Computer Vision I

Jannik Presberger, David Stein, Bjoern Andres

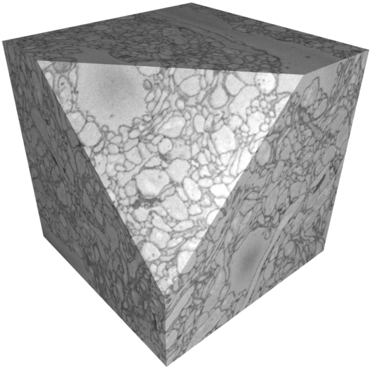
Machine Learning for Computer Vision
TU Dresden

https://mlcv.cs.tu-dresden.de/courses/25-winter/cv1/

Winter Term 2025/2026

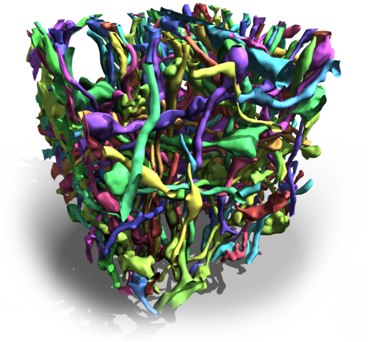
- So far, we have studied **pixel classification**, a problem whose feasible solutions define decisions at the pixels of an image
- Next, we will study **image segmentation**, a problem whose feasible solutions decide whether pairs of pixels are assigned to the same or distinct components of the image
- Image segmentation has applications where components of the image are indistinguishable by appearance (see next slide)

Image segmentation



Volume Image ($\sim$32 nm/voxel)[1]

$\mapsto$



Decomposition[2]

[1] Denk and Horstmann 2004. 10.1371/journal.pbio.0020329

[2] A, Köthe, Kröger, Helmstaedter, Briggman, Denk and Hamprecht 2012. 10.1016/j.media.2011.11.004

Image segmentation

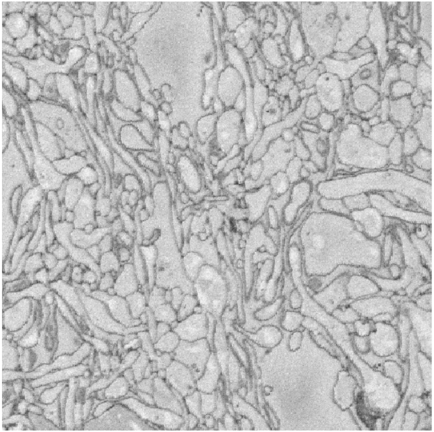


Image segmentation

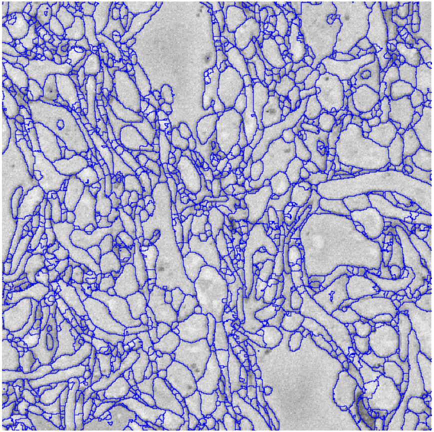


Image segmentation

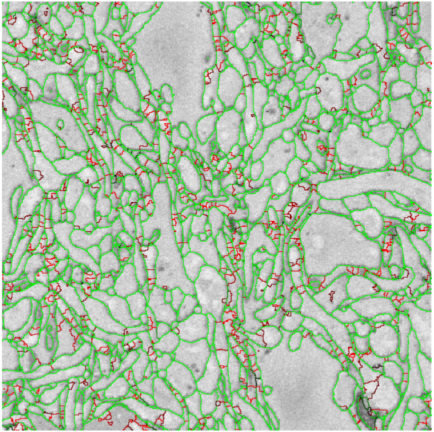
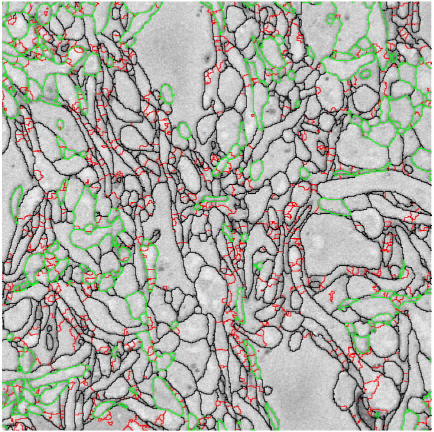
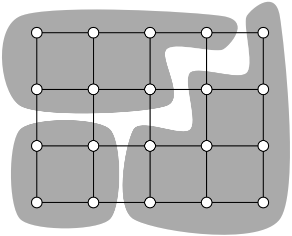


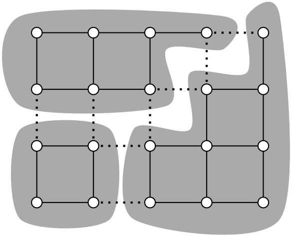
Image segmentation





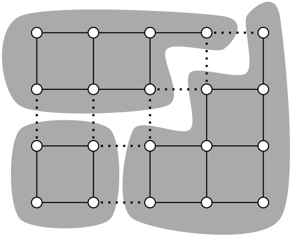
Decomposition of a graph $G = (V, E)$

- A mathematical abstraction of a segmentation of an image is a decomposition of the pixel grid graph.
- A decomposition of a graph is a partition of the node set into connected subsets (one example is depicted above in gray).



Decomposition of a graph $G = (V, E)$

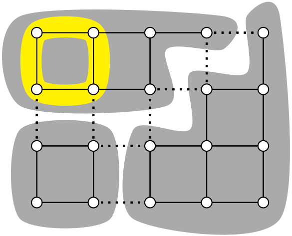
- A decomposition of a graph is characterized by the set of edges that straddle distinct components (depicted above as dotted lines)
- Those subsets of edges are called **multicuts** of the graph



Multicut of a graph $G = (V, E)$

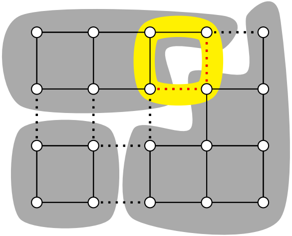
- A decomposition of a graph is characterized by the set of edges that straddle distinct components (depicted above as dotted lines)
- Those subsets of edges are called **multicuts** of the graph

Image segmentation



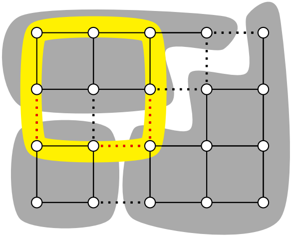
Multicut of a graph $G = (V, E)$

- A subset of edges is a multicuts iff no cycle in the graph intersects with the subset in precisely one edge



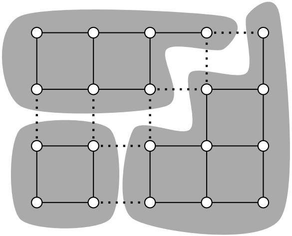
Multicut of a graph $G = (V, E)$

- A subset of edges is a multicuts iff no cycle in the graph intersects with the subset in precisely one edge



Multicut of a graph $G = (V, E)$

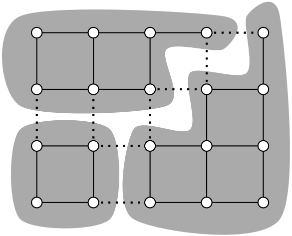
- A subset of edges is a multicuts iff no cycle in the graph intersects with the subset in precisely one edge



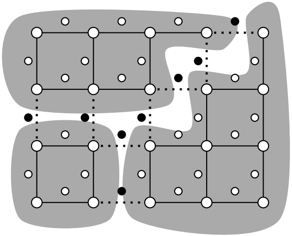
Multicut of a graph $G = (V, E)$

$$\text{multicuts}(G) := \{M \subseteq E \,|\, \forall C \in \text{cycles}(G) :\ |M \cap C| \neq 1\}$$

Image segmentation

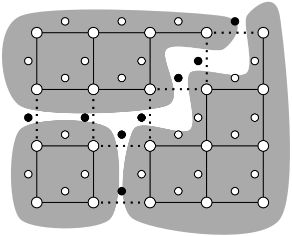


Multicut of a graph $G = (V, E)$



Multicut of a graph $G = (V, E)$

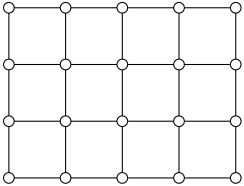
- The characteristic function $y\colon E \to \{0,1\}$ of a multicut $y^{-1}(1)$ can be used to encode the decomposition induced by the multicut in an $|E|$-dimensional 01-vector
- For any $e \in E$, $y_e = 1$ indicates that an edge is cut, straddling distinct components



Multicut of a graph $G = (V, E)$

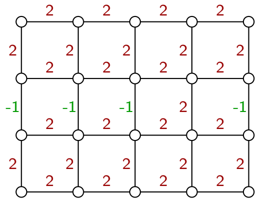
- The set of the characteristic functions of all multicuts of G:

$$Y_G := \left\{ y \in \{0,1\}^E \;\middle|\; \forall (V_C, E_C) \in \mathsf{cycles}(G)\, \forall e \in E_C : \, y_e \leq \sum_{f \in E_C \setminus \{e\}} y_f \right\}$$



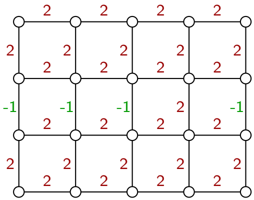
Graph $G = (V, E)$

- An instance of the image segmentation problem is given by a graph $G = (V, E)$ and, for every edge $e = \{v, w\} \in E$, a (positive or negative) cost $c_e \in \mathbb{R}$ that is payed iff the incident pixels v and w are put in distinct components
- Such costs can be learned (as described earlier in the course), e.g., $c_e = -f_\theta(x_e)$, or more specifically, $c_e = -\langle \theta, x_e \rangle$.



Graph $G = (V, E)$. Edge costs $c : E \to \mathbb{R}$

- An instance of the image segmentation problem is given by a graph $G = (V, E)$ and, for every edge $e = \{v, w\} \in E$, a (positive or negative) cost $c_e \in \mathbb{R}$ that is payed iff the incident pixels v and w are put in distinct components
- Such costs can be learned (as described earlier in the course), e.g., $c_e = -f_\theta(x_e)$, or more specifically, $c_e = -\langle \theta, x_e \rangle$.



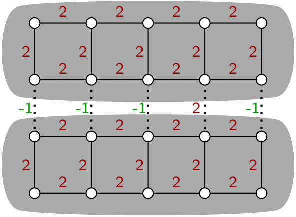
Graph $G = (V, E)$. Edge costs $c : E \to \mathbb{R}$

- Image segmentation problem:

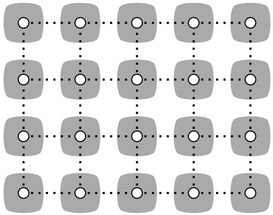
$$\min_{y \in Y_G} \sum_{e \in E} c_e \, y_e$$

- The optimal solution is shown on the next slide

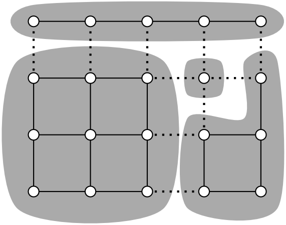
Image segmentation



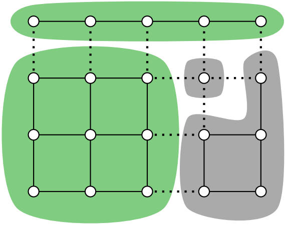
Graph $G = (V, E)$. Edge costs $c : E \to \mathbb{R}$



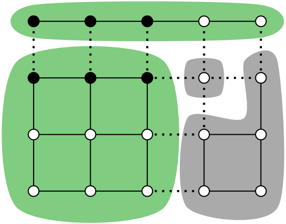
- One technique for finding feasible solutions to an image segmentation problem is **local search**.
- Starting from the finest decomposition into singleton components (depicted above), we greedily join neighboring components as long as this improves the cost (see next slide).



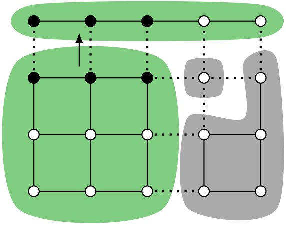
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components



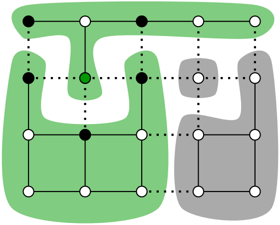
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green)



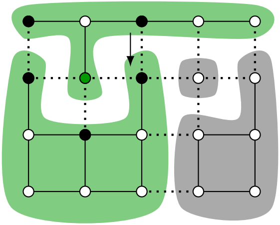
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black)



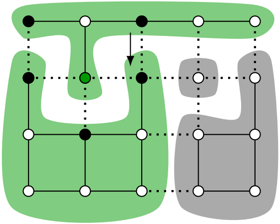
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.



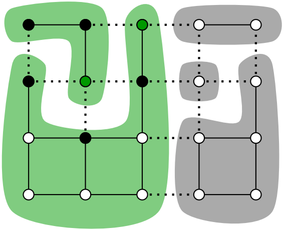
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost



- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost



- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost



- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost

Definition. Let $G = (V, E)$ be any graph.

- A subgraph $G' = (V', E')$ of G is called a **component (cluster)** of G iff G' is non-empty, node-induced (i.e. $E' = E \cap \binom{V'}{2}$) and connected.
- A partition Π of the node set V is called a **decomposition (clustering)** of G iff, for every $U \in \Pi$, the subgraph $(U, E \cap \binom{U}{2})$ of G induced by U is connected (and thus a component of G).
- Let D_G denote the set of all decompositions of G.

Definition. Let $G = (V, E)$ be any graph. A subset $M \subseteq E$ of edges is called a **multicut** of G iff, for every cycle $C \subseteq E$ of G: $|C \cap M| \neq 1$. Let M_G denote the set of all multicuts of G.

Lemma. For any decomposition of a graph G, the set of those edges that straddle distinct components is a multicut of G. This multicut is said to be **induced** by the decomposition. The map from decompositions to induced multicuts is a **bijection** from D_G to M_G.

Lemma. For any graph $G = (V, E)$ and any $y \in \{0,1\}^E$, the set $y^{-1}(1)$ is a multicut of G iff the following inequalities are satisfied:

$$\forall (V_C, E_C) \in \mathsf{cycles}(G)\ \forall e \in E_C\colon \quad y_e \leq \sum_{f \in E_C \setminus \{e\}} y_f \tag{1}$$

Definition. For any graph $G = (V, E)$, any $c \in \mathbb{R}^E$ and the set Y_G of all $y \in \{0,1\}^E$ that satisfy (1), the **minimum cost multicut problem** has the form

$$\min_{y \in \{0,1\}^E} \quad \underbrace{\sum_{e \in E} c_e\, y_e}_{=:\varphi(y)}$$

$$\text{subject to} \quad \forall (V_C, E_C) \in \mathsf{cycles}(G)\ \forall e \in E_C\colon \quad y_e \leq \sum_{f \in E_C \setminus \{e\}} y_f$$

Definition. For any graph $G = (V, E)$ and any decomposition Π of G, let y^Π the characteristic function of the multicut induced by Π, i.e. $y^\Pi \in \{0, 1\}^E$ such that

$$\forall e \in E\colon \quad y_e^\Pi = 0 \;\Leftrightarrow\; \exists U \in \Pi\colon e \subseteq U \tag{2}$$

Remark: The characteristic function $y \in \{0, 1\}^E$ of a multicut $y^{-1}(1)$ makes explicit for every edge $\{u, v\} = e \in E$ whether the incident nodes u and v belong to the same component ($y_e = 0$) or distinct components ($y_e = 1$).

Greedy joining algorithm:

- The greedy joining algorithm is a local search algorithm that starts from any initial decomposition.
- It searches for decompositions with lower cost by joining pairs of **neighboring (!)** components recursively.
- As components can only grow and the number of components decreases by one in every step, one typically starts from the finest decomposition Π_0 into one-elementary components.

Definition. Let $G = (V, E)$ be any graph.

- For any disjoint sets $B, C \subseteq V$, the pair $\{B, C\}$ is called **neighboring** in G iff there exist nodes $b \in B$ and $c \in C$ such that $\{b, c\} \in E$.
- For any decomposition Π of G, we define

$$\mathcal{E}_\Pi = \left\{ \{B, C\} \in \tbinom{\Pi}{2} \;\middle|\; \exists b \in B \, \exists c \in C \colon \{b, c\} \in E \right\} \quad . \tag{3}$$

- For any decomposition Π of G and any $\{B, C\} \in \mathcal{E}_\Pi$, let $\mathsf{join}_{BC}[\Pi]$ be the decomposition of G obtained by joining the sets B and C in Π, i.e.

$$\mathsf{join}_{BC}[\Pi] = (\Pi \setminus \{B, C\}) \cup \{B \cup C\} \quad . \tag{4}$$

$\Pi' =$ greedy-joining(Π)

choose $\{B, C\} \in \underset{\{B',C'\} \in \mathcal{E}_\Pi}{\operatorname{argmin}} \ \varphi(y^{\mathsf{join}_{B'C'}[\Pi]}) - \varphi(y^\Pi)$
if $\varphi(y^{\mathsf{join}_{BC}[\Pi]}) - \varphi(y^\Pi) < 0$
    $\Pi' :=$ greedy-joining(join$_{BC}[\Pi]$)
else
    $\Pi' := \Pi$

Greedy moving algorithm:

- The greedy moving algorithm is a local search algorithm that starts from any initial decomposition, e.g., the fixed point of greedy joining.
- It searches for decompositions with lower cost by recursively moving individual nodes from one component to a **neighboring (!)** component, possibly a new one.
- When a **cut node** is moved out of a component or a node is moved to a new component, the number of components increases. When the last node is moved out of a component, the number of components decreases.

Definition. For any graph $G = (V, E)$ and any decomposition Π of G, the decomposition Π is called **coarsest** iff, for every $U \in \Pi$, the component $(U, E \cap \binom{U}{2})$ induced by U is maximal.

Lemma. For any graph G, the coarsest decomposition is unique. We denote it by Π^*_G.

Definition. For any graph $G = (V, E)$, any decomposition Π of G and any $a \in V$, let U_a the unique $U_a \in \Pi$ such that $a \in U_a$, and let

$$\mathcal{N}_a = \{\emptyset\} \cup \{W \in \Pi \,|\, a \notin W \,\wedge\, \exists w \in W \colon \{a, w\} \in E\} \tag{5}$$

$$G_a = \left(U_a \setminus \{a\},\, E \cap \tbinom{U_a \setminus \{a\}}{2}\right) \tag{6}$$

For any $U \in \mathcal{N}_a$, let $\mathsf{move}_{aU}[\Pi]$ the decomposition of G obtained by moving the node a to the set U, i.e.

$$\mathsf{move}_{aU}[\Pi] = \Pi \setminus \{U_a, U\} \cup \{U \cup \{a\}\} \cup \Pi^*_{G_a} \quad . \tag{7}$$

$\Pi' =$ greedy-moving(Π)

choose $(a, U) \in \underset{a' \in A,\ U' \in \mathcal{N}_{a'}}{\operatorname{argmin}} \varphi(y^{\mathsf{move}_{a'U'}[\Pi]}) - \varphi(y^{\Pi})$

if $\varphi(y^{\mathsf{move}_{aU}[\Pi]}) - \varphi(y^{\Pi}) < 0$

  $\Pi' :=$ greedy-moving$(\mathsf{move}_{aU}[\Pi])$

else

  $\Pi' := \Pi$

Greedy moving using the technique of Kernighan and Lin:

- Both algorithms discussed above terminate as soon as no transformation (join and move, resp.) leads to a decomposition with strictly lower cost.
- This can be sub-optimal in case transformations that increase the cost at one point in the recursion can lead to transformations that decrease the cost at later points in the recursion and the decrease overcompensates the increase.
- A generalization of local search introduced by Kernighan and Lin (1970) can escape such sub-optimal fixed points.
- Its application to greedy moving (next slide) builds a sequence of moves and then carries out the first t moves whose cumulative decrease in cost is optimal.

$\Pi' = \text{greedy-moving-kl}(\Pi)$

$\Pi_0 := \Pi$
$\delta_0 := 0$
$A_0 := A$
$t := 0$
repeat (build sequence of moves)
  choose $(a_t, U_t) \in \underset{a \in A_t, U \in \mathcal{N}_a}{\operatorname{argmin}} \varphi(y^{\mathsf{move}_{aU}[\Pi_t]}) - \varphi(y^{\Pi_t})$
  $\Pi_{t+1} := \mathsf{move}_{a_t U_t}[\Pi_t]$
  $\delta_{t+1} := \varphi(y^{\Pi_{t+1}}) - \varphi(y^{\Pi_t})$
  $A_{t+1} := A_t \setminus \{a_t\}$ (move a_t only once)
  $t := t + 1$
until $A_t = \emptyset$

$\hat{t} := \min \underset{t' \in \{0,\ldots,|A|\}}{\operatorname{argmin}} \sum_{\tau=0}^{t'} \delta_\tau$ (choose sub-sequence)

if $\sum_{\tau=0}^{\hat{t}} \delta_\tau < 0$
  $\Pi' := \text{greedy-moving-kl}(\Pi_{\hat{t}})$ (recurse)
else
  $\Pi' := \Pi$ (terminate)

So far, we have studied

- **pixel classification**, a problem whose feasible solutions define decisions at the pixels of an image
- **image decomposition**, a problem whose feasible solutions decide whether pairs of pixels are assigned to the same or distinct components of the image.

Applications exists for which both problems are too restrictive:

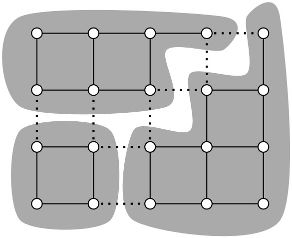
- In **pixel classification**, there is no way of assigning neighboring pixels of the same class to distinct components of the image.
- In **image decomposition**, there is no way of expressing that a unique decision shall be made for pixels that belong to the same component of the image.

Semantic image segmentation

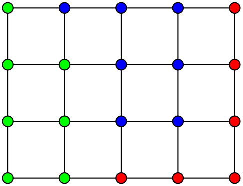
M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. CVPR 2016. See also: https://www.cityscapes-dataset.com/

- One application where a joint generalization of pixel classification and image decomposition is useful is called **semantic segmentation**.
- In the above image, thin boundaries are left between pixels of the same class (e.g. pedestrian) that belong to different instances of the class (e.g. distinct pedestrians).
- Next, we are going to introduce a strict generalization of both, pixel classification and image decomposition that does not require these boundaries.

Semantic image segmentation



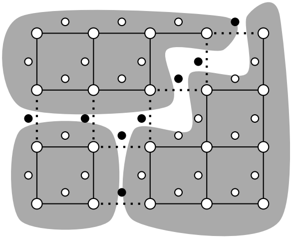
Graph Decomposition



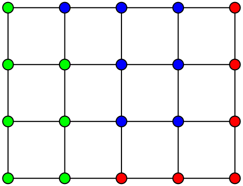
Node Labeling

We state an optimization problem whose feasible solutions define both, a **decomposition** of a graph $G = (V, E)$ and a **labeling** $l\colon V \to L$ of its nodes.

Semantic image segmentation



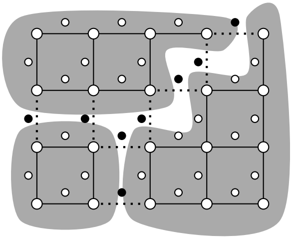
Graph Decomposition



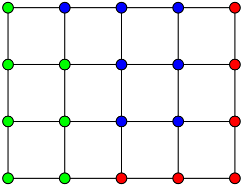
Node Labeling

We encode feasible decompositions by multicuts:

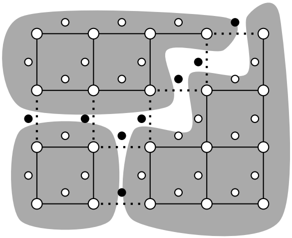
$$Y_G := \left\{ y : E \to \{0,1\} \;\middle|\; \forall C \in \mathsf{cycles}(G)\, \forall e \in C\colon\; y_e \le \sum_{f \in C \setminus \{e\}} y_f \right\}$$



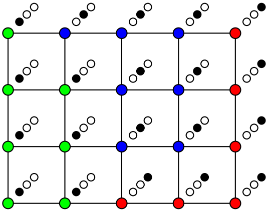
Graph Decomposition



Node Labeling



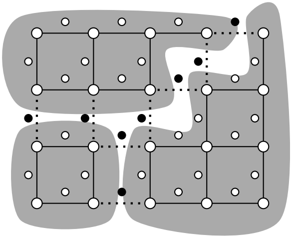
Graph Decomposition



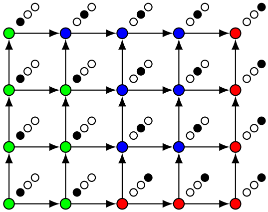
Node Labeling

We encode feasible node labelings by binary vectors:

$$Z_{VL} := \left\{ z : V \times L \to \{0,1\} \;\middle|\; \forall v \in V : \sum_{l \in L} z_{vl} = 1 \right\}$$

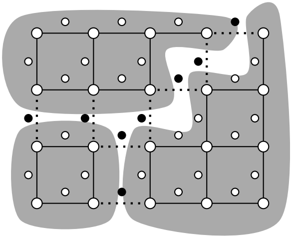


Graph Decomposition

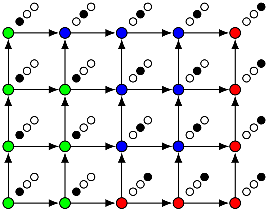


Node Labeling

We choose an arbitrary orientation (V, A) of the edges E, i.e., for each $v, w \in V$, we have $\{v, w\} \in E$ if and only if either $(v, w) \in A$ or $(w, v) \in A$.



Graph Decomposition



Node Labeling

W.r.t. the orientation (V, A) of the graph $G = (V, E)$, the set L of labels, any (costs) $c\colon V \times L \to \mathbb{R}$ and any (costs) $c', c''\colon A \times L^2 \to \mathbb{R}$, the instance of the **joint graph decomposition and node labeling problem** has the form

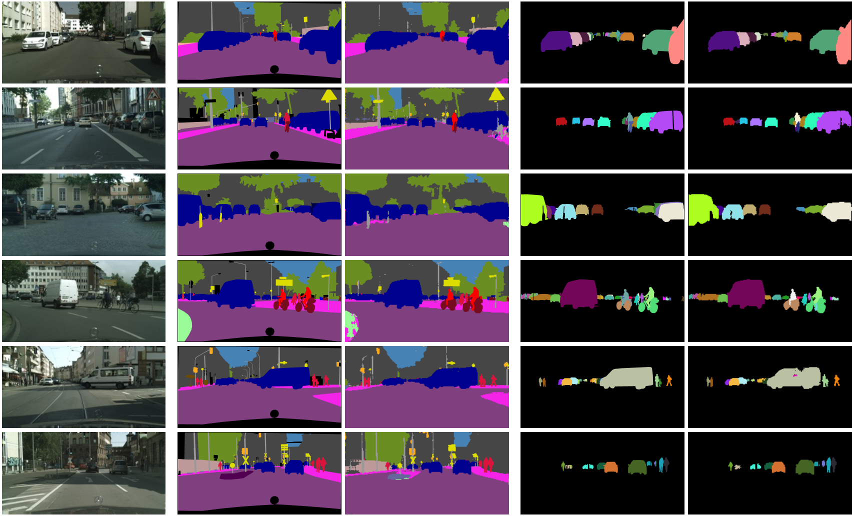
$$\min_{(y,z) \in Y_G \times Z_{VL}} \sum_{v \in V} \sum_{l \in L} c_{vl}\, z_{vl} + \sum_{(v,w) \in A} \sum_{(l,l') \in L^2} c'_{vwll'}\, z_{vl}\, z_{wl'}\, y_{\{v,w\}} + \sum_{(v,w) \in A} \sum_{(l,l') \in L^2} c''_{vwll'}\, z_{vl}\, z_{wl'}\, (1 - y_{\{v,w\}})$$

Adaptation of the local search algorithms greedy joining and greedy moving:

- Initialize the decomposition such that each node defines a singleton component
- Initialize the labels optimally wrt. c, i.e., irrespective of c' and c''.
- Apply **greedy joining** as follows, until termination: For every pair of neighboring components, consider the difference in cost that results from
  1. the joining of these components, followed by
  2. the re-labeling of the nodes of the joint component obtained by greedily changing one label at a time, until termination.
- Apply **greedy moving** as follows, until termination: For every node at a boundary of components, consider the different in cost that results from moving this node to the neighboring component, followed by a locally optimal change of its label.

Semantic image segmentation

Image | Labeling: Truth, Prediction | Decomposition: Truth, Prediction



Kirillov et al. 2017. doi:10.1109/CVPR.2017.774

Summary:

- The task of **image segmentation** assumes the form of the minimum cost multicut problem, typically with cost coefficients learned from data.
- Local search algorithms for this problem include greedy joining, greedy moving and greedy moving using the technique of Kernighan and Lin.
- The task of **semantic image segmentation** assumes the form of a joint graph decomposition and node labeling problem.
- The local search algorithms greedy joining and greedy moving can be adapted to this task.