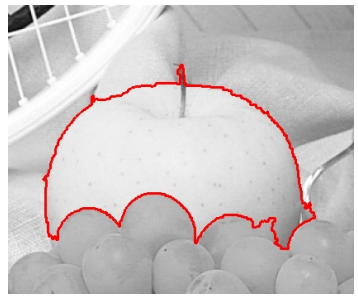

RandWalk


ShtPath


MaxSF


PW $q = 2$

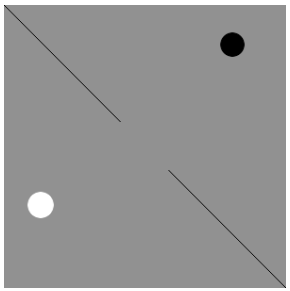

MaxSF


PW $q = 2$

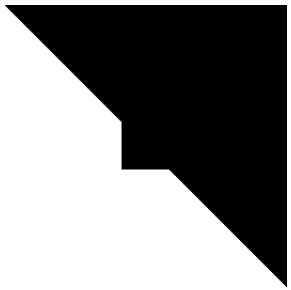
Algorithms comparison

- Evaluation on GrabCut database
- 2 sets of seeds to study robustness to seeds centering
  1. seeds well centered around boundaries:
     Best performer : Shrt path, worst performer : GraphCuts
  2. seeds less centered around boundaries: From best to worst : GraphCuts, PWshed, Random Walker, MaxSF, Shrt path
- Algorithms behavior on plateaus

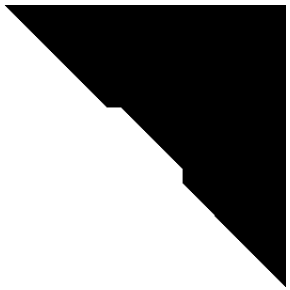
Seeded image



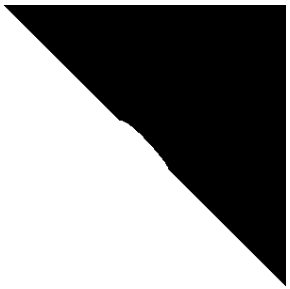
Graph Cuts



Shortest Paths, Watershed



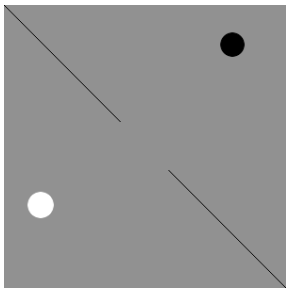
Random Walker, PW $q = 2$



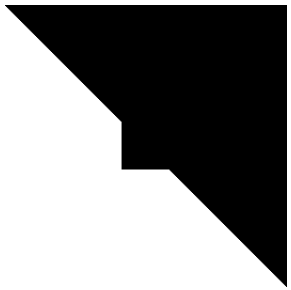
Algorithms comparison

- Evaluation on GrabCut database
- 2 sets of seeds to study robustness to seeds centering
  1. seeds well centered around boundaries:
     Best performer : Shrt path, worst performer : GraphCuts
  2. seeds less centered around boundaries: From best to worst : GraphCuts, PWshed, Random Walker, MaxSF, Shrt path
- Algorithms behavior on plateaus

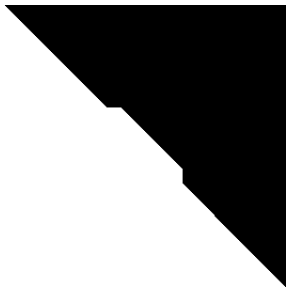
Seeded image



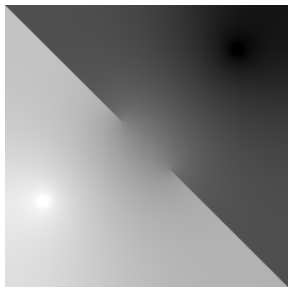
Graph Cuts



Shortest Paths, Watershed



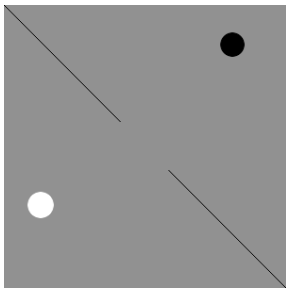
Random Walker, PW $q = 2$



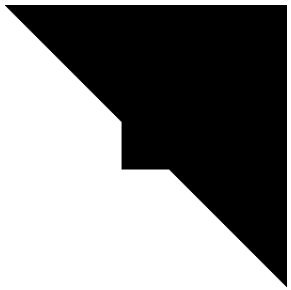
Algorithms comparison

- Evaluation on GrabCut database
- 2 sets of seeds to study robustness to seeds centering
  1. seeds well centered around boundaries:
     Best performer : Shrt path, worst performer : GraphCuts
  2. seeds less centered around boundaries: From best to worst : GraphCuts, PWshed, Random Walker, MaxSF, Shrt path
- Algorithms behavior on plateaus

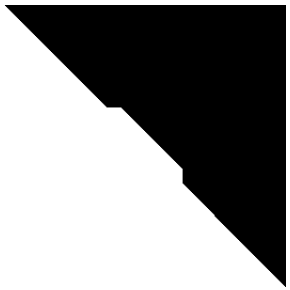
Seeded image



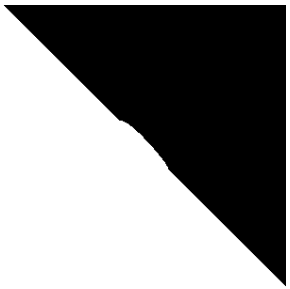
Graph Cuts



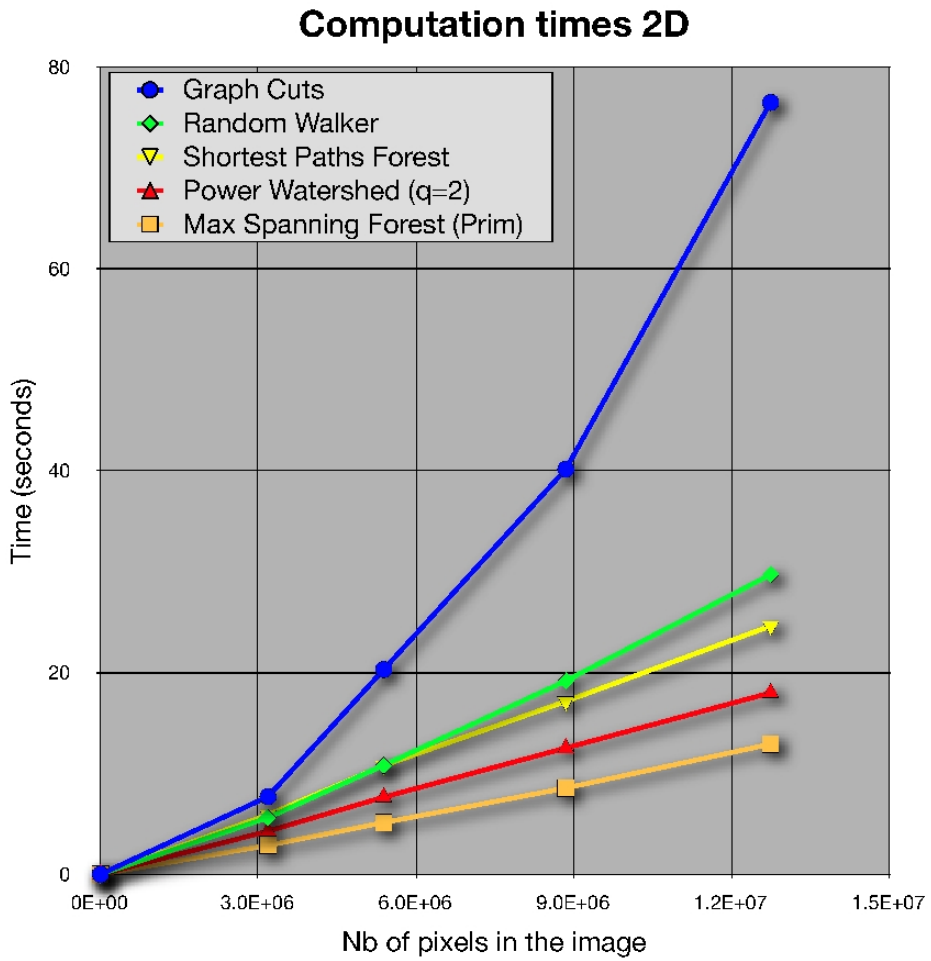
Shortest Paths, Watershed

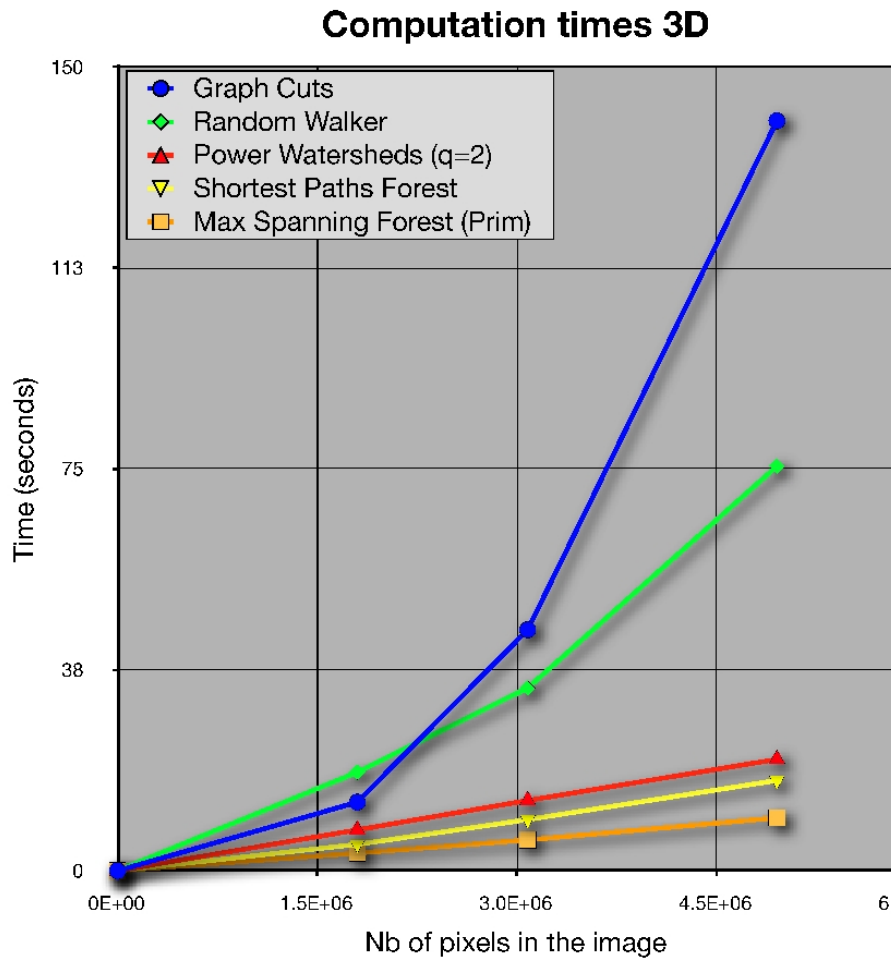


Random Walker, PW $q = 2$



Computation time



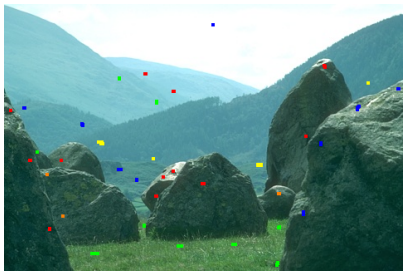


Optimal multilabels segmentation

- l solutions $x^1, x^2, \ldots x^l$ computed
- x^k computed by enforcing $\begin{cases} x^k(l^k) = 1 \\ x^k(l^q) = 0 \text{ for all } q \neq k. \end{cases}$
- 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



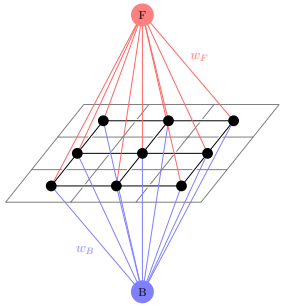
Segmentation by PowerWatershed ($q = 2$)

Question 4

How to define a new graph to perform unseeded image segmentation?

Unseeded segmentation

$$\bar{x} = \lim_{p \to \infty} \arg\min_{x} \sum_{e_{ij} \in E} w_{ij}^{p} |x_i - x_j|^q + \sum_{v_i} {w_{F_i}}^{p} |x_i - 1|^q + \sum_{v_i} {w_{Bi}}^{p} |x_i|^q$$



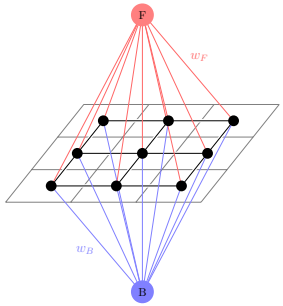
Image

Graph Cuts

Watershed

Unseeded segmentation

$$\bar{x} = \lim_{p \to \infty} \arg\min_{x} \sum_{e_{ij} \in E} w_{ij}^{p} |x_i - x_j|^q + \sum_{v_i} {w_{F_i}}^{p} |x_i - 1|^q + \sum_{v_i} {w_{Bi}}^{p} |x_i|^q$$



Image

Graph Cuts

Watershed

This is the first time that it is shown how to incorporate data unary terms into watershed computation.

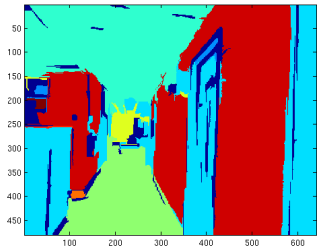
Question 4

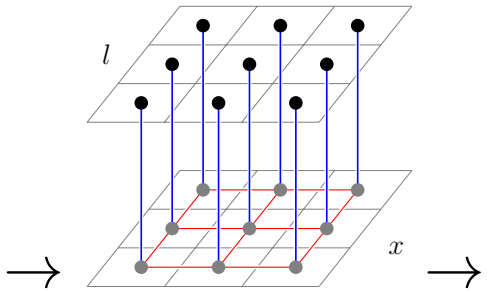
How to define a new graph to perform semantic segmentation?

Semantic Segmentation

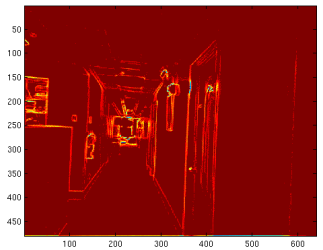
Image

Unary weights

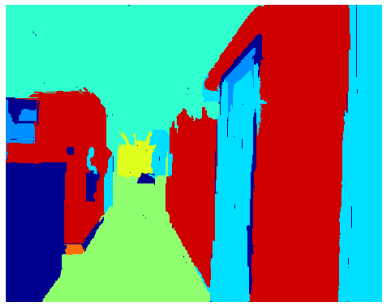




Pairwise weights

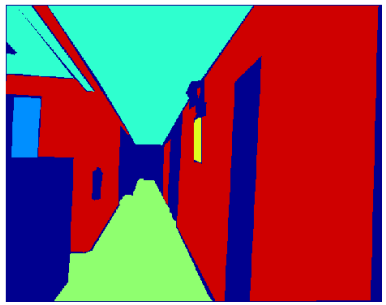


Power Watershed Result

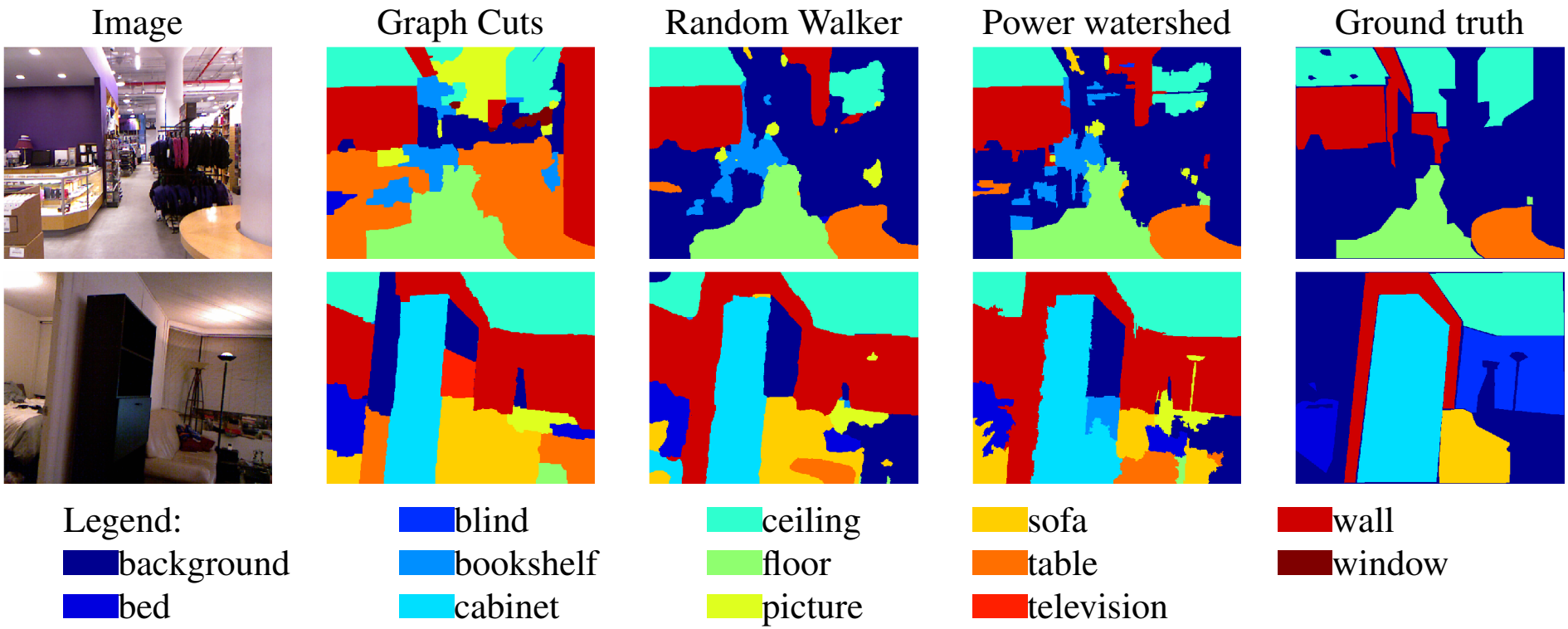


background
bookshelf
cabinet
floor
ceiling
wall
picture

Ground truth



Semantic Segmentation



Non-convex diffusion using power watersheds

- Anisotropic diffusion [Perona-Malik 1990]

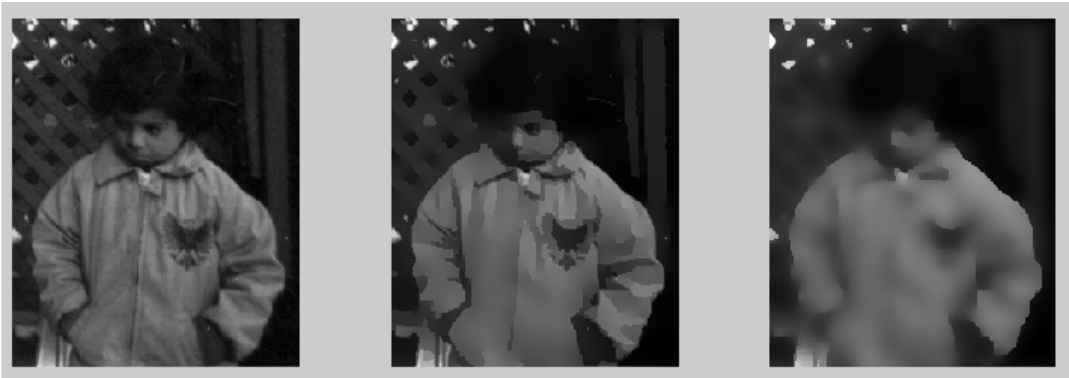


Image 100 iterations 200 iterations

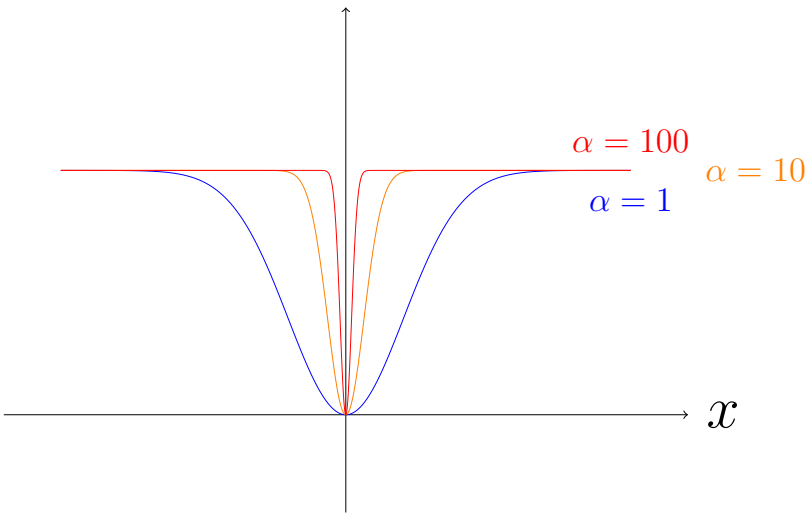
Goals of this work:

- perform anisotropic diffusion using an ℓ_0 norm to avoid the blurring effect
- optimize a non convex energy using Power Watershed [Couprie-Grady-Najman-Talbot, ICIP 2010]

Anisotropic diffusion and ℓ_0 norm

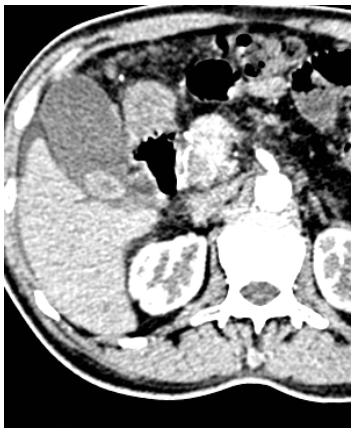
$$x^* = \arg\min_x \underbrace{\sum_{e_{ij}\in E} \sigma(x_i - x_j)}_{\text{smoothness term}} + \underbrace{\lambda \sum_{v_i \in V} \sigma(x_i - f_i)}_{\text{data fidelity term}}$$

$$\sigma(x) = 1 - \mathrm{e}^{-\alpha x^2}$$



Leads to piecewise constant results

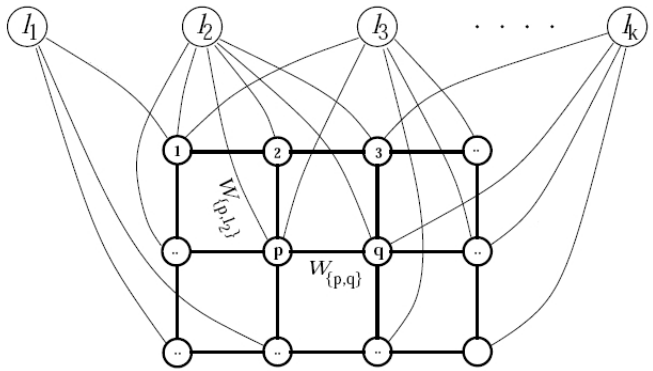
Original image



PW result

Stereovision using power watershed

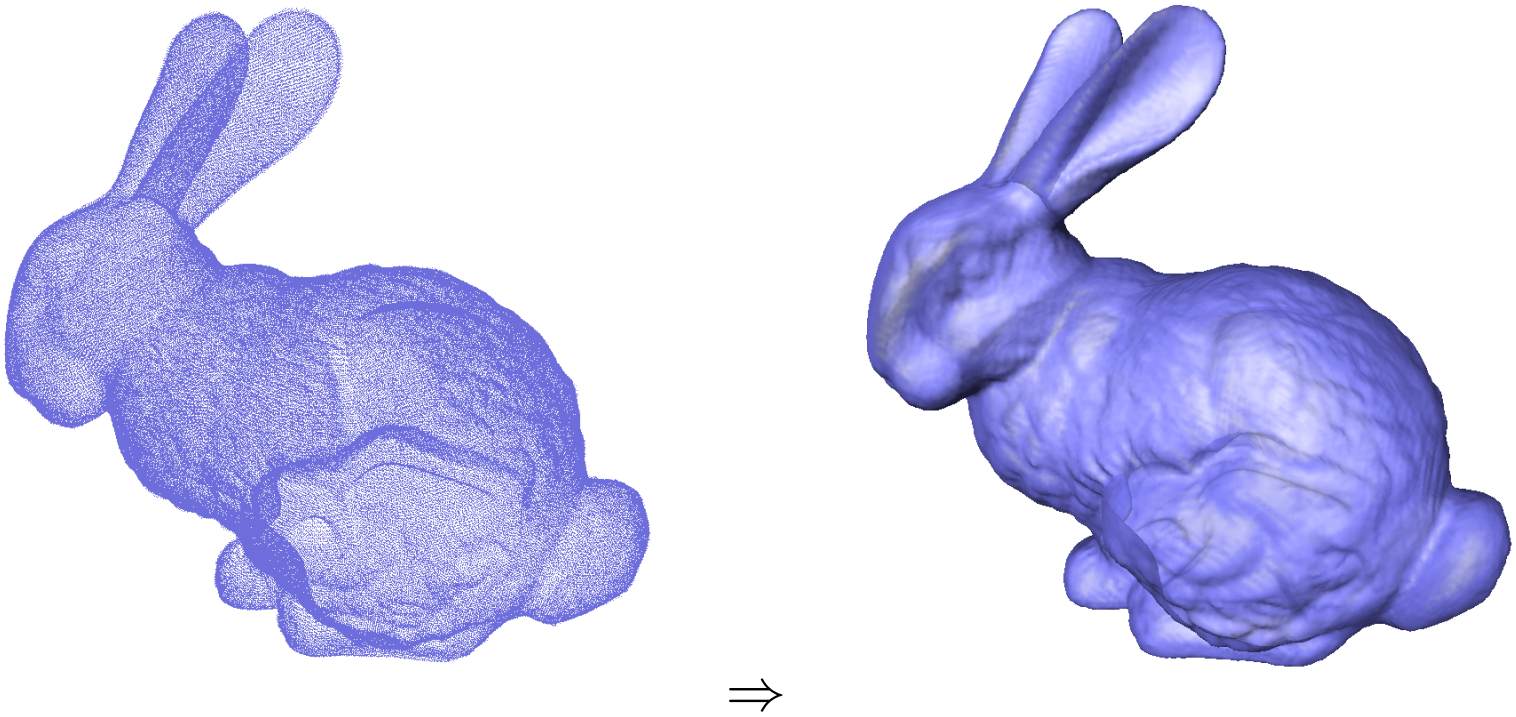
- Compute the disparity map from two aligned images



- Labels correspond to the disparities, weights to similarity coefficients between blocks



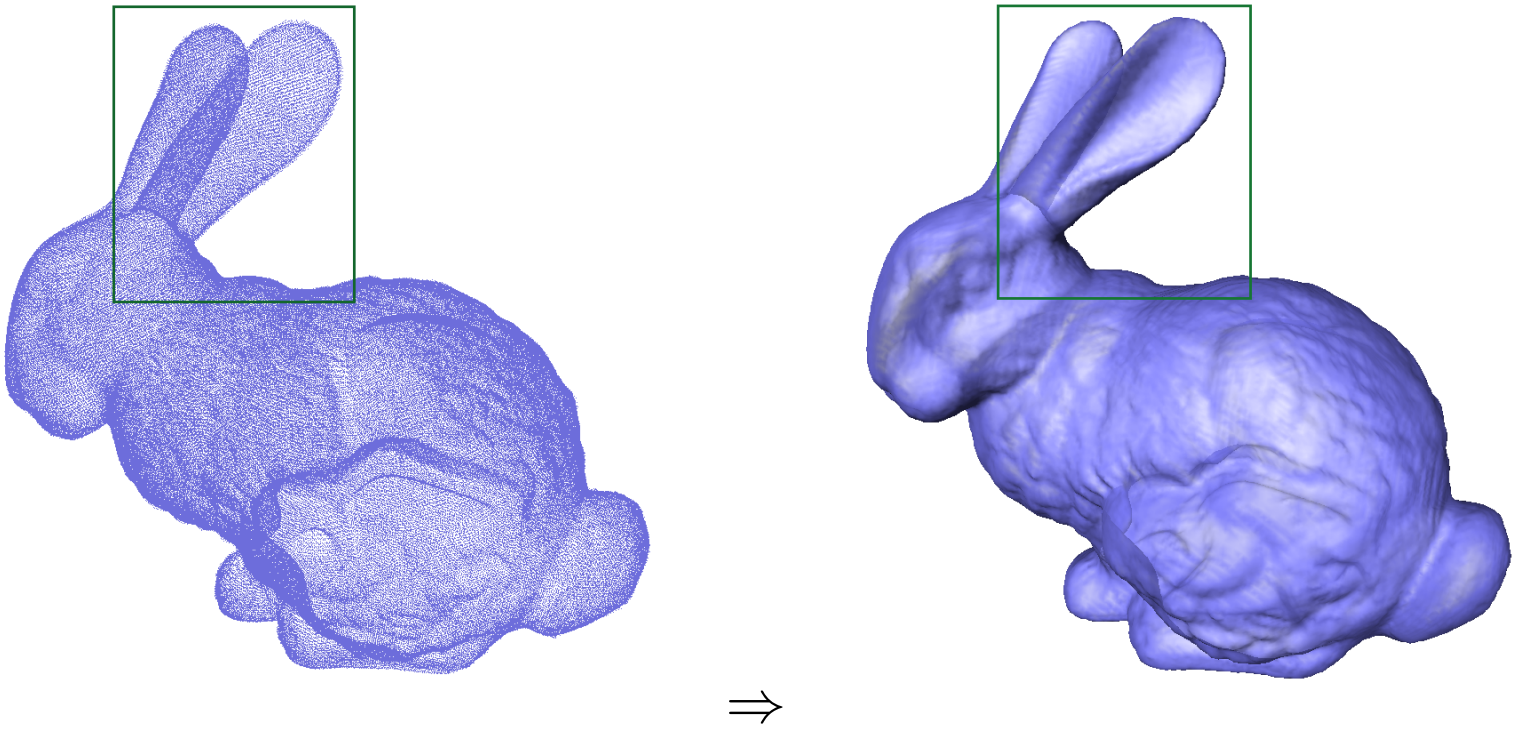
Question 6: Surface reconstruction from a noisy set of dots



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

Joint work with Xavier Bresson

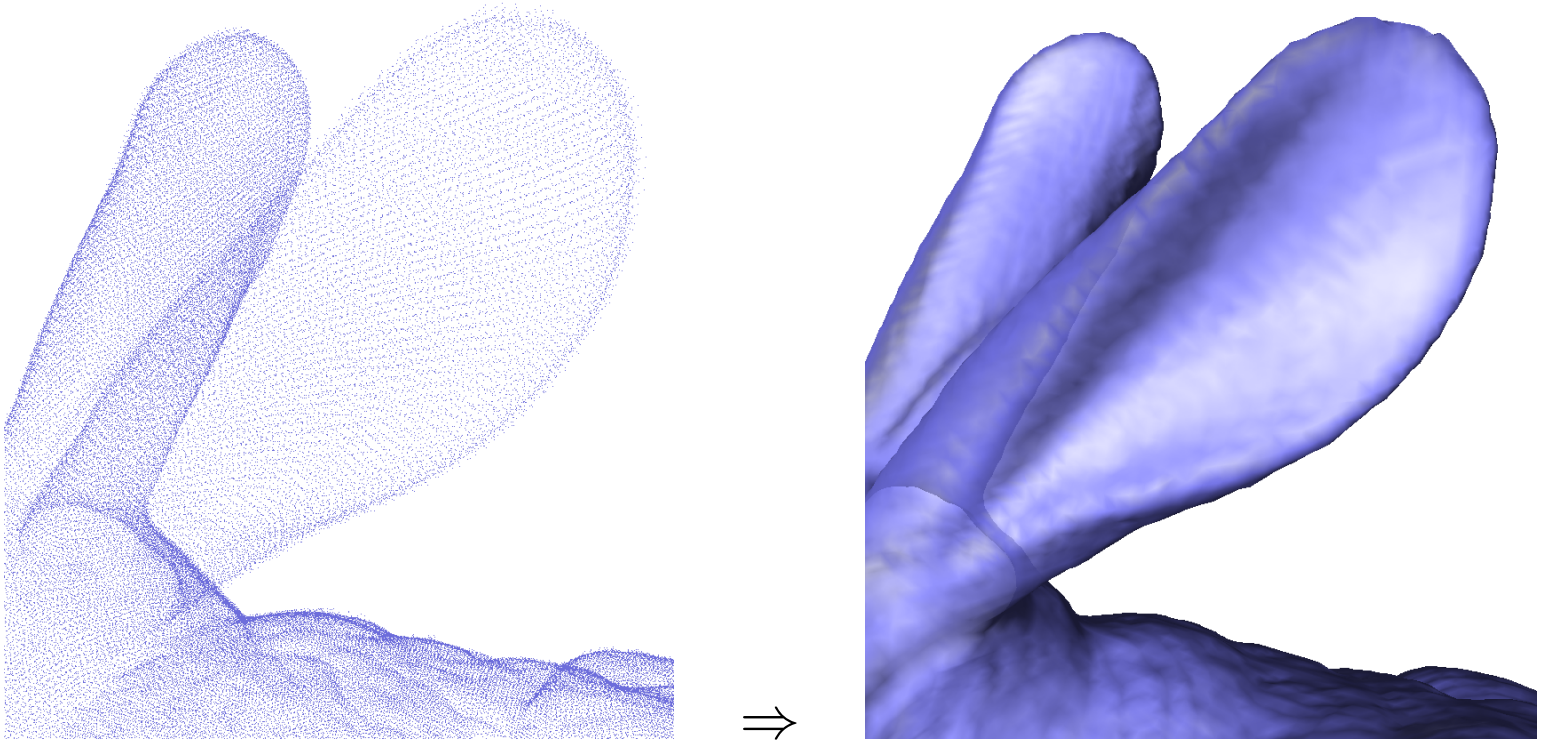
Question 6: Surface reconstruction from a noisy set of dots



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

Joint work with Xavier Bresson

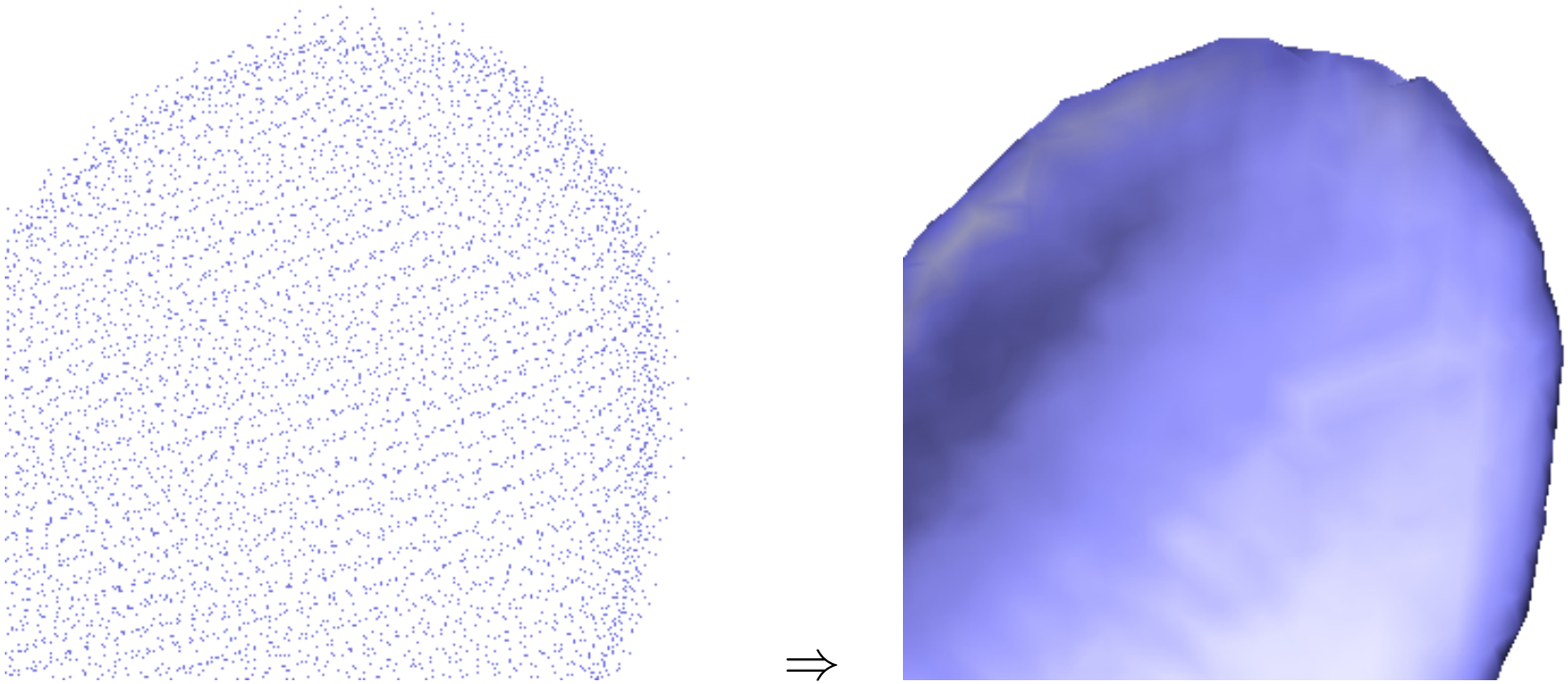
Surface reconstruction from a noisy set of dots



- 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

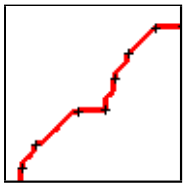
How to solve this problem

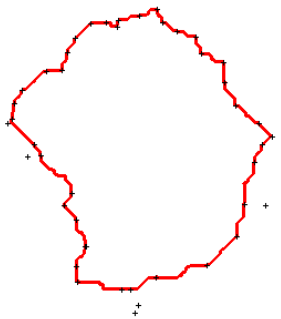
- Graph : 3D grid
- Here x represents the object indicator to recover.

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

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

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



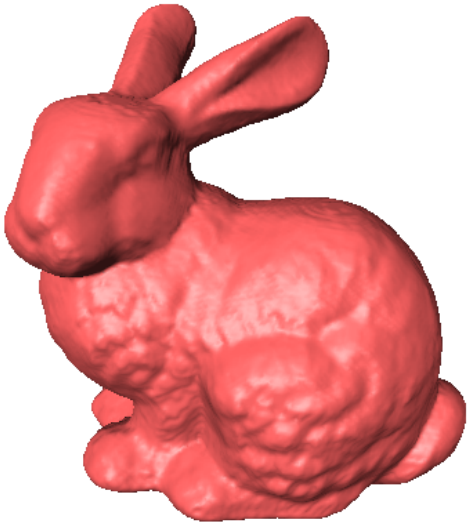


Power watershed solution

Why PW are a good fit for this problem ?

numerous plateaus around the dots to fit → smooth isosurface is obtained

Comparisons



Total variation

Size of required seeds

Graph cuts

Size of required seeds

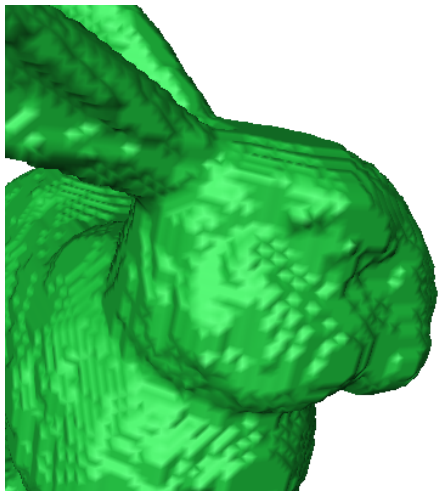
estimation required

Power watershed

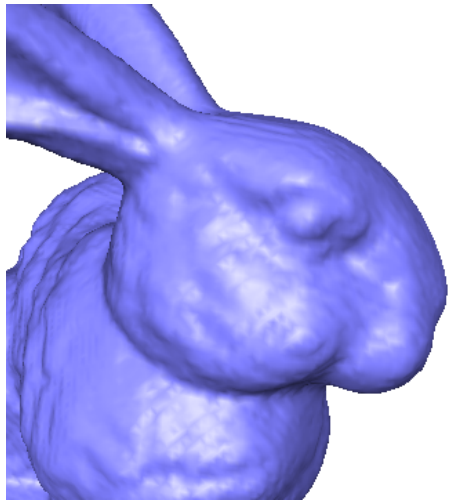
Size of required seeds

Comparisons

Total variation



Graph cuts



Power watershed

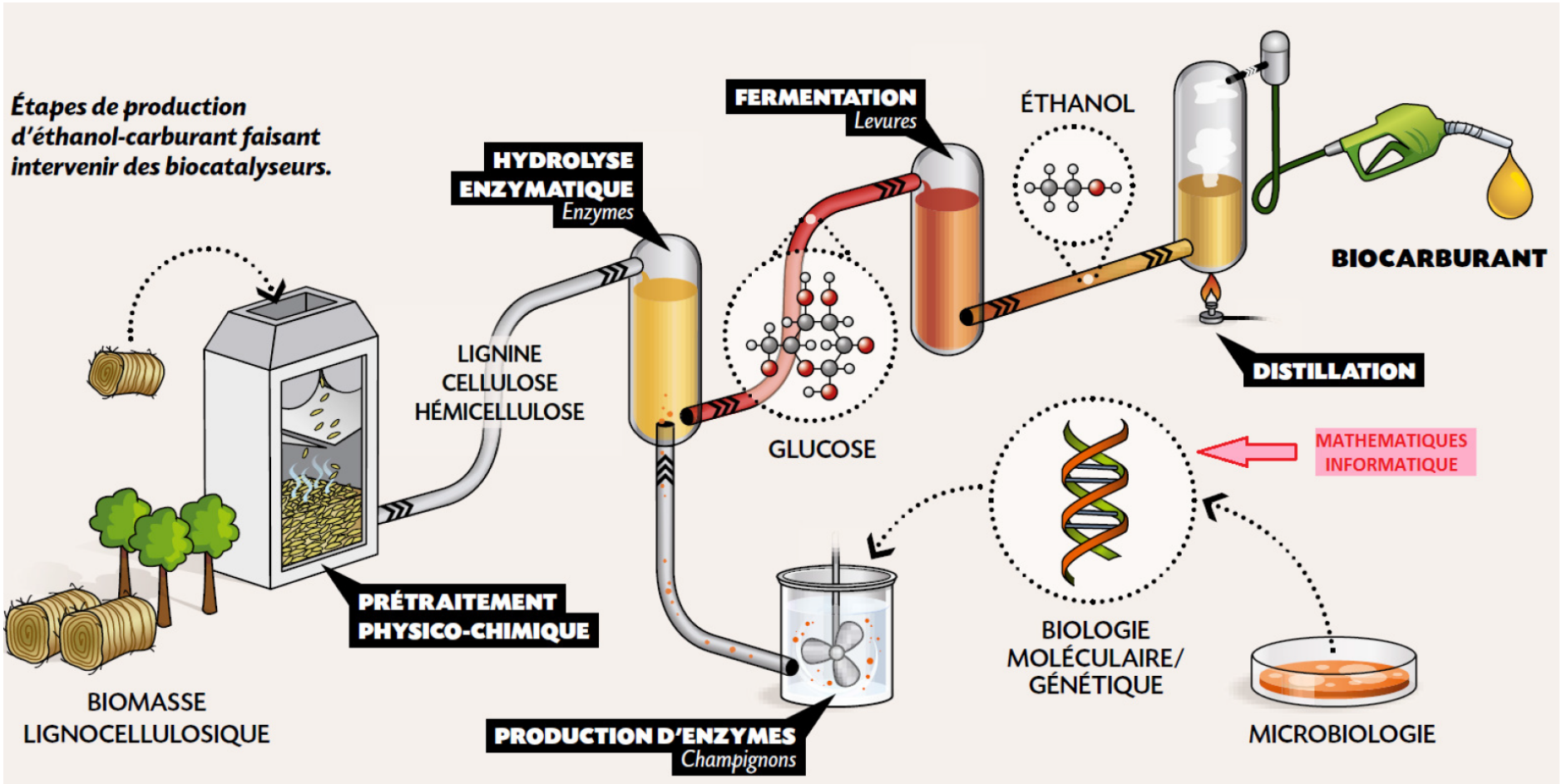
- Fast, accurate, globally optimal surface reconstruction from noisy set of dots
- Robust to markers placement
- No post-processing smoothing step

Biological Regularization A-priori for Network Inference (BRANE)

- I - Standard graph-based methods
- II - Unifying optimization Framework
- **III - Biological applications**
- IV - Image generation using adversarial networks

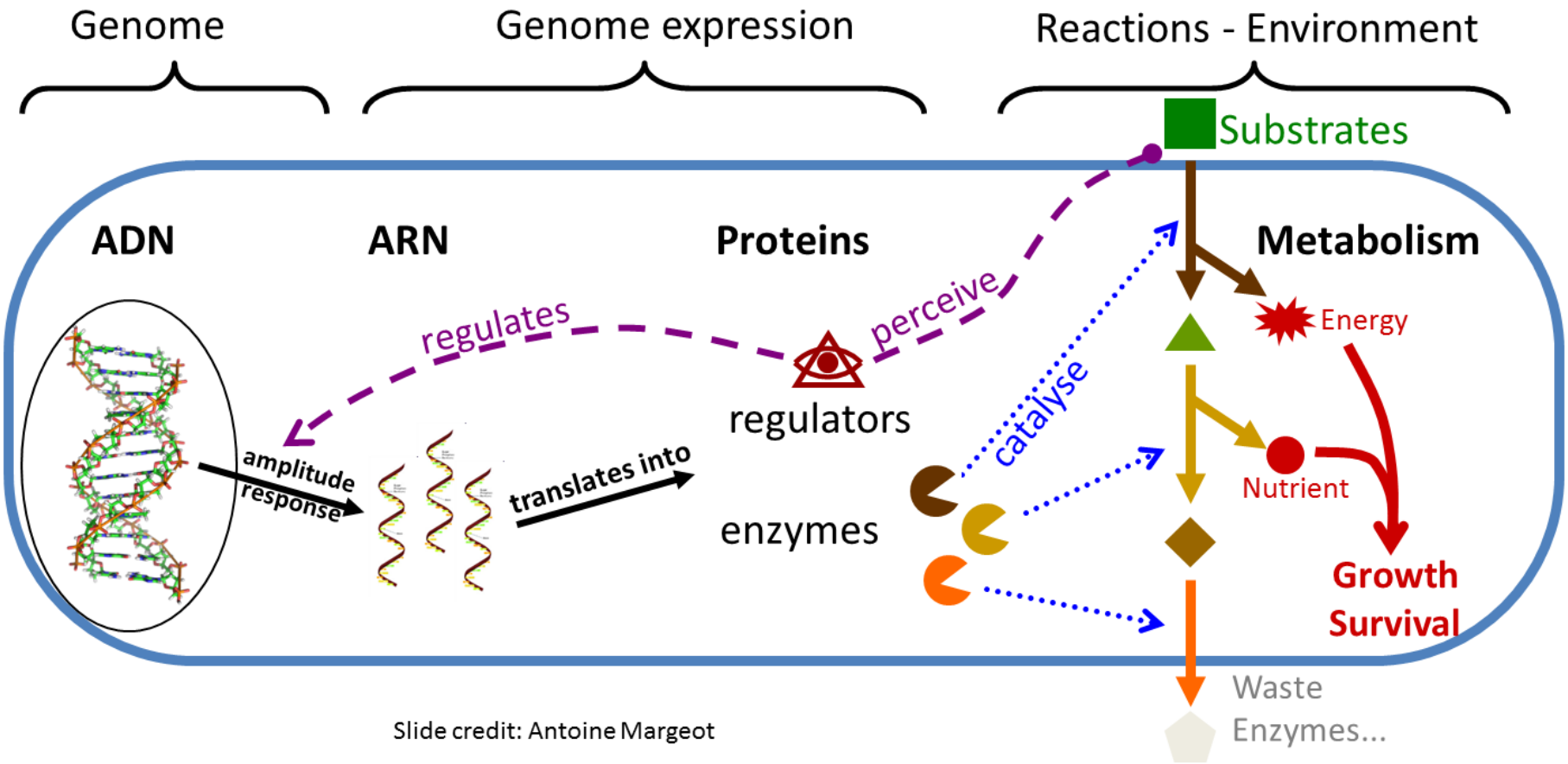
Joint work with Aurelie pirayre, Laurent Duval, Frederique Bidard and Jean-Christophe Pesquet

Context: Second generation of biofuel production



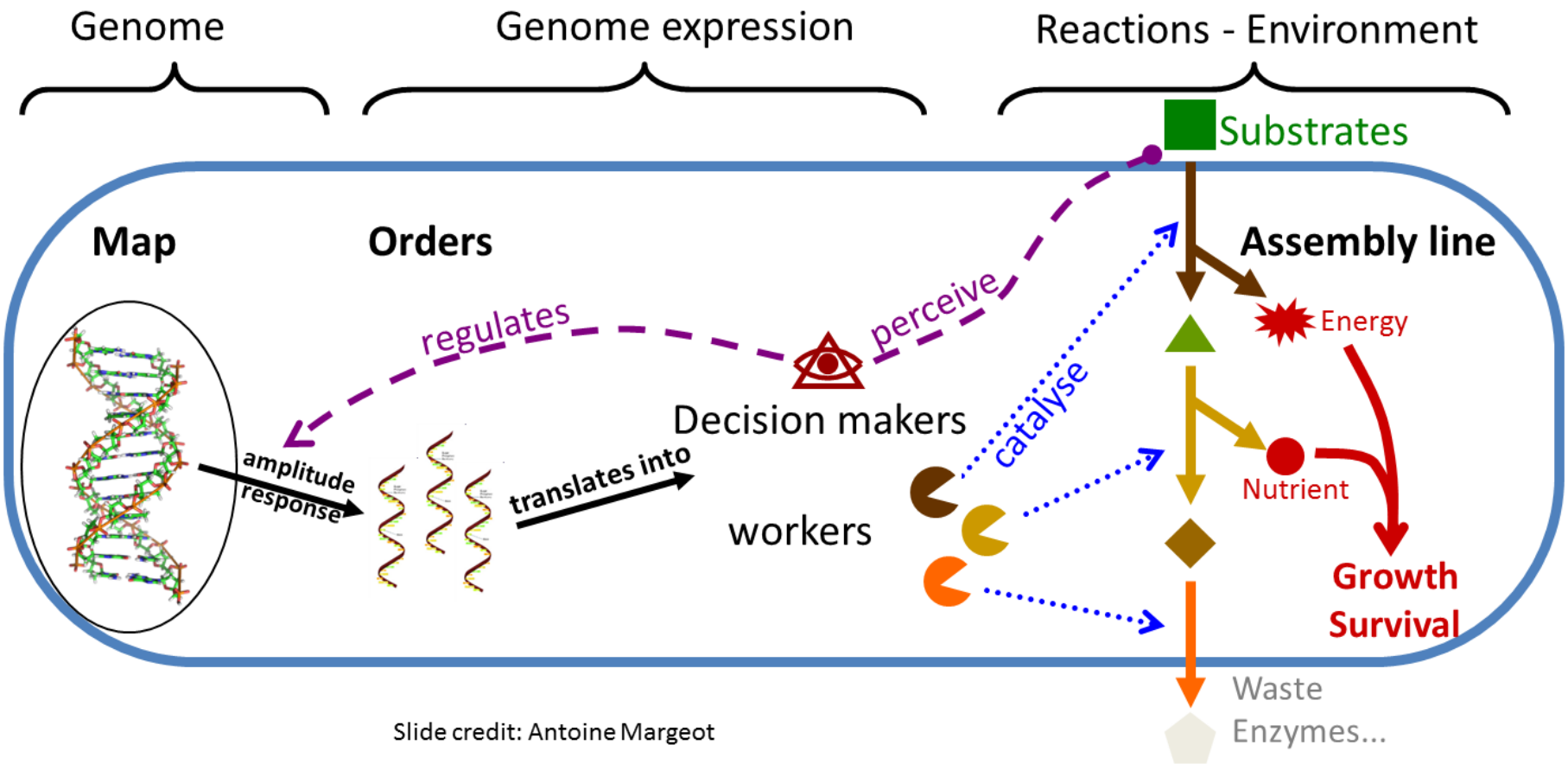
Slide credit: IFPEN

A micro-organism: how does it work?



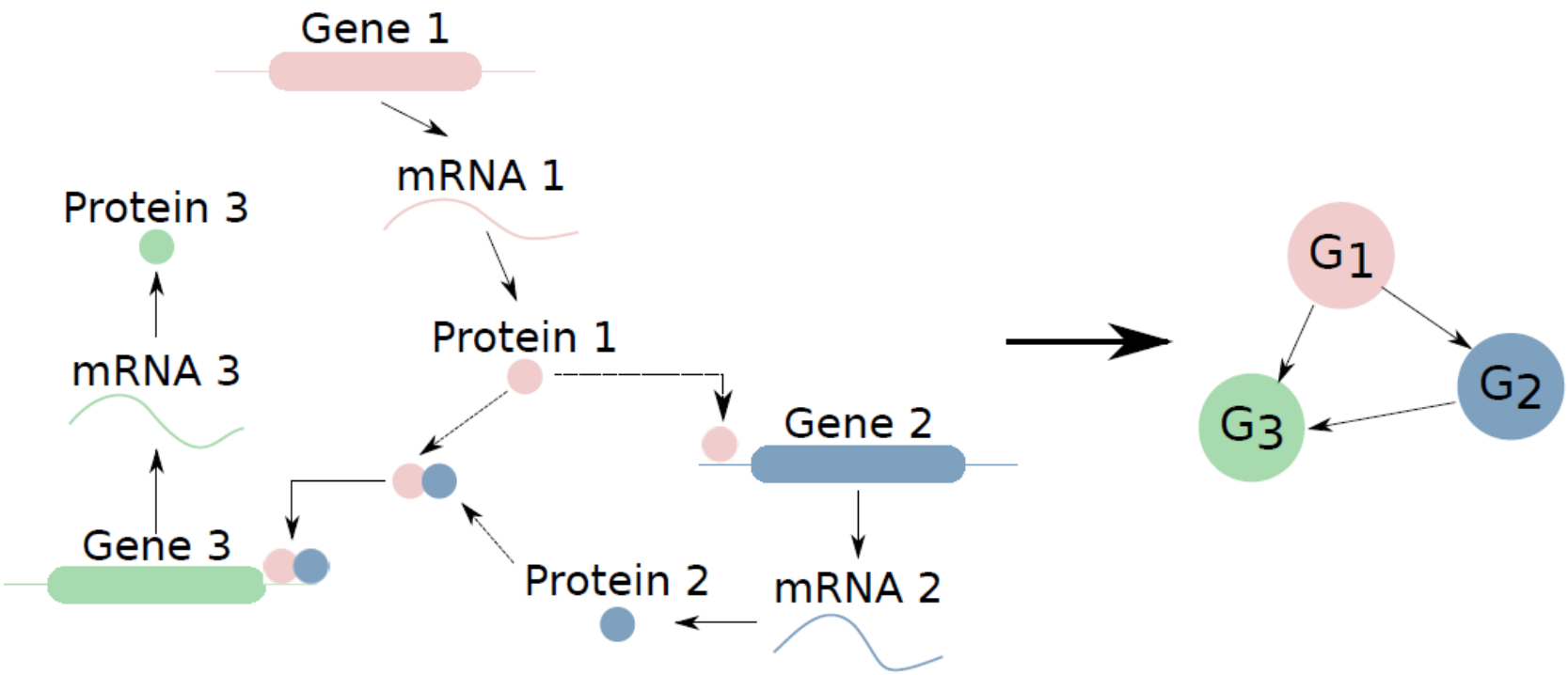
Slide credit: Antoine Margeot

A limited but useful analogy: a factory



Slide credit: Antoine Margeot

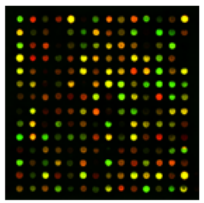
Regulation network



Transcription factors (TF) are proteins that regulate some gene expressions.

Slide credit: Pierre Geurts

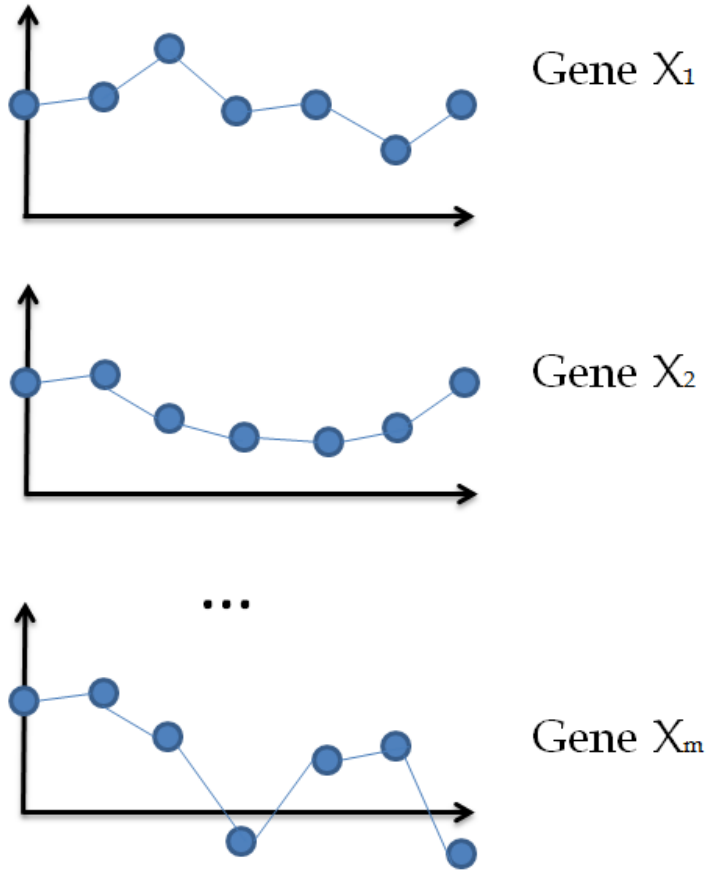
Overview of our gene expression data



$\sim$ 10000 genes

X_1	X_2	$\cdots$	X_m
-0.61	0.41	$\cdots$	0.51
-2.3	0.1	$\cdots$	-0.21
0.33	-0.45	$\cdots$	0.3
0.23	0.87	$\cdots$	0.09
$\cdots$	$\cdots$	$\cdots$	$\cdots$
-0.69	-0.61	$\cdots$	0.02

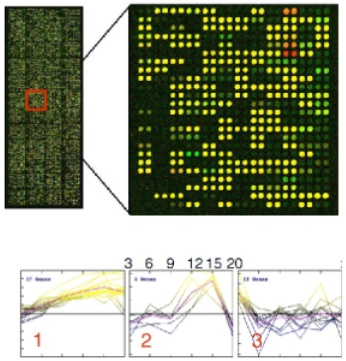
$\sim$ 100 cond.



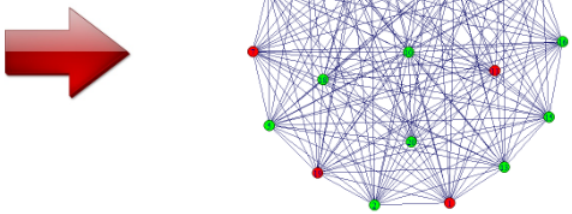
Slide credit: Van anh Huynh-Thu

Overview of our problem

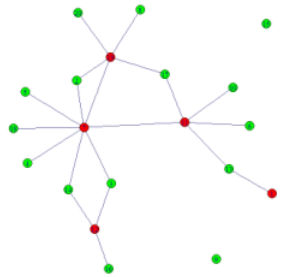
Transcriptomic data



Complete Graph weighted by pairwise gene similarity measure w



Inferred GRN (Gene Regulatory Network)



An edge selection problem

- We note $x_{i,j}$ the **binary label of edge presence**: $\forall (i,j) \in \mathbb{V}^2$

$$x_{i,j} = \begin{cases} 1 & \text{if } e_{i,j} \in \mathcal{E}^*, \\ 0 & \text{otherwise.} \end{cases}$$

An edge selection problem

- We note $x_{i,j}$ the **binary label of edge presence**: $\forall (i,j) \in \mathbb{V}^2$

$$x_{i,j} = \begin{cases} 1 & \text{if } e_{i,j} \in \mathcal{E}^*, \\ 0 & \text{otherwise.} \end{cases}$$

- Thresholding cost function for given weights ω:

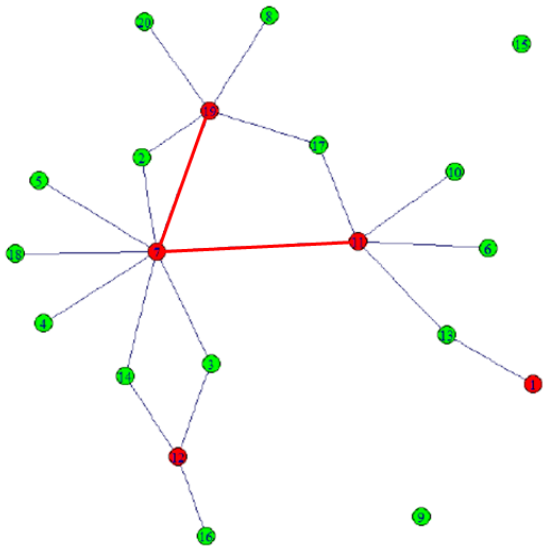
$$\underset{x \in \{0,1\}^n}{\text{maximize}} \quad \sum_{(i,j) \in \mathbb{V}^2} \omega_{i,j}\, x_{i,j} + \lambda(1 - x_{i,j})$$

- Explicit form:

$$x^*_{i,j} = \begin{cases} 1 & \text{if } \omega_{i,j} > \lambda \\ 0 & \text{otherwise.} \end{cases}$$

What additional knowledge may we include?

- Very often, a list of putative transcription factors (TFs) is known by biologists
- **Observation 1: Regulation type statistics** Regulation relationships between two TFs are less frequent that others
- Example: on this graph: 2 TF-TF edges / 20 edges (ratio TF/nonTF genes = $4/13 > 0.3 >> 0.1$)



- Question 7: How to define gene network inference as an optimization problem using Observation 1?

The generalized cost function

- Selecting strongly weighted edges

The generalized cost function

- Selecting strongly weighted edges
- Favoring the selection of edges involving one TF (Obs. 1)
  $\lambda_{i,j}$ corresponds to a weight depending whether the genes i and/or j are Transcription Factors (TF) or not.

$$\min_{x \in \mathbb{R}^N} \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} w_{i,j} |x_{i,j} - 1|$$

$$\lambda_{i,j} = \begin{cases} 2\lambda_{TF} & \text{if } i \in \mathcal{T} \text{ and } j \in \mathcal{T} \\ \lambda_{TF} + \lambda_{\bar{TF}} & \text{otherwise.} \end{cases}, \text{ with } \lambda_{TF} > \lambda_{\bar{TF}}.$$

The generalized cost function

- Selecting strongly weighted edges
- Favoring the selection of edges involving one TF (Obs. 1)
  $\lambda_{i,j}$ corresponds to a weight depending whether the genes i and/or j are Transcription Factors (TF) or not.

$$\min_{x \in \mathbb{R}^N} \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} w_{i,j} |x_{i,j} - 1| + \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} \lambda_{i,j} x_{i,j}$$

$$\lambda_{i,j} = \begin{cases} 2\lambda_{TF} & \text{if } i \in \mathcal{T} \text{ and } j \in \mathcal{T} \\ \lambda_{TF} + \lambda_{\bar{TF}} & \text{otherwise.} \end{cases}, \text{ with } \lambda_{TF} > \lambda_{\bar{TF}}.$$

The generalized cost function

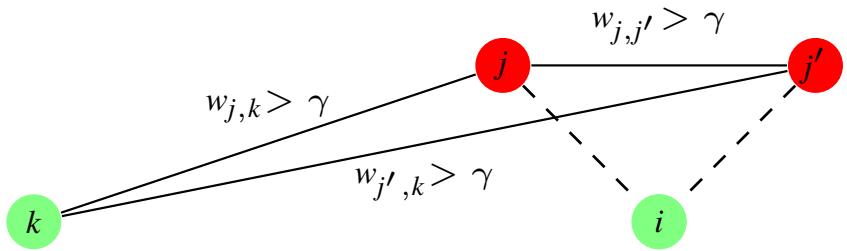
- Selecting strongly weighted edges
- Favoring the selection of edges involving one TF (Obs. 1)
  $\lambda_{i,j}$ corresponds to a weight depending whether the genes i and/or j are Transcription Factors (TF) or not.

$$\min_{x\in\mathbb{R}^N} \sum_{(i,j)\in\mathcal{V}\times\mathcal{V}} w_{i,j}|x_{i,j}-1| + \sum_{(i,j)\in\mathcal{V}\times\mathcal{V}} \lambda_{i,j}x_{i,j} + \mu\Phi(\mathcal{N}_{i,j}),$$

$$\lambda_{i,j} = \begin{cases} 2\lambda_{TF} & \text{if } i \in \mathcal{T} \text{ and } j \in \mathcal{T} \\ \lambda_{TF} + \lambda_{\bar{TF}} & \text{otherwise.} \end{cases}, \text{ with } \lambda_{TF} > \lambda_{\bar{TF}}.$$

What additional knowledge may we include?

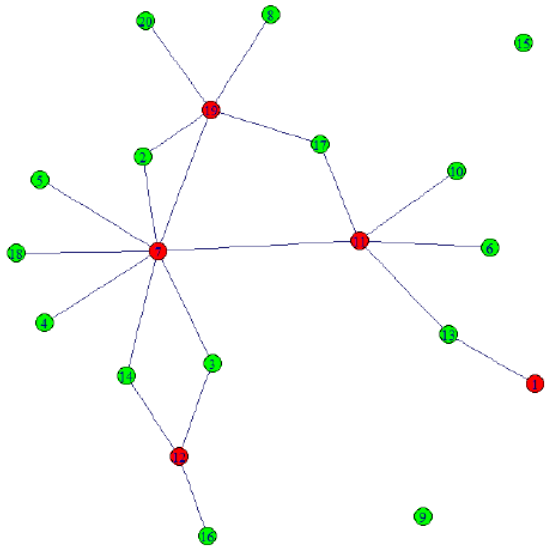
- **Observation 2: Coupling property** If two transcription factors are co-regulated and regulate at least one gene, it is probable that any other gene (not) regulated by one of these TF is (not) regulated by the other.



Question 8: How to incorporate a gene coupling a-priori?

What additional knowledge may we include?

- **Observation 3: Average Connectivity** Non TF genes are not regulated by a large number of genes. Typically, the average degree is of 3.
- Example: on this graph, the average degree of non-TF genes is 1.5.



- Question 9: how to incorporate an average degree a-priori?

The generalized cost function

How to define an appropriate cost function?

The generalized cost function

How to define an appropriate cost function?

- Selecting strongly weighted edges

$$\min_{x \in \mathbb{R}^N} \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} w_{i,j} |x_{i,j} - 1|$$

The generalized cost function

How to define an appropriate cost function?

- Selecting strongly weighted edges
- Favoring the selection of edges involving one TF (Obs. 1)
  $\lambda_{i,j}$ corresponds to a weight depending whether the genes i and/or j are Transcription Factors (TF) or not.

$$\min_{x \in \mathbb{R}^N} \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} w_{i,j} |x_{i,j} - 1| + \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} \lambda_{i,j} x_{i,j}$$

The generalized cost function

How to define an appropriate cost function?

- Selecting strongly weighted edges
- Favoring the selection of edges involving one TF (Obs. 1)
  $\lambda_{i,j}$ corresponds to a weight depending whether the genes i and/or j are Transcription Factors (TF) or not.
- Structural a priori (Observations 2 and 3)

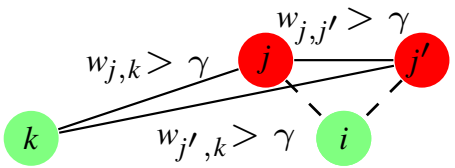
$$\min_{x \in \mathbb{R}^N} \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} w_{i,j} |x_{i,j} - 1| + \sum_{(i,j) \in \mathcal{V} \times \mathcal{V}} \lambda_{i,j} x_{i,j} + \mu \Phi(\mathcal{N}_{i,j}),$$

where

- $\Phi(\mathcal{N}_{i,j})$ denotes a structural a priori involving the local neighborhood $\mathcal{N}_{i,j}$ of $e_{i,j}$
- μ is a regularization parameter

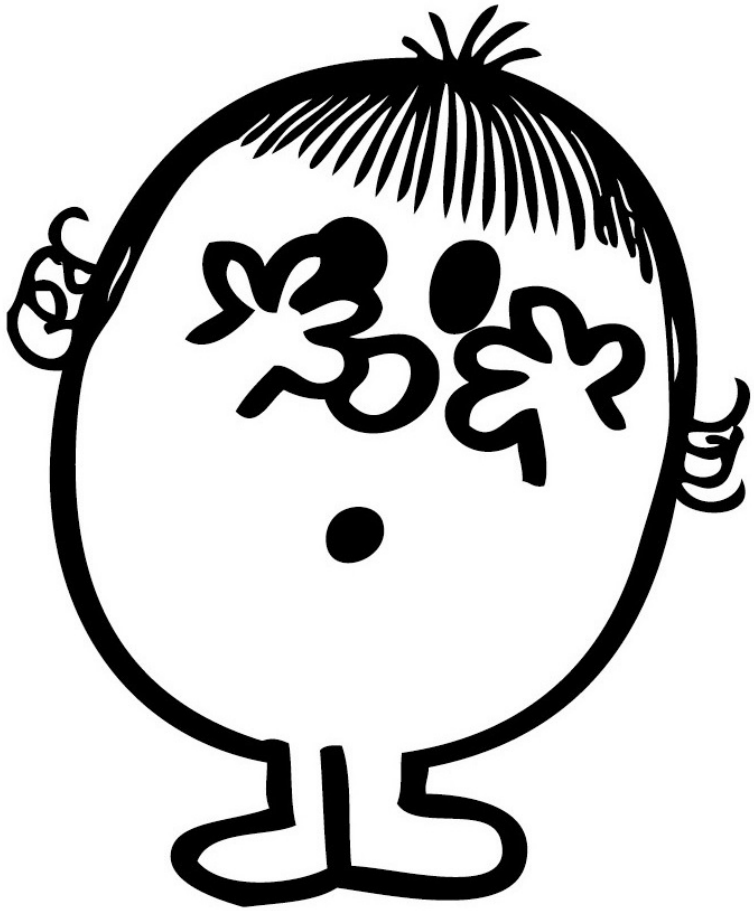
Structural a priori

Let $\mathcal{T} \subset \mathcal{V}$ be a set of transcription factors (TFs)

Method	BRANE cut	BRANE Relax
Structural a priori	Coupling property	Average connectivity
Principle		Genes are usually regulated by a small number d of TFs.
Mathematical form	$\Phi(\mathcal{N}_{i,j}) = \sum_{\substack{i \in \mathcal{V} \setminus \mathcal{T}, \\ (j,j') \in \mathcal{T} \times \mathcal{T}}} \alpha_{i,j,j'} \lvert x_{i,j} - x_{i,j'} \rvert$	$\Phi(\mathcal{N}_{i,j}) = \sum_{i \in \mathcal{V} \setminus \mathcal{T}} \left(\sum_{j=1}^{g} x_{i,j} - d \right)^2$
Optimization strategy	Discrete (Maximal Flow)	Relaxed (Forward-Backward)

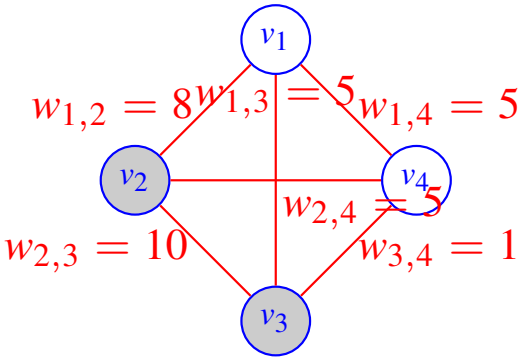
The discrete one: BRANE cut

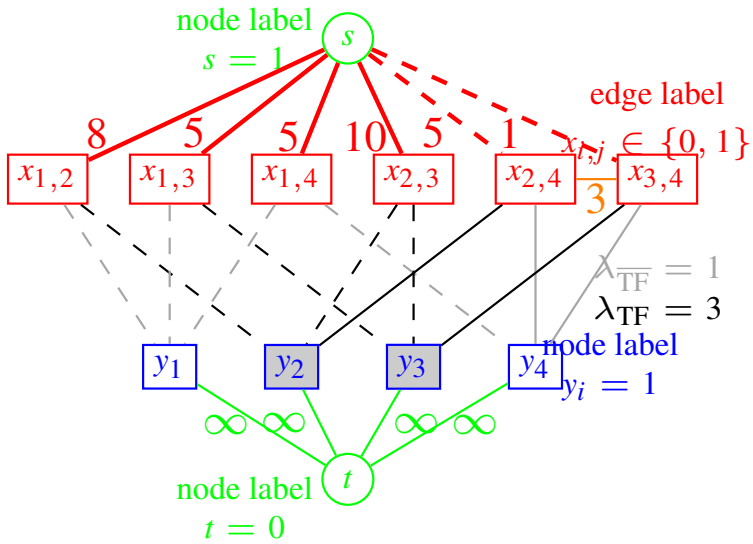
We know how to obtain a discrete solution for x, where $x \in \{0, 1\}^N$

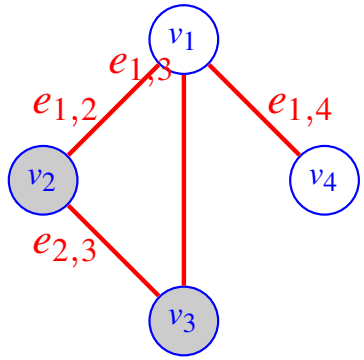


Maximal flow algorithm in BRANE cut

$$\underset{\mathbf{x}\in\{0,1\}^N}{\text{minimize}} \sum_{(i,j)\in\mathcal{V}\times\mathcal{V}} w_{i,j}|x_{i,j}-1| \;+\; \sum_{(i,j)\in\mathcal{V}\times\mathcal{V}} \lambda_{i,j}x_{i,j} \;+\; \mu \sum_{\substack{i\in\mathcal{V}\setminus\mathcal{T},\\ (j,j')\in\mathcal{T}\times\mathcal{T}}} \alpha_{i,j,j'}|x_{i,j}-x_{i,j'}|$$

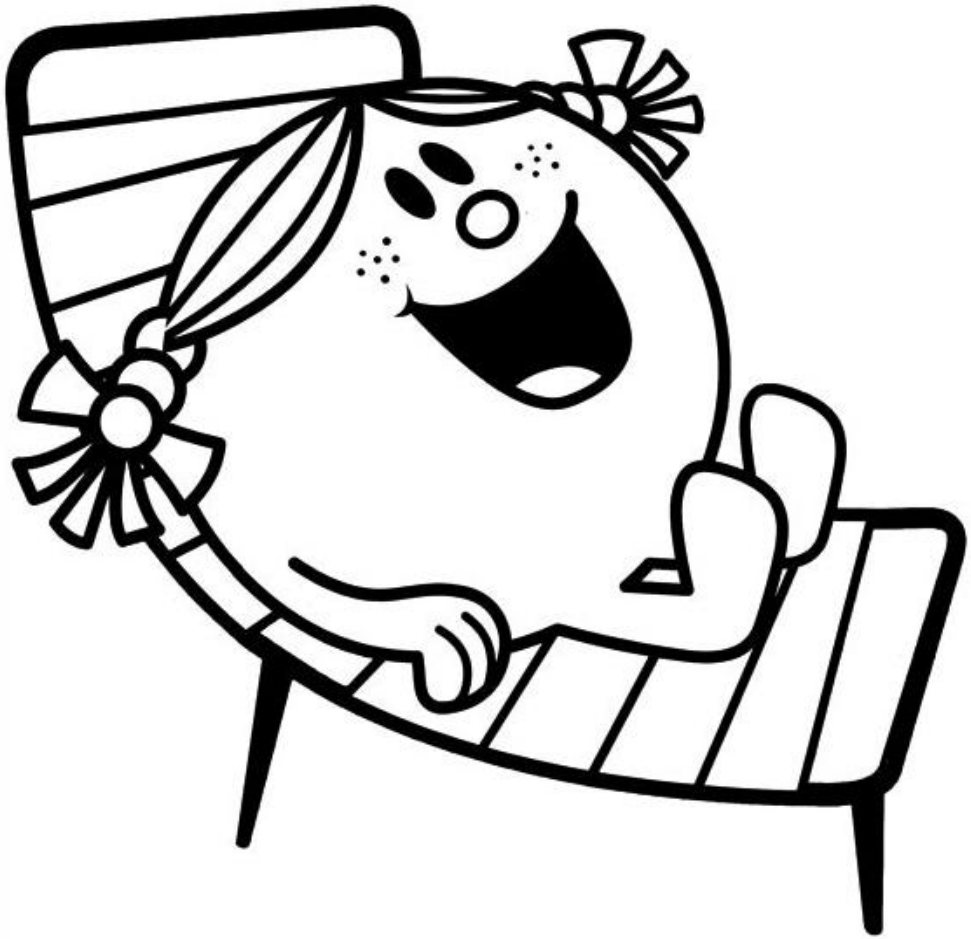






The relaxed one: BRANE relaxed

We have to relax x, that is to say $x \in [0, 1]^N$



Forward-backward algorithm in BRANE relax

$$\underset{x\in[0,1]^N}{\text{minimize}} \underbrace{w^\top(\mathbb{1}-x)+\lambda^\top x+\mu\,\|\Omega x-d\|^2}_{f_1(x)}$$

Algorithm 1: Projected gradient descent algorithm

Fix $x_0 \in \mathbb{R}^N$;

for $n = 0, 1, \ldots$ **do**

$$y_n^{(n)} = x_n^{(n)} - \gamma_n \quad \nabla f_1(x_n);$$
$$x_{n+1}^{(k_n)} = P_{[0,1]^N}(y_n^{(n)}) \ ;$$

Forward-backward algorithm in BRANE relax

$$\underset{x\in[0,1]^N}{\text{minimize}} \underbrace{w^\top(\mathbb{1}-x)+\lambda^\top x+\mu\left\|\Omega x-d\right\|^2}_{f_1(x)}+\underbrace{\iota_{[0,1]^N}(x)}_{f_2(x)}$$

Algorithm 6: Forward-Backward algorithm (general version)

Fix $x_0 \in \mathbb{R}^N$;

for $n = 0, 1, \ldots$ **do**

$$y_n^{(n)} = x_n^{(n)} - \gamma_n \quad \nabla f_1(x_n);$$
$$x_{n+1}^{(n)} = \text{prox}_{\gamma_n \quad f_2^{(n)}}(y_n^{(n)}) ;$$

Forward-backward algorithm in BRANE relax

$$\underset{x\in[0,1]^N}{\text{minimize}} \underbrace{w^\top(\mathbb{1}-x)+\lambda^\top x+\mu\left\|\Omega x-d\right\|^2}_{f_1(x)}+\underbrace{\iota_{[0,1]^N}(x)}_{f_2(x)}$$

Algorithm 11: Accelerated Forward-Backward algorithm

Fix $x_0 \in \mathbb{R}^N$;

for $n = 0, 1, \ldots$ **do**

$$y_n^{(n)} = x_n^{(n)} - \gamma_n A_n^{-1} \nabla f_1(x_n);$$
$$x_{n+1}^{(n)} = \text{prox}_{\gamma_n^{-1} A_n, f_2^{(n)}}(y_n^{(n)}) ;$$

[Chouzenoux et al., J. Optim. Theory Appl. 2014]

Forward-backward algorithm in BRANE relax

$$\underset{x\in[0,1]^N}{\text{minimize}} \underbrace{w^\top(\mathbb{1}-x)+\lambda^\top x+\mu\left\|\Omega x-d\right\|^2}_{f_1(x)}+\underbrace{\iota_{[0,1]^N}(x)}_{f_2(x)}$$

Algorithm 16: Block Accelerated Forward-Backward algorithm

Fix $x_0 \in \mathbb{R}^N$;

for $n = 0, 1, \ldots$ **do**

Select the index $k_n \in \{1, \ldots, p\}$ of a block of variables

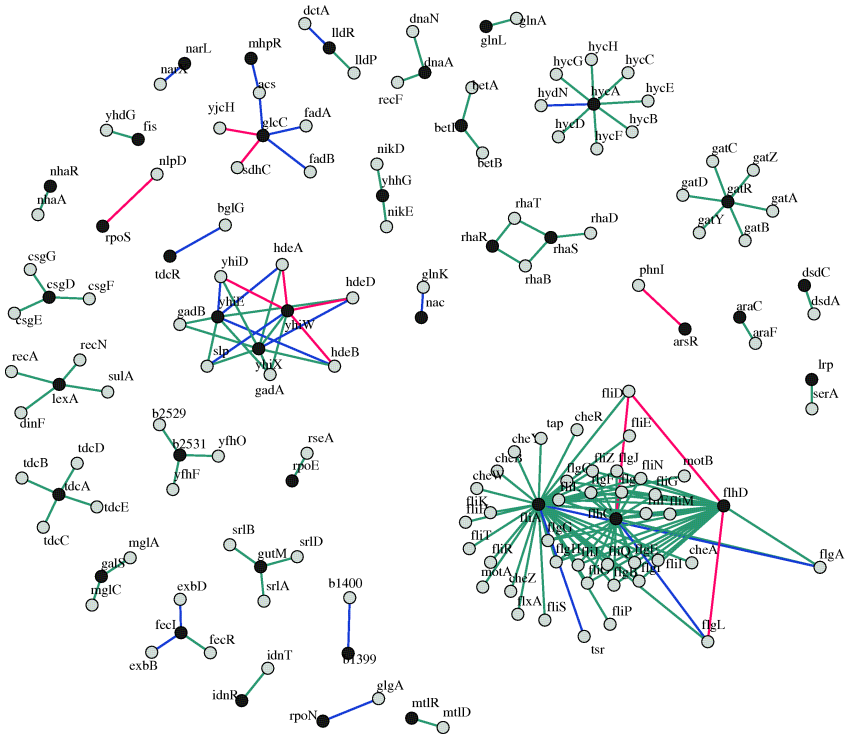
$y_n^{(k_n)} = x_n^{(k_n)} - \gamma_n A_{k_n}^{-1} \nabla f_1(x_n)$;

$x_{n+1}^{(k_n)} = \text{prox}_{\gamma_n^{-1} A_{k_n}, f_2^{(k_n)}}(y_n^{(k_n)})$;

$x_{n+1}^{(k)} = x_n^{(k)}, \quad k \in \{1, \ldots, p\} \setminus \{k_n\}$; [Chouzenoux et al. 2013]

Results

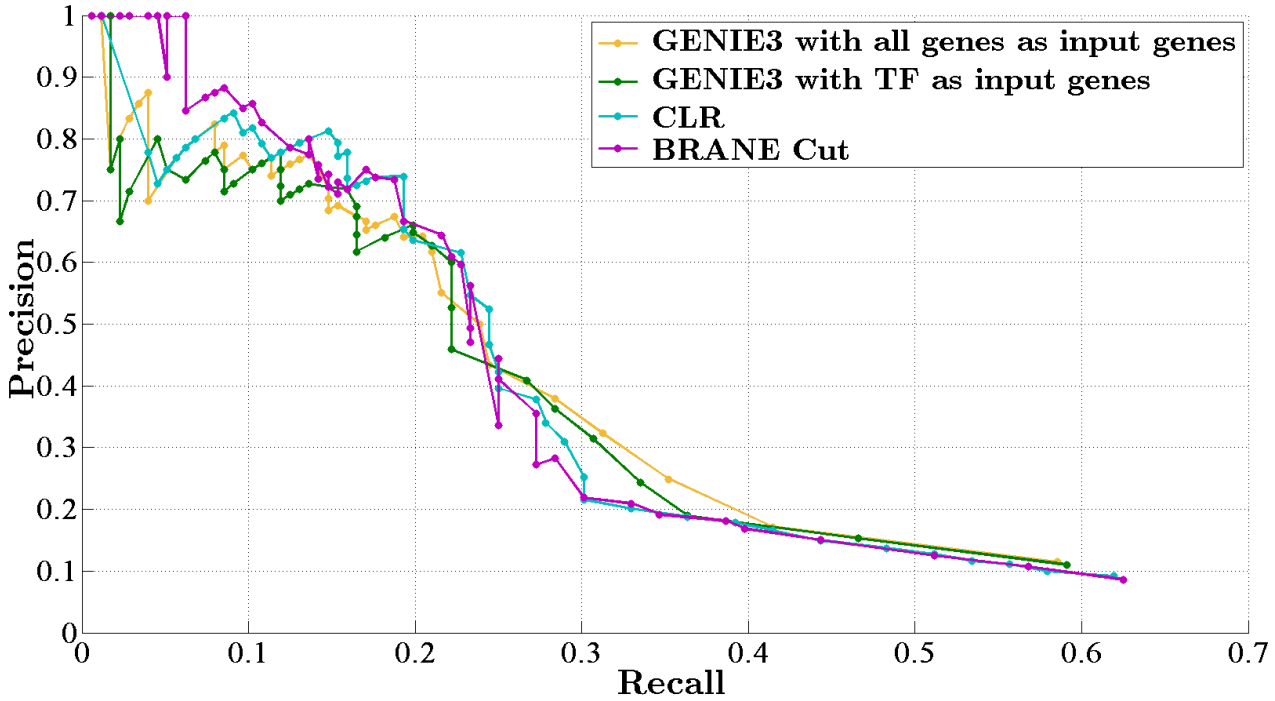
Network obtained with the E. Coli dataset



Legend: black nodes: transcription factors, gray nodes: other genes. green edges: inferred regulations also reported in the gold standard, blue edges: new inferred regulations that are also inferred by CLR, and pink edges: new inferred regulations.

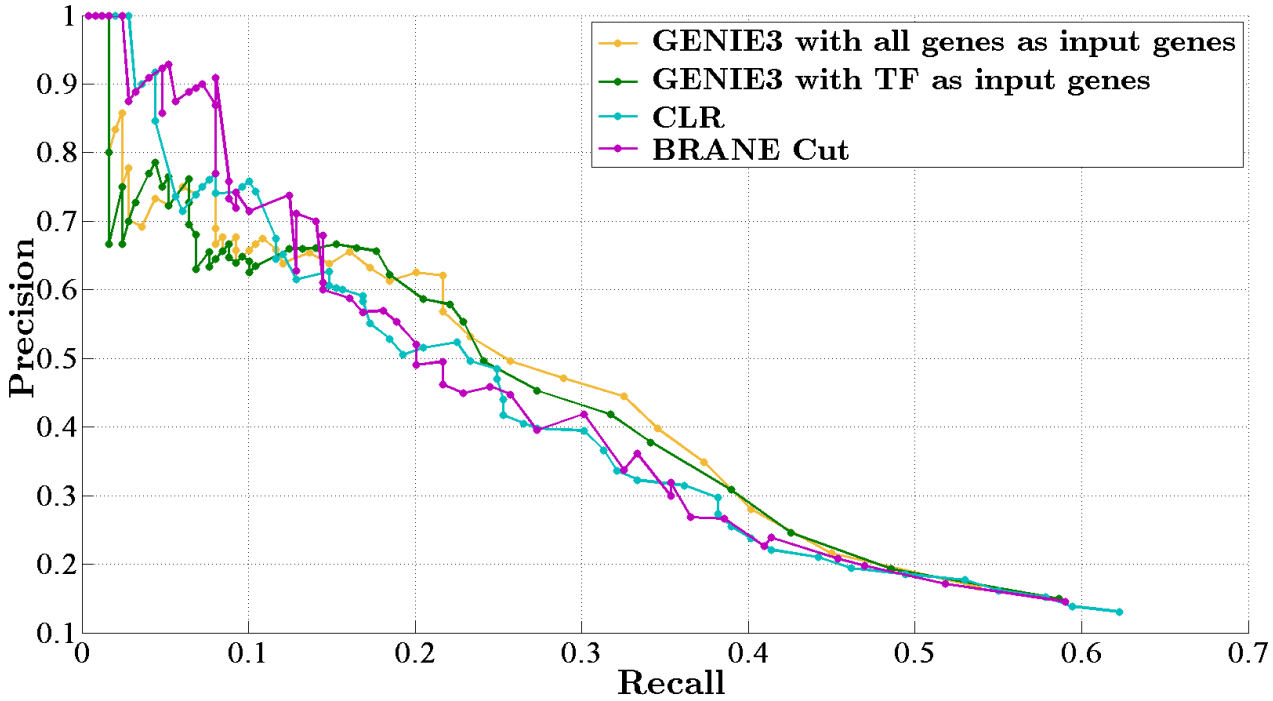
Results on DREAM4 data (multifactorial challenge)

Network index		1	2	3	4	5
AUPR	GENIE3[1]	0.239	0.260	0.316	0.301	0.295
	CLR	**0.249**	0.258	0.294	0.296	0.299
	BRANE Cut	**0.256**	**0.261**	**0.317**	**0.317**	**0.316**
	BRANE Relax	0.246	**0.264**	**0.321**	**0.317**	**0.317**



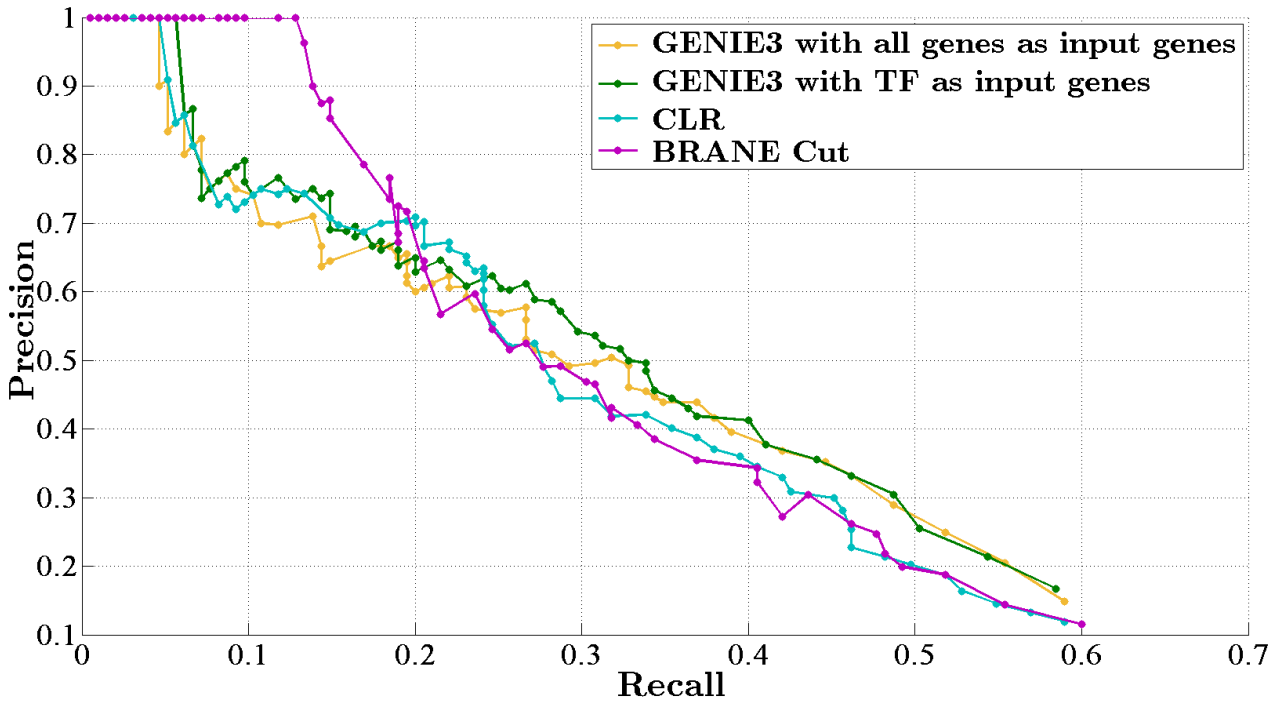
Results on DREAM4 data (multifactorial challenge)

Network index		1	2	3	4	5
AUPR	GENIE3[1]	0.239	0.260	0.316	0.301	0.295
	CLR	**0.249**	0.258	0.294	0.296	0.299
	BRANE Cut	**0.256**	**0.261**	**0.317**	**0.317**	**0.316**
	BRANE Relax	0.246	**0.264**	**0.321**	**0.317**	**0.317**



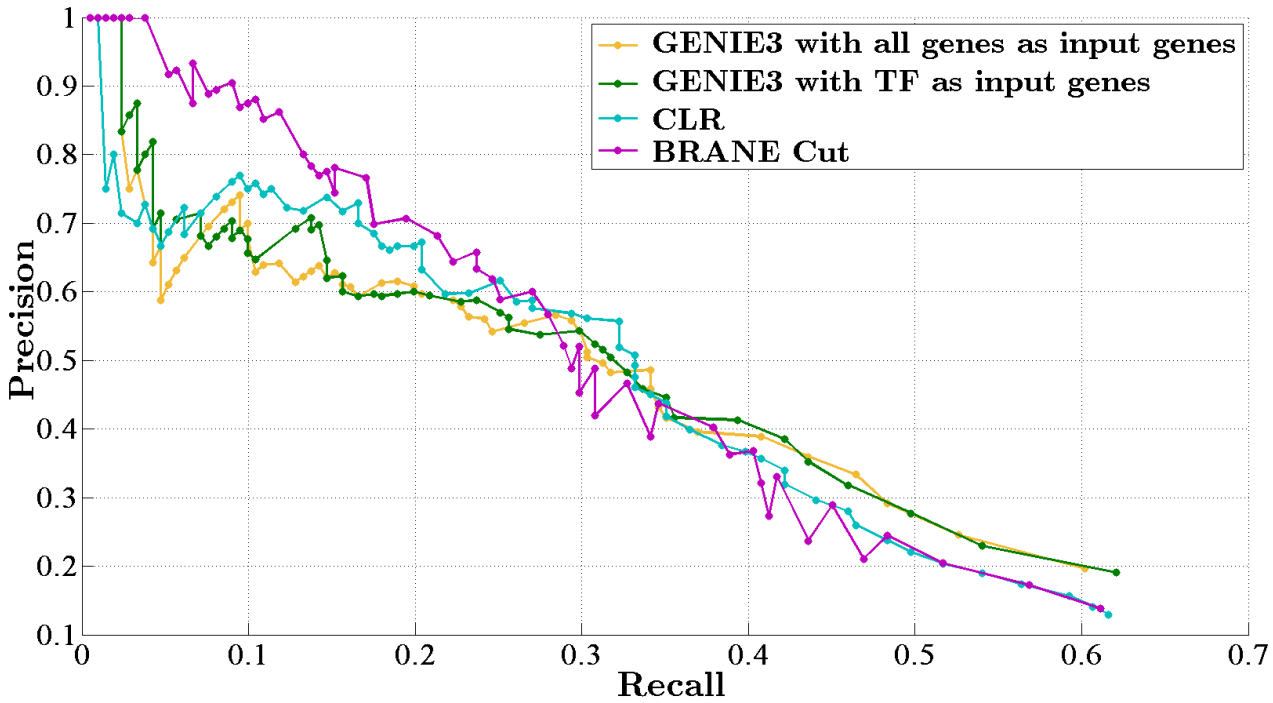
Results on DREAM4 data (multifactorial challenge)

Network index		1	2	3	4	5
AUPR	GENIE3[1]	0.239	0.260	0.316	0.301	0.295
	CLR	**0.249**	0.258	0.294	0.296	0.299
	BRANE Cut	**0.256**	**0.261**	**0.317**	**0.317**	**0.316**
	BRANE Relax	0.246	**0.264**	**0.321**	**0.317**	**0.317**



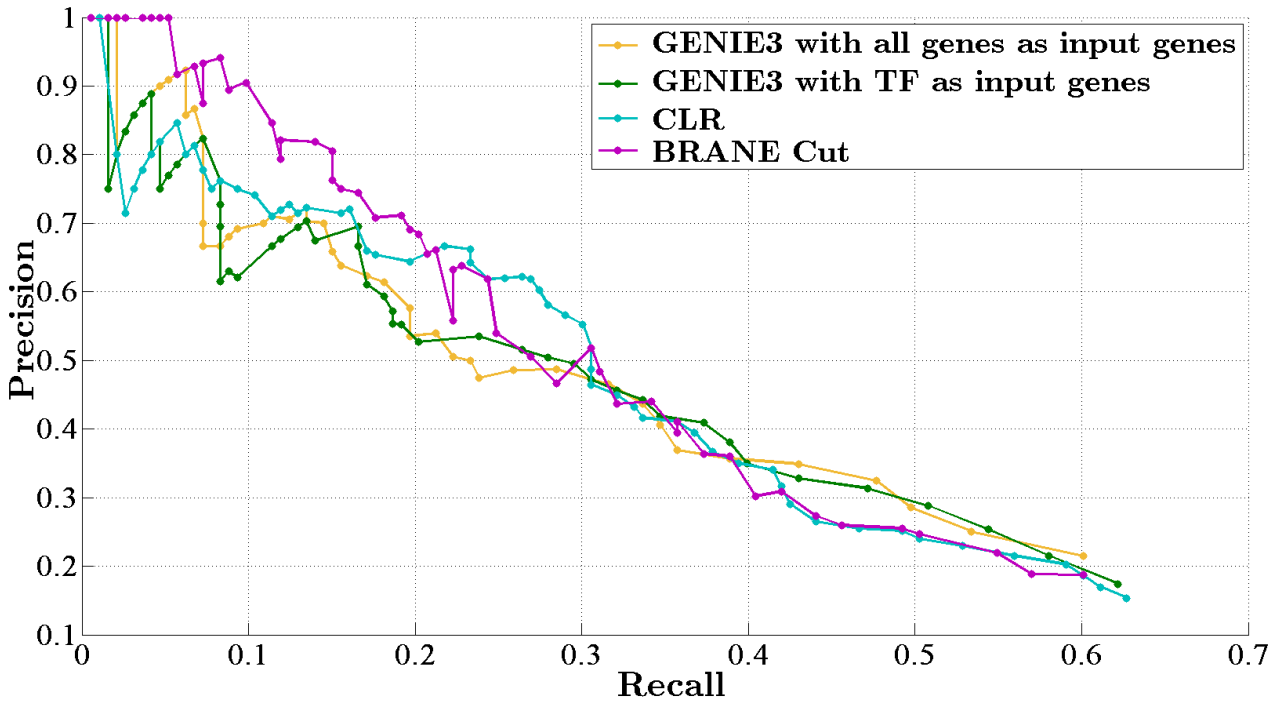
Results on DREAM4 data (multifactorial challenge)

Network index		1	2	3	4	5
AUPR	GENIE3[1]	0.239	0.260	0.316	0.301	0.295
	CLR	**0.249**	0.258	0.294	0.296	0.299
	BRANE Cut	**0.256**	**0.261**	**0.317**	**0.317**	**0.316**
	BRANE Relax	0.246	**0.264**	**0.321**	**0.317**	**0.317**



Results on DREAM4 data (multifactorial challenge)

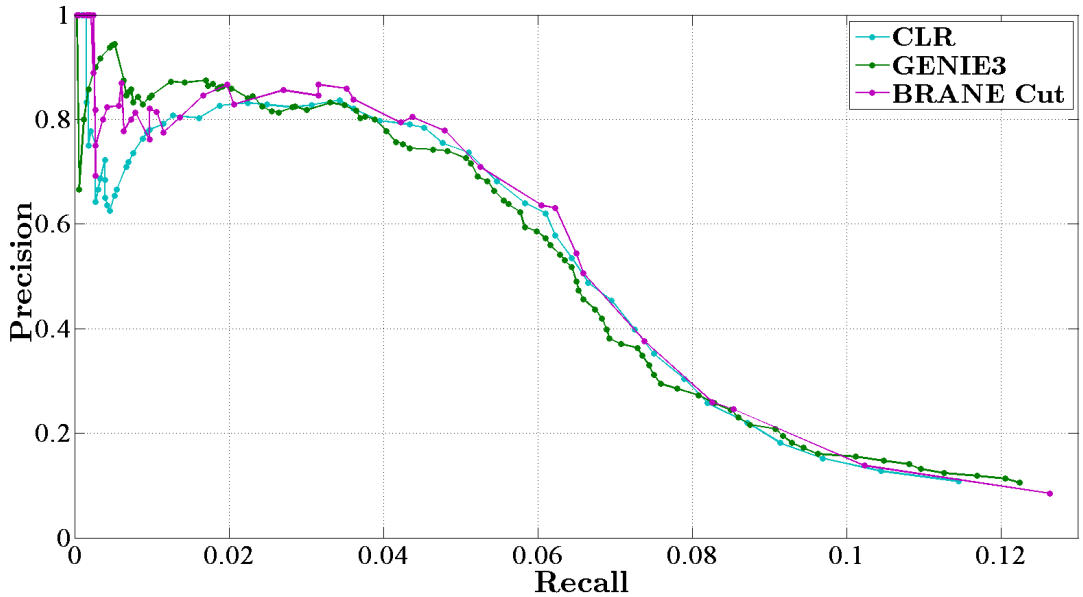
Network index		1	2	3	4	5
AUPR	GENIE3[1]	0.239	0.260	0.316	0.301	0.295
	CLR	**0.249**	0.258	0.294	0.296	0.299
	BRANE Cut	**0.256**	**0.261**	**0.317**	**0.317**	**0.316**
	BRANE Relax	0.246	**0.264**	**0.321**	**0.317**	**0.317**



BRANE Cut results on *E.coli* data

	CLR	GENIE3	BC^1	BC^2
AUPR ($\times 10^{-2}$)	6.11	6.31	6.39	**6.45**
Total comput. time (min)	30.0	420.0	30.0	30.1

	vs CLR	vs GENIE3
AUPR gain	5.9%	2.2%
Comput. time gain	none	7 $\times$ faster



Conclusions

Summary

- Incorporating biological contrains in a mathematical optimization formulation allows us to have **optimality guaranties** on the obtained solution
- BRANE cut and BRANE relax **improve** the gene regulation networks obtained by **previous methods** given their weights as input
- **Low computation time** (negligible in comparison to the necessary weights computation)

Inference improvement using clustering

- Edge selection step: binary edge labeling $\mathbf{x} \in \{0,1\}^n$

We want to

- favor strongly weighted edges

$$\underset{\mathbf{x}\in\{0,1\}^n}{\text{maximize}} \sum_{(i,j)\in\mathbb{V}^2} \omega_{i,j}\, x_{i,j} + \lambda(1 - x_{i,j}),$$

Inference improvement using clustering

- Edge selection step: binary edge labeling $\mathbf{x} \in \{0,1\}^n$
- Gene clustering step: node labeling $\mathbf{y} \in \mathbb{N}^G$

We want to

- favor strongly weighted edges

$$\underset{\mathbf{x}\in\{0,1\}^n}{\text{maximize}} \quad \sum_{(i,j)\in\mathbb{V}^2} \omega_{i,j}\, x_{i,j} + \lambda(1 - x_{i,j}),$$

Inference improvement using clustering

- Edge selection step: binary edge labeling $\mathbf{x} \in \{0,1\}^n$
- Gene clustering step: node labeling $\mathbf{y} \in \mathbb{N}^G$

We want to

- favor strongly weighted edges
- reduce weight $\omega_{i,j}$ if nodes v_i and v_j belong to distinct clusters
- cost function : $f(y_i, y_j) = \frac{\beta - \mathbb{1}(y_i \neq y_j)}{\beta}$, where $\beta > 1$ controls clustering

$$\underset{\substack{\mathbf{x}\in\{0,1\}^n \\ \mathbf{y}\in\mathbb{N}^G}}{\text{maximize}} \sum_{(i,j)\in\mathbb{V}^2} f(y_i, y_j)\, \omega_{i,j}\, x_{i,j} + \lambda(1 - x_{i,j}),$$

Inference improvement using clustering

- Edge selection step: binary edge labeling $\mathbf{x} \in \{0,1\}^n$
- Gene clustering step: node labeling $\mathbf{y} \in \mathbb{N}^G$

We want to

- favor strongly weighted edges
- reduce weight $\omega_{i,j}$ if nodes v_i and v_j belong to distinct clusters
- cost function : $f(y_i, y_j) = \frac{\beta - \mathbb{1}(y_i \neq y_j)}{\beta}$, where $\beta > 1$ controls clustering

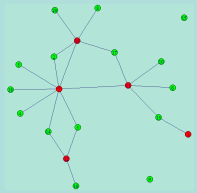
$$\underset{\substack{\mathbf{x}\in\{0,1\}^n \\ \mathbf{y}\in\mathbb{N}^G}}{\text{maximize}} \sum_{(i,j)\in\mathbb{V}^2} f(y_i, y_j)\, \omega_{i,j}\, x_{i,j} + \lambda(1 - x_{i,j}),$$

Can be improved by integrating biological and structural *a priori*

Enforcing modular structure

- Edge selection step: binary edge labeling $\mathbf{x} \in \{0,1\}^n$
- Gene clustering step: node labeling $\mathbf{y} \in \mathbb{N}^G$

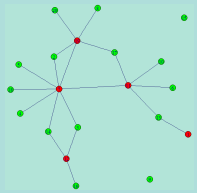
We want to promote a modular structure organized around central nodes (TFs)



Enforcing modular structure

- Edge selection step: binary edge labeling $\mathbf{x} \in \{0,1\}^n$
- Gene clustering step: node labeling $\mathbf{y} \in \mathbb{N}^G$

We want to promote a modular structure organized around central nodes (TFs)



- Let $\mathcal{T}$ be a set of central nodes (TFs) and $\mu_{i,j}$ a parameter controlling the modular structure

$$\underset{\substack{\mathbf{x}\in\{0,1\}^n \\ \mathbf{y}\in\mathbb{N}^G}}{\text{maximize}} \quad \sum_{(i,j)\in\mathbb{V}^2} f(y_i,y_j)\,\omega_{i,j}\,x_{i,j} + \lambda(1-x_{i,j}) + \sum_{\substack{i\in\mathbb{V} \\ j\in\mathbb{T}}} \mu_{i,j}\mathbb{1}(y_i=j).$$

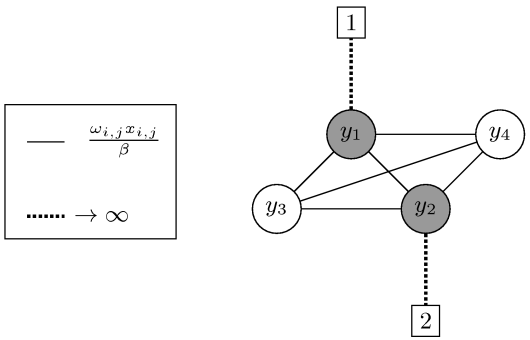
Optimization strategy

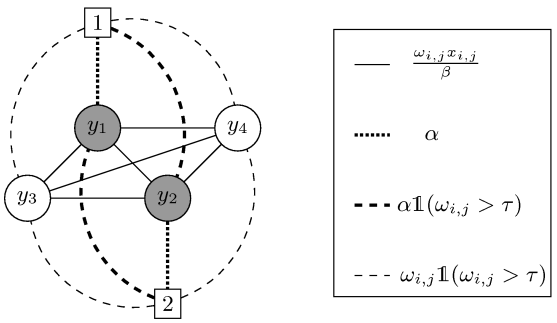
$$\underset{\substack{\mathbf{x}\in\{0,1\}^n \\ \mathbf{y}\in\mathbb{N}^G}}{\text{maximize}} \quad \sum_{(i,j)\in\mathbb{V}^2} f(y_i, y_j)\,\omega_{i,j}\,x_{i,j} + \lambda(1 - x_{i,j}) + \sum_{\substack{i\in\mathbb{V} \\ j\in\mathbb{T}}} \mu_{i,j}\mathbb{1}(y_i = j).$$

Optimization strategy

$$\underset{\substack{\mathbf{x}\in\{0,1\}^n \\ \mathbf{y}\in\mathbb{N}^G}}{\text{maximize}} \quad \sum_{(i,j)\in\mathbb{V}^2} f(y_i, y_j)\,\omega_{i,j}\,x_{i,j} + \lambda(1 - x_{i,j}) + \sum_{\substack{i\in\mathbb{V} \\ j\in\mathbb{T}}} \mu_{i,j}\mathbb{1}(y_i = j).$$

Hard-clustering *[Pirayre, 2015]*	*Soft*-clustering *[Pirayre, 2016]*
# clusters = # TF	# clusters < # TF
$\mu_{i,j} = \begin{cases} \to \infty & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$	$\mu_{i,j} = \begin{cases} \alpha & \text{if } i = j \\ \alpha\mathbb{1}(\omega_{i,j} > \tau) & \text{if } i \neq j \text{ and } i \in \mathbb{T} \\ \omega_{i,j}\mathbb{1}(\omega_{i,j} > \tau) & \text{if } i \neq j \text{ and } i \notin \mathbb{T} \end{cases}$





Optimization strategy

Alternating optimization

$$\underset{\substack{\mathbf{x}\in\{0,1\}^n \\ \mathbf{y}\in\mathbb{N}^G}}{\text{maximize}} \quad \sum_{(i,j)\in\mathbb{V}^2} f(y_i, y_j)\,\omega_{i,j}\,x_{i,j} + \lambda(1 - x_{i,j}) + \sum_{\substack{i\in\mathbb{V} \\ j\in\mathbb{T}}} \mu_{i,j}\mathbb{1}(y_i = j).$$

Optimization strategy

Alternating optimization

$$\underset{\substack{\mathbf{x}\in\{0,1\}^n \\ \mathbf{y}\in\mathbb{N}^G}}{\text{maximize}} \quad \sum_{(i,j)\in\mathbb{V}^2} f(y_i, y_j)\,\omega_{i,j}\,x_{i,j} + \lambda(1 - x_{i,j}) + \sum_{\substack{i\in\mathbb{V} \\ j\in\mathbb{T}}} \mu_{i,j}\mathbb{1}(y_i = j).$$

- At y fixed and x variable:

$$\text{minimize}\,\mathbf{x} \in \{0,1\}^n \sum_{(i,j)\in\mathbb{V}^2} f(y_i, y_j)\,\omega_{i,j}\,x_{i,j} + \lambda(1 - x_{i,j})$$

- Explicit form:

$$x^*_{i,j} = \begin{cases} 1 & \text{if } f(y_i, y_j)\,\omega_{i,j} > \lambda \\ 0 & \text{otherwise.} \end{cases}$$

Multi-labels and relaxation

- At x fixed and y variable:

$$\text{minimize}\, \mathbf{y} \in \mathbb{N}^G \sum_{(i,j)\in\mathbb{V}^2} \frac{\omega_{i,j}\, x_{i,j}}{\beta} \mathbb{1}(y_i \neq y_j) + \sum_{i\in\mathbb{V},\, j\in\mathbb{T}} \mu_{i,j} \mathbb{1}(y_i \neq j)$$

Multi-labels and relaxation

- At x fixed and y variable:

$$\text{minimize}\, \mathbf{y} \in \mathbb{N}^G \sum_{(i,j)\in\mathbb{V}^2} \frac{\omega_{i,j}\, x_{i,j}}{\beta} \mathbb{1}(y_i \neq y_j) + \sum_{i\in\mathbb{V},\, j\in\mathbb{T}} \mu_{i,j} \mathbb{1}(y_i \neq j) \Rightarrow \text{NP-ha}$$

Multi-labels and relaxation

- At x fixed and y variable:

$$\text{minimize}\, \mathbf{y} \in \mathbb{N}^G \sum_{(i,j)\in\mathbb{V}^2} \frac{\omega_{i,j}\, x_{i,j}}{\beta} \mathbb{1}(y_i \neq y_j) + \sum_{i\in\mathbb{V},\, j\in\mathbb{T}} \mu_{i,j} \mathbb{1}(y_i \neq j) \Rightarrow \text{NP-ha}$$

- discrete problem $\Rightarrow$ quadratic relaxation
- T-classes problem $\Rightarrow$ T binary sub-problems
  - label restriction to $\mathbb{T}$: $\{s^{(1)}, \ldots, s^{(T)}\}$ such that $s_j^{(t)} = 1$ if $j = t$ and 0 otherwise.
  - $\mathcal{Y} = \{y^{(1)}, \ldots, y^{(T)}\}$ such that $y^{(t)} \in [0,1]^G$

Multi-labels and relaxation

- At x fixed and y variable:

$$\text{minimize}\, \mathbf{y} \in \mathbb{N}^G \sum_{(i,j)\in\mathbb{V}^2} \frac{\omega_{i,j}\, x_{i,j}}{\beta} \mathbb{1}(y_i \neq y_j) + \sum_{i\in\mathbb{V},\, j\in\mathbb{T}} \mu_{i,j} \mathbb{1}(y_i \neq j) \Rightarrow \text{NP-ha}$$

- discrete problem $\Rightarrow$ quadratic relaxation
- T-classes problem $\Rightarrow$ T binary sub-problems
  - label restriction to $\mathbb{T}$: $\{s^{(1)}, \ldots, s^{(T)}\}$ such that $s_j^{(t)} = 1$ if $j = t$ and 0 otherwise.
  - $\mathcal{Y} = \{y^{(1)}, \ldots, y^{(T)}\}$ such that $y^{(t)} \in [0,1]^G$

Problem re-expressed as:

$$\text{minimize}\, \mathcal{Y} \sum_{t=1}^{T} \left(\sum_{(i,j)\in\mathbb{V}^2} \frac{\omega_{i,j}\, x_{i,j}}{\beta} \left(y_i^{(t)} - y_j^{(t)} \right)^2 + \sum_{i\in\mathbb{V},\, j\in\mathbb{T}} \mu_{i,j} \left(y_i^{(t)} - s_j^{(t)} \right)^2 \right).$$

Optimization strategy

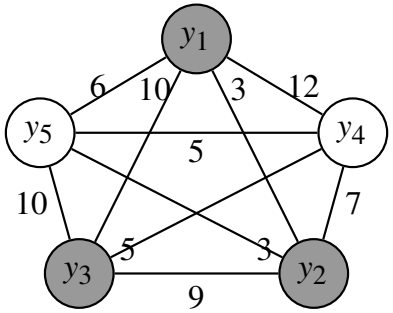
$$\text{minimize}\, \mathcal{Y} \sum_{t=1}^{T} \left(\sum_{(i,j)\in\mathbb{V}^2} \frac{\omega_{i,j}\, x_{i,j}}{\beta} \left(y_i^{(t)} - y_j^{(t)} \right)^2 + \sum_{i\in\mathbb{V},\, j\in\mathbb{T}} \mu_{i,j} \left(y_i^{(t)} - s_j^{(t)} \right)^2 \right).$$

- This problem is called the Combinatorial Dirichlet problem
- Random Walker algorithm
- Minimization *via* solving a linear system of equation *[Grady, 2006]*
- Final labeling: node i affected to the label t for which $y_i^{(t)}$ is maximal

$$y_i^* = \arg\max t \in \mathbb{T}\; y_i^{(t)}$$

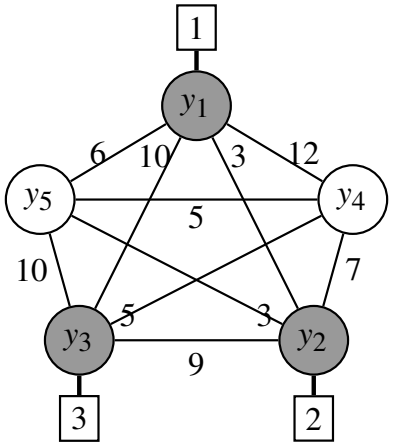
Optimization strategy

We want to obtain the optimal labeling $\mathbf{y}^*$ based on an weighted graph $\Rightarrow$ Random Walker algorithm



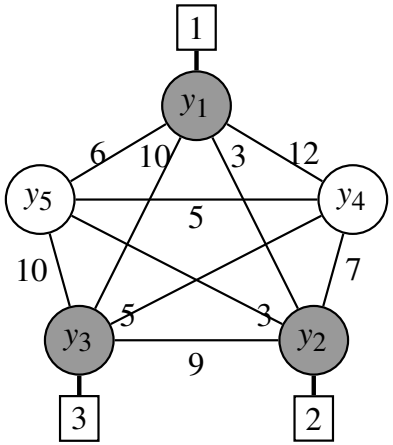
Optimization strategy

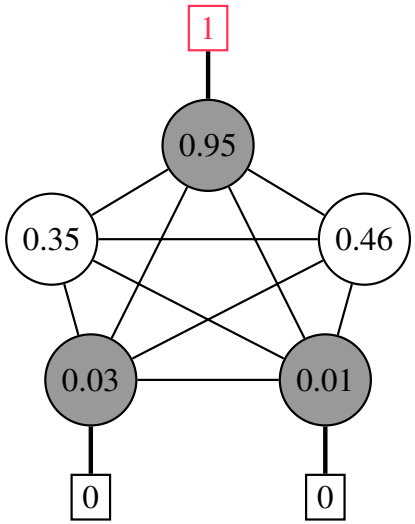
We want to obtain the optimal labeling $\mathbf{y}^*$ based on an weighted graph $\Rightarrow$ Random Walker algorithm



Optimization strategy

We want to obtain the optimal labeling $\mathbf{y}^*$ based on an weighted graph $\Rightarrow$ Random Walker algorithm

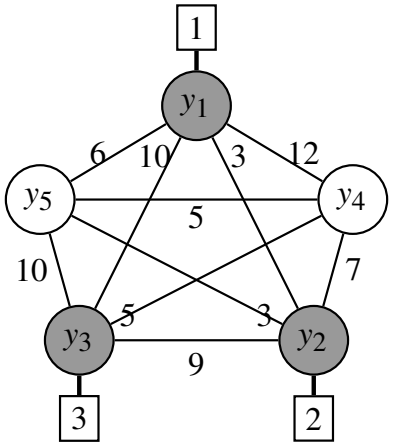


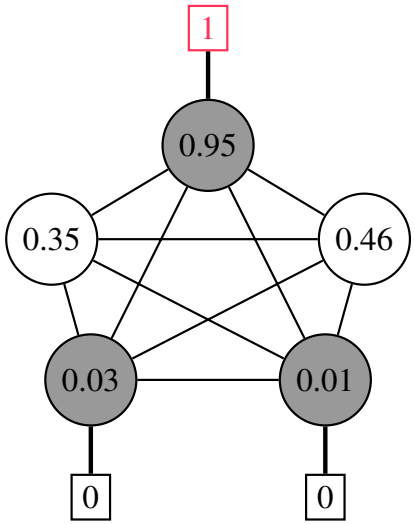


$y^{(1)}$

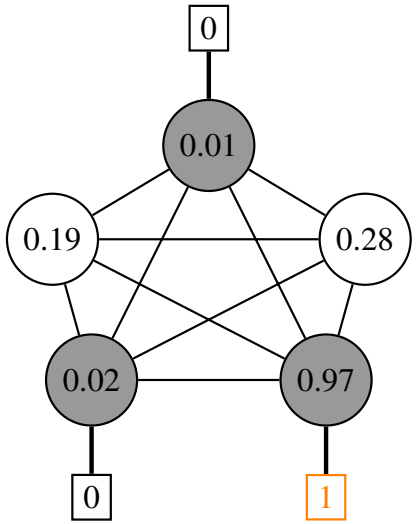
Optimization strategy

We want to obtain the optimal labeling $\mathbf{y}^*$ based on an weighted graph $\Rightarrow$ Random Walker algorithm





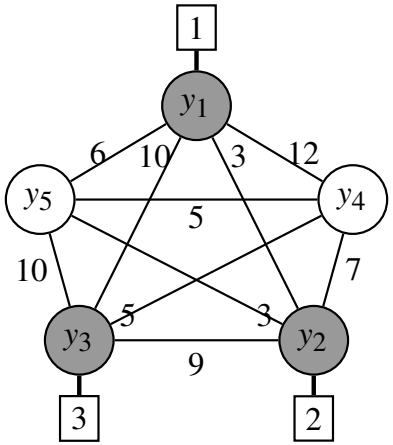
$y^{(1)}$

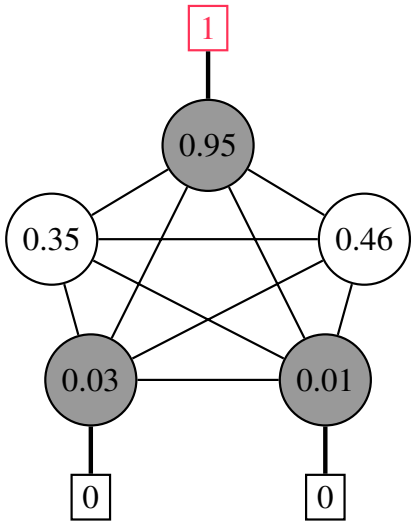


$y^{(2)}$

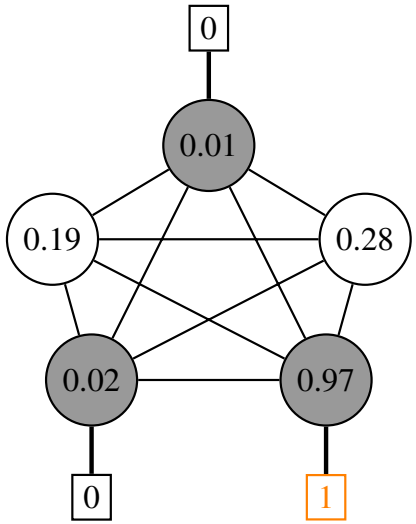
Optimization strategy

We want to obtain the optimal labeling $\mathbf{y}^*$ based on an weighted graph $\Rightarrow$ Random Walker algorithm

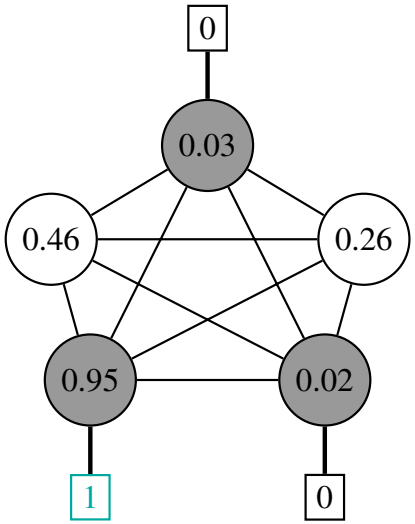




$y^{(1)}$



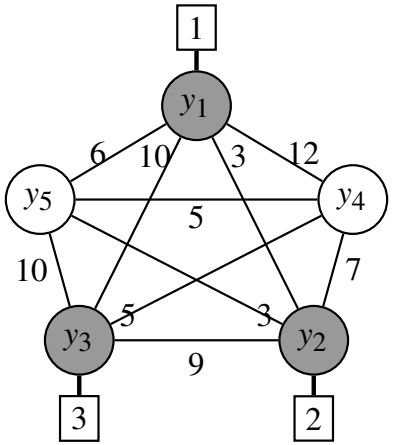
$y^{(2)}$

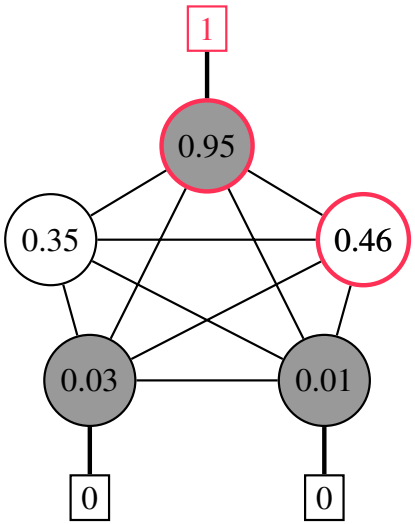


$y^{(3)}$

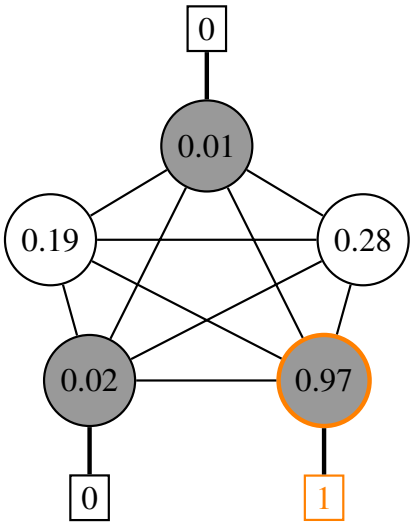
Optimization strategy

We want to obtain the optimal labeling $\mathbf{y}^*$ based on an weighted graph $\Rightarrow$ Random Walker algorithm

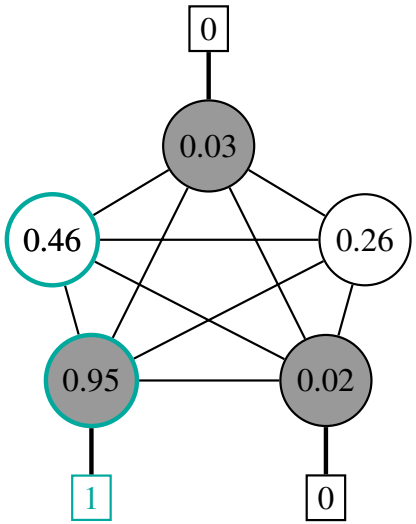




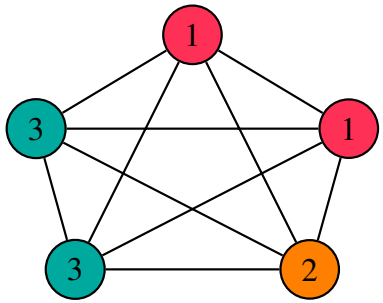
$y^{(1)}$



$y^{(2)}$



$y^{(3)}$



$y^* = \{1, 1, 2, 3, 3\}$

Benchmark data: DREAM4 and DREAM5

- DREAM4

Network	1	2	3	4	5	Average
CLR	0.256	0.275	0.314	0.313	0.313	0.294
BRANE Clust	0.275	0.337	0.360	0.335	0.342	0.330
Gain	7.3 %	22.6 %	14.5 %	7.0 %	9.1 %	12.1 %
GENIE3	0.269	0.288	0.331	0.323	0.329	0.308
BRANE Clust	**0.287**	**0.348**	**0.364**	**0.371**	**0.367**	**0.347**
Gain	6.5 %	20.9 %	10.0 %	15.0 %	11.6 %	12.8 %

- DREAM5

Network	1	3	4
CLR	0.252	0.0378	0.0080
BRANE Clust	0.253	0.0399	0.0073
GENIE3	0.283	0.0488	0.0081
BRANE Clust	**0.327**	**0.0536**	**0.0083**

Escherichia coli network

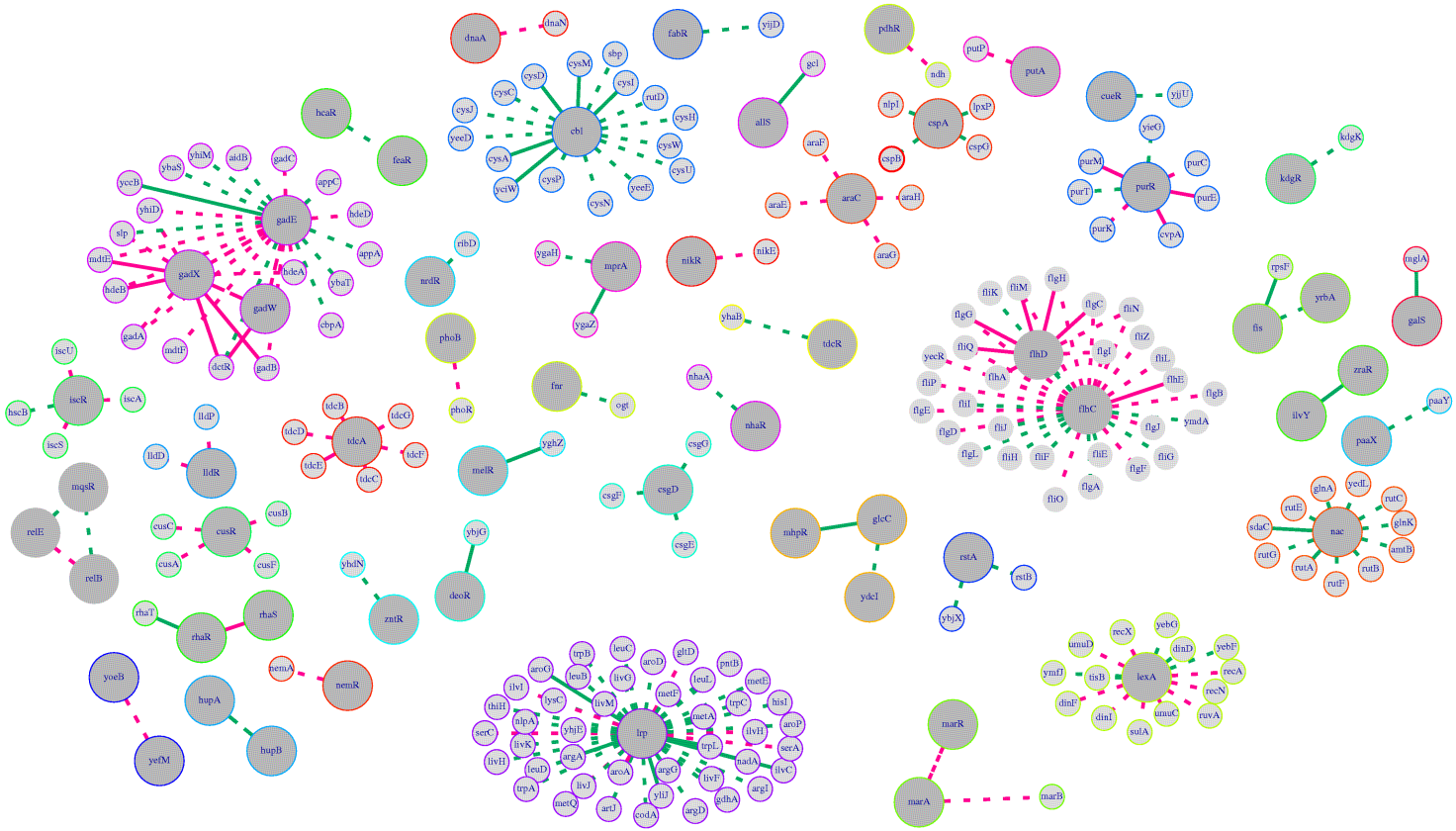


Figure: Network built using BRANE Clust on GENIE3 weights and containing 236 edges. Large dark gray nodes refer to transcription factors (TFs). Inferred edges also reported in the ground truth are colored in pink while predictive edges are green. Dashed edges correspond to links inferred by both BRANE Clust and GENIE3 while solid links refer to edges specifically inferred by BRANE Clust. The node contours are colored according to the clusters to which they belong to.

Conclusions and perspectives

BRANE Clust: Biologically Related A priori Network Enhancement using Clustering

Conclusions

- Inference and clustering alternate optimization: convergence guarantee
- Incorporating clustering steps gives promising result
- Enforcing a modular structure around central nodes improves results

Perspectives

- Clustering fusion improvement
- Joint clustering and inference (instead of cluster-assisted inference)