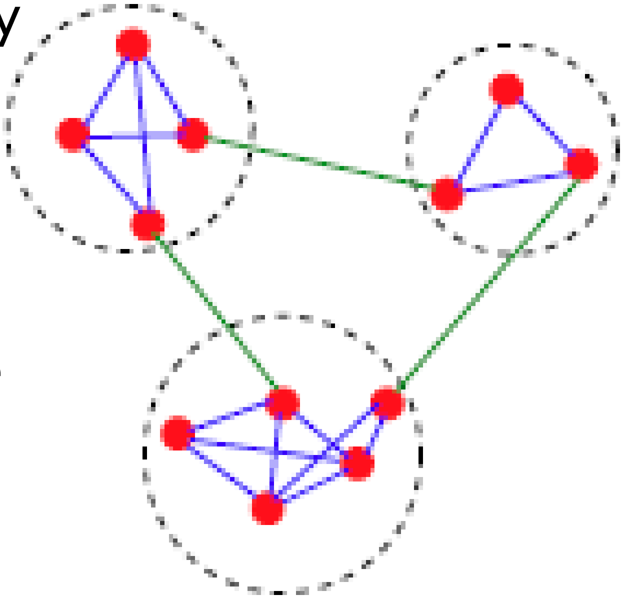
Community Detection: Clustering

Peter M. Kogge

Real World Graphs

- Subsets of vertices tend to "**cluster**"
  - A.K.A form "**Communities**"
- Communities then studied independently
  - Look at structure of subgraph
  - Particularly look for vertices "at center" (**Centrality**)
- Hierarchical decomposition also possible
  - Communication between clusters may be "different" from within cluster
- Splitting graphs into communities called "partitioning"
- Survey: "Community detection in graphs"
  - https://www.sciencedirect.com/science/article/pii/S0370157309002841#!



"Community detection in graphs" Fig. 1

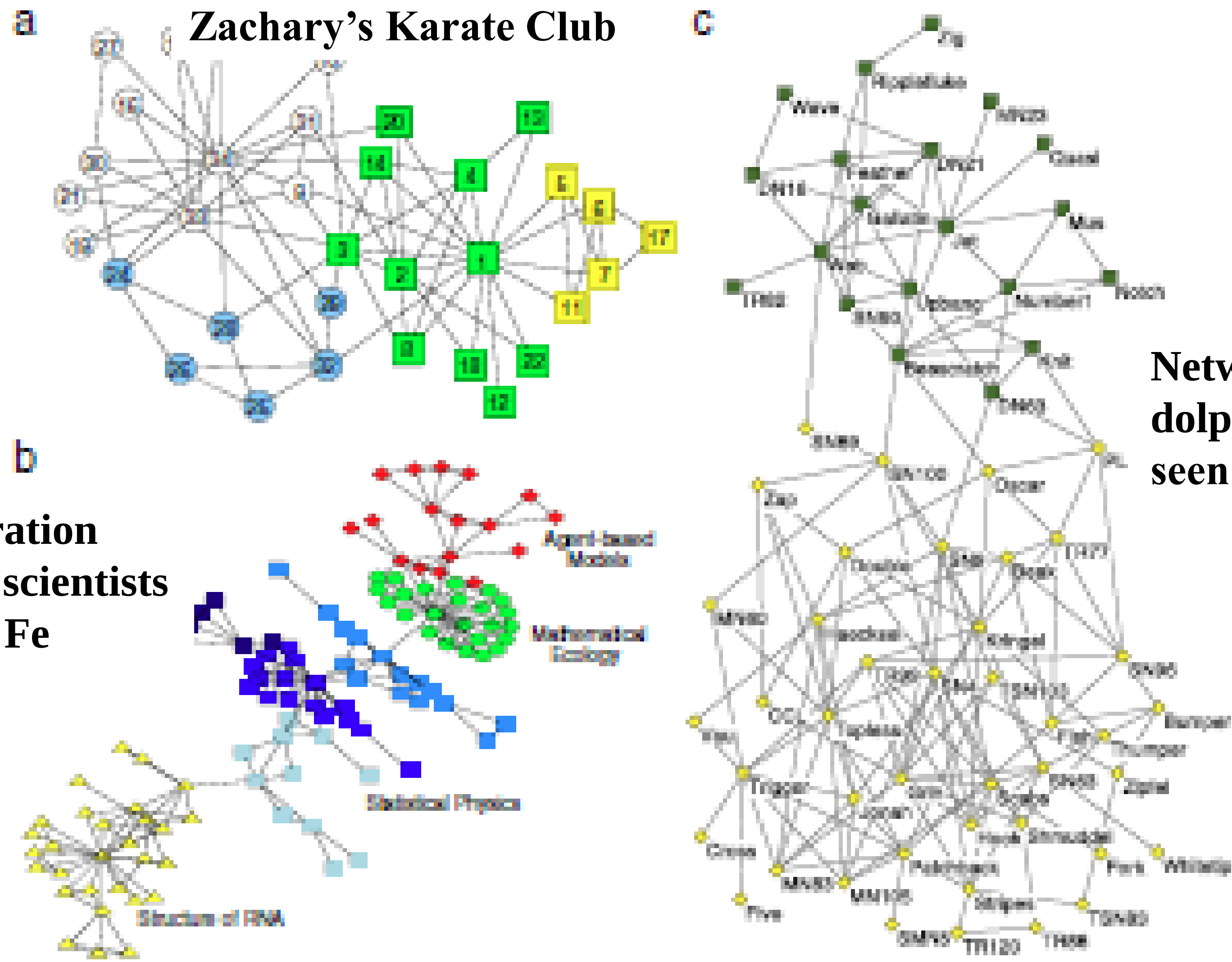
Real World Problems

- Social networks
- Proteins that perform same function in a cell
- Co-locating web pages with similar topics
- Clusters of customers with similar buying
- Ad hoc networks formed by interacting nodes in same region
- Organization in business firms
  - Pyramidal at top level
  - Departments more like clusters (with “central” manager)
- Parallel computing: allocate tasks to nodes to minimize communication

Paper: 78-81,156-161

Sample Graphs

Zachary's Karate Club



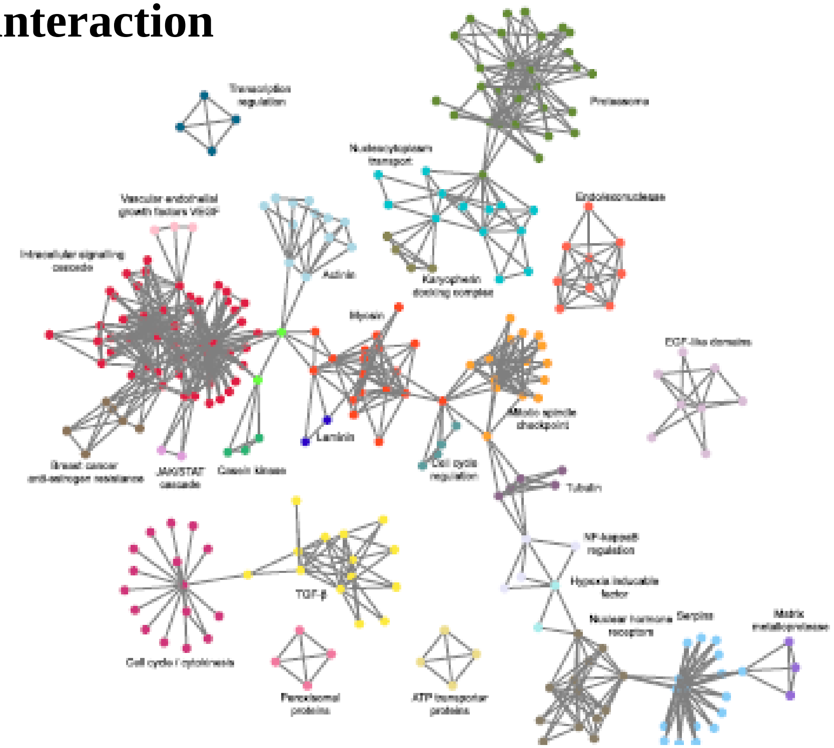
Collaboration between scientists at Santa Fe Institute

Network of dolphins when seen "together"

"Community detection in graphs" Fig. 2

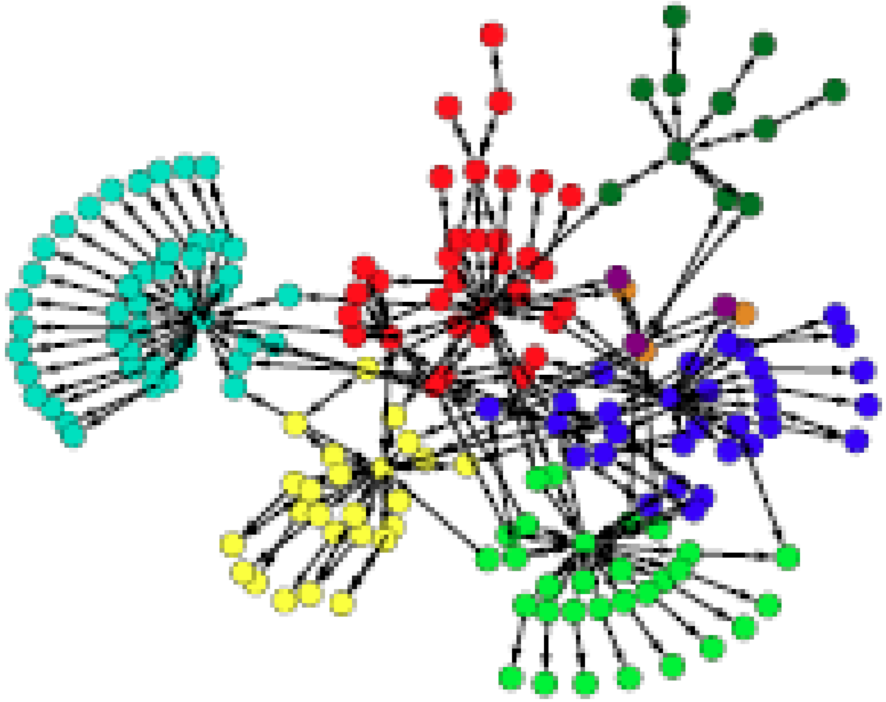
More Graphs

Protein-protein interaction



"Community detection in graphs" Fig. 3

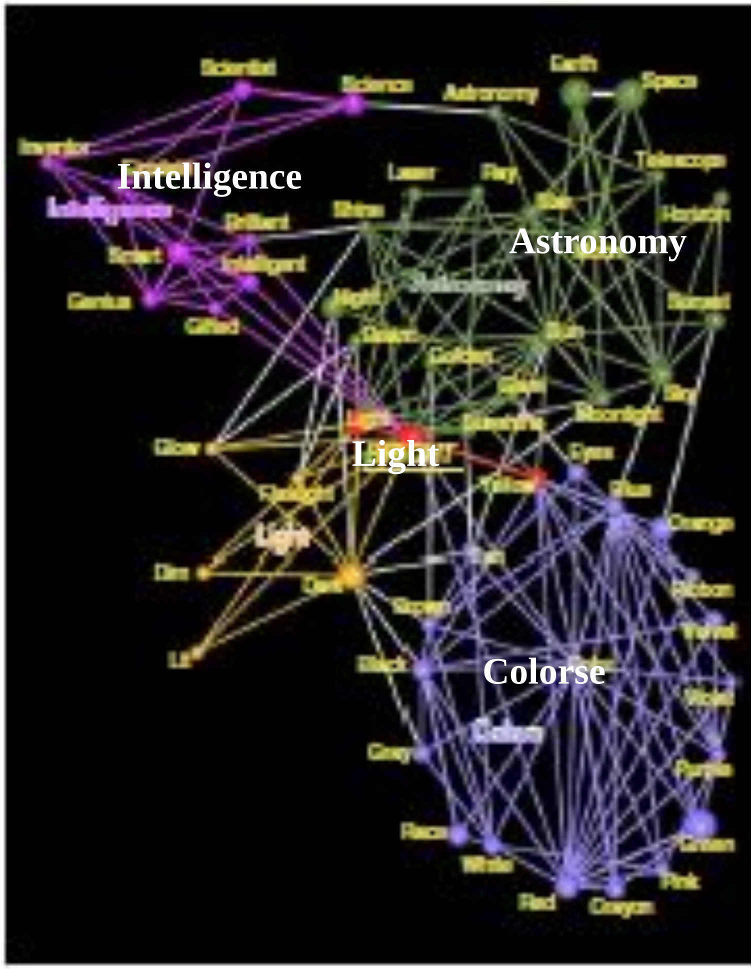
Hyperlinks between web pages



"Community detection in graphs" Fig. 4

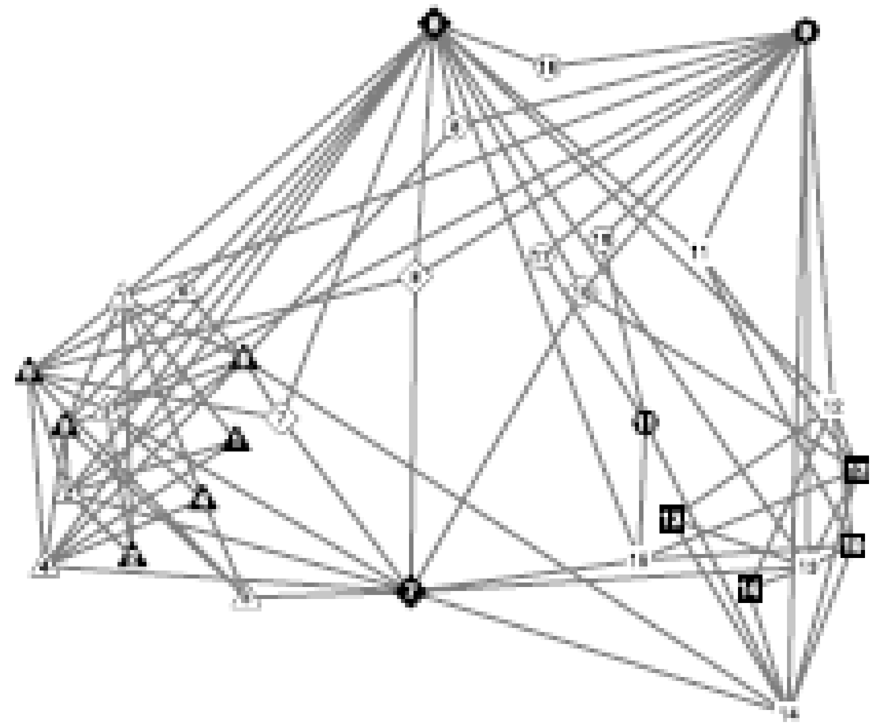
More Graphs

Word Associations



“Community detection in graphs” Fig. 5

Bipartite graphs:
Black: people
White: events



“Community detection in graphs” Fig. 6

Clustering

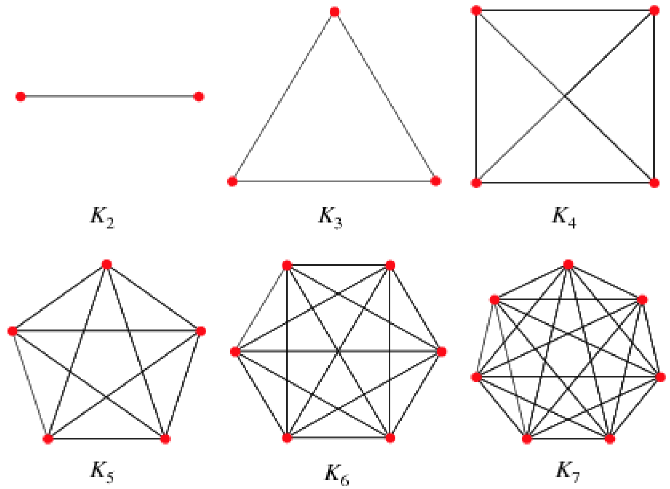
- Terms like community not well-defined
  - Often imprecise and app-dependent
- Id of clusters often reasonable with “sparse graphs,” i.e. M (edges) of O(N)
- If M >> N (many edges), networks become too homogeneous
- Key discriminator: what do edges connect
  - **Internal**: between two vertices in <u>same</u> community
  - **External**: between two vertices in <u>different</u> communities

Edge Density

- g a subgraph of G forming a community
  - $|G| = n$ and $|g| = n_g$
- **$k(u)$** = degree of vertex u in subgraph g
  - **$k_{int}(u)$** = # of edges from u to others in its community
  - **$k_{ext}(u)$** = # of edges from u to vertices not in g
  - $k = k_{int}(u) + k_{ext}(u)$
- Density:
  - **Average link: $\delta(g)$** = # edges of g / $(n_g(n_g-1)/2)$
  - **Intra-cluster**: **$\delta_{int}(g)$** = #_intra-cluster_edges / $(n_g(n_g-1)/2)$
  - **Inter-cluster**: **$\delta_{ext}(g)$** = #_inter-cluster_edges / $(n_g(n_g-1)/2)$
  - Note: $(n_g(n_g-1)/2)$ is # of edges in g if fully connected
- Expect: $\delta_{int}(g) > \delta(g) > \delta_{ext}(g)$
  - The larger the $\delta_{int}(g) - \delta_{ext}(g)$, the "more connected"

Definitions of "Locality"

- Community has "few" edges to rest of graph
- **Maximal subgraphs**: Adding new vertices does not improve community criteria
- One "ideal" definition: a **clique**
  - All vertices have edges to each other
  - But all vertices are "symmetric"
  - Other communities have "center"
  - Finding cliques is NP-Complete



http://mathworld.wolfram.com/images/eps-gif/CompleteGraphs_801.gif

- Relaxed definition: **n-clique**
  - Distance between any two vertices ≤ n
  - 1-clique is a clique
- **n-clan**: n-clique with diameter ≤ n
- **n-club**: n-clique maximal subgraph of diameter n

Adjacency of Vertices

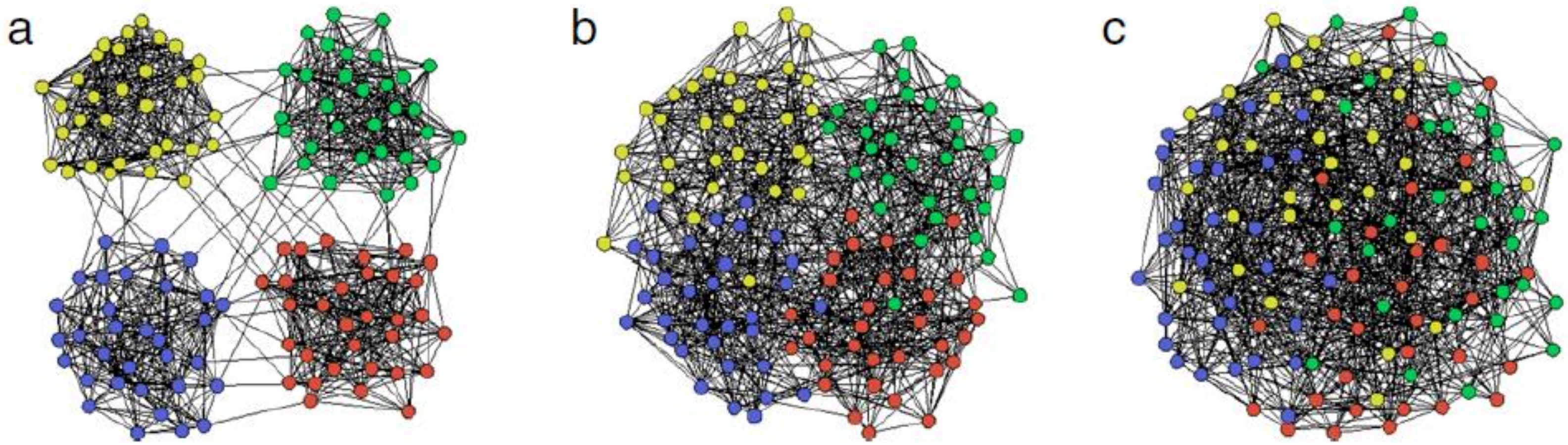
- Vertex must be connected to some minimal number of vertices in community
- **k-plex**: each vertex connected to all but at most k others in community
- **k-core**: connected to at least k others
- **LS-set** or **strong community**:
  - For all u in community $k_{int}(u) > k_{ext}(u)$

Algorithms

- **Graph partitioning**: cut into subgrahs such that number of cross subgroup edges (**cut set**) is minimal
  - **Minimal bisection**: recursively cut in half
    - Kernigan-Lin algorithm
  - Spectral bisection: use spectrum of Laplacian matrix
- **Hierarchical Clustering**: find “similar” subgraphs
- Divisive: find edges between communities & delete
  - Girvan Newman algorithm
- Others
  - Partitional, Spectral Clustering
  - Modularity optimization
  - Simulated annealing
  - Extremal optimization

Benchmark Graphs

- Planted L-partition model
  - L groups of g vertices each
  - Intra-group vertices linked with probability p_{in}
  - Inter-group vertices linked with probability p_{out}
  - When $p_{in} > p_{out}$ graph has “community” structure



“Community detection in graphs” Fig. 30

Metrics When Clusters Known

- Fraction of correctly classified vertices
- **Pair counting**: # of pairs in same partition in both predicted and known
  - Rand Index
  - Mirkin Metric
  - Jaccard Index
- **Cluster Matching**: largest overlaps between pairs of clusters of different partitions
- **Information Theory**: compute "information theory" of partitions

Kernigan-Lin Algorithm

- Goal: partition graph into 2 nearly equal subgraphs that minimize weight of crossing from one to other
  - If unweighted, minimize crossing edge count
- Repeated Greedy Algorithm
  - Keep a running partition and crossing weight
  - Pair up vertices from 2 partitions
  - Compute reduction in crossing weight
  - Choose pair that reduces crossing weight
  - Repeat

Girvan Newman algorithm

- Edge Betweenness of an edge e: # of shortest paths between 2 vertices thru e
  - Edges between clusters will have a “lot” of shortest paths thru them
- Algorithm:
  - Compute edge betweenness of all edges
  - Remove edge with highest betweenness
  - Recalculate
  - Repeat
- When no paths between some edges are found, we have found clusters