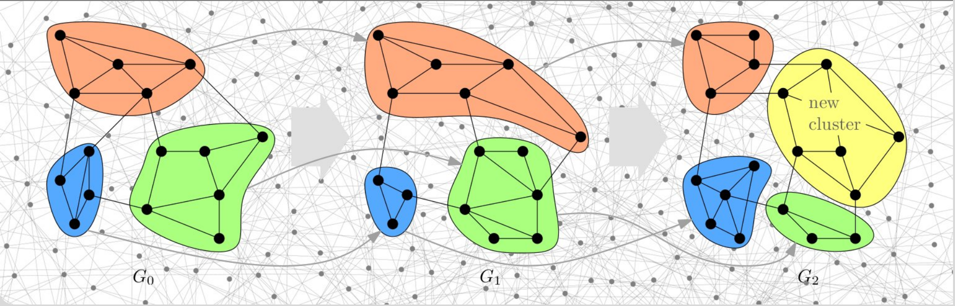
Algorithm Engineering for Graph Clustering

School on Graph Theory, Algorithms and Applications

Dorothea Wagner | Erice, Italy, 25. September – 3. October, 2011

KARLSRUHE INSTITUTE OF TECHNOLOGY – INSTITUTE OF THEORETICAL INFORMATICS

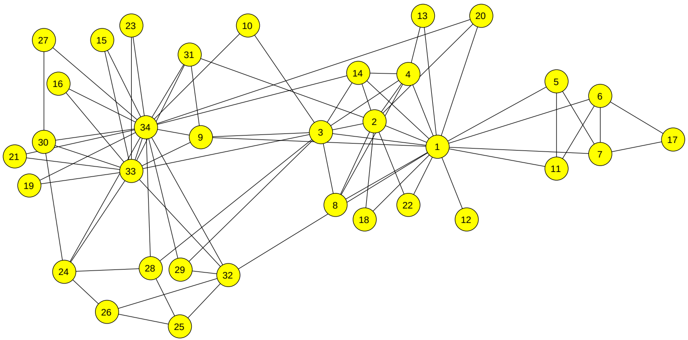


Structure

Structure

Scenario of Network Analysis

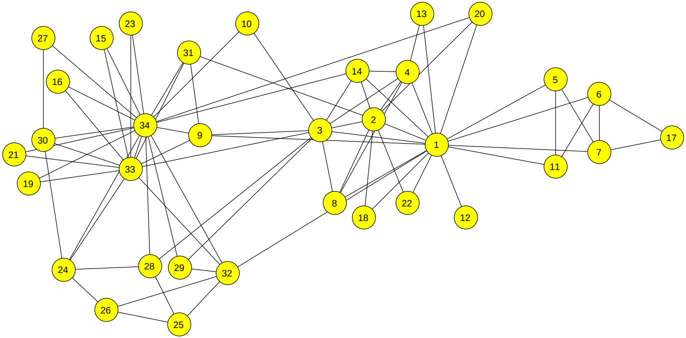
Given a network . . .



- explore the instance
- derive its structure
- identify its properties

Scenario of Network Analysis

Given a network . . .



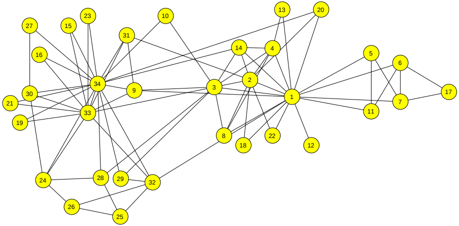
- explore the instance
- derive its structure
- identify its properties

How can we learn about the instance?

Structure

An Archetypal Example

"Zachary's Karate Club", a real, social network

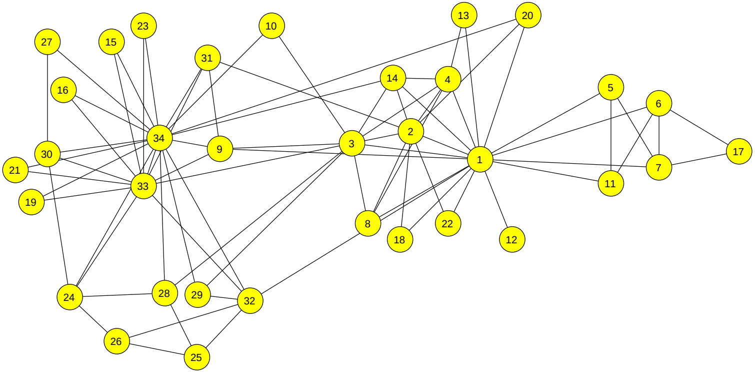


- 2 years of observation
- 34 vertices = members
- 78 edges = social ties
- club split up after dispute
- manager vs. trainers
- archon of toy examples

Caused by an *"unequal flow of sentiments and information across the ties"* a *"factional division led to a formal separation of the club"*.

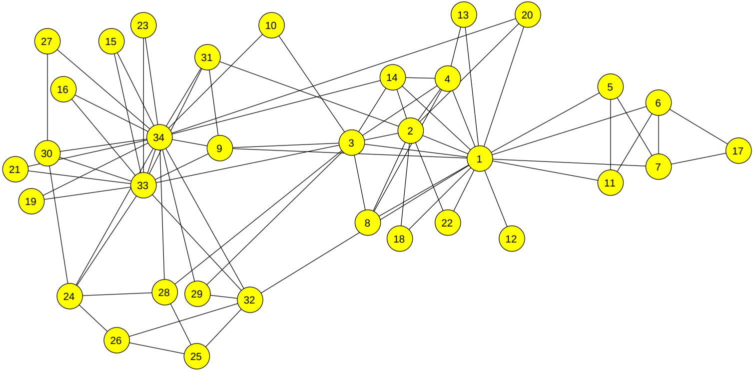
[Wayne Zachary: An Information Flow Model for Conflict and Fission in Small Groups, '77]

A Glimpse of Network Analysis



A Glimpse of Network Analysis

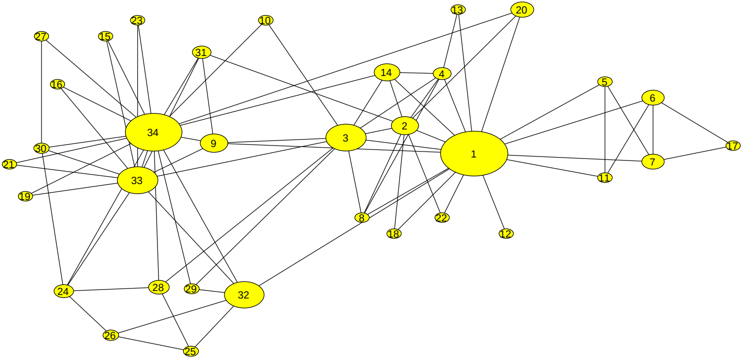
size, density,



#vertices = 34, #edges = 78, ∅-degree = 4.6

A Glimpse of Network Analysis

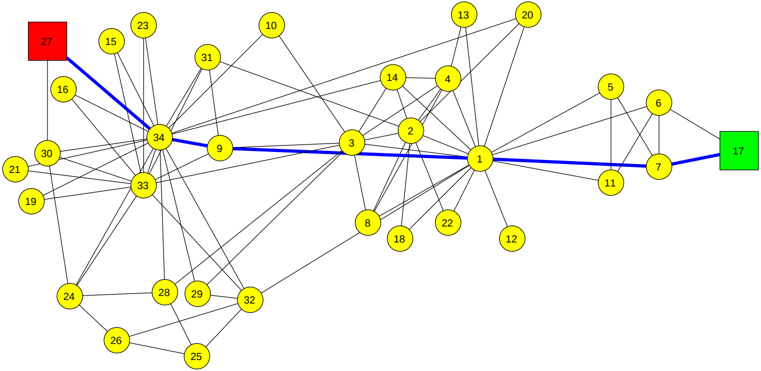
size, density, centrality / importance,



size = importance

A Glimpse of Network Analysis

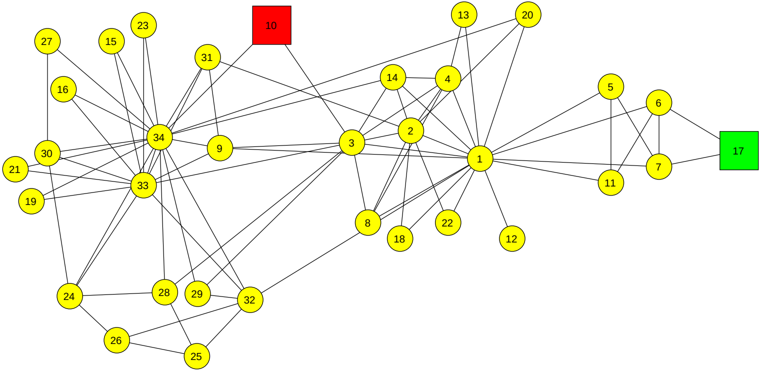
size, density, centrality / importance, distances,



distance between red and green

A Glimpse of Network Analysis

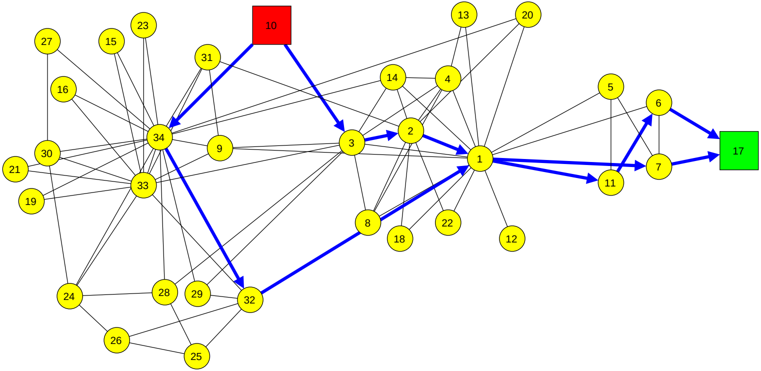
size, density, centrality / importance, distances, maximum flow,



flow from red to green

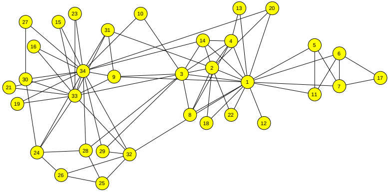
A Glimpse of Network Analysis

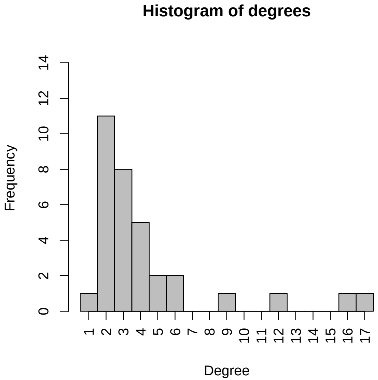
size, density, centrality / importance, distances, maximum flow,



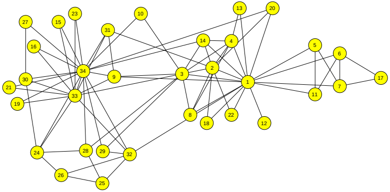
flow from red to green

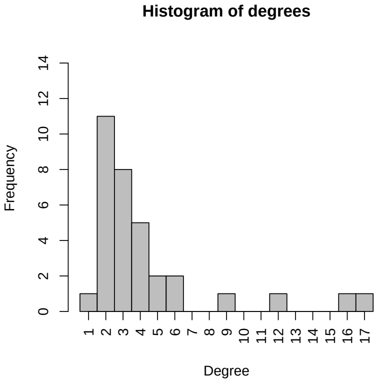
Degree Distribution





Degree Distribution





typical:
- (very few very-low-degree vertices)
- exponential-like distribution
- long tail or rare high degrees

Two Basic Models and their Degrees

Question: What is a reasonable random model emulating real networks?

Two Basic Models and their Degrees

Question: What is a reasonable random model emulating real networks?

Gilbert's model $G(n, p)$ / Erdős-Rény model $G(n, m)$

1. introduce n vertices
2. for each pair $\{u, v\} \in \binom{V}{2}$, connect $\{u, v\}$ with probability p

degree distribution (binomial): $P(d) = \binom{n-1}{d} \cdot p^d (1-p)^{n-1-k}$

Two Basic Models and their Degrees

Question: What is a reasonable random model emulating real networks?

Gilbert's model $G(n,p)$ / Erdős-Rényi model $G(n,m)$

1. introduce n vertices
2. for each pair $\{u,v\} \in \binom{V}{2}$, connect $\{u,v\}$ with probability p

degree distribution (binomial): $P(d) = \binom{n-1}{d} \cdot p^d(1-p)^{n-1-k}$

Cumulative Advantage / Preferential Attachment (n,a)

for vertices $v_1,\ldots,v_n$ do
    add vertex v_i to graph
    for v_i's edges $e_1,\ldots,e_a$ do
        connect v_i to (other) (non-adjacent) vertex u, with prob.$\sim \deg(u)$

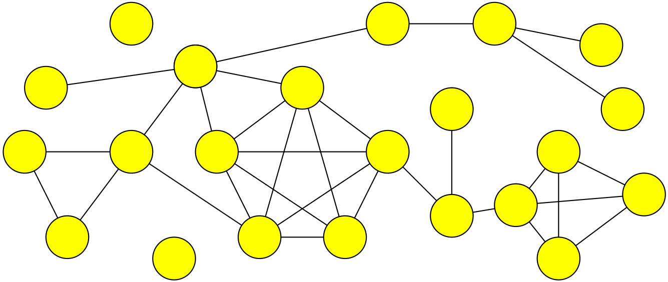
degree distribution: $P(d) \sim d^{-\gamma}$ (with $\gamma = 3$ for this specific setup)

Core-Decomposition: Definition

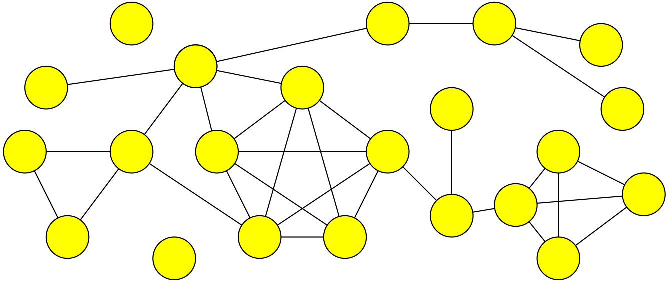
Karlsruher Institut für Technologie

Definition (k-core of a graph)

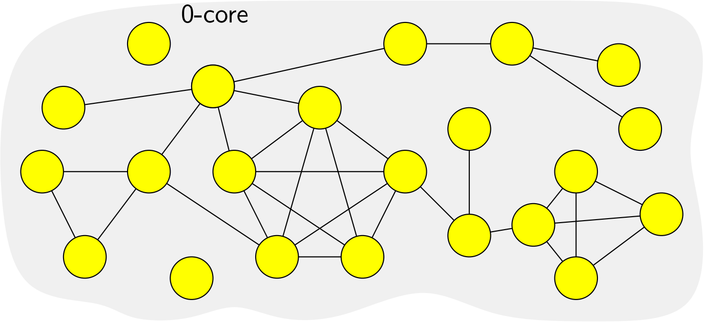
Maximum subset of vertices $V_k \subseteq V$ such that each $v \in V_k$ has at least k neighbors in V_k (i.e.: $\forall v \in V_k : |N(v) \cap V_k| \geq k$).



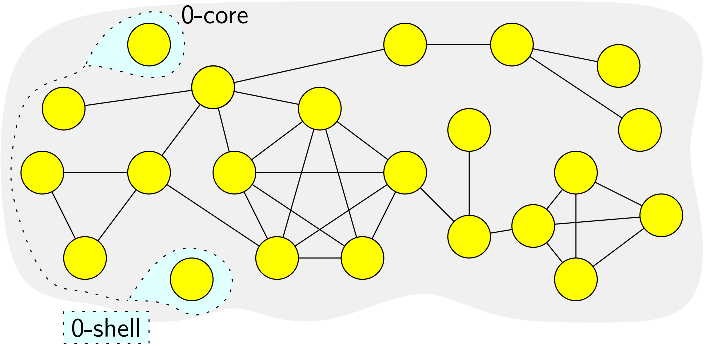
Core-Decomposition: Example



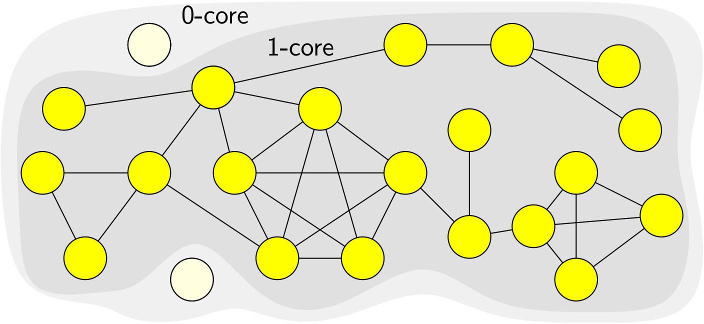
Core-Decomposition: Example



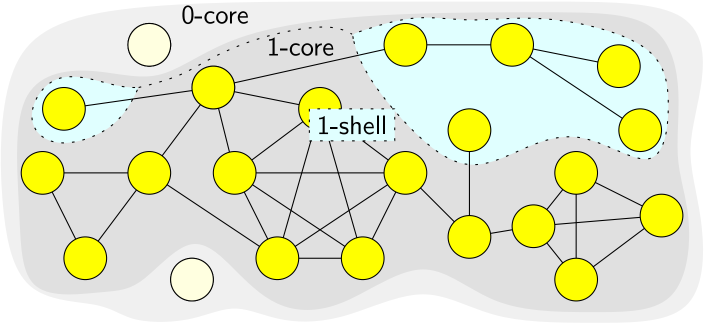
Core-Decomposition: Example



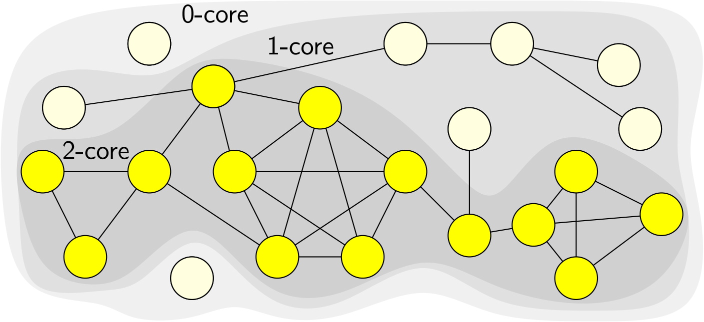
Core-Decomposition: Example



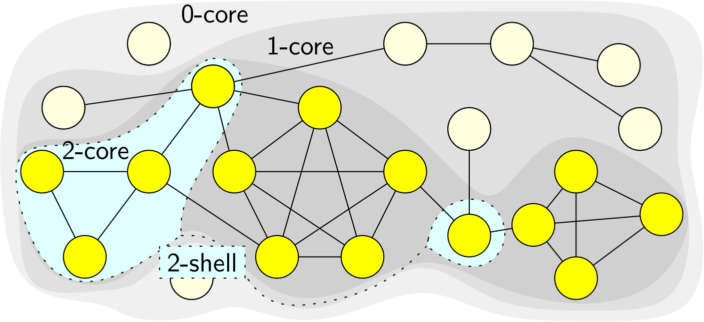
Core-Decomposition: Example



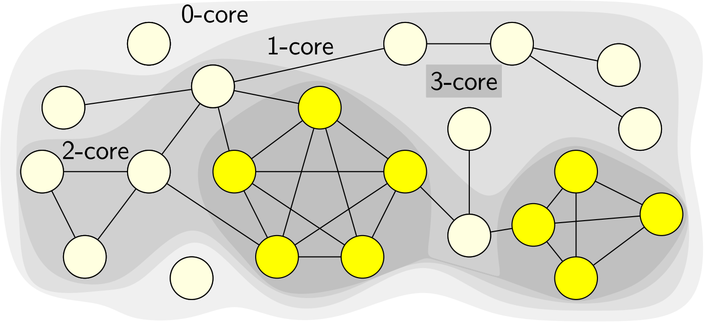
Core-Decomposition: Example



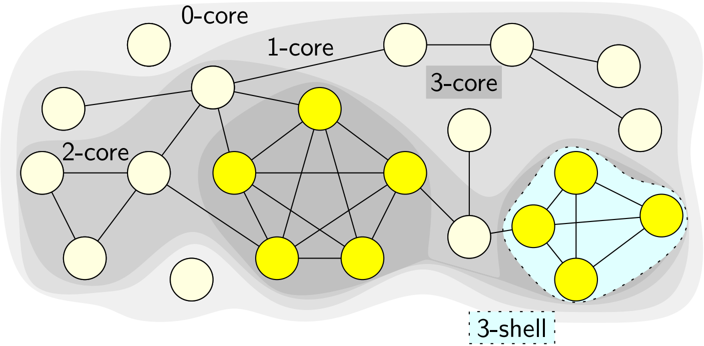
Core-Decomposition: Example



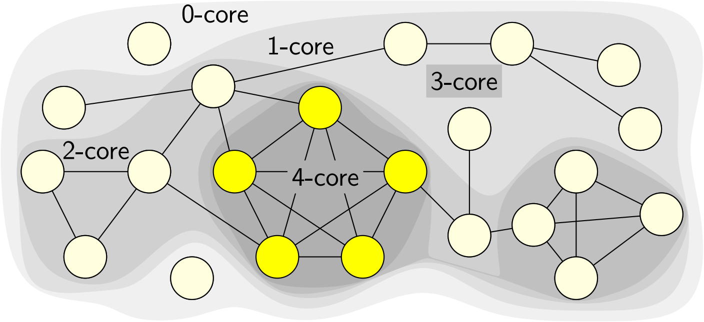
Core-Decomposition: Example



Core-Decomposition: Example



Core-Decomposition: Example



Centrality

Karlsruher Institut für Technologie

Idea: "[quantify intuition that some vertices are more central than others]"[1]
(note: *coreness* and *degree* are special measures for centrality)

[1] [[Brandes, Erlebach (eds.) '05, Network Analysis, Methodological Foundations]]

Centrality

Karlsruher Institut für Technologie

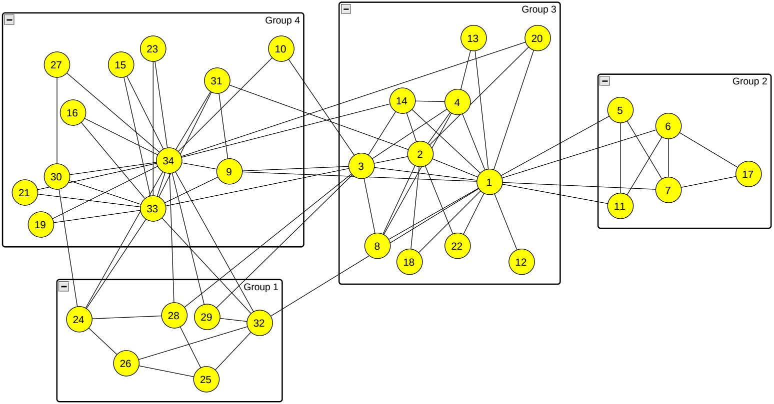
Idea: “[quantify intuition that some vertices are more central than others]”[1]
(note: *coreness* and *degree* are special measures for centrality)
Other important centrality measures are:

- *eccentricity:* $1/$ distance to farthest vertex in G
- *closeness:* $1/$ sum of distances to all vertices in G
- *stress:* total number of shortest paths requiring vertex v
- *betweenness:* sum of ratios of shortest paths requiring vertex v (over all pairs $s, t \neq v$)
- *reach:* maximum over all shortest s-t-paths v participates in: $\min\{\text{distance}(s, v), \text{distance}(v, t)\}$
- *Katz:* spectral, random walk, PageRank TODO

[1] [[Brandes, Erlebach (eds.) ’05, Network Analysis, Methodological Foundations]]

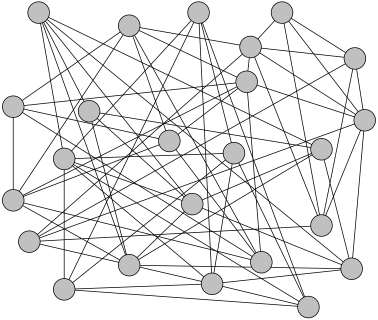
A Glimpse of Network Analysis

graph clustering / detecting communities

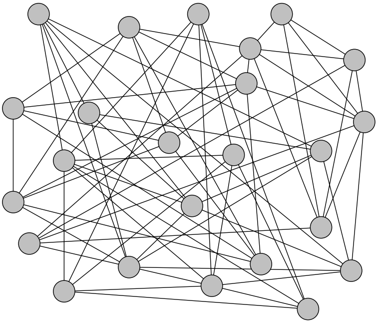


box = cluster

Why Graph Clustering?



Why Graph Clustering?

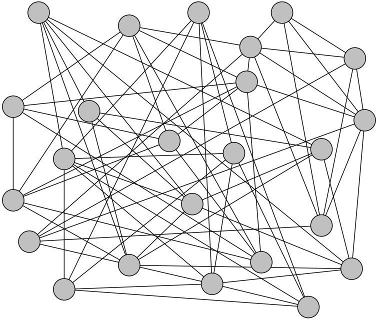


=

nodes	24
edges	57
cyclic	yes
planar	no
components	1
density	0.2065
avg. degree	4.75
core depth	4
triples	233
triangles	40
transitivity	0.515
clust. coeff.	0.6157

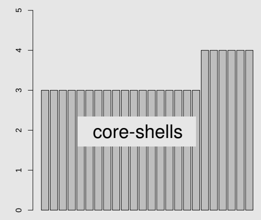
diverse field of network analysis

Why Graph Clustering?



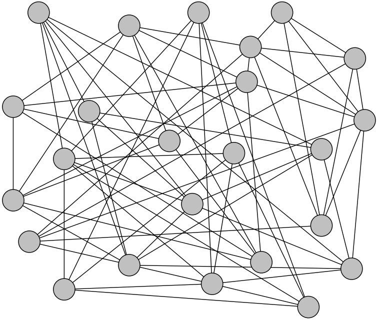
=

nodes	24
edges	57
cyclic	yes
planar	no
components	1
densit	
avg. d	
core d	
triples	
triangl	
transit	
clust. c	



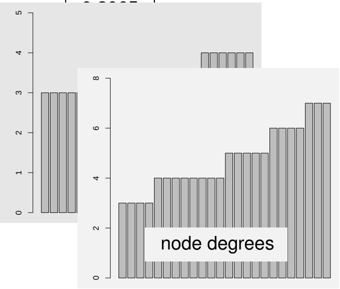
diverse field of network analysis

Why Graph Clustering?



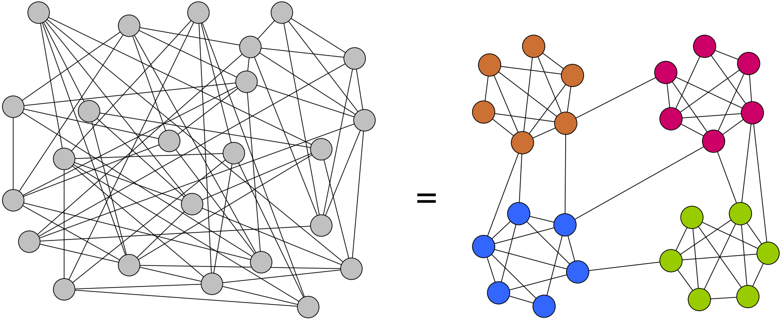
=

nodes	24
edges	57
cyclic	yes
planar	no
components	1
density	
avg. d	
core d	
triples	
triangl	
transit	
clust. c	



diverse field of network analysis

Why Graph Clustering?

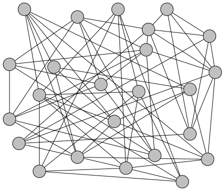


diverse field of network analysis
graph clustering = search for structure in networks

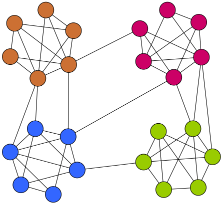
Structure

1. Introduction
   - Scenario: Network Analysis
   - Paradigm of Clustering
   - Example Applications
2. Formalization of Aims and Objectives
   - Objective Functions
3. Algorithmic Approaches
   - Greedy Merge
   - Local Moving and Multilevel
   - Clustering with Minimum-Cut Tree
   - Integer Linear Programs
   - Other Algorithmic Approaches
4. Experimental Evaluation
   - The Role of Test Data in Algorithm Engineering
   - Comparing Clusterings
5. Dynamic Graph Clustering
   - Online Dynamic Clustering
   - Offline Dynamic Clustering
6. Appendix

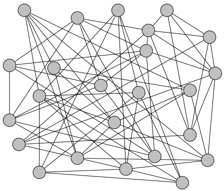
Clustering: Intuition to Formalization



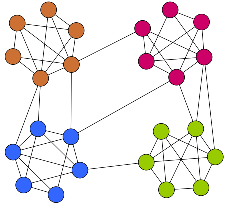
- **Task:** partition graph into natural groups
- **Paradigm:** intra-cluster density vs. inter-cluster sparsity



Clustering: Intuition to Formalization



- **Task:** partition graph into natural groups
- **Paradigm:** intra-cluster density vs. inter-cluster sparsity



Different approaches exist to formalize this paradigm, usually:

Paradigm of Graph Clustering

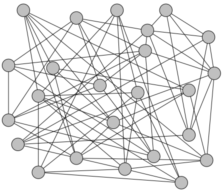
Intra-cluster density vs. inter-cluster sparsity

$\Downarrow$

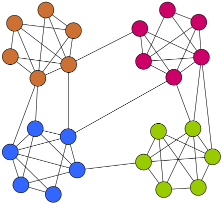
Mathematical Formalization

quality measures for clusterings

Clustering: Intuition to Formalization



- **Task:** partition graph into natural groups
- **Paradigm:** intra-cluster density vs. inter-cluster sparsity



Different approaches exist to formalize this paradigm, usually:

Paradigm of Graph Clustering

Intra-cluster density vs. inter-cluster sparsity

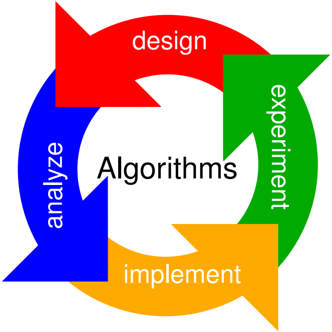
$$\Downarrow$$

Mathematical Formalization

quality measures for clusterings

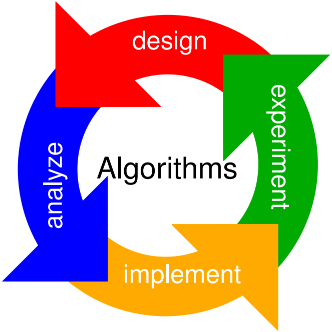
- Many exist, optimization generally (NP-)hard
- There is no single, universally best strategy

Algorithm Engineering



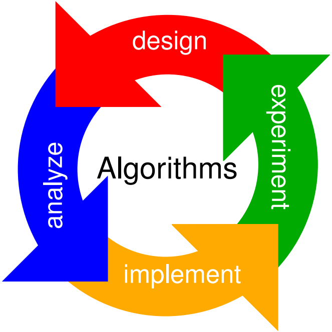
- modelling reality is hard

Algorithm Engineering



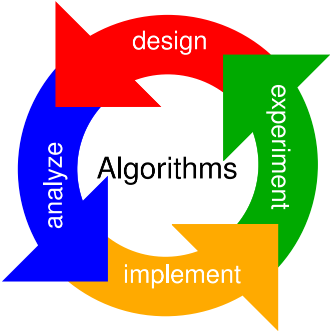
- modelling reality is hard
- finding optima is hard

Algorithm Engineering



- modelling reality is hard
- finding optima is hard
- satisfying needs of application is hard

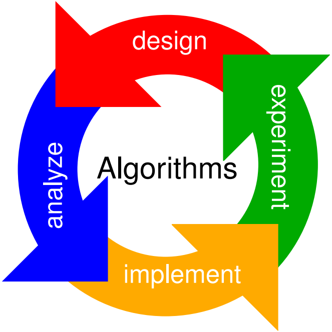
Algorithm Engineering



- modelling reality is hard
- finding optima is hard
- satisfying needs of application is hard

- still, we do need to cluster

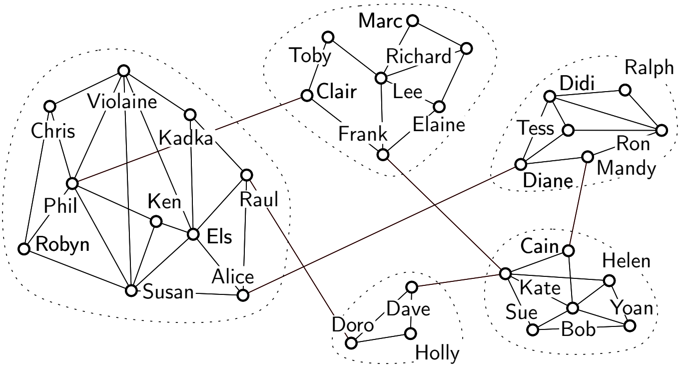
Algorithm Engineering



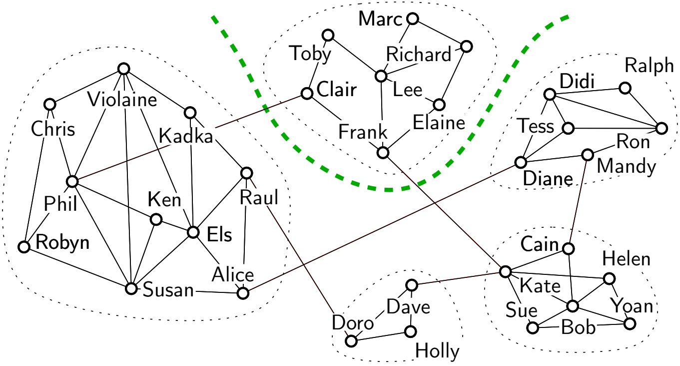
- modelling reality is hard
- finding optima is hard
- satisfying needs of application is hard

- still, we do need to cluster
- $\Rightarrow$ need good foundation

Formalization via Bottleneck



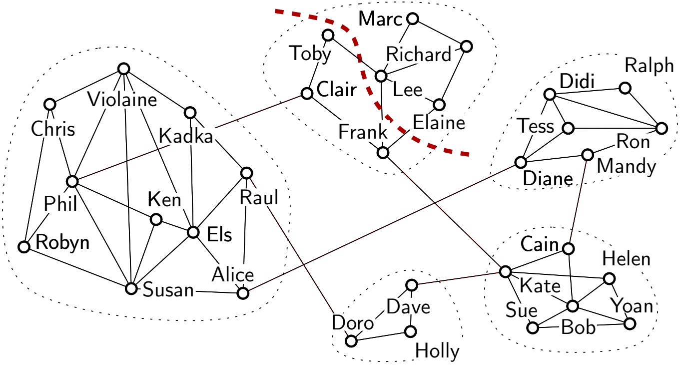
Formalization via Bottleneck



Quality of the clustering, upper cluster:

- inter-cluster sparsity: 2 *edges for cutting off* 7 *nodes* **(cheap)**

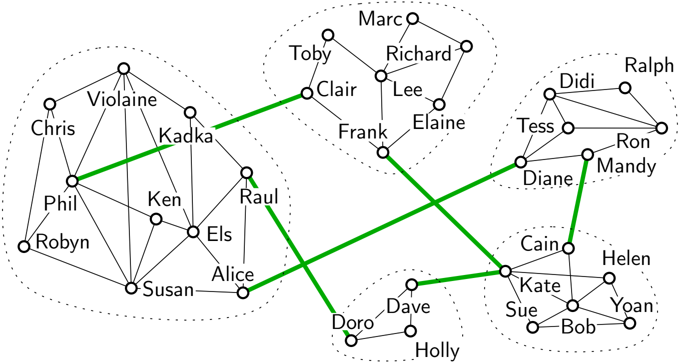
Formalization via Bottleneck



Quality of the clustering, upper cluster:

- inter-cluster sparsity: 2 *edges for cutting off* 7 *nodes* **(cheap)**
- intra-cluster density: best addit. cut:
  3 *edges for cutting off* 4 *nodes* **(expensive)**

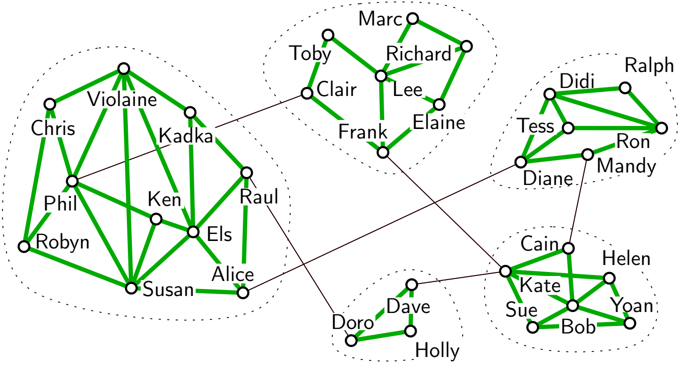
Formalization: Counting Edges



Measuring clustering quality by counting edges:

- inter-cluster sparsity: 6 *edges of ca. 800 node pairs* **(few)**

Formalization: Counting Edges



Measuring clustering quality by counting edges:

- inter-cluster sparsity: 6 *edges of ca. 800 node pairs* **(few)**
- intra-cluster density: 53 *edges of* 99 *node pairs* **(many)**

Clustering vs. Partitioning

	clustering	**partitioning**
purpose	analysis (pred.)	handling of instance
. . . and then?	zoom/abstraction	computations on parts
# of parts	open	predefined (upper bound)
size of parts	open	upper bound (or even fixed)
criteria	various (later)	weighted cuts
constraints	often none	see above
applications	various (later)	often: distributed finite element methods on 3d-meshes of objects

Bicriterial Formulations

observations:

1. clusterings often “nice” if balanced (like partition)
2. *intra-density vs. inter-sparsity* is bicriterial

bicriterial (or multi-) measures for clusterings can help:

- constrain sparsity within clusters
- constrain density between clusters
- explicitly formulate desiderata

(more on bicriteria later)

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality
- disjoint cliques should approach maximum quality

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality
- disjoint cliques should approach maximum quality
- locality of the measure (being better/worse in one part does not depend on what is done in other part of graph)

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality
- disjoint cliques should approach maximum quality
- locality of the measure (being better/worse in one part does not depend on what is done in other part of graph)
- double the instance, what should happen . . . same result

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality
- disjoint cliques should approach maximum quality
- locality of the measure (being better/worse in one part does not depend on what is done in other part of graph)
- double the instance, what should happen . . . same result
- comparable results across instances

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality
- disjoint cliques should approach maximum quality
- locality of the measure (being better/worse in one part does not depend on what is done in other part of graph)
- double the instance, what should happen . . . same result
- comparable results across instances
- fulfill the desiderata of the application

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality
- disjoint cliques should approach maximum quality
- locality of the measure (being better/worse in one part does not depend on what is done in other part of graph)
- double the instance, what should happen . . . same result
- comparable results across instances
- fulfill the desiderata of the application
- . . .

Postulations to a Measure

Given a graph G and a clustering $\mathcal{C}$, a *quality measure should behave as follows:*

- more intra-edges $\Rightarrow$ higher quality
- less inter-edges $\Rightarrow$ higher quality
- cliques must never be separated
- clusters must be connected
- random clusterings should have bad quality
- disjoint cliques should approach maximum quality
- locality of the measure (being better/worse in one part does not depend on what is done in other part of graph)
- double the instance, what should happen ... same result
- comparable results across instances
- fulfill the desiderata of the application
- ...

exercise: choose desiderata and design a measure!

A Theorem of Impossibility

A warning theorem on the field of data clustering:

Theorem

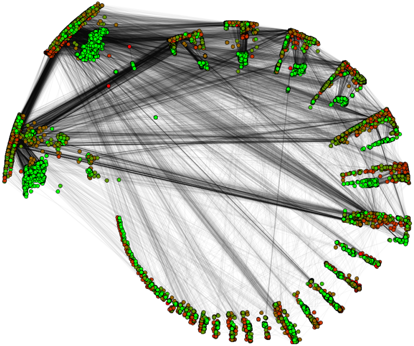
Given set S.
Let $f : d \mapsto \Gamma$ be a function on a distance function d on set S, returning a clustering Γ.
No function f can simultaneously fulfill the following:

- *Scale-Invariance*
  for any distance function d and any $\alpha > 0$, we have $f(d) = f(\alpha \cdot d)$
- *Richness*
  for any given clustering Γ, we should be able to define a distance function d such that $f(d) = \Gamma$
- *Consistency*
  if we build d' from d by reducing intra-distances and increasing inter-distances, we should have $f(d') = f(d)$

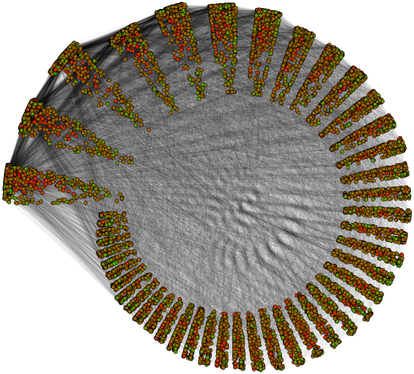
[Jon Kleinerg: An Impossibility Theorem for Clusterings, 2002]

Still, there is structure in networks!

we want/need to find it, and it is doable in practice



AS network, decomposed by a clustering; nodes with a high (low) betweenness are colored red (green)



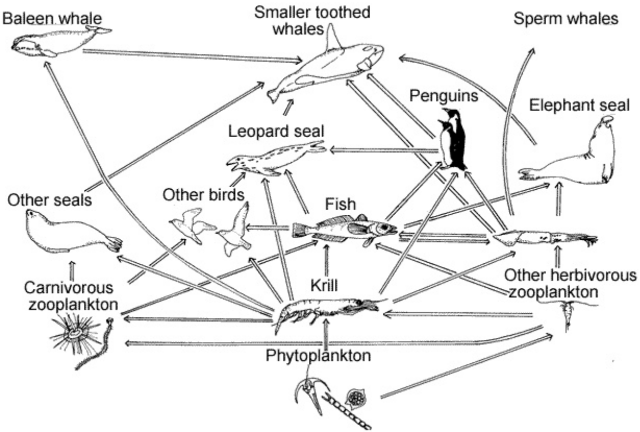
a network created with BRITE, designed to emulate the AS topology

[Görke at al.: LunarVis – analytic visualizations of large graphs, 2008]
[Medina et al.: BRITE – an approach to universal topology generation, 2001]

Structure

1. Introduction
   - Scenario: Network Analysis
   - Paradigm of Clustering
   - Example Applications
2. Formalization of Aims and Objectives
   - Objective Functions
3. Algorithmic Approaches
   - Greedy Merge
   - Local Moving and Multilevel
   - Clustering with Minimum-Cut Tree
   - Integer Linear Programs
   - Other Algorithmic Approaches
4. Experimental Evaluation
   - The Role of Test Data in Algorithm Engineering
   - Comparing Clusterings
5. Dynamic Graph Clustering
   - Online Dynamic Clustering
   - Offline Dynamic Clustering
6. Appendix

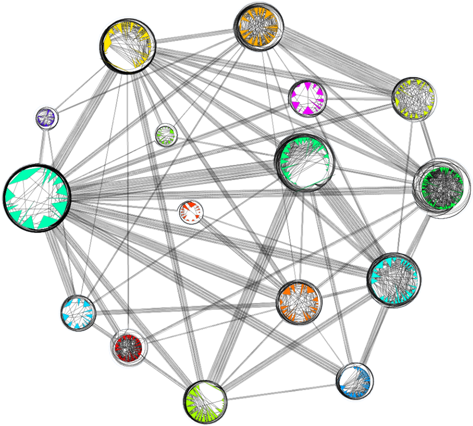
Applications



A simplified ecosystem: the antarctic food web
(source: Antarctica)

cluster ≈ self-sustaining / indivisible subsystem

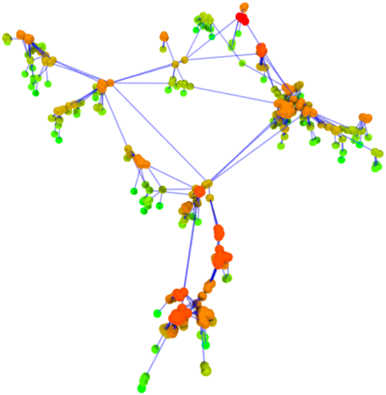
Applications



Company-internal email traffic, groups are departments

(source: )

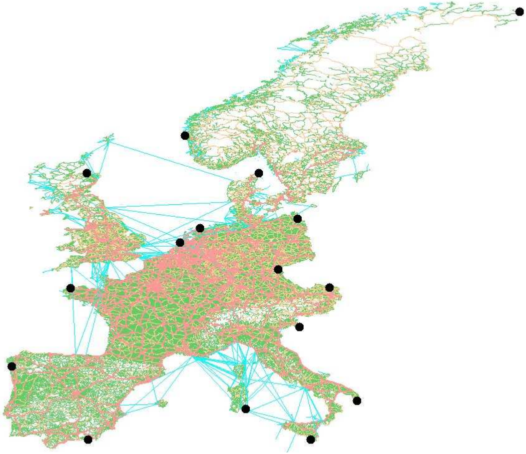
Applications



Excerptof the network of Amazon recommendations, around
"*VW Beetle Repairs*"
(source: [Gaertler 07])

cluster $\approx$ customer profile

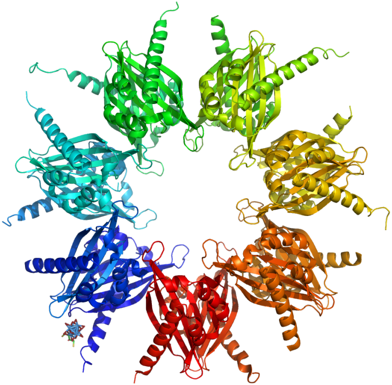
Applications



Road network of Europe, 18M vertices, 42M vertices
(source: PTV)

cluster ≈ urban areas used for preprocessing in route planning

Applications

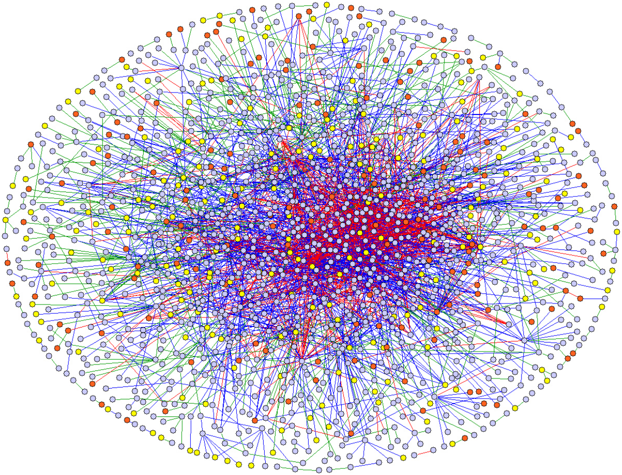


molecular structure of a protein

(Ca^{2+}/Calmodulin-dependent kinase II (CaMKII)
source: protein database www.rcsb.org)

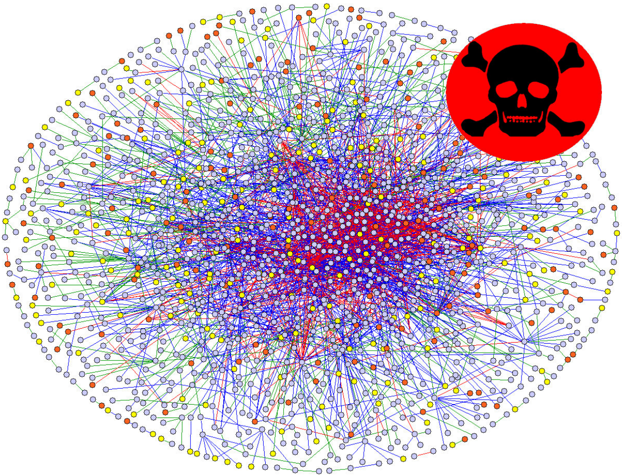
cluster ≈ functional unit (*domain*) of a protein

Applications



protein interactions
(source: Max-Delbrück-Centre for molecular medicine, www.mdc-berlin.de)

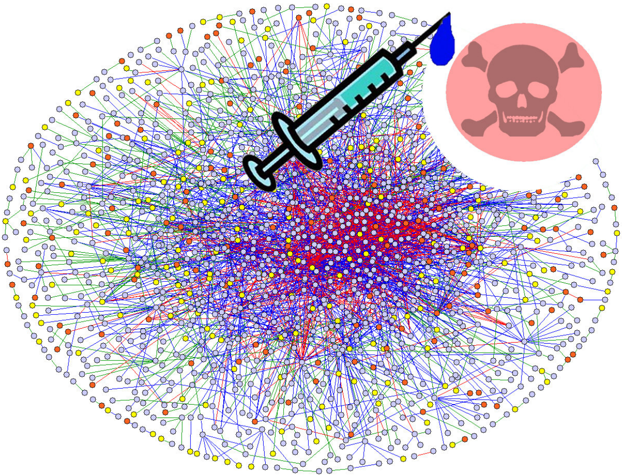
Applications



protein interactions
(source: Max-Delbrück-Centre for molecular medicine, www.mdc-berlin.de)

cluster ≈ isolatable seat of disease

Applications



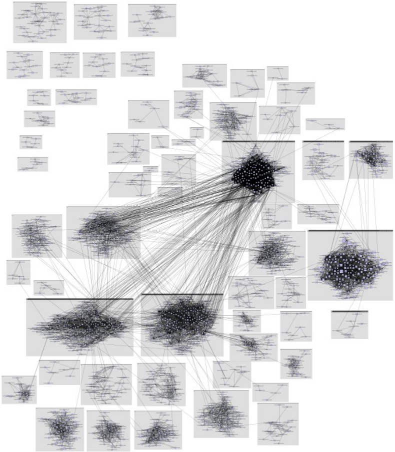
protein interactions
(source: Max-Delbrück-Centre for molecular medicine, www.mdc-berlin.de)

cluster ≈ isolatable seat of disease

Application: Shopping Data

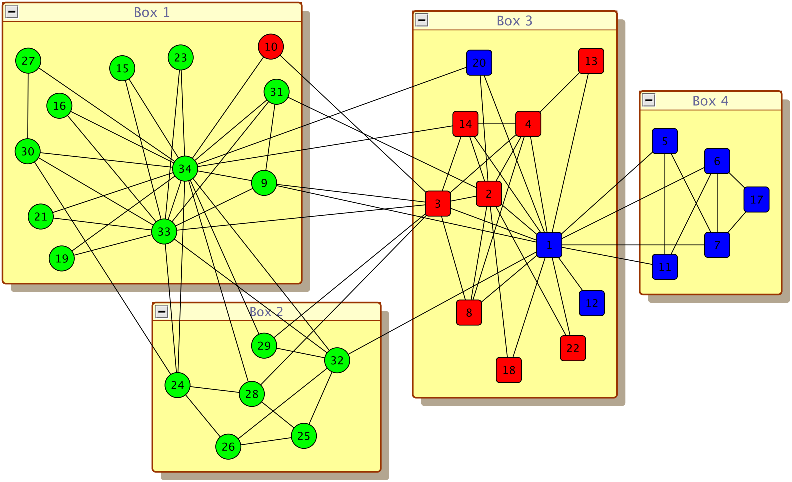
Data:

- **product base**
  product, classification, brand, setup block, . . .
- **store base**
  location, type, size, catchment area, . . .
- **customer base**
  customer number, age, gender, post code, . . .
- **receipts**
  “Who bought what, when and where?”



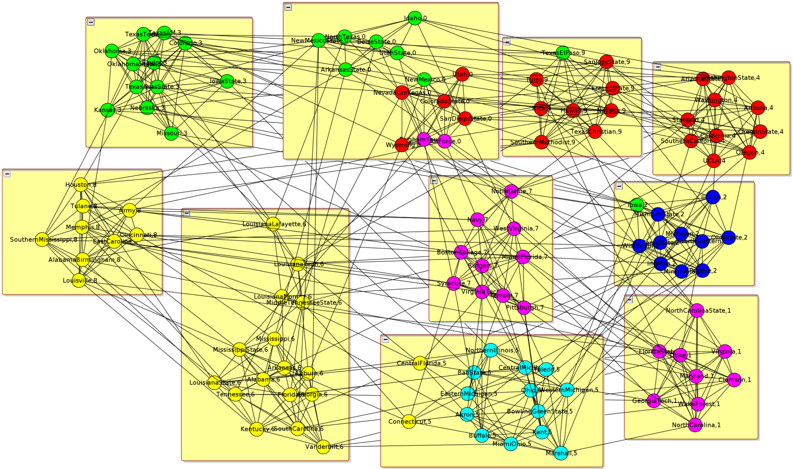
(example network of *receipt-similarity* one store, one month, clustered)

Scaling of Real-World Instances



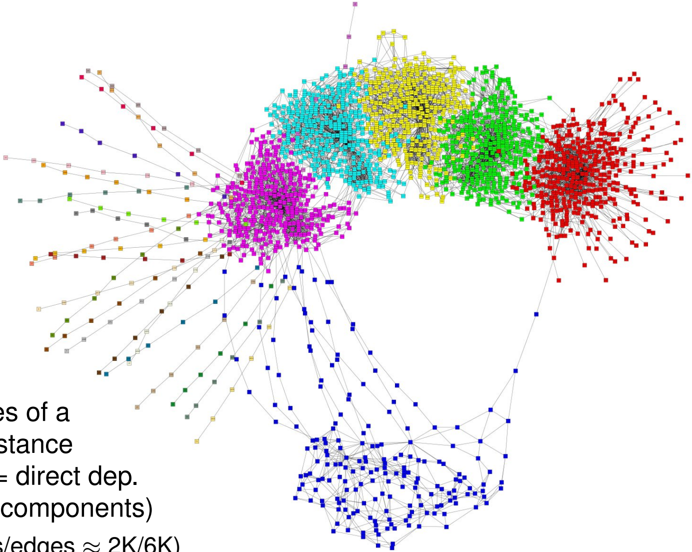
„Zachary's Karate Club")
(vertices/edges = 34/78)

Scaling of Real-World Instances



„US college football“ teams and matches
(vertices/edges = 115/616)

Scaling of Real-World Instances

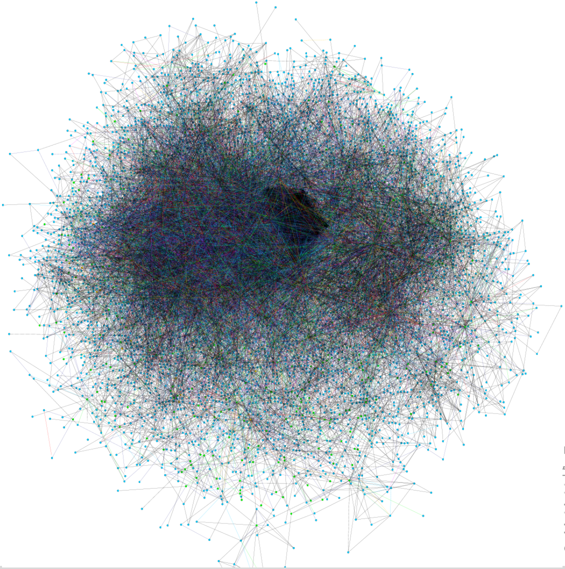


variables of a
SAT-instance
edges = direct dep.
(electr. components)

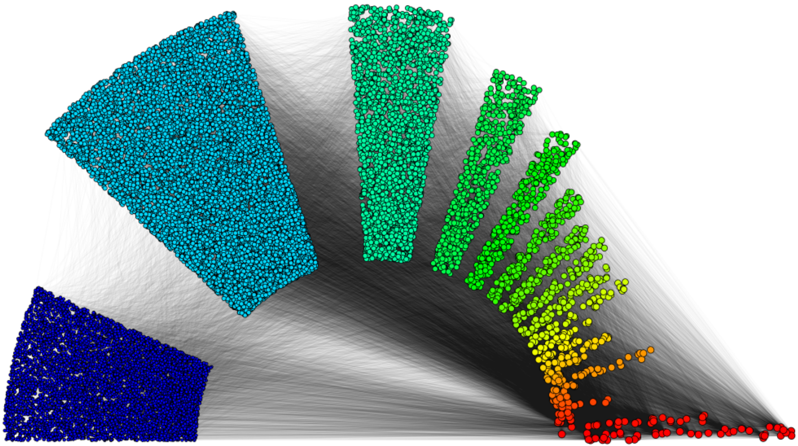
(vertices/edges ≈ 2K/6K)

Scaling of Real-World Instances

sci. collaborations:
3-hop neighorhood
von D. Wagner
(*DBLP*)
(vertices/edges ≈ 10k/40k)



Scaling of Real-World Instances



physical Internet: autonomous systemes

(vertices/edges $\approx$ 20K/60K)

Scaling of Real-World Instances

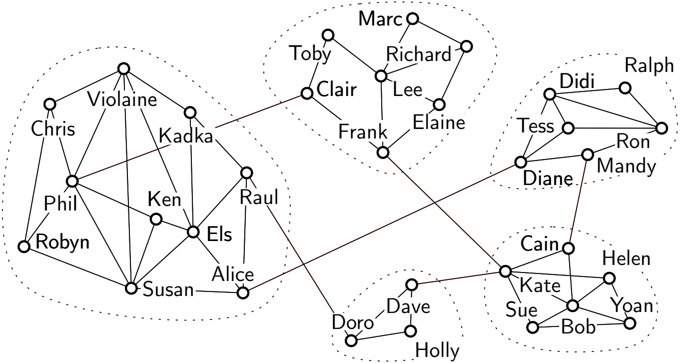
...no limit to be expected...

instance	vertices	edges
coauthors in *DBLP*	300K	1M
roads in the USA	24M	60M
WWW: .UK-domain '02	20M	500M
(neurons in human brain	$\gtrsim 10^{11}$	$\sim 10^{17}$)

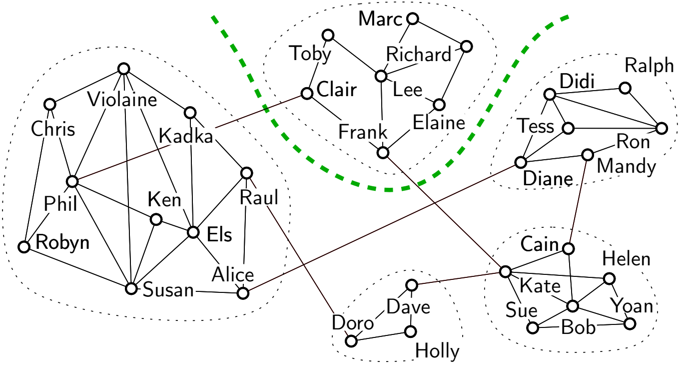
Structure

Structure

Formalization via Bottleneck



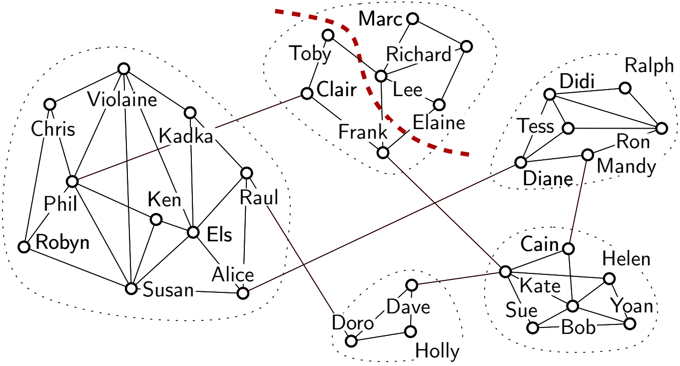
Formalization via Bottleneck



Quality of the clustering, upper cluster:

- inter-cluster sparsity: *2 edges for cutting off 7 nodes* **(cheap)**

Formalization via Bottleneck



Quality of the clustering, upper cluster:

- inter-cluster sparsity: *2 edges for cutting off 7 nodes* **(cheap)**
- intra-cluster density: best addit. cut:
  3 edges for cutting off 4 nodes **(expensive)**

Examples: Conductance, Expansion

generalization to case of weighted edges $\omega(e) \not\equiv 1$
e.g.: $\omega(E') = \sum_{e \in E'} \omega(e)$ or $\omega(v) = \sum_{e \sim v} \omega(e)$

conductance of a cut $(C, V \setminus C)$:

$$\varphi(C, V \setminus C) := \frac{\omega(E(C, V \setminus C))}{\min\left\{\sum_{v \in C} \omega(v), \sum_{v \in V \setminus C} \omega(v)\right\}}$$

(i.e.: thickness of bottleneck which cuts off C)

inter-cluster conductance $(\mathcal{C}) := 1 - \max_{C \in \mathcal{C}} \varphi(C, V \setminus C)$
(i.e.: $1-$ worst bottleneck induced by some $C \in \mathcal{C}$)

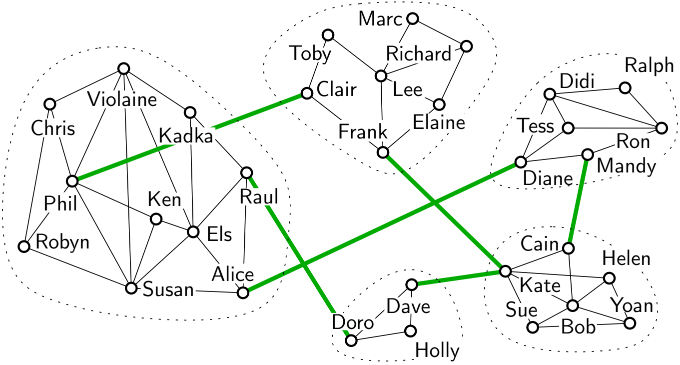
intra-cluster conductance $(\mathcal{C}) := \min_{C \in \mathcal{C}} \min_{P \uplus Q = C} \varphi_{|C}(P, Q)$
(i.e.: best bottleneck still left uncut inside some $C \in \mathcal{C}$)

expansion of a cut $(C, V \setminus C)$:

$$\psi(C, V \setminus C) := \frac{\omega(E(C, V \setminus C))}{\min\left\{|C|, |V \setminus C|\right\}}$$

(i.e.: in φ, replace $\omega(v)$ by 1; *intra-* and *inter-cluster expansion* analogously)

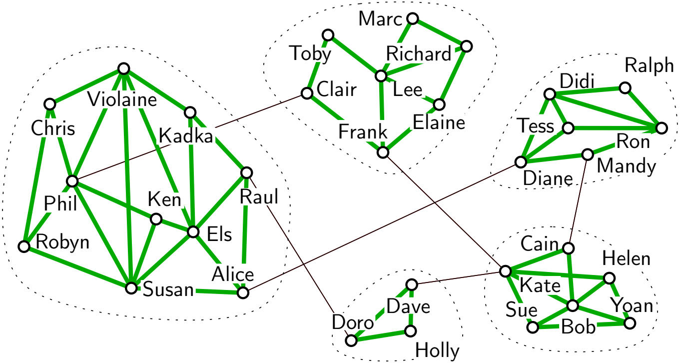
Formalization: Counting Edges



Measuring clustering quality by counting edges:

- inter-cluster sparsity: 6 *edges of ca. 800 node pairs* **(few)**

Formalization: Counting Edges



Measuring clustering quality by counting edges:

- inter-cluster sparsity: 6 *edges of ca. 800 node pairs* **(few)**
- intra-cluster density: 53 *edges of* 99 *node pairs* **(many)**

Example Counting Measures

coverage: $\text{cov}(\mathcal{C}) := \frac{\text{\# intra-cluster edges}}{\text{\# edges}}$

(i.e.: fraction of covered edges)

performance: $\text{perf}(\mathcal{C}) := \frac{\text{\# intra-cluster edges} + \text{\# absent inter-cluster edges}}{\frac{1}{2}n(n-1)}$

(i.e.: fraction of correctly classified pairs of nodes)

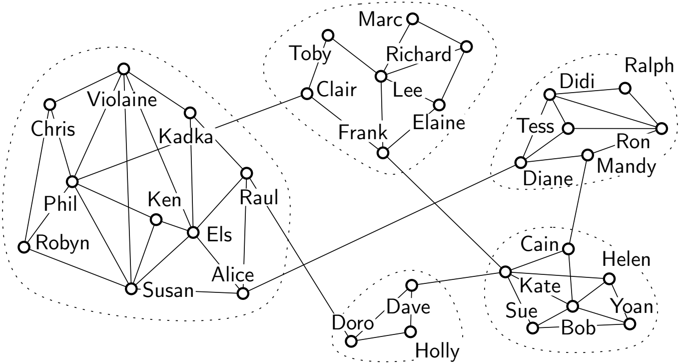
density: $\text{den}(\mathcal{C}) := \frac{1}{2}\frac{\text{\# intra-cluster edges}}{\text{\# possible intra-cluster edges}} + \frac{1}{2}\frac{\text{\# absent inter-cluster edges}}{\text{\# possible inter-cluster edges}}$

(i.e.: fractions of correct intra- and inter-edges)

modularity: $\text{mod}(\mathcal{C}) := \text{cov}(\mathcal{C}) - \mathbb{E}[\text{cov}(\mathcal{C})]$

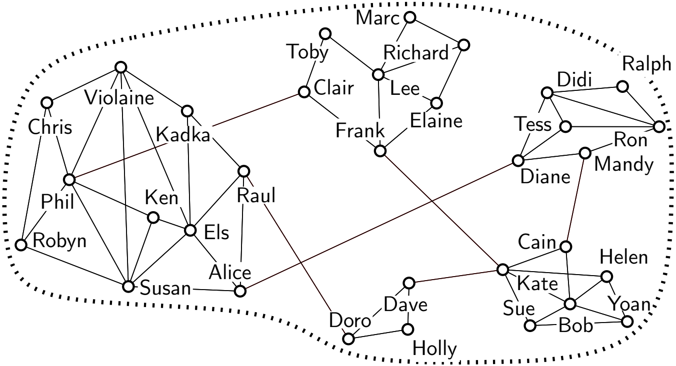
(i.e.: how clear is the clustering, compared to random network?)

Motivation for Modularity



- *coverage* $= \dfrac{\text{\# intra-cluster edges}}{\text{\# edges}} \approx 0.9$

Motivation for Modularity



- $\textit{coverage} = \frac{\text{\# intra-cluster edges}}{\text{\# edges}} \approx 0.9$
- **only one cluster $\Rightarrow$ *coverage* = 1.0**

A Promising Remedy

[Girvan and Newman: Finding and evaluating community structure in networks, '04]:

"... if we subtract from [coverage] the ***expected*** *value [...], we do get a useful measure."*

A Promising Remedy

[Girvan and Newman: Finding and evaluating community structure in networks, '04]:

*"... if we subtract from [coverage] the **expected** value [...], we do get a useful measure."*

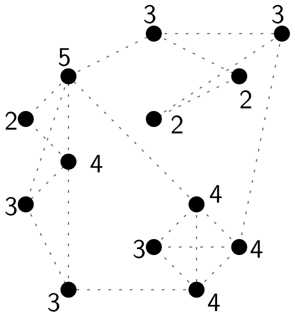
Modularity

$$\begin{aligned} \mathrm{mod}(\mathcal{C}) :=&\ \mathit{cov}(\mathcal{C}) - \mathbb{E}(\mathit{cov}(\mathcal{C})) \\ =&\ \frac{\#\text{ intra-cluster edges}}{|\#\mathit{edges}|} - \frac{1}{4|\#\mathit{edges}|^2} \sum_{C \in \mathcal{C}} \left(\sum_{v \in C} \deg(v) \right)^2 \end{aligned}$$

Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

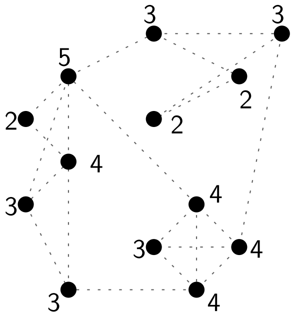
1. start with set V



Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

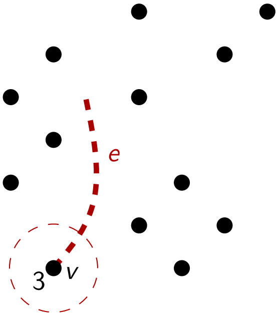
1. start with set V
2. keep expected degrees



Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

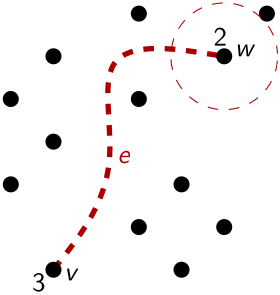
1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$



Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

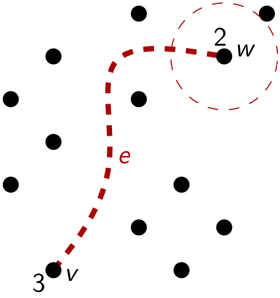
1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$



Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

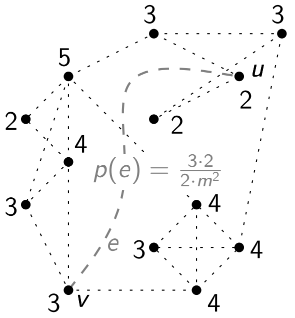
1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice



Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

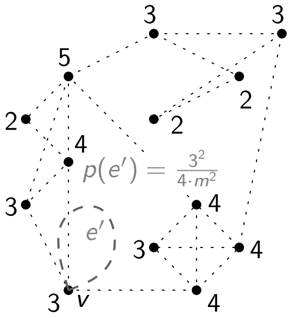
1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v) \cdot \deg(w)}{2|E|^2}$



Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

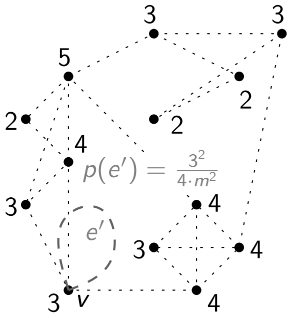
1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v) \cdot \deg(w)}{2|E|^2}$ (non-loop)



Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v)\cdot\deg(w)}{2|E|^2}$ (non-loop)

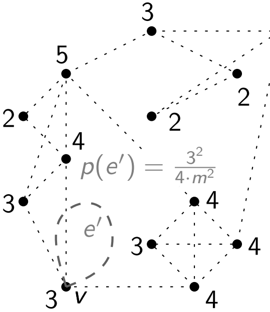


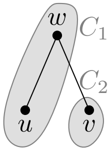
$$\text{mod}(\mathcal{C}) = \frac{\#\text{ intra-cluster edges}}{|\#edges|} - \frac{1}{4|\#edges|^2}\sum_{C\in\mathcal{C}}\left(\sum_{v\in C}\deg(v)\right)^2$$

Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

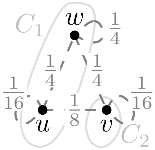
1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v) \cdot \deg(w)}{2|E|^2}$ (non-loop)





$G, \mathcal{C}(G)$

$\text{cov} = \frac{1}{2}$



(Ω, p),

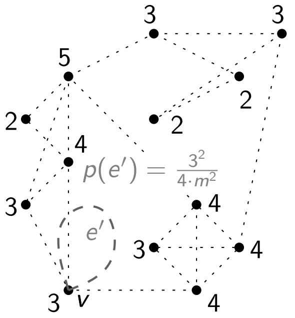
$\mathbb{E}[\text{cov}] = \frac{5}{8}$

$\text{mod} = -\frac{1}{8}$ (bad!)

Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v) \cdot \deg(w)}{2|E|^2}$ (non-loop)



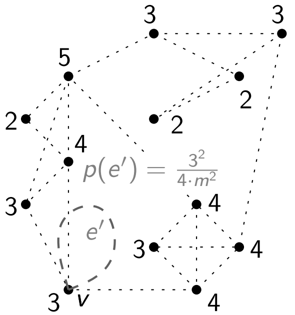
Some thoughts:

- (Ω, p) needs loops + parallel edges
  hazardous: e.g., paper on loop removal uses loop-agnostic formula!

Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v) \cdot \deg(w)}{2|E|^2}$ (non-loop)



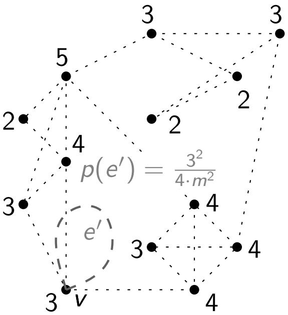
Some thoughts:

- (Ω, p) needs loops + parallel edges
  hazardous: e.g., paper on loop removal uses loop-agnostic formula!
- coverage is bad, why use it, why subtraction? **exercise: use performance**

Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v)\cdot\deg(w)}{2|E|^2}$ (non-loop)



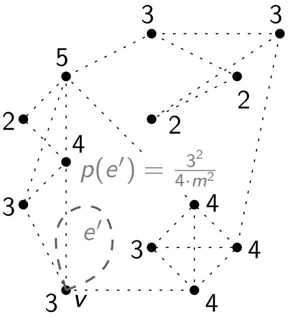
Some thoughts:

- (Ω, p) needs loops + parallel edges
  hazardous: e.g., paper on loop removal uses loop-agnostic formula!
- coverage is bad, why use it, why subtraction? **exercise: use performance**
- modularity vs. ground-truth & other indices?

Probability Space of Modularity

Intuition: Keep expected node degrees, randomly throw in edges

1. start with set V
2. keep expected degrees
3. edge attaches to node v with $p = \frac{\deg(v)}{2|E|}$
4. other end attaches to w with $p = \frac{\deg(w)}{2|E|}$
5. forward and backward $\rightsquigarrow$ count twice
6. $p(e) = \frac{\deg(v) \cdot \deg(w)}{2|E|^2}$ (non-loop)



Some thoughts:

- (Ω, p) needs loops + parallel edges
  hazardous: e.g., paper on loop removal uses loop-agnostic formula!
- coverage is bad, why use it, why subtraction? **exercise: use performance**
- modularity vs. ground-truth & other indices?
- actual optimization?

Modularity in Practice

- easy to use & implement
- reasonable behavior on many practical instances
- $\rightsquigarrow$ heavily used in various fields
  - ecosystem exploration
  - collaboration analyses
  - biochemistry
  - structure of the internet (AS-graph, www, routers)
- close to human intuition of quality
  [Görke et al.: Comp. aspects of lucidity-driven clustering, 2010]

Modularity in Practice

- easy to use & implement
- reasonable behavior on many practical instances
- $\rightsquigarrow$ heavily used in various fields
  - ecosystem exploration
  - collaboration analyses
  - biochemistry
  - structure of the internet (AS-graph, www, routers)
- close to human intuition of quality
  [Görke et al.: Comp. aspects of lucidity-driven clustering, 2010]

- scaling behavior (double instance, result differs) [folklore]
- non-locality of optimal clustering [folklore]
- resolution limit (no tiny and large clusters at the same time)
  [Fortunato and Barthelemy '07]
- large sparse graph $\rightsquigarrow$ high values, balanced clusters **exercise** [Good et al.: The performance of modularity maximization in practical contexts, 2009]

Modularity, Algorithmic Theory

The complexity of modularity optimization:

- finding $\mathcal{C}$ with maximum modularity is NP-hard
  $\rightsquigarrow$ reduction from 3-PARTITION
- restriction to $|\mathcal{C}| = 2$ also hard $\Rightarrow$ not FPT wrt. $|\mathcal{C}|$
- greedy maximization (later) does not approximate
- very limited families combinatorially solvable
- ILP-formulation, feasible for $\approx |V| \leq 200$

[Brandes et al.: On modularity clustering, 2008]

- diverse results on approximability on specific classes of graphs

[DasGupta, Devine: On the complexity of newman's community finding approach for biological and social networks, 2011]

Explicitly Bicriterial Approach

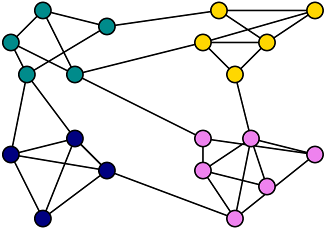
Optimization problem:

- guaranteed intra-cluster density
- good inter-cluster sparsity

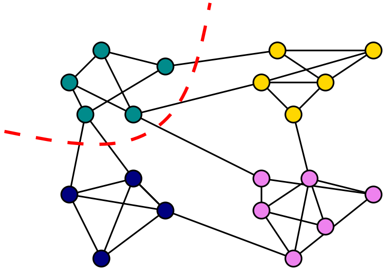
Approach:

- systematic collection of sparsity and density measures

Inter-cluster-sparsity: A cut-based point of view

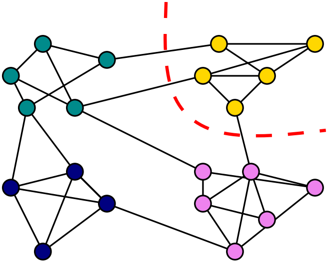


Inter-cluster-sparsity: A cut-based point of view



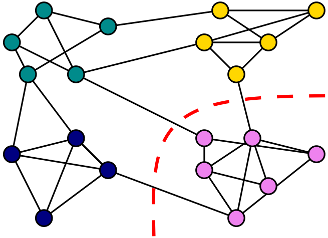
- **Isolated View:** Each cluster induces a cut

Inter-cluster-sparsity: A cut-based point of view



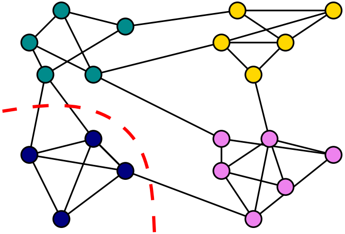
- **Isolated View:** Each cluster induces a cut

Inter-cluster-sparsity: A cut-based point of view



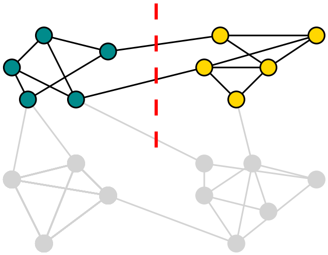
- **Isolated View:** Each cluster induces a cut

Inter-cluster-sparsity: A cut-based point of view



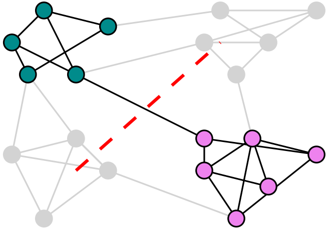
- **Isolated View:** Each cluster induces a cut

Inter-cluster-sparsity: A cut-based point of view



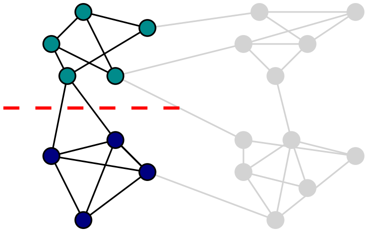
- **Isolated View:** Each cluster induces a cut
- **Pairwise View:** Each pair of clusters induces a cut in their subgraph

Inter-cluster-sparsity: A cut-based point of view



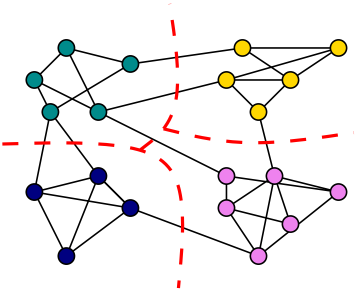
- **Isolated View:** Each cluster induces a cut
- **Pairwise View:** Each pair of clusters induces a cut in their subgraph

Inter-cluster-sparsity: A cut-based point of view



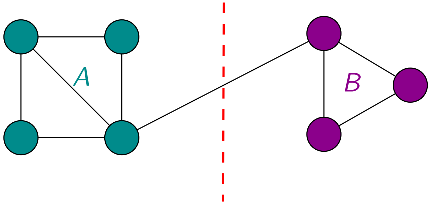
- **Isolated View:** Each cluster induces a cut
- **Pairwise View:** Each pair of clusters induces a cut in their subgraph

Inter-cluster-sparsity: A cut-based point of view

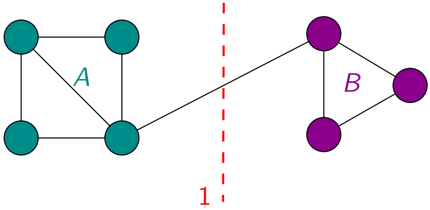


- **Isolated View:** Each cluster induces a cut
- **Pairwise View:** Each pair of clusters induces a cut in their subgraph
- **Global View:** A clustering with *k* clusters induces a k-way cut

Inter-cluster sparsity: Cut Measures

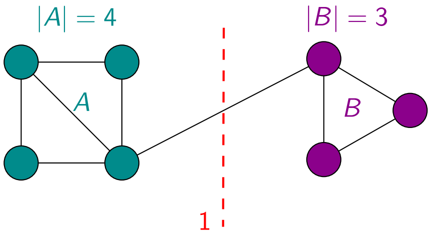


Inter-cluster sparsity: Cut Measures



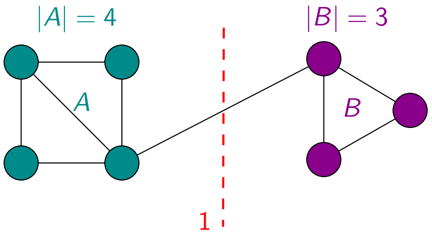
- Number of cut-edges: $|E(A, B)| = 1$

Inter-cluster sparsity: Cut Measures



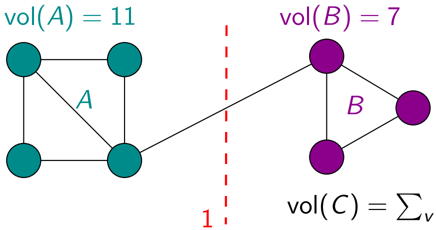
- Number of cut-edges: $|E(A, B)| = 1$
- Density: $\frac{|E(A,B)|}{|A||B|} = \frac{1}{12}$

Inter-cluster sparsity: Cut Measures



- Number of cut-edges: $|E(A, B)| = 1$
- Density: $\frac{|E(A,B)|}{|A||B|} = \frac{1}{12}$
- Expansion: $\frac{|E(A,B)|}{\min\{|A|,|B|\}} = \frac{1}{3}$

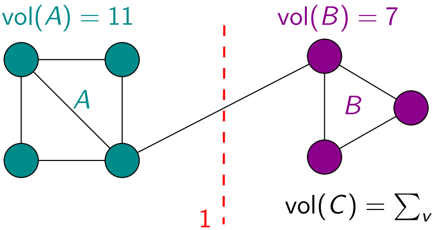
Inter-cluster sparsity: Cut Measures



$\mathrm{vol}(C) = \sum_{v \in C} \deg(v)$

- Number of cut-edges: $|E(A,B)| = 1$
- Density: $\frac{|E(A,B)|}{|A||B|} = \frac{1}{12}$
- Expansion: $\frac{|E(A,B)|}{\min\{|A|,|B|\}} = \frac{1}{3}$
- Conductance: $\frac{|E(A,B)}{\min\{\mathrm{vol}(A),\mathrm{vol}B\}} = \frac{1}{7}$

Inter-cluster sparsity: Cut Measures



$\text{vol}(C) = \sum_{v \in C} \deg(v)$

- Number of cut-edges: $|E(A, B)| = 1$
- Density: $\frac{|E(A,B)|}{|A||B|} = \frac{1}{12}$
- Expansion: $\frac{|E(A,B)|}{\min\{|A|,|B|\}} = \frac{1}{3}$
- Conductance: $\frac{|E(A,B)}{\min\{\text{vol}(A),\text{vol}B\}} = \frac{1}{7}$

$\Rightarrow$ Number of cut-edges and density can be generalized to k-way cuts

Inter-cluster Sparsity: Degrees of Freedom

Karlsruher Institut für Technologie

Set of cuts

- isolated (one for each cluster)
- pairwise (one for each pair of clusters)
- global (k-way cut)

Inter-cluster Sparsity: Degrees of Freedom

Set of cuts

- isolated (one for each cluster)
- pairwise (one for each pair of clusters)
- global (k-way cut)

Measures

- number of cut-edges
- density
- conductance
- expansion

Inter-cluster Sparsity: Degrees of Freedom

Set of cuts

- isolated (one for each cluster)
- pairwise (one for each pair of clusters)
- global (k-way cut)

Measures

- number of cut-edges
- density
- conductance
- expansion

Combinations

- average sparsity
- minimum sparsity

Inter-cluster Sparsity: Degrees of Freedom

Karlsruher Institut für Technologie

Set of cuts

- isolated (one for each cluster)
- pairwise (one for each pair of clusters)
- global (k-way cut)

Measures

- number of cut-edges
- density
- conductance
- expansion

Combinations

- average sparsity
- minimum sparsity

$\Rightarrow$ **14 (reasonable) inter-cluster sparsity measures**

Inter-cluster Sparsity: Degrees of Freedom

Karlsruher Institut für Technologie

Set of cuts

- isolated (one for each cluster)
- pairwise (one for each pair of clusters)
- global (k-way cut)

Measures

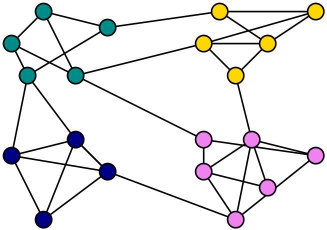
- number of cut-edges
- density
- conductance
- expansion

Combinations

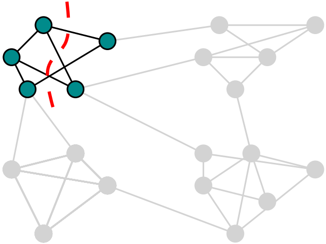
- average sparsity
- minimum sparsity

$\Rightarrow$ **14 (reasonable) inter-cluster sparsity measures**

Intra-cluster density

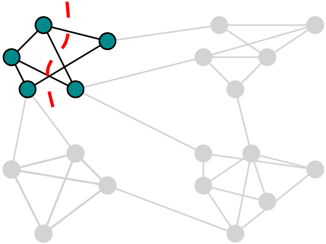


Intra-cluster density



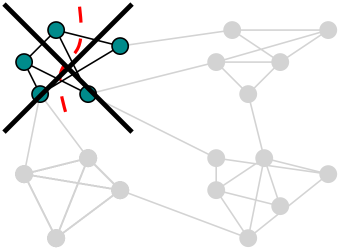
- Definitions analoguous to inter-cluster sparsity possible

Intra-cluster density



- Definitions analoguous to inter-cluster sparsity possible
- Finding cut with optimal density/conductance/expansion is NP-hard

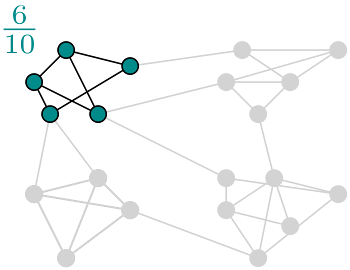
Intra-cluster density



- Definitions analoguous to inter-cluster sparsity possible
- Finding cut with optimal density/conductance/expansion is NP-hard

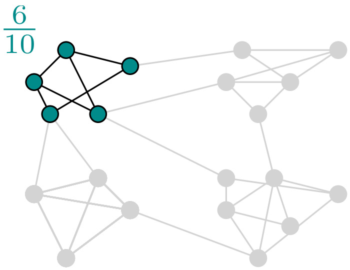
Intra-cluster density

Karlsruher Institut für Technologie



- Definitions analoguous to inter-cluster sparsity possible
- Finding cut with optimal density/conductance/expansion is NP-hard
- Practical approach: evaluate $\frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|}$

Intra-cluster density



- Definitions analoguous to inter-cluster sparsity possible
- Finding cut with optimal density/conductance/expansion is NP-hard
- Practical approach: evaluate $\frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|}$

$\Rightarrow$ **minimum/average/global intra-cluster density**

Problem Statement

Karlsruher Institut für Technologie

Density-Constrained Clustering[2]

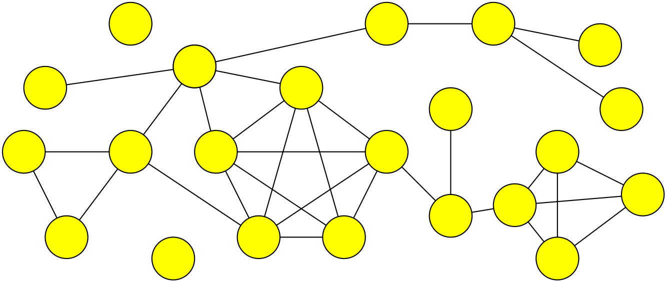
Given a graph $G = (V, E)$, among all clusterings with an intra-cluster density of no less than α, find a clustering $\mathcal{C}$ with optimum inter-cluster sparsity.

- 3 possible intra-cluster density measure
- 14 possible inter-cluster sparsity measures
- $\Rightarrow$ Family of 42 optimization problems

[2] [Schumm et al.: Density-constrained graph clustering, 2011]

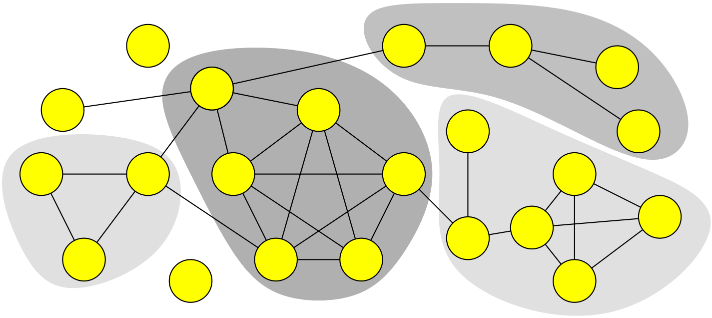
Cluster Editing Set

"How many edges must be inserted or deleted to arrive at disjoint cliques?"



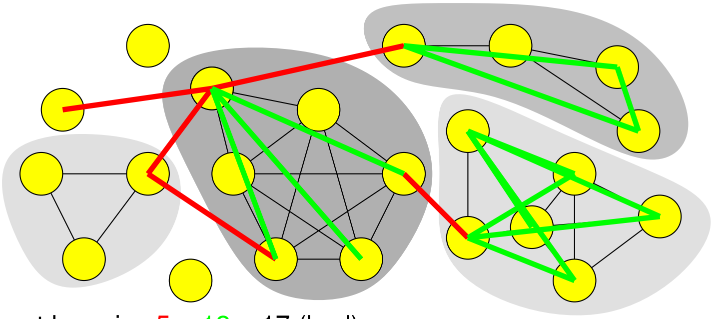
Cluster Editing Set

"How many edges must be inserted or deleted to arrive at disjoint cliques?"



Cluster Editing Set

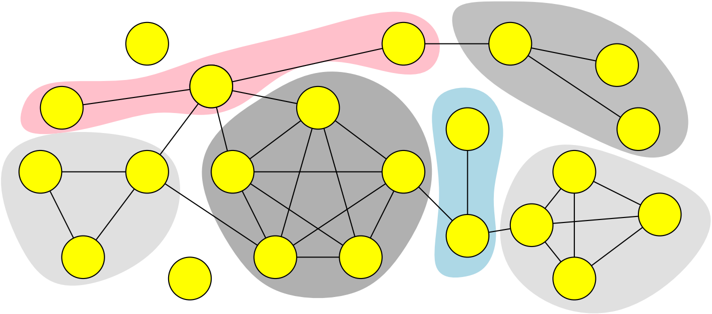
"How many edges must be inserted or deleted to arrive at disjoint cliques?"



editing set has size 5 + 12 = 17 (bad)

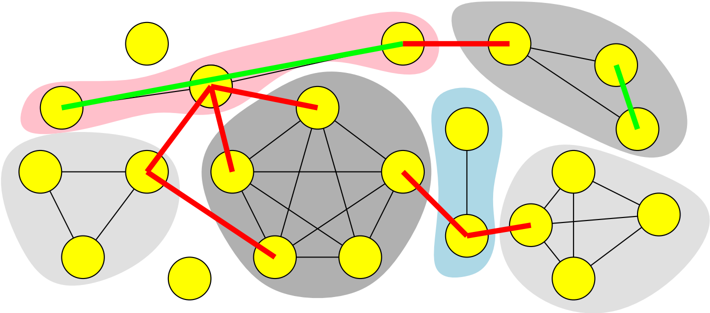
Cluster Editing Set

"How many edges must be inserted or deleted to arrive at disjoint cliques?"



Cluster Editing Set

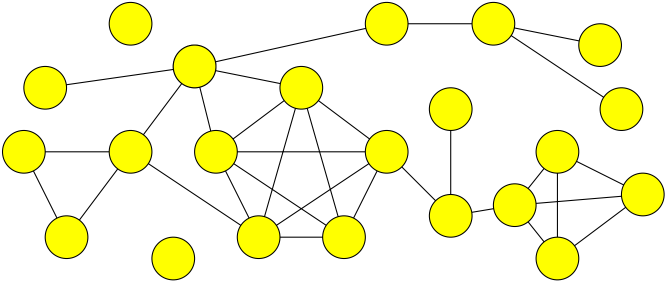
"How many edges must be inserted or deleted to arrive at disjoint cliques?"



editing set has size 3 + 7 = 10 (better)

Cluster Editing Set

"How many edges must be inserted or deleted to arrive at disjoint cliques?"

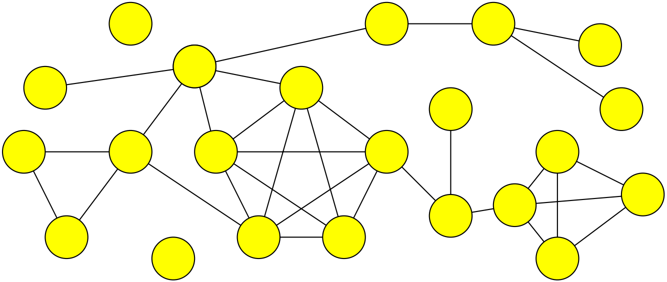


Task: find clustering with minimum cluster editing set

- NP-complete
- nicely approachable with FPT-techniques ($|V| > 1000$)
- popular in biology

Cluster Editing Set

"How many edges must be inserted or deleted to arrive at disjoint cliques?"



Task: find clustering with minimum cluster editing set

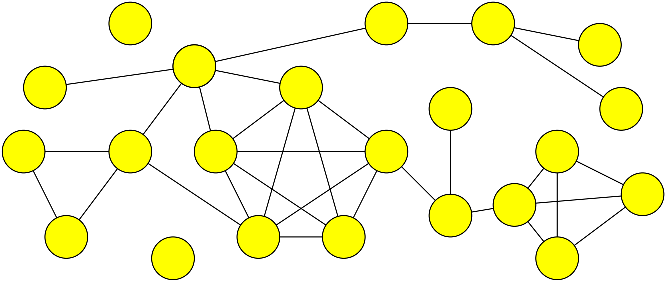
- NP-complete
- nicely approachable with FPT-techniques ($|V| > 1000$)
- popular in biology

[e.g., Böcker et al.: Exact algorithms for cluster editing: evaluation and experiments]

Cluster Editing Set

“How many edges must be inserted or deleted to arrive at disjoint cliques?”

exercise: cluster editing set of Zachary?



Task: find clustering with minimum cluster editing set

- NP-complete
- nicely approachable with FPT-techniques ($|V| > 1000$)
- popular in biology

[e.g., Böcker et al.: Exact algorithms for cluster editing: evaluation and experiments]

Formal Weaknesses?

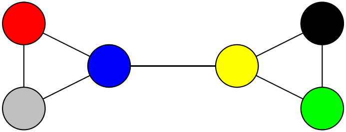
Karlsruher Institut für Technologie

exercise: do the proposed measures violate any desiderata?

Structure

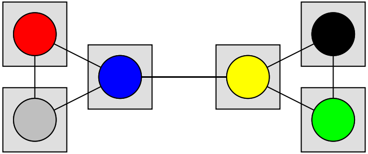
How to Cluster?

- *Optimization* of quality function:



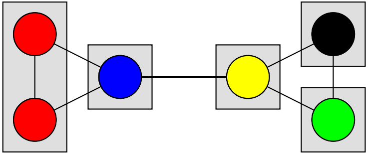
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons*



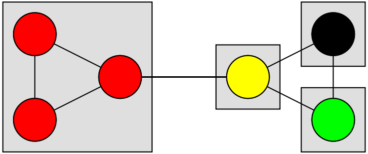
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters



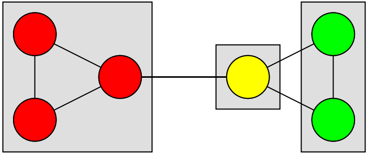
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters



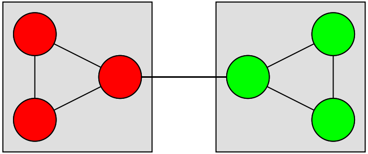
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters



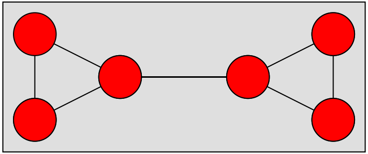
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters



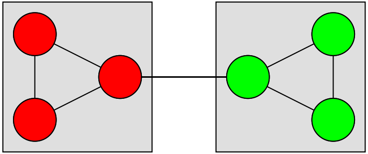
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters
  - Top-down: start with the *one-cluster*



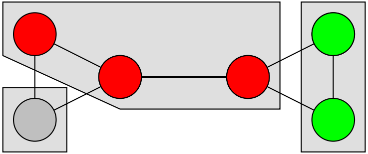
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters
  - Top-down: start with the *one-cluster* $\Rightarrow$ split clusters



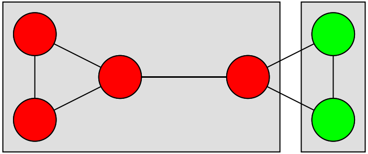
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters
  - Top-down: start with the *one-cluster* $\Rightarrow$ split clusters
  - Local Opt.: start with random clustering



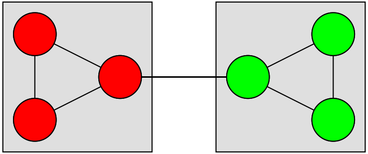
How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters
  - Top-down: start with the *one-cluster* $\Rightarrow$ split clusters
  - Local Opt.: start with random clustering $\Rightarrow$ migrate nodes



How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters
  - Top-down: start with the *one-cluster* ⇒ split clusters
  - Local Opt.: start with random clustering ⇒ migrate nodes



How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters
  - Top-down: start with the *one-cluster* $\Rightarrow$ split clusters
  - Local Opt.: start with random clustering $\Rightarrow$ migrate nodes
- Variants of *recursive min-cutting*

How to Cluster?

Karlsruher Institut für Technologie

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters
  - Top-down: start with the *one-cluster* ⇒ split clusters
  - Local Opt.: start with random clustering ⇒ migrate nodes
- Variants of *recursive min-cutting*
- *Percolation* of network by removal of highly central edges

How to Cluster?

Karlsruher Institut für Technologie

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters
  - Top-down: start with the *one-cluster* $\Rightarrow$ split clusters
  - Local Opt.: start with random clustering $\Rightarrow$ migrate nodes
- Variants of *recursive min-cutting*
- *Percolation* of network by removal of highly central edges
- *Spectral methods* using eigenanalysis of adjacency Laplacian

How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters
  - Top-down: start with the *one-cluster* ⇒ split clusters
  - Local Opt.: start with random clustering ⇒ migrate nodes
- Variants of *recursive min-cutting*
- *Percolation* of network by removal of highly central edges
- *Spectral methods* using eigenanalysis of adjacency Laplacian
- *Direct identification* of dense substructures

How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* $\Rightarrow$ merge clusters
  - Top-down: start with the *one-cluster* $\Rightarrow$ split clusters
  - Local Opt.: start with random clustering $\Rightarrow$ migrate nodes
- Variants of *recursive min-cutting*
- *Percolation* of network by removal of highly central edges
- *Spectral methods* using eigenanalysis of adjacency Laplacian
- *Direct identification* of dense substructures
- *Random walks*

How to Cluster?

- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters
  - Top-down: start with the *one-cluster* ⇒ split clusters
  - Local Opt.: start with random clustering ⇒ migrate nodes
- Variants of *recursive min-cutting*
- *Percolation* of network by removal of highly central edges
- *Spectral methods* using eigenanalysis of adjacency Laplacian
- *Direct identification* of dense substructures
- *Random walks*
- *Geometric approaches*

How to Cluster?

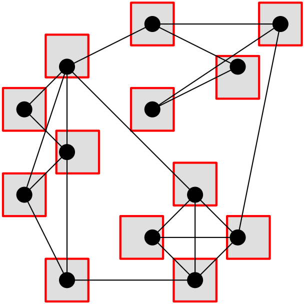
- *Optimization* of quality function:
  - Bottom-up: start with *singletons* ⇒ merge clusters
  - Top-down: start with the *one-cluster* ⇒ split clusters
  - Local Opt.: start with random clustering ⇒ migrate nodes
- Variants of *recursive min-cutting*
- *Percolation* of network by removal of highly central edges
- *Spectral methods* using eigenanalysis of adjacency Laplacian
- *Direct identification* of dense substructures
- *Random walks*
- *Geometric approaches*
- ...

Structure

Greedy Agglomeration / Merge

dendrogram

current clustering

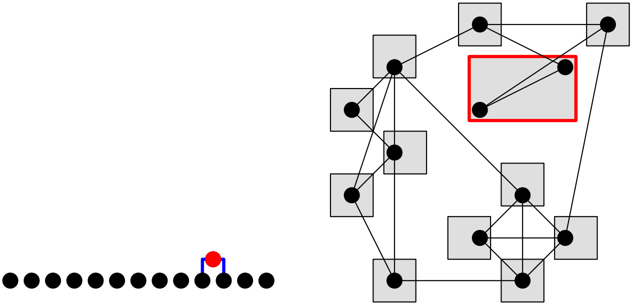


1. start: *singletons*

Greedy Agglomeration / Merge

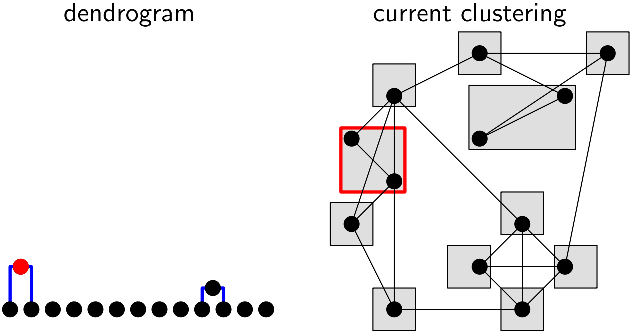
dendrogram

current clustering



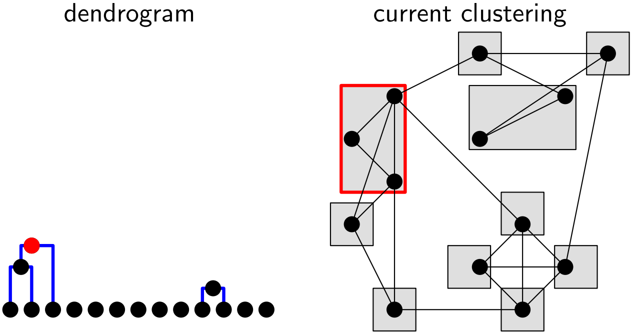
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

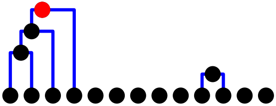
Greedy Agglomeration / Merge



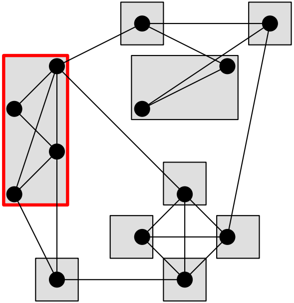
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge

dendrogram

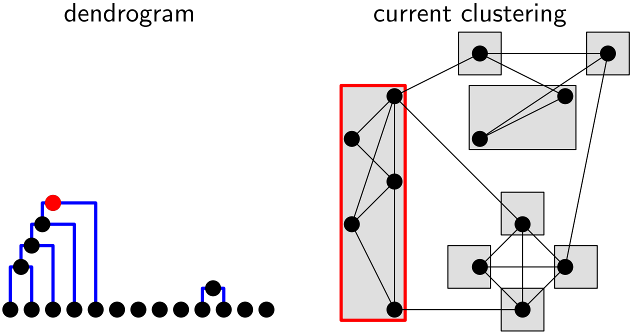


current clustering



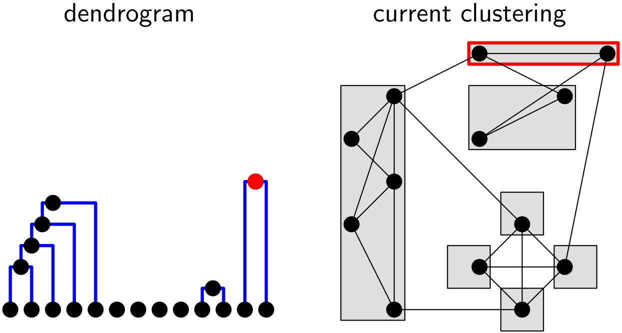
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



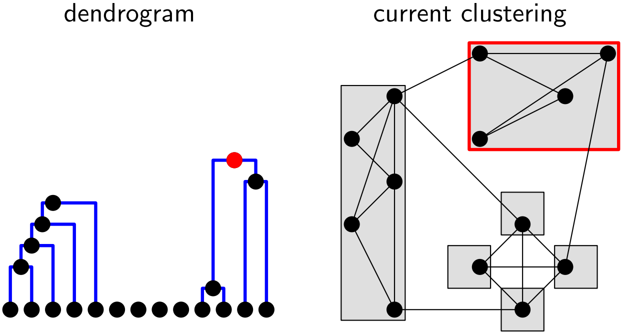
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



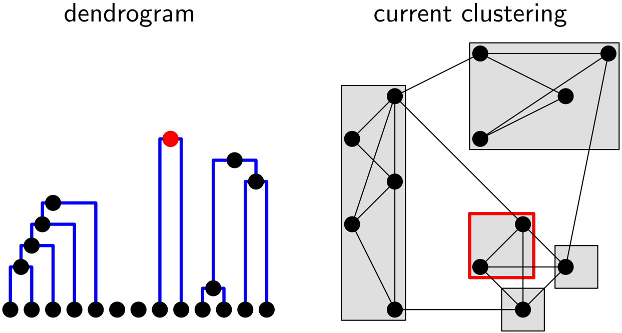
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



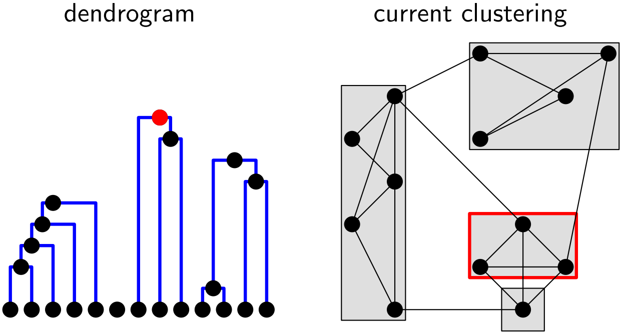
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

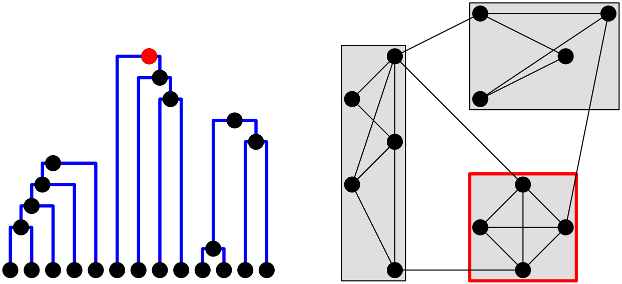
Greedy Agglomeration / Merge



1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

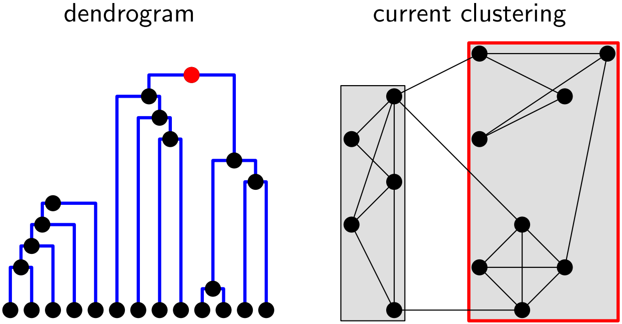
Greedy Agglomeration / Merge

dendrogram current clustering



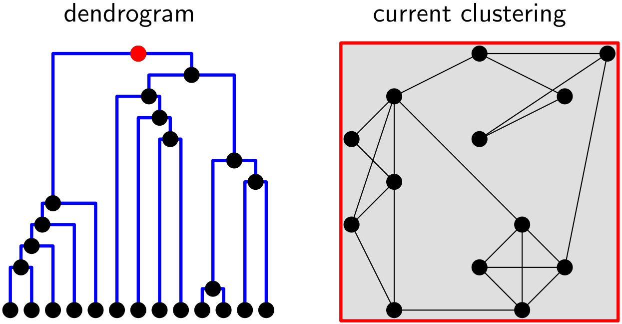
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



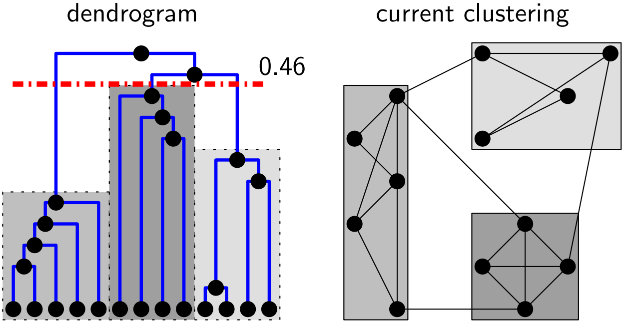
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



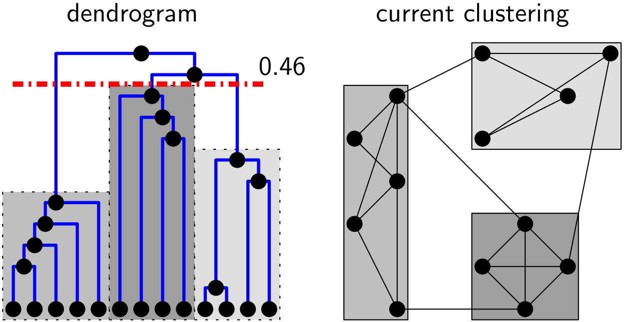
1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)

Greedy Agglomeration / Merge



1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)
3. result: best intermediate clustering

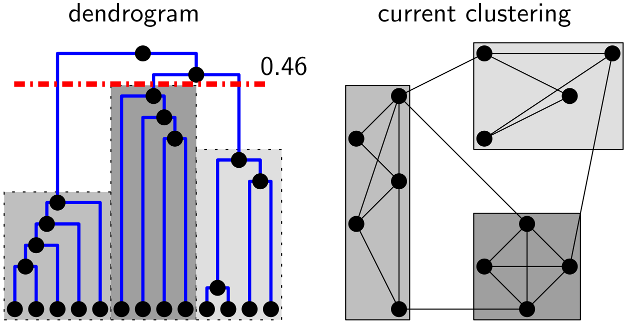
Greedy Agglomeration / Merge



1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)
3. result: best intermediate clustering

modularity: $O(n^2 \log n)$ oder $O(md \log n)$; often close to $O(n \log^2 n)$

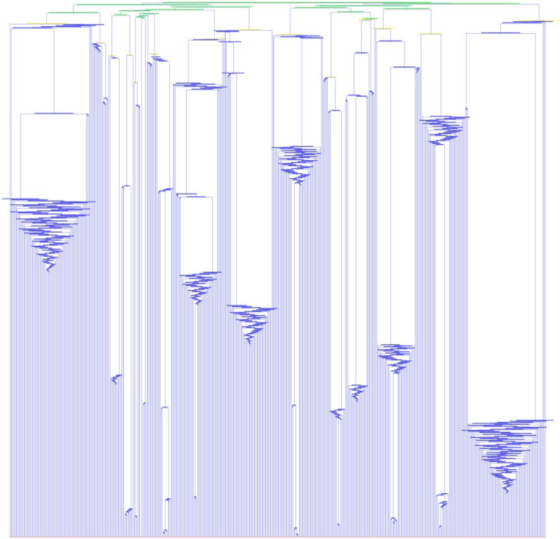
Greedy Agglomeration / Merge



1. start: *singletons*
2. iterative *agglomerations*, yielding highest gain in quailty (or least decrease)
3. result: best intermediate clustering

modularity: $O(n^2 \log n)$ oder $O(md \log n)$; often close to $O(n \log^2 n)$
other objective functions, like bicriterial formulations?

Larger Dendrogram



collaboration network, $|V| \approx 1000$

Influence of Measures on Algorithm: Coarseness

Rough Intuition

Influence of Measures on Algorithm: Coarseness

Rough Intuition

Question

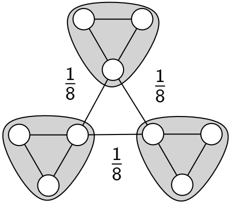
Without constraints, is there always a merge that improves the objective function?

(Un-)Boundedness

Definition

An objective function measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance
is bounded

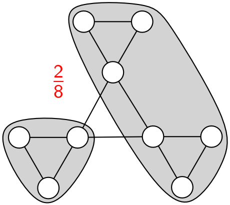


(Un-)Boundedness

Definition

An objective function measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance is bounded

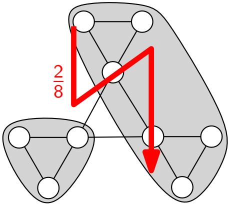


(Un-)Boundedness

Definition

An objective function measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance
is bounded

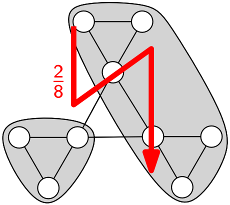


(Un-)Boundedness

Definition

An objective function measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance is bounded



e.g., modularity is bounded

Inter-cluster Sparsity: Degrees of Freedom (Rep.)

Set of cuts

- isolated (one for each cluster)
- pairwise (one for each pair of clusters)
- global (k-way cut)

Measures

- number of cut-edges
- density
- conductance
- expansion

Combinations

- average sparsity
- minimum sparsity

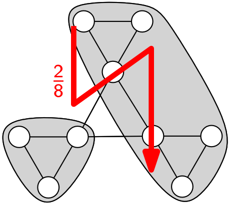
$\Rightarrow$ **14 (reasonable) inter-cluster sparsity measures**

(Un-)Boundedness

Definition

An inter-cluster sparsity measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance is bounded



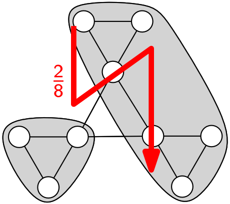
e.g., modularity is bounded

(Un-)Boundedness

Definition

An inter-cluster sparsity measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance is bounded



bounded

- mpxc
- mpxe
- apxd
- apxc
- aixd

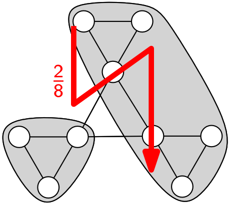
e.g., modularity is bounded

(Un-)Boundedness

Definition

An inter-cluster sparsity measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance is bounded



e.g., modularity is bounded

bounded

- mpxc
- mpxe
- apxd
- apxc
- aixd

unbounded

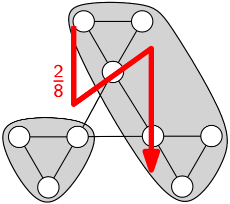
- nxe
- gxd
- mixc
- mixd
- mixe
- aixc
- aixe
- mixd
- mpxd

(Un-)Boundedness

Definition

An inter-cluster sparsity measure f is *unbounded* if for any clustering $\mathcal{C}$ with $|\mathcal{C}| > 1$ there exists a merge that does not deteriorate f.

Max. pw. inter-cluster conductance is bounded



e.g., modularity is bounded

bounded

- mpxc
- mpxe
- apxd
- apxc
- aixd

unbounded

- nxe
- gxd
- mixc
- mixd
- mixe
- aixc
- aixe
- mixd
- mpxd

exercise: reachability, proofs for (un)boundedness

Greedy Merge and Modularity

Karlsruher Institut für Technologie

- greedy maximization does not approximate $\begin{array}{l} \texttt{nan} \\ > 2 \end{array}$ $\begin{array}{l} \text{worst tie-breaking} \\ \text{best tie-breaking} \end{array}$

Greedy Merge and Modularity

- greedy maximization does not approximate $\frac{\texttt{worst tie-breaking}}{\texttt{best tie-breaking}}$: nan worst tie-breaking; > 2 best tie-breaking
- modularity has a single peak during agglomeration **(exercise)**

Greedy Merge and Modularity

- greedy maximization does not approximate $\begin{array}{ll} \texttt{nan} & \text{worst tie-breaking} \\ > 2 & \text{best tie-breaking} \end{array}$
- modularity has a single peak during agglomeration **(exercise)**
- simple to implement and rather successful

Greedy Merge and Modularity

- greedy maximization does not approximate $\begin{array}{l} \texttt{nan} \\ > 2 \end{array}$ $\dfrac{\text{worst tie-breaking}}{\text{best tie-breaking}}$
- modularity has a single peak during agglomeration **(exercise)**
- simple to implement and rather successful
- inefficient for large graphs

Greedy Merge and Modularity

- greedy maximization does not approximate $\frac{\text{worst tie-breaking}}{\text{best tie-breaking}} \text{nan} > 2$
- modularity has a single peak during agglomeration **(exercise)**
- simple to implement and rather successful
- inefficient for large graphs
- only known kernelization: degree-1 vertices **(exercise)**

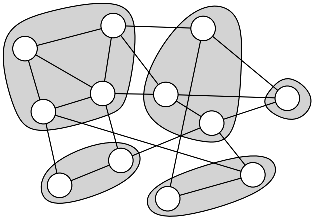
Greedy Merge and Modularity

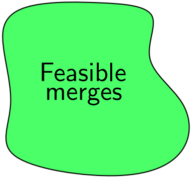
- greedy maximization does not approximate $\frac{\text{worst tie-breaking}}{\text{best tie-breaking}} \quad \texttt{nan} > 2$
- modularity has a single peak during agglomeration **(exercise)**
- simple to implement and rather successful
- inefficient for large graphs
- only known kernelization: degree-1 vertices **(exercise)**

Data structure and maintained information for efficient greedy agglomeration?

Influence of Measures on Efficiency

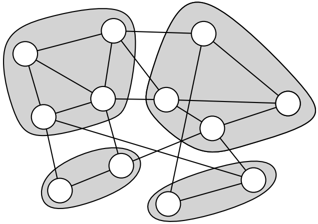
In a bicriterial setting with constraints:

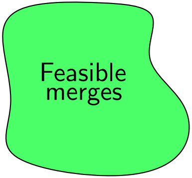




Influence of Measures on Efficiency

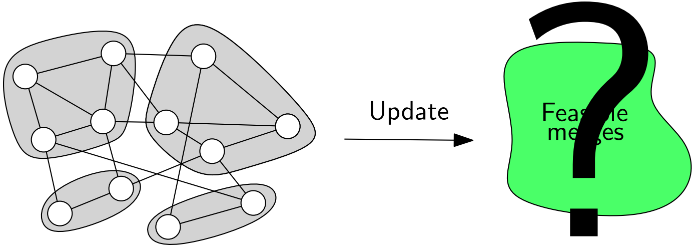
In a bicriterial setting with constraints:





Influence of Measures on Efficiency

In a bicriterial setting with constraints:

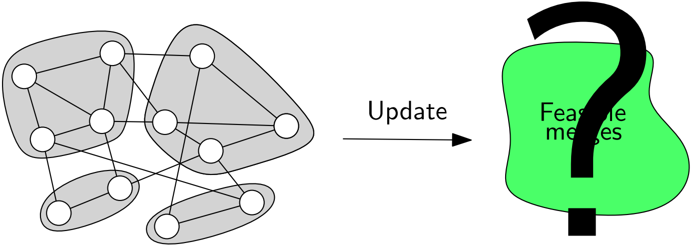


Question

Does feasibility of a merge only depend on involved clusters?

Influence of Measures on Efficiency

In a bicriterial setting with constraints:



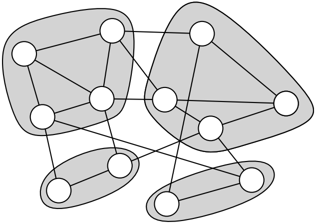
Question

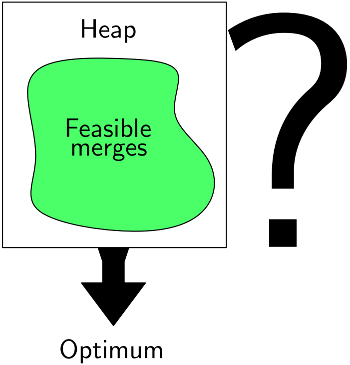
Does feasibility of a merge only depend on involved clusters?

$\Rightarrow$ **Context freeness of a constraint**

Influence of Measures on Efficiency

In a general setting:



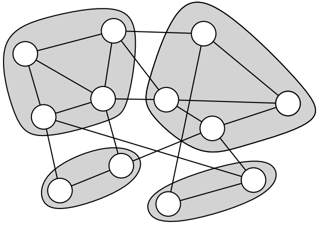


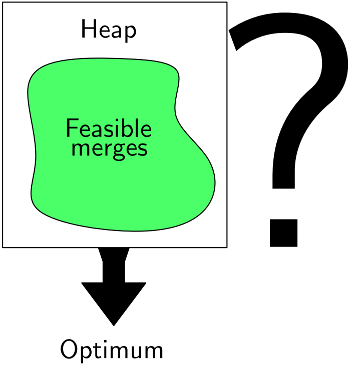
Question

Given *context freeness*, can the set of feasible merges be efficiently maintained in a heap?

Influence of Measures on Efficiency

In a general setting:



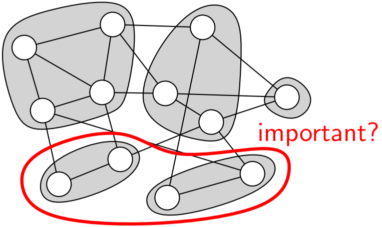


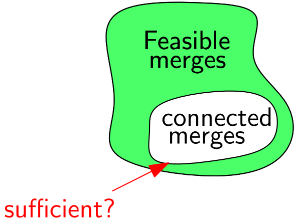
Question

Given *context freeness*, can the set of feasible merges be efficiently maintained in a heap?

⇒ **Locality of an objective function**

Influence of Measures on Efficiency

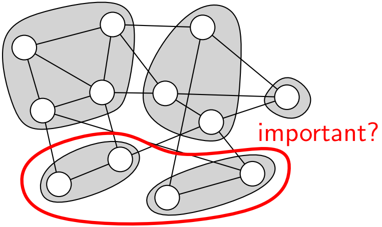


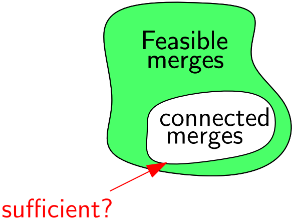


Question

Do we have to consider pairs of unconnected clusters?

Influence of Measures on Efficiency





Question

Do we have to consider pairs of unconnected clusters?

$\Rightarrow$ **Connectedness of an objective function**

Influence of Measures on Efficiency

(Given the necessary data can efficiently be maintained:)

Context insensitivity + Locality = $O(n^2 \log n)$ running time

Influence of Measures on Efficiency

(Given the necessary data can efficiently be maintained:)

Context insensitivity + Locality = $O(n^2 \log n)$ running time

Context insensitivity + Locality + Connectedness = $O(md \log n)$ running time & linear space

Influence of Measures on Efficiency

(Given the necessary data can efficiently be maintained:)

Context insensitivity + Locality = $O(n^2 \log n)$ running time

Context insensitivity + Locality + Connectedness = $O(md \log n)$ running time & linear space

For more information on these topics:
[Schumm et al.: Density-constrained graph clustering, 2011]

Context Freeness

Definition

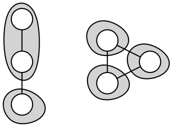
A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

Context Freeness

Definition

A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

- E.g., global intra-cluster density is not context free



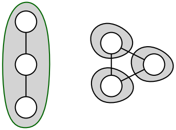
Constraint: $\frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|} = \frac{1}{1} \geq 0.7$

Context Freeness

Definition

A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

- E.g., global intra-cluster density is not context free



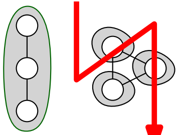
$$\text{Constraint: } \frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|} = \frac{2}{3} < 0.7$$

Context Freeness

Definition

A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

- E.g., global intra-cluster density is not context free



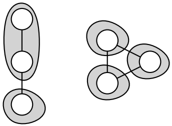
$$\text{Constraint: } \frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|} = \frac{2}{3} < 0.7$$

Context Freeness

Definition

A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

- E.g., global intra-cluster density is not context free



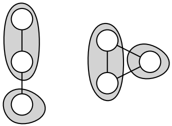
$$\text{Constraint: } \frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|} = \frac{1}{1} \geq 0.7$$

Context Freeness

Definition

A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

- E.g., global intra-cluster density is not context free



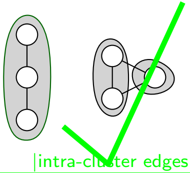
Constraint: $\frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|} = \frac{2}{2} \geq 0.7$

Context Freeness

Definition

A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

- E.g., global intra-cluster density is not context free



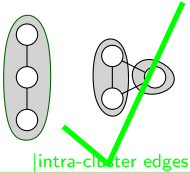
$$\text{Constraint: } \frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|} = \frac{3}{4} \geq 0.7$$

Context Freeness

Definition

A constraint is *context free*, if the feasibility of a merge does not depend on the remainder of the clustering.

- E.g., global intra-cluster density is not context free



$$\text{Constraint: } \frac{|\text{intra-cluster edges}|}{|\text{possible intra-cluster edges}|} = \frac{3}{4} \geq 0.7$$

- E.g., minimum intra-cluster density is context free

Locality: Intuition

Locality is a property of an objective functions

Example: Maximum isolated inter-cluster conductance

First approach: Use gain in inter-cluster sparsity as key

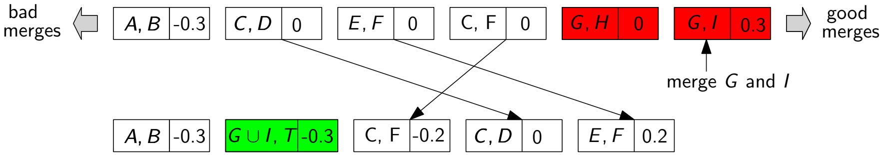
bad merges ⇦ | A, B | -0.3 | | C, D | 0 | | E, F | 0 | | C, F | 0 | | G, H | 0 | | G, I | 0.3 | ⇨ good merges

Locality: Intuition

Locality is a property of an objective functions

Example: Maximum isolated inter-cluster conductance

First approach: Use gain in inter-cluster sparsity as key

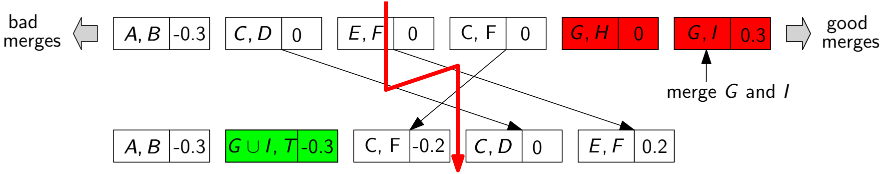


Locality: Intuition

Locality is a property of an objective functions

Example: Maximum isolated inter-cluster conductance

First approach: Use gain in inter-cluster sparsity as key

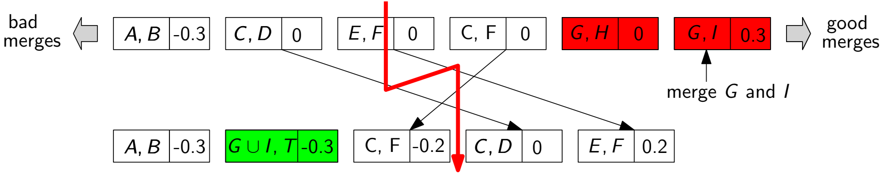


Locality: Intuition

Locality is a property of an objective functions

Example: Maximum isolated inter-cluster conductance

First approach: Use gain in inter-cluster sparsity as key



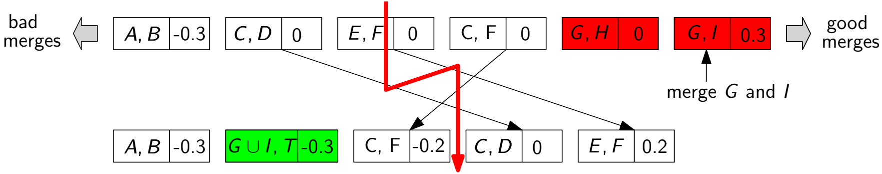
Clever tie-breaking possible?

Locality: Intuition

Locality is a property of an objective functions

Example: Maximum isolated inter-cluster conductance

First approach: Use gain in inter-cluster sparsity as key



Clever tie-breaking possible?

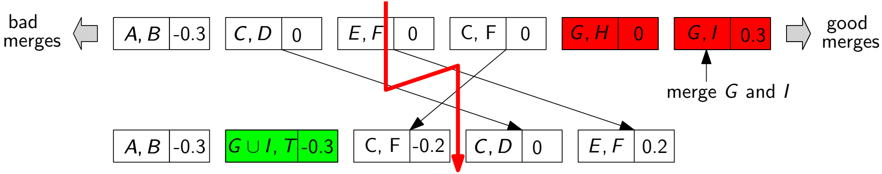
Needed: Suitable order that does not change if unrelated clusters merge

Locality: Intuition

Locality is a property of an objective functions

Example: Maximum isolated inter-cluster conductance

First approach: Use gain in inter-cluster sparsity as key



Clever tie-breaking possible?

Needed: Suitable order that does not change if unrelated clusters merge

Existence of such an order $\approx$ Locality of the inter-cluster measure

Example: Max. Isolated Inter-cluster Conductance

Current sequence of conductance of all clusters (sorted)

A	0.5	B	0.4	C	0.3	D	0.3	E	0.1

Example: Max. Isolated Inter-cluster Conductance

Current sequence of conductance of all clusters (sorted)

A	0.5	B	0.4	C	0.3	D	0.3	E	0.1

Sequence if A and B are merged

$A \cup B$	0.45	C	0.3	D	0.3	E	0.1

Example: Max. Isolated Inter-cluster Conductance

Current sequence of conductance of all clusters (sorted)

A	0.5	B	0.4	C	0.3	D	0.3	E	0.1

Sequence if A and B are merged

$A \cup B$	0.45	C	0.3	D	0.3	E	0.1

Sequence if A and D are merged

$A \cup D$	0.45	B	0.4	C	0.3	E	0.1

Example: Max. Isolated Inter-cluster Conductance

Current sequence of conductance of all clusters (sorted)

A	0.5	B	0.4	C	0.3	D	0.3	E	0.1

Sequence if A and B are merged

$A \cup B$	0.45	C	0.3	D	0.3	E	0.1

Sequence if A and D are merged

$A \cup D$	0.45	B	0.4	C	0.3	E	0.1

compare lexicographically:

Merging A and B is better!

Example: Max. Isolated Inter-cluster Conductance

Current sequence of conductance of all clusters (sorted)

A	0.5	B	0.4	C	0.3	D	0.3	E	0.1

Sequence if A and B are merged

$A \cup B$	0.45	C	0.3	D	0.3	E	0.1

Sequence if A and D are merged

$A \cup D$	0.45	B	0.4	C	0.3	E	0.1

compare lexicographically:

Merging A and B is better!

- Ordering merges lexicographically is stable
- Two merges can be compared in constant time by comparing keys consisting of three numbers

Example: Max. Isolated Inter-cluster Conductance

Current sequence of conductance of all clusters (sorted)

A	0.5	B	0.4	C	0.3	D	0.3	E	0.1

Sequence if A and B are merged

$A \cup B$	0.45	C	0.3	D	0.3	E	0.1

compare lexicographically:

Merging A and B is better!

Sequence if A and D are merged

$A \cup D$	0.45	B	0.4	C	0.3	E	0.1

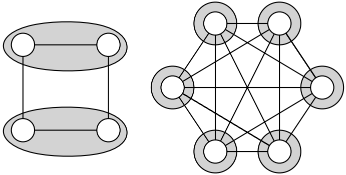
- Ordering merges lexicographically is stable
- Two merges can be compared in constant time by comparing keys consisting of three numbers

$\Rightarrow$ **Maximum isolated inter-cluster conductance is local**

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

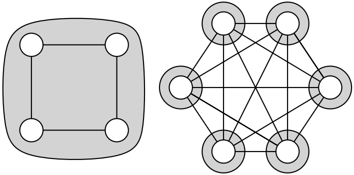


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{17}{43}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

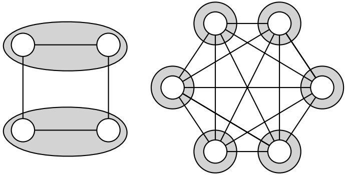


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{15}{39}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

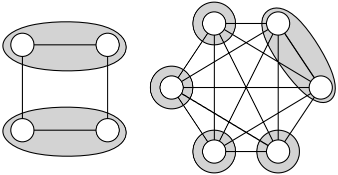


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{17}{43}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

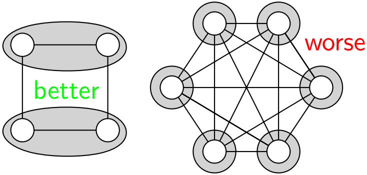


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{16}{42}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

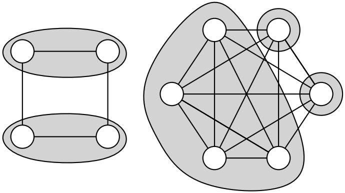


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{17}{43}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

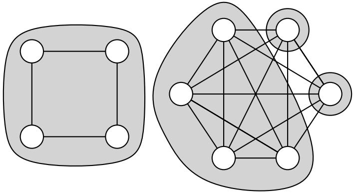


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{9}{37}$$

Locality: Results

Does such an order exist for all objective functions?

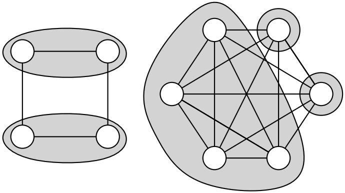
global inter-cluster density is not local



$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{7}{33}$$

Does such an order exist for all objective functions?

global inter-cluster density is not local

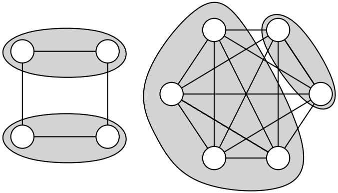


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{9}{37}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

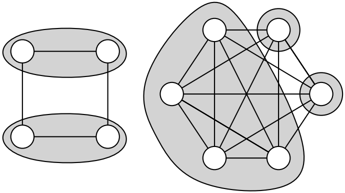


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{8}{36}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

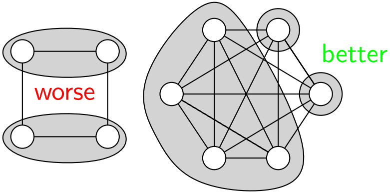


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{9}{37}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local

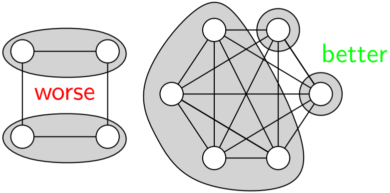


$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{8}{36}$$

Locality: Results

Does such an order exist for all objective functions?

global inter-cluster density is not local



$$\frac{|\text{inter-cluster edges}|}{|\text{possible inter-cluster edges}|} = \frac{8}{36}$$

local

- mixd
- mixc
- mixe
- aixd
- aixc
- aixe
- nxe

not local

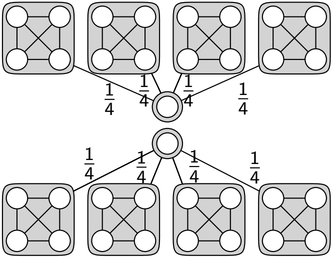
- mpxd
- apxd
- mpxc
- mpxe
- gxd
- apxe
- apxc

Disconnectedness

Definition

An objective function *f* is *connected* if merging unconnected clusters is never the best option with respect to *f*.

max. pw. inter-cluster conductance is not connected

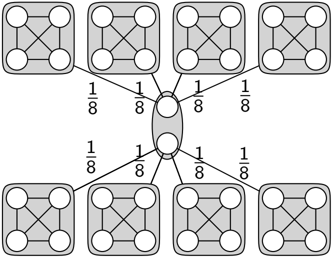


Disconnectedness

Definition

An objective function f is *connected* if merging unconnected clusters is never the best option with respect to f.

max. pw. inter-cluster conductance
is not connected

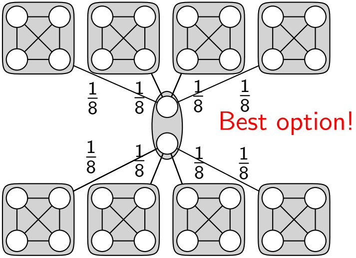


Disconnectedness

Definition

An objective function f is *connected* if merging unconnected clusters is never the best option with respect to f.

max. pw. inter-cluster conductance is not connected

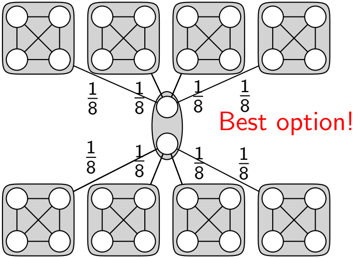


Disconnectedness

Definition

An objective function f is *connected* if merging unconnected clusters is never the best option with respect to f.

max. pw. inter-cluster conductance is not connected



connected

- nxe

unconnected

- gxd
- mixc
- mixd
- mixe
- aixc
- aixe
- mixd
- mpxd
- mpxc
- mpxe
- apxd
- apxc
- aixd

A Matrix for Maintaining Merges

Update of *merge-matrix*, when merging clusters C_i and C_j:

		j		
		$\vdots$		
i	$\cdots$	$\Delta S_{i,j}$	$\cdots$	
		$\vdots$		

A Matrix for Maintaining Merges

Update of *merge-matrix*, when merging clusters C_i and C_j:

- Additive update of gain, easy to handle (e.g., modularity):
  $\Delta S_{(ij),k} = \Delta S_{i,k} + \Delta S_{j,k}$

		j		
		$\vdots$		
i	$\cdots$	$\Delta S_{i,j}$	$\cdots$	
		$\vdots$		

A Matrix for Maintaining Merges

Update of *merge-matrix*, when merging clusters C_i and C_j:

- Additive update of gain, easy to handle (e.g., modularity):
  $\Delta S_{(ij),k} = \Delta S_{i,k} + \Delta S_{j,k}$
  Unaffected entries $\Delta S_{m,n}$ unchanged $\Rightarrow$ log-linear effort (heap)

		j		
		$\vdots$		
i	$\cdots$	$\Delta S_{i,j}$	$\cdots$	
		$\vdots$		

A Matrix for Maintaining Merges

Update of *merge-matrix*, when merging clusters C_i and C_j:

- Additive update of gain, easy to handle (e.g., modularity):
  $\Delta S_{(ij),k} = \Delta S_{i,k} + \Delta S_{j,k}$
  Unaffected entries $\Delta S_{m,n}$ unchanged $\Rightarrow$ log-linear effort (heap)
- Divisive update, more complicated (e.g., global inter-cluster density):
  $\Delta S_{m,n} = \frac{A+\Delta A_{m,n}}{B+\Delta B_{m,n}} - \frac{A}{B}$

		j		
		$\vdots$		
i	$\cdots$	$\Delta S_{i,j}$	$\cdots$	
		$\vdots$		

A Matrix for Maintaining Merges

Update of *merge-matrix*, when merging clusters C_i and C_j:

- Additive update of gain, easy to handle (e.g., modularity):
  $\Delta S_{(ij),k} = \Delta S_{i,k} + \Delta S_{j,k}$
  Unaffected entries $\Delta S_{m,n}$ unchanged $\Rightarrow$ log-linear effort (heap)
- Divisive update, more complicated (e.g., global inter-cluster density):
  $\Delta S_{m,n} = \frac{A+\Delta A_{m,n}}{B+\Delta B_{m,n}} - \frac{A}{B}$
  Easy to compute,

$$
\begin{array}{c|c|c|c|c}
 & & j & & \\
\hline
 & & \vdots & & \\
\hline
i & \cdots & \Delta S_{i,j} & \cdots & \\
\hline
 & & \vdots & & \\
\hline
 & & & &
\end{array}
$$

A Matrix for Maintaining Merges

Update of *merge-matrix*, when merging clusters C_i and C_j:

- Additive update of gain, easy to handle (e.g., modularity):
  $\Delta S_{(ij),k} = \Delta S_{i,k} + \Delta S_{j,k}$
  Unaffected entries $\Delta S_{m,n}$ unchanged $\Rightarrow$ log-linear effort (heap)
- Divisive update, more complicated (e.g., global inter-cluster density):
  $\Delta S_{m,n} = \frac{A+\Delta A_{m,n}}{B+\Delta B_{m,n}} - \frac{A}{B}$
  Easy to compute, but **whole** matrix needs update $\Rightarrow \Omega(n^2)$

		j		
		$\vdots$		
i	$\cdots$	$\Delta S_{i,j}$	$\cdots$	
		$\vdots$		

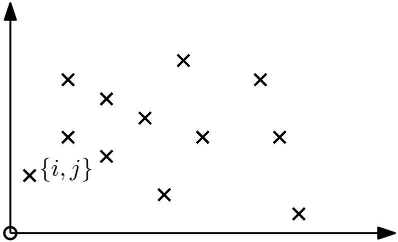
Quick Divisive Merge

Key idea: geometric (2d) data structure for matrix

Quick Divisive Merge

Key idea: geometric (2d) data structure for matrix

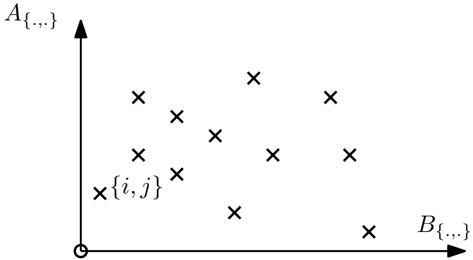
- One data point for each pair of clusters $\{C_i, C_j\}$



Quick Divisive Merge

Key idea: geometric (2d) data structure for matrix

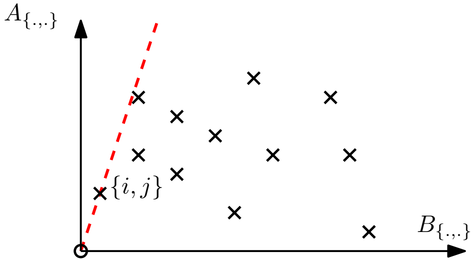
- One data point for each pair of clusters $\{C_i, C_j\}$
- y-coord.: $\Delta A_{i,j}$, x-coord.: $\Delta B_{i,j}$



Quick Divisive Merge

Key idea: geometric (2d) data structure for matrix

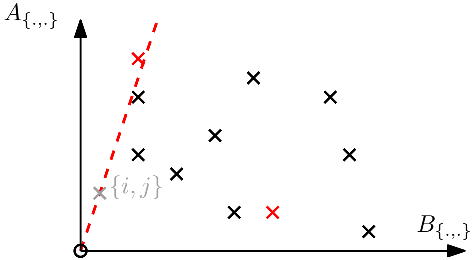
- One data point for each pair of clusters $\{C_i, C_j\}$
- y-coord.: $\Delta A_{i,j}$, x-coord.: $\Delta B_{i,j}$
- Find best merge with *tangent query* from origin in $O(\log(\#\text{ points})) = O(\log(|V|^2))$ (Brodal & Jacob [02])



Quick Divisive Merge

Key idea: geometric (2d) data structure for matrix

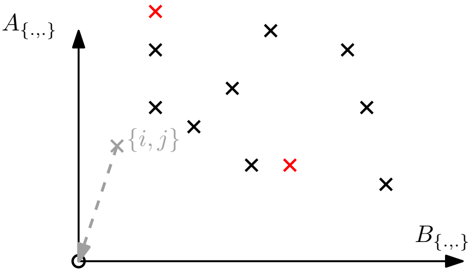
- One data point for each pair of clusters $\{C_i, C_j\}$
- y-coord.: $\Delta A_{i,j}$, x-coord.: $\Delta B_{i,j}$
- Find best merge with *tangent query* from origin in $O(\log(\#\text{ points})) = O(\log(|V|^2))$ (Brodal & Jacob [02])
- $O(n)$ *real* updates (as in subtractive case)



Quick Divisive Merge

Key idea: geometric (2d) data structure for matrix

- One data point for each pair of clusters $\{C_i, C_j\}$
- y-coord.: $\Delta A_{i,j}$, x-coord.: $\Delta B_{i,j}$
- Find best merge with *tangent query* from origin in $O(\log(\#\text{ points})) = O(\log(|V|^2))$ (Brodal & Jacob [02])
- $O(n)$ *real* updates (as in subtractive case)
- Update of nominator and denominator by *shifting origin*

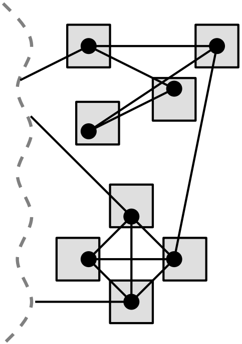


Structure

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

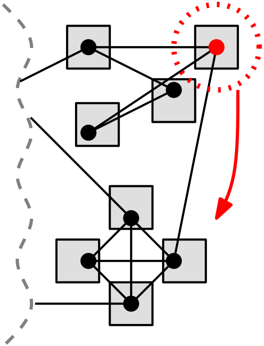


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

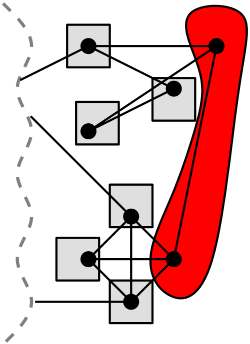


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

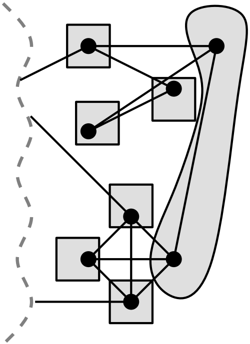


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

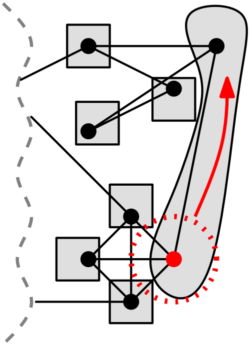


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

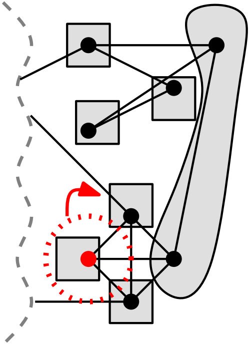


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

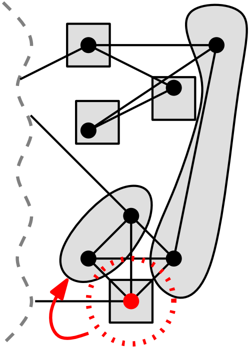


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

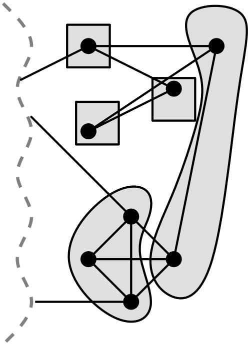


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

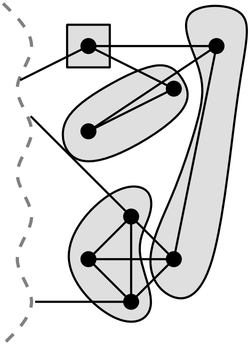


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

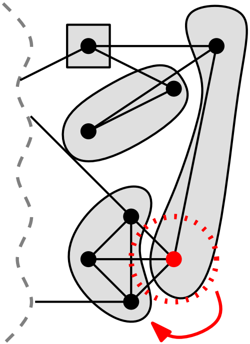


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions

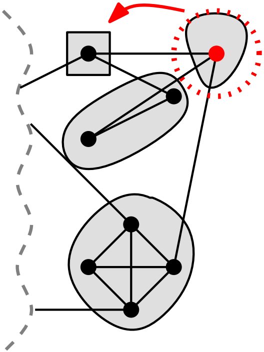


[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning

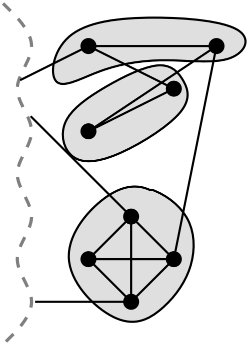
- locally greedy
- node shifts
- hierarchical contractions



[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

A common technique in graph partitioning



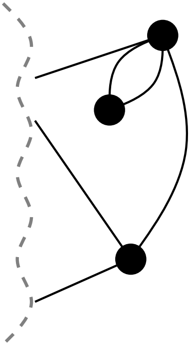
- locally greedy
- node shifts
- hierarchical contractions

[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

Local Movement

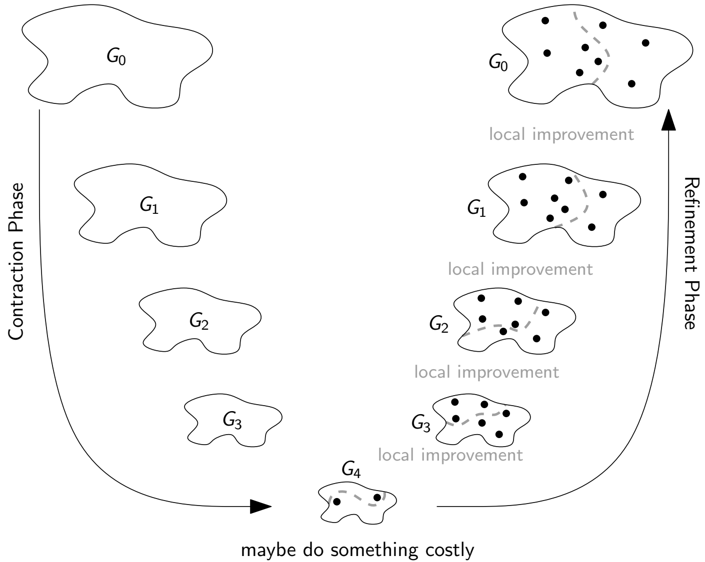
A common technique in graph partitioning

- locally greedy
- node shifts
- hierarchical contractions



[e.g., for modularity: Blondel et al.: Fast unfolding of communities in large networks, 2008]

The Multilevel Approach



Local vs. Global Greedy

pro local movement

- local qualitatively superior
- local quicker in practice
- better avoidance of local minima, larger search space
- refinement adds a few more percent to quality of local approach

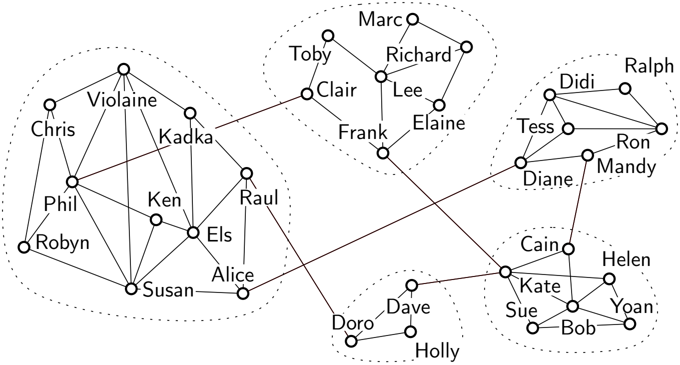
pro global agglomeration

- global easier to implement (no contraction, updates)
- runtime guarantees stronger for global
- global yields continuous hierarchy

exercise: collect edges to neighbors

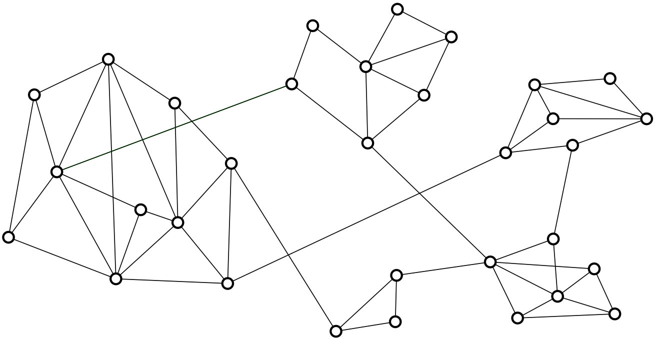
Structure

Min-Cut Tree Clustering



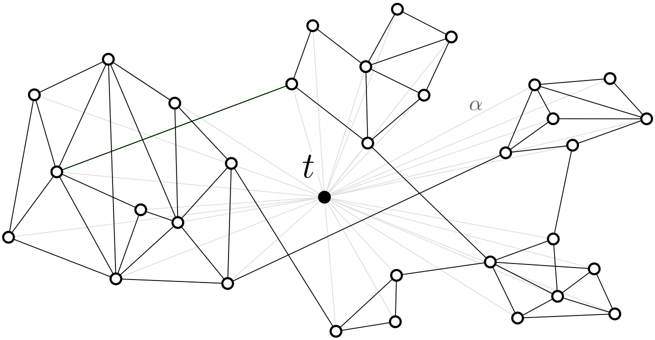
1. original graph

Min-Cut Tree Clustering



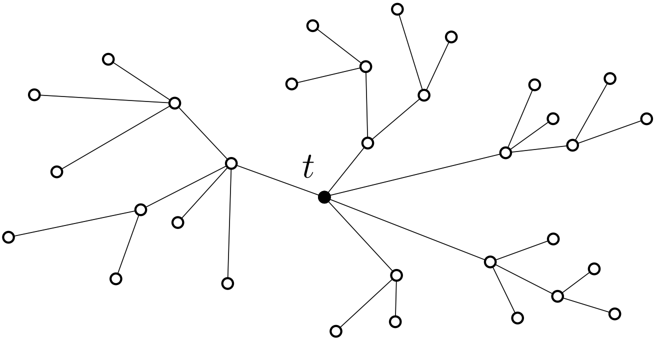
1. original graph

Min-Cut Tree Clustering



1. original graph
2. star-center t, α

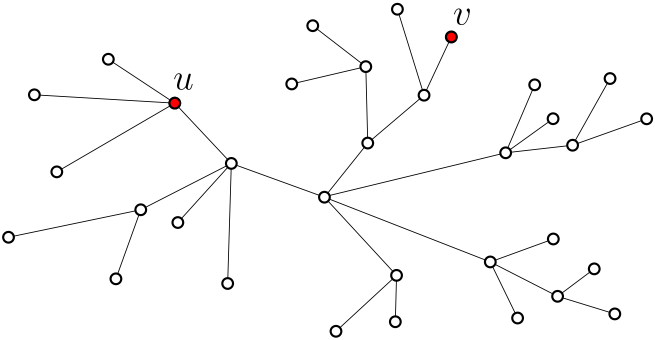
Min-Cut Tree Clustering



1. original graph
2. star-center t, α
3. *min-cut tree*

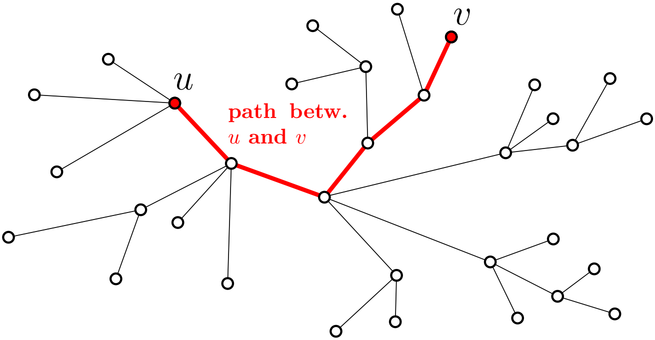
- coined in [Gomory and Hu '61]
- simplified in [Gusfield '90]
- construction via $(n-1)$ max-flows
  (variants e.g. $\widetilde{O}(mn)$ for unweighted)

Min-Cut Tree Clustering



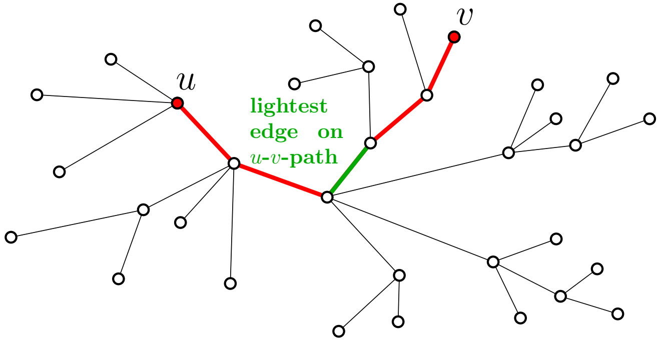
1. original graph
2. star-center t, α
3. *min-cut tree*

Min-Cut Tree Clustering



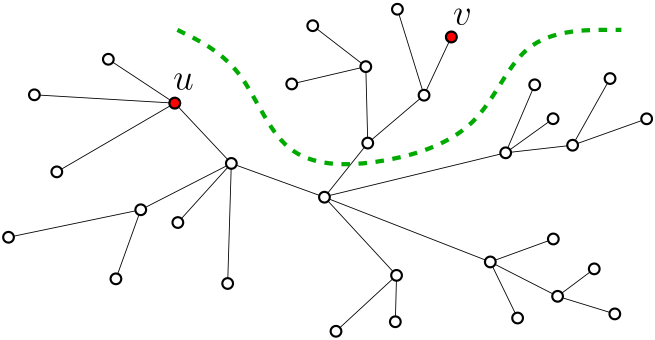
1. original graph
2. star-center t, α
3. *min-cut tree*

Min-Cut Tree Clustering



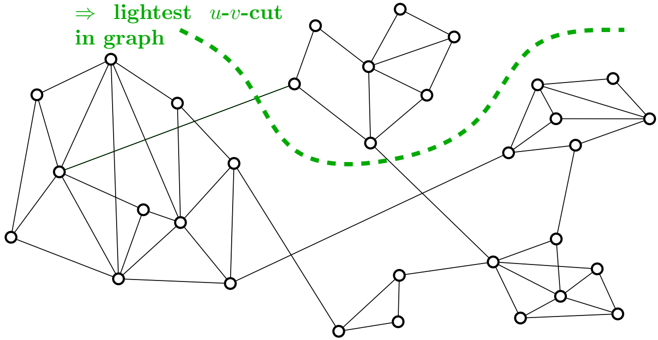
1. original graph
2. star-center *t*, α
3. *min-cut tree*

Min-Cut Tree Clustering



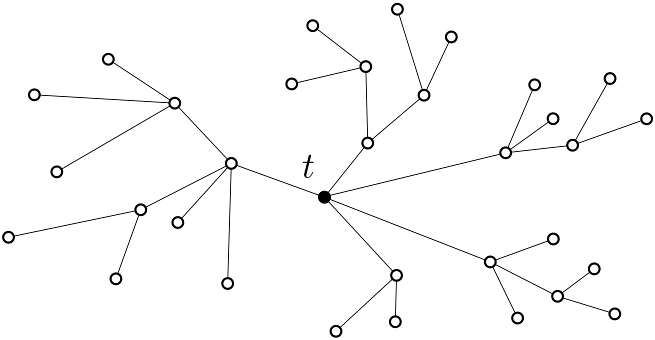
1. original graph
2. star-center *t*, α
3. *min-cut tree*

Min-Cut Tree Clustering



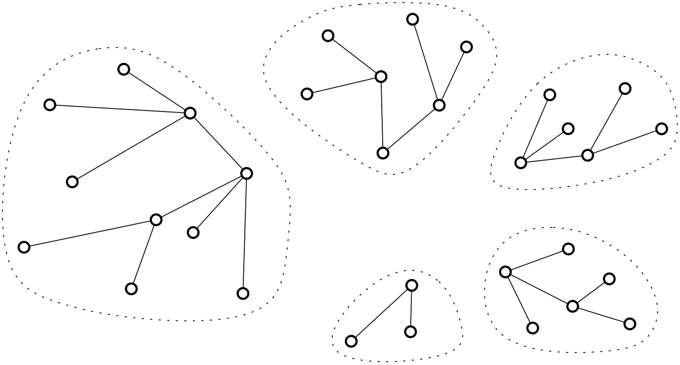
1. original graph
2. star-center t, α
3. *min-cut tree*

Min-Cut Tree Clustering



1. original graph
2. star-center t, α
3. *min-cut tree*

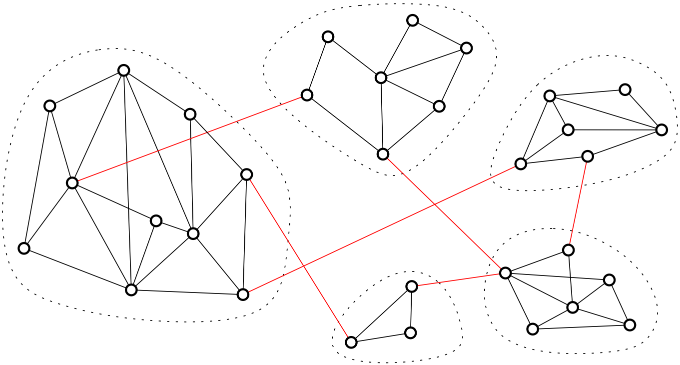
Min-Cut Tree Clustering



1. original graph
2. star-center t, α
3. *min-cut tree*
4. delete center $\Rightarrow$ clustering

Min-Cut Tree Clustering

Karlsruher Institut für Technologie

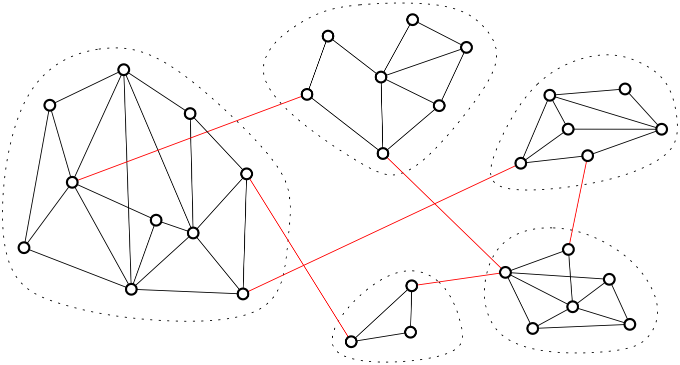


1. original graph
2. star-center t, α
3. *min-cut tree*
4. delete center $\Rightarrow$ clustering

quality guarantee: [Flake et al. '04]

intra-cluster expansion $\geq \alpha \geq$ *inter-cluster expansion**

Min-Cut Tree Clustering

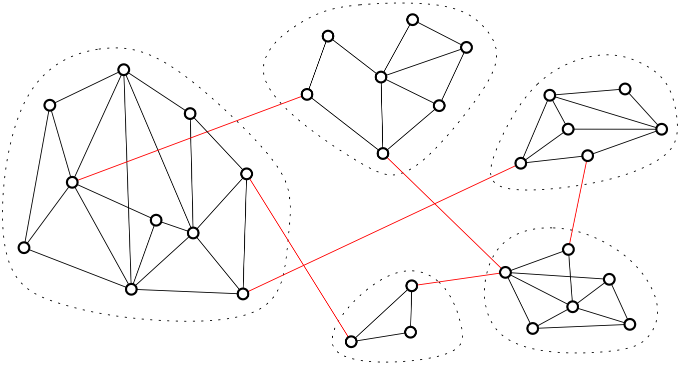


1. original graph
2. star-center t, α
3. *min-cut tree*
4. delete center $\Rightarrow$ clustering

quality guarantee: [Flake et al. '04]

$$\frac{\omega(E(P,C\setminus P))}{\min\{|P|,|C\setminus P|\}} \geq \alpha \geq \frac{\omega(E(C,V\setminus C))}{|V\setminus C|}$$

Min-Cut Tree Clustering



1. original graph
2. star-center t, α
3. *min-cut tree*
4. delete center $\Rightarrow$ clustering

quality guarantee: [Flake et al. '04]

$$\frac{\omega(E(P,C\setminus P))}{\min\{|P|,|C\setminus P|\}} \geq \alpha \geq \frac{\omega(E(C,V\setminus C))}{|V\setminus C|}$$

exercise: prove bounds!

Discussion: Min-Cut Tree Clustering

KIT
Karlsruher Institut für Technologie

- a vertex v that served to identify a cluster-defining v-t-cut is called the *representative* of the respective cluster

- scaling α yields a nested hierarchy of clusterings
- hierarchy has depth $\leq n - 1$

Discussion: Min-Cut Tree Clustering

- a vertex v that served to identify a cluster-defining v-t-cut is called the *representative* of the respective cluster

- scaling α yields a nested hierarchy of clusterings
- hierarchy has depth $\leq n-1$

- yields a guarantee (very rare!)
- user needs to choose suitable α carefully
- high runtime: $O(n)$ max-flow computations

Discussion: Min-Cut Tree Clustering

- a vertex v that served to identify a cluster-defining v-t-cut is called the *representative* of the respective cluster

- scaling α yields a nested hierarchy of clusterings
- hierarchy has depth $\leq n - 1$

- yields a guarantee (very rare!)
- user needs to choose suitable α carefully
- high runtime: $O(n)$ max-flow computations

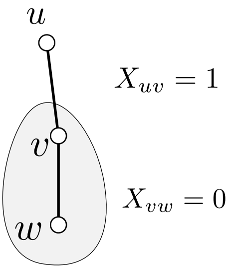
- no "minimum" in denominator of inter-cluster expansion$^* = \frac{\omega(E(C, V \setminus C))}{|V \setminus C|}$ (otherwise not always solvable)

Structure

Optimization: ILP Approach

1. introduce decision variables

$$\forall \{u, v\} \in \binom{V}{2} : X_{uv} = \begin{cases} 0 & \text{if } \mathcal{C}(u) = \mathcal{C}(v) \\ 1 & \text{otherwise} \end{cases}$$



2. ensure valid clustering with constraints (transitivity):

$$\forall \{u, v, w\} \in \binom{V}{3} : \begin{cases} X_{uv} + X_{vw} - X_{uw} \geq 0 \\ X_{uv} + X_{uw} - X_{vw} \geq 0 \\ X_{uw} + X_{vw} - X_{uv} \geq 0 \end{cases}$$

3. reflexivity and symmetry for free

Optimization: ILP Approach

4. optimize target function, e.g., modularity:

$$\mathrm{mod}_{\mathrm{ILP}}(G, \mathcal{C}_G) = \sum_{\{u,v\} \in \binom{V}{2}} \left(\omega(u,v) - \frac{\omega(u) \cdot \omega(v)}{2 \cdot \omega(E)} \right) \cdot X_{uv}$$

Optimization: ILP Approach

4. optimize target function, e.g., modularity:

$$\mathrm{mod}_{\mathrm{ILP}}(G, \mathcal{C}_G) = \sum_{\{u,v\} \in \binom{V}{2}} \left(\omega(u,v) - \frac{\omega(u) \cdot \omega(v)}{2 \cdot \omega(E)} \right) \cdot X_{uv}$$

Countless other constraints and objectives possible, e.g.,:

- bounded cluster sizes
- intra-/inter-expansion as constraint of objectives
- multicriteria objective functions
- maximum pairwise inter-cluster conductance (cumbersome)
- . . . **exercises**

Optimization: ILP Approach

4. optimize target function, e.g., modularity:

$$\mathrm{mod}_{\mathrm{ILP}}(G, \mathcal{C}_G) = \sum_{\{u,v\} \in \binom{V}{2}} \left(\omega(u,v) - \frac{\omega(u) \cdot \omega(v)}{2 \cdot \omega(E)} \right) \cdot X_{uv}$$

Countless other constraints and objectives possible, e.g.,:

- bounded cluster sizes
- intra-/inter-expansion as constraint of objectives
- multicriteria objective functions
- maximum pairwise inter-cluster conductance (cumbersome)
- …**exercises**

Example runtimes:

- modularity, 300 vertices, 1 day
- objective mpxc, constraint gid, 50 vertices, 1 day

[Görke: An algorithmic walk from static to dynamic graph clustering, 2010]
[Schumm et al.: Density-constrained graph clustering (technical report), 2011]

Structure

Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality

algorithm: ORCA

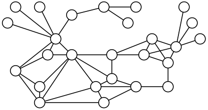
ORCA **R**eduction-

and **C**ontr**A**ction-based Clustering

Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality



algorithm: ORCA

ORCA **R**eduction-

and **C**ontr**A**ction-based Clustering

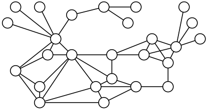
Clustering Huge Graphs

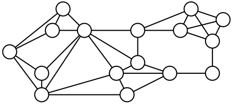
goals:

- fast on huge graphs
- with very good quality

algorithm: ORCA

ORCA **R**eduction-
and **C**ontr**A**ction-based Clustering





main ingredients

- reduction: *2-core*

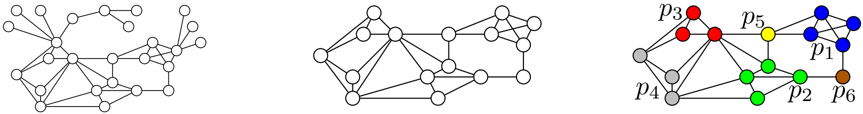
Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality

algorithm: ORCA

ORCA **R**eduction- and **C**ontr**A**ction-based Clustering



main ingredients

- reduction: *2-core*
- local: γ*-cliques*

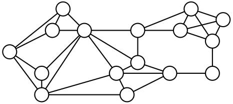
Clustering Huge Graphs

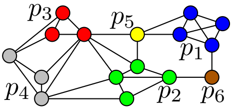
goals:

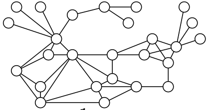
- fast on huge graphs
- with very good quality

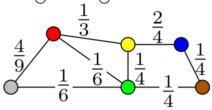
algorithm: ORCA

ORCA **R**eduction-
and **C**ontr**A**ction-based Clustering









main ingredients

- reduction: *2-core*
- local: γ-*cliques*
- contractions

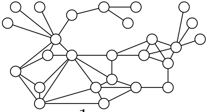
Clustering Huge Graphs

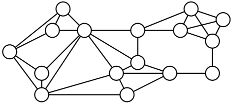
goals:

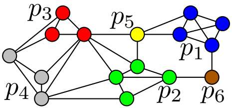
- fast on huge graphs
- with very good quality

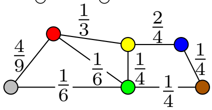
algorithm: ORCA

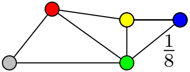
ORCA **R**eduction-
and **C**ontr**A**ction-based Clustering











main ingredients

- reduction: *2-core*
- local: γ*-cliques*
- contractions

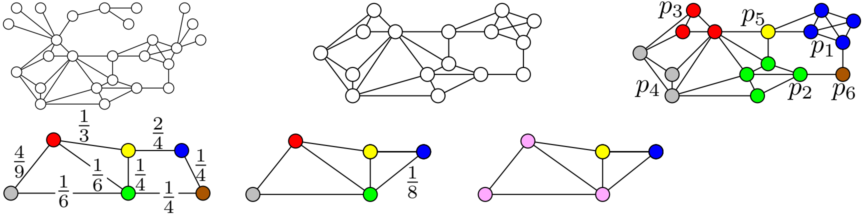
Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality

algorithm: ORCA

ORCA **R**eduction- and **C**ontr**A**ction-based Clustering



main ingredients

- reduction: *2-core*
- local: γ-*cliques*
- contractions

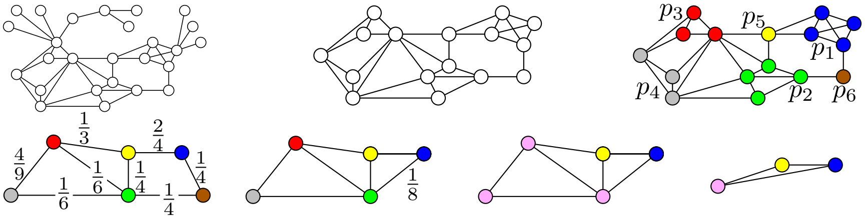
Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality

algorithm: ORCA

ORCA **R**eduction-
and **C**ontr**A**ction-based Clustering



main ingredients

- reduction: *2-core*
- local: γ*-cliques*
- contractions

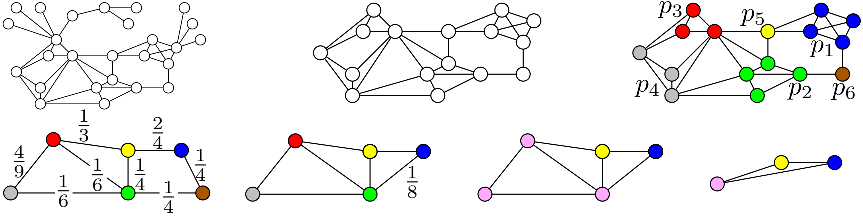
Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality

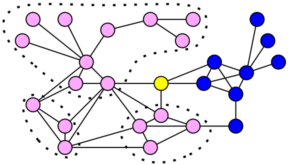
algorithm: ORCA

ORCA **R**eduction-
and **C**ontr**A**ction-based Clustering



main ingredients

- reduction: *2-core*
- local: γ*-cliques*
- contractions
- hierarchy



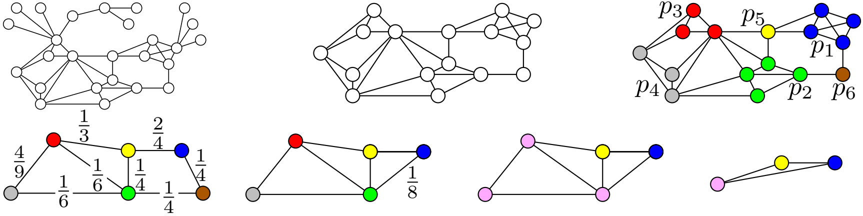
Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality

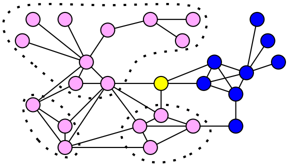
algorithm: ORCA

ORCA **R**eduction-
and **C**ontr**A**ction-based Clustering



main ingredients

- reduction: *2-core*
- local: γ*-cliques*
- contractions
- hierarchy



ORCA clusters

- 1M/10M in seconds
- 20M/.5B in 2h
- not “one-dimensional”
- with good quality

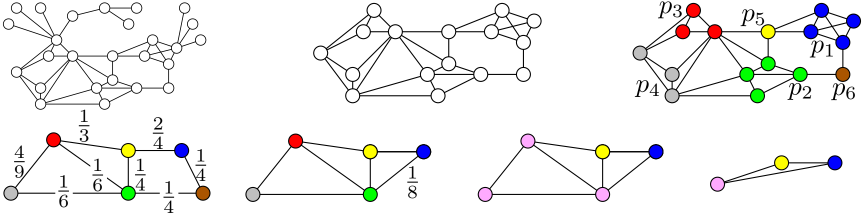
Clustering Huge Graphs

goals:

- fast on huge graphs
- with very good quality

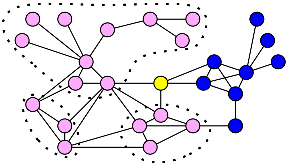
algorithm: ORCA

ORCA **R**eduction-
and **C**ontr**A**ction-based Clustering



main ingredients

- reduction: *2-core*
- local: γ*-cliques*
- contractions
- hierarchy



[Delling et al.: ORCA, '09]

ORCA clusters

- 1M/10M in seconds
- 20M/.5B in 2h
- not "one-dimensional"
- with good quality

Other Approaches (I)

Karlsruher Institut für Technologie

Spectral Clustering

1. use adjacency matrix/Laplacian
2. project points into low (k-) dimensional space
3. assign points to closest axes (e.g. [Kannan et al. '00])
   or use k-means on embedding (e.g. [Shi and Malik '00])

- blurs the line between data- and graph clustering
- variants solve, e.g., the relaxed RATIOCUT problem

Random Walks

1. simulate long random walk through graph
2. convergence $\rightsquigarrow$ transition matrix induces clusters

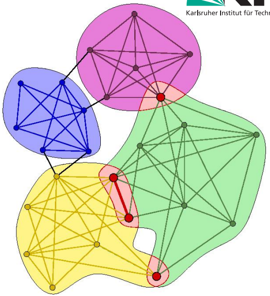
- pioneered in [van Dongen '02]
- related to spectral graph theory

Other Approaches (II)

Clique-Percolation

1. "roll" small clique-template through graph
2. reachable parts in same cluster

- resonable but sensitive to structure
- slow on large instances
- overlapping clusters!
- [Palla et al.: Uncovering the overlapping community structure of complex networks in nature and society, 2005]



(template: K_4)

Network Percolation

1. iteratively remove most central edges in graph
2. stop at threshold $\rightsquigarrow$ components induce clusters

- done, e.g., by [Girvan and Newman: Finding and evaluating community structure in networks '02]
- slow due to computation of centrality
- comp.: percolation theory from mathematics

Other Approaches (III)

- quantification of *node fitness*, migration/survival
- best clustering when using scalable parameter:
  find largest plateau in plot of $|\mathcal{C}|$
  [Santo Fortunato: Detecting the overlapping and hierarchical community structure in complex networks, 2009]
- direct translation of graph to data points $\rightsquigarrow$ k-means
  e.g.: [Gregor Stachowiak, student thesis, 2011]
- randomized rounding of linear programs
- emulating electricity: clustering by voltage potential
- ...

- overviews:
  [Brandes, Erlebach (eds.) '05, Network Analysis, Methodological Foundations]
  [Satu Elisa Schaeffer: Graph Clustering, 2007]
  [Santo Fortunato: Community Structure in Graphs, 2009]
  [Robert Görke: An algorithmic walk from static to dynamic graph clustering, 2010]

Structure

Why Experiments?

"...an experiment and emprical data are more valuable than an estimate; an estimate is more valuable than an approximate calculation; an approximate calculation is more valuable than a rigorous result."
[Dorogovtsev and Mendes: Evolution of Networks, 2003 (physicists)]

Why Experiments?

“. . . an experiment and empirical data are more valuable than an estimate; an estimate is more valuable than an approximate calculation; an approximate calculation is more valuable than a rigorous result.”

[Dorogovtsev and Mendes: Evolution of Networks, 2003 (physicists)]

mathematicians
& computer scientists
disagree!

Why Experiments?

"…an experiment and empirical data are more valuable than an estimate; an estimate is more valuable than an approximate calculation; an approximate calculation is more valuable than a rigorous result."
[Dorogovtsev and Mendes: Evolution of Networks, 2003 (physicists)]

mathematicians & computer scientists disagree!

Specifica in field of graph clustering:

- generally, "rigorous results" are always preferable, however …
- hard problems, very few results possible/probable
  (e.g. expansion via min-cut tree,
  or polylogarithmic quality via iterative. cond. cutting)
- performances of algorithms depend on graph type
  (again hard to capture, analytically)
- constants in runtimes do matter (huge networks)

Why Experiments?

"…an experiment and empirical data are more valuable than an estimate; an estimate is more valuable than an approximate calculation; an approximate calculation is more valuable than a rigorous result."
[Dorogovtsev and Mendes: Evolution of Networks, 2003 (physicists)]

mathematicians & computer scientists disagree!

Specifica in field of graph clustering:

- generally, "rigorous results" are always preferable, however …
- hard problems, very few results possible/probable
  (e.g. expansion via min-cut tree,
  or polylogarithmic quality via iterative. cond. cutting)
- performances of algorithms depend on graph type
  (again hard to capture, analytically)
- constants in runtimes do matter (huge networks)

$\Rightarrow$ **graph clustering does need extensive experiments**

Real-World Networks

- hand-selected collections
- biased in origin, size, structure, . . .
- are they representative, diverse, and suitatble?
- some rather widespread, e.g.:

Real-World Networks

- hand-selected collections
- biased in origin, size, structure, ...
- are they representative, diverse, and suitatble?
- some rather widespread, e.g.:

source	web address
Arenas	`http://deim.urv.cat/~aarenas/data/welcome.htm`
ANoack	`http://www-sst.informatik.tu-cottbus.de/~an/GD/`
Cx-Nets	`http://cxnets.googlepages.com/`
GraphDrawing	`http://vlado.fmf.uni-lj.si/pub/networks/data/GD/GD.htm`
Newman	`http://www-personal.umich.edu/~mejn/netdata/`
pajek	`http://vlado.fmf.uni-lj.si/pub/networks/data/`
UriAlon	`http://www.weizmann.ac.il/mcb/UriAlon/`
Walshaw	`http://staffweb.cms.gre.ac.uk/~c.walshaw/partition/`
DIMACS	`http://www.cc.gatech.edu/dimacs10/`

table partially taken from and many used in:
[Noack and Rotta: Multi-level algorithms for modularity clustering, 2009]

Generated Instances

real-word networks cannot answer all questions, e.g.:

- does a measure fulfill specific desiderata on scaling?
- scaling of algorithm's runtime on specific graph family?
- quality of algorithm sensitive to graph density?
- ...

Generated Instances

real-word networks cannot answer all questions, e.g.:

- does a measure fulfill specific desiderata on scaling?
- scaling of algorithm's runtime on specific graph family?
- quality of algorithm sensitive to graph density?
- ...

$\Rightarrow$ targeted experiments using artificial instances necessary

Generated Instances

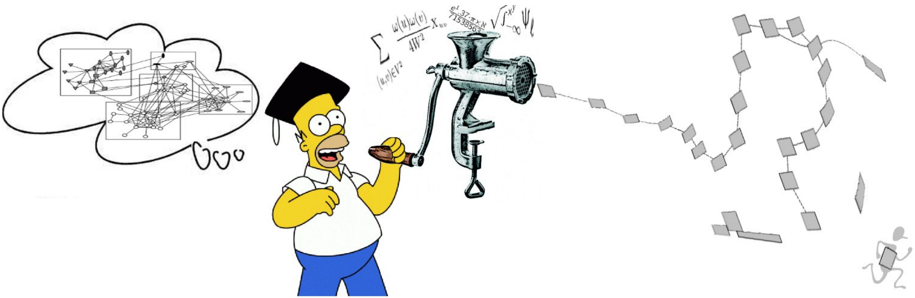
real-word networks cannot answer all questions, e.g.:

- does a measure fulfill specific desiderata on scaling?
- scaling of algorithm's runtime on specific graph family?
- quality of algorithm sensitive to graph density?
- . . .

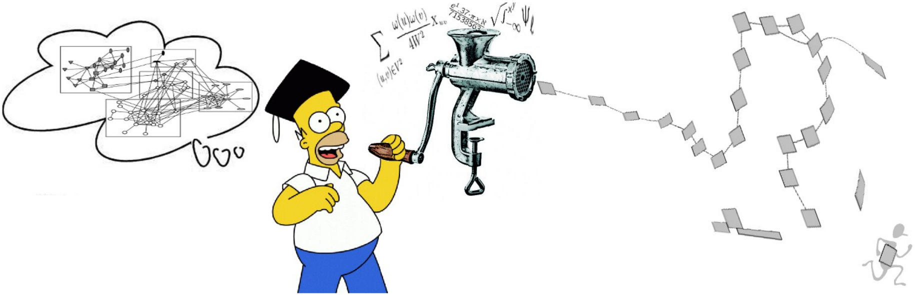
⇒ targeted experiments using artificial instances necessary

some resources:
[Brandes et al.: Experiments on graph clustering algorithms, 2003]
[Delling et al.: Generating Significant Graph Clusterings, 2006]
[Gaertler et al.: PhD Thesis, 2007 (Unit-Test-oriented evaluations)]
[Görke and Staudt: A generator for dynamic clustered random graphs, 2009
download static & dynamic generators:
`http://i11www.iti.uni-karlsruhe.de/en/projects/spp1307/index`]
[Görke et al.: Computational aspects of lucidity-driven graph clustering, 2010]

Artificial Test Instances . . .



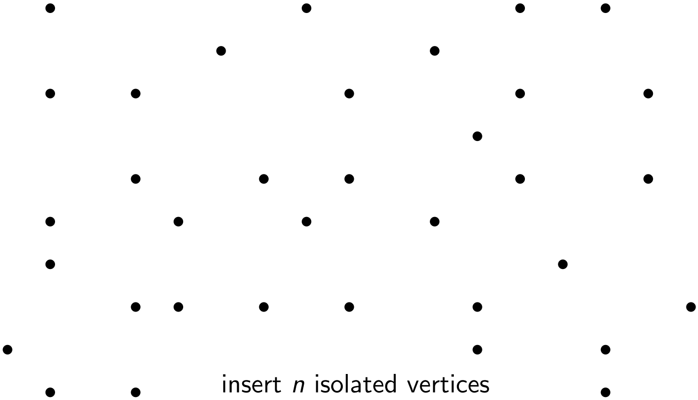
Artificial Test Instances . . .



The design of artificial test instances requires care!

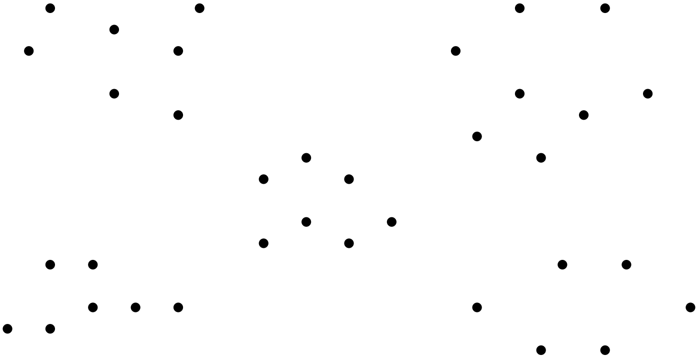
$G(n, p_{\text{in}}, p_{\text{out}})$

...a simple model for random preclustered graphs



$G(n, p_{\text{in}}, p_{\text{out}})$

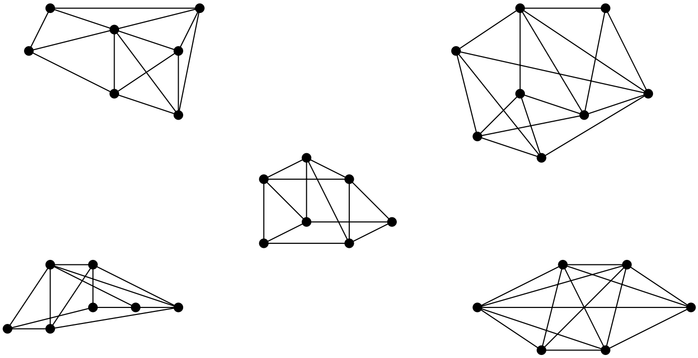
...a simple model for random preclustered graphs



partition V into random/controlled number of blocks

$G(n, p_{\mathbf{in}}, p_{\mathbf{out}})$

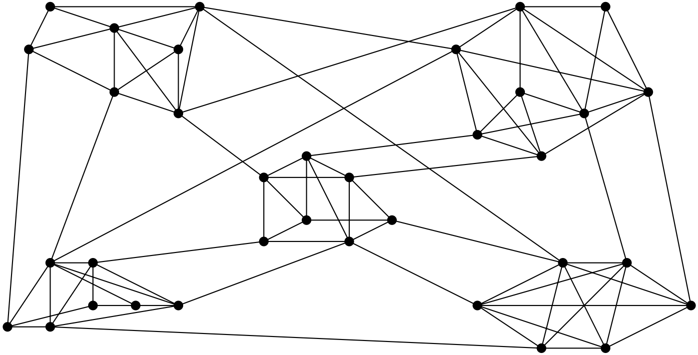
...a simple model for random preclustered graphs



intra-edges: build a cluster $G(|C_i|, p_{\text{in},i})$ from each block i

$G(n, p_{\mathbf{in}}, p_{\mathbf{out}})$

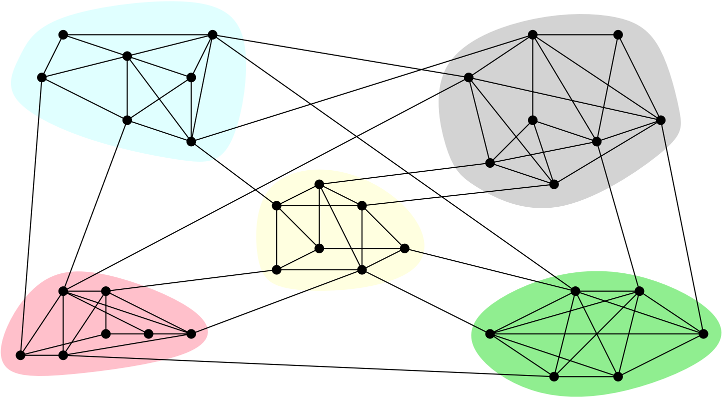
…a simple model for random preclustered graphs



inter-edges: connect inter-cluster pairs with probability p_{out}

$G(n, p_{\mathbf{in}}, p_{\mathbf{out}})$

…a simple model for random preclustered graphs

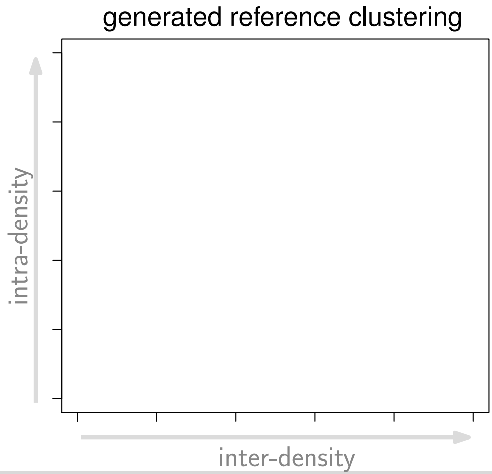


a random graph with a planted clustering

- indisputably(?) scalable intra- and inter-cluster quality
- scalable in size, generation algorithm very fast ($O(m+n)$)
- reference / ground truth to compare to

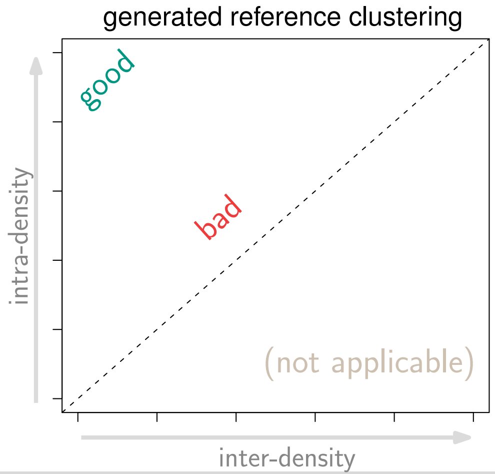
Example Experimental Setup

- systematic exp. evaluation: measure modularity vs. intuition



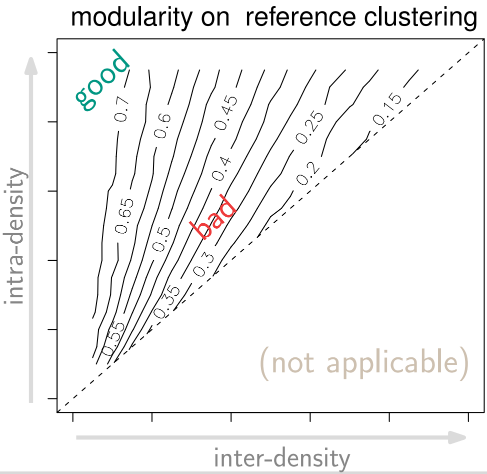
Example Experimental Setup

- systematic exp. evaluation: measure modularity vs. intuition



Example Experimental Setup

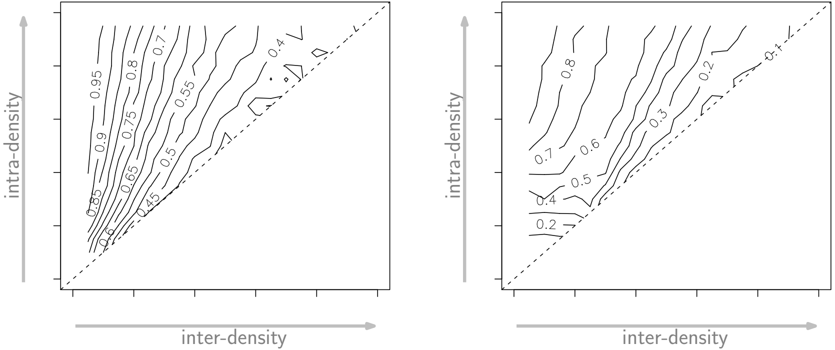
- systematic exp. evaluation: measure modularity vs. intuition $\surd$



Example Experimental Setup

- systematic exp. evaluation: measure modularity vs. intuition $\surd$
- modularity as objective function for maximization:
  greedy maxim. vs. established algorithms wrt. established measures $\surd$

established quality measure *max. is. inter-cl. conductance* (mixc)



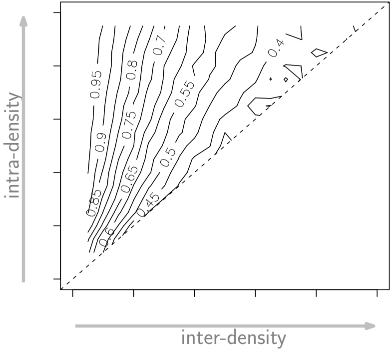
algo.: greedy modularity-max.

algo.: *MCL* (rand. walks, est.)

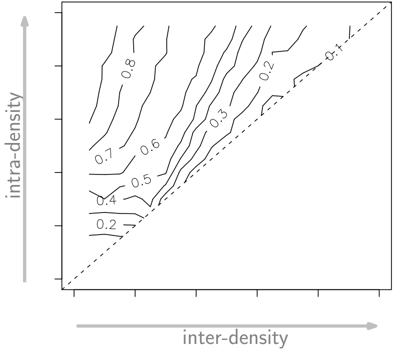
Example Experimental Setup

- systematic exp. evaluation: measure modularity vs. intuition $\surd$
- modularity as objective function for maximization:
  greedy maxim. vs. established algorithms wrt. established measures $\surd$

established quality measure *max. is. inter-cl. conductance* (mixc)



algo.: greedy modularity-max.

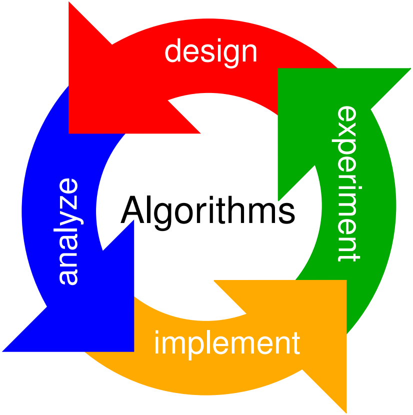


algo.: *MCL* (rand. walks, est.)

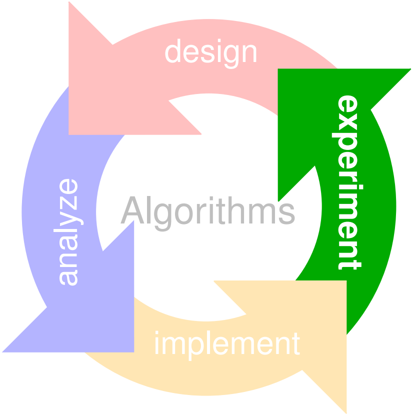
[Görke et al.: Computational aspects of lucidity-driven graph clustering, 2010]

Structure

A Crucial Ingredient of AE

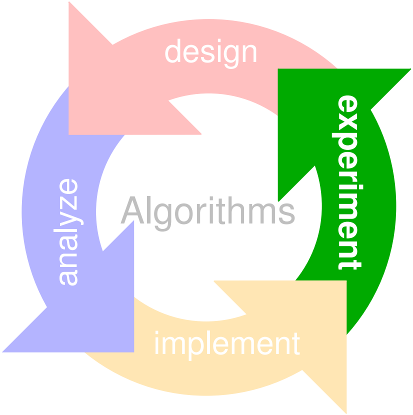


A Crucial Ingredient of AE



instances for experimental evaluation

A Crucial Ingredient of AE



instances for experimental evaluation

“do we care enough about them?”

Meaning of Experiments in AE

State properties of algorithms via experimental evaluation

Algorithm theory:

- asymptotic runtimes: worst/average case, smoothed analysis
- quality: optimality, diverse approximation guarantees
- outcomes on special families of instances

Do experimental evaluations in AE mirror this?

Experiments in AE

Good, valuable evaluations require:

- relevant experiments
- controlled experiments
- provable properties of test instances
- behavior of algorithms on different generated instances
- outcomes on special families of graphs

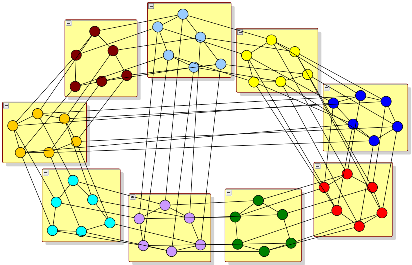
Experiments in AE

Good, valuable evaluations require:

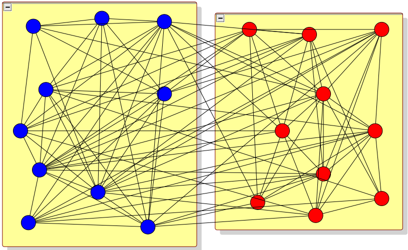
- relevant experiments
- controlled experiments
- provable properties of test instances
- behavior of algorithms on different generated instances
- outcomes on special families of graphs

Postulation 1	more insights into characteristics of instances, characteristics that affect behavior of algorithm
Postulation 2	suitable generated instances compulsive: hard, variable outcome/properties, easy

Pitfalls of Quality Indices



meaningful clustering of six-sided tube with irregularities
coverage = 0.43



random split of a $G(20, \frac{1}{2})$ random graph
coverage = 0.66

“A Good Graph Clustering”

Widespread quality function: *modularity*

$$\text{mod}(\mathcal{C}) := \text{cov}(\mathcal{C}) - \mathbb{E}(\text{cov}(\mathcal{C}))$$

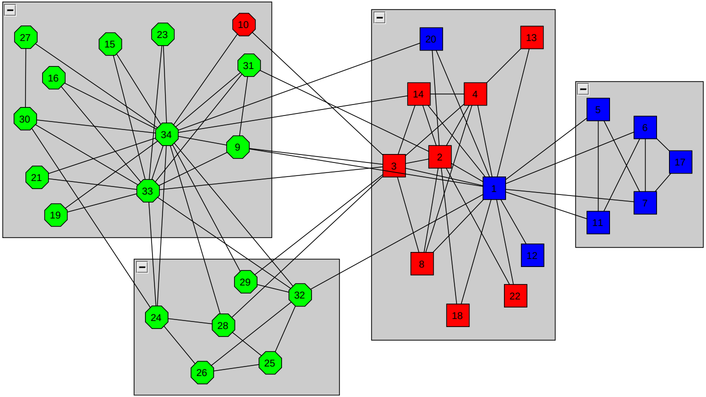
$$= \frac{\#\text{ intra-cluster edges}}{|\#edges|} - \frac{1}{4|\#edges|^2} \sum_{C \in \mathcal{C}} \left(\sum_{v \in C} \deg(v) \right)^2$$

“In practice, values [. . .] from about 0.3 to 0.7. Higher values are rare.”
[Girvan & Newman ’04]

“. . . in practice [. . .] a value above about 0.3 *is a good indicator of significant community structure . . . ”*
[Newman et al. ’04]

Archon of Benchmark Graphs

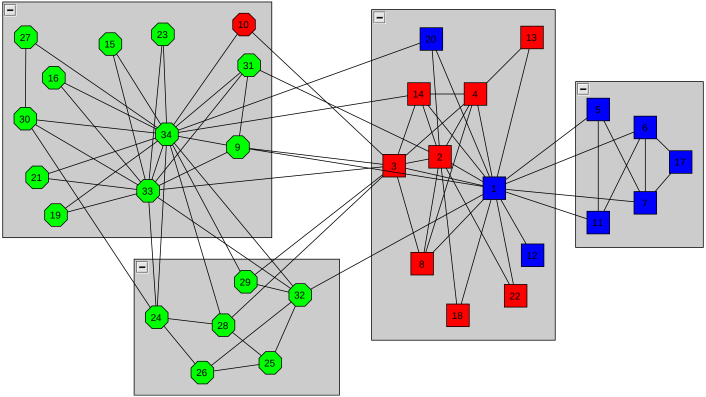
...one of roughly a dozen real-world instances:
Zachary's Karate Club



modularity:
reality (shapes) & heuristic (col.) & optimum (boxes) ≈ 0.4

Archon of Benchmark Graphs

…one of roughly a dozen real-world instances:
Zachary's Karate Club



But [Reichard & Bornhold '08]: $\mathbb{E}(\text{modularity}) \approx 0.42 > 0.4$!
(in an *Erdős-Rényi* model)

Large Networks

Example results of ORCA on webgraphs
(**O**RCA **R**eduction and **C**ontr**A**ction-based Clustering
[Delling et al.: ORCA Reduction and ContrAction Graph Clustering, 2009]

Instance	n/m	Algorithm	clusters	icc	perf.	cov.	mod.
cnr-2000	325 556	local greedy	242	0.8571	0.9799	0.9971	0.9130
	5 565 376	ORCA	110	0.0002	0.9632	0.9427	0.8567
eu-2005	862 664	local greedy	326	0.7668	0.9643	0.9708	0.9376
	32 778 307	ORCA	217	0.0002	0.9458	0.7965	0.7014
in-2004	1 382 908	local greedy	1004	0.0000	0.9931	0.9234	0.9094
	27 560 318	ORCA	740	0.0002	0.9877	0.9503	0.9288
uk-2002	18 520 486	local greedy	6280	0.0000	0.9981	0.5693	0.5671
	529 444 599	ORCA	66595	0.0000	0.9995	0.8758	0.8749

Large Networks

Karlsruher Institut für Technologie

Example results of ORCA on webgraphs
(**O**RCA **R**eduction and **C**ontr**A**ction-based Clustering
[Delling et al.: ORCA Reduction and ContrAction Graph Clustering, 2009]

Instance	n/m	Algorithm	clusters	icc	perf.	cov.	mod.
cnr-2000	325 556	local greedy	242	0.8571	0.9799	0.9971	0.9130
	5 565 376	ORCA	110	0.0002	0.9632	0.9427	0.8567
eu-2005	862 664	local greedy	326	0.7668	0.9643	0.9708	0.9376
	32 778 307	ORCA	217	0.0002	0.9458	0.7965	0.7014
in-2004	1 382 908	local greedy	1004	0.0000	0.9931	0.9234	0.9094
	27 560 318	ORCA	740	0.0002	0.9877	0.9503	0.9288
uk-2002	18 520 486	local greedy	6280	0.0000	0.9981	0.5693	0.5671
	529 444 599	ORCA	66595	0.0000	0.9995	0.8758	0.8749

$\Rightarrow$ With size, *sparseness* gets the upper hand

Large Networks

Example results of ORCA on webgraphs
(**O**RCA **R**eduction and **C**ontr**A**ction-based Clustering
[Delling et al.: ORCA Reduction and ContrAction Graph Clustering, 2009]

Instance	n/m	Algorithm	clusters	icc	perf.	cov.	mod.
cnr-2000	325 556	local greedy	242	0.8571	0.9799	0.9971	0.9130
	5 565 376	ORCA	110	0.0002	0.9632	0.9427	0.8567
eu-2005	862 664	local greedy	326	0.7668	0.9643	0.9708	0.9376
	32 778 307	ORCA	217	0.0002	0.9458	0.7965	0.7014
in-2004	1 382 908	local greedy	1004	0.0000	0.9931	0.9234	0.9094
	27 560 318	ORCA	740	0.0002	0.9877	0.9503	0.9288
uk-2002	18 520 486	local greedy	6280	0.0000	0.9981	0.5693	0.5671
	529 444 599	ORCA	66595	0.0000	0.9995	0.8758	0.8749

$\Rightarrow$ With size, *sparseness* gets the upper hand

"Modularity tends to 1 for practical instances" [Good et al. 09]

Graph Partitioning

Established benchmark library: “Walshaw’s test set”
(Walshaw gathered instances from existing evaluations)

Good, since:

- well known, accepted
- good comparability with other methods
- easy access, immediate usability

Graph Partitioning

Established benchmark library: “Walshaw’s test set”
(Walshaw gathered instances from existing evaluations)

Good, since:
- well known, accepted
- good comparability with other methods
- easy access, immediate usability

Downside:
! contains other peoples’ handpicked instances
$\Rightarrow$ probably “good” instances. . .
$\Rightarrow$ bias? representative?

Graph Partitioning

Established benchmark library: “Walshaw’s test set”
(Walshaw gathered instances from existing evaluations)

Good, since:
- well known, accepted
- good comparability with other methods
- easy access, immediate usability

Downside:
! contains other peoples’ handpicked instances
$\Rightarrow$ probably “good” instances...
$\Rightarrow$ bias? representative?

Own experience:
- traditional network analysis
  $\Rightarrow$ no good characterization
- social networks $\Rightarrow$ bad runtime & cuts

Sensor Networks

Topics:

- minimum energy communication
- localization (without gps)
- interference minimization
- efficient data collection at sink node
- . . .

Sensor Networks

Topics:

- minimum energy communication
- localization (without gps)
- interference minimization
- efficient data collection at sink node
- ...

problem for research: real experimental setups rare
$\Rightarrow$ simulations, random instances

Sensor Networks

Topics:

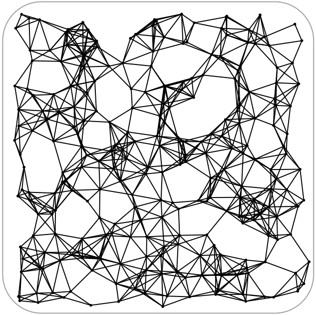
- minimum energy communication
- localization (without gps)
- interference minimization
- efficient data collection at sink node
- ...

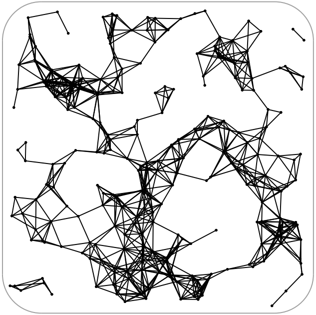
problem for research: real experimental setups rare
$\Rightarrow$ simulations, random instances

random instances: distribution of sensor nodes in plane
parameters: $|V|$, transmission radius $\Rightarrow$ average degree

Sensor Network Simulations

Bad: *"real randomness not nice":* smaller holes ⇒ better results!





1. place nodes on grid
2. slightly perturb

⇒ no big holes
⇒ method works

1. place nodes uniformly at random

⇒ harder challenges for method

Other researchers copy to compete . . .

Random Planar Graphs

Some generators in the literature:

- *CHT*[3] from LEDA
- *Delaunay* from LEDA
- *Node insertion* from LEDA
- *Node expansion* [Krug 08]
- *Edge-on-off* Markov chain [Denise et al. 07]
- *Boltzmann sampler* [Fusy 07]

[3] Convex Hull Triangulation

Random Planar Graphs

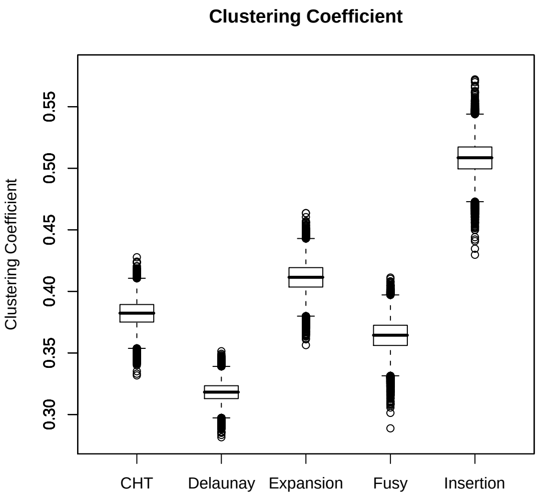
Some generators in the literature:

- *CHT*[3] from LEDA
- *Delaunay* from LEDA
- *Node insertion* from LEDA
- *Node expansion* [Krug 08]
- *Edge-on-off* Markov chain [Denise et al. 07]
- *Boltzmann sampler* [Fusy 07]

Equivalent? Biased?
"Does the choice impact my experiments?"

[3] Convex Hull Triangulation

Random Planar Graphs

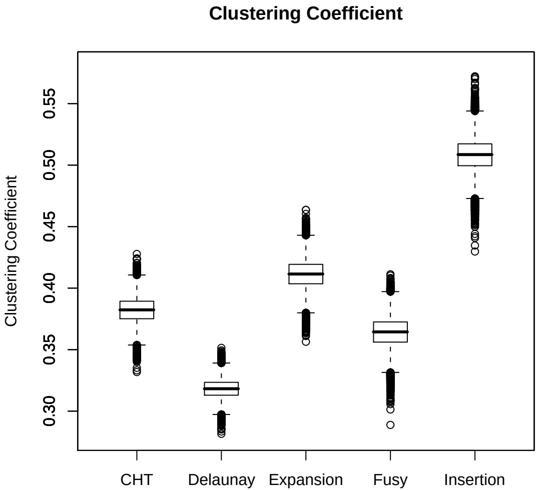


Fixed $|V|$

stat. sign.

different behavior!

Random Planar Graphs



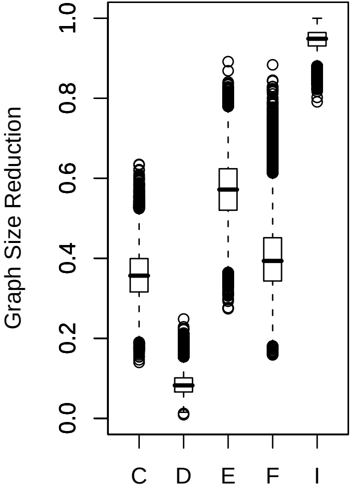
Fixed $|V|$

stat. sign.

different behavior!

Does it matter?

Random Planar Graphs



Yes, it matters!

algorithm:
k-VertexCover FPT

measure:
reduction by 0,1,2-kernelization

Concluding Claim

In algorithm engineering we need more

- test instances with provable properties,
  (hard, variable outcome/properties, easy, etc.)
  leading to...
- controlled experiments

Concluding Claim

In algorithm engineering we need more

- test instances with provable properties,
  (hard, variable outcome/properties, easy, etc.)
  leading to...
- controlled experiments

- insights on outcomes on special families, or at least...
- on behavior of algorithms on different generated instances

Concluding Claim

Karlsruher Institut für Technologie

In algorithm engineering we need more

- test instances with provable properties,
  (hard, variable outcome/properties, easy, etc.)
  leading to...
- controlled experiments

- insights on outcomes on special families, or at least...
- on behavior of algorithms on different generated instances

- insights into characteristics of instances, characteristics that affect behavior of algorithm

Structure

Three Paradigms

1. **Quality-based:** comparison by quality index
   $\Rightarrow$ structurally different clusterings may be of same quality

Three Paradigms

1. **Quality-based:** comparison by quality index
   $\Rightarrow$ structurally different clusterings may be of same quality

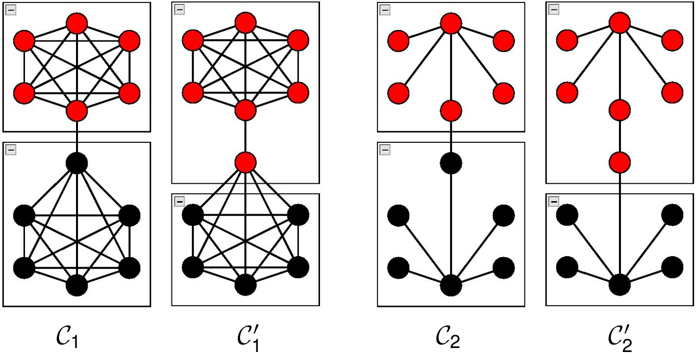
2. **Set-based:** distance depends only on partition of V
   long history in data mining
   $\Rightarrow$ independent of graph structure

Three Paradigms

1. **Quality-based:** comparison by quality index
   $\Rightarrow$ structurally different clusterings may be of same quality
2. **Set-based:** distance depends only on partition of V
   long history in data mining
   $\Rightarrow$ independent of graph structure
3. **Graph-based:** distance depends on partitioning of nodes and structure of graph

Comparing Graph Clusterings

. . . how set-based measures fail intuition



"Are the two left-hand clusterings less similar than those on the right?"

Augmenting Set-based Measurements 1/3

Counting Pairs:

- compare co-classification of all pairs of nodes
- example: *Rand ('71)* index

Augmenting Set-based Measurements 1/3

Counting Pairs:

- compare co-classification of all pairs of nodes
- example: *Rand ('71)* index

- instead of all node pairs $\Rightarrow$ use only connected pairs

$$\mathcal{R}(\mathcal{C},\mathcal{C}') := 1 - \frac{n_{\text{tog,tog}} + n_{\text{sep,sep}}}{\frac{1}{2}|N|(|N|-1)} \quad \leadsto \quad 1 - \frac{e_{\text{tog,tog}} + e_{\text{sep,sep}}}{|E|}$$

$n_{\text{tog,tog}} =$ number of pairs of nodes that are together in $\mathcal{C}$ and together in $\mathcal{C}'$,
$n_{\text{sep,sep}}$ analogous,
$e_{\text{tog,tog}} =$ number of edges that are intra for $\mathcal{C}$ and for $\mathcal{C}'$,
$e_{\text{sep,sep}}$ analogous

Augmenting Set-based Measurements 1/3

Counting Pairs:

- compare co-classification of all pairs of nodes
- example: *Rand ('71)* index

- instead of all node pairs $\Rightarrow$ use only connected pairs

$$\mathcal{R}(\mathcal{C},\mathcal{C}') := 1 - \frac{n_{\text{tog,tog}} + n_{\text{sep,sep}}}{\frac{1}{2}|N|(|N|-1)} \quad \rightsquigarrow \quad 1 - \frac{e_{\text{tog,tog}} + e_{\text{sep,sep}}}{|E|}$$

$n_{\text{tog,tog}}$ = number of pairs of nodes that are together in $\mathcal{C}$ and together in $\mathcal{C}'$,
$n_{\text{sep,sep}}$ analogous,
$e_{\text{tog,tog}}$ = number of edges that are intra for $\mathcal{C}$ and for $\mathcal{C}'$,
$e_{\text{sep,sep}}$ analogous

(will be used later when clustering dynamic graphs)

Augmenting Set-based Measurements 2/3

Karlsruher Institut für Technologie

Maximum Overlap:

- match clusters
- example: *normalized van Dongen ('00)* index

Augmenting Set-based Measurements 2/3

Maximum Overlap:

- match clusters
- example: *normalized van Dongen ('00)* index
- $\Rightarrow$ weight nodes by their degrees

$$\mathcal{NVD}(\mathcal{C},\mathcal{C}') := 1 \quad - \frac{1}{2n}\sum_{C_i\in\mathcal{C}} \max_{C'_j\in\mathcal{C}'} \underbrace{m_{ij}}_{\rightsquigarrow m^d_{ij}} \quad - \frac{1}{2n}\sum_{C'_j\in\mathcal{C}'} \max_{C_i\in\mathcal{C}} \underbrace{m_{ij}}_{\rightsquigarrow m^d_{ij}}$$

with $m_{ij} = |C_i \cap C'_j|$, where $C_i \in \mathcal{C}$ and $C'_j \in \mathcal{C}'$ (so-called *confusion matrix*)

Augmenting Set-based Measurements 3/3

Information Theory:

- ”what do you know about $\mathcal{C}'$ if $\mathcal{C}$ is given?”
- example: *Fred and Jain ('03)* index

Augmenting Set-based Measurements 3/3

Information Theory:

- "what do you know about $\mathcal{C}'$ if $\mathcal{C}$ is given?"
- example: *Fred and Jain ('03)* index

- $\Rightarrow$ weight probabilities by sum of node degrees

$$\mathcal{F}\mathcal{J}(\mathcal{C},\mathcal{C}') := \frac{2\mathcal{I}(\mathcal{C},\mathcal{C}')}{\mathcal{H}(\mathcal{C}) + \mathcal{H}(\mathcal{C}')}$$

Node-Entropy $\rightsquigarrow$ Edge-Entropy

Augmenting Set-based Measurements 3/3

Information Theory:

- "what do you know about $\mathcal{C}'$ if $\mathcal{C}$ is given?"
- example: *Fred and Jain ('03)* index

- $\Rightarrow$ weight probabilities by sum of node degrees

$$\mathcal{FJ}(\mathcal{C},\mathcal{C}') := \frac{2\mathcal{I}(\mathcal{C},\mathcal{C}')}{\mathcal{H}(\mathcal{C})+\mathcal{H}(\mathcal{C}')}$$

Node-Entropy $\rightsquigarrow$ Edge-Entropy

for all augmentations:
G regular $\Rightarrow$ graph-based $\cong$ set-based

(see [Delling at al.: Engineering comparators for graph clusterings, 2008] for definitions)

Editing Set Difference: Def.

Karlsruher Institut für Technologie

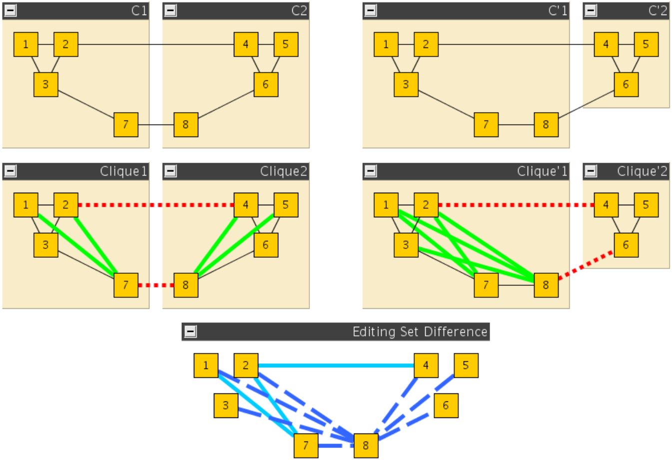
Definition (Editing Set Difference)

Let the Editing Sets of $G, \mathcal{C}$ and of $G, \mathcal{C}'$ be $F_{\mathcal{C}}, F_{\mathcal{C}'}$

$$\mathcal{ESD}(\mathcal{C}, \mathcal{C}') := \frac{|F_{\mathcal{C}} \triangle F_{\mathcal{C}'}|}{|F_{\mathcal{C}} \cup F_{\mathcal{C}'}|} = 1 - \frac{|F_{\mathcal{C}} \cap F_{\mathcal{C}'}|}{|F_{\mathcal{C}} \cup F_{\mathcal{C}'}|}$$

where $\triangle$ denotes the geometric difference between two sets

Editing Set Difference: Example

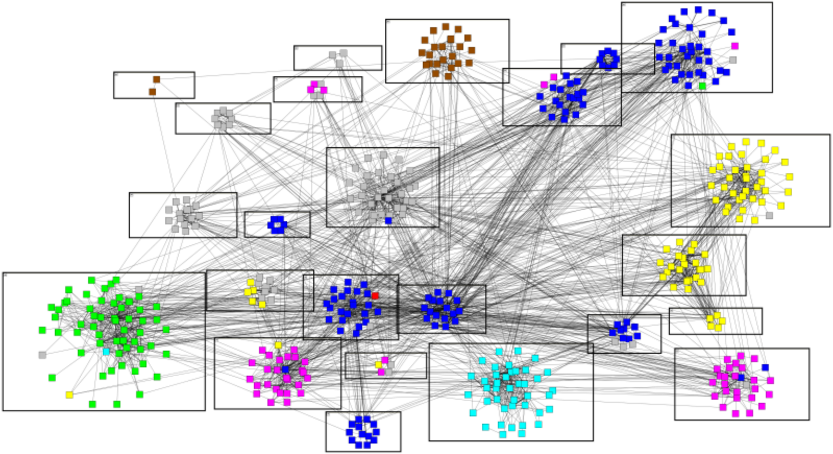


two clusterings

their editing sets

difference:
$1 - \frac{3}{10} = 0.7$

Real-World Graphs



. . . nice and necessary, but what about systematic insights?

Apparent Behavior

- High-degree nodes have more influence
- Cutting many edges $\leadsto$ higher difference

. . . average / practical / large-scale behavior?

Formalizing Postulations

The measure shall be *sensitive to the graph structure:*

Good vs. Random Clustering:

- $\textsc{dist}$(good cl., random cl.) $\approx$ big

Formalizing Postulations

The measure shall be *sensitive to the graph structure:*

Good vs. Random Clustering:

- DIST(good cl., random cl.) $\approx$ big
- DIST(good cl., random cl.) $\leq$ DIST(excellent cl., random cl.)

Formalizing Postulations

The measure shall be *sensitive to the graph structure:*

Good vs. Random Clustering:

- $\textsc{dist}$(good cl., random cl.) $\approx$ big
- $\textsc{dist}$(good cl., random cl.) $\leq$ $\textsc{dist}$(excellent cl., random cl.)

Perturbation:

Formalizing Postulations

Karlsruher Institut für Technologie

The measure shall be *sensitive to the graph structure:*

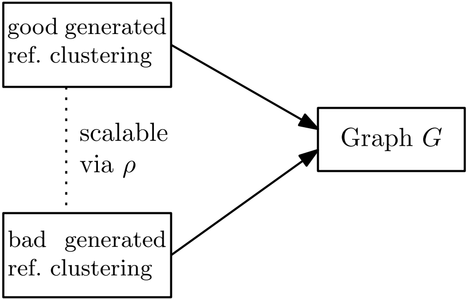
Good vs. Random Clustering:

- DIST(good cl., random cl.) $\approx$ big
- DIST(good cl., random cl.) $\leq$ DIST(excellent cl., random cl.)

Perturbation:

- DIST(good cl., Δgood cl.) $>$ DIST(Δgood cl., $\Delta\Delta$good cl.)

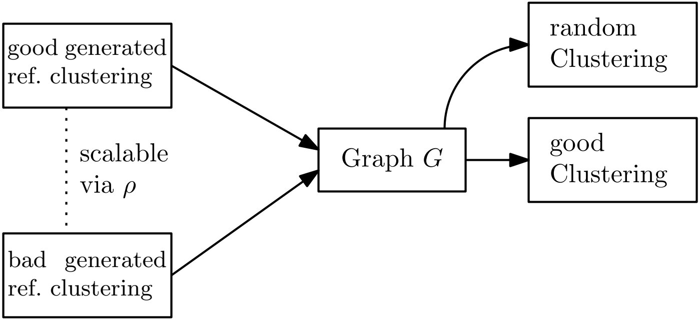
Experimental Setup



Systematic evaluation with *random preclustered graphs*:

- random graphs of parameterized structure
- *implanted* community structure with tunable significance ρ

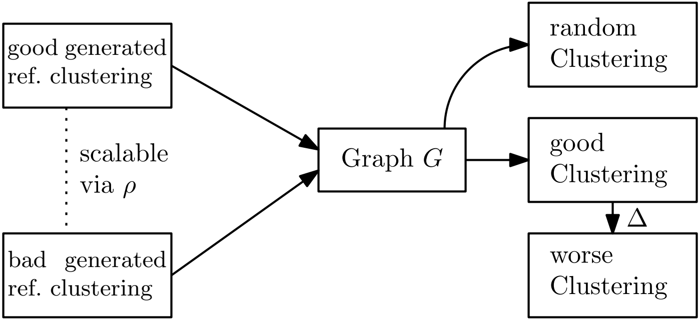
Experimental Setup



testing *Good vs. Random Clustering* by:

- $\mathcal{C}$: implanted (good) clustering
- $\mathcal{C}_R$: random clustering, same graph amd parameters

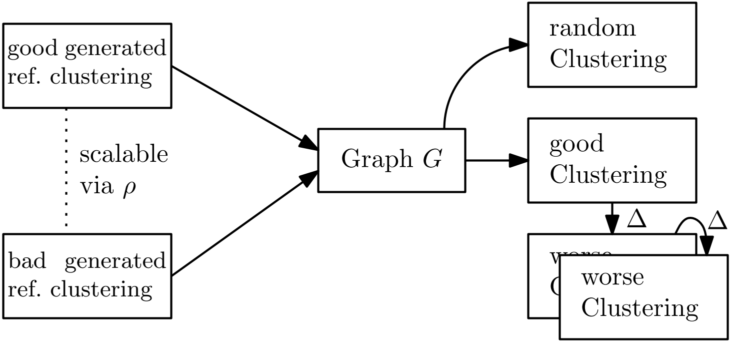
Experimental Setup



testing *Perturbation* by:

- $\mathcal{C}$: implanted (good) clustering
- $\Delta\mathcal{C}$: locally *worsen* clustering by moving nodes (max. dec. of quality (*modularity*), 0 – 500 moved nodes

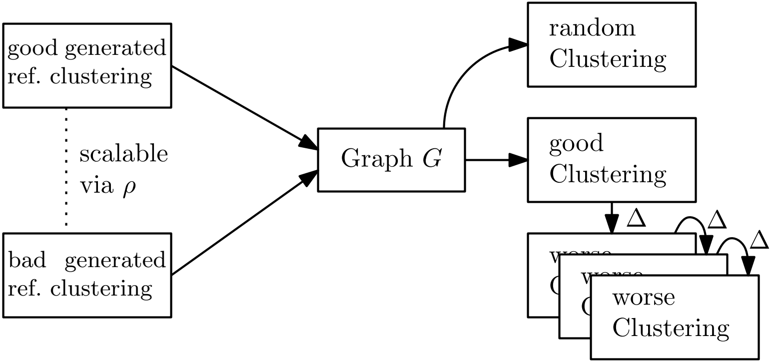
Experimental Setup



testing *Perturbation* by:

- $\mathcal{C}$: implanted (good) clustering
- $\Delta\mathcal{C}$: locally *worsen* clustering by moving nodes (max. dec. of quality (*modularity*), 0 – 500 moved nodes

Experimental Setup

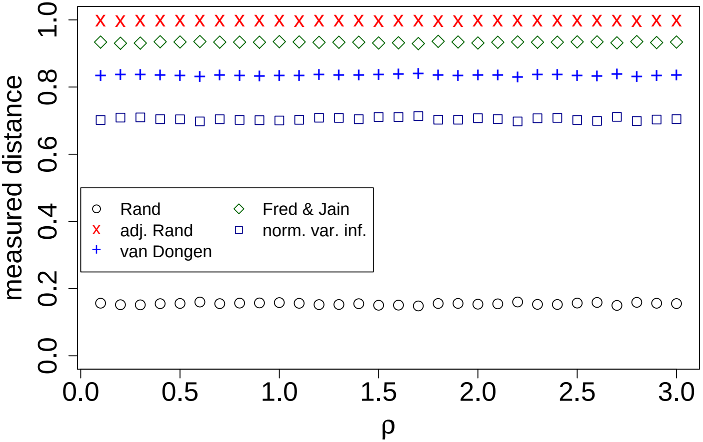


testing *Perturbation* by:

- $\mathcal{C}$: implanted (good) clustering
- $\Delta\mathcal{C}$: locally *worsen* clustering by moving nodes
  (max. dec. of quality (*modularity*), 0 – 500 moved nodes

Good vs. Rand. Clustering, $\mathcal{C}$ vs. $\mathcal{C}_R$

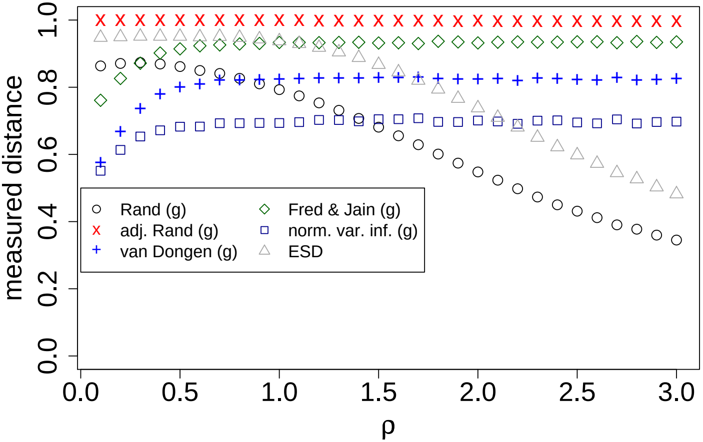
set-based measures fail postulations



ρ tunes clarity of implanted clustering (high $\rho \Rightarrow$ bad clust.)

Good vs. Rand. Clustering, $\mathcal{C}$ vs. $\mathcal{C}_R$

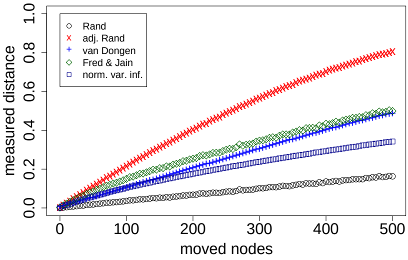
some graph-based measures comply



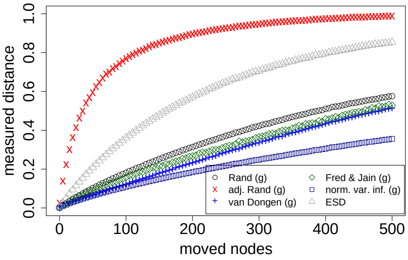
ρ tunes clarity of implanted clustering (high $\rho \Rightarrow$ bad clust.)

Perturbation of Good Clustering

set-based vs. graph-based



set-based
almost linear

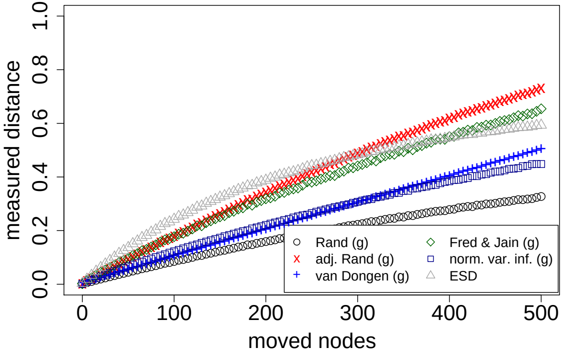


graph-based *ESD* and *adj. Rand*
non-linear

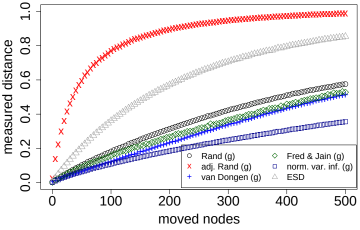
distance from implanted clustering after node migrations

Perturbation of Decent Clustering

set-based vs. graph-based



graph-based
less clear community structure



graph-based
clear community structure

distance from implanted clustering after node migrations

Summary on Comparators

- many well known measures for comparing set partitions
- point sets $\neq$ graphs $\Rightarrow$ edges are neglected!
- should not be used for graph clusterings!

Summary on Comparators

- many well known measures for comparing set partitions
- point sets $\neq$ graphs $\Rightarrow$ edges are neglected!
- should not be used for graph clusterings!

- systematic augmentation of many existing measures
- design of a new measure: *Editing Set Difference* $\mathcal{ESD}$
- recommended: $\mathcal{ESD}$, *graph-based (adjusted) Rand*

Summary on Comparators

- many well known measures for comparing set partitions
- point sets $\neq$ graphs $\Rightarrow$ edges are neglected!
- should not be used for graph clusterings!

- systematic augmentation of many existing measures
- design of a new measure: *Editing Set Difference* $\mathcal{ESD}$
- recommended: $\mathcal{ESD}$, *graph-based (adjusted) Rand*

- evaluation on real-world networks
- systematic evaluation of behavior $\Rightarrow$ feasibility

Summary on Comparators

Karlsruher Institut für Technologie

- many well known measures for comparing set partitions
- point sets $\neq$ graphs $\Rightarrow$ edges are neglected!
- should not be used for graph clusterings!

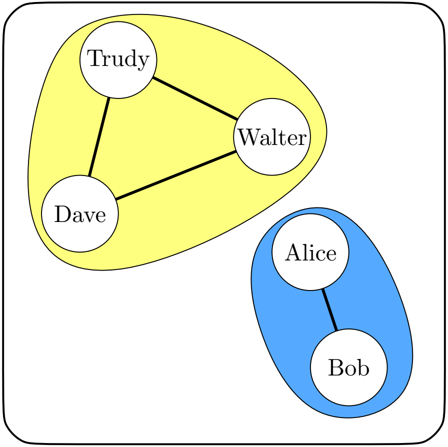
- systematic augmentation of many existing measures
- design of a new measure: *Editing Set Difference* $\mathcal{ESD}$
- recommended: $\mathcal{ESD}$, *graph-based (adjusted) Rand*

- evaluation on real-world networks
- systematic evaluation of behavior $\Rightarrow$ feasibility

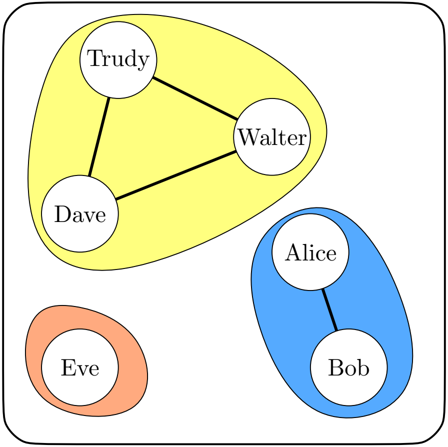
[Delling at al.: Engineering comparators for graph clusterings, 2008]

Structure

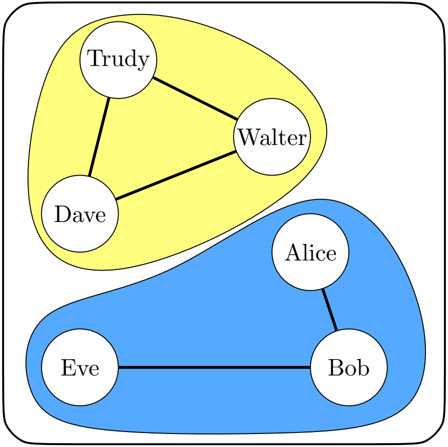
A Dynamic Network



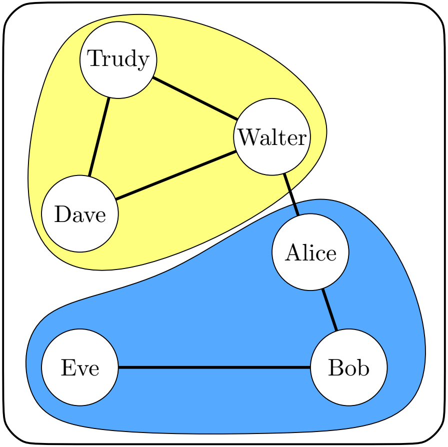
A Dynamic Network



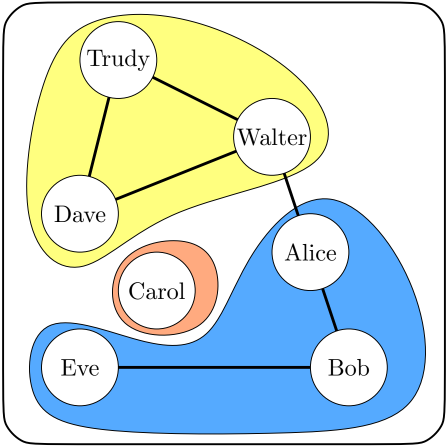
A Dynamic Network



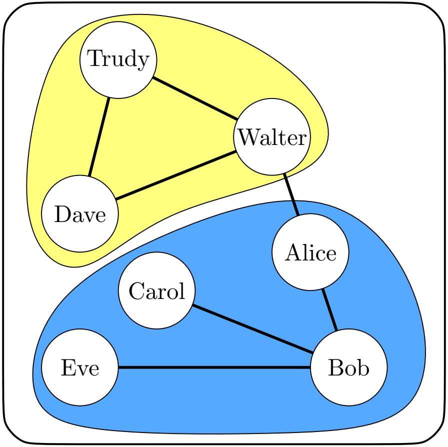
A Dynamic Network



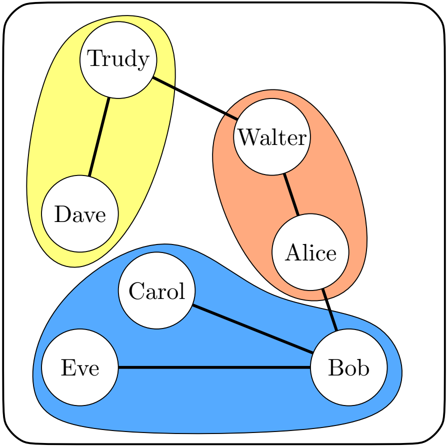
A Dynamic Network



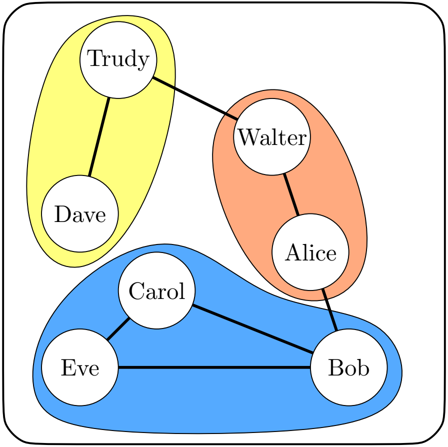
A Dynamic Network



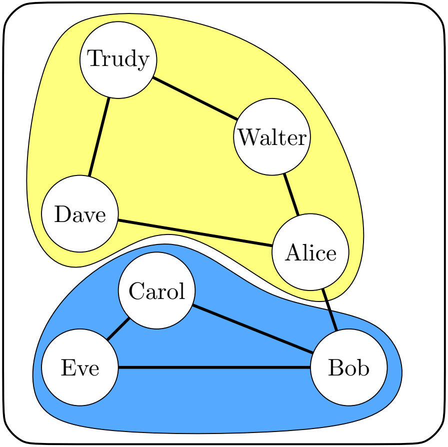
A Dynamic Network



A Dynamic Network

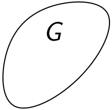


A Dynamic Network



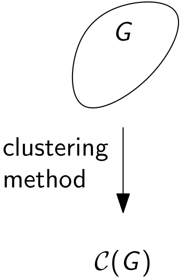
Structure

The Update Problem



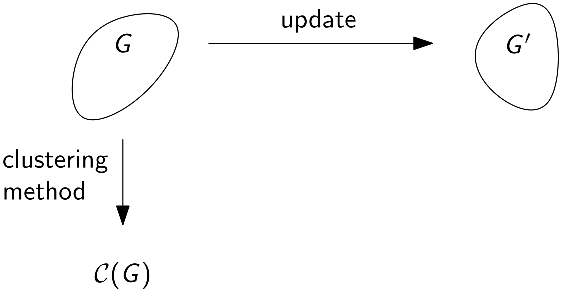
Given: graph G,

The Update Problem



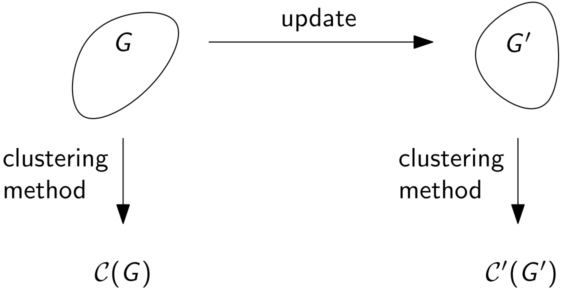
Given: graph G, technique $\mathcal{T}$, $\Rightarrow$ clustering $\mathcal{C}$

The Update Problem



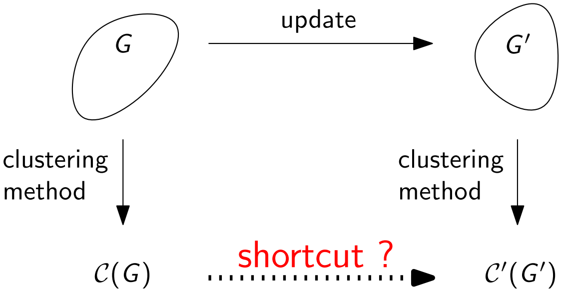
Given: graph G, technique $\mathcal{T}$, $\Rightarrow$ clustering $\mathcal{C}$
Then: modification Δ, $\Rightarrow$ graph G',

The Update Problem



Given: graph G, technique $\mathcal{T}$, $\Rightarrow$ clustering $\mathcal{C}$
Then: modification Δ, $\Rightarrow$ graph G', $\mathcal{T}$ $\Rightarrow$ clustering $\mathcal{C}'(G')$

The Update Problem



Given: graph G, technique $\mathcal{T}$, $\Rightarrow$ clustering $\mathcal{C}$
Then: modification Δ, $\Rightarrow$ graph G', $\mathcal{T} \Rightarrow$ clustering $\mathcal{C}'(G')$
Question: Is there a shortcut ?

Online Dynamic Graph Clustering

Dynamic Instances

changing networks with evolving group structure

$$\Downarrow$$

Dynamic Approach

update previous clustering reacting to changes in the graph

$$\begin{tikzcd} G \arrow[rr, "\Delta"] \arrow[d, "\mathcal{T}"'] & & G' \arrow[d, "\mathcal{T}"] \\ \mathcal{C}(G) \arrow[rr, dashed, "\mathcal{A}"] & & \mathcal{C}'(G') \end{tikzcd}$$

Clustering update problem

Criteria

- speed
- quality
- smooth transitions

Complexity of Optimization

Generally NP-hard if static problem is hard
$\Rightarrow$ generally NP-hard.

e.g., modularity:

Theorem

MODOPT is NP-hard [Brandes et al. 2008]

$$\Downarrow$$

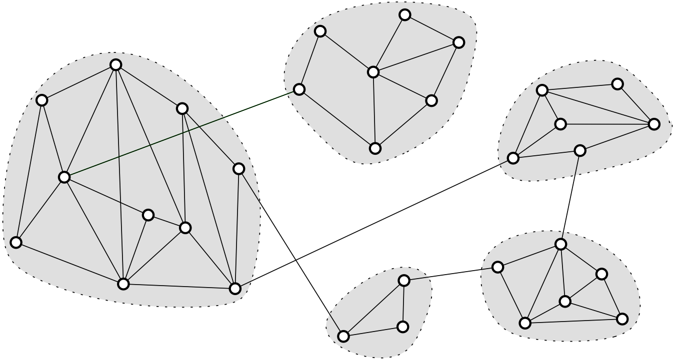
Corollary

DYNMODOPT is NP-hard [Görke et al. 2010]

idea for reduction: incrementally find static optimum with an efficient dynamic algorithm

Heuristics Based on Locality

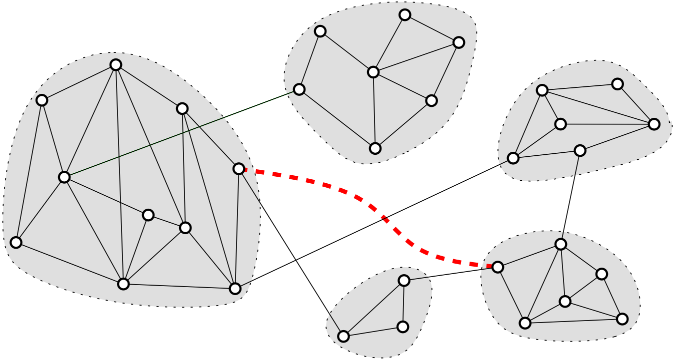
Dynamic *modularity*-maximization *without provable* quality



Heuristics Based on Locality

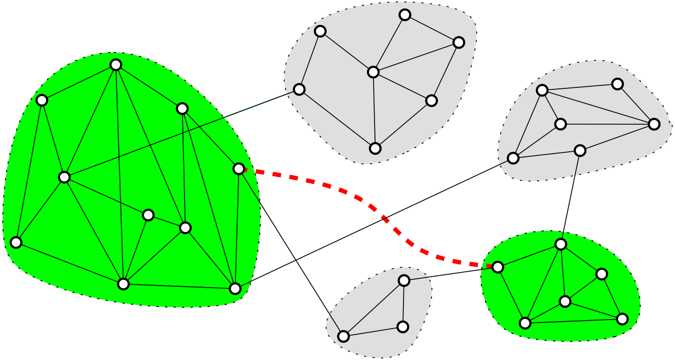
Karlsruher Institut für Technologie

Dynamic *modularity*-maximization *without provable* quality



Heuristics Based on Locality

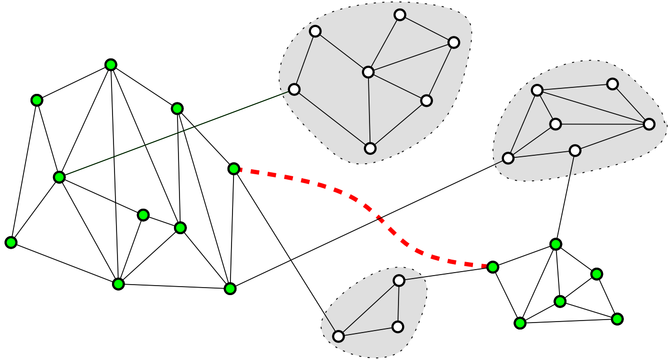
Dynamic *modularity*-maximization *without provable* quality



Changes in the graph invalidate:
affected clusters

Heuristics Based on Locality

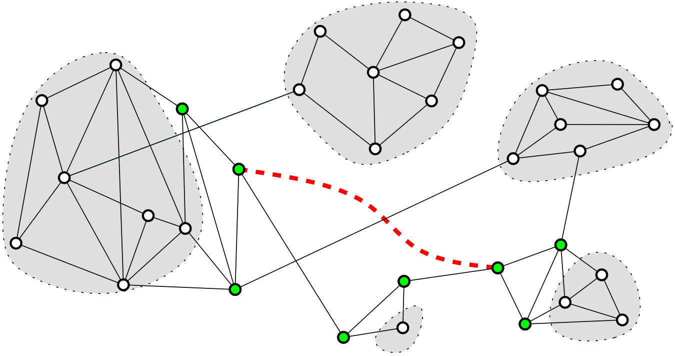
Dynamic *modularity*-maximization *without provable* quality



Changes in the graph invalidate:
affected clusters

Heuristics Based on Locality

Dynamic *modularity*-maximization *without provable* quality

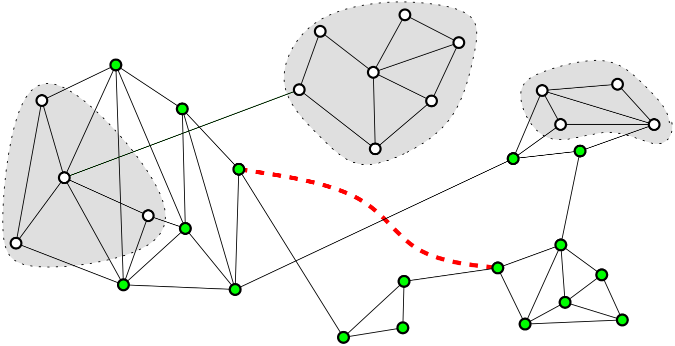


Changes in the graph invalidate:

local area: 1-hop neighborhood

Heuristics Based on Locality

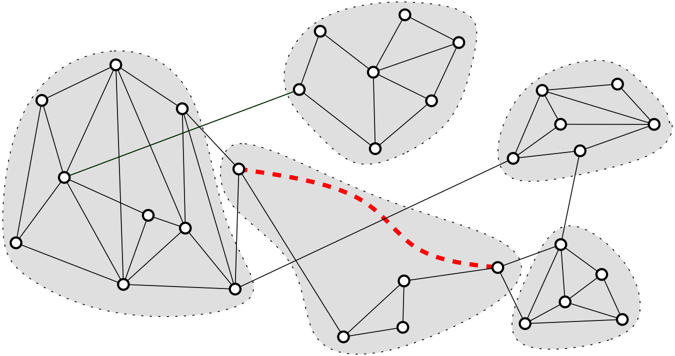
Dynamic *modularity*-maximization *without provable* quality



Changes in the graph invalidate:
local area: 2-hop neighborhood

Heuristics Based on Locality

Dynamic *modularity*-maximization *without provable* quality

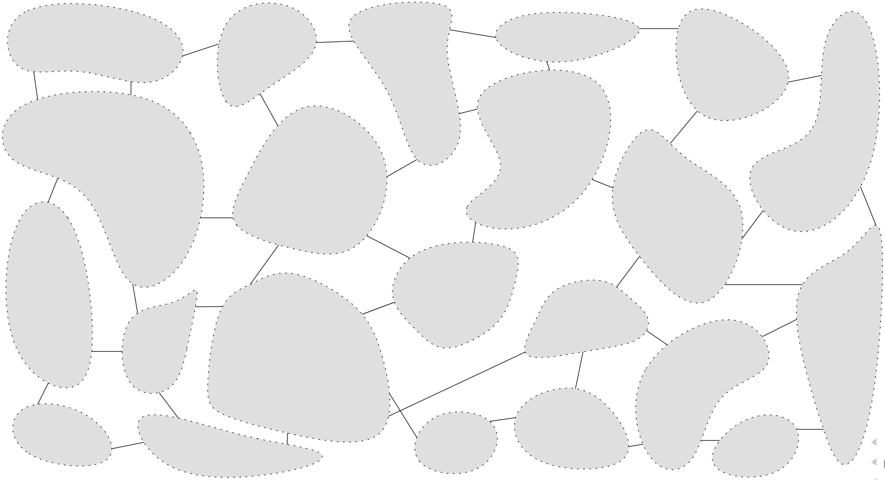


Changes in the graph invalidate:

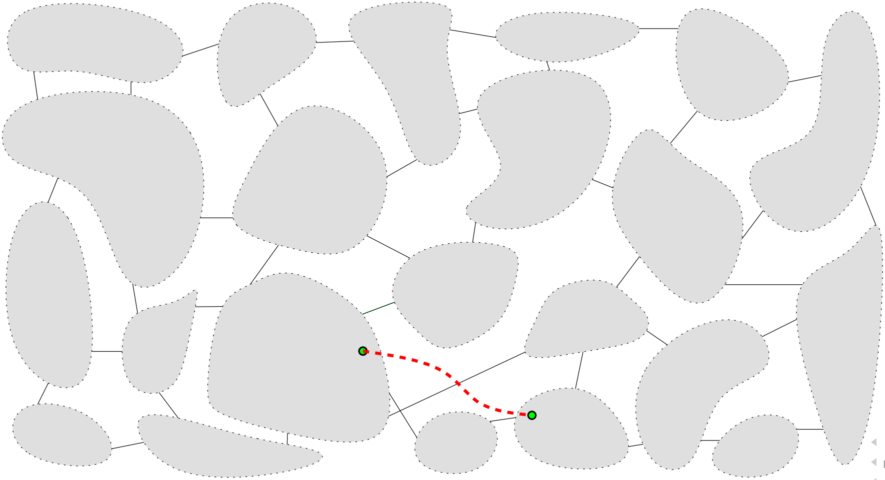
local area: 2-hop neighborhood

⇒ **update of clustering**

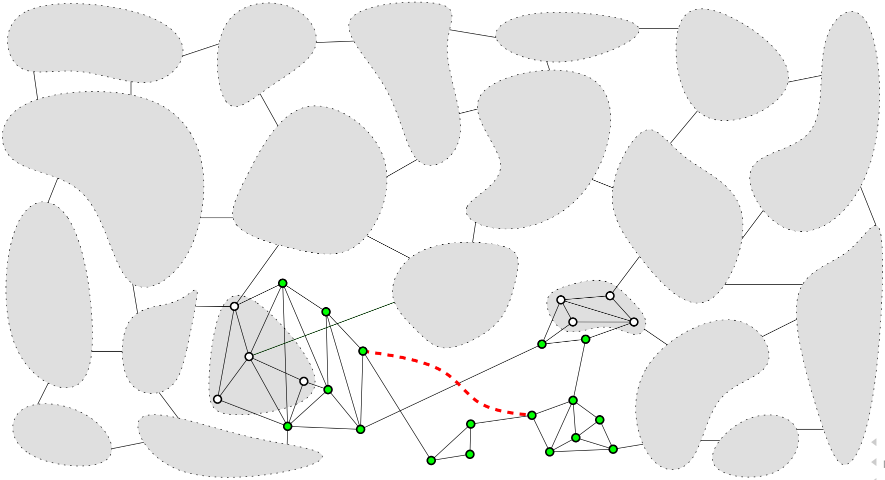
Prep Strategies



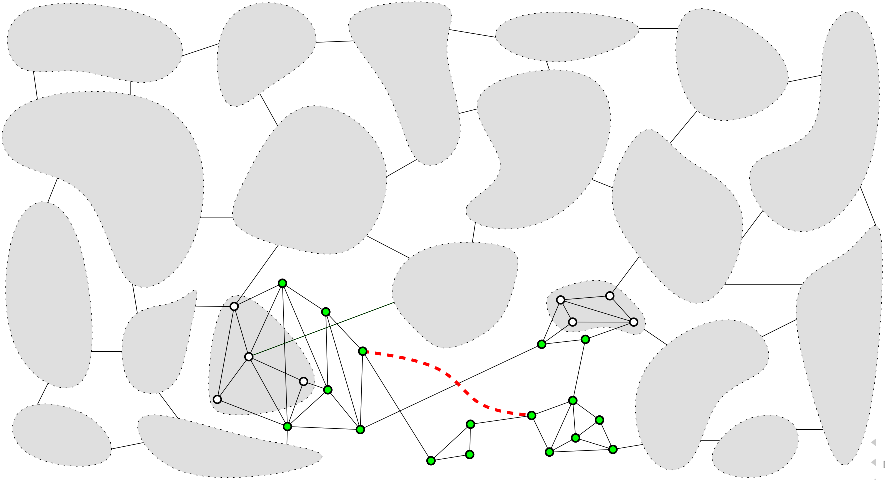
Prep Strategies



Prep Strategies

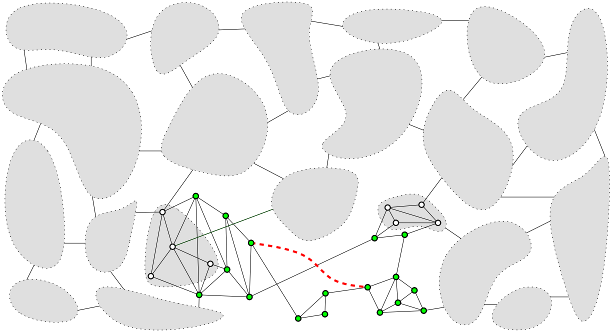


Prep Strategies



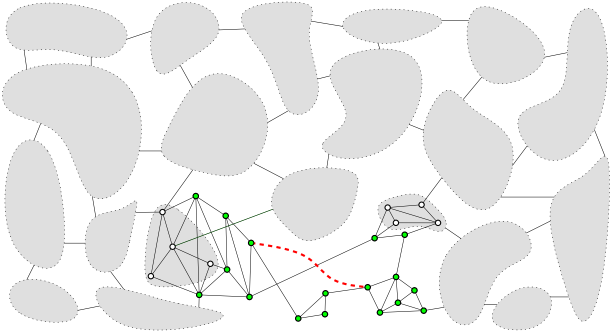
locality assumption

Local Heuristic in Bigger Context



motivation: local changes ⇒ local consequences,
„revolutions“ rare in practice

Local Heuristic in Bigger Context



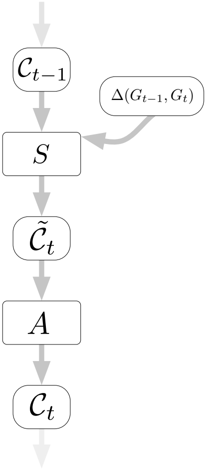
motivation: local changes ⇒ local consequences, „revolutions" rare in practice

hope: small changes ⇒ smooth transitions
small search space ⇒ fast
local optimization ⇒ quality ?

Prep Strategies: Concept

prep strategy S

- reacts to changes
- prepares half-finished *preclustering* $\tilde{\mathcal{C}}$
- passes $\tilde{\mathcal{C}}$ on to algorithm



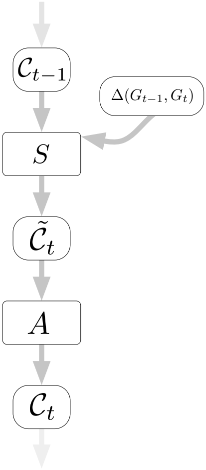
Prep Strategies: Concept

prep strategy S

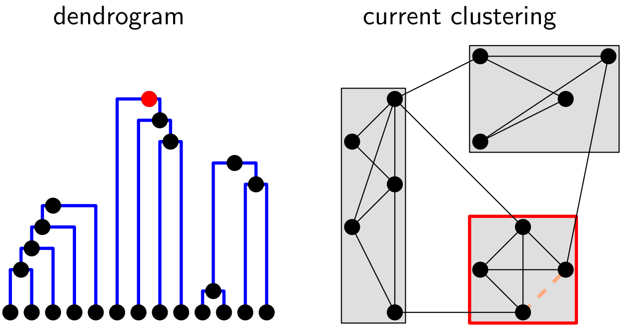
- reacts to changes
- prepares half-finished *preclustering* $\tilde{\mathcal{C}}$
- passes $\tilde{\mathcal{C}}$ on to algorithm

strategies based, e.g., on

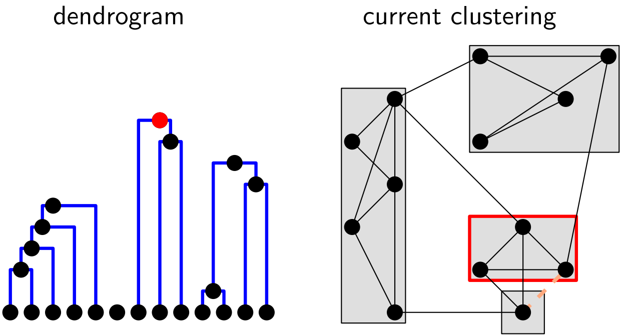
- limited local search
- backtracking the dendrogram



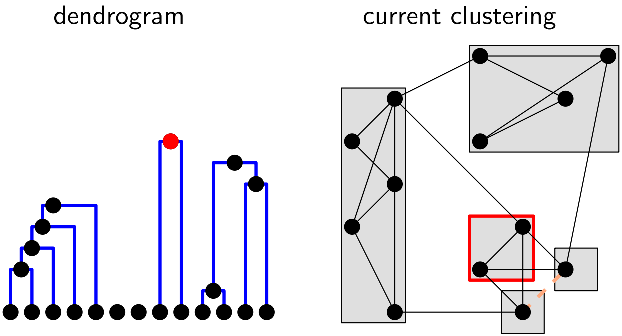
Prep Strategy BT: Illustration



Prep Strategy BT: Illustration



Prep Strategy BT: Illustration



Prep Strategy Backtrack

Backtrack Global's merges according to heuristic rules

Measuring Smoothness

Distance measures based on:

- counting pairs
- maximum overlap
- information theory

[3] [Delling et al.: Engineering comparators for graph clustering, 2008]

Measuring Smoothness

Distance measures based on:

- **counting pairs**
- maximum overlap
- information theory

Example: *Rand*-distance

[3] [Delling et al.: Engineering comparators for graph clustering, 2008]

Measuring Smoothness

Distance measures based on:

- **counting pairs**
- maximum overlap
- information theory

Example: *Rand*-distance

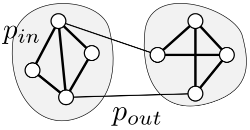
"compare co-classification of all pairs of nodes"

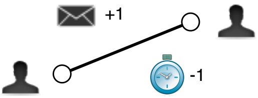
$$\mathcal{R}(\mathcal{C},\mathcal{C}') := \underbrace{1 - \frac{n_{\text{tog,tog}} + n_{\text{sep,sep}}}{\frac{1}{2}|N|(|N|-1)}}_{(\textit{Rand}\text{, set-based})} \rightsquigarrow \underbrace{1 - \frac{e_{\text{tog,tog}} + e_{\text{sep,sep}}}{|E|}}_{(\text{graph-based } \textit{Rand}^{3})}$$

[3] [Delling et al.: Engineering comparators for graph clustering, 2008]

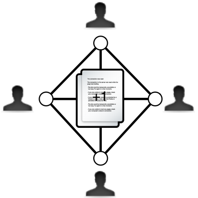
Dynamic Graph Instances

- generator for dynamic clustered random graphs
  [Görke and Staudt: A generator for dynamic clustered random graphs, 2009]



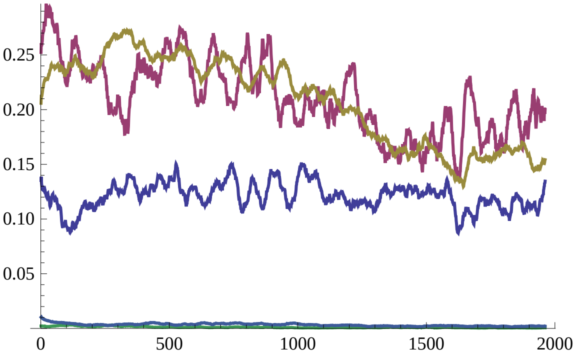


- e-mail graph of KIT CompSci



- arXiv collaboration graph

Dynamic *Modularity*-Clustering: Smooth Transitions



distances between time steps
(graph-based *Rand*-distance)

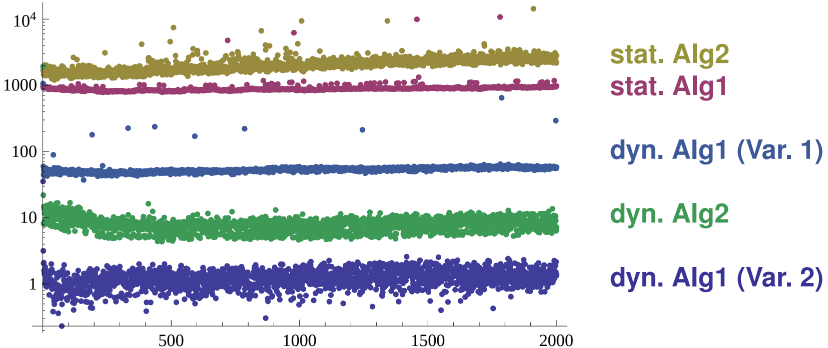
stat. Alg1
stat. Alg2
dyn. Alg1 (Var. 2)

dyn. Alg2 and
dyn. Alg1 (Var. 1)

$\Rightarrow$ Dynamics yield smoother transitions

[Görke et al.: Modularity-driven clustering of dynamic graphs 2010]

Dynamic *Modularity*-Clustering: Runtime

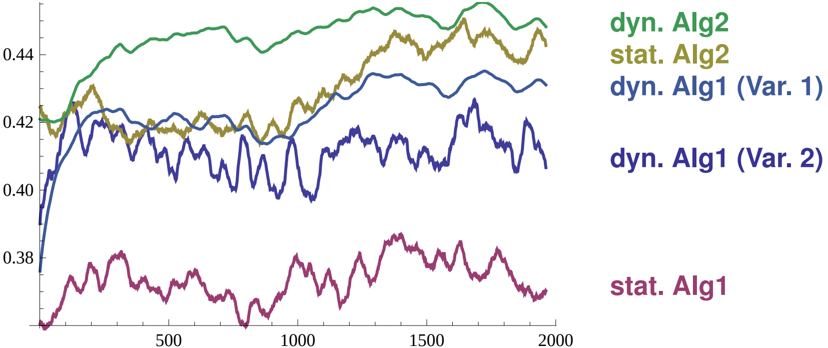


runtime per time step
(milliseconds, logarithmic *y*-axis)

⇒ dynamics yield lower runtimes

[Görke et al.: Modularity-driven clustering of dynamic graphs 2010]

Dynamic *Modularity*-Clustering: Quality

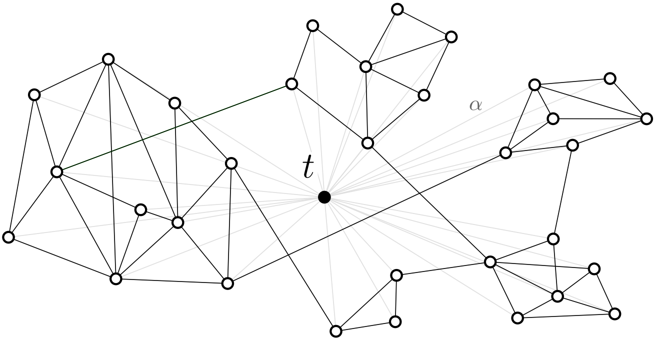


quality over time
(modularity)

⇒ dynamics yield higher quality

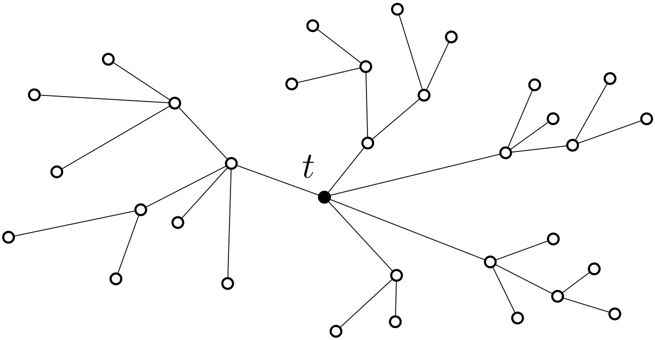
[Görke et al.: Modularity-driven clustering of dynamic graphs 2010]

Min-Cut Tree Clustering (Rep.)



1. original graph
2. star-center *t*, α

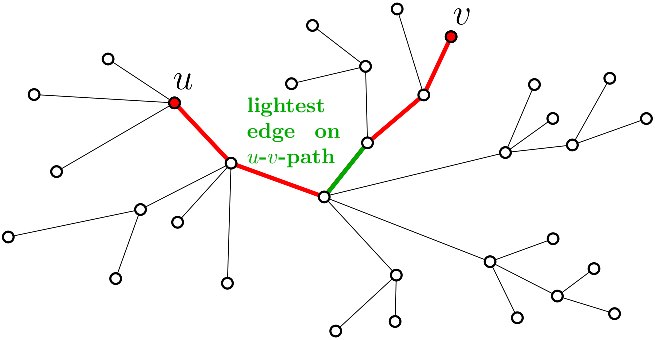
Min-Cut Tree Clustering (Rep.)



1. original graph
2. star-center t, α
3. *min-cut tree*

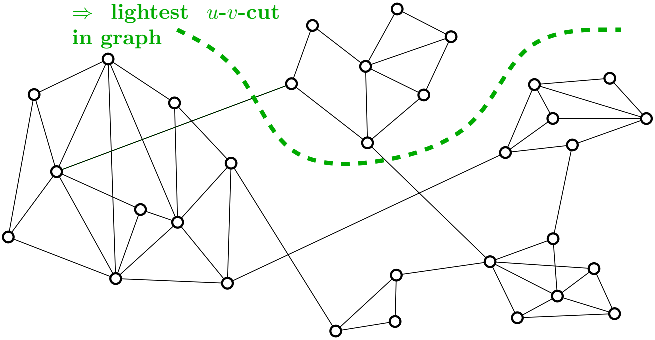
- coined in [Gomory and Hu '61]
- simplified in [Gusfield '90]
- construction via $(n-1)$ max-flows
  (variants e.g. $\widetilde{O}(mn)$ for unweighted)

Min-Cut Tree Clustering (Rep.)



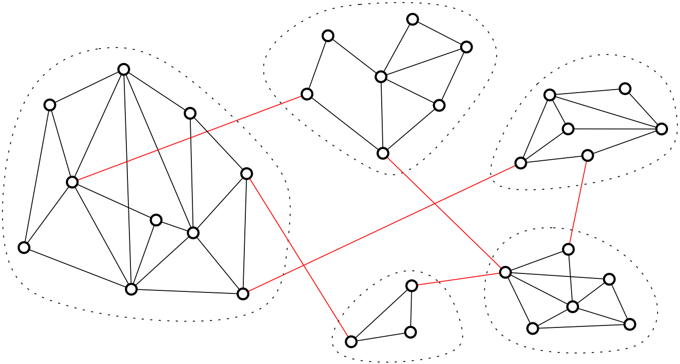
1. original graph
2. star-center *t*, α
3. *min-cut tree*

Min-Cut Tree Clustering (Rep.)



1. original graph
2. star-center t, α
3. *min-cut tree*

Min-Cut Tree Clustering (Rep.)



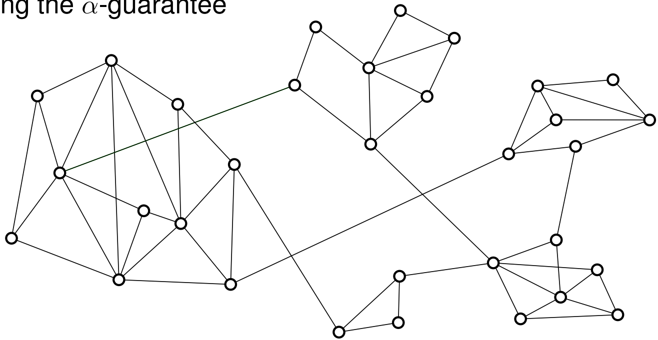
1. original graph
2. star-center t, α
3. *min-cut tree*
4. delete center $\Rightarrow$ clustering

quality guarantee: [Flake et al. '04]

intra-cluster expansion $\geq \alpha \geq$ *inter-cluster expansion**

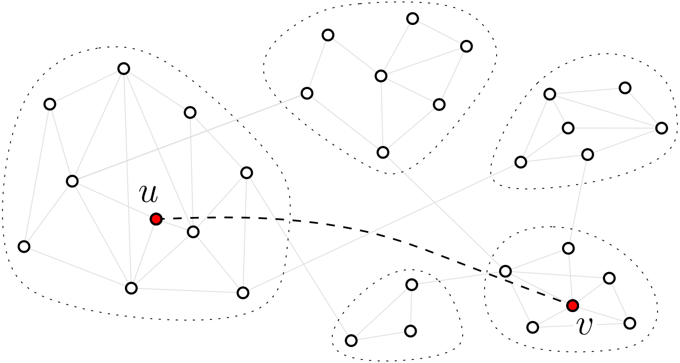
Dynamizing the Clustering

maintaining the α-guarantee



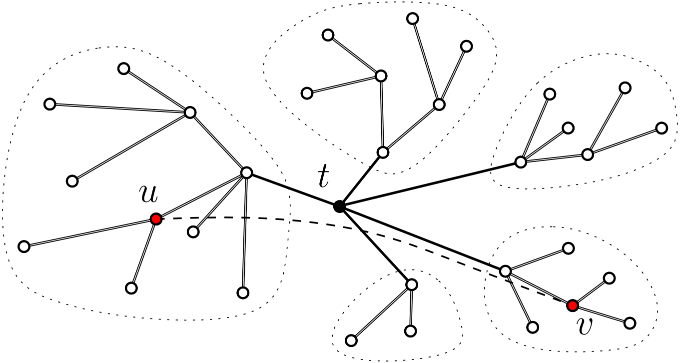
- new edge in graph affects *u*-*v*-path in tree

Dynamizing the Clustering



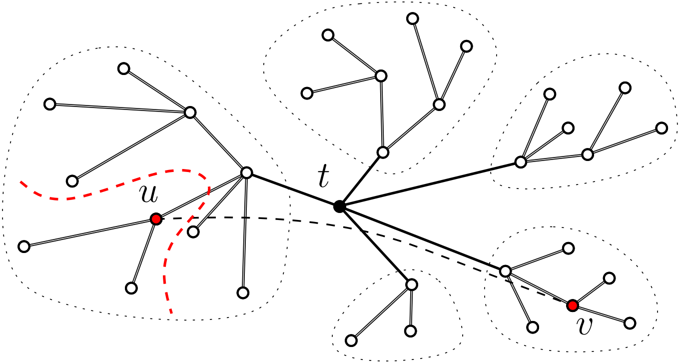
- new edge in graph affects *u-v-path* in tree

Dynamizing the Clustering



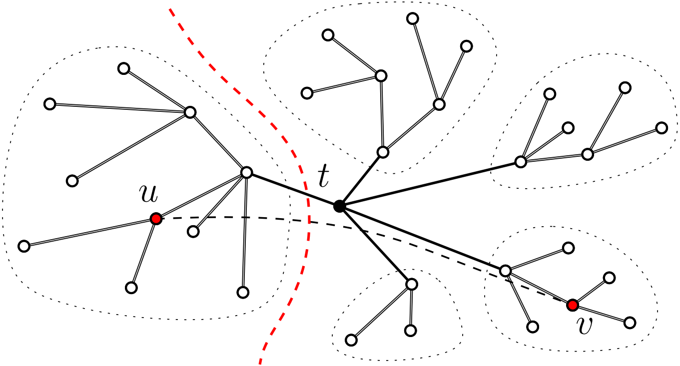
- new edge in graph affects *u*-*v*-path in tree

Dynamizing the Clustering



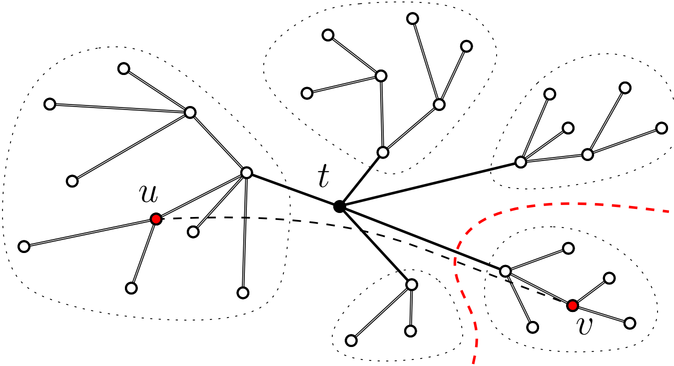
- new edge in graph affects *u-v-path* in tree

Dynamizing the Clustering



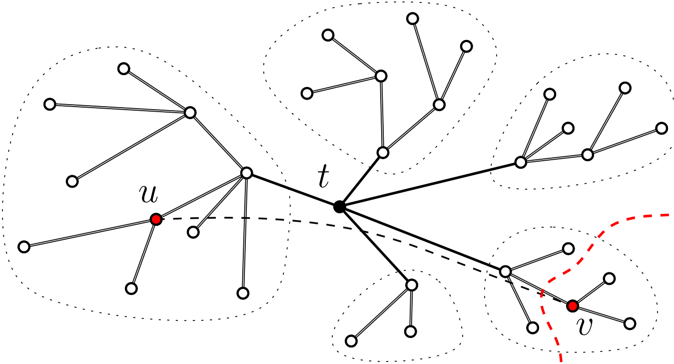
- new edge in graph affects *u*-*v*-path in tree

Dynamizing the Clustering



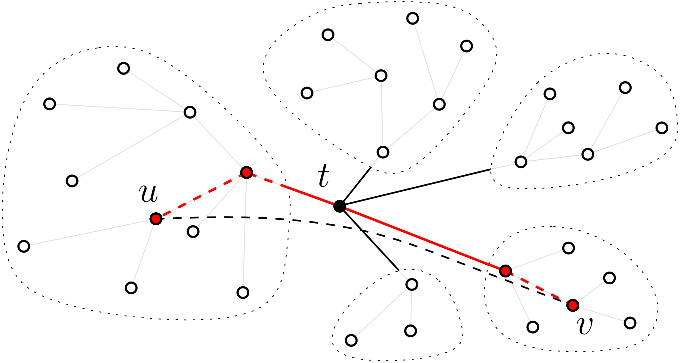
- new edge in graph affects *u*-*v*-path in tree

Dynamizing the Clustering



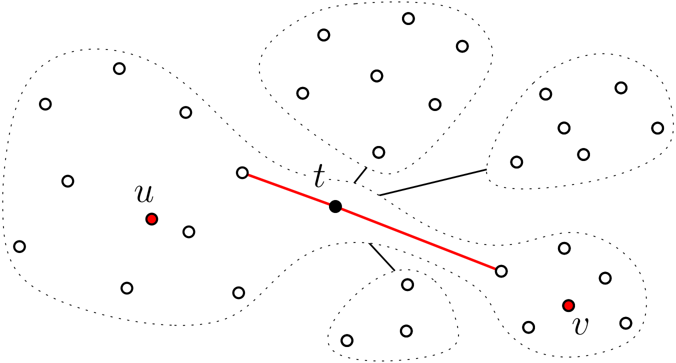
- new edge in graph affects *u*-*v*-path in tree

Dynamizing the Clustering



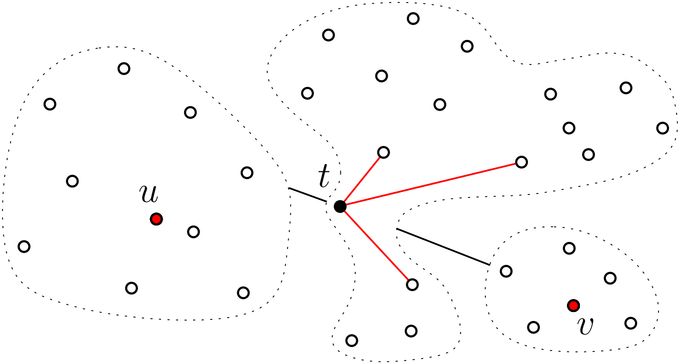
- new edge in graph affects *u*-*v*-path in tree
- safe parts of the clustering reusable

Dynamizing the Clustering



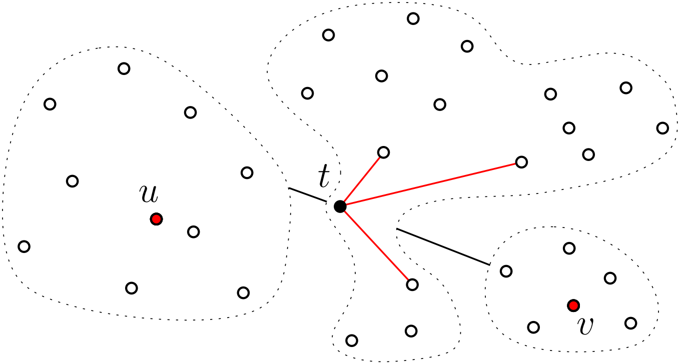
- new edge in graph affects *u*-*v*-path in tree
- safe parts of the clustering reusable
- *Theorem:* “starting here is feasible”

Dynamizing the Clustering



- new edge in graph affects *u*-*v*-path in tree
- safe parts of the clustering reusable
- *Theorem:* “starting here is feasible”
- edge deletion affects off-path cuts analogously

Dynamizing the Clustering

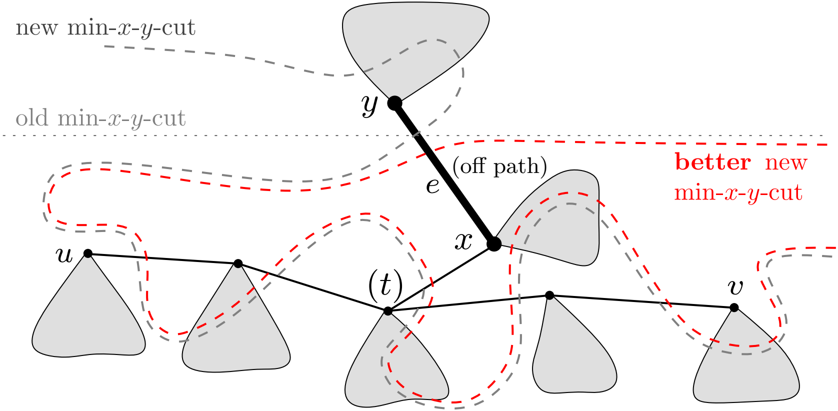


- new edge in graph affects *u*-*v*-path in tree
- safe parts of the clustering reusable
- *Theorem:* “starting here is feasible”
- edge deletion affects off-path cuts analogously

[Hartmann et al.: Dynamic Graph Clustering Using Minimum-Cut Trees, 2009]

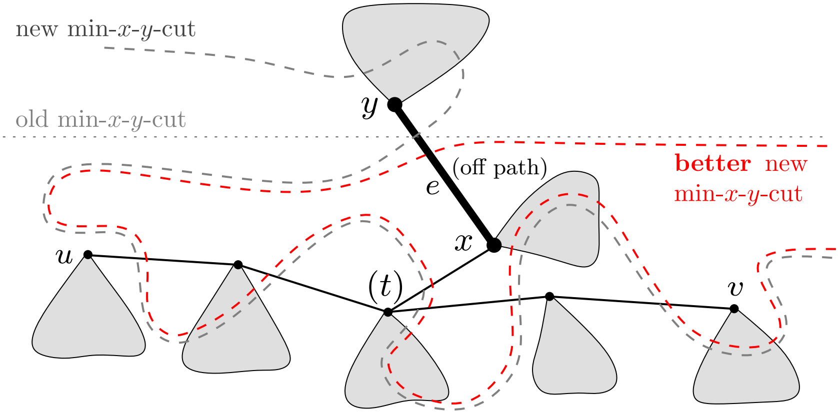
Dynamizing the Clustering

a glimpse into retaining clusters and saving more effort
example: *inter-cluster edge deletion* $\rightsquigarrow$ check/correct old $\mathcal{C}$



Dynamizing the Clustering

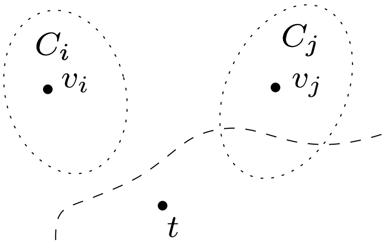
a glimpse into retaining clusters and saving more effort
example: *inter-cluster edge deletion* $\rightsquigarrow$ check/correct old $\mathcal{C}$



Lemma $\Rightarrow$ cut (gray) can be reshaped, retaining old cluster (red)

Dynamizing the Clustering: Case 1

adjustments for all pairs of clusters:

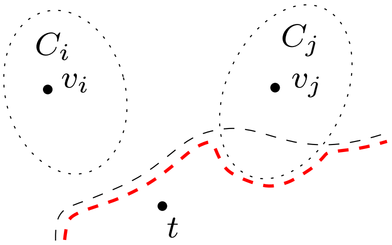


v_i is tree-reference vertex of C_i

check old cluster C_i in new graph $\rightsquigarrow$ new v_i-t-cut separates v_j, t
Lemma (last slide) $\Rightarrow$ cut can be reshaped, retaining C_i (black)

Dynamizing the Clustering: Case 1

adjustments for all pairs of clusters:

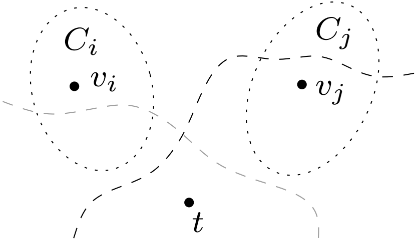


v_i is tree-reference vertex of C_i

check old cluster C_i in new graph $\leadsto$ new v_i-t-cut separates v_j, t
- *Lemma* (last slide) $\Rightarrow$ cut can be reshaped, retaining C_i (black)
- *Lemma* $\Rightarrow$ cut can be re-reshaped, swallowing C_j (red)

Dynamizing the Clustering: Case 2

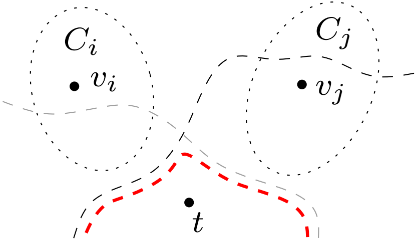
adjustments for all pairs of clusters:



symmetric case: new v_j-t-cut (gray) separates v_i from t, **after** new v_i-t-cut (black) does not separate v_j from t

Dynamizing the Clustering: Case 2

adjustments for all pairs of clusters:

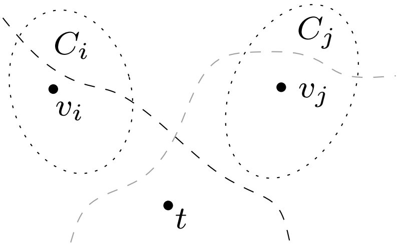


symmetric case: new v_j-t-cut (gray) separates v_i from t, **after** new v_i-t-cut (black) does not separate v_j from t

Lemma $\Rightarrow$ reshape cut (red)

Dynamizing the Clustering: Case 3

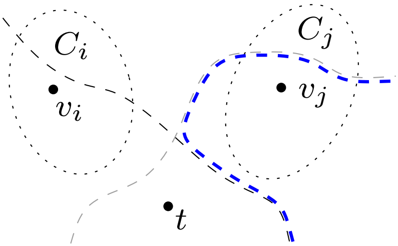
adjustments for all pairs of clusters:



both new cuts do not separate other vertex from t

Dynamizing the Clustering: Case 3

adjustments for all pairs of clusters:

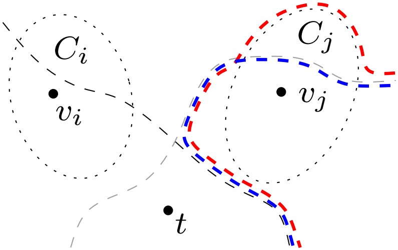


both new cuts do not separate other vertex from t

folklore $\Rightarrow$ crossing min-cuts $\rightsquigarrow$ non-crossing (blue for v_j's cut)

Dynamizing the Clustering: Case 3

adjustments for all pairs of clusters:



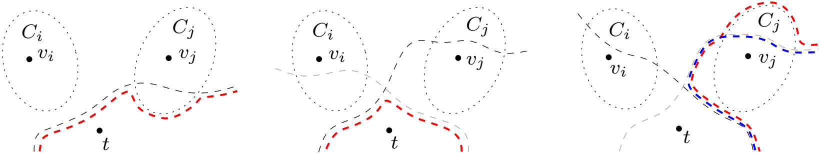
both new cuts do not separate other vertex from t

folklore $\Rightarrow$ crossing min-cuts $\rightsquigarrow$ non-crossing (blue for v_j's cut)

Lemma $\Rightarrow$ reshape to retain old cluster (red for v_j's cut)

Dynamizing the Clustering

adjustments for all pairs of clusters:



- adjust cuts to all other clusters C_j
- old cuts retained or swallowed
- order affects speed, but lemmata hold for all orders
- similar arguments for intra-/inter-cluster addition/deletion

$\Rightarrow$ **smoothness + speed**

Results of the Dynamization

instance: email network, 12 000 time steps
measurement: number of s-t-cut calculations (vs. static)

runtime saved $> 90\%$

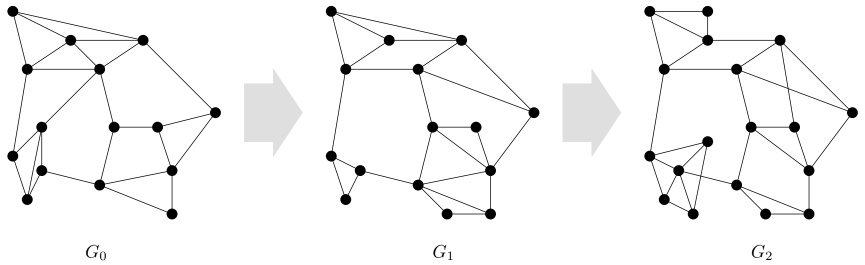
smooth transitions: minimal changes per step (max. 2 cluster split)

continuity: old clustering valid? $\Rightarrow$ confirmed via only 2 s-t-cuts

quality: $\surd$

Structure

Clustering Graph Sequences

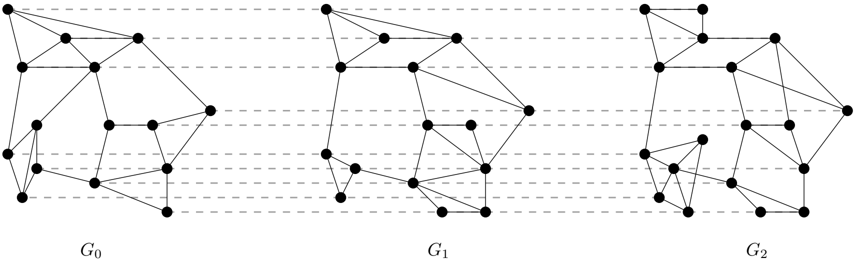


Goals:
1. a clustering of each step
2. tracking clusters
3. smooth transitions
4. find critical changes

naïve approach: comparison of unrelated, static clusterings

[4] [Gaertler et al.: How to cluster an evolving graph, 2006]

Clustering Graph Sequences

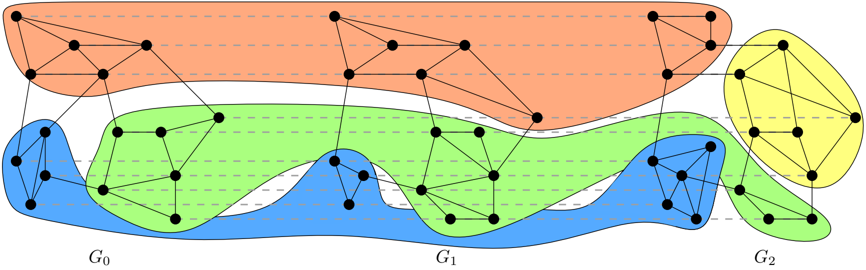


Goals:
1. a clustering of each step
2. tracking clusters
3. smooth transitions
4. find critical changes

naïve approach: comparison of unrelated, static clusterings
time-expanded clustering[4]: stack snapshots

[4] [Gaertler et al.: How to cluster an evolving graph, 2006]

Clustering Graph Sequences



Goals:

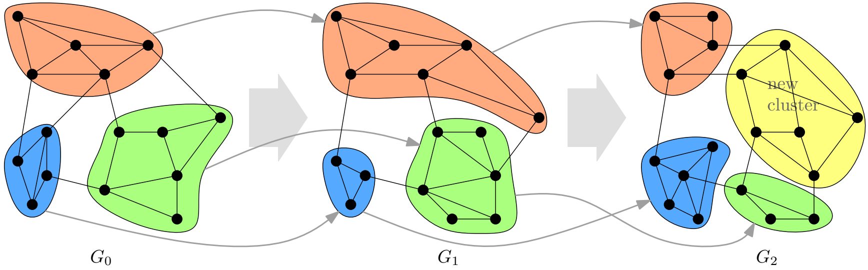
1. a clustering of each step
2. tracking clusters
3. smooth transitions
4. find critical changes

naïve approach: comparison of unrelated, static clusterings

time-expanded clustering[4]: stack snapshots ⇒ cluster

[4] [Gaertler et al.: How to cluster an evolving graph, 2006]

Clustering Graph Sequences

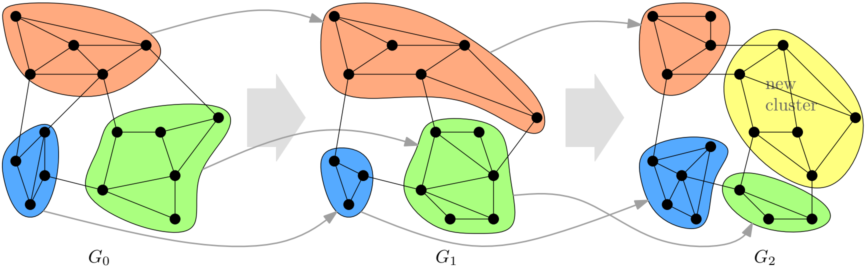


Goals:
1. a clustering of each step
2. tracking clusters
3. smooth transitions
4. find critical changes

naïve approach: comparison of unrelated, static clusterings
time-expanded clustering[4]: stack snapshots ⇒ cluster ⇒ evolution visible

[4] [Gaertler et al.: How to cluster an evolving graph, 2006]

Clustering Graph Sequences



Goals:
1. a clustering of each step
2. tracking clusters
3. smooth transitions
4. find critical changes

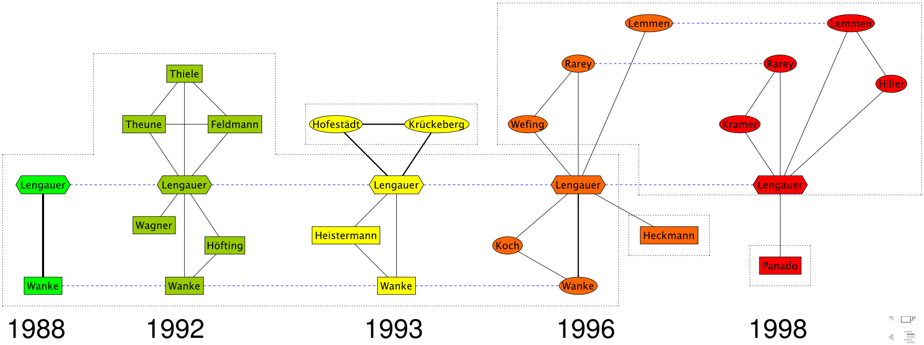
applications:
- email-network
- customer profiles
- technological trends

naïve approach: comparison of unrelated, static clusterings
time-expanded clustering[4]: stack snapshots ⇒ cluster ⇒ evolution visible

[4] [Gaertler et al.: How to cluster an evolving graph, 2006]

Time-Expanded Graph Clustering

- **Thomas Lengauer** started career as a *computer scientist* (rectangles)
- cooperations with *biologists* (ellipses)
- . . . today a renown bioinformatician



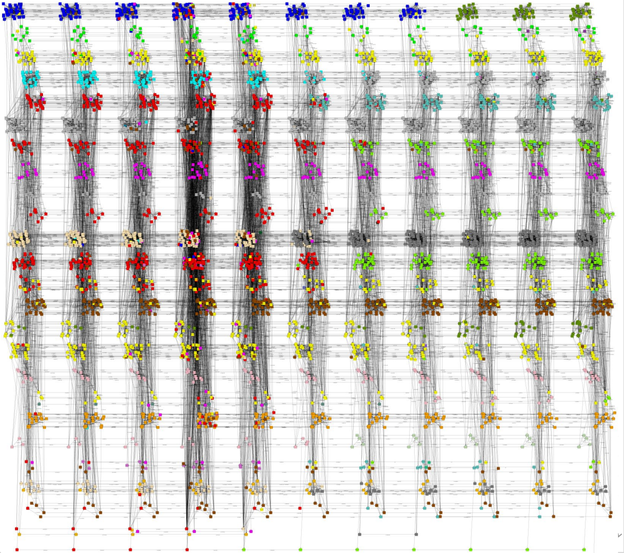
Application: Email-Network [Görke: Diss. 2010]

email data (KIT's CompSci)

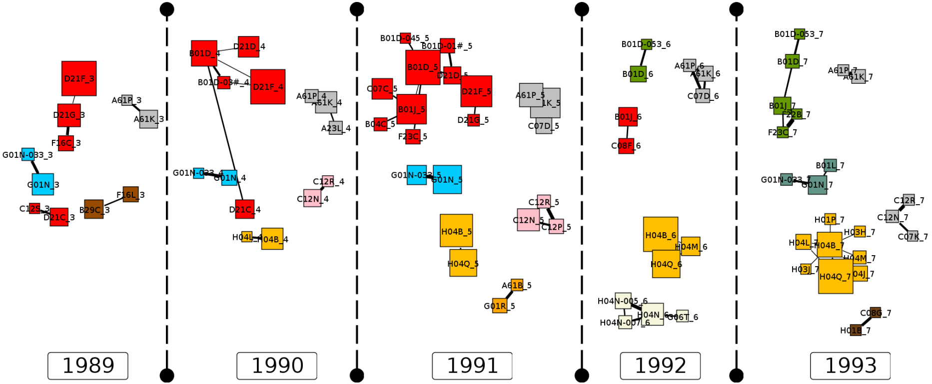
- nodes = employee
- edge = emails
- 36 months collected

- vertical str. = month
- horiz. str. = chair
- color = clustering

⇒ colors in vertical strip = current cooperations



Application: Technological Trends



Patent Categories, Finland, interconnected via multi-category patents

break: from 1992 on telecommunications dominate
before: paper, pharmaceuticals, material sciences
later: paper forms new cluster

Explicitly Bicriterial Formulations

1. Among all sequences $\zeta = (\mathcal{C}_0, \ldots, \mathcal{C}_{t_{\max}})$ of clusterings of $\mathcal{G}$ with $\mathcal{C}_i(G_i)$ optimal regarding quality, find the sequence ζ_{smooth} that minimizes $\sum_{i=1}^{t_{\max}} \text{distance}(\mathcal{C}_{i-1}, \mathcal{C}_i)$.

2. Among all sequences $\zeta = (\mathcal{C}_0, \ldots, \mathcal{C}_{t_{\max}})$ of clusterings of $\mathcal{G}$ with $\sum_{i=1}^{t_{\max}} \text{distance}(\mathcal{C}_{i-1}, \mathcal{C}_i) \leq D$, find the sequence ζ_{good} that maximizes $\sum_{i=0}^{t_{\max}} \text{quality}(\mathcal{C}_i)$.

3. Is there a sequence $\zeta = (\mathcal{C}_0, \ldots, \mathcal{C}_{t_{\max}})$ of clusterings of $\mathcal{G}$ such that $\forall i : \text{quality}(\mathcal{C}_i) \geq \alpha(\mathcal{C}_i^{\text{optimal}})$ and $\forall i \geq 1 : \text{distance}(\mathcal{C}_{i-1}, \mathcal{C}_i) \leq \beta$?

4. Among all sequences $\zeta = (\mathcal{C}_0, \ldots, \mathcal{C}_{t_{\max}})$ of clusterings of $\mathcal{G}$ find the sequence ζ_{best} which optimizes $\alpha \sum_{i=0}^{t_{\max}} \text{quality}(\mathcal{C}_i) + \beta \sum_{i=1}^{t_{\max}} \text{distance}(\mathcal{C}_{i-1}, \mathcal{C}_i)$.

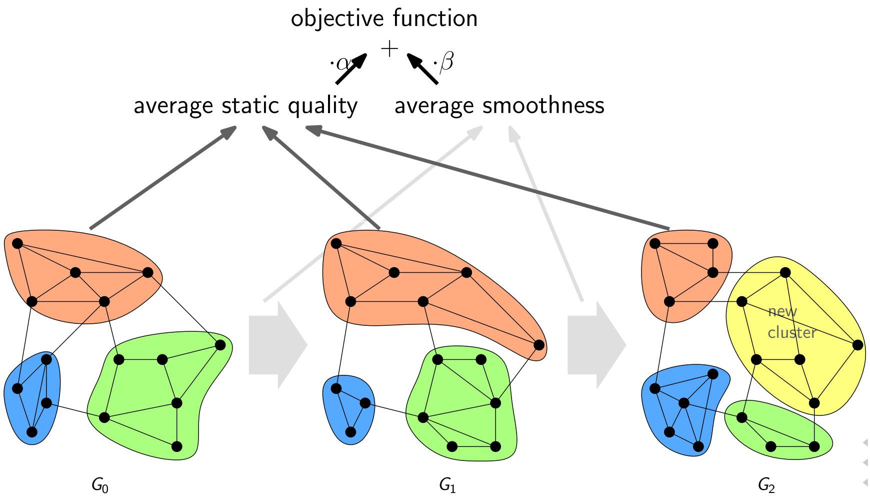
5. ...

[Görke: An algorithmic walk from static to dynamic graph clustering, 2010]

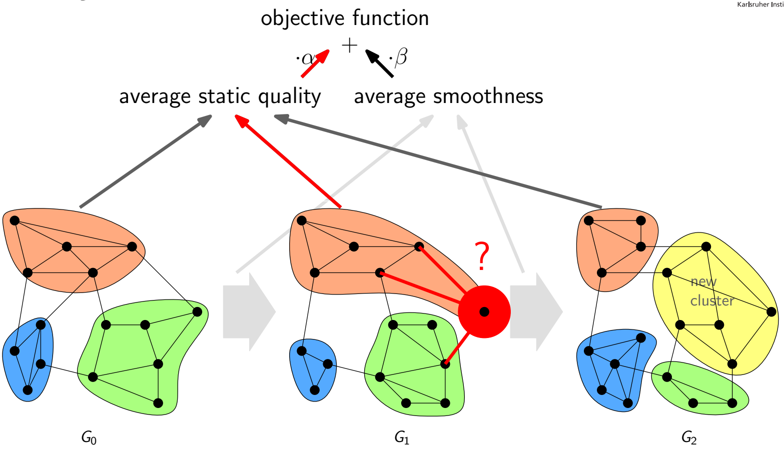
Explicitly Bicriterial ILP (generic)

- set up a static ILP for each time step
- link "neighboring" variables $X^t_{u,v}$ and $X^{t+1}_{u,v}$,
  e.g., $Z^t_{u,v} = X^t_{u,v}$ XOR $X^{t+1}_{u,v}$
- formulate smoothness by Z-variables
  e.g., Rand index: $\mathrm{dist}(\mathcal{C}_t, \mathcal{C}_{t+1}) := 1 - \frac{\sum_{u<v}(1-Z_{uv}(t))}{\frac{1}{2}n(n-1)}$
- use quality / smoothness as constraint / objective
  (some formulations do need quite some thought)
- overview of many variables, constraints and objectives:
  [Schumm et al.: Density-Constrained Graph Clustering, 2010, full technical report version]

Sum of Quality and Smoothness



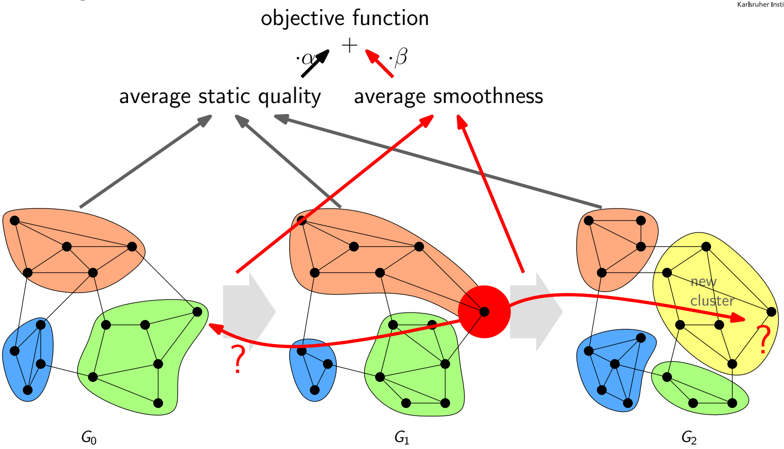
Explicitly Bicriterial Heuristic



Using, e.g., a local greedy heuristic, for each vertex move:

- calculate gain in snapshot quality

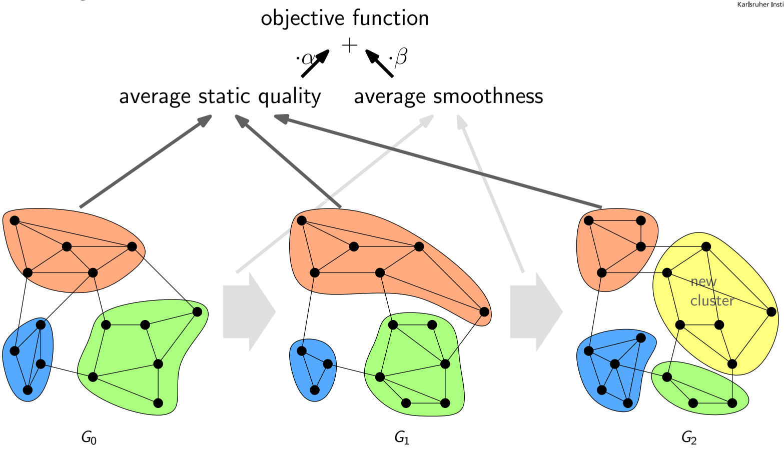
Explicitly Bicriterial Heuristic



Using, e.g., a local greedy heuristic, for each vertex move:

- calculate gain in snapshot quality
- calculate gain in smoothness

Explicitly Bicriterial Heuristic

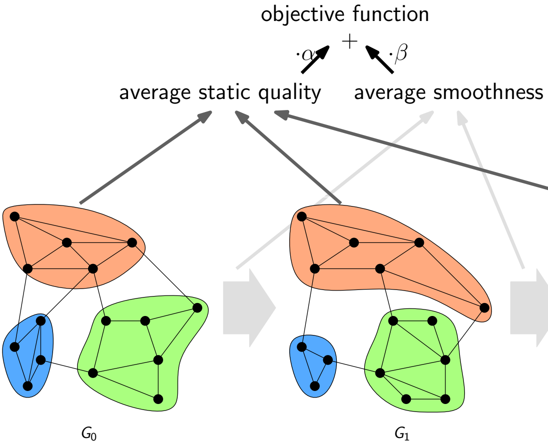


Using, e.g., a local greedy heuristic, for each vertex move:

- calculate gain in snapshot quality
- calculate gain in smoothness

complications: (non-)sequential, contraction, refinement, being efficient

Explicitly Bicriterial Heuristic



[Lisowski: Modularity-basiertes Clustern von dynamischen Graphen im Offline-Fall, 2011]

Using, e.g., a local greedy heuristic, for each vertex move:

- calculate gain in snapshot quality
- calculate gain in smoothness

complications: (non-)sequential, contraction, refinement, being efficient

Selected Summary of Lecture

Objective Functions

a formal foundation for the informal paradigm of *intra-cluster density and inter-cluster sparsity*

Selected Summary of Lecture

KIT
Karlsruher Institut für Technologie

Objective Functions

a formal foundation for the informal paradigm of *intra-cluster density and inter-cluster sparsity*

Algorithmic Approaches

focus: interplay of measure and greedy merge

Selected Summary of Lecture

Objective Functions

a formal foundation for the informal paradigm of *intra-cluster density and inter-cluster sparsity*

Algorithmic Approaches

focus: interplay of measure and greedy merge

Experiments

- designing experiments properly
- test data in algorithm engineering

Selected Summary of Lecture

Objective Functions

a formal foundation for the informal paradigm of *intra-cluster density and inter-cluster sparsity*

Algorithmic Approaches

focus: interplay of measure and greedy merge

Experiments

- designing experiments properly
- test data in algorithm engineering

Comparing Clusterings

- crucial topic in clustering
- showcase for experimentation

Selected Summary of Lecture

Objective Functions

a formal foundation for the informal paradigm of *intra-cluster density and inter-cluster sparsity*

Algorithmic Approaches

focus: interplay of measure and greedy merge

Experiments

- designing experiments properly
- test data in algorithm engineering

Comparing Clusterings

- crucial topic in clustering
- showcase for experimentation

Dynamic Clustering

- formalization of aims
- online and offline methods

Structure

Core-Decomposition: Algorithm

Karlsruher Institut für Technologie

Given: graph G
Wanted: *coreness* of each vertex

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

compute degrees of all vertices
bin-sort vertices into bins $B_0 \ldots B_{\text{max deg}}$
foreach $v \in V$ *in sorted order* **do**
    core(v) $\leftarrow$ deg(v)
    foreach $u \in N(v)$ **do**
        if deg(u) $>$ deg(v) **then**
            deg(u) $\leftarrow$ deg(u) $- 1$
            move u to end of its preceding bin

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

```
compute degrees of all vertices                      // O(max{m, n})
bin-sort vertices into bins B_0 ... B_max deg
foreach v ∈ V in sorted order do
    core(v) ← deg(v)
    foreach u ∈ N(v) do
        if deg(u) > deg(v) then
            deg(u) ← deg(u) − 1
            move u to end of its preceding bin
```

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

```
compute degrees of all vertices                          // O(max{m, n})
bin-sort vertices into bins B_0 ... B_max deg            // O(n)
foreach v ∈ V in sorted order do
    core(v) ← deg(v)
    foreach u ∈ N(v) do
        if deg(u) > deg(v) then
            deg(u) ← deg(u) − 1
            move u to end of its preceding bin
```

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

compute degrees of all vertices // $O(\max\{m, n\})$
bin-sort vertices into bins $B_0 \dots B_{\max \deg}$ // $O(n)$
foreach $v \in V$ *in sorted order* **do** // n times
    $\text{core}(v) \leftarrow \deg(v)$
    foreach $u \in N(v)$ **do**
        if $\deg(u) > \deg(v)$ **then**
            $\deg(u) \leftarrow \deg(u) - 1$
            move u to end of its preceding bin

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

```
compute degrees of all vertices                         // O(max{m, n})
bin-sort vertices into bins B_0 ... B_{max deg}         // O(n)
foreach v ∈ V in sorted order do                        // n times
    core(v) ← deg(v)                                    // O(1)
    foreach u ∈ N(v) do
        if deg(u) > deg(v) then
            deg(u) ← deg(u) − 1
            move u to end of its preceding bin
```

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

compute degrees of all vertices // $O(\max\{m, n\})$
bin-sort vertices into bins $B_0 \dots B_{\text{max deg}}$ // $O(n)$
foreach $v \in V$ *in sorted order* **do** // n times
  core(v) $\leftarrow$ deg(v) // $O(1)$
  foreach $u \in N(v)$ **do** // amortized m times
    if deg(u) $>$ deg(v) **then**
      deg(u) $\leftarrow$ deg(u) $- 1$
      move u to end of its preceding bin

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

```
compute degrees of all vertices                      // O(max{m, n})
bin-sort vertices into bins B_0 ... B_max deg        // O(n)
foreach v ∈ V in sorted order do                     // n times
    core(v) ← deg(v)                                 // O(1)
    foreach u ∈ N(v) do                              // amortized m times
        if deg(u) > deg(v) then                      // O(1)
            deg(u) ← deg(u) − 1
            move u to end of its preceding bin
```

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

compute degrees of all vertices // $O(\max\{m, n\})$
bin-sort vertices into bins $B_0 \dots B_{\text{max deg}}$ // $O(n)$
foreach $v \in V$ *in sorted order* **do** // n times
  $\text{core}(v) \leftarrow \text{deg}(v)$ // $O(1)$
  foreach $u \in N(v)$ **do** // amortized m times
    if $\text{deg}(u) > \text{deg}(v)$ **then** // $O(1)$
      $\text{deg}(u) \leftarrow \text{deg}(u) - 1$ // $O(1)$
      move u to end of its preceding bin

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

---

compute degrees of all vertices // $O(\max\{m, n\})$
bin-sort vertices into bins $B_0 \dots B_{\text{max deg}}$ // $O(n)$
foreach $v \in V$ *in sorted order* **do** // n times
    core$(v) \leftarrow$ deg(v) // $O(1)$
    foreach $u \in N(v)$ **do** // amortized m times
        if deg$(u) >$ deg(v) **then** // $O(1)$
            deg$(u) \leftarrow$ deg$(u) - 1$ // $O(1)$
            move u to end of its preceding bin // $O(1)$, needs care

---

Core-Decomposition: Algorithm

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

```
compute degrees of all vertices                       // O(max{m, n})
bin-sort vertices into bins B_0 ... B_max deg         // O(n)
foreach v ∈ V in sorted order do                      // n times
    core(v) ← deg(v)                                  // O(1)
    foreach u ∈ N(v) do                               // amortized m times
        if deg(u) > deg(v) then                       // O(1)
            deg(u) ← deg(u) − 1                       // O(1)
            move u to end of its preceding bin        // O(1), needs care
```

Complexity: $O(\max\{m, n\})$

Core-Decomposition: Algorithm

Karlsruher Institut für Technologie

Given: graph G
Wanted: *coreness* of each vertex

Algorithm: `Core-Decomposition`

```
compute degrees of all vertices                        // O(max{m, n})
bin-sort vertices into bins B_0 ... B_max deg          // O(n)
foreach v ∈ V in sorted order do                       // n times
    core(v) ← deg(v)                                   // O(1)
    foreach u ∈ N(v) do                                // amortized m times
        if deg(u) > deg(v) then                        // O(1)
            deg(u) ← deg(u) − 1                        // O(1)
            move u to end of its preceding bin         // O(1), needs care
```

Complexity: $O(\max\{m, n\})$

see, e.g., [Batagelj, Zaveršnik '02, *An O(m) Algorithm for Cores Decomposition of Networks]*
or [Brandes, Erlebach (eds.) '05, Network Analysis, Methodological Foundations]