Mining Networks

Chapter 5 — Community Detection

Bogumił Kamiński, Paweł Prałat, and François Théberge

Updated: 2025/03/04

Department of Mathematics, Toronto Metropolitan University

File: DS8014-Chapter05

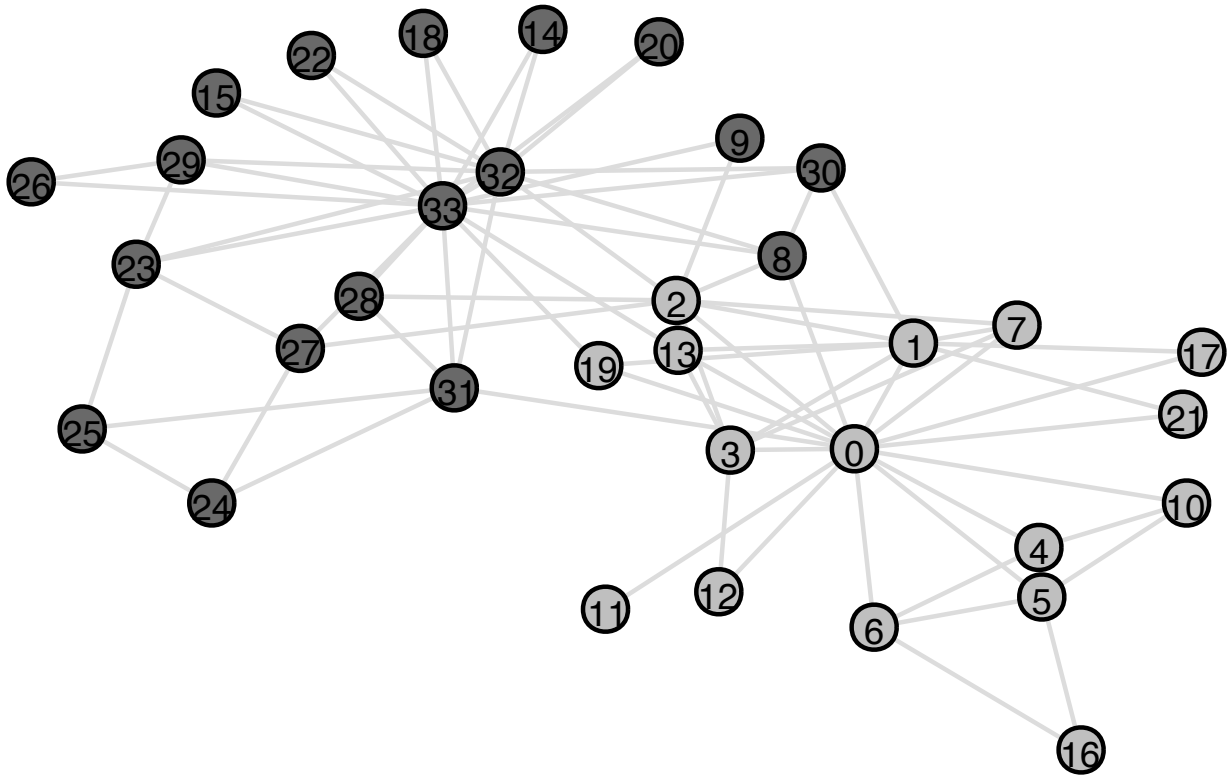
Toronto Metropolitan University

Overview

1. Basic Properties of Communities
2. Synthetic Models with Community Structure
3. Graph Modularity
4. Hierarchical Clustering
5. A Few Other Methods
6. Experiments

Introduction

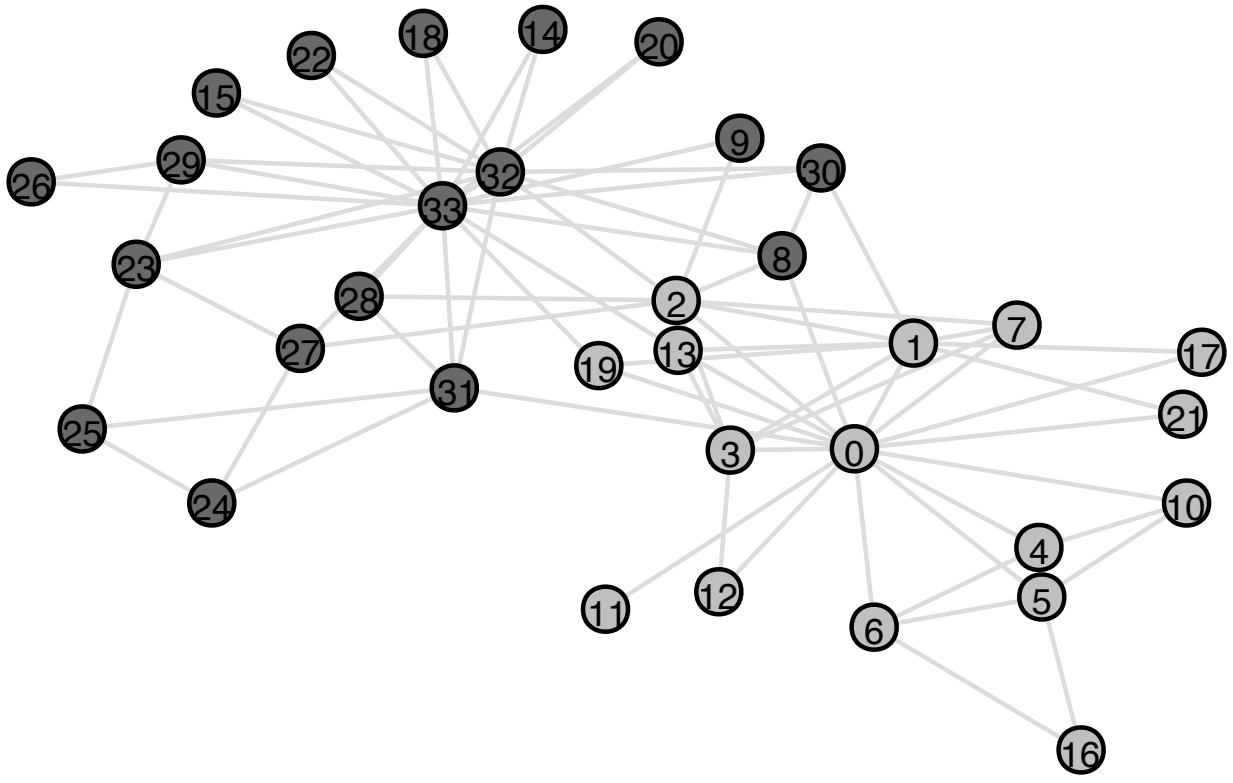
Zachary's Karate Club: 0 – instructor, 33 – administrator



A network has community structure if its set of nodes can be split into a number of subsets such that each subset is densely internally connected.

Introduction

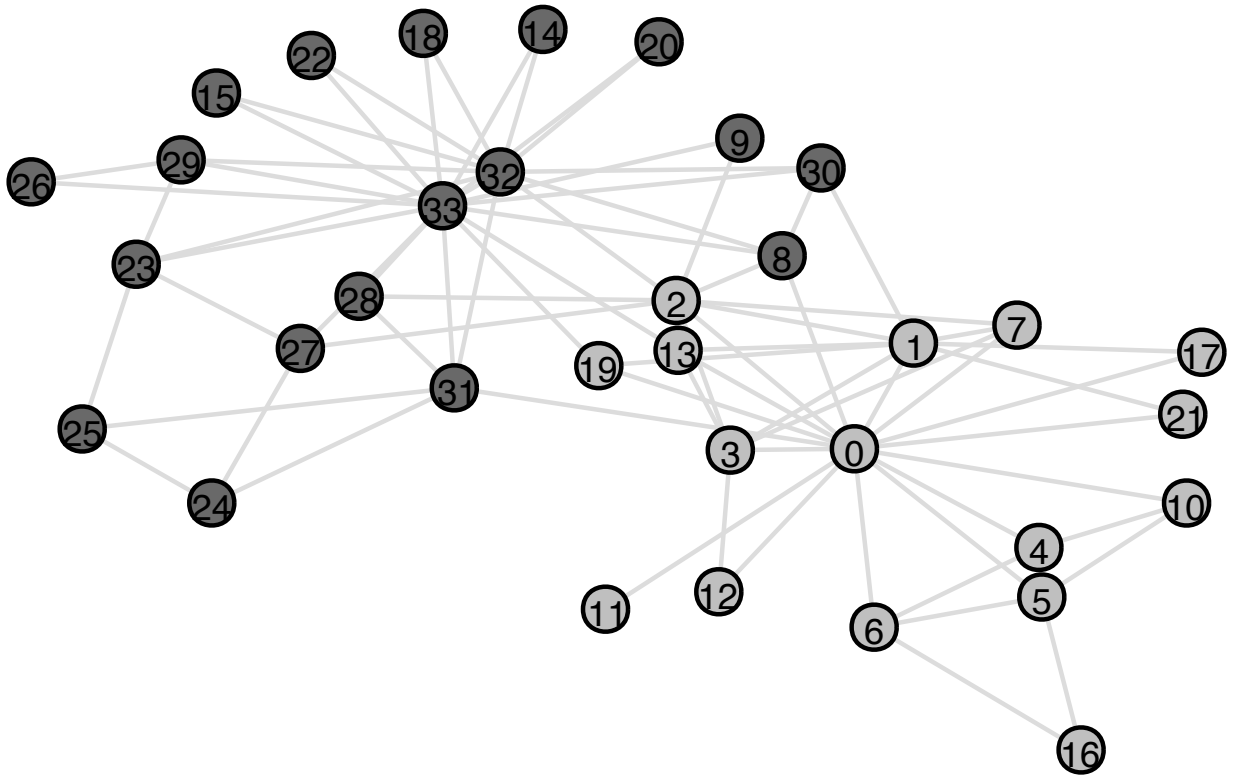
Zachary's Karate Club: 0 – instructor, 33 – administrator



social networks: communities based on common location of their users, their interests, occupation, gender, age, etc.

Introduction

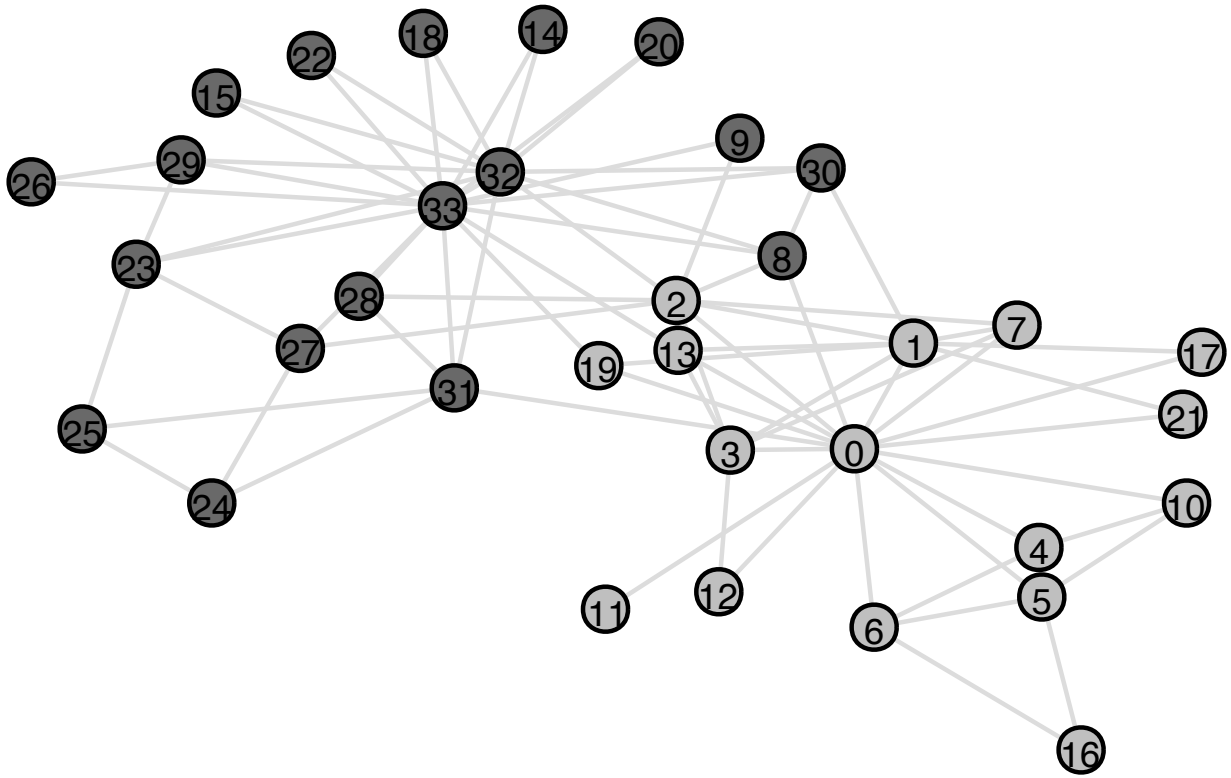
Zachary's Karate Club: 0 – instructor, 33 – administrator



web graph: web pages that belong to the same community are on a similar topic.

Introduction

Zachary's Karate Club: 0 – instructor, 33 – administrator



protein-protein interaction networks: proteins that belong to the same community are often associated with a particular biological function within the organizm.

Introduction

Finding the right partition that represents the community structure is a challenging but important problem for a number of reasons.

Introduction

Finding the right partition that represents the community structure is a challenging but important problem for a number of reasons.

Communities allow us to...
– see a "big picture" (large scale map with individual communities represented as meta-nodes),

Introduction

Finding the right partition that represents the community structure is a challenging but important problem for a number of reasons.

Communities allow us to...
– see a "big picture" (large scale map with individual communities represented as meta-nodes),
– better understand the function of the system represented by the network,

Introduction

Finding the right partition that represents the community structure is a challenging but important problem for a number of reasons.

Communities allow us to...
– see a "big picture" (large scale map with individual communities represented as meta-nodes),
– better understand the function of the system represented by the network,
– classify the nodes based on the position they have in their own clusters and how they are connected to other clusters: roles and importance,
– ...

Basic Properties of Communities

Ground Truth

Collaboration network:

nodes – researchers, edges – collaborations.

Ground Truth

Collaboration network:
nodes – researchers, edges – collaborations.

Mathematicians are interested in doing mathematics and, as a result, tend to work together forming community of mathematicians (there are more edges between mathematicians than between mathematicians and biologists).

Ground Truth

Collaboration network:
nodes – researchers, edges – collaborations.

Mathematicians are interested in doing mathematics and, as a result, tend to work together forming community of mathematicians (there are more edges between mathematicians than between mathematicians and biologists).

We do not know where are mathematicians (ground truth) but we want to uncover this hidden, underlying attributes of nodes based on a graph that is visible to us—reversed engineering process.

Ground Truth

It is extremely rare but sometimes we do have access to ground truth (Zachary's Karate Club).

Ground Truth

It is extremely rare but sometimes we do have access to ground truth (Zachary's Karate Club).

There is need for synthetic networks with community structure (in order to tune community detection algorithms and to benchmark them in a fair and rigorous way).

Ground Truth

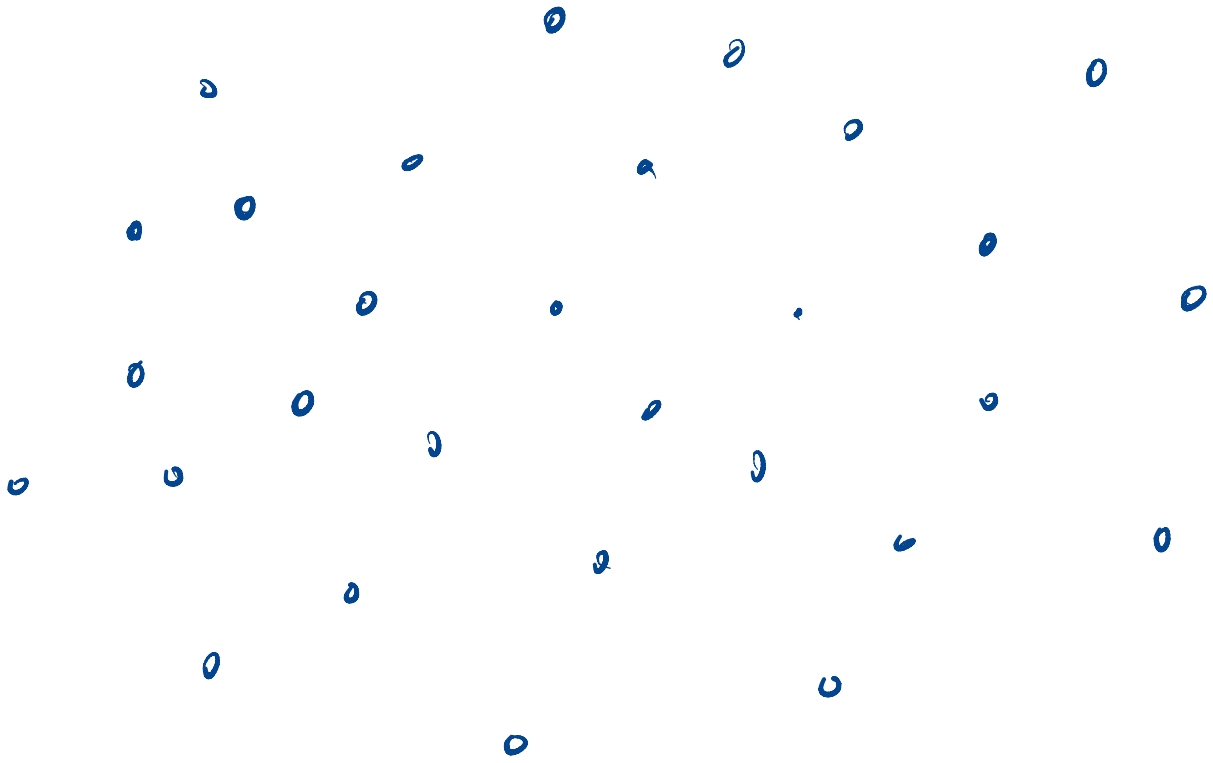
The ground truth could simply be a partition of nodes (the number of parts can be arbitrary).

Partition

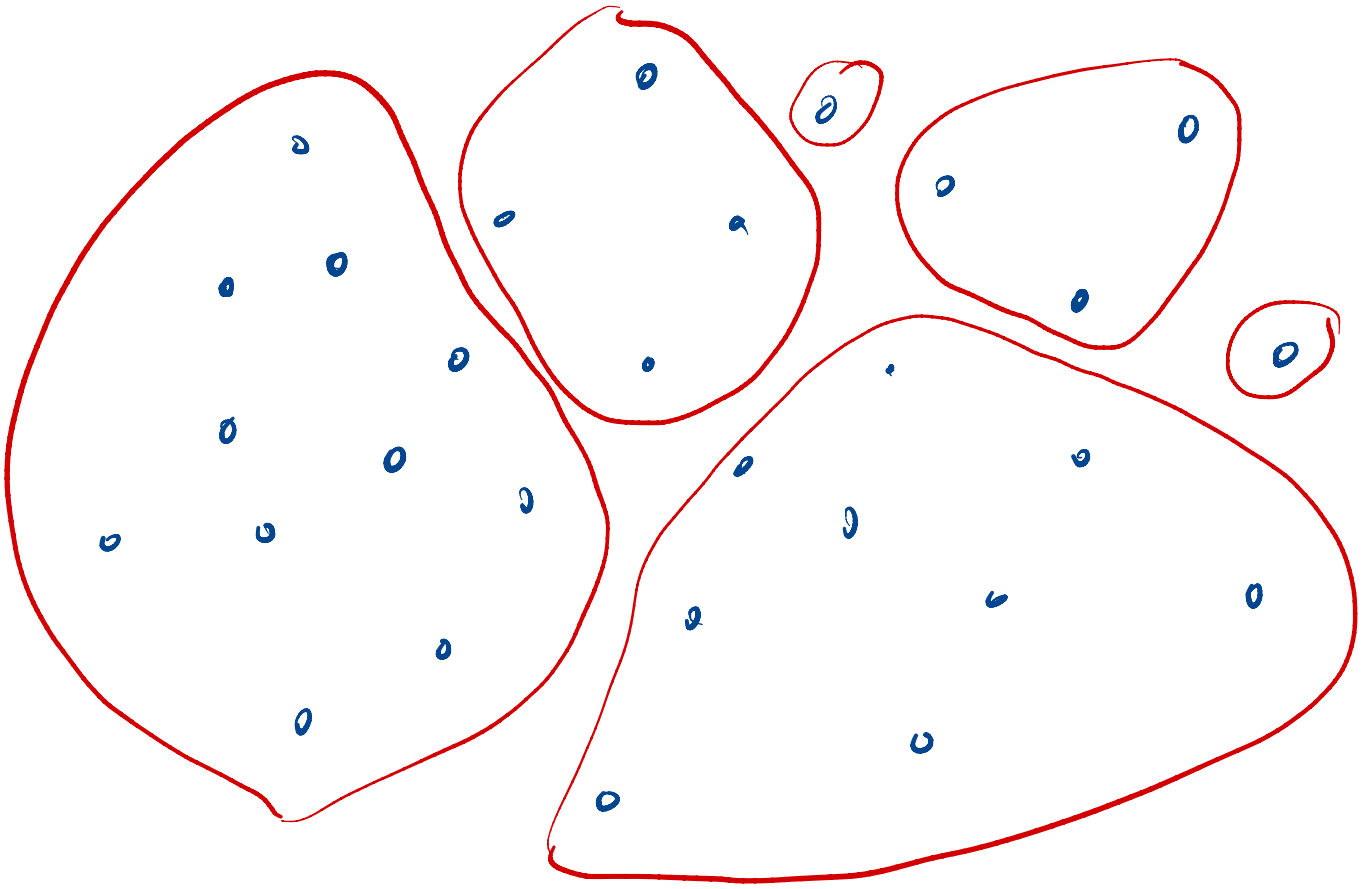
Let V be any finite set (in our scenario, a set of nodes). A partition of V is a grouping of its elements into non-empty subsets, in such a way that every element is included in exactly one subset.

In other words, $V_1, V_2, \ldots, V_\ell$ is a partition if and only if $V_1 \cup V_2 \cup \ldots \cup V_\ell = V$, $V_i \neq \emptyset$ for any $1 \leq i \leq \ell$, and $V_i \cap V_j = \emptyset$ for any $1 \leq i < j \leq \ell$.

Ground Truth

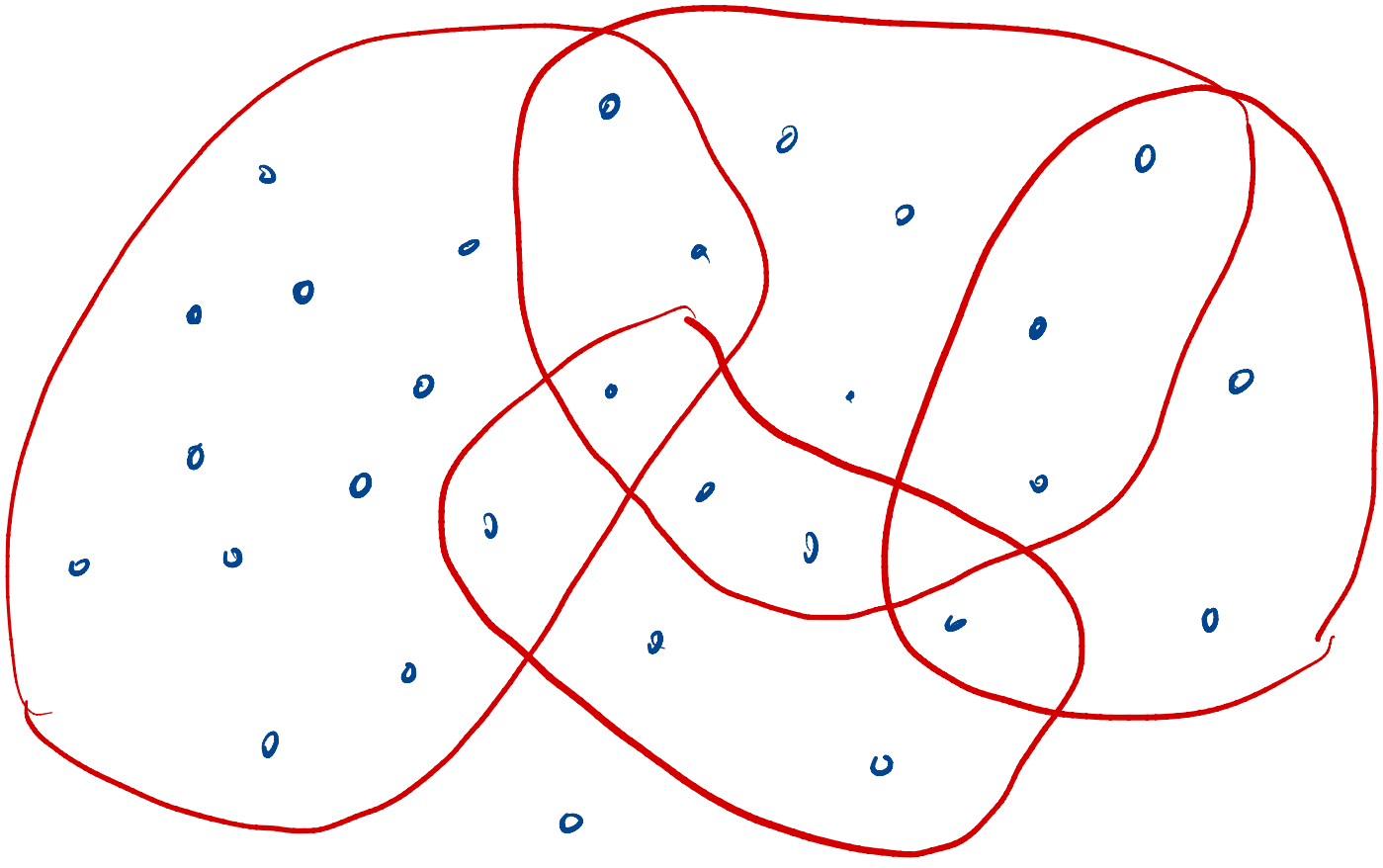


Ground Truth



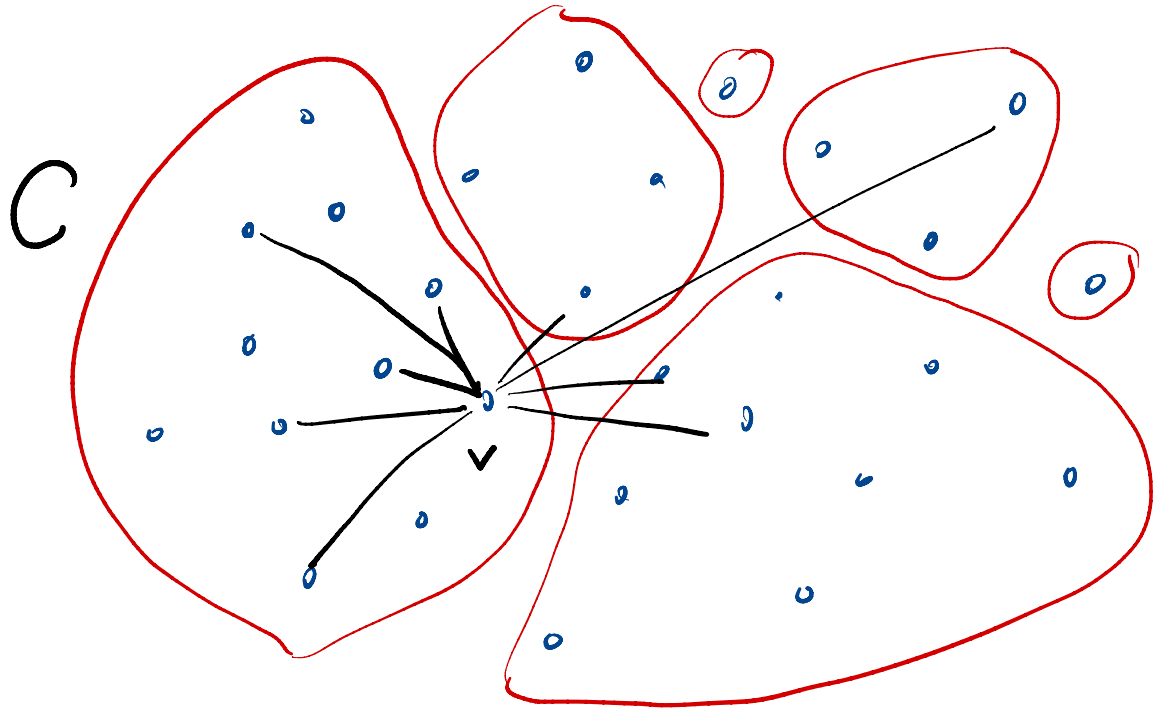
Partition

Ground Truth



Overlapping Communities and/or Outliers — see supplementary material

Communities



Consider any $v \in C$.

internal degree: $\deg^{int}(v) = |N(v) \cap C|$

external degree: $\deg^{ext}(v) = |N(v) \setminus C| = \deg(v) - \deg^{int}(v)$

Strong/Weak Community

Set $C \subseteq V$ forms a strong community if each node in C has more neighbours in C than outside of C, that is, for each $v \in C$ we have

$$\deg^{int}(v) > \deg^{ext}(v).$$

Strong/Weak Community

Set $C \subseteq V$ forms a strong community if each node in C has more neighbours in C than outside of C, that is, for each $v \in C$ we have

$$\deg^{int}(v) > \deg^{ext}(v).$$

Set $C \subseteq V$ forms a weak community if

$$\sum_{v \in C} \deg^{int}(v) > \sum_{v \in C} \deg^{ext}(v).$$

Strong/Weak Community

Set $C \subseteq V$ forms a strong community if each node in C has more neighbours in C than outside of C, that is, for each $v \in C$ we have

$$\deg^{int}(v) > \deg^{ext}(v).$$

Set $C \subseteq V$ forms a weak community if

$$\sum_{v \in C} \deg^{int}(v) > \sum_{v \in C} \deg^{ext}(v).$$

(strong $\Rightarrow$ weak but weak $\not\Rightarrow$ strong)

Communities

"Toy example": two researchers have the same number of friends on Instagram (say, 100) but they belong to two different communities.

Alice (data scientist) is part of a large community whereas Bob (mathematician) belongs to a small community.

Communities

"Toy example": two researchers have the same number of friends on Instagram (say, 100) but they belong to two different communities.

Alice (data scientist) is part of a large community whereas Bob (mathematician) belongs to a small community.

60% of friends of Alice do data science making her a clear member of that community.

Communities

"Toy example": two researchers have the same number of friends on Instagram (say, 100) but they belong to two different communities.

Alice (data scientist) is part of a large community whereas Bob (mathematician) belongs to a small community.

60% of friends of Alice do data science making her a clear member of that community.

What about Bob? Should we expect that more than 50% of his friends to be working in his field of research?

Communities

"Toy example": two researchers have the same number of friends on Instagram (say, 100) but they belong to two different communities.

Alice (data scientist) is part of a large community whereas Bob (mathematician) belongs to a small community.

60% of friends of Alice do data science making her a clear member of that community.

What about Bob? Should we expect that more than 50% of his friends to be working in his field of research?

No! In fact, it might be the case that there are less than 50 people around the world working in his field!

Communities

The number of internal neighbours of a given node should be proportional to the size of the community this node belongs to. But the probability that a node is adjacent to another member of its community should be larger than the probability of being adjacent to a random node in the whole graph.

Communities

The number of internal neighbours of a given node should be proportional to the size of the community this node belongs to. But the probability that a node is adjacent to another member of its community should be larger than the probability of being adjacent to a random node in the whole graph.

Community

Let $C \subseteq V$ ($C \neq \emptyset$ and $C \neq V$). C is a community if each node in C has proportionally more neighbours in C than outside of C, that is, for each $v \in C$ we have

$$\frac{\deg^{int}(v)}{|C|} > \frac{\deg^{ext}(v)}{|V \setminus C|}.$$

Node Roles

Partition $\mathcal{A} = \{A_1, A_2, \ldots, A_\ell\}$ of V; (ground truth or a partition returned by some community detection algorithm).

$\deg_{A_i}(v) = |N(v) \cap A_i|$ (number of neighbours of v in part A_i),
$\deg^{int}(v) = \deg_{A_i}(v)$ for the unique $i \in [\ell]$ for which $v \in A_i$.

Node Roles

Partition $\mathcal{A} = \{A_1, A_2, \ldots, A_\ell\}$ of V; (ground truth or a partition returned by some community detection algorithm).

$\deg_{A_i}(v) = |N(v) \cap A_i|$ (number of neighbours of v in part A_i),
$\deg^{int}(v) = \deg_{A_i}(v)$ for the unique $i \in [\ell]$ for which $v \in A_i$.

Normalized within-module degree

The normalized within-module degree of a node v (with respect to $\mathcal{A}$) is defined as follows:

$$z(v) = \frac{\deg^{int}(v) - \mu(v)}{\sigma(v)},$$

where $\mu(v)/\sigma(v)$ are the mean/standard deviation of $\deg^{int}(u)$ over all nodes in the part v belongs to.

Node Roles

$z(v)$ captures how strongly a particular node is connected to other nodes within its own community, completely ignoring edges between communities.

Node Roles

$z(v)$ captures how strongly a particular node is connected to other nodes within its own community, completely ignoring edges between communities.

$z(v)$ is a familiar Z-score as it measures how many standard deviations the internal degree of v deviates from the mean.

Node Roles

$z(v)$ captures how strongly a particular node is connected to other nodes within its own community, completely ignoring edges between communities.

$z(v)$ is a familiar Z-score as it measures how many standard deviations the internal degree of v deviates from the mean.

$z(v)$ is large and positive: v is tightly connected to other nodes within the community.
$|z(v)|$ is large and $z(v)$ is negative: v is loosely connected to other peers.

Node Roles

$z(v)$ captures how strongly a particular node is connected to other nodes within its own community, completely ignoring edges between communities.

$z(v)$ is a familiar Z-score as it measures how many standard deviations the internal degree of v deviates from the mean.

$z(v)$ is large and positive: v is tightly connected to other nodes within the community.
$|z(v)|$ is large and $z(v)$ is negative: v is loosely connected to other peers.

If $z(v) > 2.5$, then node v is classified as a hub; otherwise, it is a non-hub.

Node Roles

Another important aspect: how edges (both internal and external) are distributed between all parts of the partition $\mathcal{A}$.

Participation coefficient

The participation coefficient of a node v (with respect to $\mathcal{A}$) is defined as follows:

$$p(v) = 1 - \sum_{i=1}^{\ell} \left(\frac{\deg_{A_i}(v)}{\deg(v)} \right)^2 .$$

Node Roles

Another important aspect: how edges (both internal and external) are distributed between all parts of the partition $\mathcal{A}$.

Participation coefficient

The participation coefficient of a node v (with respect to $\mathcal{A}$) is defined as follows:

$$p(v) = 1 - \sum_{i=1}^{\ell} \left(\frac{\deg_{A_i}(v)}{\deg(v)} \right)^2 .$$

If v has neighbours exclusively in one part, then $p(v) = 0$.

If the neighbours of v are homogeneously distributed among all parts, then $p(v) \approx 1 - 1/\ell \approx 1$.

Node Roles

Non-hubs are partitioned into the following four classes:

– ultra-peripheral nodes (almost all edges are internal) if $p(v) < 0.05$,

– peripheral nodes (most edges are internal) if $0.05 \leq p(v) < 0.62$,

– connector nodes (most edges are external) if $0.62 \leq p(v) < 0.80$,

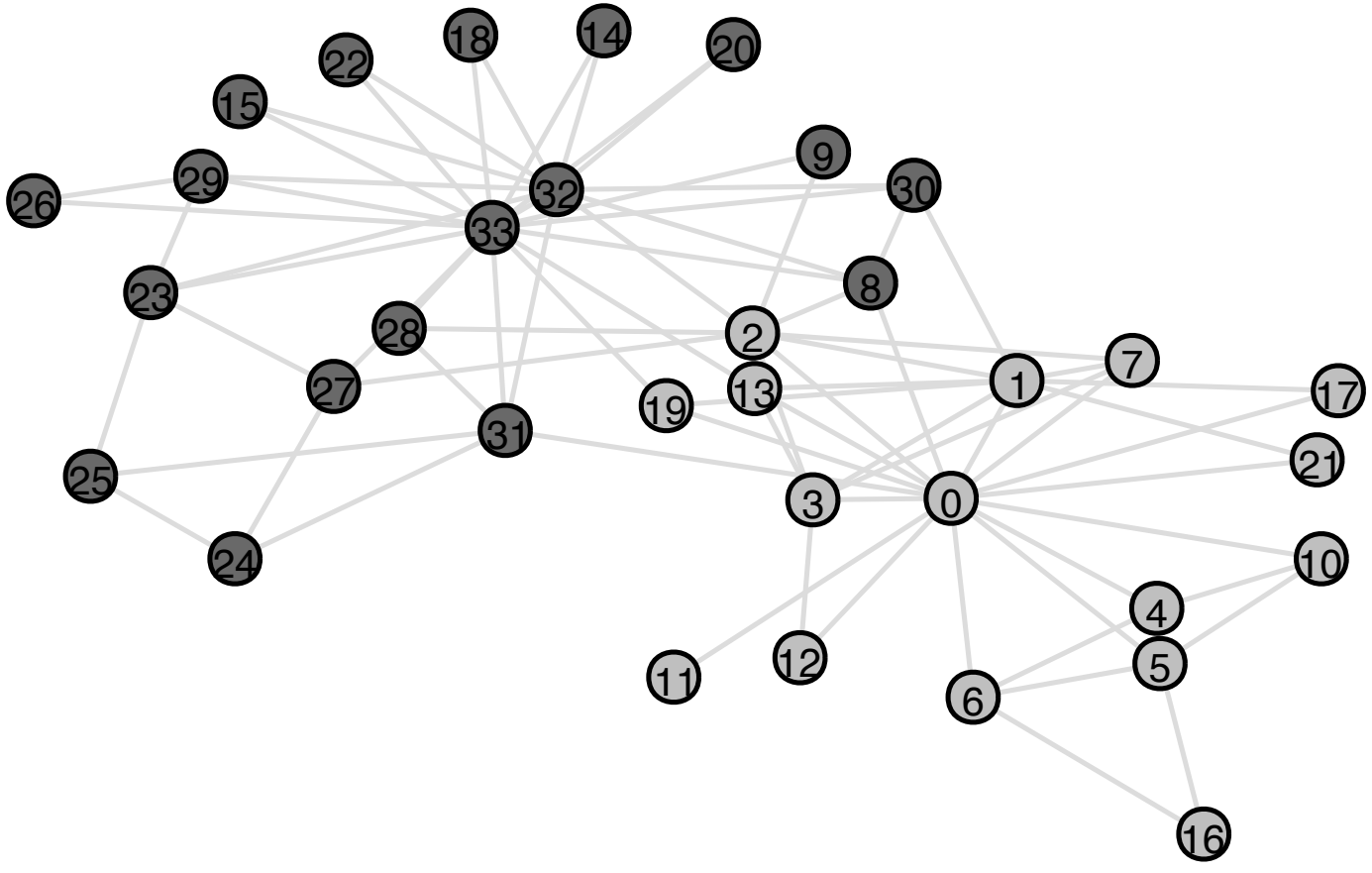
– kinless nodes (neighbours are homogeneously distributed) if $p(v) \geq 0.80$.

Node Roles

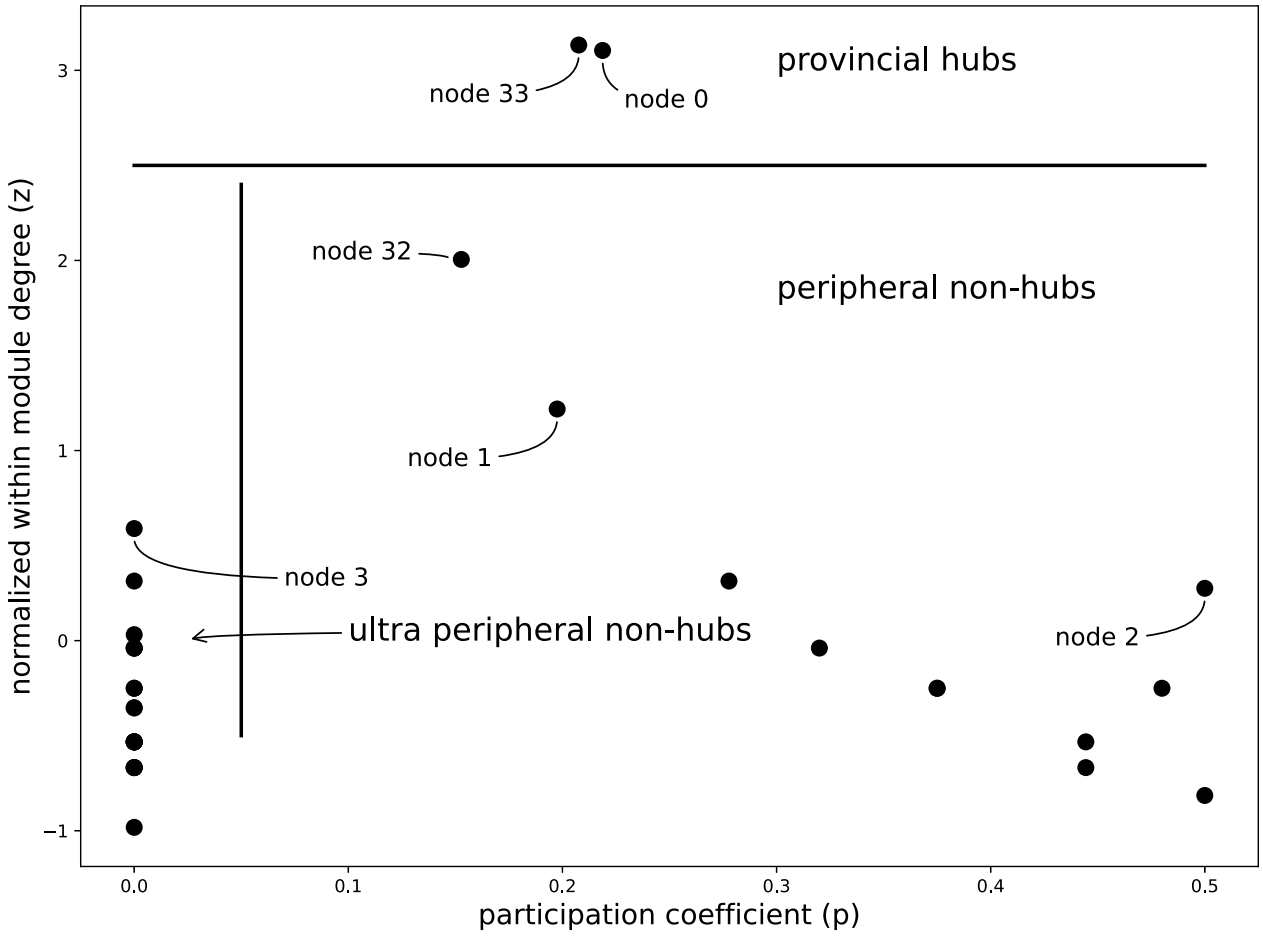
Non-hubs are partitioned into the following four classes:
– ultra-peripheral nodes (almost all edges are internal) if
$p(v) < 0.05$,
– peripheral nodes (most edges are internal) if
$0.05 \leq p(v) < 0.62$,
– connector nodes (most edges are external) if
$0.62 \leq p(v) < 0.80$,
– kinless nodes (neighbours are homogeneously distributed) if
$p(v) \geq 0.80$.

On the other hand, hubs are partitioned into three classes:
– provincial hubs if $p(v) < 0.30$,
– connector hubs if $0.30 \leq p(v) < 0.75$,
– kinless hubs if $p(v) \geq 0.75$.

Node Roles


18
14
20
22
15
9
29
30
26
32
33
8
23
28
2
7
27
13
1
17
19
31
21
25
3
0
24
10
4
11
12
5
6
16

Node Roles


provincial hubs
node 33
node 0
node 32
peripheral non-hubs
node 1
node 3
ultra peripheral non-hubs
node 2
normalized within module degree (z)
participation coefficient (p)
3
2
1
0
−1
0.0
0.1
0.2
0.3
0.4
0.5

Node Roles

Anomaly Score

The anomaly score of a node v (with respect to $\mathcal{A}$) is defined as follows:

$$cd(v) = \frac{\deg(v)}{\deg^{int}(v)}.$$

Node Roles

Anomaly Score

The anomaly score of a node v (with respect to $\mathcal{A}$) is defined as follows:

$$cd(v) = \frac{\deg(v)}{\deg^{int}(v)}.$$

Participation coefficient and anomaly score ignore the sizes of communities. It is natural to expect that if a node belongs to a larger community it should have more neighbours in it by pure luck.

Anomaly Score

The anomaly score of a node v (with respect to $\mathcal{A}$) is defined as follows:

$$cd(v) = \frac{\deg(v)}{\deg^{int}(v)}.$$

Participation coefficient and anomaly score ignore the sizes of communities. It is natural to expect that if a node belongs to a larger community it should have more neighbours in it by pure luck.

We can model "pure luck" with the Chung-Lu model.

Node Roles

Community Association Strength

The community association strength of a node v (with respect to community A_i from $\mathcal{A}$) is defined as follows:

$$cas(v) = \frac{\deg_{A_i}(v)}{\deg(v)} - \frac{\mathrm{vol}(A_i) - \deg(v)}{\mathrm{vol}(V)}.$$

Community Distribution Distance

The community distribution distance of a node v (with respect to $\mathcal{A}$) is defined as follows:

$$cdd(v) = \left(\sum_{i=1}^{\ell} \left(\frac{\deg_{A_i}(v)}{\deg(v)} - \frac{\mathrm{vol}(A_i)}{\mathrm{vol}(V)} \right)^2 \right)^{1/2}.$$

The Number of Partitions

Bell number = the number of partitions of a set of size n.

The Number of Partitions

Bell number = the number of partitions of a set of size n.

Clearly, $B_n \geq 2^n$ (the number of partitions into 2 parts) but, in fact, B_n grows even faster!

The Number of Partitions

Bell number = the number of partitions of a set of size n.

Clearly, $B_n \geq 2^n$ (the number of partitions into 2 parts) but, in fact, B_n grows even faster!

Asymptotic behaviour:

$$\frac{\ln B_n}{n} = \ln n - \ln \ln n - 1 + o(1),$$

which implies that

$$B_n = \left(\frac{n}{(e + o(1)) \ln n} \right)^n.$$

The Number of Partitions

Bell number = the number of partitions of a set of size n.

Clearly, $B_n \geq 2^n$ (the number of partitions into 2 parts) but, in fact, B_n grows even faster!

Asymptotic behaviour:

$$\frac{\ln B_n}{n} = \ln n - \ln \ln n - 1 + o(1),$$

which implies that

$$B_n = \left(\frac{n}{(e + o(1)) \ln n} \right)^n.$$

Important implication: we will not be able to investigate all partitions even if, say, $n = 20$. We aim for heuristic algorithms.

Synthetic Models with Community Structure

Synthetic Models with Community Structure

In order to evaluate algorithms in a fair and rigorous way, one need a good benchmark: large synthetic network with an engineered ground truth.

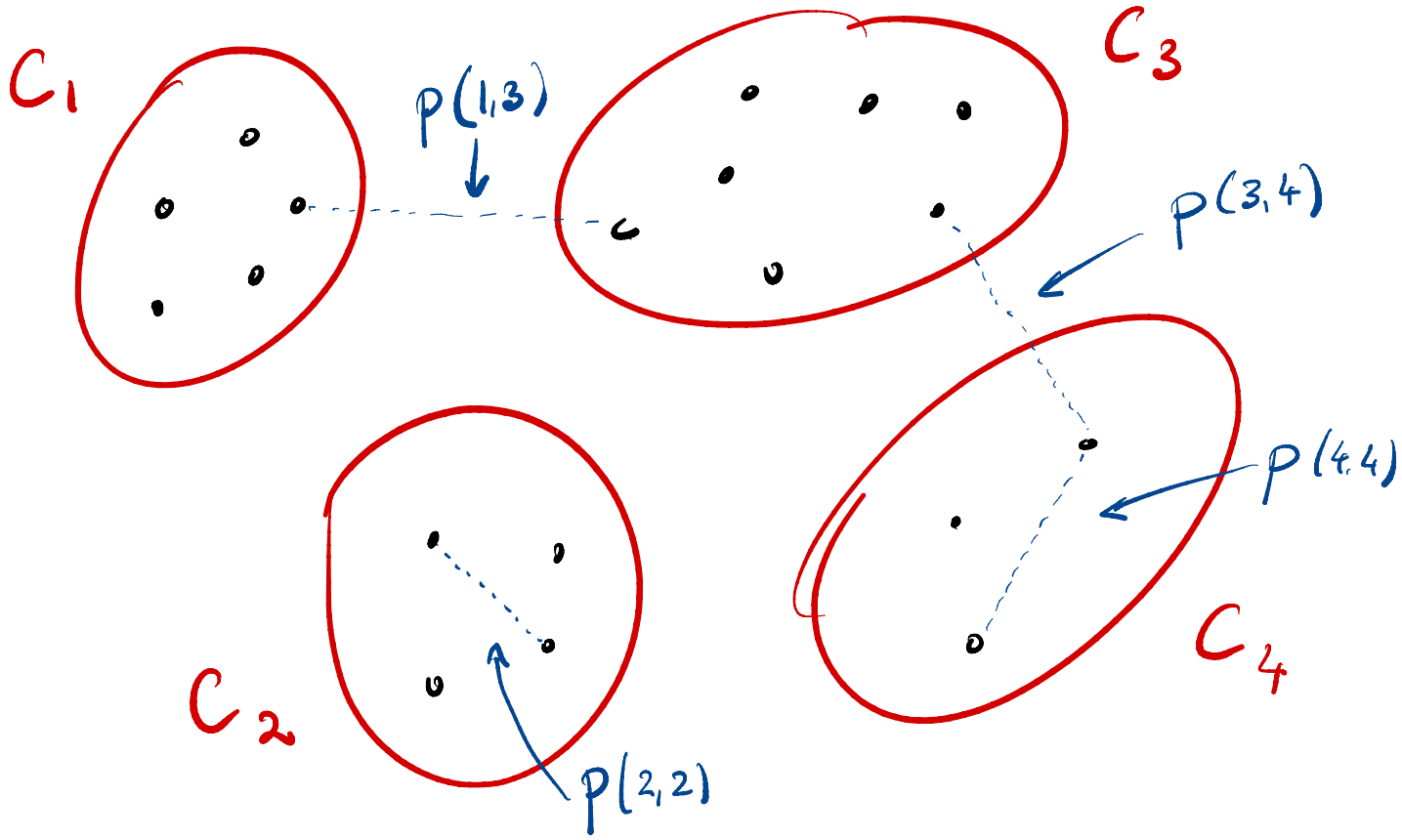
Synthetic Models with Community Structure

In order to evaluate algorithms in a fair and rigorous way, one need a good benchmark: large synthetic network with an engineered ground truth.

We present a few random graph models with community structure and then show how to use them to make such evaluation.

Stochastic Block Model

Simple and natural generalization of $\mathcal{G}(n,p)$.



Stochastic Block Model

Simple and natural generalization of $\mathcal{G}(n,p)$.

Stochastic Block Model

Let $\mathbf{P} = (p(i,j))_{i,j\in[\ell]}$ be a symmetric $\ell \times \ell$ matrix with all entries in $[0,1]$ and let $\mathbf{N} = (n_i)_{i\in[\ell]}$ be a vector of ℓ natural numbers. The stochastic block model $\mathcal{S}(\mathbf{P},\mathbf{N})$ can be generated by starting with an empty graph on the set of nodes $[n] = \{1,2,\ldots,n\}$ with $n = \sum_{i\in[\ell]} n_i$ that is partitioned into ℓ subsets $C_1, C_2, \ldots, C_\ell$, called communities, with $|C_i| = n_i$, $i \in [\ell]$. For each pair of nodes $u, v \in [n]$ with $u < v$, $u \in C_i$, and $v \in C_j$, we independently introduce an edge uv in $\mathcal{S}(\mathbf{P},\mathbf{N})$ with probability $p(i,j)$.

If $\ell = 1$, then we recover $\mathcal{G}(n,p)$.

Artificial Benchmark for Community Detection (ABCD)

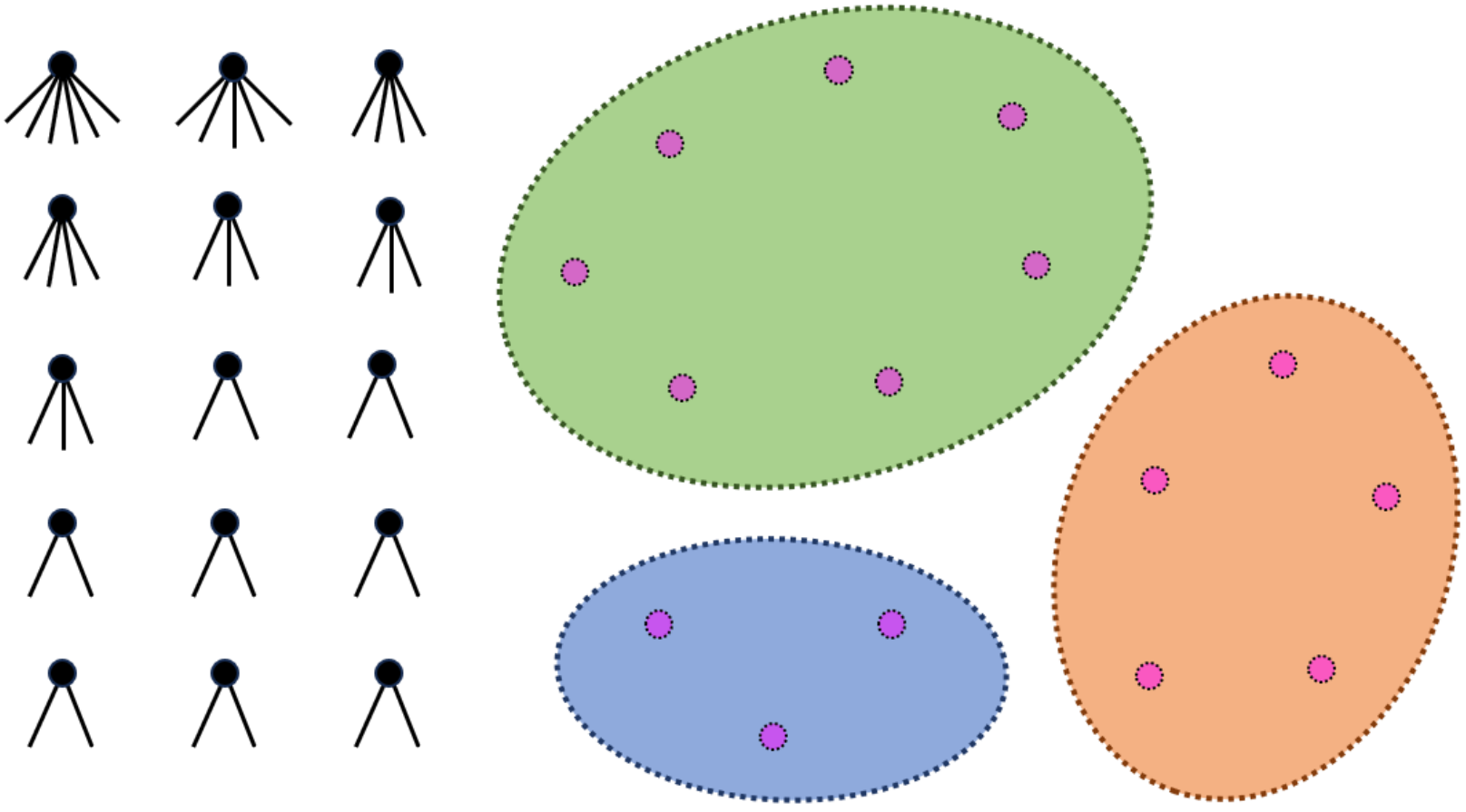
Here is an alternative.

ABCD model

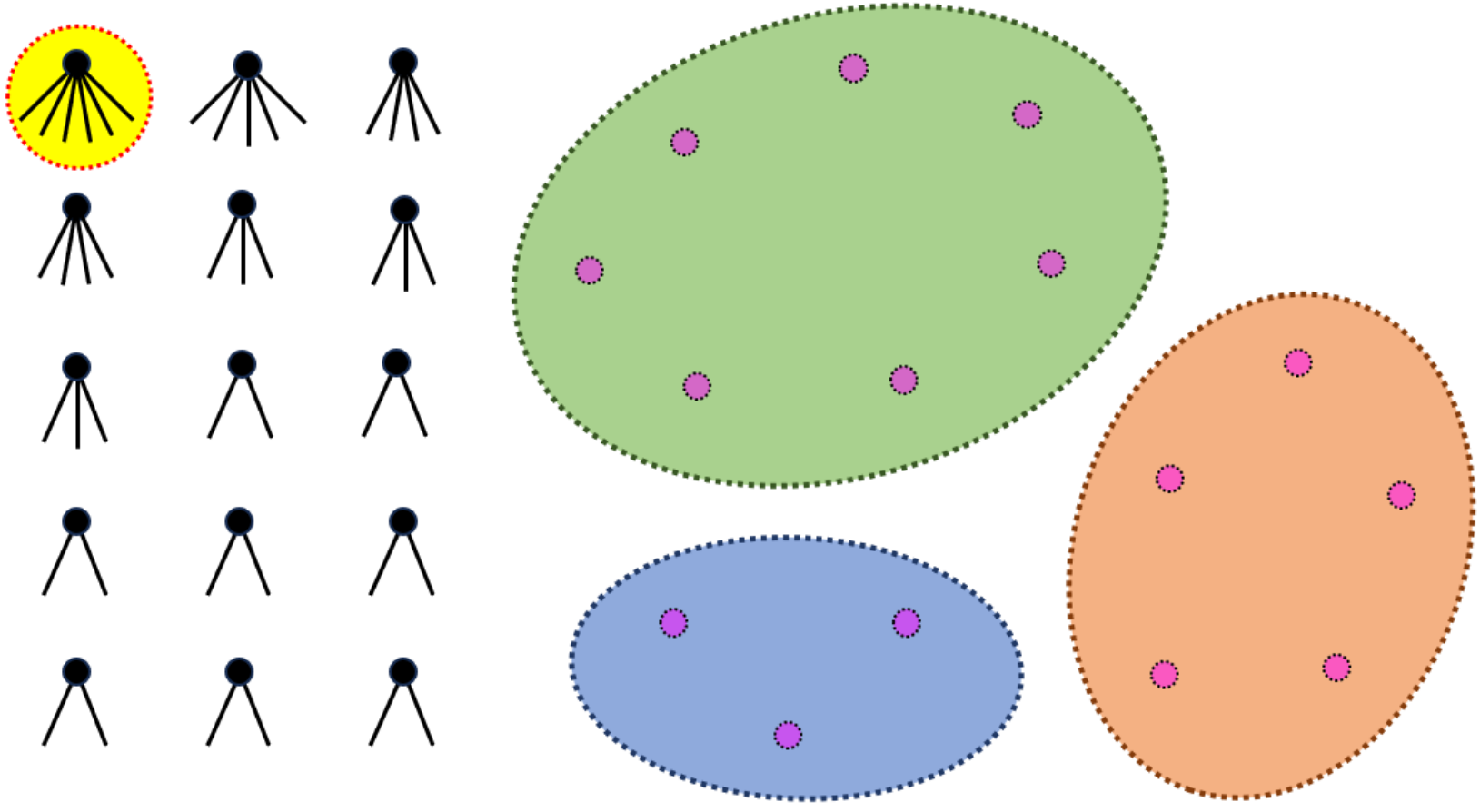
Let $\gamma \in [2,3]$, $\tau \in [1,2]$, and $\xi \in [0,1]$ (ξ is called the mixing parameter). The ABCD model $\mathscr{A}(\gamma, \tau, \mu)$ generates random graphs with community sizes and node degrees following truncated power law distributions with exponents τ and, respectively, γ. The mixing parameter ξ controls the fraction of edges that are between communities.

The high level description is almost the same as for LFR but, of course, details differ—see the book.

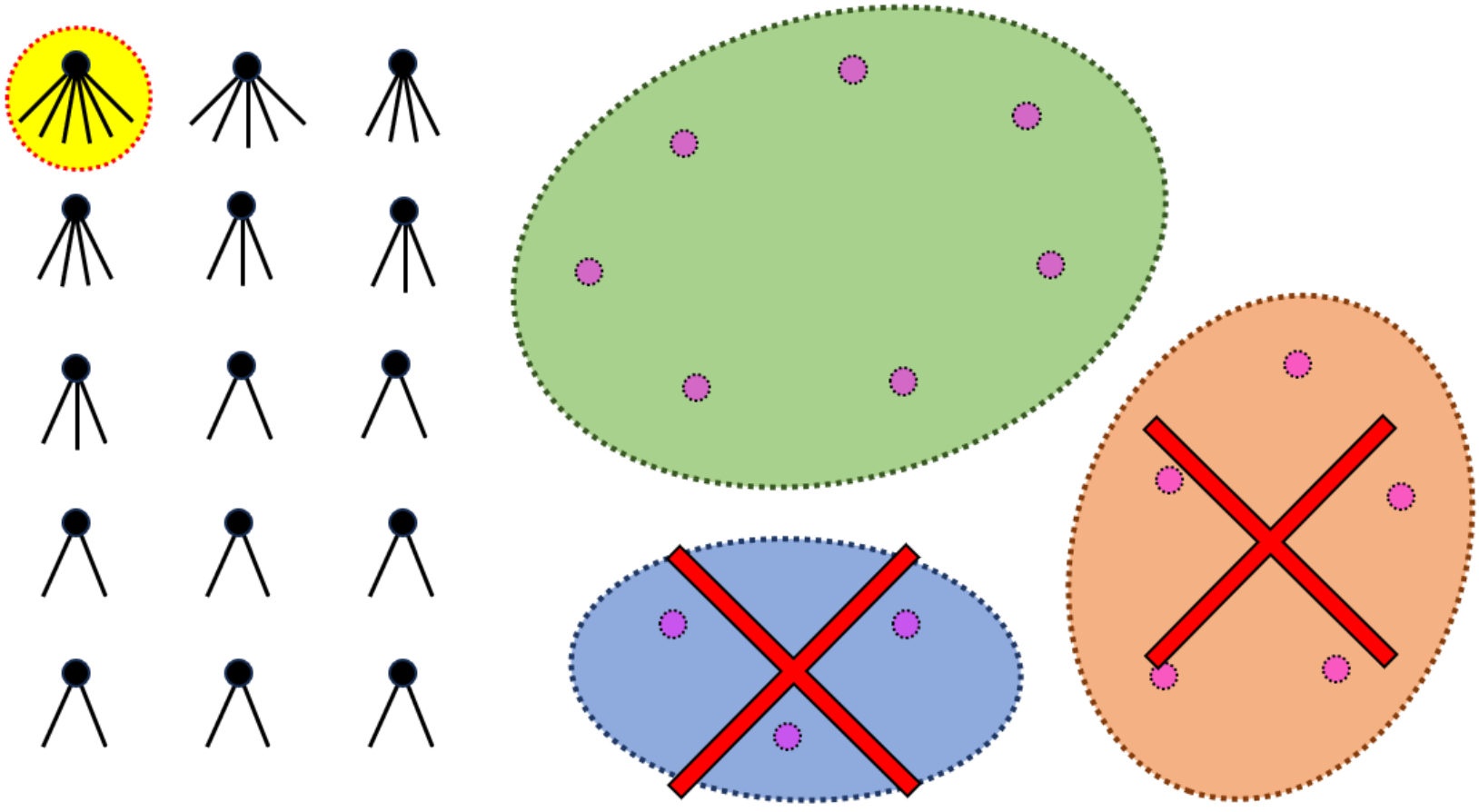
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



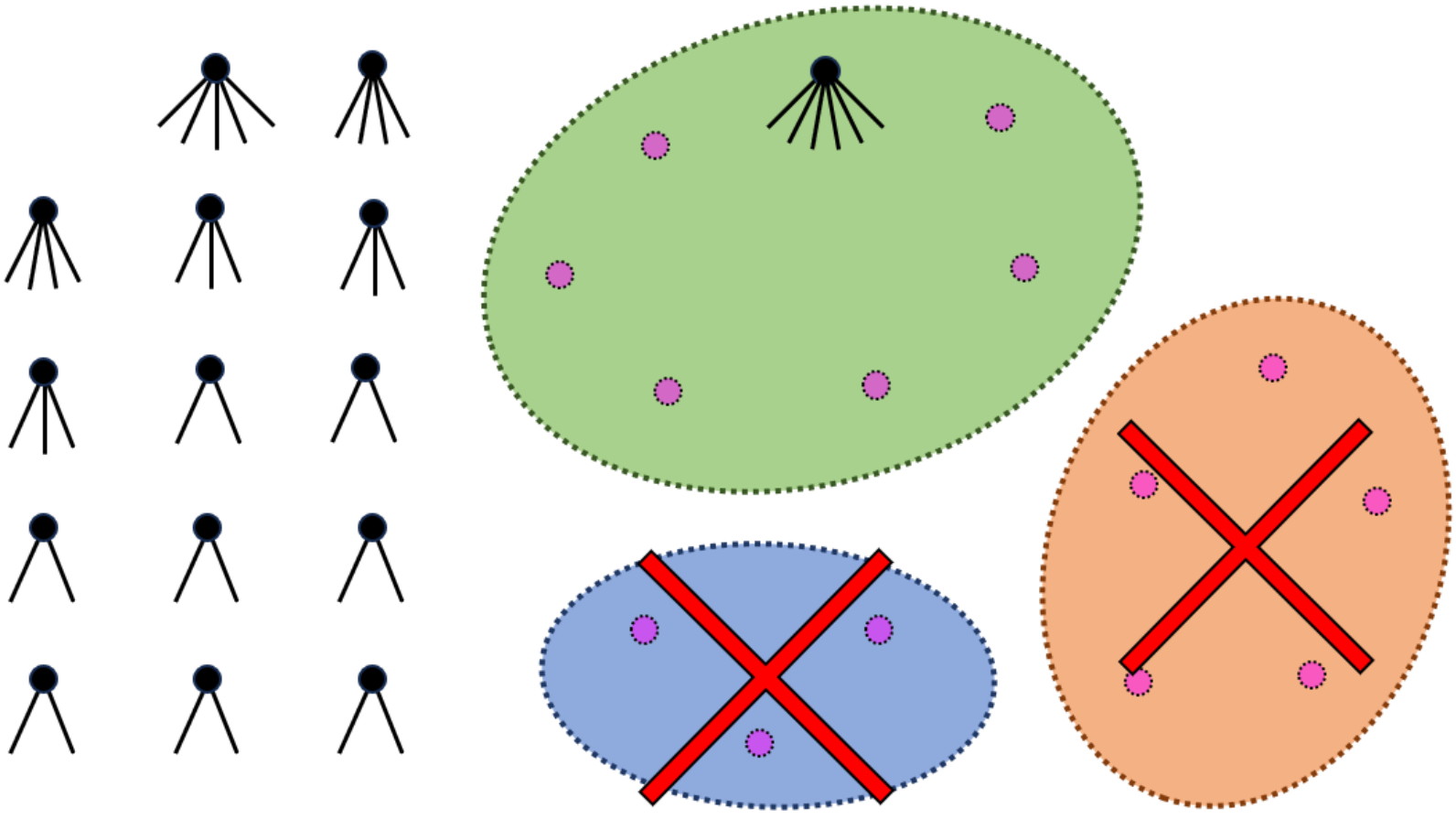
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



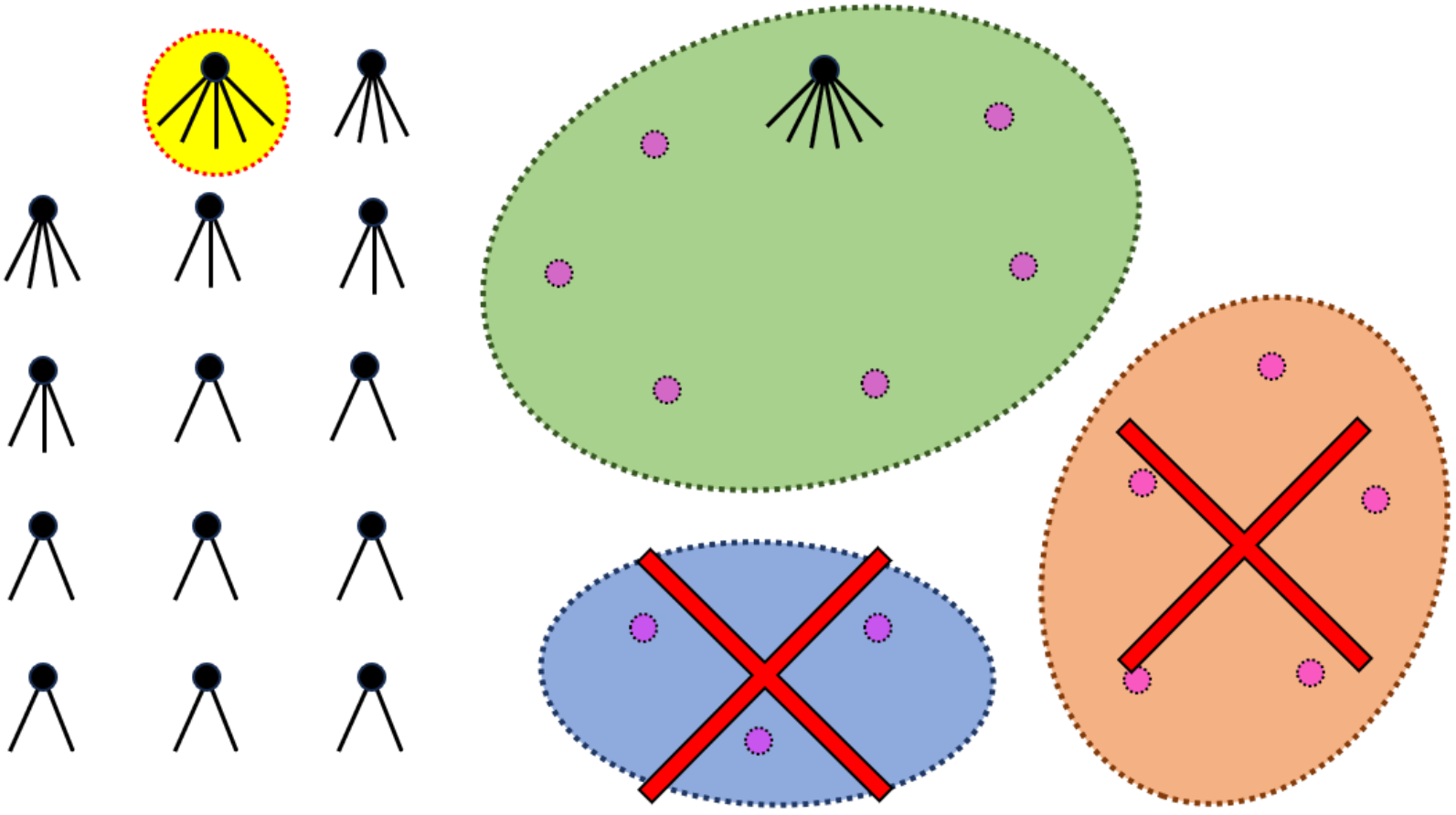
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



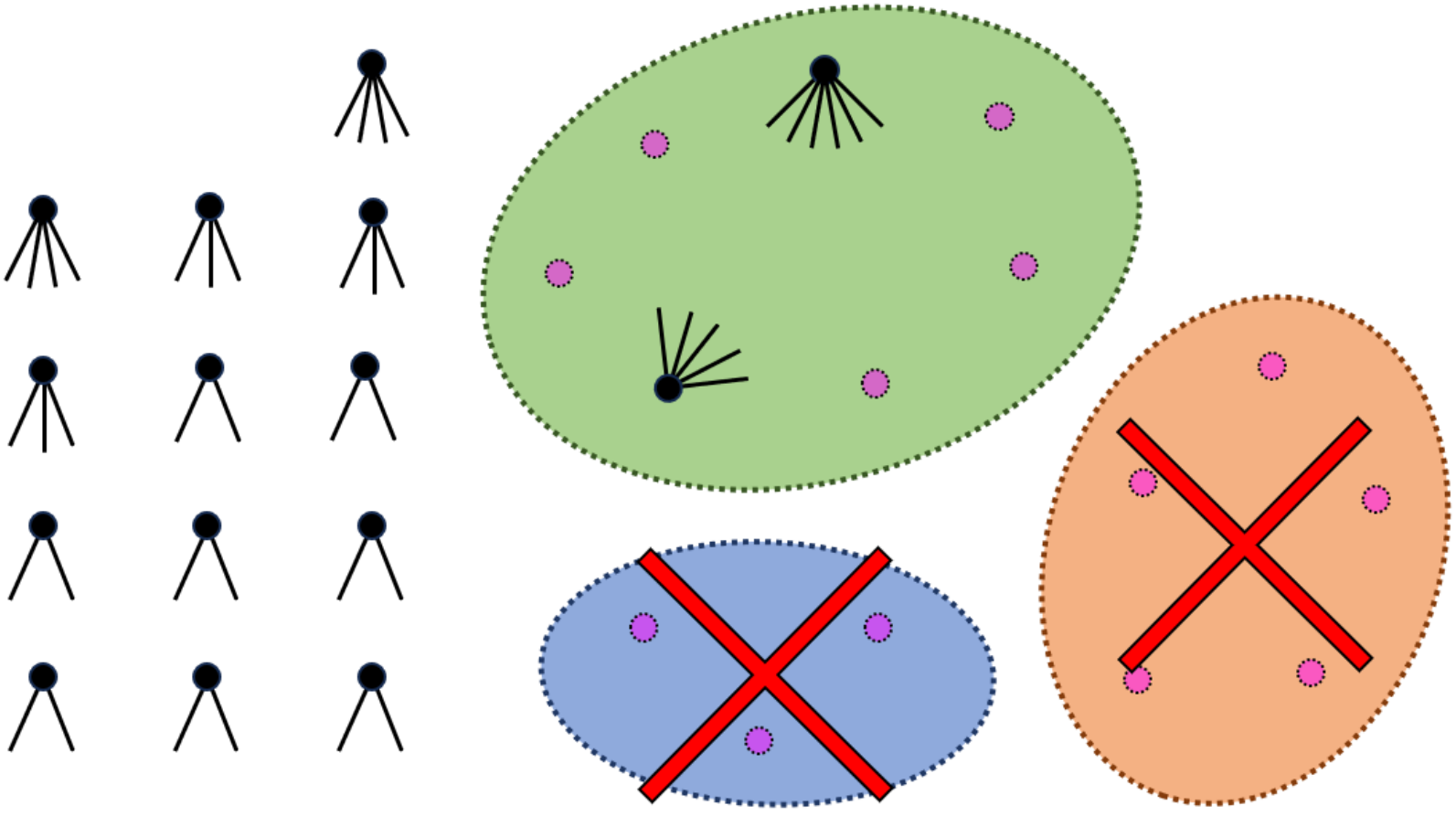
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



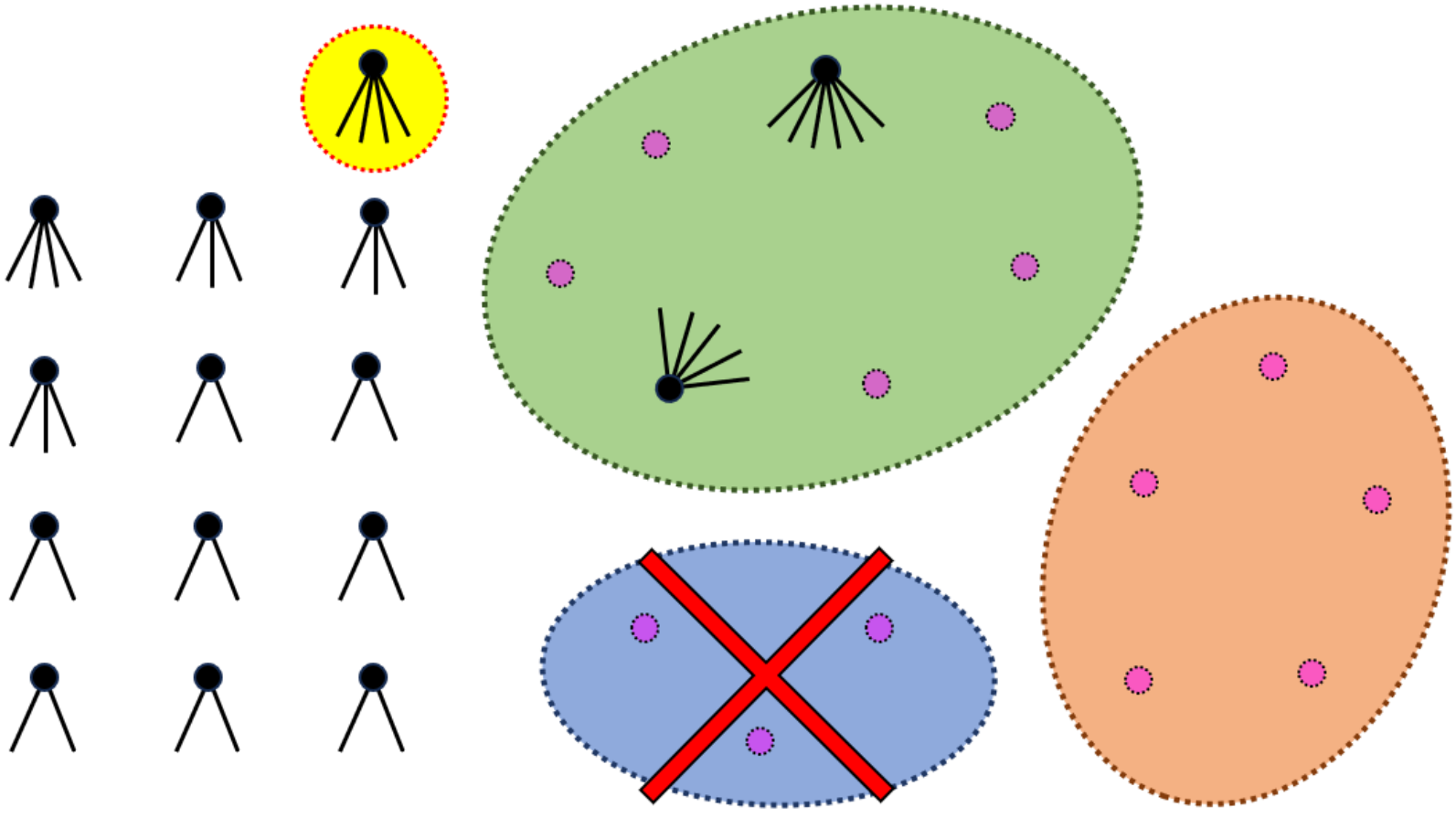
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



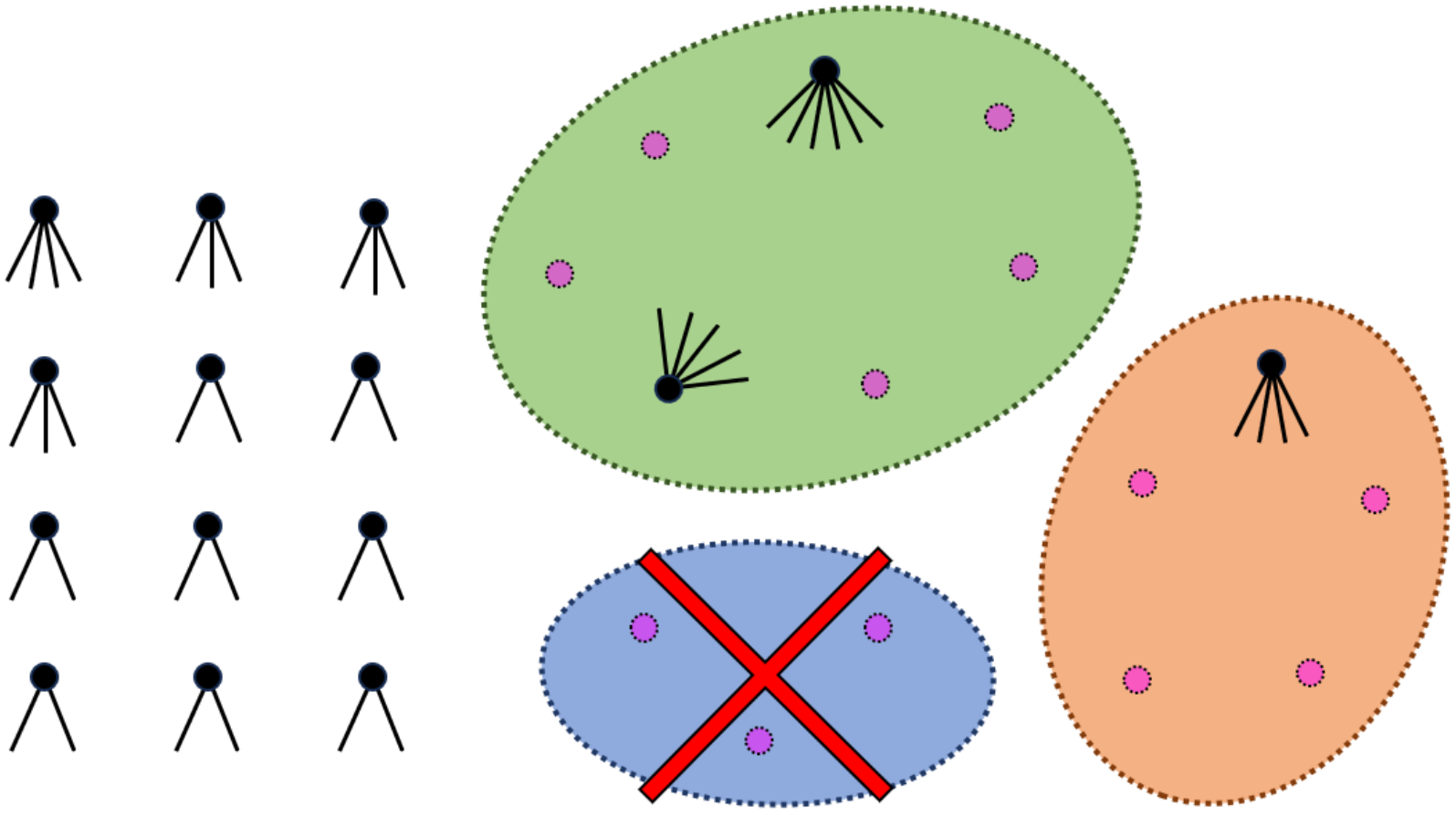
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



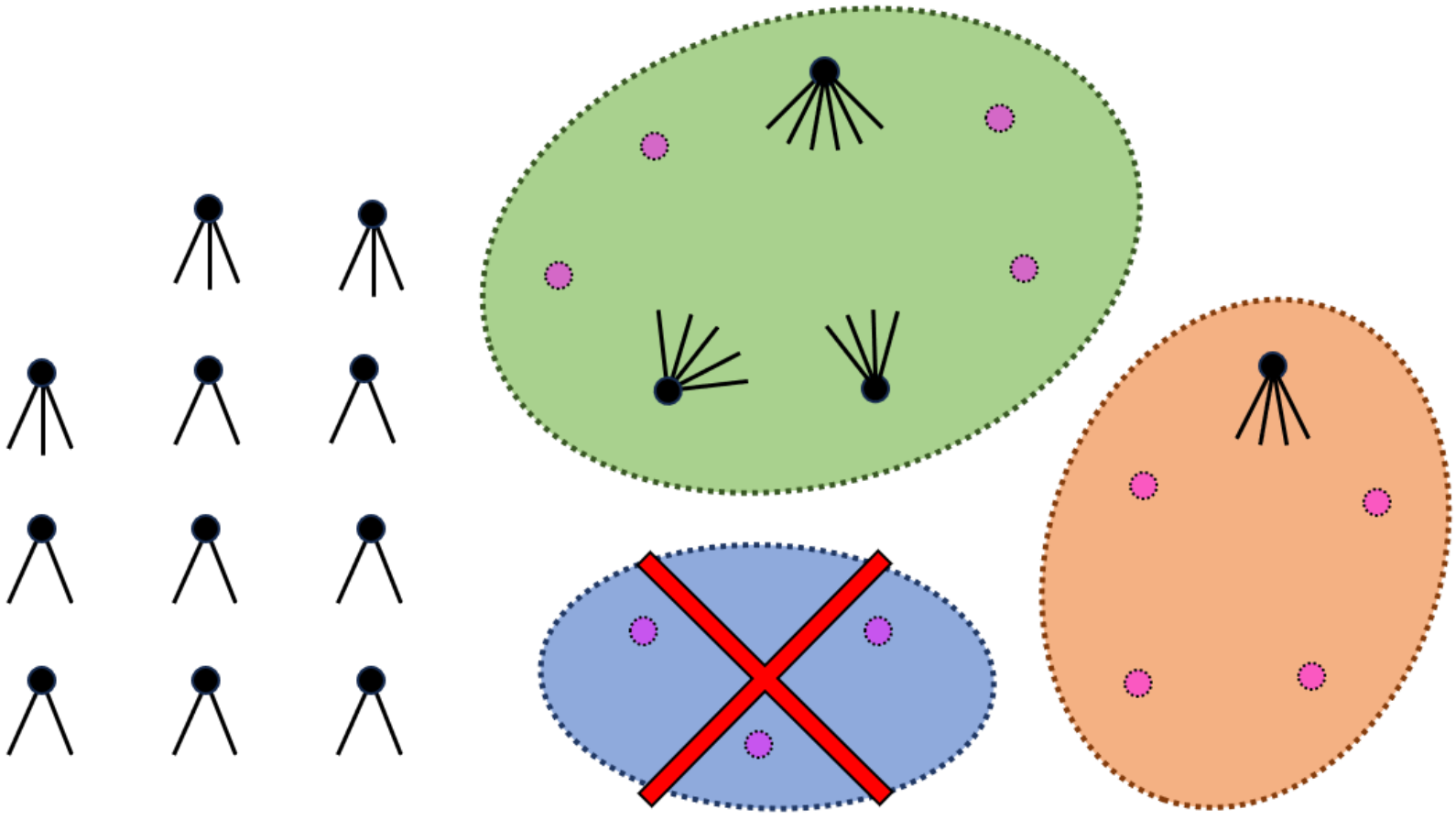
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



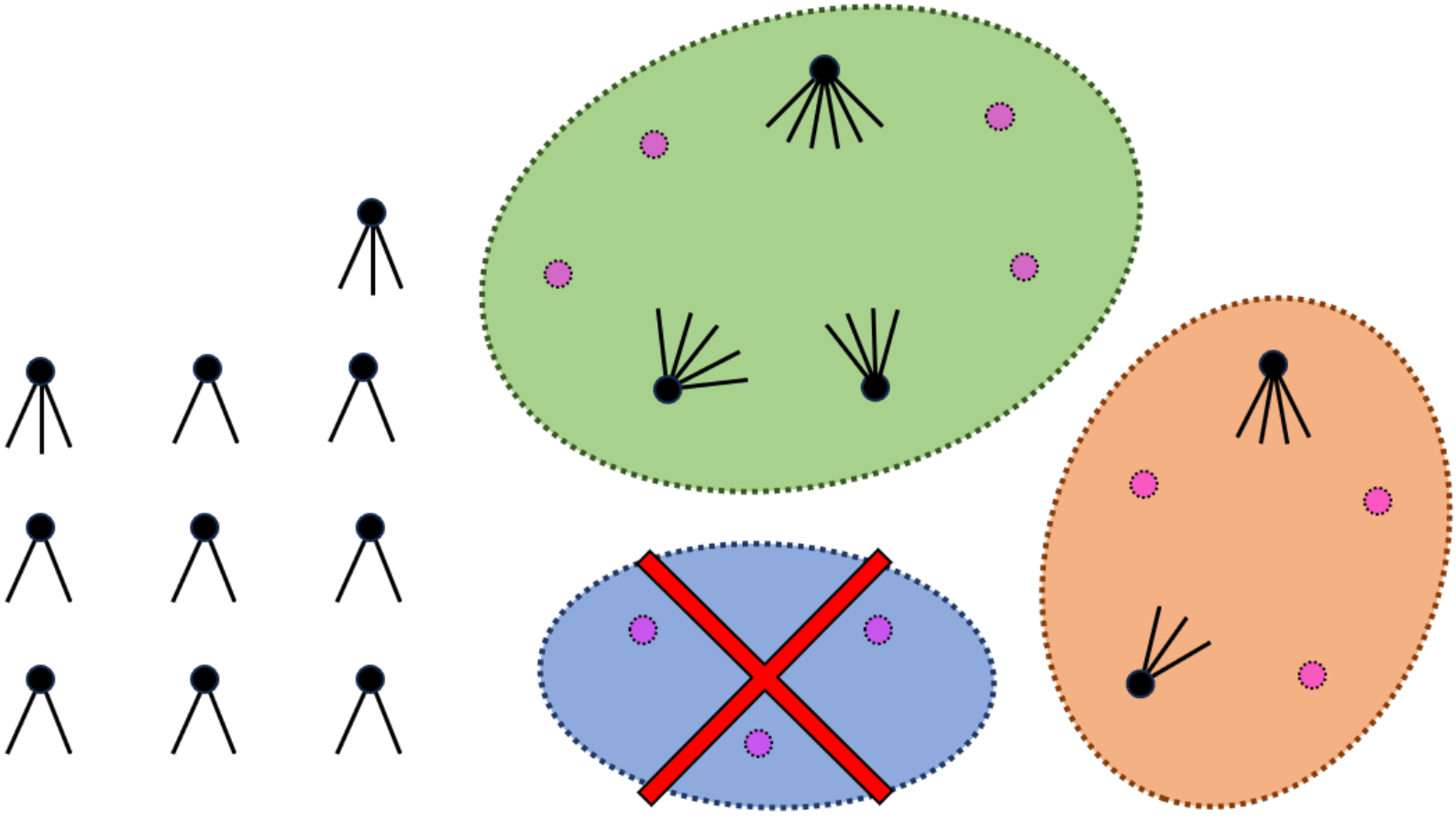
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



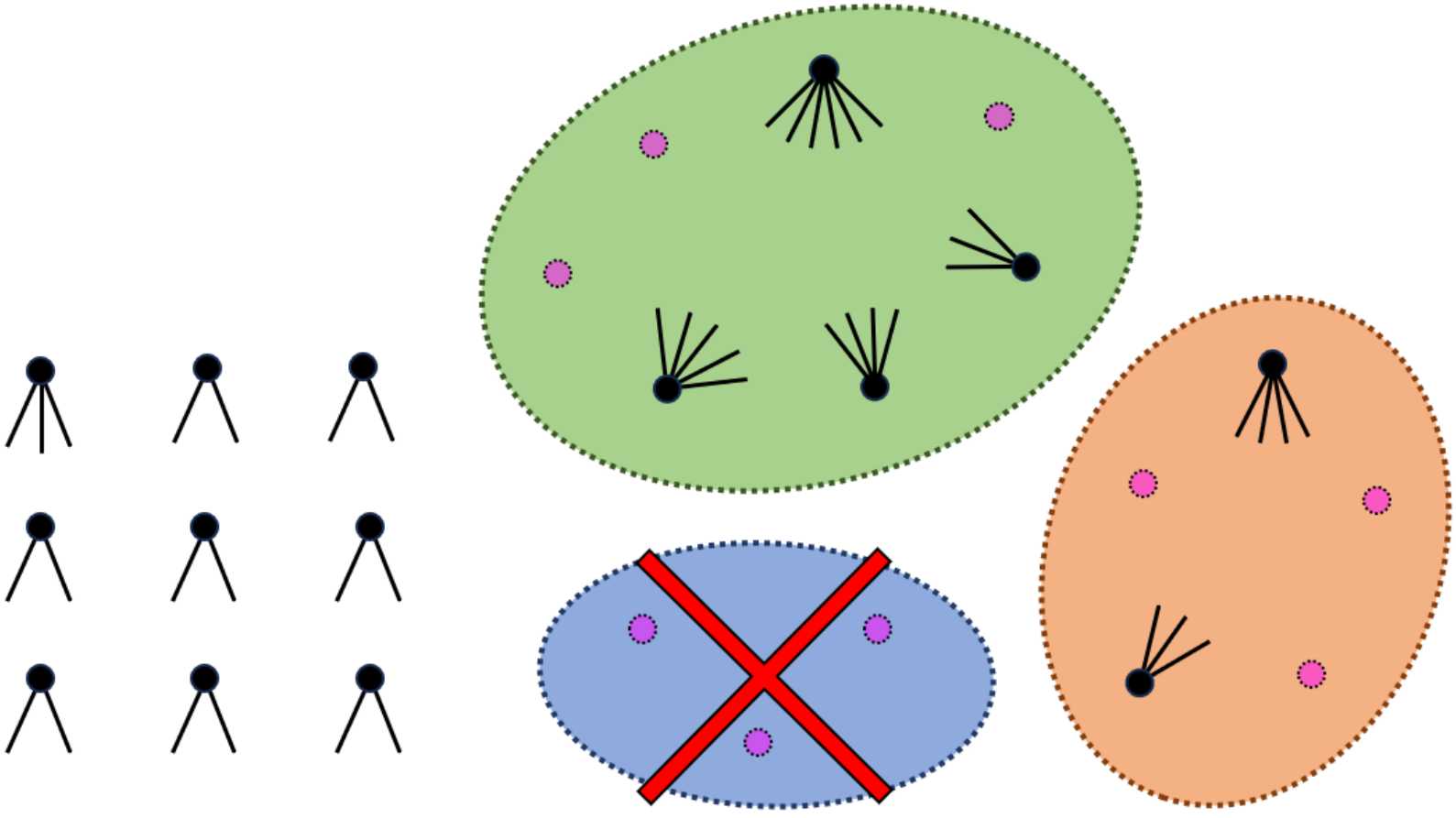
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



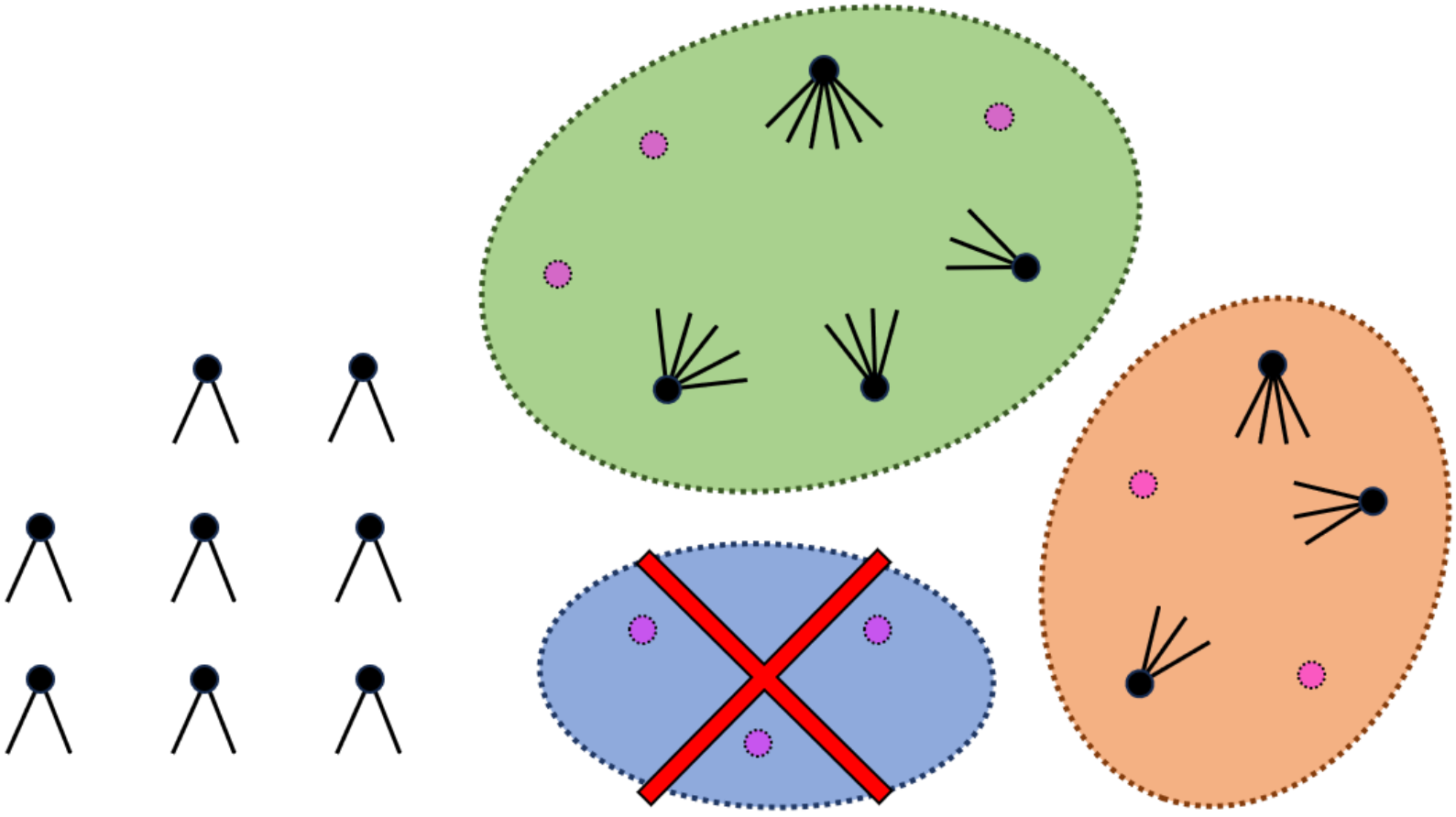
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



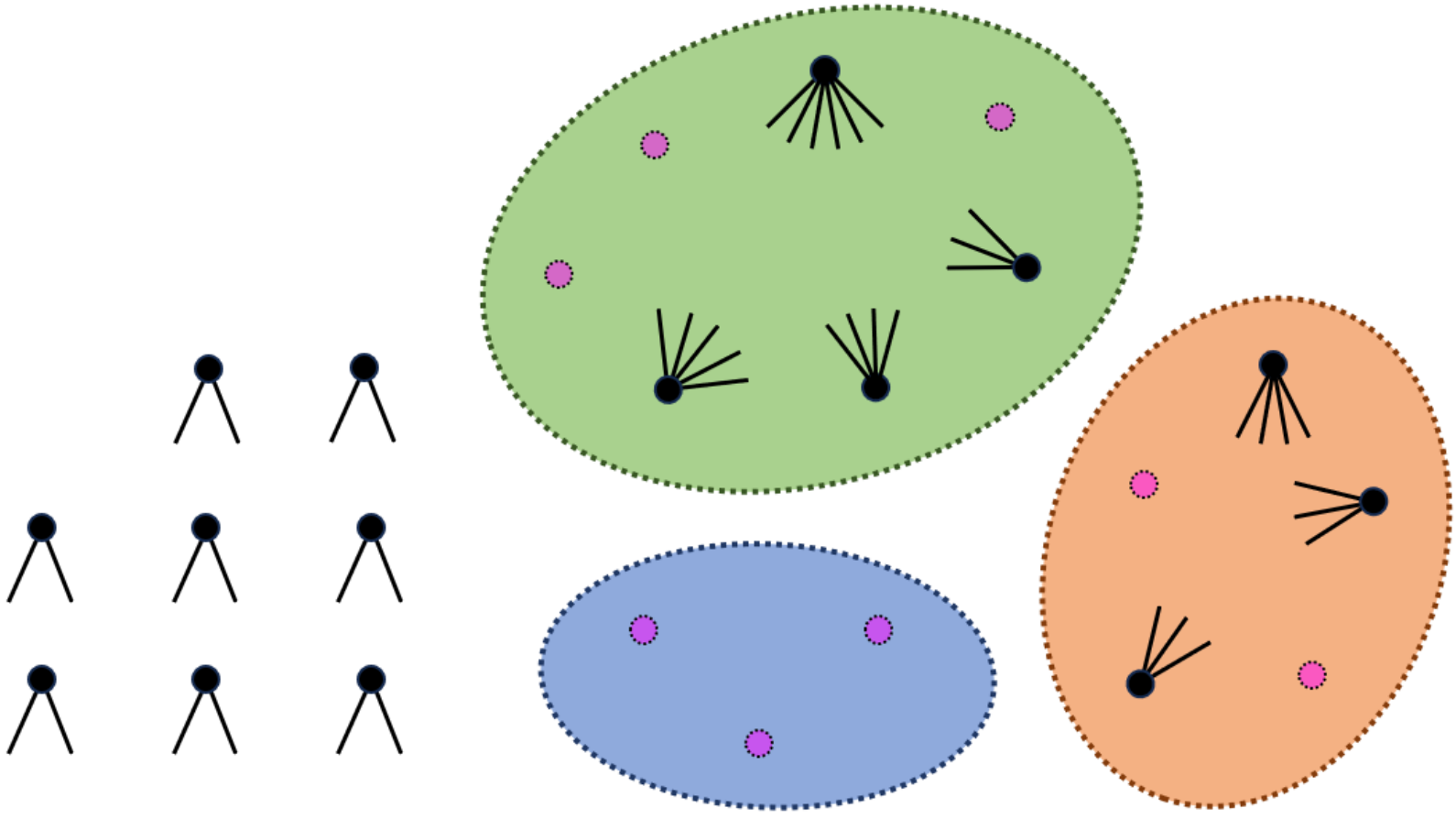
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



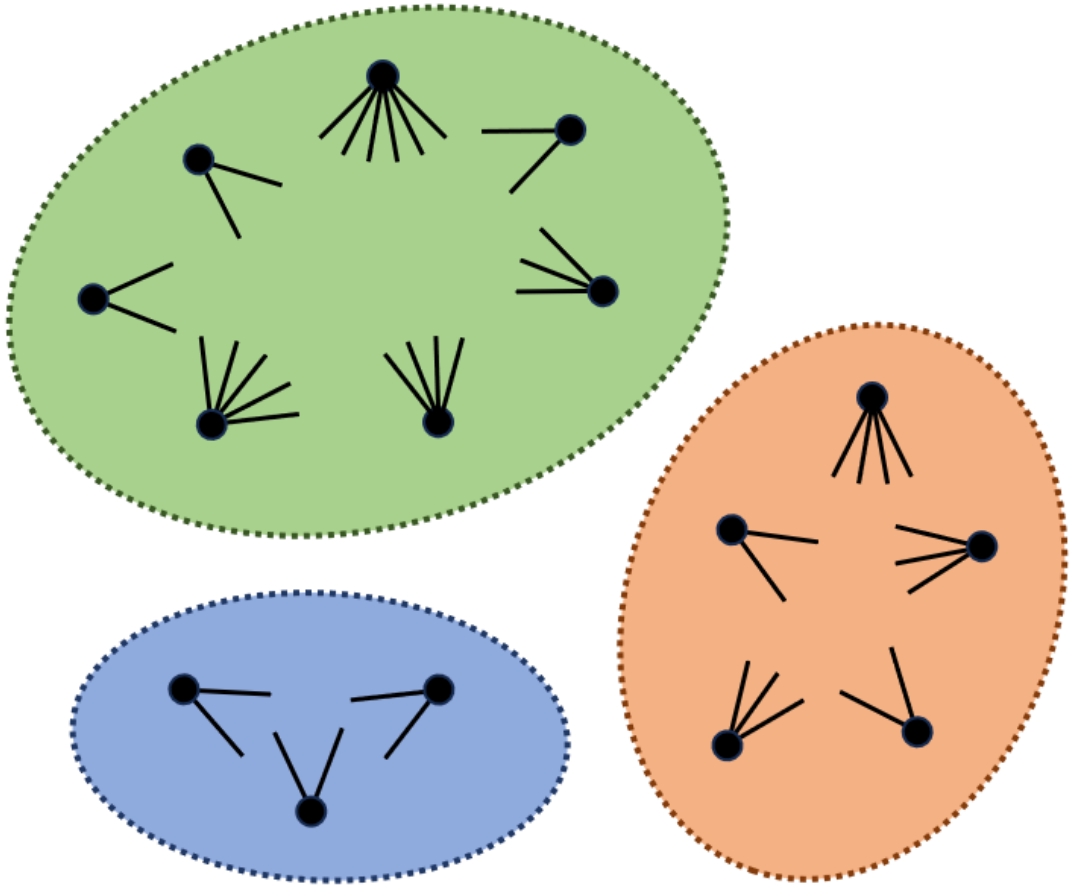
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)

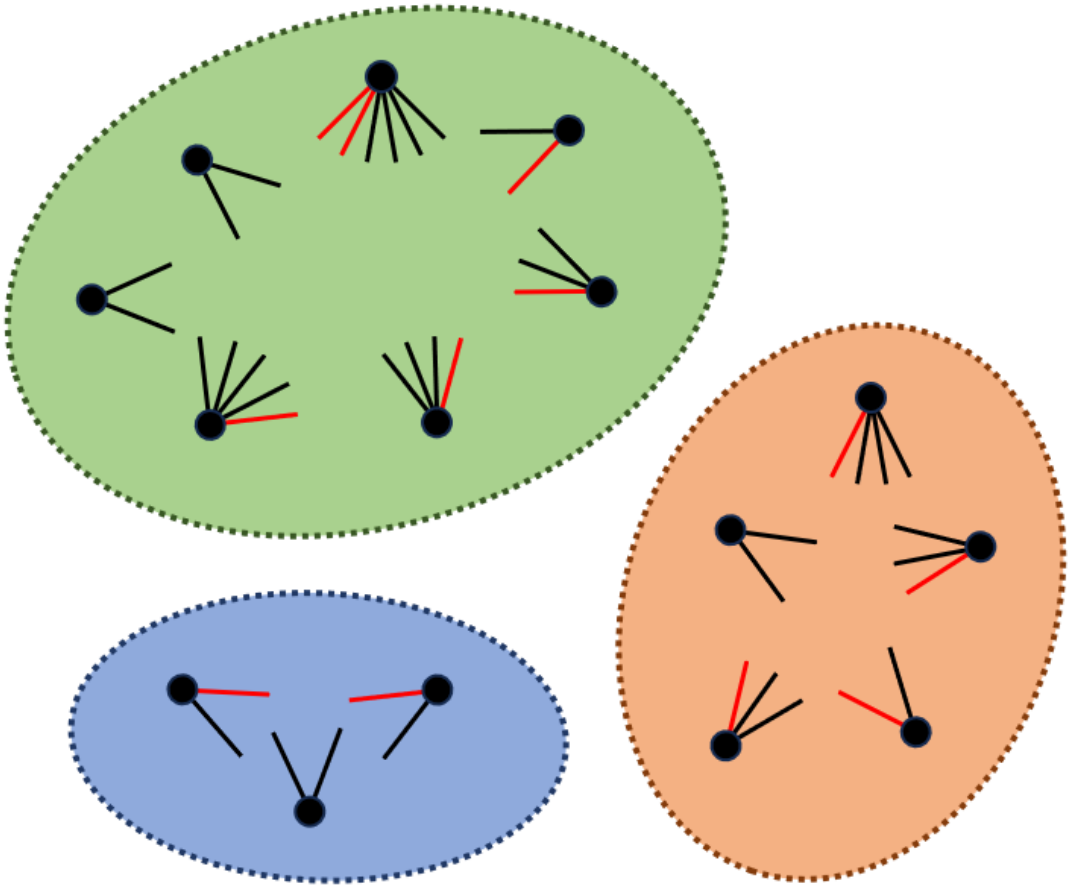


Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)

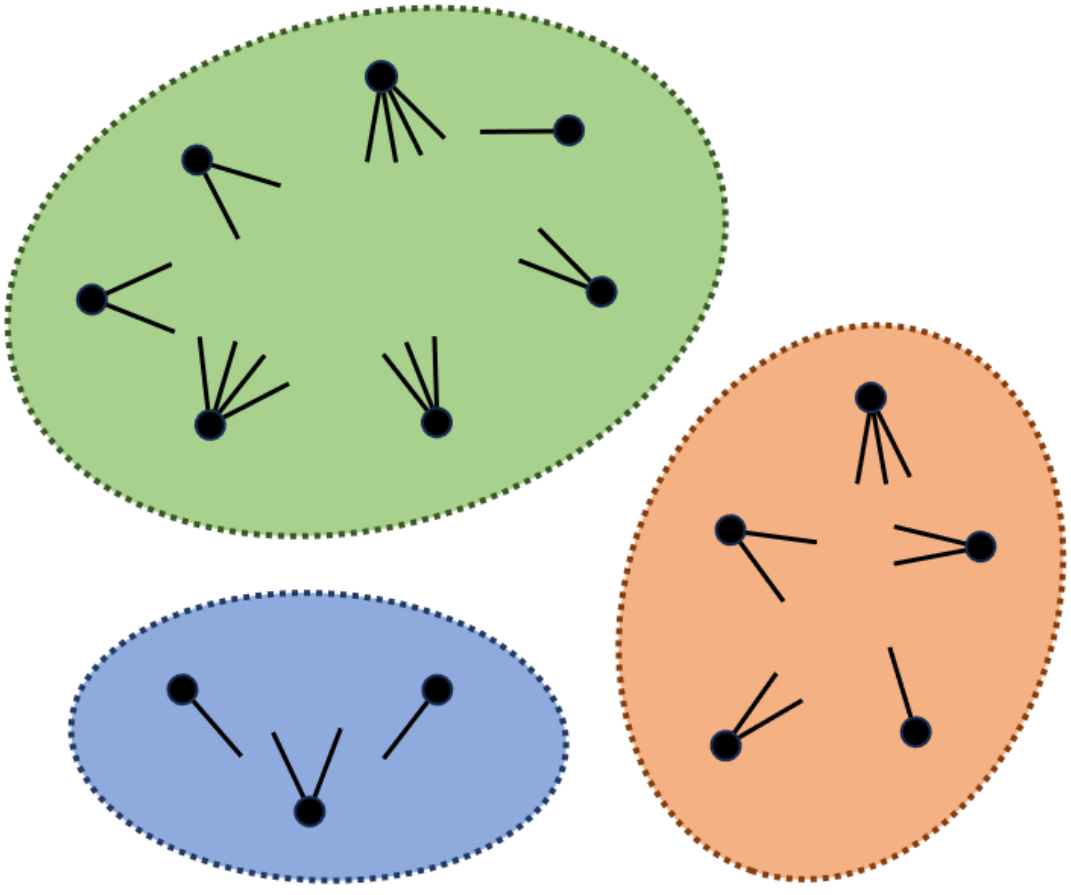


Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)

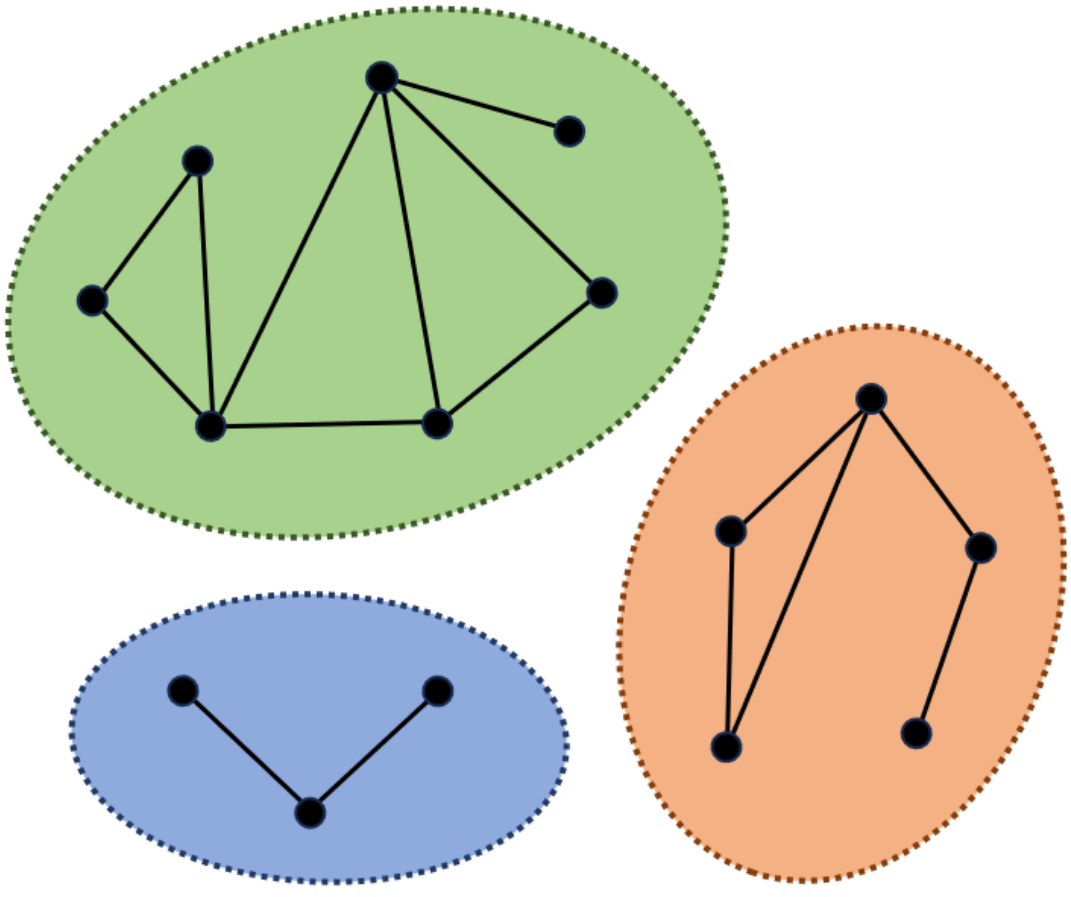
$\xi = 0.25$



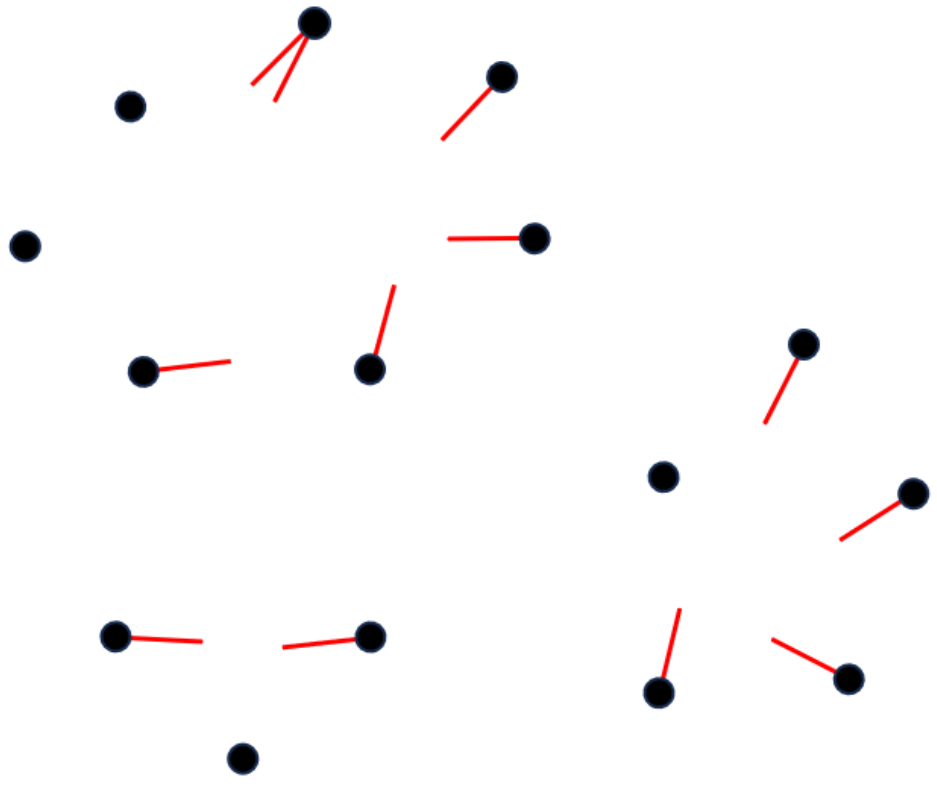
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



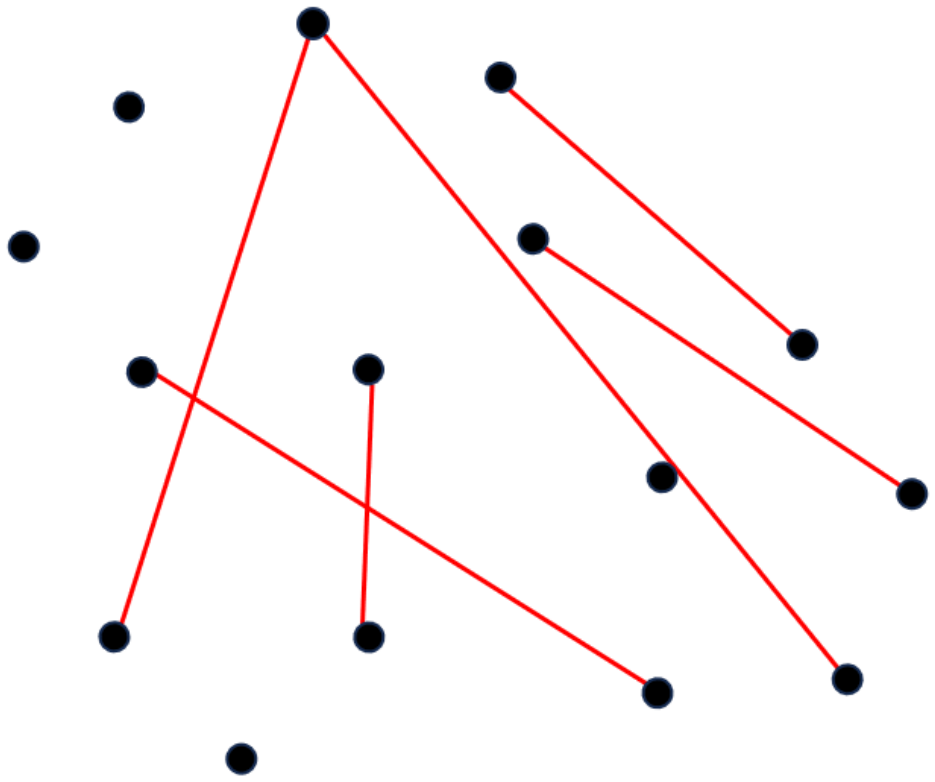
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



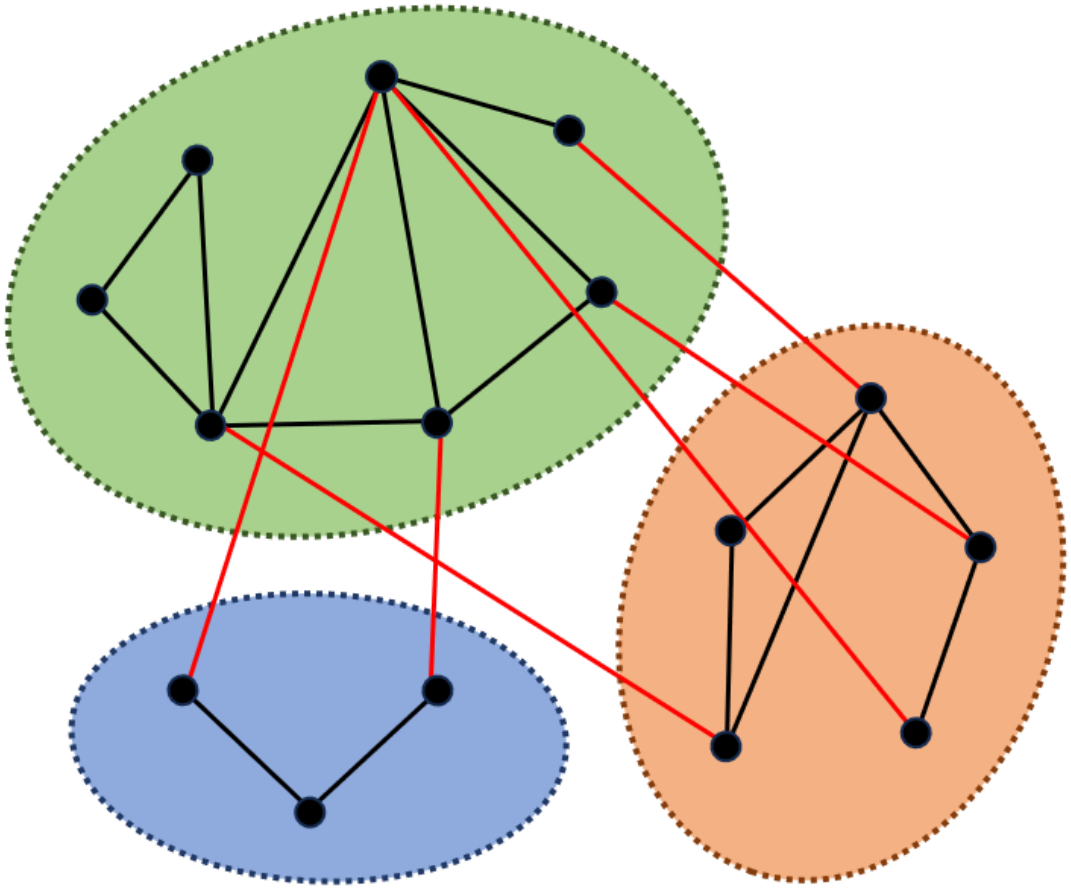
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



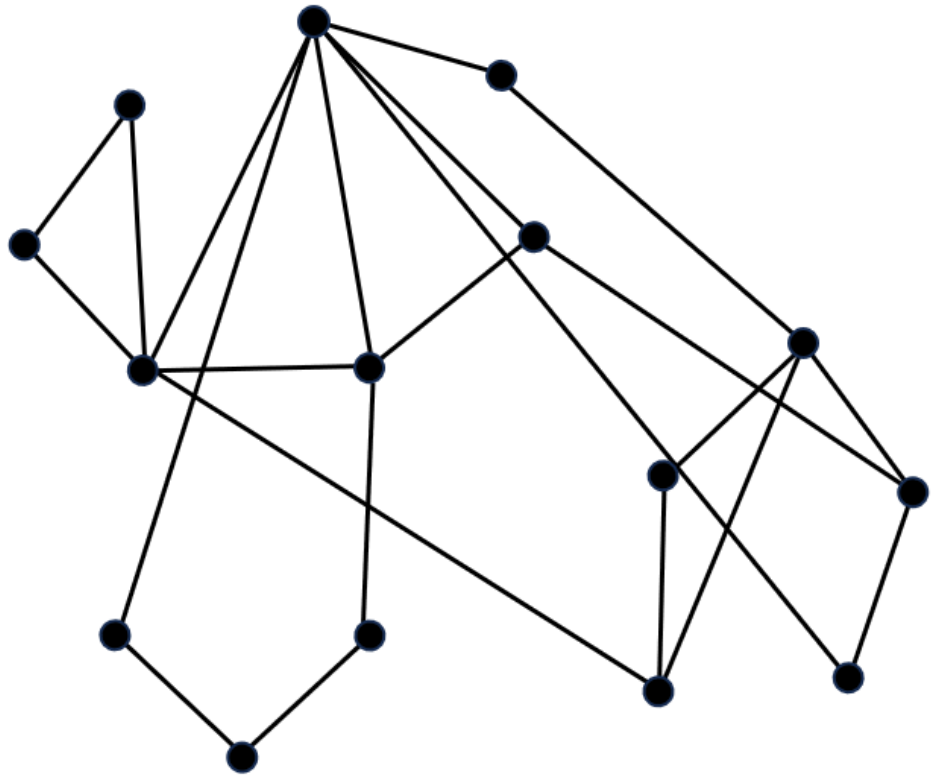
Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)



Artificial **B**enchmark for **C**ommunity **D**etection (**ABCD**)

The building blocks in the model are flexible.

- Outliers: ABCD+o
- Hypergraphs: h-ABCD
- Outliers + Overlapping: ABCD+o^2
- Multilayered networks: mABCD
- Fast implementation that uses multiple threads: ABCDe

Graph Modularity

Definition

Modularity for graphs is based on the comparison between:

a) the actual density of edges inside a community (the edge contribution), and

b) the expected density if nodes of the graph were wired randomly, regardless of community structure (the degree tax).

Definition

Modularity for graphs is based on the comparison between:

a) the actual density of edges inside a community (the edge contribution), and

b) the expected density if nodes of the graph were wired randomly, regardless of community structure (the degree tax).

Such reference random graph is known in this context as the null-model. We will use the Chung-Lu model.

Definition

Modularity for graphs is based on the comparison between:

a) the actual density of edges inside a community (the edge contribution), and

b) the expected density if nodes of the graph were wired randomly, regardless of community structure (the degree tax).

Such reference random graph is known in this context as the null-model. We will use the Chung-Lu model.

Despite some known issues with the modularity function (such as the "resolution limit"), many popular algorithms use it.

Definition

Modularity for graphs is based on the comparison between:

a) the actual density of edges inside a community (the edge contribution), and

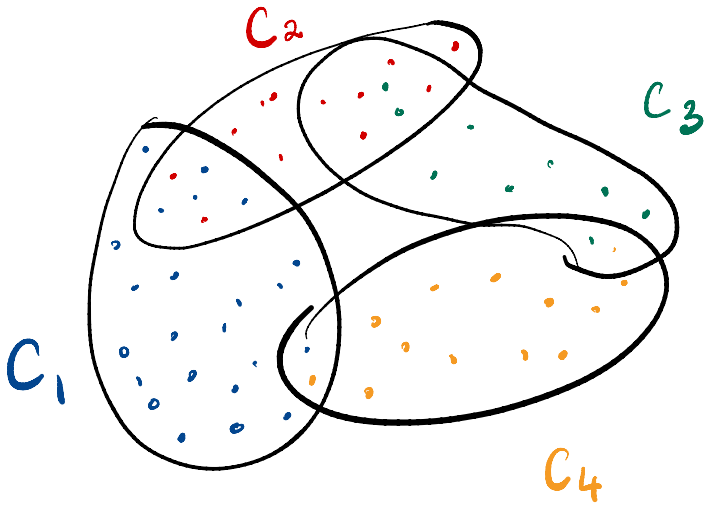
b) the expected density if nodes of the graph were wired randomly, regardless of community structure (the degree tax).

Such reference random graph is known in this context as the null-model. We will use the Chung-Lu model.

Despite some known issues with the modularity function (such as the "resolution limit"), many popular algorithms use it.

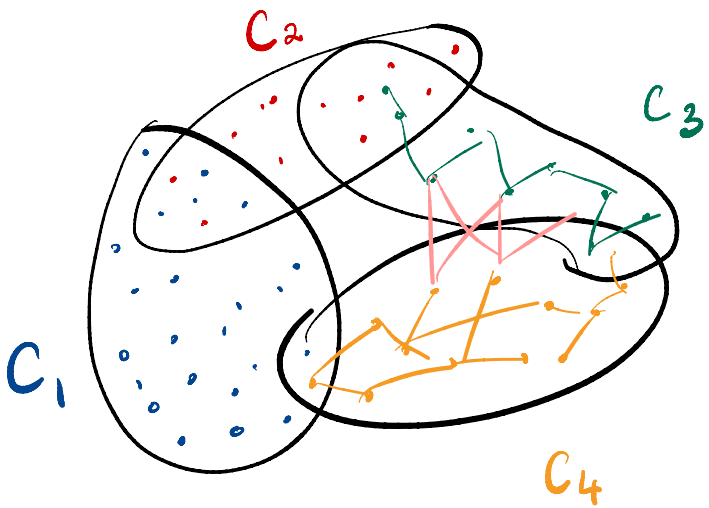
Can be easily generalized to directed graphs.

Definition



Let $\mathcal{A} = \{A_1, A_2, \ldots, A_\ell\}$ be a given partition of V.

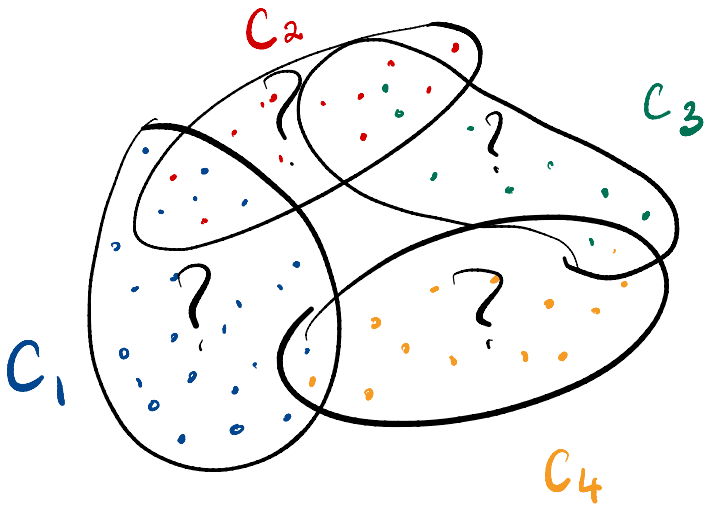
Definition



Let $\mathscr{A} = \{A_1, A_2, \ldots, A_\ell\}$ be a given partition of V.

This partition captured $\sum_{A_i \in \mathscr{A}} e_G(A_i)/|E|$ fraction of edges (edge contribution). Should we be happy with this?

Definition



Let $\mathscr{A} = \{A_1, A_2, \ldots, A_\ell\}$ be a given partition of V.

This partition captured $\sum_{A_i \in \mathscr{A}} e_G(A_i)/|E|$ fraction of edges (edge contribution). Should we be happy with this?

No! Compare it to the expected fraction of edges captured by this partition in the Chung-Lu model with the (expected) degree distribution $\mathbf{d} = \big(\deg(v_1), \deg(v_2), \ldots, \deg(v_n)\big)$.

Definition

The expected fraction of edges is equal to

$$\frac{1}{|E|}\left(\sum_{v_j v_k \in \binom{A_i}{2}} \frac{\deg(v_j)\deg(v_k)}{2|E|} + \sum_{v_j \in A_i} \frac{\deg^2(v_j)}{4|E|}\right)$$

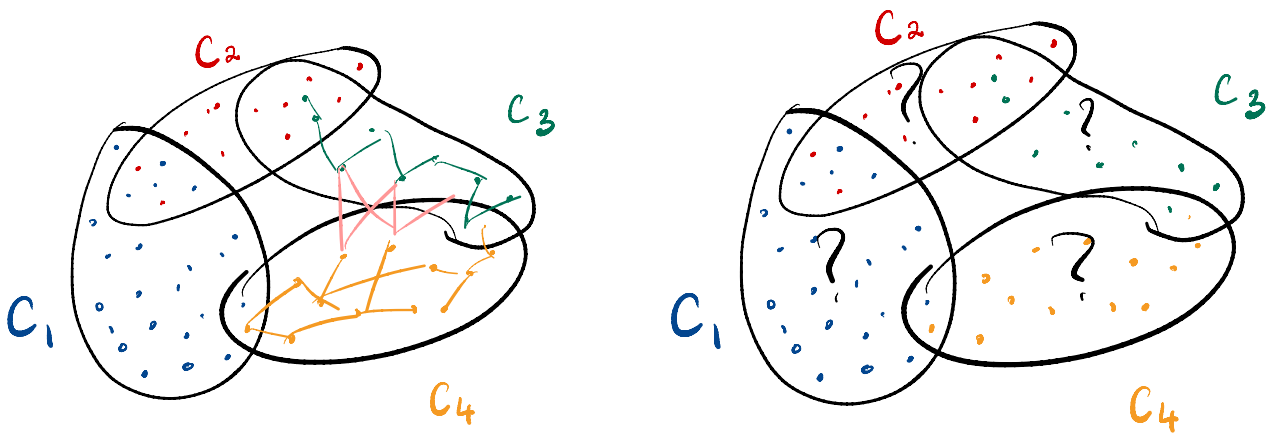
$$= \frac{1}{4|E|^2} \sum_{v_j \in A_i} \sum_{v_k \in A_i} \deg(v_j)\deg(v_k)$$

$$= \frac{1}{4|E|^2}\left(\sum_{v_j \in A_i} \deg(v_j)\right)^2 = \frac{(\mathrm{vol}(A_i))^2}{(\mathrm{vol}(V))^2}.$$

Definition

Modularity function:

$$q_G(\mathcal{A}) = \sum_{A_i \in \mathcal{A}} \frac{e_G(A_i)}{|E|} - \sum_{A_i \in \mathcal{A}} \frac{(\mathrm{vol}(A_i))^2}{(\mathrm{vol}(V))^2}$$



Definition

Modularity function:

$$q_G(\mathscr{A}) = \sum_{A_i \in \mathscr{A}} \frac{e_G(A_i)}{|E|} - \sum_{A_i \in \mathscr{A}} \frac{(\mathrm{vol}(A_i))^2}{(\mathrm{vol}(V))^2}$$

Some properties:

– $q_G(\mathscr{A}) \leq 1$,

– If $\mathscr{A} = \{V\}$, then $q_G(\mathscr{A}) = 0$,

– If $\mathscr{A} = \{\{v_1\}, \ldots, \{v_n\}\}$, then $q_G(\mathscr{A}) = -\frac{\sum \deg^2(v)}{4|E|^2} < 0$,

– $q_G(\mathscr{A}) \geq -1/2$.

Definition

Modularity function:

$$q_G(\mathscr{A}) = \sum_{A_i \in \mathscr{A}} \frac{e_G(A_i)}{|E|} - \sum_{A_i \in \mathscr{A}} \frac{(\mathrm{vol}(A_i))^2}{(\mathrm{vol}(V))^2}$$

Some properties:

– $q_G(\mathscr{A}) \leq 1$,

– If $\mathscr{A} = \{V\}$, then $q_G(\mathscr{A}) = 0$,

– If $\mathscr{A} = \{\{v_1\}, \ldots, \{v_n\}\}$, then $q_G(\mathscr{A}) = -\frac{\sum \deg^2(v)}{4|E|^2} < 0$,

– $q_G(\mathscr{A}) \geq -1/2$.

$$q^*(G) = \max_{\mathscr{A}} q_G(\mathscr{A})$$

– Well defined but impossible to find in practice *unless* G is small (then you may try Bayan algorithm).

– Used to guide a heuristic algorithms that try to maximize it.

Louvain algorithm

Initially, each node is assigned to its own community.

Louvain algorithm

Initially, each node is assigned to its own community.

Phase 1: modularity is locally optimized.

– consider nodes in a random order,

– for each node v, move v to community of some of its neighbour if it improves the modularity (can be easily and efficiently calculated!),

– if no increase is possible after considering all nodes, a local maximum value is achieved and the first phase ends.

Louvain algorithm

Phase 2: grouping nodes.

– contract nodes from each community into a single node (super-node),

– all edges within that community are replaced by a single weighted loop,

– all edges between two communities are replaced by a single weighted edge,

– typically, the resulting graph becomes much smaller.

Louvain algorithm

Phase 2: grouping nodes.

– contract nodes from each community into a single node (super-node),

– all edges within that community are replaced by a single weighted loop,

– all edges between two communities are replaced by a single weighted edge,

– typically, the resulting graph becomes much smaller.

Repeat the two phases until no improvement on the modularity function can be further achieved. The first pass is typically the most time consuming part of the algorithm.

Leiden algorithm

One problem with Louvain algorithm is that once super-nodes are created, the associated nodes will stay together forever.

Leiden algorithm

One problem with Louvain algorithm is that once super-nodes are created, the associated nodes will stay together forever.

Solution: Leiden algorithm—periodically randomly breaking down communities into smaller well-connected ones.

Ensemble Clustering algorithm for Graphs (**ECG**)

Louvain is a randomized algorithm: can be unstable, that is, the results of independent runs of the same algorithm on the very same data can vary a lot.

Ensemble Clustering algorithm for Graphs (**ECG**)

Louvain is a randomized algorithm: can be unstable, that is, the results of independent runs of the same algorithm on the very same data can vary a lot.

Solution: use ensemble clustering often referred to as consensus clustering.

Ensemble Clustering algorithm for Graphs (**ECG**)

Start with ℓ randomized level-1 partitions $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_\ell$ obtained by ℓ independent runs of the first phase of the Louvain algorithm and only once.

Ensemble Clustering algorithm for Graphs (**ECG**)

Start with ℓ randomized level-1 partitions $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_\ell$ obtained by ℓ independent runs of the first phase of the Louvain algorithm and only once.

Assign new weights to edges as follows: for each $uv \in E$,

$$w(uv) = \begin{cases} w_* + (1 - w_*) \cdot \frac{1}{\ell} \sum_{i=1}^{\ell} \alpha_i(u, v), & \text{if } uv \text{ is in the 2-core} \\ w_* & \text{otherwise,} \end{cases}$$

$w_* \in [0, 1]$ is the parameter of the algorithm, and $\alpha_i(u, v) = 1$ if u and v appear together in some part of $\mathcal{P}_i$; otherwise, $\alpha_i(u, v) = 0$. (Default values are $\ell = 16$ and $w_* = 0.05$.)

Ensemble Clustering algorithm for Graphs (**ECG**)

Start with ℓ randomized level-1 partitions $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_\ell$ obtained by ℓ independent runs of the first phase of the Louvain algorithm and only once.

Assign new weights to edges as follows: for each $uv \in E$,

$$w(uv) = \begin{cases} w_* + (1 - w_*) \cdot \frac{1}{\ell} \sum_{i=1}^{\ell} \alpha_i(u, v), & \text{if } uv \text{ is in the 2-core} \\ w_* & \text{otherwise,} \end{cases}$$

$w_* \in [0, 1]$ is the parameter of the algorithm, and $\alpha_i(u, v) = 1$ if u and v appear together in some part of $\mathcal{P}_i$; otherwise, $\alpha_i(u, v) = 0$. (Default values are $\ell = 16$ and $w_* = 0.05$.)

Perform Louvain (till the very end, *not* just level-1) on a weighted version of the initial graph.

THE END