Monday, March 9, 2026 8:46 AM

Community Detection (CD)

↳ most classic graph mining/net sci. problem in the literature

aka "clustering"

Loose definition: identifying "dense" subgraphs in some larger network

We consider communities in multiple different ways:

- Friend group
- Shared interest
- Basic interactions

→ overall, data-dependent

Classic example:

Zachary Karate Club

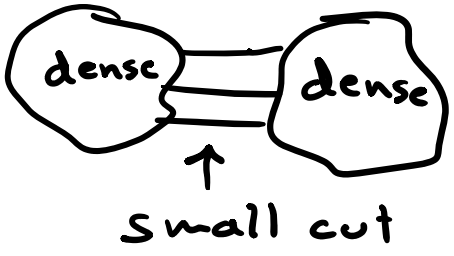
↳ individuals in karate club

edges defined friendships outside of the club

The club split in two

↳ This split followed the topology of the network

Two dense subgraphs, loosely coupled together



↳ Fundamental hypothesis of community detection

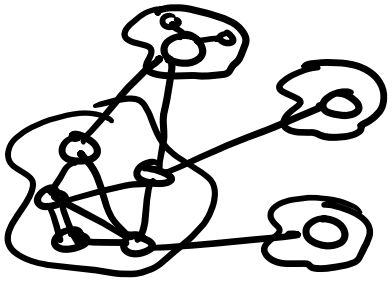
* We can define communities solely based on network topology

What we'll be discussing:

- How to define "density"
- Algorithms used for CD
- How to evaluate CD algo. outputs
- Related problems (graph partitioning)

Density Definitions

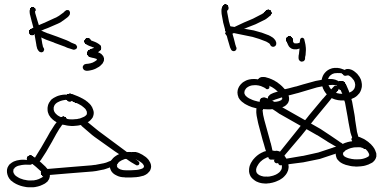
"Density" ≈ lot of edges internal to some subgraph relative to external edges



community

How dense can we get?

↳ cliques (simple graphs



(simple graphs) 

Can we define communities using cliques?

Issue: large cliques are very rare in real networks

Issue: computationally difficult
* NP-complete in general
* equivalent to subgraph matching
* $O(1.19^n)$ bound for cliques

However:

triangles (K_3) can be extremely useful
* density ~ clustering coefficient
* complexity is better $O(m^{3/2})$ ($m = |E(G)|$)

Issue: extrapolating local clustering to a broader subgraph

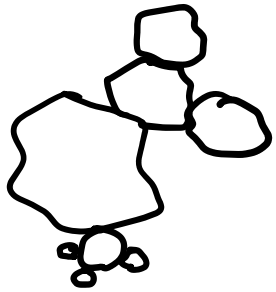
↳ communities can have a...

↳ communities can have an arbitrarily large diameter

Can we use <u>connectivity</u>?

* Connected components → not useful
* k-connected subgraphs → more useful

↗ maximal k-connected subgraph, still connected after k vertices removed



Note: captures the notion of cuts
→ inherently captures "density"

Issue: tough to compute O(poly)

Issue: doesn't necessarily correspond to our assumed definition of communitres

↳ differing scales of resolution

K-cores/shells: some issues

Better definition: relative edge density

↳ ratio of internal to external edges

strong community: $\forall v \in C : d_{int}(v) \geq d_{ext}(v)$ (community C)

weak community: $\sum_{\forall v \in C} d_{int}(v) \geq \sum_{\forall v \in C} d_{ext}(v)$

↳ Both give explicit measures of density

ANO: a global quality measure

Issue: trivial optimal solutions is just a singular community

improved measures:

modularity ⁉ conductance

(next class)

Number of Communities

Number of communities is generally unknown, will vary based on data

Challenges:

* We have a full "ground truth"
* Possible per-vertex groupings given same group distributions → exponential
* Possible groupings → super-exponential

(exponential)(super-exponential) = "quite large"

Takeaway: heuristics, greedy algos, and approximations are used in practice

---

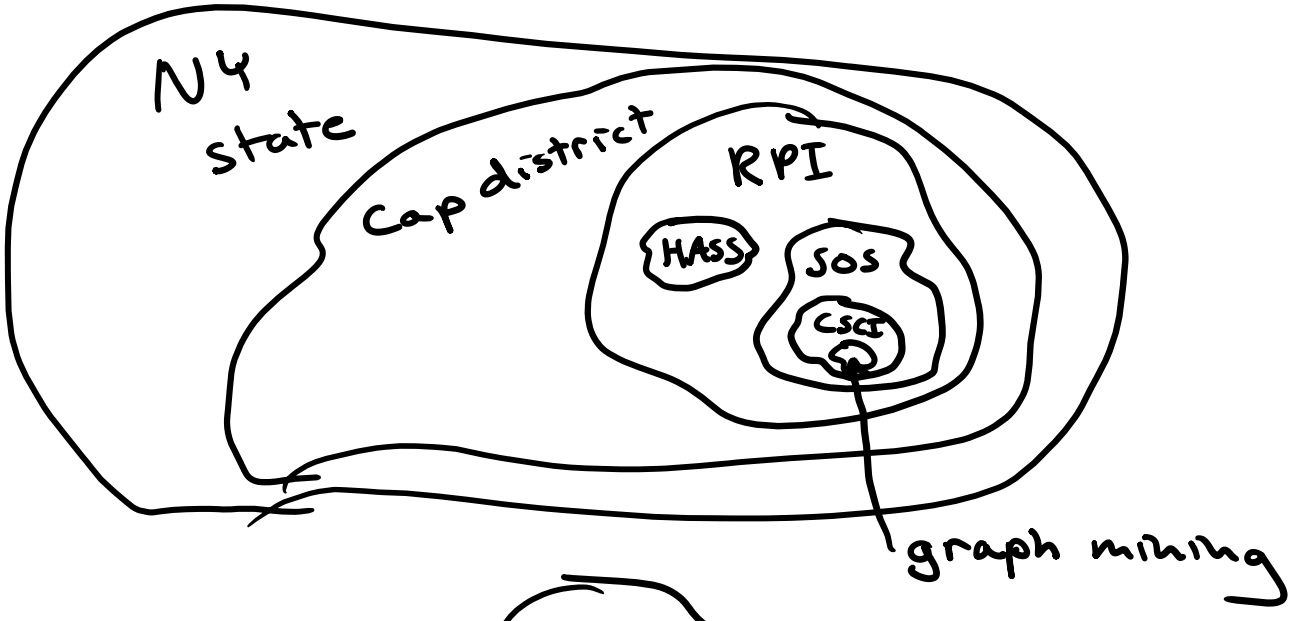
CD Algorithm classes

* Agglomerative: we combine vertices/communities to reach some local maxima for optimization

* Divisive: we cut communities in some way to reach local maxima for optimization

* Hierarchical: we create a hierarchical structure of communities with defined sub/super relationships

→ often the case in practice



Note:

communities can overlap

Communities can overlap

↳ can consider multiple outputs or multiple labeling per-vertex

---

Label Propagation

Agglomerative, can be hierachical

Iterative:

For all $v \in V(G)$:

$C(v) =$ max community label that shows up in $N(v)$

Pros: simple to implement

~$O(n)$ complexity

good in practice

Cons: can have bad results

↳ single comm. output

hierarchy is tough to infer

Rauasz Algorithm

Agglomerative

Iterate until a single community

select some (C_i, C_j) communities to merge based on a similarity metric (common neighbors)

Pros: full hierarchy captured

can modify our similarity metric

Cons: complexity not great $O(n^2)$

no global optimization metric

Girvan-Newman

Divisive

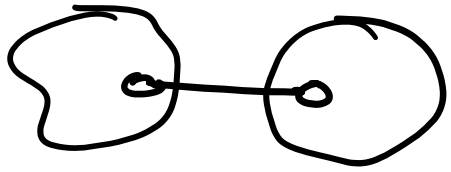
Iterate:

select e (edge) with the highest betweenness centrality and cut it

and cut it

↳ communities are connected comps.

Note: we're cutting weak ties
(or local bridges)



Pros: can work well, as it utilizes a lot of basic network growth properties

Cons: slower $O(n^3)$