Basic principles of algorithmic graph mining
Lecture 4: Spectral graph analysis

Aristides Gionis
Aalto University
Estonian Summer School on Computer and System Science 2016
Aug 22–24, 2016

course agenda

- introduction to graph mining
- computing basic graph statistics
- finding dense subgraphs
- spectral graph analysis
- additional topics and applications

spectral graph theory

objective:

- view the adjacency (or related) matrix of a graph with a linear algebra lens
- identify connections between spectral properties of such a matrix and structural properties of the graph
  - connectivity
  - bipartiteness
  - cuts
  - ...
- spectral properties = eigenvalues and eigenvectors
- in other words, what does the eigenvalues and eigenvectors of the adjacency (or related) matrix tell us about the graph?

background: eigenvalues and eigenvectors

- consider a real $n \times n$ matrix $A$, i.e., $A \in \mathbb{R}^{n \times n}$
- $\lambda \in \mathbb{C}$ is an eigenvalue of $A$
- if there exists $x \in \mathbb{C}^n$, $x \neq 0$
  - such that $A x = \lambda x$
  - such a vector $x$ is called eigenvector of $\lambda$
- alternatively,
  $$(A - \lambda I)x = 0 \quad \text{or} \quad \det(A - \lambda I) = 0$$
- it follows that $A$ has $n$ eigenvalues
  (possibly complex and possibly with multiplicity $> 1$)

background: eigenvalues and eigenvectors

- consider a real and symmetric $n \times n$ matrix $A$
  (e.g., the adjacency matrix of an undirected graph)
- then
  - all eigenvalues of $A$ are real
  - eigenvectors of different eigenvalues are orthogonal
    i.e., if $x_1$ an eigenvector of $\lambda_1$ and $x_2$ an eigenvector of $\lambda_2$
    then $\lambda_1 \neq \lambda_2$ implies $x_1 \perp x_2$ (or $x_1^T x_2 = 0$)
- $A$ is positive semi-definite if $x^T A x \geq 0$ for all $x \in \mathbb{R}^n$
- a symmetric positive semi-definite real matrix has real and non-negative eigenvalues
background: eigenvalues and eigenvectors

- consider a real and symmetric $n \times n$ matrix $A$
- the eigenvalues $\lambda_1, \ldots, \lambda_n$ of $A$ can be ordered
  \[ \lambda_1 \leq \ldots \leq \lambda_n \]
- theorem [variational characterization of eigenvalues]
  \[ \lambda_n = \max_{x \neq 0} \frac{x^T A x}{x^T x} \]
  \[ \lambda_1 = \min_{x \neq 0} \frac{x^T A x}{x^T x} \]
  \[ \lambda_2 = \min_{x \neq 0, x^T x = 0} \frac{x^T A x}{x^T x} \]
  and “so on” for the other eigenvalues
- very useful way to think about eigenvalues

spectral graph analysis

- apply the eigenvalue characterization for graphs
- question: which matrix to consider?
  - the adjacency matrix $A$ of the graph
  - some matrix $B$ so that $x^T B x$ is related to a structural property of the graph
- consider $G = (V, E)$ an undirected and $d$-regular graph (regular graph is used wlog for simplicity of expositions)
- let $A$ be the adjacency matrix of $G$
- laplacian matrix of $G$ as
  \[ L = I - \frac{1}{d} A \]
  or
  \[ L_{ij} = \begin{cases} 1 & \text{if } i = j \in E, i \neq j \\ -\frac{1}{d} & \text{if } (i, j) \in E, i \neq j \\ 0 & \text{if } (i, j) \not\in E, i \neq j \end{cases} \]

spectral graph analysis

the smallest eigenvalue

- apply the eigenvalue characterization theorem for $L$
- what is $\lambda_1$?
  \[ \lambda_1 = \min_{x \neq 0} \frac{x^T L x}{x^T x} = \min_{x \neq 0} \frac{\sum_{(u,v) \in E} |x_u - x_v|^2}{d \sum_{u \in V} x_u^2} \]
  observe that $\lambda_1 > 0$
  can it be $\lambda_1 = 0$?
  yes: take $x$ to be the constant vector
the second smallest eigenvalue

apply the eigenvalue characterization theorem for $L$

- what is $\lambda_2$?
  \[
  \lambda_2 = \min_{x \neq 0} \frac{x^T L x}{x^T x} = \min_{x \neq 0} \frac{\sum_{(u,v) \in E} |x_u - x_v|^2}{d \sum_{u \in V} x_u^2}
  \]
- can it be $\lambda_2 = 0$?
- $\lambda_2 = 0$ if and only if the graph is connected
  map the vertices of each connected component to a different constant

the $k$-th smallest eigenvalue

- alternative characterization for $\lambda_k$
  \[
  \lambda_k = \min_{x \neq 0} \max_{x \in S \subseteq V \atop d \sum_{u \in V} x_u^2} \frac{\sum_{(u,v) \in E} |x_u - x_v|^2}{d \sum_{u \in V} x_u^2}
  \]
- $\lambda_k = 0$ if and only if the graph has at least $k$ connected components

the largest eigenvalue

- what about $\lambda_n$?
  \[
  \lambda_n = \max_{x \neq 0} \frac{x^T L x}{x^T x} = \max_{x \neq 0} \frac{\sum_{(u,v) \in E} |x_u - x_v|^2}{d \sum_{u \in V} x_u^2}
  \]
- consider a boolean version of this problem
- restrict mapping to $\{-1, 1\}$
  \[
  \lambda_n \geq \max_{x \in \{-1, 1\}^n} \frac{\sum_{(u,v) \in E} |x_u - x_v|^2}{d \sum_{u \in V} x_u^2}
  \]

the largest eigenvalue

- mapping of vertices to $\{-1, 1\}$ corresponds to a cut $S$ then
  \[
  \lambda_n \geq \max_{x \in \{-1, 1\}^n} \frac{\sum_{(u,v) \in E} |x_u - x_v|^2}{d \sum_{u \in V} x_u^2}
  = \max_{S \subseteq V} \frac{4 E(S, V \setminus S)}{d n}
  = \max_{S \subseteq V} \frac{4 E(S, V \setminus S)}{2 |E|}
  = \frac{2 \text{maxcut}(G)}{|E|}
  \]
- it follows that if $G$ bipartite then $\lambda_n \geq 2$
  (because if $G$ bipartite exists $S$ that cuts all edges)

the largest eigenvalue

- on the other hand
  \[
  \lambda_n = \max_{x \neq 0} \frac{2d \sum_{u \in V} x_u^2 - \sum_{(u,v) \in E} (x_u + x_v)^2}{d \sum_{u \in V} x_u^2}
  = 2 \min_{x \neq 0} \frac{\sum_{(u,v) \in E} (x_u + x_v)^2}{d \sum_{u \in V} x_u^2}
  \]
- First note that $\lambda_n \leq 2$
- $\lambda_n = 2$ iff there is $x$ s.t. $x_u = -x_v$ for all $(u,v) \in E$
- $\lambda_n = 2$ iff $G$ has a bipartite connected component

summary so far

eigenvalues and structural properties of $G$:

- $\lambda_2 = 0$ iff $G$ is disconnected
- $\lambda_k = 0$ iff $G$ has at least $k$ connected components
- $\lambda_n = 2$ iff $G$ has a bipartite connected component
robustness

• how robust are these results?
• for instance, what if \( \lambda_2 = \epsilon \)?
  is the graph \( G \) almost disconnected?
  i.e., does it have small cuts?
• or, what if \( \lambda_2 = 2 - \epsilon \)?
  does it have a component that is “close” to bipartite?

the second eigenvalue

\[ \lambda_2 = \min_{x \neq 0 \atop x \neq x_1} \frac{\sum_{(u,v) \in E} (x_u - x_v)^2}{\sum_{u \in V} x_u^2} = \min_{x \neq 0 \atop x \neq x_1} \frac{d(\sum_{(u,v) \in E} (x_u - x_v)^2)}{d\sum_{u \in V} x_u^2} \]

where \( V^2 \) is the set of ordered pairs of vertices

why?

\[ \sum_{(u,v) \in V^2} (x_u - x_v)^2 = \sum_{u \in V} x_u^2 - 2 \sum_{u \in V} x_u \sum_{v \in V} x_v = \sum_{u \in V} x_u^2 - 2 \left( \sum_{u \in V} x_u \right)^2 \]

and \( \sum_{u \in V} x_u = 0 \) since \( x^T x_1 = 0 \)

uniform sparsest cut

• it can be shown that
  \[ \lambda_2 \leq \text{usc}(G) \leq \sqrt{\lambda_2} \]
  the first inequality holds by definition of relaxation
  second inequality is constructive:
  • if \( x \) is an eigenvector of \( \lambda_2 \)
    then there is some \( t \in V \) such that
    the cut \( \langle S, V \setminus S \rangle = \{ u \in V \mid x_u \leq x_t \}, \{ u \in V \mid x_u > x_t \} \)
    has cost \( \text{usc}(S) \leq \sqrt{\lambda_2} \)

conductance

• conductance: another measure for cuts
• the conductance of a set \( S \subseteq V \) is defined as
  \[ \phi(S) = \frac{E(S, V \setminus S)}{d(S)} \]
  expresses the probability to “move out” of \( S \) by following a random edge from \( S \)
• we are interested in sets of small conductance
• the conductance of the graph \( G \) is defined as
  \[ \phi(G) = \min_{0 < S \subseteq V \colon |S| \geq n/2} \phi(S) \]

Cheeger’s inequality

• Cheeger’s inequality:
  \[ \frac{\lambda_2}{2} \leq \frac{\text{usc}(G)}{2} \leq \phi(G) \leq \sqrt{2 \lambda_2} \]
  ⇒ conductance is small if and only if \( \lambda_2 \) is small

• the two leftmost inequalities are “easy” to show
  the first follows by the definition of relaxation
  the second follows by
  \[ \frac{\text{usc}(S)}{2} = \frac{n E(S, V \setminus S)}{2d |S||V \setminus S|} \leq \frac{E(S, V \setminus S)}{d|S|} = \phi(S) \]
  since \( |V \setminus S| \geq n/2 \)
Cheeger's inequality

\[ \frac{\lambda_2}{2} \leq \frac{\text{usc}(G)}{2} \leq \frac{\varphi(G)}{2} \leq \sqrt{2\lambda_2} \]

- The rightmost inequality is the "difficult" one.

**Proof sketch** (three steps):
1. Consider a vector \( y > 0 \).
2. We can find a set \( S \subseteq \{ v \in V \mid y_v > 0 \} \) such that
   \[ \varphi(S) \leq \frac{\sum_{(u,v) \in E} |y_u - y_v|}{\sum_{u \in V} |y_u|} \] (no squares)
3. Pick random \( t \in [0, \max_y y_v] \) and define \( S = \{ v \mid y_v > t \} \).
4. Then \( \varphi(S) \leq \text{r.h.s. on expectation} \)
5. Thus, there is some \( t \) that the property holds.

Generalization to non-regular graphs

- \( G = (V, E) \) is undirected and non-regular
- Let \( d_u \) be the degree of vertex \( u \)
- Define \( D \) to be a diagonal matrix whose \( u \)-th diagonal element is \( d_u \)
- The normalized Laplacian matrix of \( G \) is defined
  \[ L = I - D^{-1/2} A D^{-1/2} \]
  or
  \[ L_{uv} = \begin{cases} 1 & \text{if } u = v \\ -1/\sqrt{d_u d_v} & \text{if } (u, v) \in E, u \neq v \\ 0 & \text{if } (u, v) \notin E, u \neq v \end{cases} \]

Generalization to non-regular graphs

- With the normalized Laplacian
  the eigenvalue expressions become (e.g., \( \lambda_2 \))
  \[ \lambda_2 = \min_{(x, y) \neq 0} \frac{\sum_{(u,v) \in E} (x_u - x_v)^2}{\sum_{u \in V} d_u x_u^2} \]
  where we use weighted inner product
  \[ \langle x, y \rangle_D = \sum_{u \in V} d_u x_u y_u \]

Summary so far

eigenvalues and structural properties of \( G \):
- \( \lambda_2 = 0 \) iff \( G \) is disconnected
- \( \lambda_k = 0 \) iff \( G \) has at least \( k \) connected components
- \( \lambda_2 = 2 \) iff \( G \) has a bipartite connected component
- Small \( \lambda_2 \) iff \( G \) is "almost" disconnected (small conductance)

Random walks
random walks

- consider random walk on the graph $G$ by following edges
- from vertex $i$ move to vertex $j$ with prob. $1/d_i$ if $(i,j) \in E$
- $p^{(t)}_i$ probability of being at vertex $i$ at time $t$
- process is described by equation $p^{(t+1)} = p^{(t)} P$
- process converges to stationary distribution $\pi = \pi P$
  (under certain irreducibility conditions)
- for undirected and connected graphs
  $$\pi = \frac{d_i}{2m}$$  (stationary distribution $\sim$ degree)

random walks — useful concepts

- hitting time $H(i,j)$: expected number of steps before visiting vertex $j$, starting from $i$
- commute time $\kappa(i,j)$: expected number of steps before visiting $j$ and $i$ again, starting at $i$:
  $$\kappa(i,j) = H(i,j) + H(j,i)$$
- cover time: expected number of steps to reach every node
- mixing time $\tau(\epsilon)$: a measure of how fast the random walk approaches its stationary distribution
  $$\tau(\epsilon) = \min\{t \mid d(t) \leq \epsilon\}$$
  where
  $$d(t) = \max_i ||p^t(i, \cdot) - \pi|| = \max_i \left\{ \sum_j |p^t(i,j) - \pi_j| \right\}$$

random walks vs. spectral analysis

- consider the normalized laplacian $L = I - D^{-1/2} A D^{-1/2}$
  $$L u = \lambda u$$
  $$(I - D^{-1/2} A D^{-1/2}) u = \lambda u$$
  $$(D - A) u = \lambda Du$$
  $$D u = Au + \lambda Du$$
  $$(I - \lambda) u = D^{-1} Au$$
  $$\pi u = Pu$$

  $(\lambda, u)$ is an eigenvalue–eigenvector pair for $L$ if and only if $(1 - \lambda, u)$ is an eigenvalue–eigenvector pair for $P$
  
  the eigenvector with smallest eigenvalue for $L$ is the eigenvector with largest eigenvalue for $P$

random walks vs. spectral analysis

- stochastic matrix $P$, describing the random walk
- eigenvalues: $-1 < \mu_n \leq \ldots \leq \mu_2 < \mu_1 = 1$
- spectral gap: $\gamma_1 = 1 - \mu_2 = \lambda_2$
- relaxation time: $\tau_* = \frac{1}{\gamma_*}$
- theorem: for an aperiodic, irreducible, and reversible random walk, and any $i$
  $$\left(\tau_* - 1\right) \log \left( \frac{1}{2\epsilon} \right) \leq \tau(\epsilon) \leq \tau_* \log \left( \frac{1}{2\epsilon \sqrt{\lambda_{\min}}} \right)$$

random walks vs. spectral analysis

- intuition: fast mixing related to graph being an expander

graph partitioning

small spectral gap $\iff$ large mixing time $\iff$ bottlenecks $\iff$
$\iff$ clusters $\iff$ low conductance $\iff$ small $\lambda_2$
**graph partitioning and community detection**

**motivation**
- knowledge discovery
  - partition the web into sets of related pages (web graph)
  - find groups of scientists who collaborate with each other (co-authorship graph)
  - find groups of related queries submitted in a search engine (query graph)
- performance
  - partition the nodes of a large social network into different machines so that, to a large extent, friends are in the same machine (social networks)

**graph partitioning**

(Zachary's karate club network, figure from [Newman and Girvan, 2004])

---

**basic spectral-partition algorithm**

1. form normalized Laplacian $L' = I - D^{-1/2} A D^{-1/2}$
2. compute eigenvector $x_2$ (Fielder vector)
3. order vertices according their coefficient value on $x_2$
4. consider only sweeping cuts: splits that respect the order
5. take the sweeping cut $S$ that minimizes $\phi(S)$

**theorem:** the basic spectral partition algorithm finds a cut $S$ such that $\phi(S) \leq 2 \sqrt{\phi(G)}$

**proof:** by Cheeger inequality

$$\phi(S) \leq 2 \sqrt{\lambda_2} \leq 2 \sqrt{2} \cdot \phi(G)$$

---

**spectral partitioning rules**

1. **conductance:** find the partition that minimizes $\phi(G)$
2. **bisection:** split in two equal parts
3. **sign:** separate positive and negative values
4. **gap:** separate according to the largest gap

---

**other common spectral-partitioning algorithms**

1. utilize more eigenvectors than just the Fielder vector
   use $k$ eigenvectors
2. different versions of the Laplacian matrix

---

**using $k$ eigenvectors**

- **ideal scenario:** the graph consists of $k$ disconnected components (perfect clusters)
- **then:** eigenvalue 0 of the Laplacian has multiplicity $k$
  - the eigenspace of eigenvalue 0 is spanned by indicator vectors of the graph components
using $k$ eigenvectors

using $k$ eigenvectors

robustness under perturbations: if the graph has less well-separated components the previous structure holds approximately

clustering of Euclidean points can be used to separate the components

using $k$ eigenvectors

laplacian matrices

- normalized laplacian: $L = I - D^{-1/2} A D^{-1/2}$
- unnormalized laplacian: $L_u = D - A$
- normalized "random-walk" laplacian: $L_{rw} = I - D^{-1} A$
all laplacian matrices are related

- unnormalized Laplacian: \( \lambda_2 = \min_{|x|=1} \sum_{(i,j) \in E} (x_i - x_j)^2 \)
- normalized Laplacian:
  \[
  \lambda_2 = \min_{|x|=1} \sum_{(i,j) \in E} \left( \frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2
  \]

\((\lambda, u)\) is an eigenvalue/vector of \(L_{ra}\) if and only if
\((\lambda, D^{1/2}u)\) is an eigenvalue/vector of \(L\).

\((\lambda, u)\) is an eigenvalue/vector of \(L_{ra}\) if and only if
\((\lambda, u)\) solve the generalized eigen-problem \(L_{ra} u = \lambda D u\)

algorithm 1: unnormalized spectral clustering

input graph adjacency matrix \(A\), number \(k\)
1. form diagonal matrix \(D\)
2. form unnormalized Laplacian \(L = D - A\)
3. compute the first \(k\) eigenvectors \(u_1, \ldots, u_k\) of \(L\)
4. form matrix \(U \in \mathbb{R}^{n \times k}\) with columns \(u_1, \ldots, u_k\)
5. consider the \(i\)-th row of \(U\) as point \(y_i \in \mathbb{R}^k, i = 1, \ldots, n\)
6. cluster the points \(\{y_i\}_{i=1}^n\) into clusters \(C_1, \ldots, C_k\)
e.g., with \(k\)-means clustering
output clusters \(A_1, \ldots, A_k\) with \(A_i = \{j \mid y_j \in C_i\}\)

algorithm 2: normalized spectral clustering

[Shi and Malik, 2000]
input graph adjacency matrix \(A\), number \(k\)
1. form diagonal matrix \(D\)
2. form unnormalized Laplacian \(L = D - A\)
3. compute the first \(k\) eigenvectors \(u_1, \ldots, u_k\) of the
   generalized eigenproblem \(LU = \lambda DU\) (eigvecrs of \(L_{ra}\))
4. form matrix \(U \in \mathbb{R}^{n \times k}\) with columns \(u_1, \ldots, u_k\)
5. consider the \(i\)-th row of \(U\) as point \(y_i \in \mathbb{R}^k, i = 1, \ldots, n\)
6. cluster the points \(\{y_i\}_{i=1}^n\) into clusters \(C_1, \ldots, C_k\)
e.g., with \(k\)-means clustering
output clusters \(A_1, \ldots, A_k\) with \(A_i = \{j \mid y_j \in C_i\}\)

algorithm 3: normalized spectral clustering

[Ng et al., 2001]
input graph adjacency matrix \(A\), number \(k\)
1. form diagonal matrix \(D\)
2. form normalized Laplacian \(L' = I - D^{-1/2}AD^{-1/2}\)
3. compute the first \(k\) eigenvectors \(u_1, \ldots, u_k\) of \(L'\)
4. form matrix \(U \in \mathbb{R}^{n \times k}\) with columns \(u_1, \ldots, u_k\)
5. normalize \(U\) so that rows have norm 1
6. consider the \(i\)-th row of \(U\) as point \(y_i \in \mathbb{R}^k, i = 1, \ldots, n\)
7. cluster the points \(\{y_i\}_{i=1}^n\) into clusters \(C_1, \ldots, C_k\)
e.g., with \(k\)-means clustering
output clusters \(A_1, \ldots, A_k\) with \(A_i = \{j \mid y_j \in C_i\}\)

notes on the spectral algorithms

- quite similar except for using different Laplacians
- can be used to cluster any type of data, not just graphs
- form all-pairs similarity matrix and use as adjacency matrix
- computation of the first eigenvectors of sparse matrices
  can be done efficiently using the Lanczos method

Zachary’s karate-club network
**Zachary’s karate-club network**

- Unnormalized Laplacian
- Normalized symmetric Laplacian
- Normalized random walk Laplacian

**Which Laplacian to use?**

[von Luxburg, 2007]

- When graph vertices have about the same degree all laplacians are about the same
- For skewed degree distributions, normalized laplacians tend to perform better
- Normalized laplacians are associated with conductance, which is a good objective (conductance involves \( \text{vol}(S) \) rather than \(|S|\) and captures better the community structure)

**Modularity**

- What measures (conduction) useful to find one component
- How to find many components?
- Related question: What is the optimal number of partitions?
- Modularity has been used to answer those questions
  [Newman and Girvan, 2004]
- Originally developed to find the optimal number of partitions in hierarchical graph partitioning

**Values of modularity**

- 0 random structure; 1 strong community structure
- \([0.3...0.7]\): Typical good structure; can be negative, too
- \(Q\) measure is not monotone with \(k\)

\[
Q = \frac{1}{2m} \sum_i (A_{ij} - P_{ij}) \chi(C_i, C_j)
\]

\[
= \frac{1}{2m} \sum_i (A_{ij} - \frac{d_i d_j}{2m}) \chi(C_i, C_j)
\]

\[
= \sum_c \left[ \frac{m_c}{2m} - \left( \frac{d_c}{2m} \right)^2 \right]
\]

\(P_{ij} = 2mp_{ij} = 2m(d_i/2m)(d_j/2m) = (d_i d_j/2m)\)

- \(m_c\): Edges within cluster \(c\)
- \(d_c\): Total degree of cluster \(c\)

(figures from [Clauset et al., 2004])
optimizing modularity

- problem: find the partitioning that optimizes modularity
- NP-hard problem [Brandes et al., 2006]
- top-down approaches [Newman and Girvan, 2004]
- spectral approaches [Smyth and White, 2005]
- mathematical-programming [Agarwal and Kempe, 2008]

top-down algorithms for optimizing modularity

[Newman and Girvan, 2004]
- a set of algorithms based on removing edges from the graph, one at a time
- the graph gets progressively disconnected, creating a hierarchy of communities

頂部向下的算法

general scheme

1. **Top-Down**
2. compute betweenness value of all edges
3. remove the edge with the highest betweenness
4. recompute betweenness value of all remaining edges
5. repeat until no edges left

shortest-path betweenness

- how to compute shortest-path betweenness?
  - BFS from each vertex
  - leads to $O(mn)$ for all edge betweenness
  - OK if there are single paths to all vertices
**shortest-path betweenness**

- stochastic matrix of random walk is $P = D^{-1} A$
- $s$ is the vector with 1 at position $s$ and 0 elsewhere
- probability distribution over vertices at time $n$ is $s P^n$
- expected number of visits at each vertex given by
  \[ \sum_n s P^n = s (1 - P)^{-1} \]
- $c_u = E[\text{# times passing from } u \text{ to } v] = [s (1 - P)^{-1}]_u \frac{1}{d_u}$
  \[ c = s (1 - P)^{-1} D^{-1} = s (D - A)^{-1} \]
- define random-walk betweenness at $(u, v)$ as $|c_u - c_v|$

**random-walk betweenness**

- [Newman and Girvan, 2004] recommend shortest-path betweenness

**other modularity-based algorithms**

**spectral approach** [Smyth and White, 2005]

\[
Q = \sum_{c=1}^{k} \left[ \frac{m_c}{2m} - \left( \frac{d_c}{2m} \right)^2 \right] \propto \sum_{c=1}^{k} \left[ (2m) m_c - d_c^2 \right]
\]

\[
= \sum_{c=1}^{k} \left[ (2m) \sum_{j=1}^{n} w_{ij} x_{ij} x_{jc} - \left( \sum_{j=1}^{n} d_j x_{jc} \right)^2 \right]
\]

\[
= \sum_{c=1}^{k} \left[ (2m) x_{c}^T W x_{c} - x_{c}^T D x_{c} \right]
\]

\[
= \text{tr}(X^T (W' - D) X)
\]

where $X = [x_1 \ldots x_k] = [x_c]$ point-cluster assignment matrix

**spectral-based modularity optimization**

Maximize $\text{tr}(X^T (W' - D) X)$

such that $X$ is an assignment matrix

Solution:

\[
L_Q X = X \Lambda
\]

Where $L_Q = W' - D$, $Q$-Laplacian

- standard eigenvalue problem
- but solution is fractional, we want integral
- treat rows of $X$ as vectors and cluster graph vertices using $k$-means
- [Smyth and White, 2005] propose two algorithms, based on this idea
spectral-based modularity optimization

spectral algorithms perform almost as good as the agglomerative, but they are more efficient

\[ Q \propto \sum_{i,j=1}^{n} B_{ij}(1 - x_{ij}) \]

where

\[ x_{ij} = \begin{cases} 
0 & \text{if } i \text{ and } j \text{ get assigned to the same cluster} \\
1 & \text{otherwise} 
\end{cases} \]

it should be

\[ x_{ik} \leq x_{ij} + x_{jk} \quad \text{for all vertices } i, j, k \]

solve the integer program with triangle inequality constraints

mathematical-programming approach for modularity optimization

[Agarwal and Kempe, 2008]

- integer program is NP-hard
- relax integrality constraints
- replace \( x_{ij} \in \{0, 1\} \) with \( 0 \leq x_{ij} \leq 1 \)
- corresponding linear program can be solved in polynomial time
- solve linear program and round the fractional solution
- place in the same cluster vertices \( i \) and \( j \) if \( x_{ij} \) is small (pivot algorithm [Alon et al., 2008])

need for scalable algorithms

- spectral, agglomerative, LP-based algorithms
- not scalable to very large graphs
- handle datasets with billions of vertices and edges
  - facebook: \(~1\) 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 online content indexing using twitter posts
  - election trends, twitter as election barometer

other modularity-based algorithms

mathematical programming [Agarwal and Kempe, 2008]

\[ Q \propto \sum_{i,j=1}^{n} B_{ij}(1 - x_{ij}) \]

where

\[ x_{ij} = \begin{cases} 
0 & \text{if } i \text{ and } j \text{ get assigned to the same cluster} \\
1 & \text{otherwise} 
\end{cases} \]

it should be

\[ x_{ik} \leq x_{ij} + x_{jk} \quad \text{for all vertices } i, j, k \]

solve the integer program with triangle inequality constraints

Results

<table>
<thead>
<tr>
<th>Network</th>
<th>size</th>
<th>n</th>
<th>GN</th>
<th>DA</th>
<th>EG</th>
<th>VP</th>
<th>LP</th>
<th>UB</th>
</tr>
</thead>
<tbody>
<tr>
<td>KARATE</td>
<td>40</td>
<td>40</td>
<td>0.40</td>
<td>0.419</td>
<td>0.419</td>
<td>0.420</td>
<td>0.420</td>
<td>0.420</td>
</tr>
<tr>
<td>DOLPH</td>
<td>62</td>
<td>62</td>
<td>0.560</td>
<td>-</td>
<td>-</td>
<td>0.560</td>
<td>0.560</td>
<td>0.560</td>
</tr>
<tr>
<td>MIS</td>
<td>74</td>
<td>74</td>
<td>0.540</td>
<td>-</td>
<td>-</td>
<td>0.560</td>
<td>0.560</td>
<td>0.560</td>
</tr>
<tr>
<td>BOOKS</td>
<td>105</td>
<td>105</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.526</td>
<td>0.526</td>
<td>0.526</td>
</tr>
<tr>
<td>BALL</td>
<td>115</td>
<td>115</td>
<td>0.694</td>
<td>-</td>
<td>-</td>
<td>0.694</td>
<td>0.694</td>
<td>0.694</td>
</tr>
<tr>
<td>TASS</td>
<td>188</td>
<td>188</td>
<td>0.401</td>
<td>0.419</td>
<td>0.414</td>
<td>0.445</td>
<td>0.445</td>
<td>0.445</td>
</tr>
<tr>
<td>COLL</td>
<td>234</td>
<td>234</td>
<td>0.729</td>
<td>-</td>
<td>-</td>
<td>0.803</td>
<td>0.803</td>
<td>0.803</td>
</tr>
<tr>
<td>META</td>
<td>453</td>
<td>453</td>
<td>0.403</td>
<td>0.434</td>
<td>0.435</td>
<td>0.450</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMAIL</td>
<td>1135</td>
<td>1135</td>
<td>0.352</td>
<td>0.574</td>
<td>0.572</td>
<td>0.579</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2. The modularity obtained by many of the previously published methods and by the methods introduced in this paper, along with the upper bound.

(graph from [Agarwal and Kempe, 2008])

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

\[ G = (V, E) \]

- 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)
- followed by uncoarsening phase in which the bisection is successively refined and applied to larger graphs

summary

- spectral analysis reveals structural properties of a graph
- used for graph partitioning, but also for other problems
- well-studied area, many results and techniques
- for graph partitioning and community detection many other methods are available

acknowledgements

Luca Trevisan

references

Modularity-maximizing graph communities via mathematical programming.
The European Physical Journal B, 66(3).

Aggregating inconsistent information: ranking and clustering.
Journal of the ACM (JACM), 55(5).

Brandes, U., Delling, D., Gaertler, M., Görke, R., Höfler, M., Nikoloski, Z.,
Maximizing modularity is hard.

Finding community structure in very large networks.
arXiv.org.

references (cont.)

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

Fast algorithm for detecting community structure in networks.
Physical review E, 69(6).

Finding and evaluating community structure in networks.
Physical Review E, 69(2).

On spectral clustering: Analysis and an algorithm.
NIPS.

Normalized cuts and image segmentation.
IEEE transactions on Pattern Analysis and Machine Intelligence, 22(8).

references (cont.)

A spectral clustering approach to finding communities in graphs.
SDM.

A Tutorial on Spectral Clustering.
arXiv.org.