Random Graphs and Complex Networks
T-79.7003

Charalampos E. Tsourakakis

Aalto University

13 December 2013

Class website

http://www.math.cmu.edu/~ctsourak/
t797003-graphs-and-networks.html

Dense subgraphs

What is a dense subgraph?

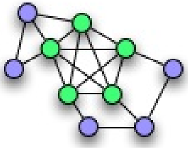
- a set of vertices with abundance of edges
- a highly connected subgraph
- key primitive for detecting communities
- related problem to community detection and graph partitioning, but not identical
  - not constrainted for a disjoint partition of all vertices

Applications of finding dense subgraphs

- thematic communities and spam link farms [Kumar et al., 1999]
- graph visualization [Alvarez-Hamelin et al., 2005]
- real-time story identification [Angel et al., 2012]
- motif detection [Fratkin et al., 2006]
- epilepsy prediction [Iasemidis et al., 2003]
- finding correlated genes [Zhang and Horvath, 2005]
- many more ...

Density measures

- consider subgraph induced by $S \subseteq V$ of $G = (V, E)$
- clique: each vertex in S is connected to every other vertex in S



- α-quasiclique: the set S has at least $\alpha|S|(|S|-1)/2$ edges
- k-core: every vertex in S is connected to at least k other vertices in S

Density measures

- consider subgraph induced by $S \subseteq V$ of $G = (V, E)$
- density:

$$\delta(S) = \frac{e[S]}{\binom{|S|}{2}} = \frac{2e[S]}{|S|(|S| - 1)}$$

- average degree:

$$d(S) = \frac{2e[S]}{|S|}$$

- *k*-densest subgraph:

$$\delta(S) = \frac{2e[S]}{|S|}, \quad \text{such that } |S| = k$$

Density measures

compare with measures we saw previously....

graph expansion:

$$\alpha(G) = \min_{S} \frac{e[S, V \setminus S]}{\min\{|S|, |V \setminus S|\}}$$

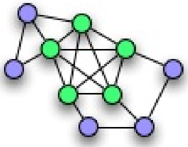
graph conductance:

$$\phi(G) = \min_{S \subseteq V} \frac{e[S, V \setminus S]}{\min\{vol(S), vol(V \setminus S)\}}$$

edges within ($e[S]$) instead of edges accross ($e[S, V \setminus S]$)

Complexity of density problems — clique

- find the max-size clique in a graph:
  NP-hard problem



- strong innaproximability result:

  for any $\epsilon > 0$, there cannot be a polynomial-time algorithm that approximates the maximum clique problem within a factor better than $\mathcal{O}(n^{1-\epsilon})$, unless $\mathbf{P} = \mathbf{NP}$

  [Håstad, 1997]

Complexity of other density problems

density	$\delta(S) = \frac{e[S]}{\binom{\lvert S\rvert}{2}}$	pick a single edge
average degree	$d(S) = \frac{2e[S]}{\lvert S\rvert}$	in **P**
k-densest subgraph	$\delta(S) = \frac{2e[S]}{\lvert S\rvert}, \lvert S\rvert = k$	**NP**-hard
DalkS	$\delta(S) = \frac{2e[S]}{\lvert S\rvert}, \lvert S\rvert \geq k$	**NP**-hard
DamkS	$\delta(S) = \frac{2e[S]}{\lvert S\rvert}, \lvert S\rvert \leq k$	*L*-reduction to DkS

Densest subgraph problem

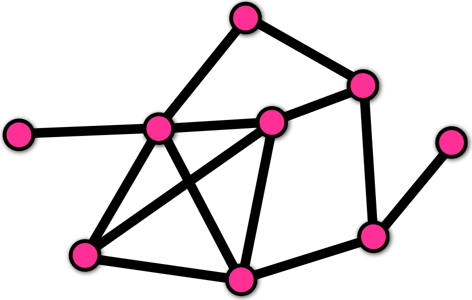
- find set of vertices $S \subseteq V$ with maximum average degree $d(S) = 2e[S]/|S|$
- solvable in polynomial time
  - max-flow [Goldberg, 1984]
  - LP relaxation [Charikar, 2000]
- simple linear-time greedy algorithm gives factor-2 approximation [Asahiro et al., 2000, Charikar, 2000]

Greedy algorithm for densest subgraph

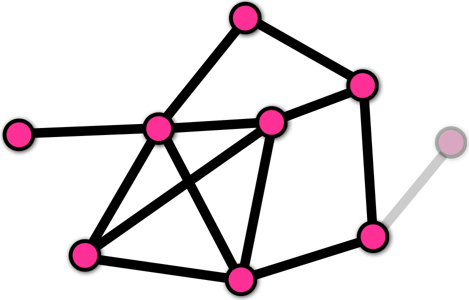
[Asahiro et al., 2000, Charikar, 2000]

input: undirected graph $G = (V, E)$
output: S, a dense sungraph of G
1 set $G_n \leftarrow G$
2 for $k \leftarrow n$ downto 1
2.1 let v be the smallest degree vertex in G_k
2.2 $G_{k-1} \leftarrow G_k \setminus \{v\}$
3 output the densest subgraph among $G_n, G_{n-1}, \ldots, G_1$

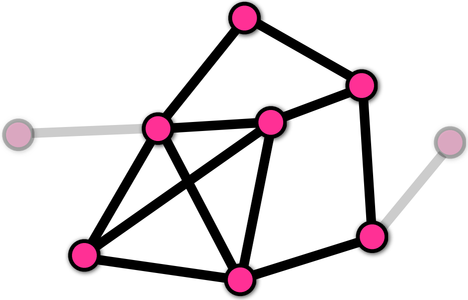
Greedy algorithm for densest subgraph — example



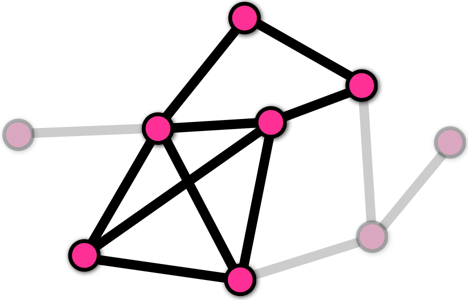
Greedy algorithm for densest subgraph — example



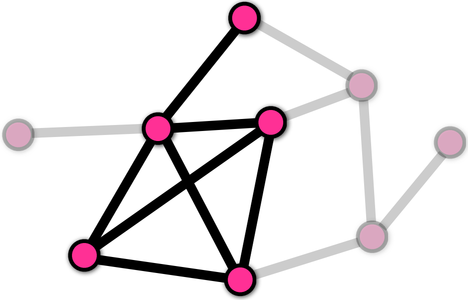
Greedy algorithm for densest subgraph — example



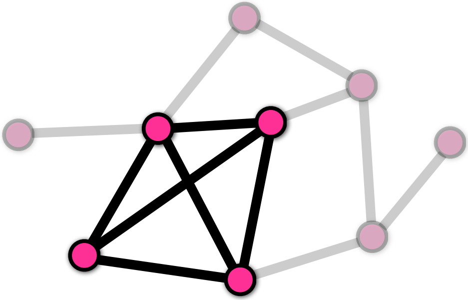
Greedy algorithm for densest subgraph — example



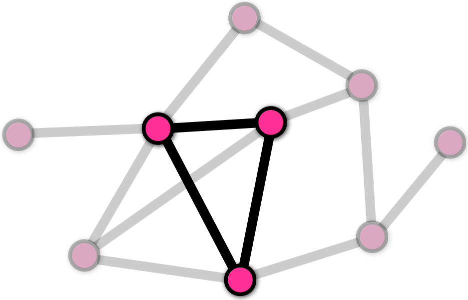
Greedy algorithm for densest subgraph — example



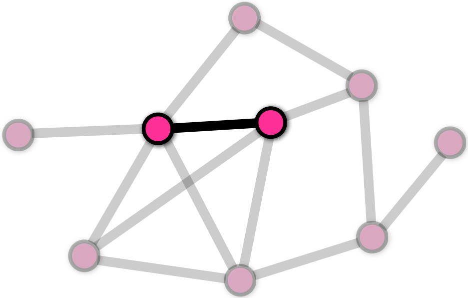
Greedy algorithm for densest subgraph — example



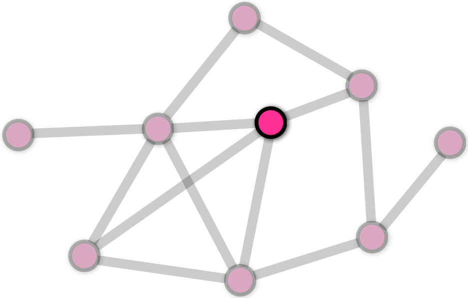
Greedy algorithm for densest subgraph — example



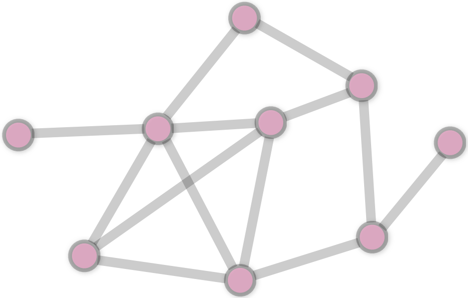
Greedy algorithm for densest subgraph — example



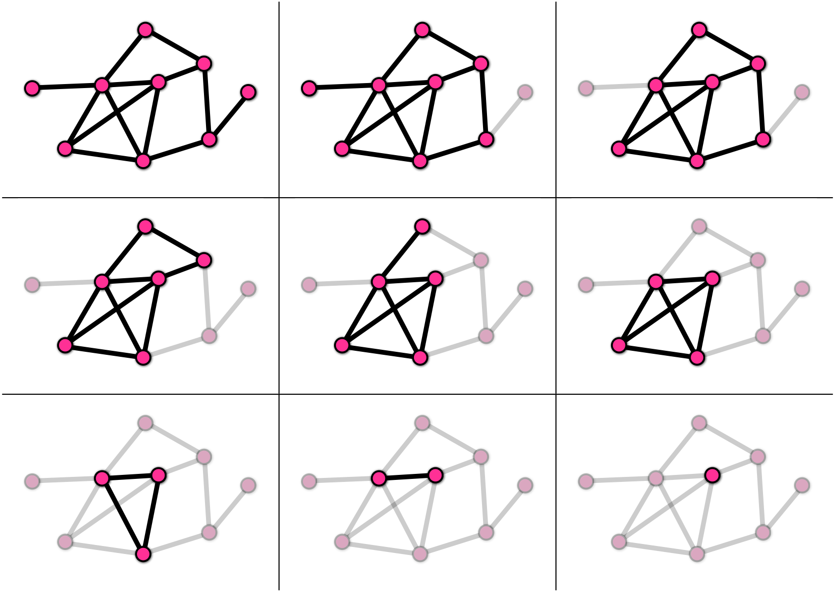
Greedy algorithm for densest subgraph — example



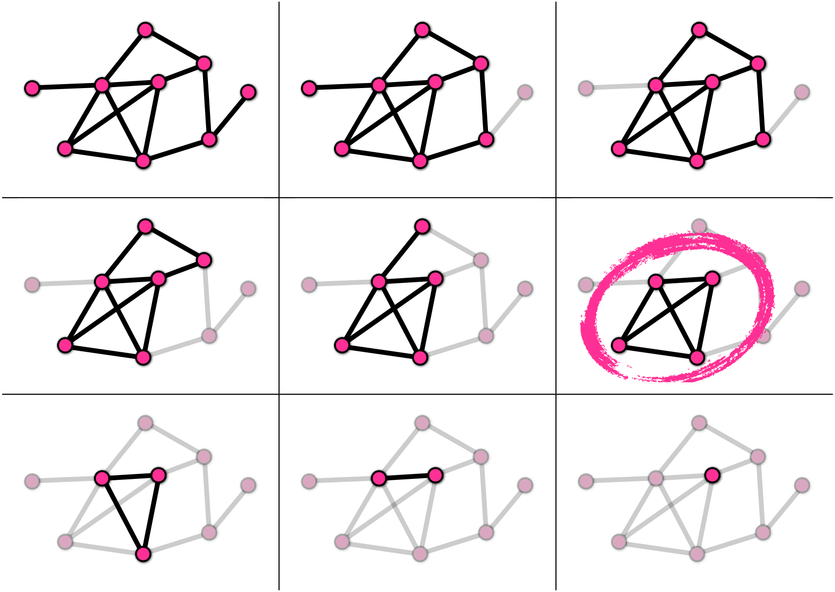
Greedy algorithm for densest subgraph — example



Greedy algorithm for densest subgraph — example



Greedy algorithm for densest subgraph — example



Approximation guarantees

Let's overload the notation $d(S)$ in what follows. Let $d(S) \leftarrow \frac{d(S)}{2}$, namely $d(S) = \frac{e[S]}{|S|}$.

Theorem

The greedy algorithm achieves a 2-approximation for the densest subgraph problem in undirected networks.

Proof.

Let the optimal value be $d(S^*) = \lambda$. Consider the vertex with the smallest (induced) degree in S^*. Let this degree be d_{min} and $|S^*| = s^*$.

$\square$

Approximation guarantees

Cont.

Proof.

By the optimality of S^*

$$\lambda = \frac{e[S^*]}{s^*} \geq \frac{e[S^*] - d_{min}}{s^* - 1} \rightarrow d_{min} \geq \lambda.$$

Consider the moment when the greedy algorithm removes a vertex that belongs in S^*. By the way the algorithm iterates, all remaining vertices have induced degree at least λ. Let S be the set of these vertices, $|S| = s$. Then, the subgraph has $\lambda s/2$ edges and the density is $d(S) = \lambda/2$. This guarantees an approximation ratio $1/2$. □

Tightness [Khuller and Saha, 2009]

Run the greedy approximation algorithm on

$$K_{d,D} \cup \underbrace{K_{d+1} \cup \ldots \cup K_{d+1}}_{D \text{ times}}.$$

What is the output?
What is the optimal solution?

Other notions and generalizations

- k-core: every vertex in S is connected to at least k other vertices in S
- α-quasiclique: the set S has at least $\alpha|S|(|S|-1)/2$ edges
- enumerate all α-quasicliques [Uno, 2010]
- dense subgraphs of directed graphs: find sets $S, T \subseteq V$ to maximize

$$d(S,T) = \frac{e[S,T]}{\sqrt{|S|\,|T|}}$$

[Charikar, 2000, Khuller and Saha, 2009]

Edge-surplus framework

- Introduced in [Tsourakakis et al., 2013, Tsourakakis, 2013]
- for a set of vertices S define edge surplus

$$f(S) = g(e[S]) - h(|S|)$$

where g and h are both strictly increasing

- optimal (g, h)-edge-surplus problem:

  find S^* such that

$$f(S^*) \geq f(S), \quad \text{for all sets } \; S \subseteq S^*$$

Edge-surplus framework

- edge surplus $f(S) = g(e[S]) - h(|S|)$
- example 1

$$g(x) = h(x) = \log x$$

find S that maximizes $\log \frac{e[S]}{|S|}$

densest-subgraph problem

- example 2

$$g(x) = x, \quad h(x) = \begin{cases} 0 & \text{if } x = k \\ +\infty & \text{otherwise} \end{cases}$$

k-densest-subgraph problem

The optimal quasiclique problem

[Tsourakakis et al., 2013, Tsourakakis, 2013]

- edge surplus $f(S) = g(e[S]) - h(|S|)$

- consider
$$g(x) = x, \quad h(x) = \alpha \frac{x(x-1)}{2}$$
find S that maximizes $e[S] - \alpha \binom{|S|}{2}$

  optimal quasiclique problem

- theorem: let $g(x) = x$ and $h(x)$ concave

  then the optimal (g, h)-edge-surplus problem is polynomially-time solvable

The optimal quasiclique problem

theorem: let $g(x) = x$ and $h(x)$ concave

then the optimal (g, h)-edge-surplus problem is polynomially-time solvable

proof

$g(x) = x$ is supermodular

if $h(x)$ concave $h(x)$ is submodular

$-h(x)$ is supermodular

$g(x) - h(x)$ is supermodular

maximizing supermodular functions is solvable in polynomial time

The optimal quasiclique problem

theorem: let $g(x) = x$ and $h(x)$ concave

then the optimal (g, h)-edge-surplus problem is polynomially-time solvable

- However, this is not a particularly interesting case. The output will be too big, if not the whole graph.

Optimal quasicliques in practice

densest subgraph vs. optimal quasiclique

	densest subgraph				optimal quasi-clique			
	$\frac{\lvert S \rvert}{\lvert V \rvert}$	δ	D	τ	$\frac{\lvert S \rvert}{\lvert V \rvert}$	δ	D	τ
Dolphins	0.32	0.33	3	0.04	0.12	0.68	2	0.32
Football	1	0.09	4	0.03	0.10	0.73	2	0.34
Jazz	0.50	0.34	3	0.08	0.15	1	1	1
Celeg. N.	0.46	0.13	3	0.05	0.07	0.61	2	0.26

[Tsourakakis et al., 2013]

Understanding the objective

[Tsourakakis, 2013]

- $\underline{\alpha = 0}$: Optimal solution=G. Not interesting
- $\underline{0 < \alpha < 1}$: In general hard. Let's see.
    - Assume that finding a hidden clique of order $O(n^{1/2-\delta})$ where $\delta > 0$ in a random binomial graph $G \sim G(n, 1/2)$ is hard.
    - Hidden clique score $= (1-\alpha)\binom{n^{1/2-\delta}}{2}$.
    - Score of a random set$= (1/2-\alpha)\binom{n^{1/2-\delta}}{2}$.
    - Set $\alpha > 1/2$ to solve the problem in expectation.
    - By setting $\alpha = 1 - \frac{1}{\Omega(n^2)}$ we solve the max-clique problem.
    - Straightforward inapproximability results.

Understanding the objective

- $\underline{\alpha = 1}$: Clearly optimal score is always 0 and achieved by an edge. In general all cliques achieve this score.
- $\underline{\alpha > 1}$ Not interesting. Let $\alpha = 1 + \epsilon$, where $\epsilon > 0$. The score is of the form $\underbrace{e[S] - \binom{|S|}{2}}_{\leq 0} - \epsilon\binom{|S|}{2}$. Hence, a single edge minimizes the score.

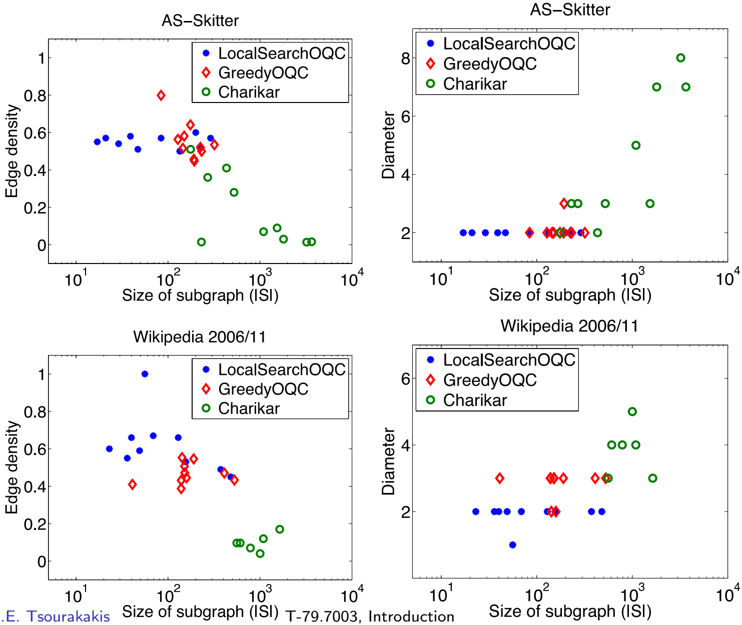
Finding and optimal quasiclique

adaptation of the greedy algorithm of [Charikar, 2000]

input: undirected graph $G = (V, E)$
output: a quasiclique S
1 set $G_n \leftarrow G$
2 for $k \leftarrow n$ downto 1
2.1 let v be the smallest degree vertex in G_k
2.2 $G_{k-1} \leftarrow G_k \setminus \{v\}$
3 output the subgraph in $G_n, \ldots, G_1$ that maximizes $f(S)$

additive approximation guarantee [Tsourakakis et al., 2013]

top-k densest subgraphs and quasicliques



Multiplicative Approximation: a 0.796-approximation algorithm

[Tsourakakis, 2013]

- $f_\alpha(S) \leftarrow f_\alpha(S) + \alpha\binom{n}{2}$. Then $f_\alpha(S) \geq 0$ for any $S \subseteq V$.
- This shifting is not necessary since the optimal objective value is positive in the interesting range of $0 < \alpha < 1$ as a single edge results in a positive score $1 - \alpha$.
- Adds a huge additive error but does not render the objective useless for all graphs.
- Result of limited value due to the large additive error for realistic cases.

Multiplicative Approximation: a 0.796-approximation algorithm

- We introduce a variable $x_i \in \{-1, +1\}$ for each vertex $i \in V = \{1, \ldots, n\}$ and an extra variable x_0 which expresses whether a vertex belongs to S or not:

$$\text{It is } i \in S \text{ if and only if } x_0 x_i = 1.$$

- Notice that the term $\frac{1+x_0x_i+x_0x_j+x_ix_j}{4}$ equals 1 if and only if both i, j belong in S, otherwise it equals 0. Furthemore, the term $\binom{n}{2}$ enters the objective as $\frac{1}{2}\sum_{i \neq j} 1$.

Multiplicative Approximation: a 0.796-approximation algorithm

Therefore, we get the following integer program:

$$
\begin{aligned}
\textbf{max} \quad & \sum_{e=(i,j)} \frac{1 + x_0 x_i + x_0 x_j + x_i x_j}{4} + \\
& \frac{\alpha}{2} \sum_{i \neq j} \Big(1 - \frac{1 + x_0 x_i + x_0 x_j + x_i x_j}{4} \Big) \\
\text{subject to } \; & x_i \in \{-1, +1\}, \text{ for all } \; i \in \{0, 1, .., n\}.
\end{aligned} \tag{1}
$$

Multiplicative Approximation: a 0.796-approximation algorithm

We relax the integrality constraint and we allow the variables to be vectors in the unit sphere in $\mathbb{R}^{n+1}$. By using the variable transformation $y_{ij} = x_i x_j$, we obtain the following semidefinite programming relaxation:

$$
\boxed{
\begin{aligned}
\textbf{max} \quad & \alpha \sum_{e=(i,j)} \frac{1 + y_{0i} + y_{0j} + y_{ij}}{4} + \\
& \frac{1}{2} \sum_{i \neq j} \left(1 - \frac{1 + y_{0i} + y_{0j} + y_{ij}}{4}\right) \\
\text{subject to} \quad & y_{ii} = 1, \text{ for all } i \in \{0, 1, .., n\} \\
\text{and} \quad & Y \succeq 0, \; Y \text{ symmetric.}
\end{aligned}
}
\qquad (2)
$$

Multiplicative Approximation: a 0.796-approximation algorithm

SDP-Edge-Surplus

- *Relaxation:* Solve the semidefinite program and compute a Cholesky decomposition of Y. Let $v_0, v_1, \ldots, v_n$ be the resulting vectors.
- *Randomized Rounding:* Randomly choose a unit length vector $r \in \mathbb{R}^{n+1}$ and set $S = \{i \in [n] : sgn(v_i r) = sgn(v_0 r)\}$.
- *Boosting Success Probability:* Repeat steps 1-2 for $t = 1, .., T$ and output the best solution found over the $T = c_{\epsilon,\alpha,\beta} \log n$ runs. Here, $1 > \epsilon > 0$ is a small positive constant and $c_{\epsilon,\alpha,\beta} = \frac{1 - \frac{(1-\epsilon)3\beta}{2(\alpha+1)}}{\epsilon \frac{3\beta}{2(\alpha+1)}}$.

Multiplicative Approximation: a 0.796-approximation algorithm

Theorem ([Tsourakakis, 2013])

Algorithm SDP-Edge-Surplus is a β-approximation algorithm for f where $\beta > 0.79607$ with probability at least $1 - O(n^{-1})$.

The community-search problem

- a dense subgraph that contains a given subset of vertices $Q \subseteq V$ (the query vertices)

- the center-piece subgraph problem
- the team formation problem
- the cocktail party problem

applications

- find the community of a given set of users
    - a meaningful way to address the issue of overlapping communities
- find a set of proteins related to a given set
- form a team to solve a problem

Center-piece subgraph [Tong and Faloutsos, 2006]

- given: graph $G = (V, E)$ and set of query vertices $Q \subseteq V$
- find: a connected subgraph H that
  (a) contains Q
  (b) optimizes a goodness function $g(H)$

- main concepts:
- `k_softAND`: a node in H should be well connected to at least k vertices of Q
- $r(i, j)$ goodness score of j wrt $q_i \in Q$
- $r(Q, j)$ goodness score of j wrt Q
- $g(H)$ goodness score of a candidate subgraph H
- $H^* = \arg\max_H g(H)$

Center-piece subgraph

[Tong and Faloutsos, 2006]

- $r(i, j)$ goodness score of j wrt $q_i \in Q$

  probability to meet j in a random walk with restart to q_i

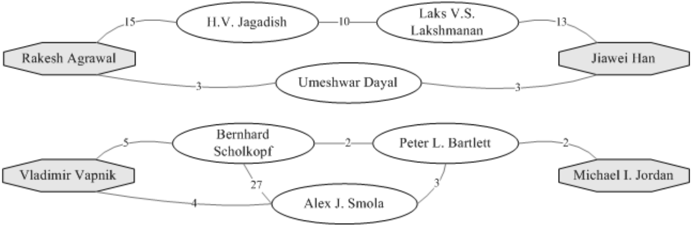
- $r(Q, j)$ goodness score of j wrt Q

  probability to meet j in a random walk with restart to k vertices of Q

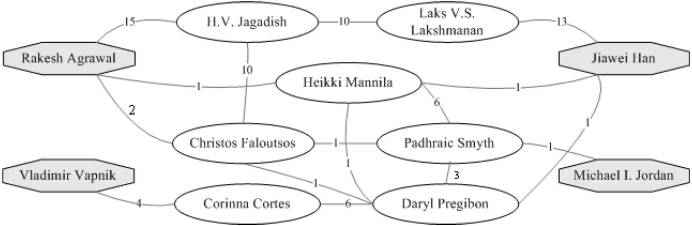
- proposed algorithm:

1. greedy: find a good destination vertex j ito add in H
2. add a path from each of top-k vertices of Q path to j
3. stop when H becomes large enough

Center-piece subgraph — example results



(a) “K_softANDquery”: $k = 2$



(b) “AND query”

[Tong and Faloutsos, 2006]

The community-search problem [Sozio and Gionis, 2010]

- given: graph $G = (V, E)$ and set of query vertices $Q \subseteq V$
- find: a connected subgraph H that
  (a) contains Q
  (b) vertices of H are close to Q
  (c) optimizes a density function $d(H)$

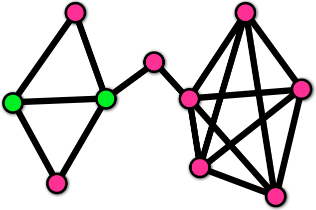
- distance constraint (b):

$$d(Q, j) = \sum_{q \in Q} d^2(q_i, j) \leq B$$

- density function (c):

  average degree, minimum degree, quasiclique, measured on the induced subgraph H

The community-search problem



both the distance constraint and the minimum-degree density help addressing the problem of free riders

The community-search problem

algorithm proposed by [Sozio and Gionis, 2010]
adaptation of the greedy algorithm of [Charikar, 2000]

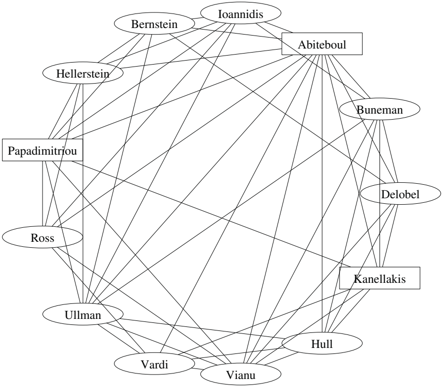
input: undirected graph $G = (V, E)$, query vertices $Q \subseteq V$
output: connected, dense subgraph H

1 set $G_n \leftarrow G$
2 for $k \leftarrow n$ downto 1
2.1 remove all vertices violating distance constraints
2.2 let v be the smallest degree vertex in G_k among all vertices not in Q
2.3 $G_{k-1} \leftarrow G_k \setminus \{v\}$
2.4 if left only with vertices in Q or disconnected graph, stop
3 output the subgraph in $G_n, \ldots, G_1$ that maximizes $f(H)$

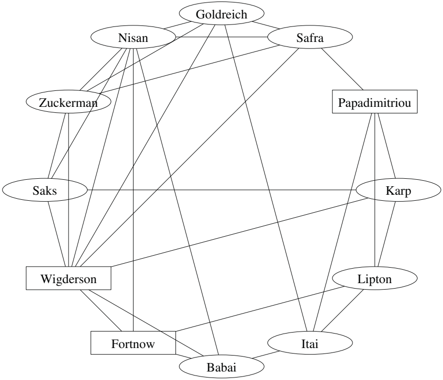
Properties of the greedy algorithm

- returns optimal solution if no size constraints or lower-bound constraints
- heuristic variants proposed when upper-bound constraints
- generalized for monotone constraints and monotone objective functions

The community-search problem — example results



(a) Database theory



(b) Complexity theory

(from [Sozio and Gionis, 2010])

Conclusions (dense subgraphs)

summary

- discussed a number of different density measures
- discussed a number of diiferent problem formulations
- polynomial-time solvable or **NP**-hard problems
- global dense subgraphs or relative to query vertices

promising future directions

- explore further the concept of α-quasiclique (no shifting, better additive guarantees)
- better algorithms for upper-bound constraints
- top-k versions of dense subgraphs
- adapt concepts for labeled graphs
- local algorithms

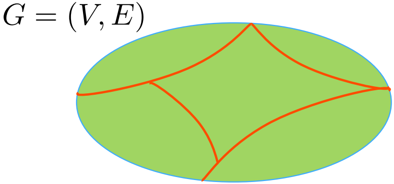
Streaming Graph partitioning

Need for scalable algorithms

- spectral, agglomerative, LP-based algorithms
- not scalable to very large graphs
- handle datasets with billions of vertices and edges
  - `facebook`: $\sim$ 1 billion users with avg degree 130
  - `twitter`: $\geq$ 1.5 billion social relations
  - `google`: web graph more than a trillion edges (2011)
- design algorithms for streaming scenarios
  - real-time story identification using `twitter` posts
  - election trends, `twitter` as election barometer

Graph partitioning

- graph partitioning is a way to split the graph vertices in multiple machines
- graph partitioning objectives guarantee low communication overhead among different machines
- additionally balanced partitioning is desirable



- each partition contains $\approx n/k$ vertices

Off-line k-way graph partitioning

METIS algorithm [Karypis and Kumar, 1998]

- popular family of algorithms and software
- multilevel algorithm
- coarsening phase in which the size of the graph is successively decreased
- followed by bisection (based on spectral or KL method)
- followed by uncoarsening phase in which the bisection is successively refined and projected to larger graphs

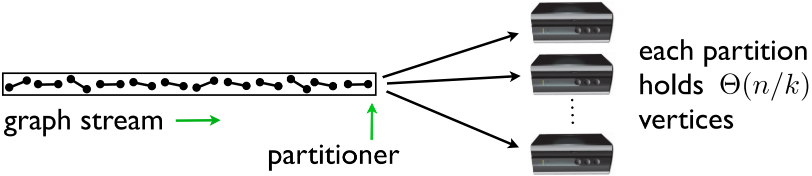
Off-line k-way graph partitioning

Krauthgamer, Naor and Schwartz [Krauthgamer et al., 2009]

- problem: minimize number of edges cut, subject to cluster sizes $\Theta(n/k)$
- approximation guarantee: $O(\sqrt{\log k \log n})$
- based on the work of Arora-Rao-Vazirani for the sparsest-cut problem ($k = 2$) [Arora et al., 2009]

streaming k-way graph partitioning

- input is a data stream
- graph is ordered
  - arbitrarily
  - breadth-first search
  - depth-first search
- generate an approximately balanced graph partitioning



Graph representations

- adjacency stream
  - at time t, a vertex arrives with its neighbors
- edge stream
  - at time t, an edge arrives

Partitioning strategies

- hashing: place a new vertex to a cluster/machine chosen uniformly at random
- neighbors heuristic: place a new vertex to the cluster/machine with the maximum number of neighbors
- non-neighbors heuristic: place a new vertex to the cluster/machine with the minimum number of non-neighbors

Partitioning strategies

[Stanton and Kliot, 2012]

- $d_c(v)$: neighbors of v in cluster c
- $t_c(v)$: number of triangles that v participates in cluster c

- balanced: vertex v goes to cluster with least number of vertices
- hashing: random assignment
- weighted degree: v goes to cluster c that maximizes $d_c(v) \cdot w(c)$
- weighted triangles: v goes to cluster j that maximizes $t_c(v)/\binom{d_c(v)}{2} \cdot w(c)$

Weight functions

- s_c: number of vertices in cluster c
- unweighted: $w(c) = 1$
- linearly weighted: $w(c) = 1 - s_c(k/n)$
- exponentially weighted: $w(c) = 1 - e^{(s_c - n/k)}$

FENNEL algorithm

[Tsourakakis et al., 2012]

$$\begin{aligned} \text{minimize}_{\mathcal{P}=(S_1,\ldots,S_k)} \quad & |\partial\, e(\mathcal{P})| \\ \text{subject to} \quad & |S_i| \leq \nu \frac{n}{k},\ \text{for all } 1 \leq i \leq k \end{aligned}$$

- hits the ARV barrier

$$\text{minimize}_{\mathcal{P}=(S_1,\ldots,S_k)} \quad |\partial\, E(\mathcal{P})| + c_{\mathrm{IN}}(\mathcal{P})$$

where $c_{\mathrm{IN}}(\mathcal{P}) = \sum_i s(|S_i|)$, so that objective self-balances

- relax hard cardinality constraints

FENNEL algorithm

[Tsourakakis et al., 2012]

- for $S \subseteq V$, $f(S) = e[S] - \alpha|S|^\gamma$, with $\gamma \geq 1$
- given partition $\mathcal{P} = (S_1, \ldots, S_k)$ of V in k parts define

$$g(\mathcal{P}) = f(S_1) + \ldots + f(S_k)$$

- the goal: maximize $g(\mathcal{P})$ over all possible k-partitions
- notice:

$$g(\mathcal{P}) = \underbrace{\sum_i e[S_1]}_{\text{number of edges cut}} - \underbrace{\alpha \sum_i |S_i|^\gamma}_{\text{minimized for balanced partition!}}$$

Connection

notice

$$f(S) = e[S] - \alpha \binom{|S|}{2}$$

- related to modularity
- related to quasicliques (see next)

FENNEL algorithm

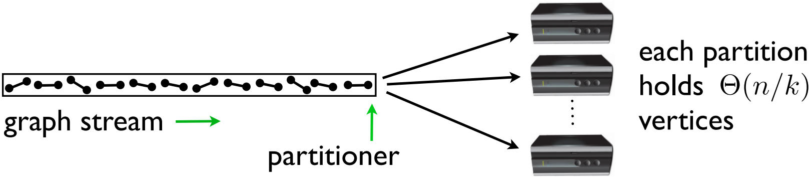
theorem [Tsourakakis et al., 2012]

- $\gamma = 2$ gives approximation factor $\log(k)/k$
  where k is the number of clusters
- random partitioning gives approximation factor $1/k$
- no dependence on n
  mainly because relaxing the hard cardinality constraints

FENNEL algorithm — greedy scheme

- $\gamma = 2$ gives non-neighbors heuristic
- $\gamma = 1$ gives neighbors heuristic
- interpolate between the two heuristics, e.g., $\gamma = 1.5$

FENNEL algorithm — greedy scheme



- send v to the partition / machine that maximizes

$$\begin{aligned} f(S_i \cup \{v\}) - f(S_i) \\ &= e[S_i \cup \{v\}] - \alpha(|S_i|+1)^{\gamma} - (e[S_i] - \alpha|S_i|^{\gamma}) \\ &= d_{S_i}(v) - \alpha\mathcal{O}(|S_i|^{\gamma-1}) \end{aligned}$$

- fast, amenable to streaming and distributed setting

FENNEL algorithm — results

$$\lambda = \frac{\#\{\text{edges cut}\}}{m} \qquad \rho = \max_{1 \le i \le k} \frac{|S_i|}{n/k}$$

		Fennel		METIS	
m	k	λ	ρ	λ	ρ
7 185 314	4	62.5 %	1.04	65.2%	1.02
6 714 510	8	82.2 %	1.04	81.5%	1.02
6 483 201	16	92.9 %	1.01	92.2%	1.02
6 364 819	32	96.3%	1.00	96.2%	1.02
6 308 013	64	98.2%	1.01	97.9%	1.02
6 279 566	128	98.4 %	1.02	98.8%	1.02

- $\gamma = 1.5$
- comparable results in quality, but FENNEL is lightway, fast, and streamable

Conclusions (graph partitioning)

summary

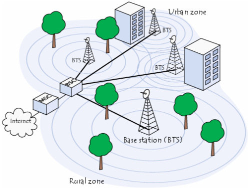
- spectral techniques, modularity-based methods, graph partitioning
- well-studied and mature area

future directions

- develop alternative notions for communities, e.g., accounting for graph labels, constraints, etc.
- further improve efficiency of methods
- overlapping communities

Rainbow connection

Rainbow connection



- Suppose we wish to route messages in a cellular network G, between any two vertices in a pipeline, and require that each link on the route between the vertices (namely, each edge on the path) is assigned a distinct channel (e.g., a distinct frequency). The minimum number of distinct channels we need to use is the rainbow connectivity of G.

Rainbow connection

- An edge colored graph G is rainbow edge connected iff any two vertices are connected by a path whose edges have distinct colors. The rainbow connectivity $rc(G)$ of a connected graph G is the smallest number of colors that are needed in order to make G rainbow edge connected.
- $rc(G) \leq n - 1$ Exercise
- $rc(G) = n - 1$ iff G is a tree Exercise
- $rc(G) = 1$ iff G is the complete graph K_n Exercise
- $rc(G) \leq n\frac{4 \log n + 3}{\delta}$ [Caro et al., 2008]

Rainbow connection

Let

$$L = \frac{\log n}{\log \log n} \tag{3}$$

and let $A \sim B$ denote $A = (1 + o(1))B$ as $n \to \infty$. We shall sketch the proof of the following theorem [Frieze and Tsourakakis, 2012a, Frieze and Tsourakakis, 2012b].

Theorem

Let $G = G(n, p), p = \frac{\log n + \omega}{n}, \omega \to \infty, \omega = o(\log n)$. Also, let Z_1 be the number of vertices of degree 1 in G. Then, with high probability(whp)

$$rc(G) \sim \max\{Z_1, L\},$$

Rainbow connection

Let a vertex be *large* if $\deg(x) \geq \log n/100$ and *small* otherwise.

Lemma

Whp, *there do not exist two small vertices within distance at most* $3L/4$.

Proof.

$$\mathbf{Pr}\left[\exists x, y \in [n] :\ \deg(x), \deg(y) \leq \log n/100, dist(x,y) \leq \frac{3L}{4}\right]$$

$$\leq \binom{n}{2} \sum_{k=1}^{3L/4} n^{k-1} p^k \left(\sum_{i=0}^{\log n/100} \binom{n-1-k}{i} p^i (1-p)^{n-1-k} \right)^2$$

Rainbow connection

Proof.

$$\leq \sum_{k=1}^{3L/4} n(2\log n)^k \left(2\binom{n}{\log n/100} p^{\log n/100}(1-p)^{n-1-\log n/100}\right)^2$$

$$\leq \sum_{k=1}^{3L/4} n(2\log n)^k \left(2(100e^{1+o(1)})^{\log n/100} n^{-1+o(1)}\right)^2$$

$$\leq \sum_{k=1}^{3L/4} n(2\log n)^k n^{-1.9}$$

$$\leq 2n(2\log n)^{3L/4} n^{-1.9}$$

$$\leq n^{-.1}.$$

$\square$

Rainbow connection

High-level sketch of the proof

1. Randomly color the edges of the graph in question, using a uniformly random coloring.
2. To prove that this works, we have to find, for each pair of vertices x, y, a large collection of edge disjoint paths joining them. It will then be easy to argue that at least one of these paths is rainbow colored.
3. To find these paths we pick a typical vertex x. We grow a regular tree T_x with root x. The depth is chosen carefully. We argue that for a typical pair of vertices x, y, many of the leaves of T_x and T_y can be put into 1-1 correspondence f so that (i) the path P_x from x to leaf v of T_x is rainbow colored, (ii) the path P_y from y to the leaf $f(v)$ of T_y is rainbow colored and (iii) P_x, P_y do not share color.

Rainbow connection

4. We argue that from most of the leaves of T_x, T_y we can grow a tree of depth approximately equal to half the diameter. These latter trees themselves contain a bit more than $n^{1/2}$ leaves. These can be constructed so that they are vertex disjoint. Now we argue that each pair of trees, one associated with x and one associated with y, are joined by an edge.
5. We now have, by construction, a large set of edge disjoint paths joining leaves v of T_x to leaves $f(v)$ of T_y. A simple estimation shows that **whp** for at least one leaf v of T_x, the path from v to $f(v)$ is rainbow colored and does not use a color already used in the path from x to v in T_x or the path from y to $f(v)$ in T_y.

Rainbow connection

Lemma

Fix $t \in \mathbb{Z}^+$ and $0 < \alpha < 1$. Then, whp *there does not exist a subset $S \subseteq [n]$, such that $|S| \leq \alpha t L$ and $e[S] \geq |S| + t$.*

Proof.

For convenience, let $s = |S|$ be the cardinality of the set S.Then,

$$\mathbf{Pr}\left[\exists S : s \leq \alpha t L \;\; \text{and} \;\; e[S] \geq s + t\right] \leq \sum_{s \leq \alpha t L} \binom{n}{s} \binom{\binom{s}{2}}{s+t} p^{s+t}$$

$\square$

Rainbow connection

Proof.

$$
\begin{aligned}
&\leq \sum_{s \leq \alpha t L} \left(\frac{ne}{s}\right)^s \left(\frac{es^2 p}{2(s+t)}\right)^{s+t} \\
&\leq \sum_{s \leq \alpha t L} (e^{2+o(1)} \log n)^s \left(\frac{es \log n}{n}\right)^t \\
&\leq \alpha t L \left((e^{2+o(1)} \log n)^{\alpha L} \left(\frac{e\alpha t \log^2 n}{n \log\log n}\right)\right)^t \\
&< \frac{1}{n^{(1-\alpha-o(1))t}}.
\end{aligned}
$$

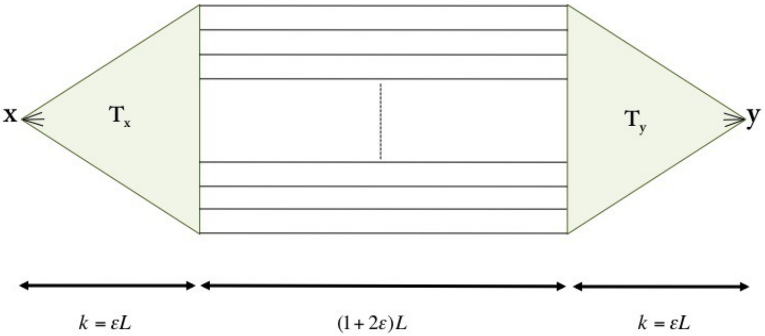
□

Rainbow connection

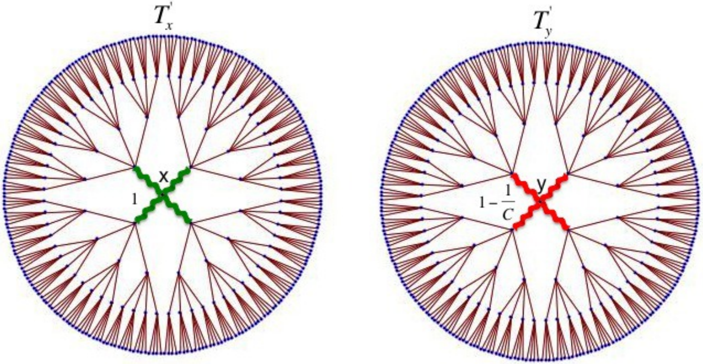
Lemma

Whp for all pairs of large vertices $x, y \in [n]$ there exists a subgraph $G_{x,y}(V_{x,y}, E_{x,y})$ of G as shown in the next figure. The subgraph consists of two isomorphic vertex disjoint trees T_x, T_y rooted at x, y each of depth k. T_x and T_y both have a branching factor of $\log n/101$. I.e. each vertex of T_x, T_y has at least $\log n/101$ neighbors, excluding its parent in the tree. Let the leaves of T_x be $x_1, x_2, \ldots, x_\tau$ where $\tau \geq n^{4\epsilon/5}$ and those of T_y be $y_1, y_2, \ldots, y_\tau$. Then $y_i = f(x_i)$ where f is a natural isomporphism that preserves the parent-child relation. Between each pair of leaves $(x_i, y_i), i = 1, 2, \ldots, \tau$ there is a path P_i of length $(1 + 2\epsilon)L$. The paths $P_i, i = 1, 2, \ldots, \tau$ are edge disjoint.

Rainbow connection



Rainbow connection



Top-down coloring, think of it as an evolutionary process. We show that there are many "alive" pairs.

Rainbow connection

Lemma

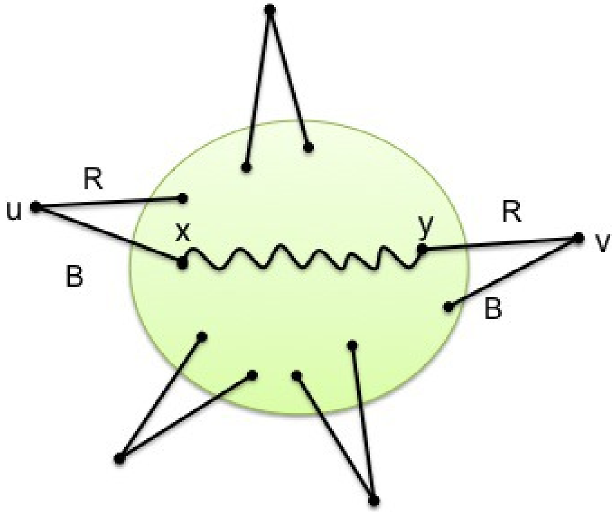
Color each edge of G using one color at random from q available. Then, the probability of having at least one rainbow path between two fixed large vertices $x, y \in [n]$ is at least $1 - \frac{1}{n^3}$.

Two key steps

- $\text{S}\textsc{tep}$ 1: Existence of at least $n^{\frac{4}{5}\epsilon}$ living pairs of leaves
- $\text{S}\textsc{tep}$ 2: Existence of rainbow paths between x, y in $G_{x,y}$

Rainbow connection

Taking care of small vertices.



Rainbow connection

Also results for random regular graphs [**?**, Frieze and Tsourakakis, 2012b].

Theorem

Let $G = G(n, r)$ be a random r-regular graph where $r \geq 3$ is a fixed integer. Then, whp

$$rc(G) = \begin{cases} O(\log^4 n) & r = 3 \\ O(\log n) & r \geq 4. \end{cases}$$

Open problem: $r = 3$

Best wishes for the rest of your studies!

references I

Alvarez-Hamelin, J. I., Dall'Asta, L., Barrat, A., and Vespignani, A. (2005).
Large scale networks fingerprinting and visualization using the *k*-core decomposition.
In *NIPS*.

Angel, A., Koudas, N., Sarkas, N., and Srivastava, D. (2012).
Dense Subgraph Maintenance under Streaming Edge Weight Updates for Real-time Story Identification.
arXiv.org.

Arora, S., Rao, S., and Vazirani, U. (2009).
Expander flows, geometric embeddings and graph partitioning.
Journal of the ACM (JACM), 56(2).

references II

Asahiro, Y., Iwama, K., Tamaki, H., and Tokuyama, T. (2000).
Greedily finding a dense subgraph.
Journal of Algorithms, 34(2):203–221.

Caro, Y., Lev, A., Roditty, Y., Tuza, Z., and Yuster, R. (2008).
On rainbow connection.
Electron. J. Combin, 15(1):R57.

Charikar, M. (2000).
Greedy approximation algorithms for finding dense components in a graph.
In *APPROX*.

references III

Fratkin, E., Naughton, B. T., Brutlag, D. L., and Batzoglou, S. (2006).
MotifCut: regulatory motifs finding with maximum density subgraphs.
Bioinformatics, 22(14).

Frieze, A. M. and Tsourakakis, C. E. (2012a).
Rainbow connection of sparse random graphs.
The Electronic Journal of Combinatorics, 19.

Frieze, A. M. and Tsourakakis, C. E. (2012b).
Rainbow connectivity of sparse random graphs.
In *APPROX-RANDOM*, pages 541–552.

Goldberg, A. V. (1984).
Finding a maximum density subgraph.
Technical report.

references IV

Håstad, J. (1997).

Clique is hard to approximate within $n^{1-\epsilon}$.

In *Electronic Colloquium on Computational Complexity (ECCC)*.

Iasemidis, L. D., Shiau, D.-S., Chaovalitwongse, W. A., Sackellares, J. C., Pardalos, P. M., Principe, J. C., Carney, P. R., Prasad, A., Veeramani, B., and Tsakalis, K. (2003).

Adaptive epileptic seizure prediction system.

IEEE Transactions on Biomedical Engineering, 50(5).

Karypis, G. and Kumar, V. (1998).

A fast and high quality multilevel scheme for partitioning irregular graphs.

SIAM J. Sci. Comput., 20(1):359–392.

references V

Khuller, S. and Saha, B. (2009).
On finding dense subgraphs.
In *ICALP*.

Krauthgamer, R., Naor, J. S., and Schwartz, R. (2009).
Partitioning graphs into balanced components.
In *SODA*.

Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. (1999).
Trawling the Web for emerging cyber-communities.
Computer Networks, 31(11–16):1481–1493.

Sozio, M. and Gionis, A. (2010).
The community-search problem and how to plan a successful cocktail party.
In *KDD*.

references VI

Stanton, I. and Kliot, G. (2012).

Streaming graph partitioning for large distributed graphs.

In *KDD*.

Tong, H. and Faloutsos, C. (2006).

Center-piece subgraphs: problem definition and fast solutions.

In *KDD*.

Tsourakakis, C. (2013).

Mathematical and Algorithmic Analysis of Network and Biological Data.

PhD thesis, Carnegie Mellon University.

references VII

Tsourakakis, C., Bonchi, F., Gionis, A., Gullo, F., and Tsiarli, M. (2013).

Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees.

In *KDD*.

Tsourakakis, C. E., Gkantsidis, C., Radunovic, B., and Vojnovic, M. (2012).

FENNEL: Streaming graph partitioning for massive scale graphs.

Technical report.

Uno, T. (2010).

An efficient algorithm for solving pseudo clique enumeration problem.

Algorithmica, 56(1).

references VIII

Zhang, B. and Horvath, S. (2005).

A general framework for weighted gene co-expression network analysis.

Statistical applications in genetics and molecular biology, 4(1):1128.