SAND2015-3664C

Sandia National Laboratories

Finding the Hierarchy of Dense Subgraphs using Nucleus Decompositions

A. Erdem Sarıyüce[*], C. Seshadhri[+], Ali Pınar[#], Ümit V. Çatalyürek[*]

[*] The Ohio State University

[+] University of California, Santa Cruz

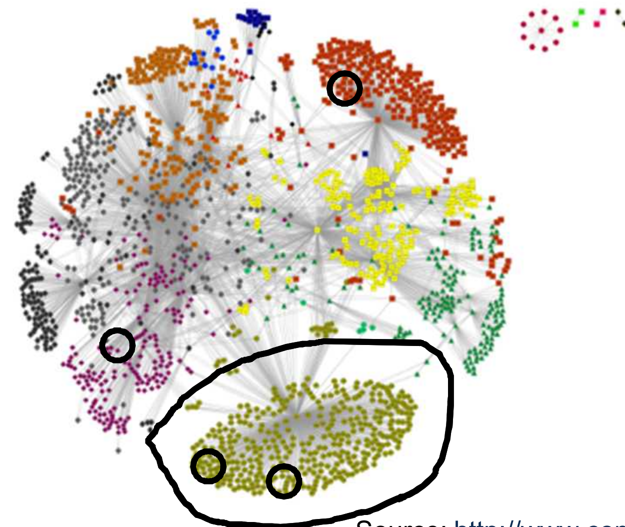
[#] Sandia National Labs, Livermore

U.S. DEPARTMENT OF ENERGY

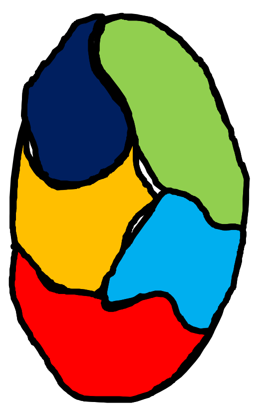
Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Graphs are globally sparse… yet locally dense.

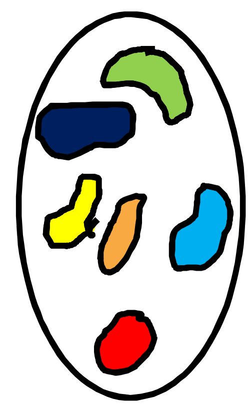
- Graphs in real world are SPARSE
  - Number of vertices = millions
  - Number of edges ≈ 10 X vertices
    - Two random vertices unlikely to be connected (prob = 10^{-5})
- But they contain many dense substructures
  - Within dense region, two random vertices highly likely to be connected (prob = 0.4)



Source: http://www.complexworld.net/virthulab/ongoing-projects-main



Community detection: label most/all vertices



Dense subgraph discovery: Regions with lots of "activity"

Many applications find dense subgraphs

Sandia National Laboratories

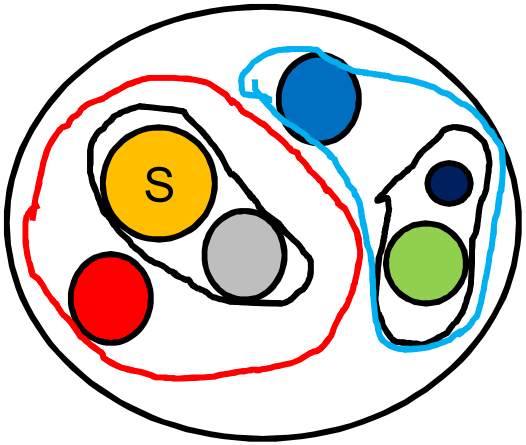
List is long, time is short. Why don't you just trust me?

- Finding communities, spam link farms [Gibson et al., 2005]
- Graph visualization [Alvarez-Hamelin et al., 2006]
- Real-time story identification [Angel et al., 2012]
- DNA motif detection [Fratkin et al., 2006]
- Finding correlated genes [Zhang and Horvath, 2005]
- Finding price value motifs in financial data [Du et al., 2009]
- Graph compression [Buehrer and Chellapilla, 2008]
- Distance query indexing [Jin et al., 2009]
- Throughput of social networking sites [Gionis et al., 2013]
- To name a few…

Dense subgraphs

Concept is intuitive, yet formalizations are tricky

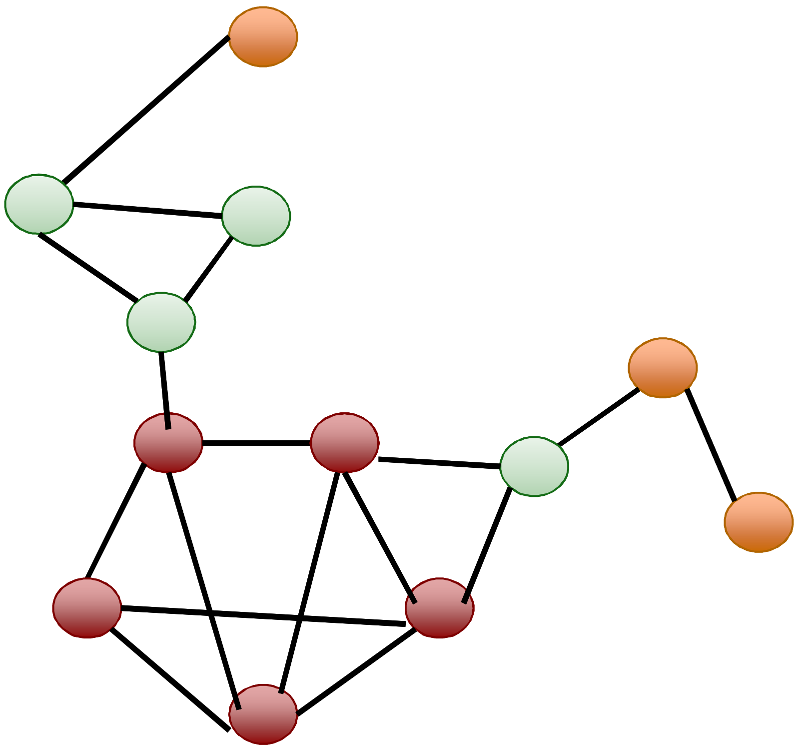
- Factors for consideration: size, density of internal edges, density of external edges
- Many formalizations lead to NP-hard problems, and heuristics are used.
- Hard to distinguish , whether an observation is an artifact of the heuristic or not.



- Our goal:
  - Can we formulate the problem such that the result is well-defined?
  - Can we find all dense graph not just the densest?
  - Is there a “natural” hierarchy of dense subgraphs?
  - Can we design efficient, provable algorithms and minimize heuristics/approximations?

K-cores in graphs

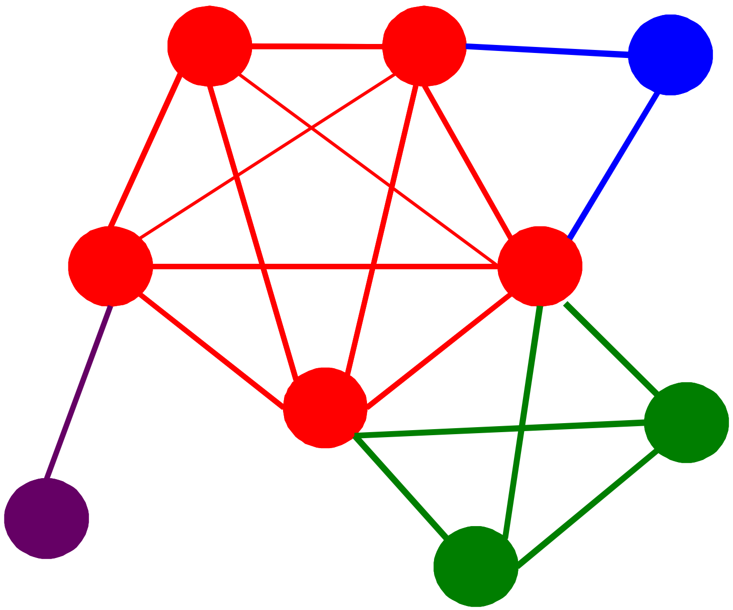
Unit of observation: vertex
Witness: Edge



- *k*-core of a graph is its largest induced subgraph, where degree of each vertex is at least *k*.
- Introduced by [Matula and Beck, 1983]
- Algorithm
  - Compute degrees
  - Iterative removal in increasing order
    - Assign K value during removal
- **O(|E|) complexity**

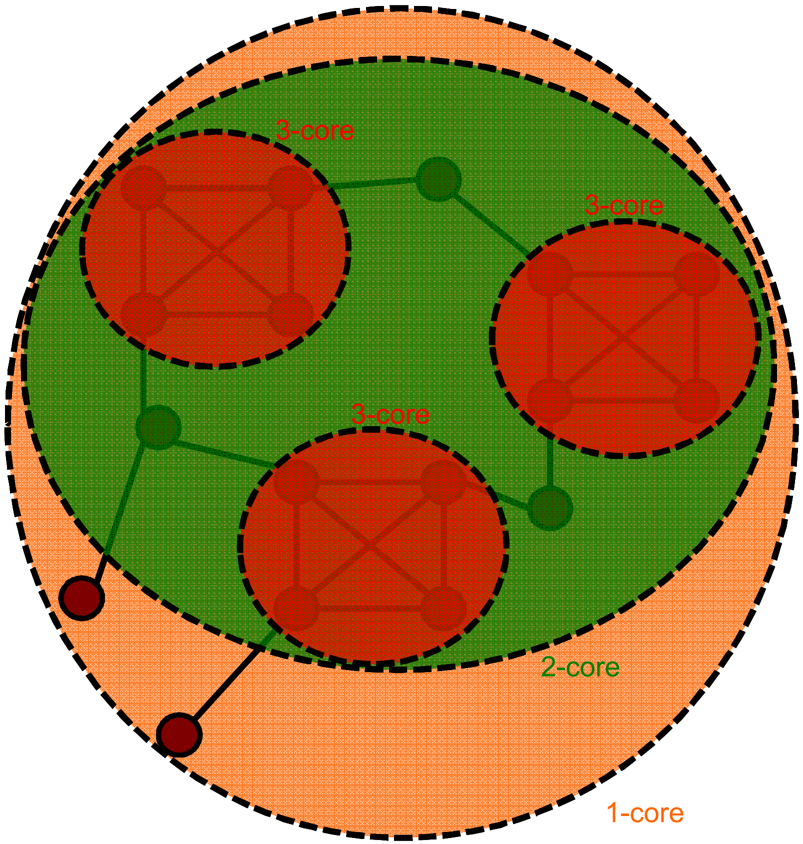
K-truss decompositions go a step further

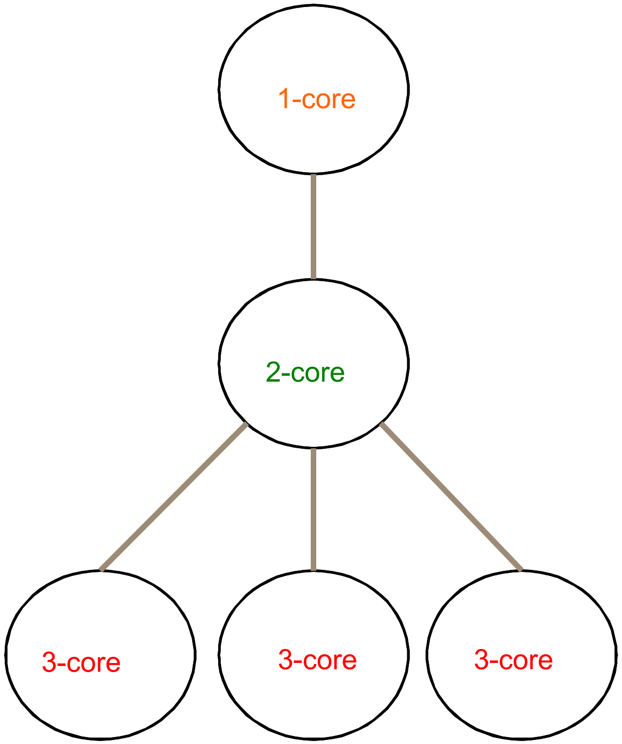
Unit of observation: Edge
Witness: Triangle



- *K*-truss of a graph is its largest induced subgraph, where each edge participates in at least *k triangles*.
- Introduced by Cohen and Parthasarathy independently.
- Applied to visualization and dense graph finding

Decompositions lead to hierarchies





Caveat: k-core decomposition typically leads to long chains as opposed to well-branched trees.

The definition of nuclei

DEFINITION 1. *Let $r < s$ be positive integers and $\mathcal{S}$ be a set of K_ss in G.*

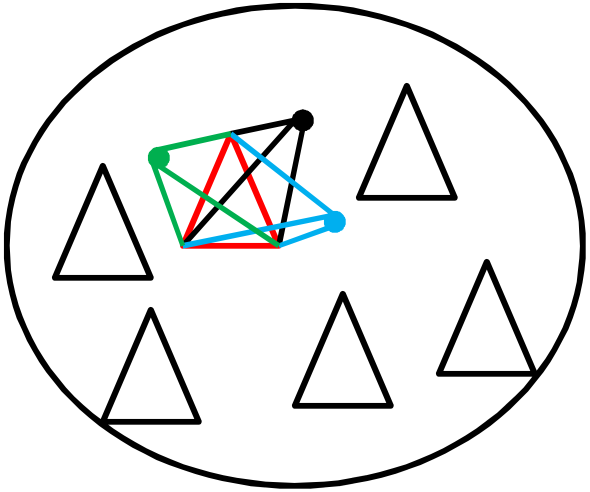
- *$K_r(\mathcal{S})$ the set of K_rs contained in some $S \in \mathcal{S}$.*
- *The number of $S \in \mathcal{S}$ containing $R \in K_r(\mathcal{S})$ is the* $\mathcal{S}$-degree *of that K_r.*
- *Two K_rs R, R' are* $\mathcal{S}$-connected *if there exists a sequence $R = R_1, R_2, \ldots, R_k = R'$ in $K_r(\mathcal{S})$ such that for each i, some $S \in \mathcal{S}$ contains $R_i \cup R_{i+1}$.*

DEFINITION 2. *Let k, r, and s be positive integers such that $r < s$. A* k-(r, s)-nucleus *is a maximal union $\mathcal{S}$ of K_ss such that:*

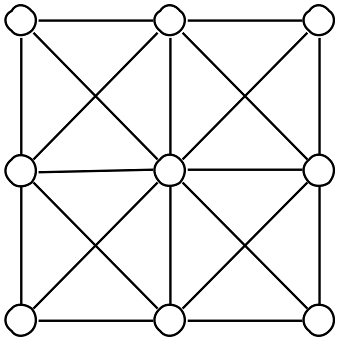
- *The $\mathcal{S}$-degree of any $R \in K_r(\mathcal{S})$ is at least k.*
- *Any $R, R' \in K_r(\mathcal{S})$ are $\mathcal{S}$-connected.*

r refers to the size of the unit of observation
s refers to the size of the witness + unit
k is the number of witnesses; not a parameter we sweep through *k*

Examples of nuclei

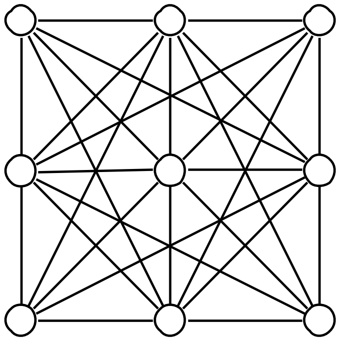


Edge (2-clique) and
3-clique interaction



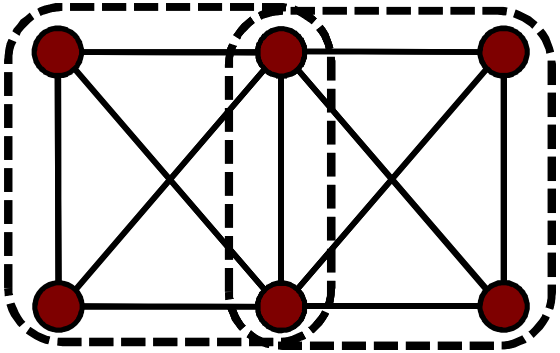
2-(2,3)
nucleus

Edge (2-clique) and
4-clique interaction



2-(2,4)
nucleus

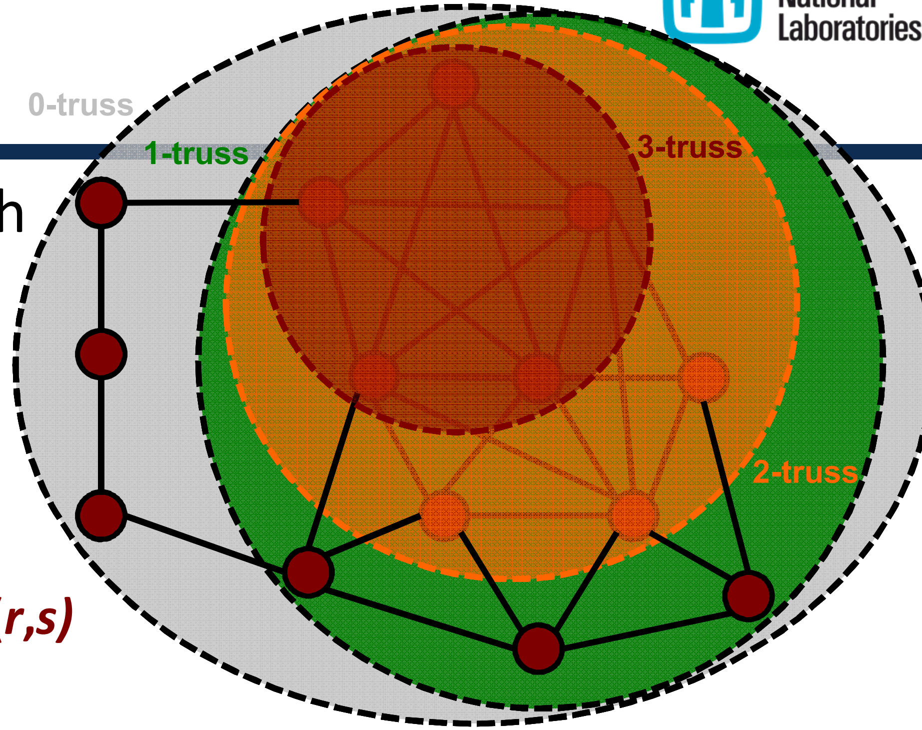
- *k*-(3,4) nucleus: subgraph formed by maximal union of triangles. Every triangle in at least *k* four-cliques
- k-(1,2) is core decomposition
- k-(2,3) is truss decomposition



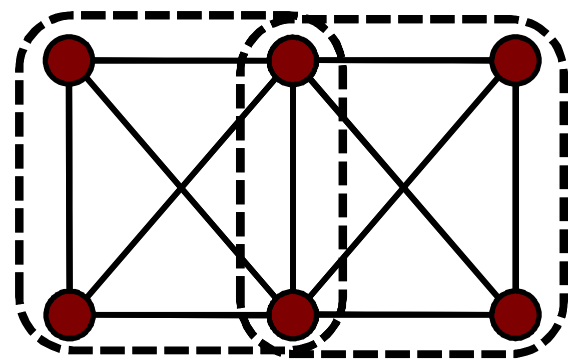
Two 1-(3,4) nuclei

Properties of nuclei decomposition

- Well-defined property of the graph
  - Not heuristic
  - No optimization
  - **Deterministic**
- Forest of nuclei
  - **Smaller *k*-(*r,s*) contained in larger *k*-(*r,s*)**
  - Hierarchy of dense subgraphs
    - Finding many and understanding relations
- **Overlaps of nuclei**
  - **For r >= 2,** lower orders structure can be shared among nuclei
  - No overlaps for *k*-cores!

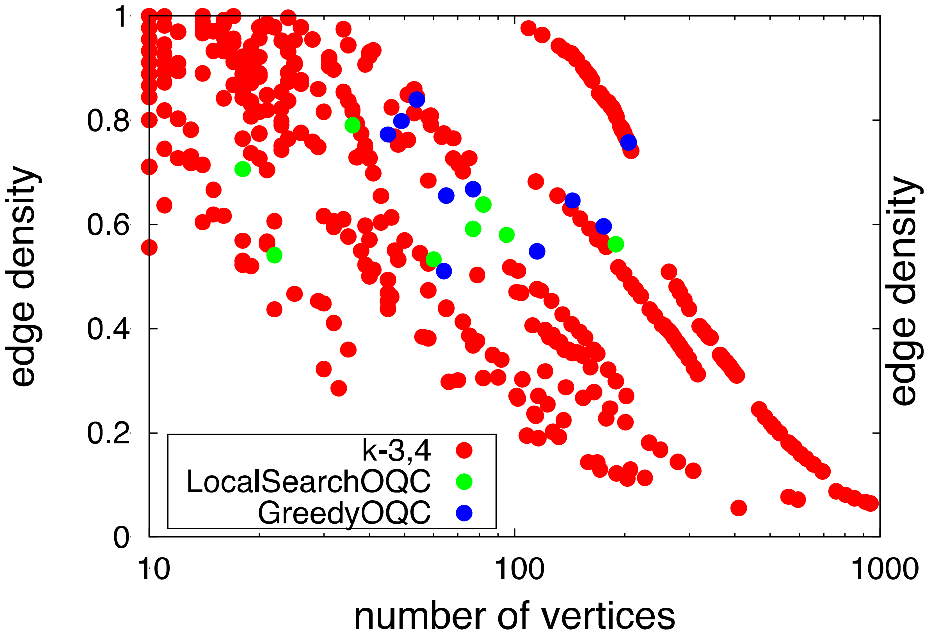


k-truss hierarchy

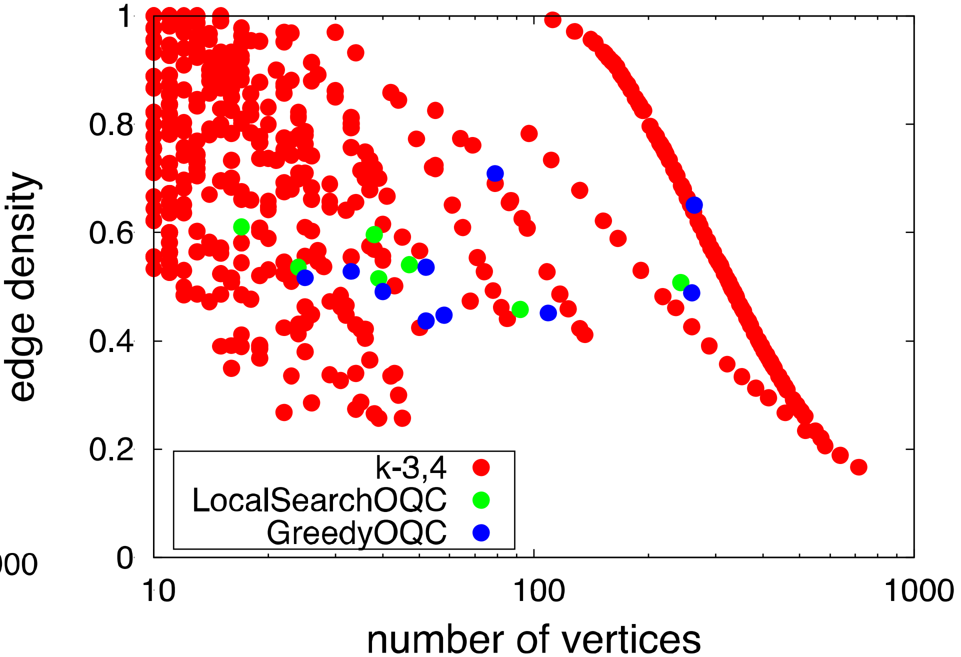


Two 1-(3,4) nuclei

Nucleus decomposition finds dense subgraphs



Facebook

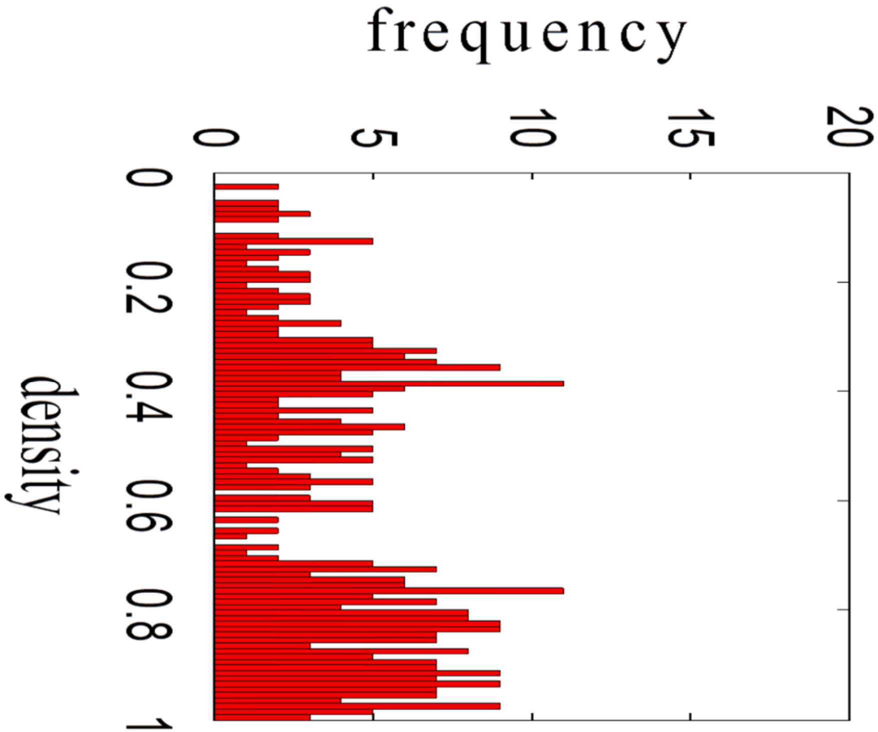


Soc-Epinions

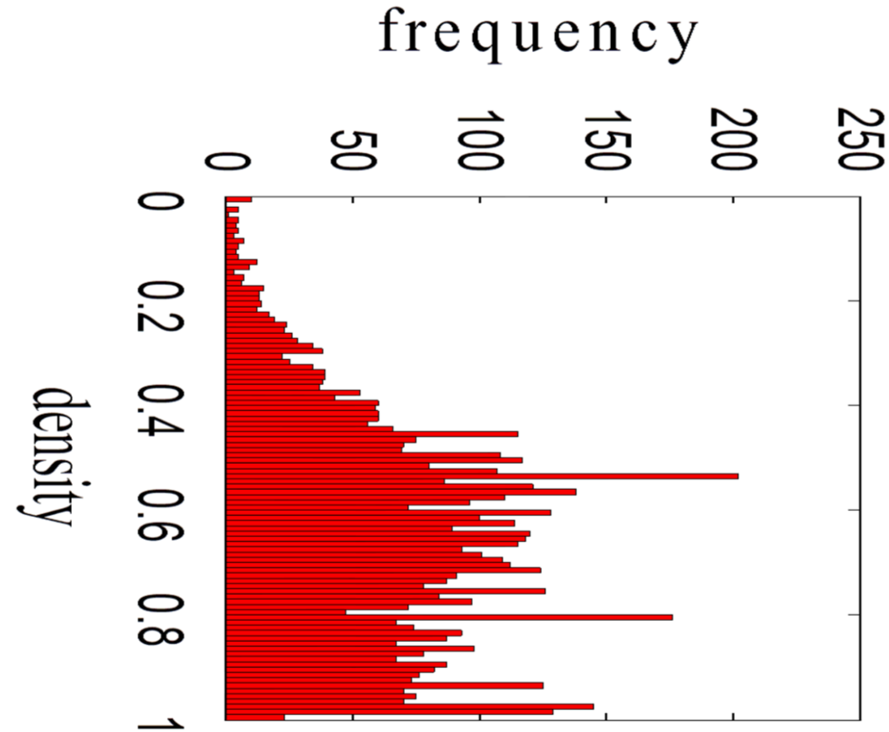
Density of S = $E(S,S)/\binom{S}{2}$

- We can find many dense subgraphs, not just one at the same time.
- Solution qualities can match the state of the art tools.

Distributions of dense structures



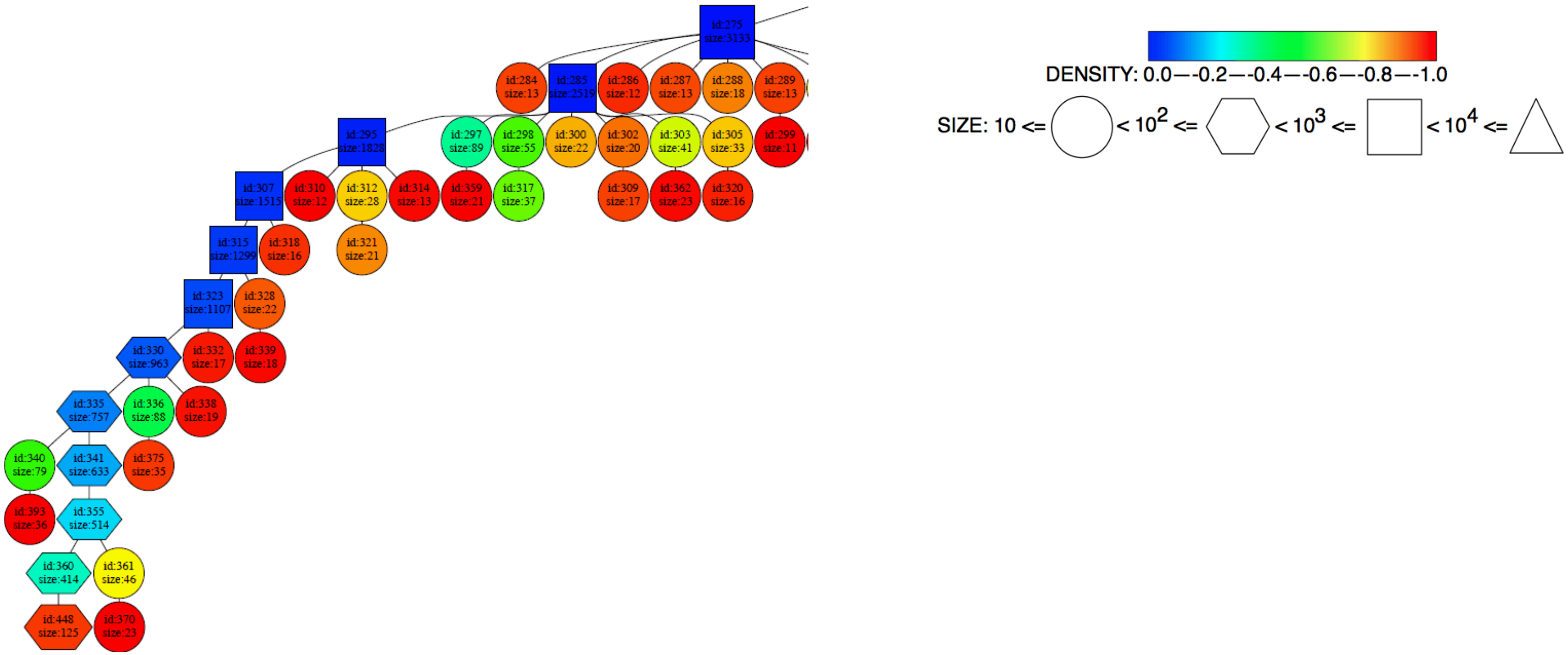
Facebook



wikipedia

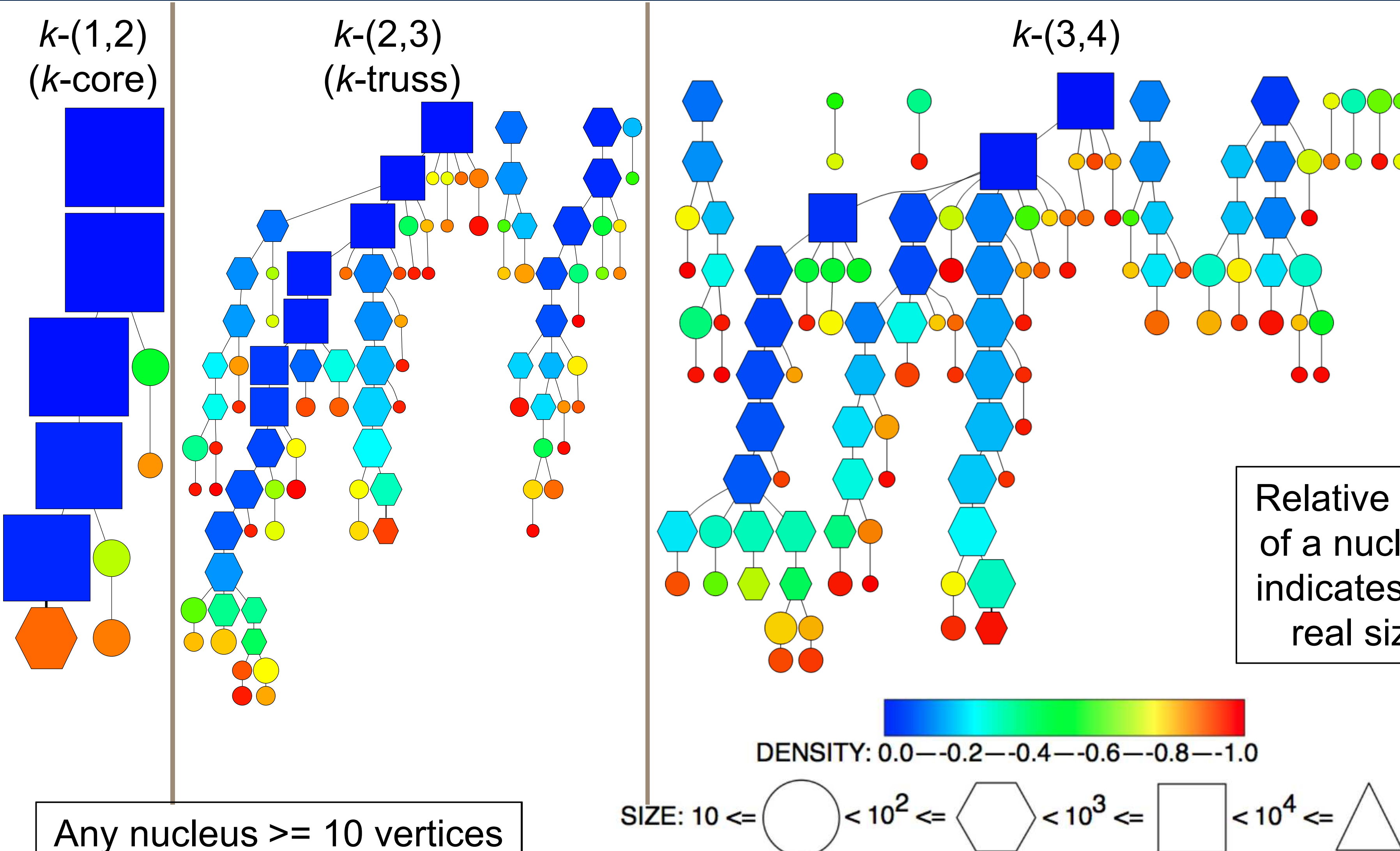
- Finding many dense structures enables producing a density structure profile.

Hierarchy reveals structure among communities.



- Results on experimental protein interaction data from Baylor College of Medicine.
- More than 50K vertices, 400K edges, but only few hundred nuclei, with tree of size 50

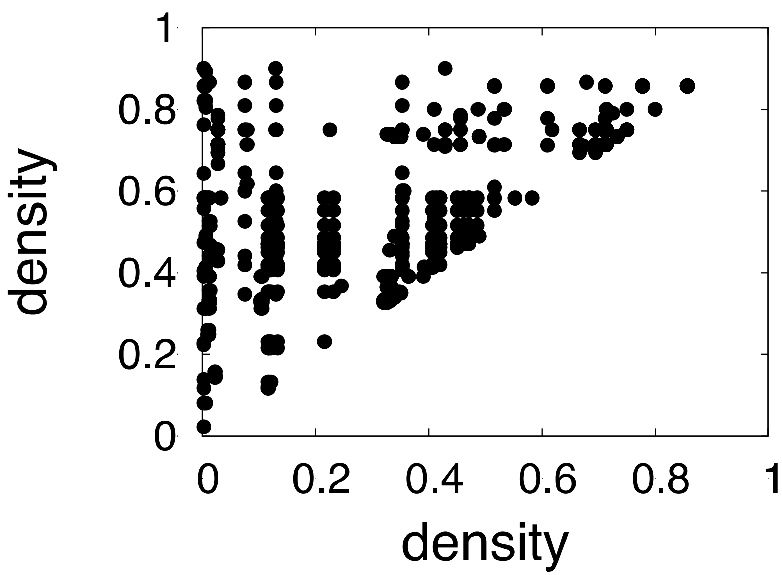
Hierarchies (facebook |V|: 4K, |E|: 88K)



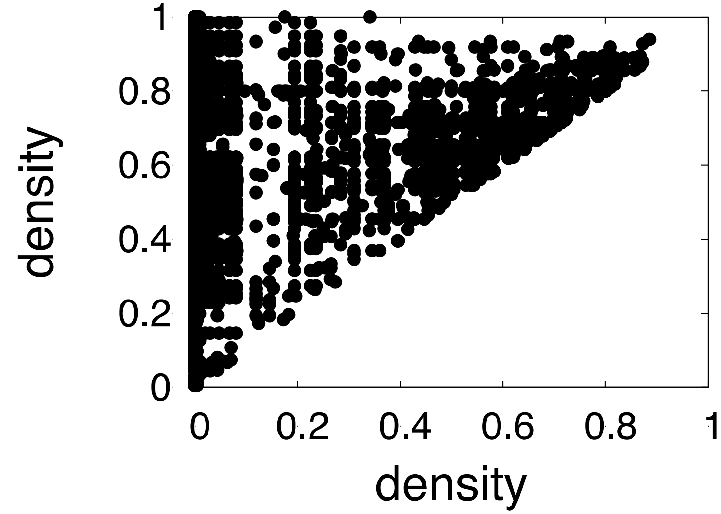
Relative size of a nucleus indicates the real size

Any nucleus >= 10 vertices

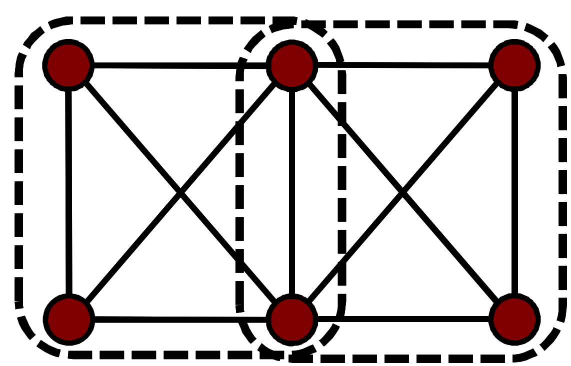
Many dense structures overlap



Web-NotreDame



Wikipedia



- Overlap size is at least 5

How to compute Nucleus Decomposition

Sandia National Laboratories

- Given *r* and *s*, **find all *k*-(*r*,*s*) nuclei**
- Just like *k*-core decomposition
- **Find K values of all K_rs**

Algorithm 1: set-k(G, r, s)

Enumerate all K_rs and K_ss in $G(V,E)$;
For every K_r R, initialize $\delta(R)$ to be the number of K_ss containing R;
Mark every K_r as unprocessed;
for each *unprocessed* K_r R *with minimum* $\delta(R)$ **do**
    $\kappa(R) = \delta(R)$;
    Find set $\mathcal{S}$ of K_ss containing R;
    for each $S \in \mathcal{S}$ **do**
        if *any* K_r $R' \subset S$ *is processed* **then**
            Continue;
        for each K_r $R' \subset S$, $R' \neq R$ **do**
            if $\delta(R') > \delta(R)$ **then**
                $\delta(R') = \delta(R') - 1$;
    Mark R as processed;
return array $\kappa(\cdot)$;

Two ways to implement:

- Enumerate all K_rs and K_ss
  - Not feasible for large *r*, *s*
  - Huge space complexity
- Construct adj. lists of K_rs online (only enumerate K_rs)
  - **Better space complexity**
  - **Time complexity is**

$$O(RT_r(G) + \textstyle\sum_v ct_r(v)d(v)^{s-r})$$

Total num of K_rs

Num of K_rs of v

Degree of v

Future Directions

- **Applications of nucleus decomposition**
  - Protein-protein and protein-gene interaction networks
  - Ongoing collaboration
- Larger values of *r* and *s*
  - Computational cost of increasing r and s is significant.
  - **Preliminary experimentation for (4,5)**
    - Very little quality benefit
  - **Is (3,4) a sweet spot?**
- Faster *k*-(3,4)
  - Clique enumeration
  - **Parallel algorithms**
    - GPU implementation of *k*-core [Jiang et al., 2014]
    - Pregel algorithm for *k*-truss [Shao et al., 2014]

Conclusions

- Nucleus decomposition is a generalization of k-core and k-truss decompositions
- It can identify many dense structures at the same time.
  - Competitive with state of the art algorithms that find a single dense structure
- It can provide a hierarchy of density structures
- It can find overlapping dense structures
- The hierarchical structure is unique; it is a property of the graph, not the algorithm being used
- Runtimes are in the order of hours for > million vertex graphs.

Questions