185.A83 Machine Learning for Health Informatics

2016S, VU, 2.0 h, 3.0 ECTS

Week 19 11.05.2016 17:00-20:00

Tutorial / Demo Talk
Graph extraction from images

b.malle@hci-kdd.org

http://hci-kdd.org/machine-learning-for-health-informatics-course

Image segmentation by Region Merging

- Paper by Felzenszwalb & Huttenlocher, 2004
- Based on Kruskals MST algorithm
- Takes input image as natural graph with vertices := pixels and edges := pixel neighborhoods
- Visits edges in ascending order of weight and merges regions if they satisfy a certain criterion
- Flexible as merging criterion can be adapted as desired (for amount, size, or shape of regions)

Felzenszwalb, P. F. & Huttenlocher, D. P. 2004. Efficient graph-based image segmentation. International Journal of Computer Vision, 59, (2), 167-181, doi:10.1023/B:VISI.0000022288.19776.77.

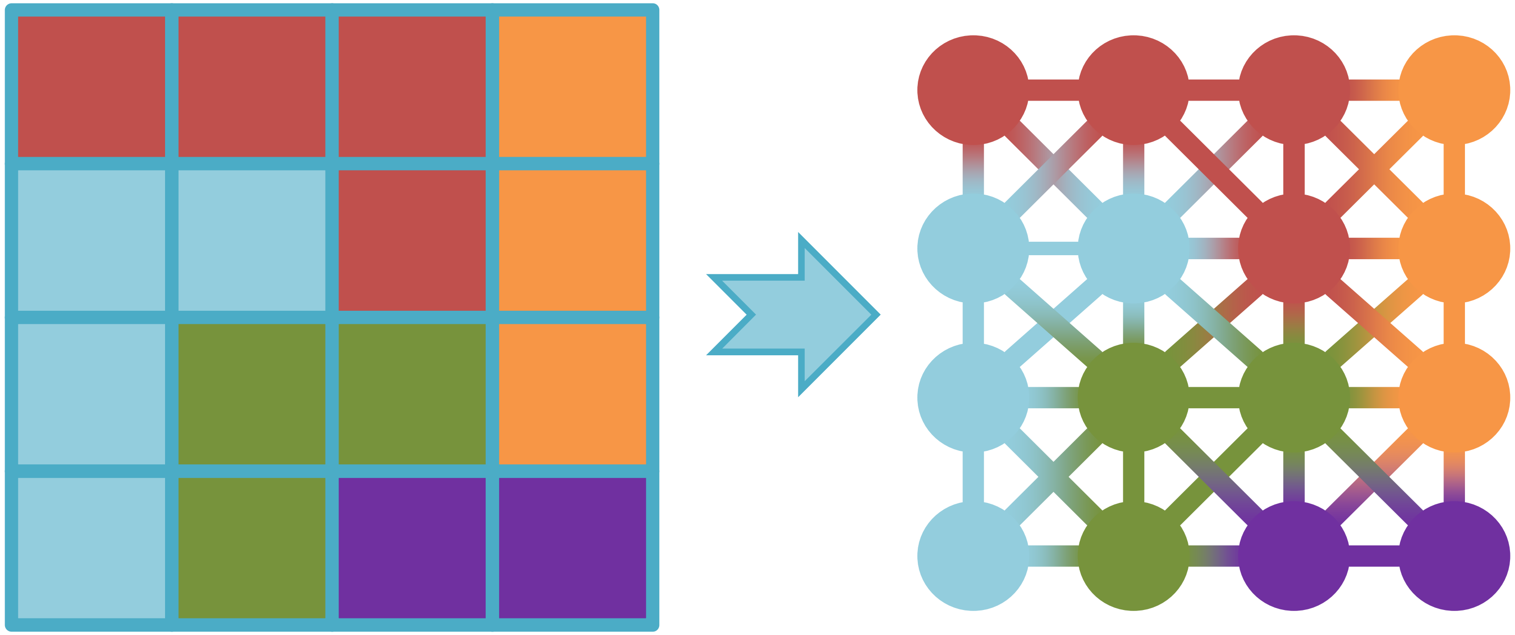
The processing pipeline…

... Consists of 4 stages:

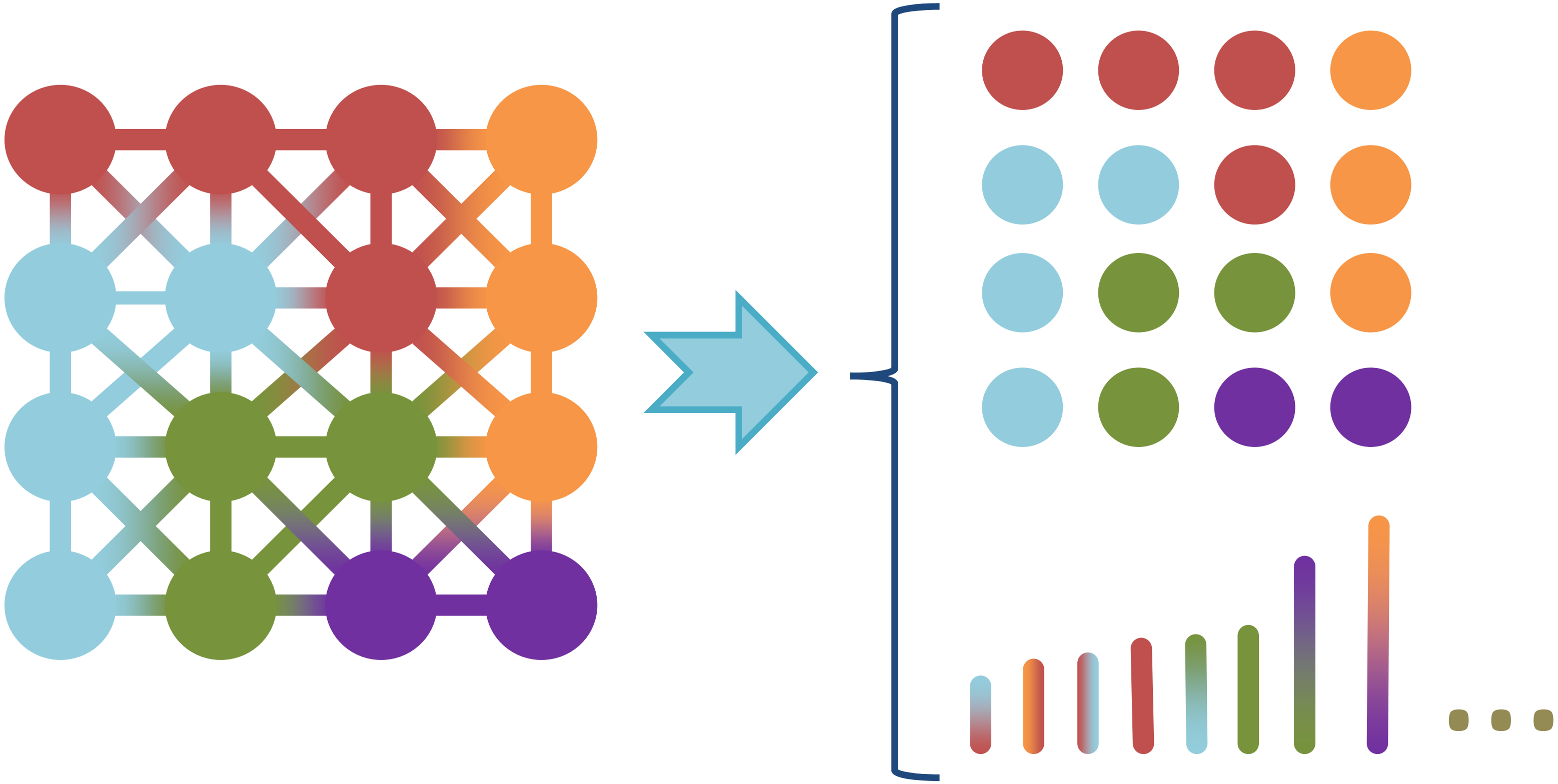
- Image preprocessing
  - Intensity image; background separation
- Algorithmic Preprocessing
  - Provides the needed datastructures for segmentation algorithms
  - Graph structure, adjacency list, edge list etc.
- Image Segmentation
  - Transforms image into a label map
- Graph extraction
  - Computing region centroids => Delaunay triangulation => feature vector extraction => graph output

Pixel based image as graph

(Edge weights correspond to color difference)



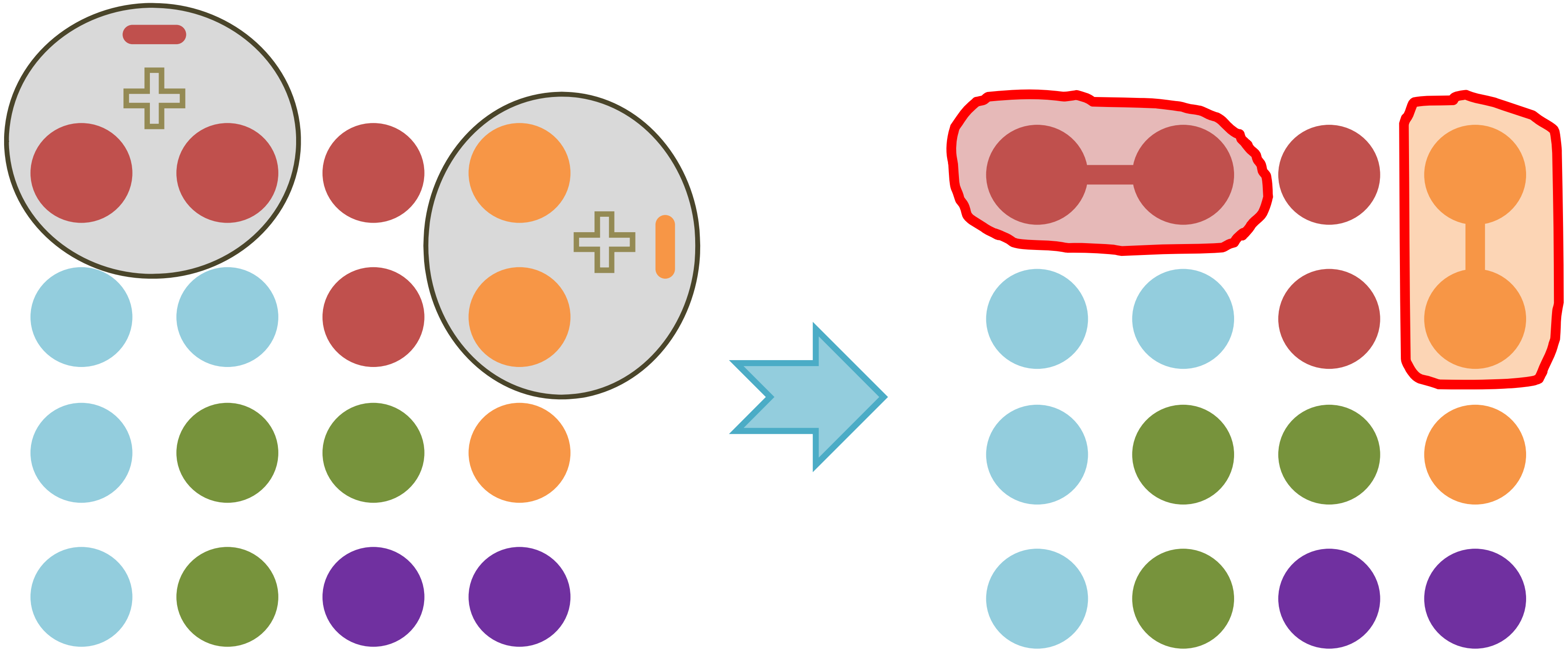
Graph => Node List + Sorting



Basic algorithm pseudo code

```
For all edges from w_min to w_max
    if ( edge is internal to region )
        continue
    else
        compute Int(C1), Int(C2), D(C1, C2)
        if ( boundary exists between C1, C2 )
            continue
        else
            merge C1, C2 into new region
```

Building Regions from pixels



Merging regions (individual case)

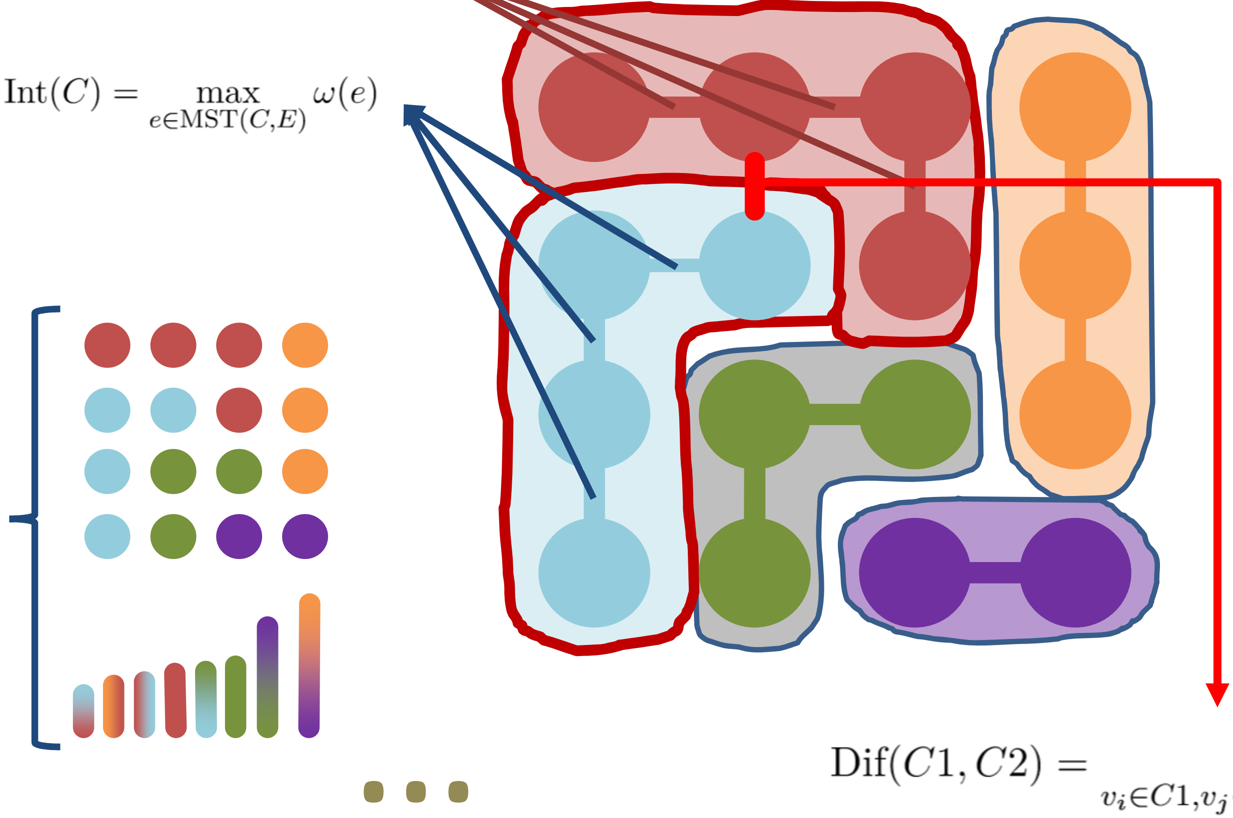
Kruskal based Region Merging

$$\mathrm{D}(C1, C2) = \begin{cases} \mathit{true} & \text{if} \quad \mathrm{Dif}(C1, C2) > \mathrm{MInt}(C1, C2) \\ \mathit{false} & \text{otherwise} \end{cases}$$

Merge!!!

$$\mathrm{Int}(C) = \max_{e \in \mathrm{MST}(C,E)} \omega(e)$$

$$\mathrm{Int}(C) = \max_{e \in \mathrm{MST}(C,E)} \omega(e)$$



In this case:

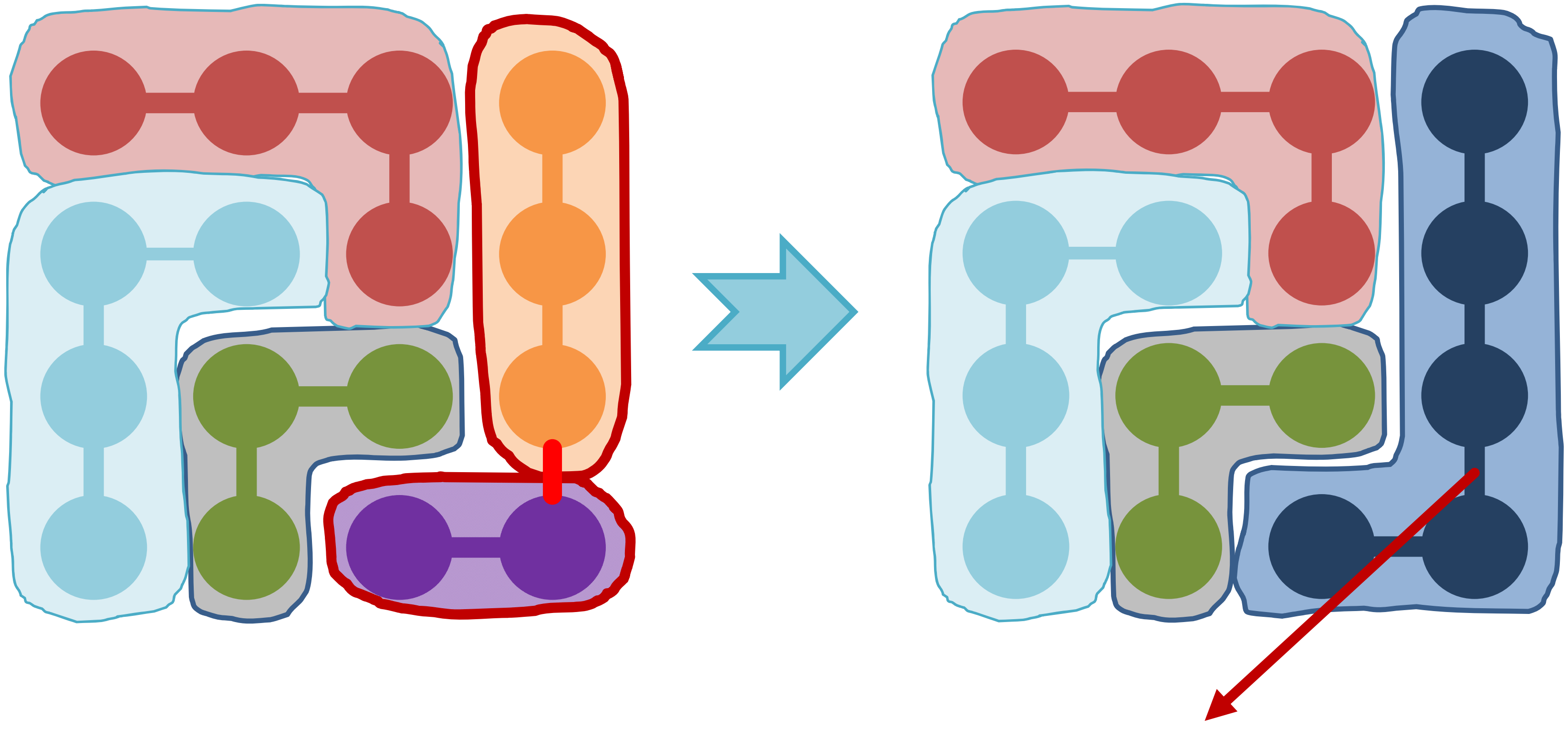
- D(C1, C2) = true
- No merge !!!

$$\mathrm{Dif}(C1, C2) = \min_{v_i \in C1, v_j \in C2, (v_i, v_j) \in E} \omega(v_i, v_j)$$

Kruskal based Region Merging

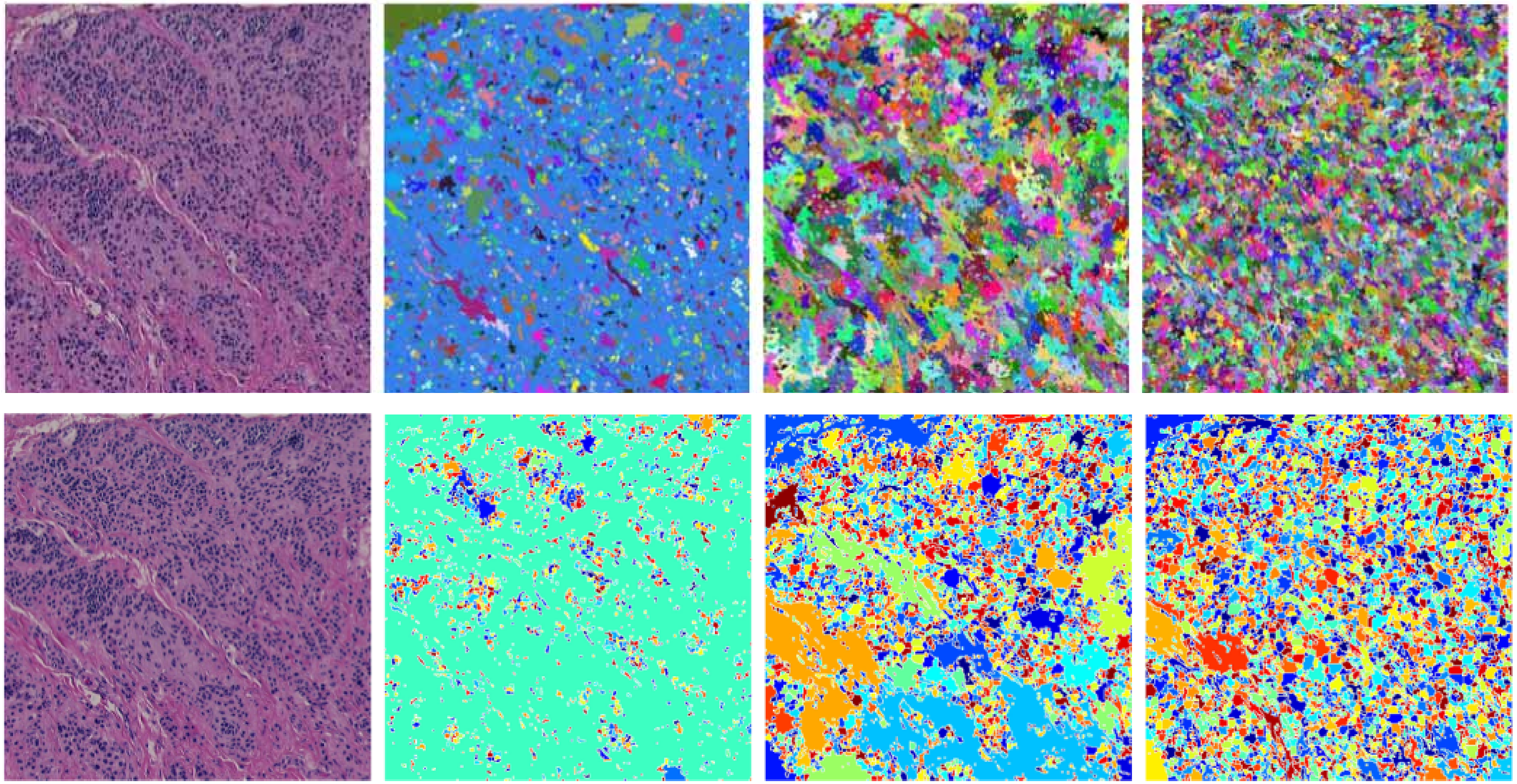
In this case:

- D(C1, C2) = false
- MERGE !!!

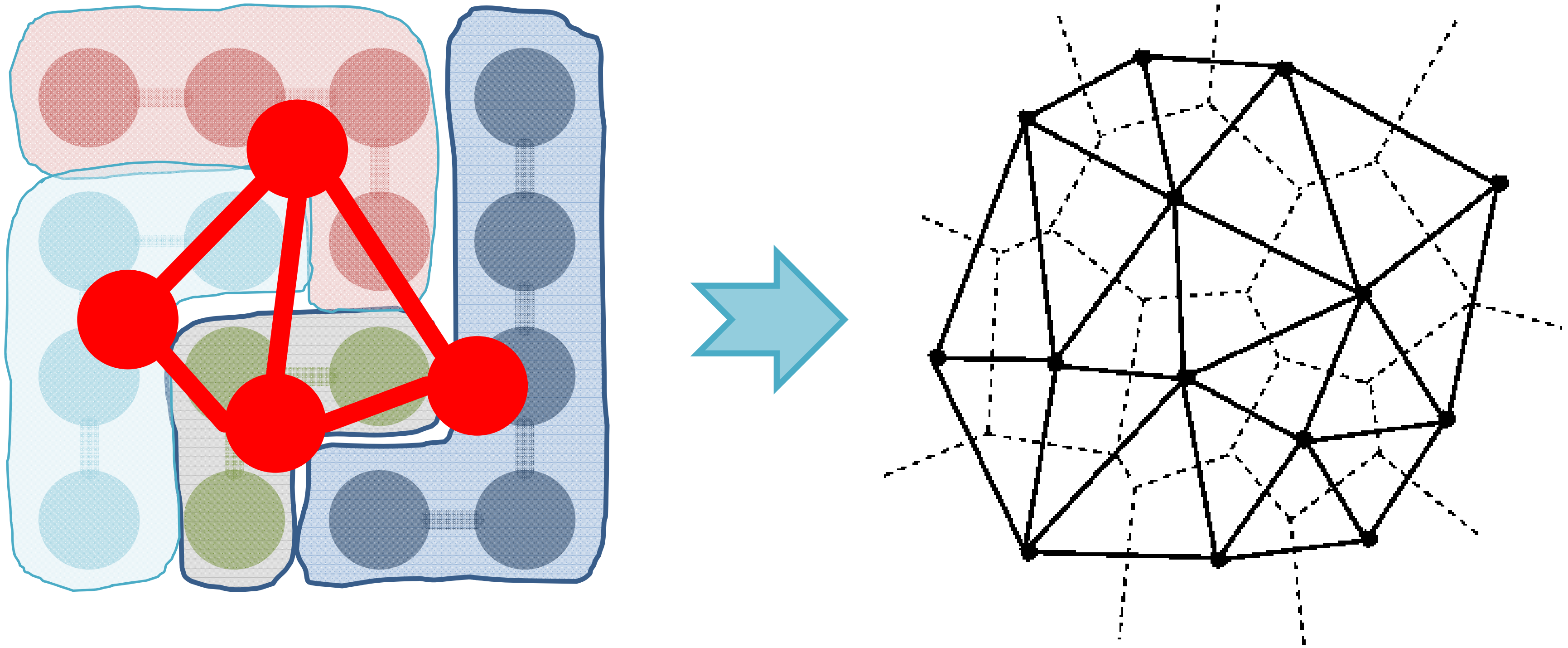


new max edge (weight)

Kruskal-based (JS) vs Watershed (Matlab)



Creating the graph: Label Map -> Delaunay Triangulation



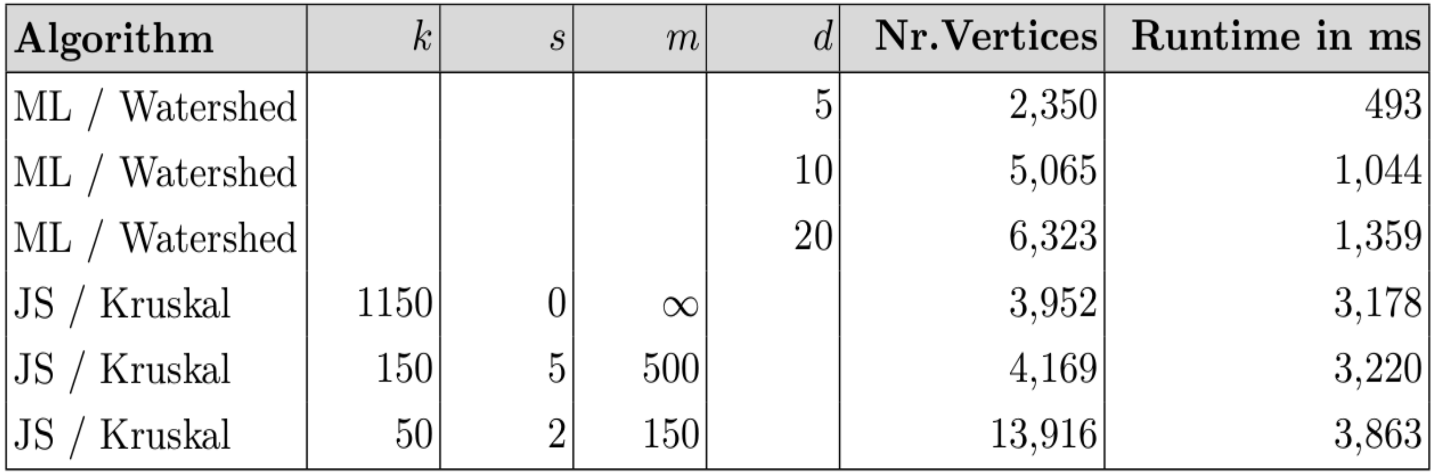
Voronoi [1] /
Delaunay triangulation [2] (tessellation)

[1] Aurenhammer, F. 1991. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23, (3), 345-405.

[2] Lee, D.-T. & Schachter, B. J. 1980. Two algorithms for constructing a Delaunay triangulation. International Journal of Computer & Information Sciences, 9, (3), 219-242.

Algorithmic performance

Kruskal-based (JS) vs Watershed (MLB)

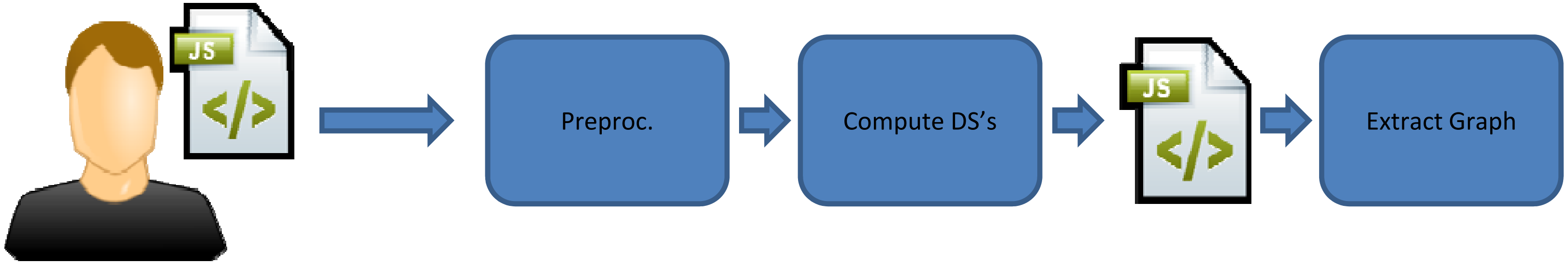


Algorithm	k	s	m	d	Nr.Vertices	Runtime in ms
ML / Watershed				5	2,350	493
ML / Watershed				10	5,065	1,044
ML / Watershed				20	6,323	1,359
JS / Kruskal	1150	0	∞		3,952	3,178
JS / Kruskal	150	5	500		4,169	3,220
JS / Kruskal	50	2	150		13,916	3,863

Holzinger, A., Malle, B. & Giuliani, N. 2014. On Graph Extraction from Image Data. In: Slezak, D., Peters, J. F., Tan, A.-H. & Schwabe, L. (eds.) Brain Informatics and Health, BIH 2014, Lecture Notes in Artificial Intelligence, LNAI 8609. Heidelberg, Berlin: Springer, pp. 552-563, doi:10.1007/978-3-319-09891-3_50.

The future: an extensible Web Research Platform

- Computational pipeline
- Include WebWorker (background Threads) and possibility to upload self-authored algorithms
  - Pipeline knows how to configure itself
  - Prepares necessary data structures
  - Executes user algorithm (which returns a region Map)
  - Pipeline finishes graph extraction out of region data



- User profiles, Group profiles
- Easy way to bookmark results, publish & reproduce

Web Resources:

- Graph extraction algorithm (as of an early Alpha of iKNOdis.js from 2014)
  https://github.com/cassinius/iKnodisJS

- Graphinius VIS module (up-to-date as of April 2016)
  https://github.com/cassinius/GraphiniusVis
  https://github.com/Niciii/GraphiniusVis

- The underlying Graphinius JS library (up-to-date as of April 2016)
  https://github.com/cassinius/GraphiniusJS

- Online graph extractor
  http://berndmalle.com/graphext/

- Online graph visualisation:
  http://berndmalle.com/GraphiniusVis/

Thank you!