Graph Layout Algorithmics

Special Topics on Visualization in Network Science
CS 7280-03

Daniel K. Weidele

September 22, 2016

Graph Layout Algorithmics

Special Topics on Visualization in Network Science

Overview

What is Network Science?[1]

Introduction

We view network science as the study of the collection, management, analysis, interpretation, and presentation of relational data.

[1] Brandes et al.: What is network science? (2013).

What is Network Science?[2]

Introduction

- *Network Model*

  phenomena $\overset{\text{abstraction}}{\rightarrow}$ network concept $\overset{\text{representation}}{\rightarrow}$ network data

- *Network Science* is the study of Network Models

[2] Brandes et al.: What is network science? (2013).

What is Network Science?[3]

Introduction

Network abstraction

A *network* is a mapping $x : \mathcal{S} \to \mathcal{W}$ assigning *values* in a range $\mathcal{W}$ to *dyads* from a finite domain $\mathcal{S} \subseteq \mathcal{N} \times \mathcal{A}$ comprised of ordered pairs of *nodes* $\mathcal{N}$ and *affiliations* $\mathcal{A}$.

[3] Brandes et al.: What is network science? (2013).

Karate club[4]

Introduction

- Karate club will split into two groups
- Can you predict the *partitioning*?

Visualization of the Karate club network

- This silly *graph layout* is not useful for the task!

[4] Zachary: An information flow model for conflict and fission in small groups (1977).

Graph layouts

Introduction

Basic notation

- Graph $G = (V, E)$
- V the set of vertices
- Edges $e_{ij} \in E$ between vertices $i, j \in V$
- Number of vertices $n = |V|$
- Number of edges $m = |E|$

Graph layout

- d-dimensional embedding of G, typically $d \in \{2, 3\}$
- vertex coordinates $X \in \mathbb{R}^{n \times d}$

Graph layouts

Introduction

Data: Graph $G = (V, E)$
Result: $X \in \mathbb{R}^{n \times 2}$
init X;
for $v \in V$ **do**
    $X[v][0] \leftarrow 0$;
    $X[v][1] \leftarrow 0$;
end

Algorithm 1: Silly layout

Graph layouts

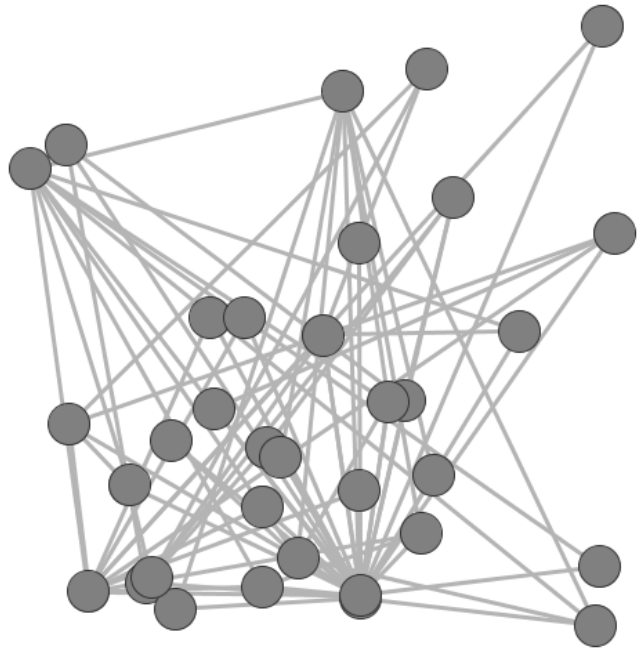
Introduction

Data: Graph $G = (V, E)$
Result: $X \in \mathbb{R}^{n \times 2}$
init X;
for $v \in V$ **do**
    $X[v][0] \leftarrow$ *random()*;
    $X[v][1] \leftarrow$ *random()*;
end

Algorithm 2: Random layout

Karate club

Can you predict the *partitioning*?



Random layout of the Karate club network

What can you see? $n \approx 30$, *density* is low, *degree* ≈ 2 to 15

A heuristic for graph drawing[5]

Spring embedding

To embed a graph we replace the vertices by steel rings and replace each edge with a spring to form a mechanical system. The vertices are placed in some initial layout and let go so that the spring forces on the rings move the system to a minimal energy state. Two practical adjustments are made to this idea: firstly, logarithmic strength springs are used; that is, the force exerted by a spring is: $c_1 \cdot \log \frac{d}{c_2}$, where d is the length of the spring, and c_1 and c_2 are constants. Experience shows that Hookes Law (linear) springs are too strong when the vertices are far apart; the logarithmic force solves this problem. Note that the springs exert no force when $d = c_2$. Secondly, we make nonadjacent vertices repel each other. An inverse square law force, $\frac{c_3}{\sqrt{d}}$, where c_3 is constant and d is the distance between the vertices, is suitable.

[5] Eades: A heuristics for graph drawing (1984).

A heuristic for graph drawing

Spring embedding[6]

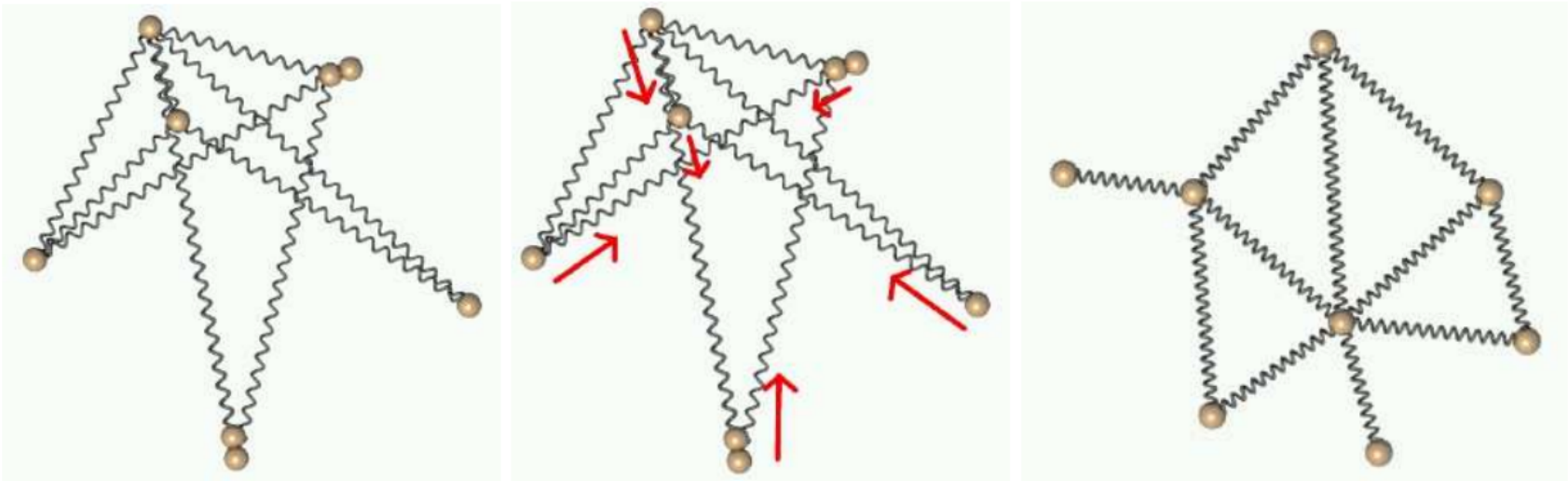


Illustration of a generic spring embedder: starting from random positions, treat the graph as spring system and look for a stable configuration.

[6] Kobourov: Spring embedders and force directed graph drawing algorithms (2012).

A heuristic for graph drawing

Spring embedding

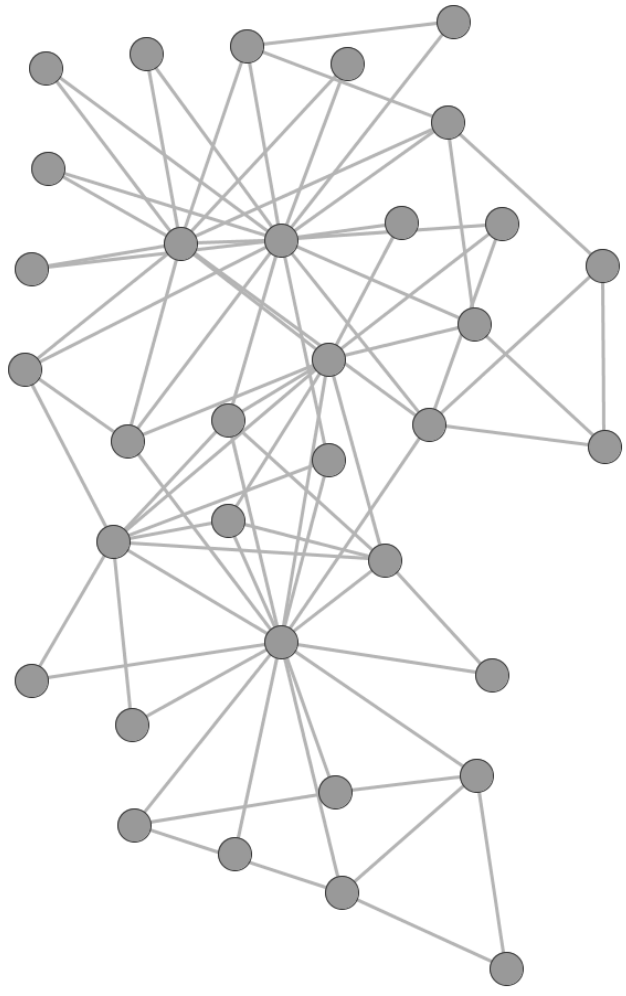
Data: Graph $G = (V, E), c_1, c_2, c_3, c_4, k$
Result: $X \in \mathbb{R}^{n \times 2}$
$X \leftarrow randomLayout(G)$; `// see Algorithm 2`
for $i \leftarrow 0$ **to** k **do**
  for $v \in V$ **do**
    for $u \in V$ **do**
      $move(u, force(u, v))$; `// Vector calculus`
    end
  end
end

Algorithm 3: Spring embedder

Originally $c_1 = 2, c_2 = 1, c_3 = 1, c_4 = 0.1$ and $k = 100$.

Karate club

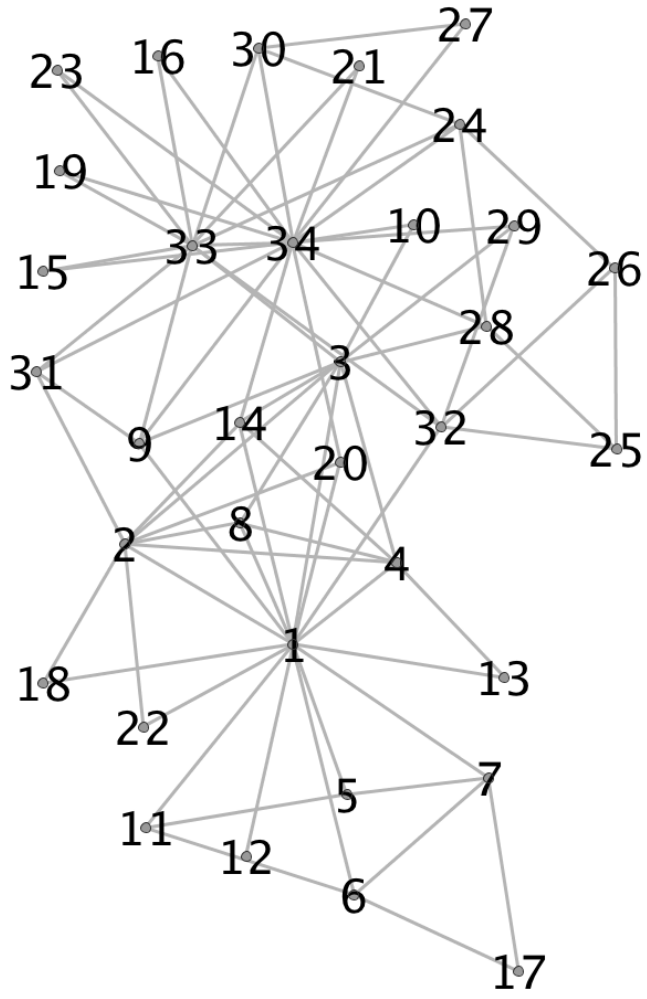
Can you predict the *partitioning*?



Spring embedding of the Karate club network

Karate club

Can you predict the *partitioning*?



Spring embedding of the Karate club network

Summary

Overview

- Basic notation
- Graph layout problem
- Spring embedding

Layout readability

Readability Metrics[7]

Layout readability

Basic idea

- Set of *fast, little measures* $m \in M$ to measure readability
- Evaluate various *aesthetic criteria* of a graph layout
- Map scores into *common range* $m : (G, X) \mapsto [0, 1]$

[7] Dunne et al.: Readability metric feedback for aiding node-link visualization designers (2015).

Avoid divisions by zero

Layout Readability

$$\text{divOrZero}(a, b) = \begin{cases} \frac{a}{b} & \text{if } b > 0 \\ 0 & \text{otherwise} \end{cases}$$

Node Overlap $\mathfrak{N}_G$

Layout Readability

Global:

$$a = \text{area}(\bigcup_{v \in V} \text{bounds}(v))$$

$$a_{max} = \text{area}(\sum_{v \in V} \text{bounds}(v))$$

$$a_\Delta = \operatorname*{argmax}_{v \in V} \text{area}(\text{bounds}(v))$$

$$\mathfrak{N}_G = \text{divOrZero}(a - a_\Delta, a_{max} - a_\Delta)$$

Node Overlap $\mathfrak{N}_V$

Layout Readability

Local (vertex):

$$a(v) = \text{area}\Big(\bigcup_{v_i \in Vn\{v\}} \text{bounds}(v) \cap \text{bounds}(v_i)\Big)$$

$$a_{max}(v) = \text{area}(\text{bounds}(v))$$

$$\mathfrak{N}_V(v) = 1 - \text{divOrZero}(a(v), a_{max}(v))$$

Edge Crossings $\mathfrak{E}_G$

Layout Readability

Global:

$$c_{all} = \sum_{i=1}^{m}(i-1) = \frac{m(m-1)}{2}$$

$$c_{impossible} = \frac{1}{2}\sum_{v \in V} d(v)(d(v)-1)$$

$$c_{max} = c_{all} - c_{impossible}$$

$$\mathfrak{E}_G = 1 - \text{divOrZero}(c, c_{max})$$

Edge Crossings $\mathfrak{E}_E$

Layout Readability

Local (edge):

$$c_{all}(e) = m - 1$$

$$c_{impossible}(e) = d(\text{source}(e)) + d(\text{target}(e)) - 2$$

$$c_{max}(e) = c_{all}(e) - c_{impossible}(e)$$

$$\mathfrak{E}_E(e) = 1 - \text{divOrZero}(c(e), c_{max}(e))$$

Edge Crossings $\mathfrak{E}_V$

Layout Readability

Local (vertex):

$$c(v) = \sum_{e \in \text{edges}(v)} c(e)$$

$$c_{max}(v) = \sum_{e \in \text{edges}(v)} c_{max}(e)$$

$$\mathfrak{E}_V(v) = 1 - \text{divOrZero}(c(v), c_{max}(v))$$

Edge Crossing Angle $\mathfrak{A}_G$

Layout Readability

Global:

$$d = \sum_{e \in E} \sum_{e_i \in c(e)} |\measuredangle - \angle(e, e_i)|$$

$$d_{max} = c\measuredangle$$

$$\mathfrak{A}_G = 1 - \text{divOrZero}(d, d_{max})$$

Edge Crossing Angle $\mathfrak{A}_E$

Layout Readability

Local (edge):

$$d(e) = \sum_{e_i \in c(e)} |\measuredangle - \angle(e, e_i)|$$

$$d_{max}(e) = c(e)\measuredangle$$

$$\mathfrak{A}_E(e) = 1 - \text{divOrZero}(d(e), d_{max}(e))$$

Angular Resolution $\mathfrak{R}_G$

Layout Readability

Global:

$$\measuredangle(v) = \frac{360^\circ}{d(v)}$$

$$d = \frac{1}{n} \sum_{v \in V} \left| \frac{\measuredangle(v) - \angle^{min}(v)}{\measuredangle(v)} \right|$$

$$\mathfrak{R}_G = 1 - d$$

Angular Resolution $\mathfrak{R}_V$

Layout Readability

Local (vertex):

$$\measuredangle(v) = \frac{360^\circ}{d(v)}$$

$$d(v) = |\frac{\measuredangle(v) - \angle^{min}(v)}{\measuredangle(v)}|$$

$$\mathfrak{R}_V(v) = 1 - d(v)$$

Distance Coherence $\mathfrak{D}_V$, $\mathfrak{D}_G$

Layout Readability

Local (vertex):

$$d_{u,v} := \text{distance between u,v in embedding}$$

$$d(u,v) := \text{graph-theoretic distance between u,v}$$

$$\mathfrak{D}_V(u) = \frac{1}{|V|-1} \sum_{v \in V \setminus \{u\}} \frac{2 - \min(2, \frac{\frac{d_{u,v}}{d(u,v)} - \overline{E}}{\sigma(\overline{E})})}{2}$$

Global:

$$\mathfrak{D}_G = \frac{1}{n \times (n-1)} \sum_{u \in V} \mathfrak{D}_V(u)$$

Distance Coherence $\mathfrak{D}_E$

Layout Readability

Local (edge):

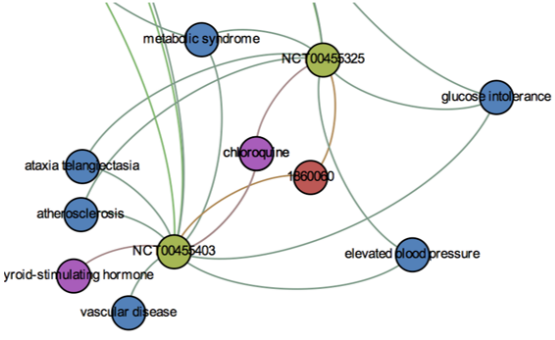
$$\mathfrak{D}_E(e) = \frac{2 - \min(2, \frac{\overline{e} - \overline{E}}{\sigma(\overline{E})})}{2}$$

Example 1: Airports, node overlap, crossing angle

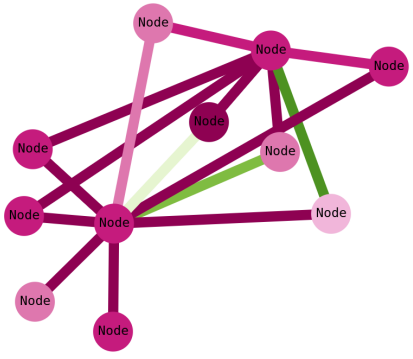
Layout Readability

Example 2: Drug network, distance coherence

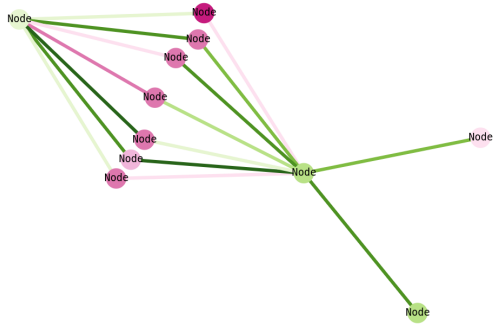
Layout Readability



Black box layout of drug network



Distance coherence



Alternative layout

Summary

Layout Readability

We have seen

- Node Overlap $\mathfrak{N}_G$, $\mathfrak{N}_V$
- Edge Crossings $\mathfrak{E}_G$, $\mathfrak{E}_E$, $\mathfrak{E}_V$
- Crossing Angle $\mathfrak{A}_G$, $\mathfrak{A}_E$
- Angular Resolution $\mathfrak{R}_G$, $\mathfrak{R}_V$
- Distance Coherence $\mathfrak{D}_G$, $\mathfrak{D}_V$, $\mathfrak{D}_E$

and there is more!

- Area coverage
- Group / cluster overlap
- Shape
- Symmetry

Layout algorithmics

Layout algorithmics

Spring embedding...

...is slow

- Naïve spring embedders are *not practical* for large networks (with complexity in $O(n^2)$)

...is not deterministic

- Random initialization leads to *different results*, every time we run the algorithm

...results in a local optimum

- The system can reach *local convergence* before some lower energy state is found.

Layout algorithmics

Spring embedding...

...is slow Barnes-Hut

- Naïve spring embedders are not practical for large networks (complexity in $O(n^2)$).

...is not deterministic Pseudorandom

- Random initialization leads to different results, every time we run the algorithm.

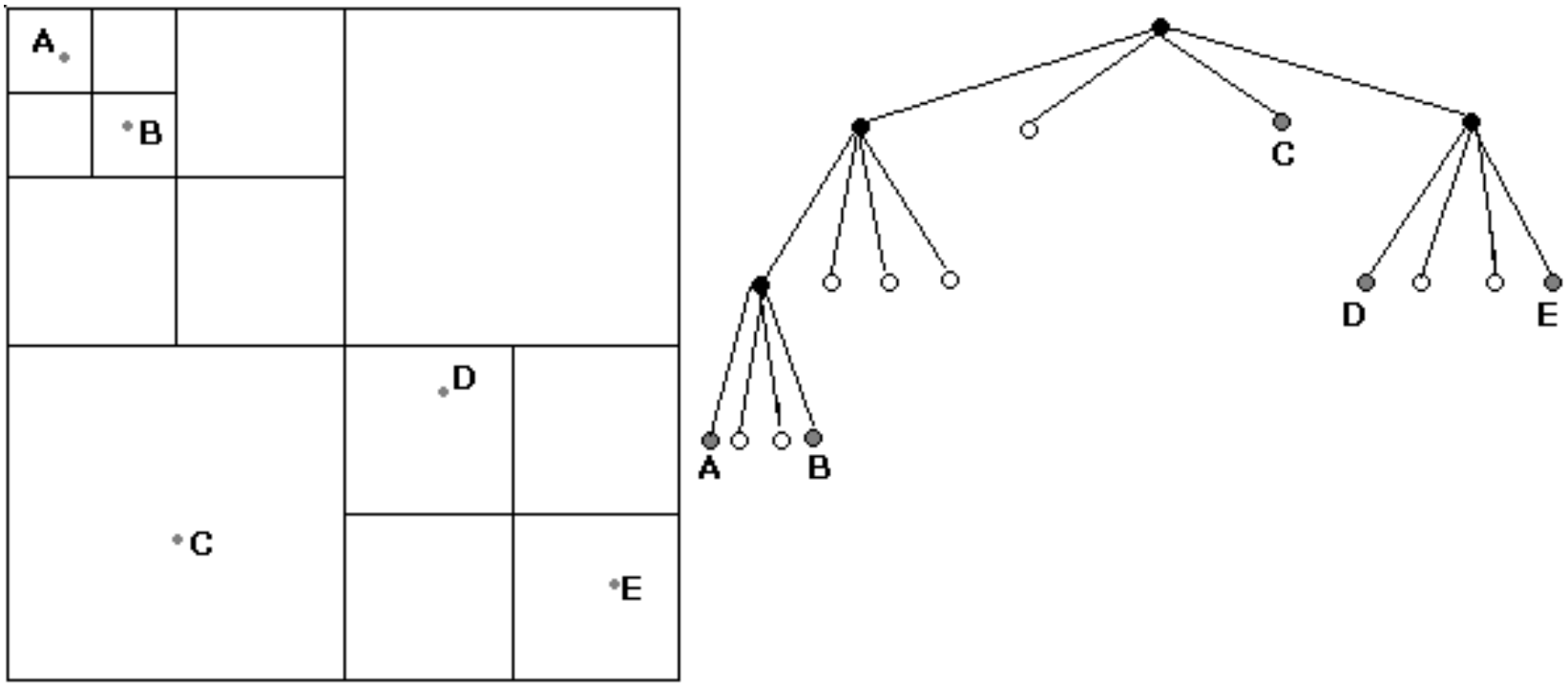
...results in a local optimum Stress Majorization

- The system can reach local convergence before some lower energy state is found.

Barnes-Hut optimization[8]

Layout algorithmics

- *Quad-tree* data structure
- Inner tree-nodes accumulate *center of* and *total mass*



Quad-tree representation of a graph layout

[8] Barnes/Hut: A hierarchical O (N log N) force-calculation algorithm (1986).

Barnes-Hut optimization

Layout algorithmics

Data: Quad-tree Q, Tree-node ∇,Vertex v, Threshold θ
Result: force vector $\vec{f}$ for v
$\vec{f} \leftarrow (0,0)$;
if ∇ *is null* **then**
  return $\vec{f}$;
end
if ∇ *is leaf* $||$ $\frac{\nabla.width}{d(\nabla,v)} < \theta$ **then**
  $\vec{f} \leftarrow \vec{f} + force(\nabla, v)$;
end
else
  $\vec{f} \leftarrow \vec{f} + self(Q, \nabla.NW, v, \theta)$; $\vec{f} \leftarrow \vec{f} + self(Q, \nabla.NE, v, \theta)$;
  $\vec{f} \leftarrow \vec{f} + self(Q, \nabla.SW, v, \theta)$; $\vec{f} \leftarrow \vec{f} + self(Q, \nabla.SW, v, \theta)$;
end
return $\vec{f}$;

Algorithm 4: Barnes-Hut optimization

Barnes-Hut optimization

Layout algorithmics

Data: Graph $G = (V, E), k$, Threshold θ
Result: $X \in \mathbb{R}^{n \times 2}$
$X \leftarrow randomLayout(G)$ init Q;
for $i \leftarrow 0$ **to** k **do**
  for $v \in V$ **do**
    $temp \leftarrow v$;
    $v \leftarrow$ move(v, BarnesHut($Q, Q.root, v, \theta$));
    update($Q, v, temp$); // $O(logn)$
  end
end

Algorithm 5: Force-layout with Barnes-Hut optimization

Classical MDS[9]

Layout algorithmics

- δ_{ij}: graph-theoretic distance between nodes i and j
- Find coordinates $X \in \mathbb{R}^{n \times d}$, such that $\delta_{ij} \approx ||x_i - x_j||$.
- Consider matrix $B = XX^T$ of inner products $b_{ij} = {x_i}^T x_j$.
- Can be shown that

$$b_{ij} = -\frac{1}{2}(\delta_{ij}^2 - \frac{1}{n}\sum_{r=1}^{n}\delta_{rj}^2 - \sum_{s=1}^{n}\delta_{is}^2 + \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n}\delta_{rs}^2)$$

- From $B = U\lambda U^T$ we can derive coordinates $X = U(d)\lambda_{(d)}^{1/2}$, with $\lambda_{(d)}$ the diagonal matrix of d largest Eigenvalues.

$\rightarrow$ **Global optimal solution**

$\rightarrow$ Problem: Eigendecomposition of B in $O(n^3)$

[9] Torgerson: Multidimensional scaling: I. Theory and method (1952).

Pivot MDS[10]

Layout algorithmics

- Idea: Base decomposition on selected $n \times k$-submatrix C:

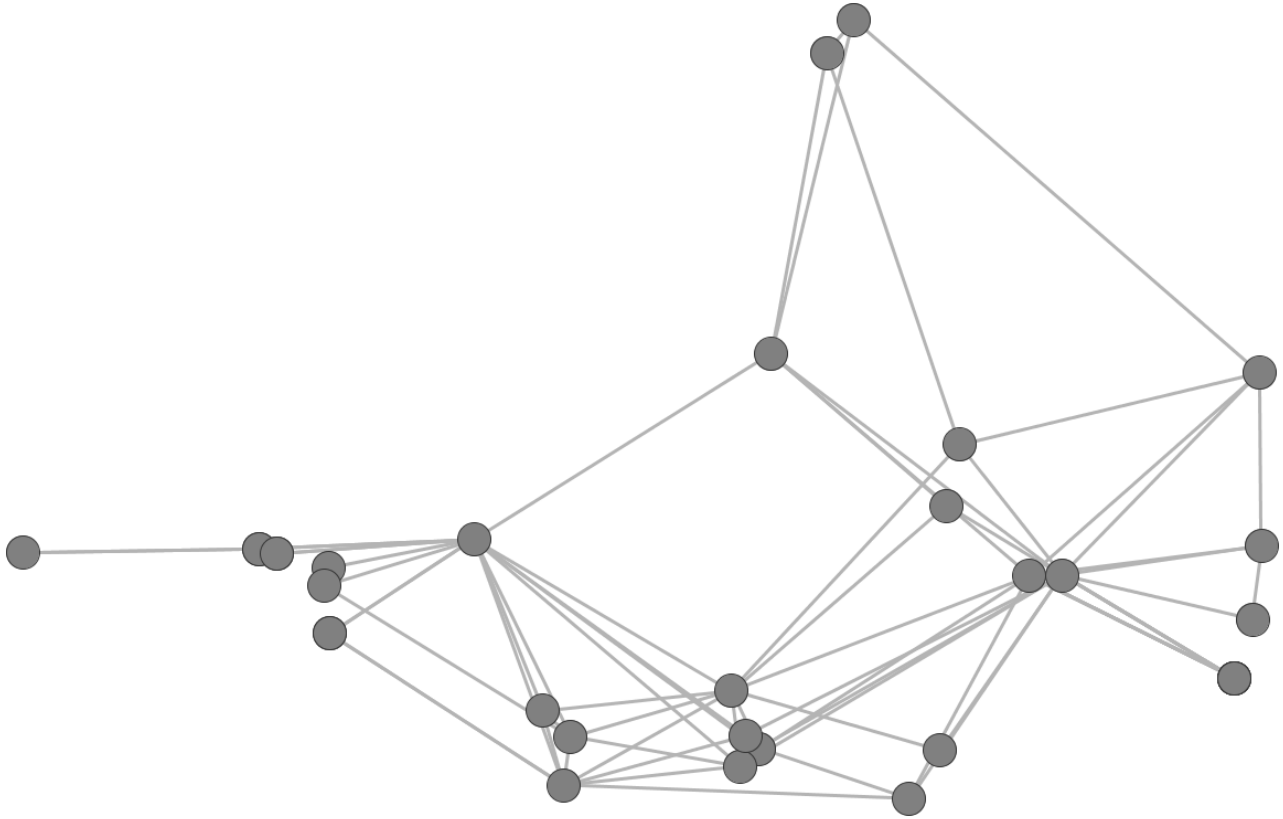
$$c_{ij} = -\frac{1}{2}(\delta_{ij}^2 - \frac{1}{\mathbf{n}} \sum_{r=1}^{n} \delta_{rj}^2 - \sum_{s=1}^{\mathbf{k}} \delta_{is}^2 + \frac{1}{\mathbf{nk}} \sum_{r=1}^{n} \sum_{s=1}^{n} \delta_{rs}^2)$$

$\rightarrow$ In $O(k^3 + k^2 n)$

[10] Brandes/Pich: Eigensolver methods for progressive multidimensional scaling of large data (2007).

Karate club

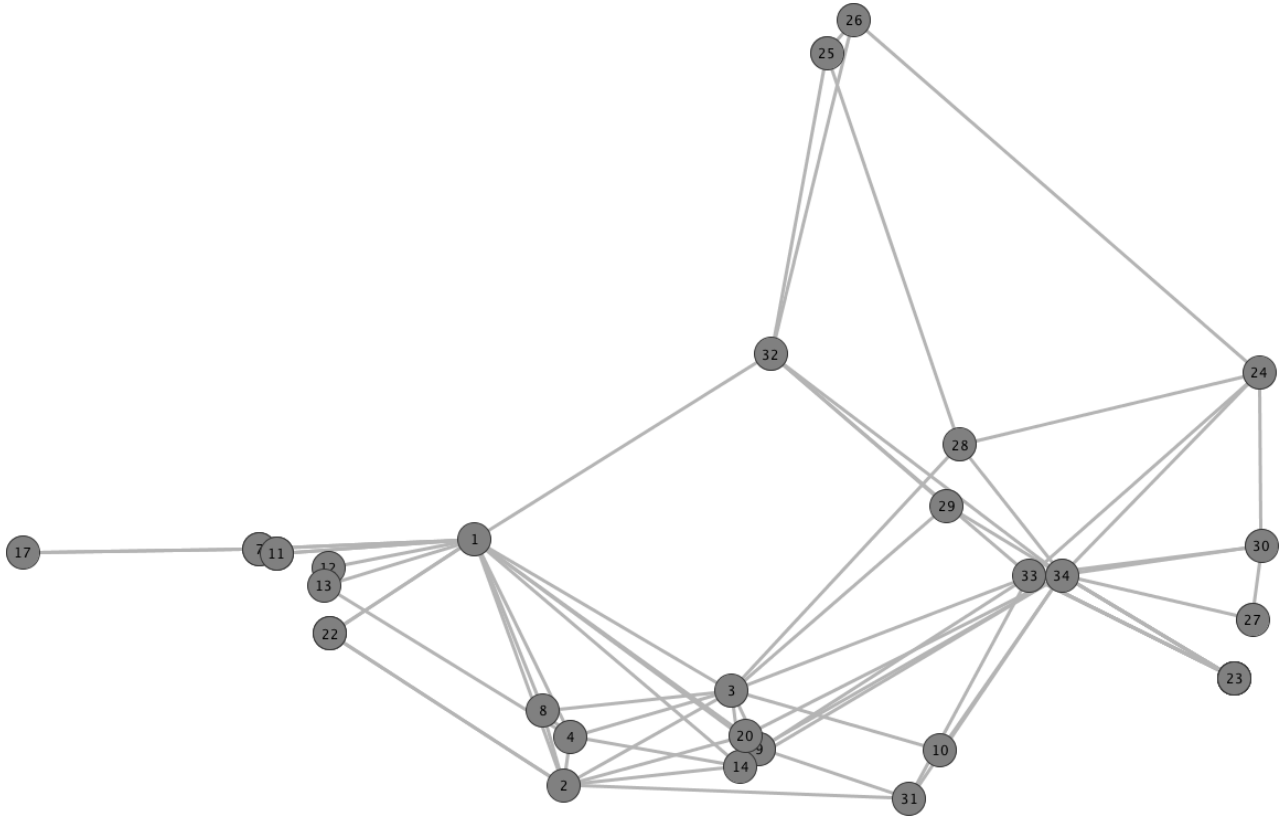
Can you predict the *partitioning*?



Classical MDS of the Karate club network

Karate club

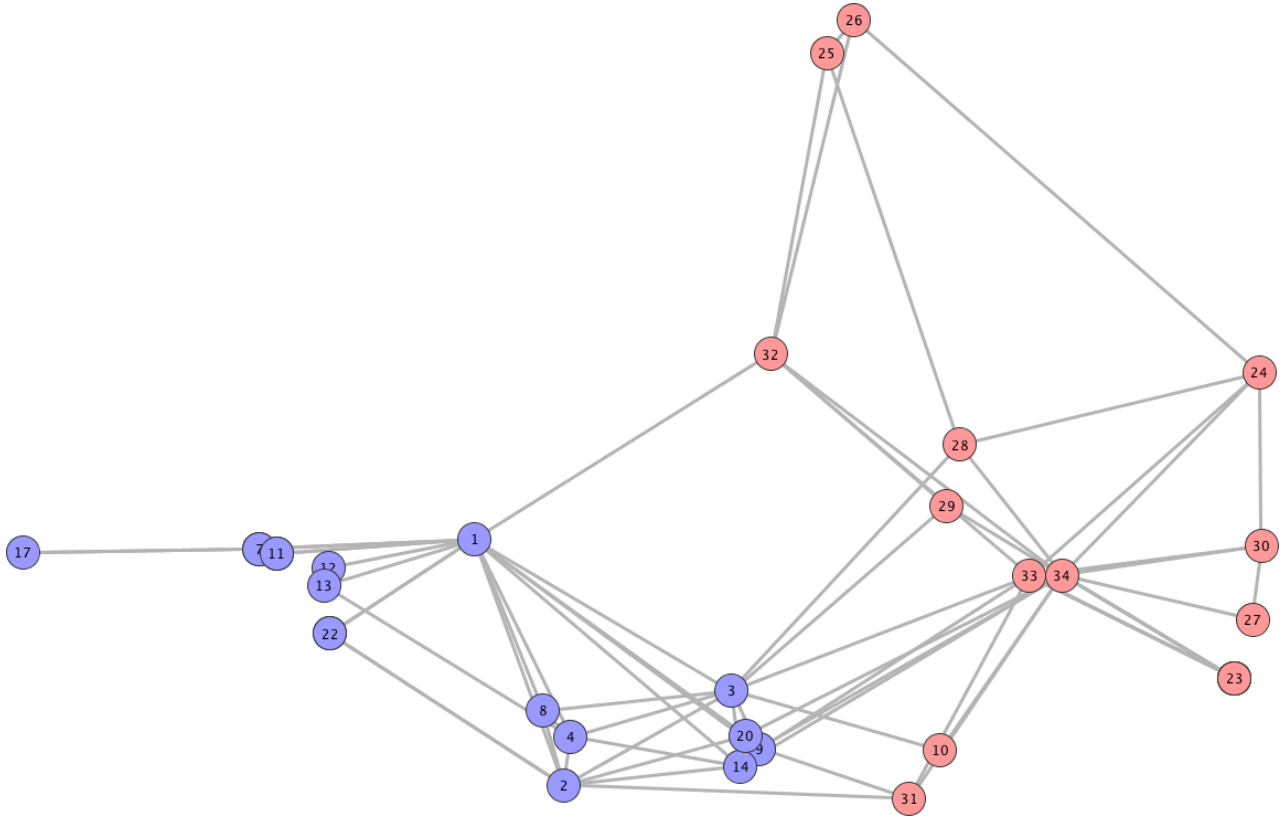
Can you predict the *partitioning*?



Classical MDS of the Karate club network

Karate club

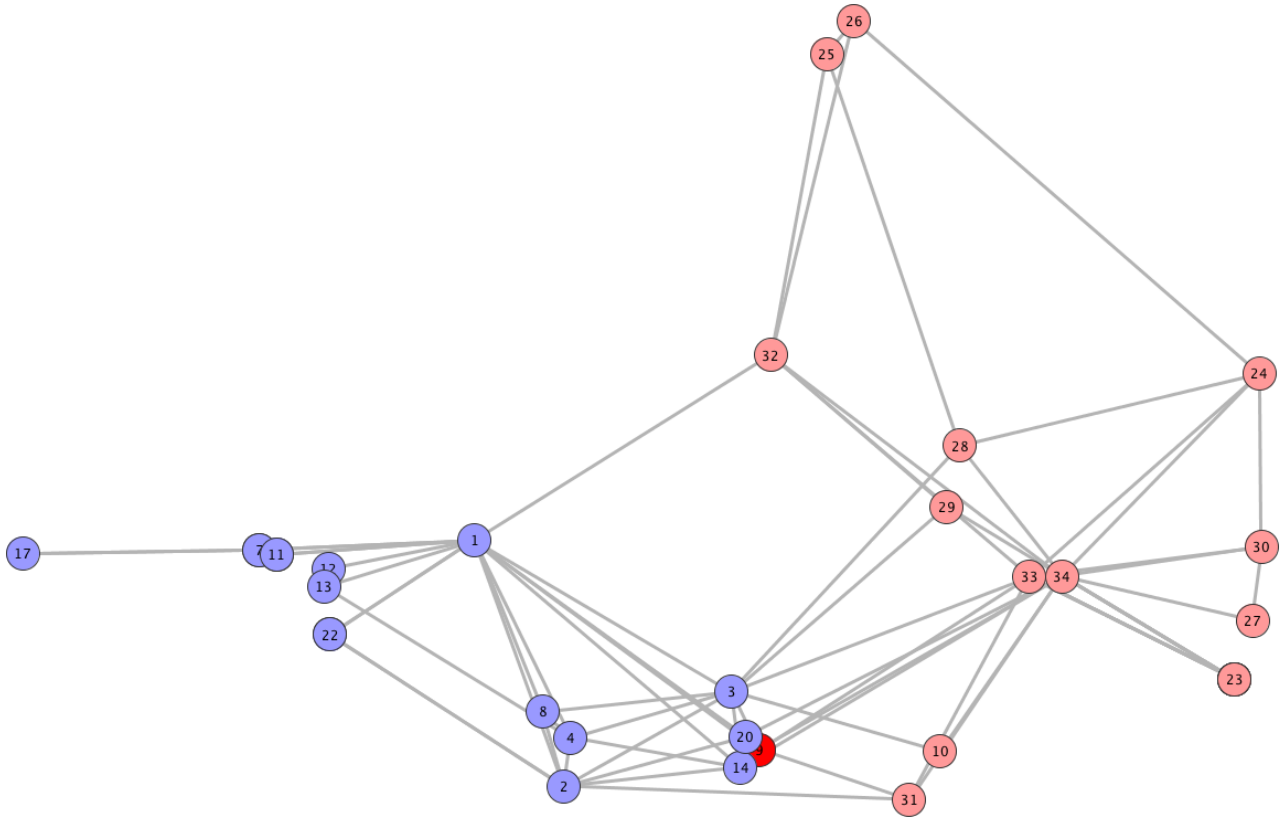
Can you predict the *partitioning*?



Classical MDS of the Karate club network

Karate club

Can you predict the *partitioning*?



Vertex 9 is overlapped - squared effect of Classical MDS

Stress Majorization[12]

Layout algorithmics

- Classical MDS fits squared distances d_{ij}^2 which overemphasizes large distances:

  $strain(X) = \sum_{i,j}(b_{ij} - x_i^T x_j)^2$

- Idea: Postprocessing step to minimize stress

  $stress(X) = \sum_{i,j} w_{ij}(d_{ij} - ||x_i - x_j||)^2$, usually $w_{ij} = \frac{1}{d_{ij}}$

$\rightarrow$ In $O(n^2)$

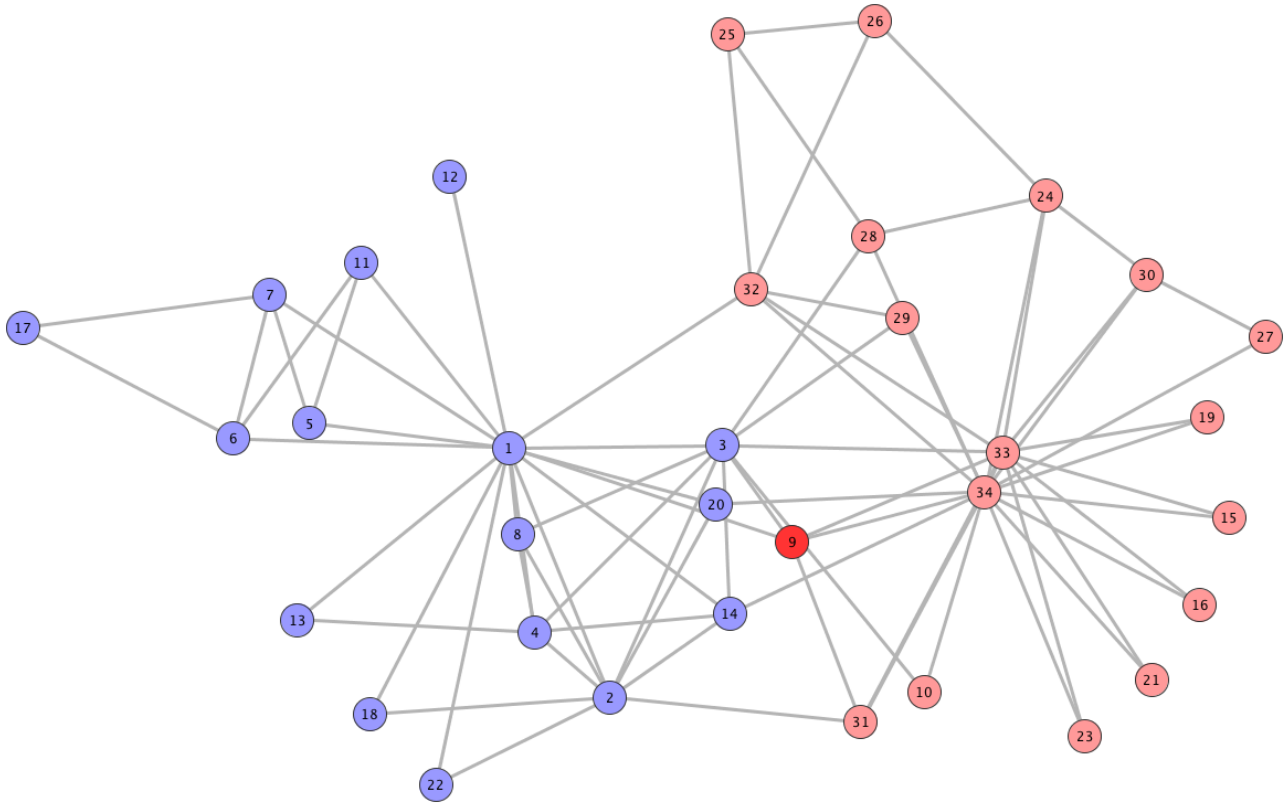
$\rightarrow$ But hope is near[11]
(Graph Drawing 2016, September 19-21 in Athens, Greece)

[11] Ortmann/Klimenta/Brandes: A Sparse Stress Model (2016).

[12] Gansner/Koren/North: Graph drawing by stress majorization (2005).

Karate club

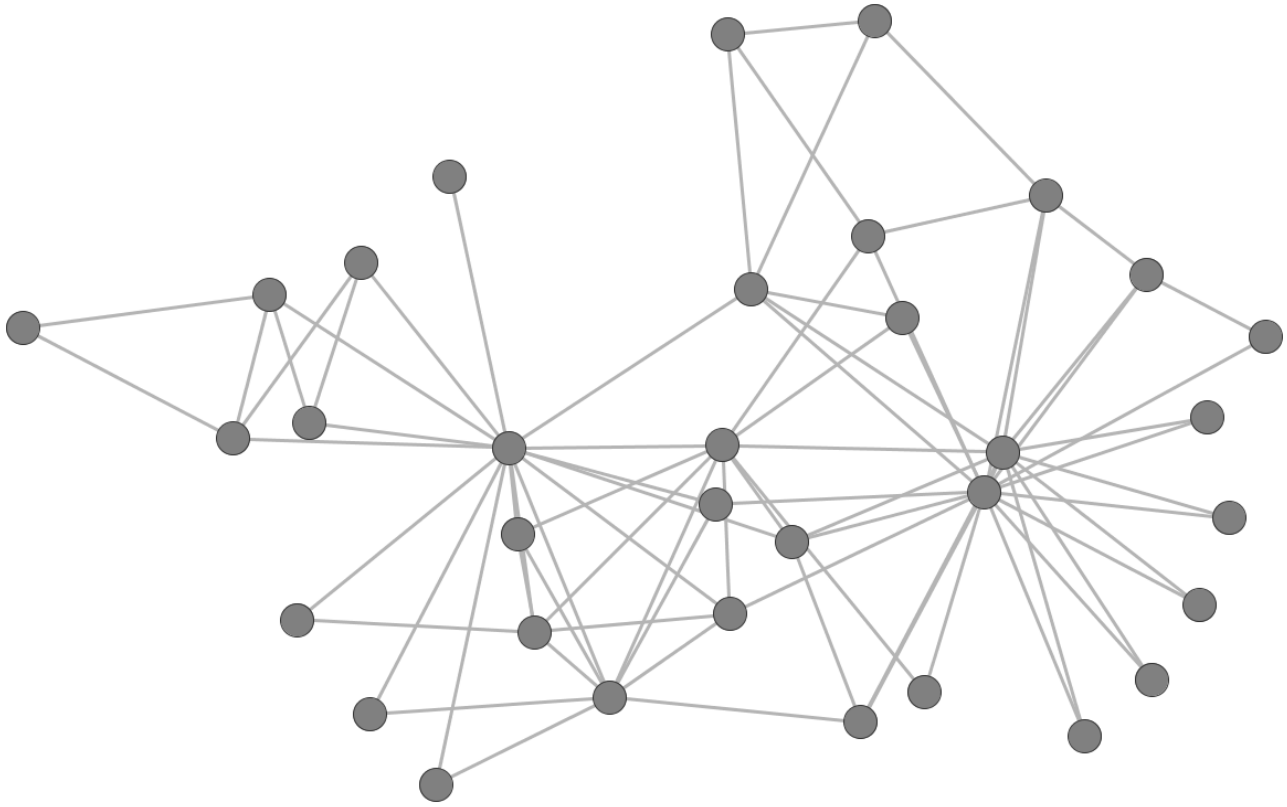
Can you predict the *partitioning*?



Stress layout of the Karate club network

Karate club

Can you predict the *partitioning*?



Stress layout of the Karate club network

Archaeological case study

A Case Study on Network Visualization

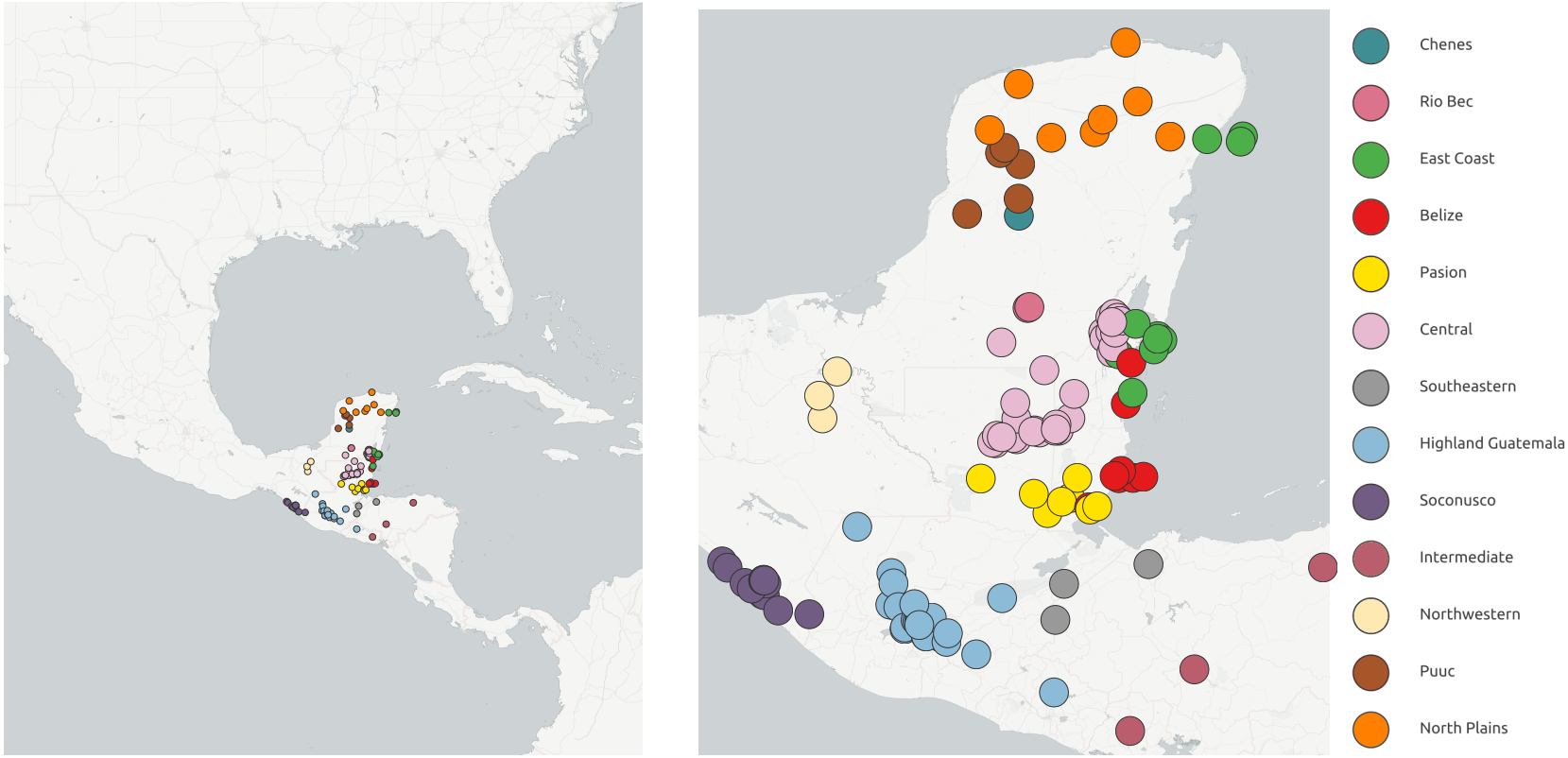
Mark Golitko, James Meierhoff, Gary M. Feinman and Patrick Ryan Williams. *Complexities of collapse: the evidence of Maya obsidian as revealed by social network graphical analysis*. Antiquity, Cambridge University Press (2012).

Maya Obsidian data:

- $\approx$ 120 Maya sites
- 5 sources of obsidian
- 4 temporal periods:
  - Classic (AD 250/300 - 800)
  - Terminal Classic (AD 800 - 1050)
  - Early Postclassic (AD 1050 - 1300)
  - Late Postclassic (AD 1300 - 1520)

Geographical Overview

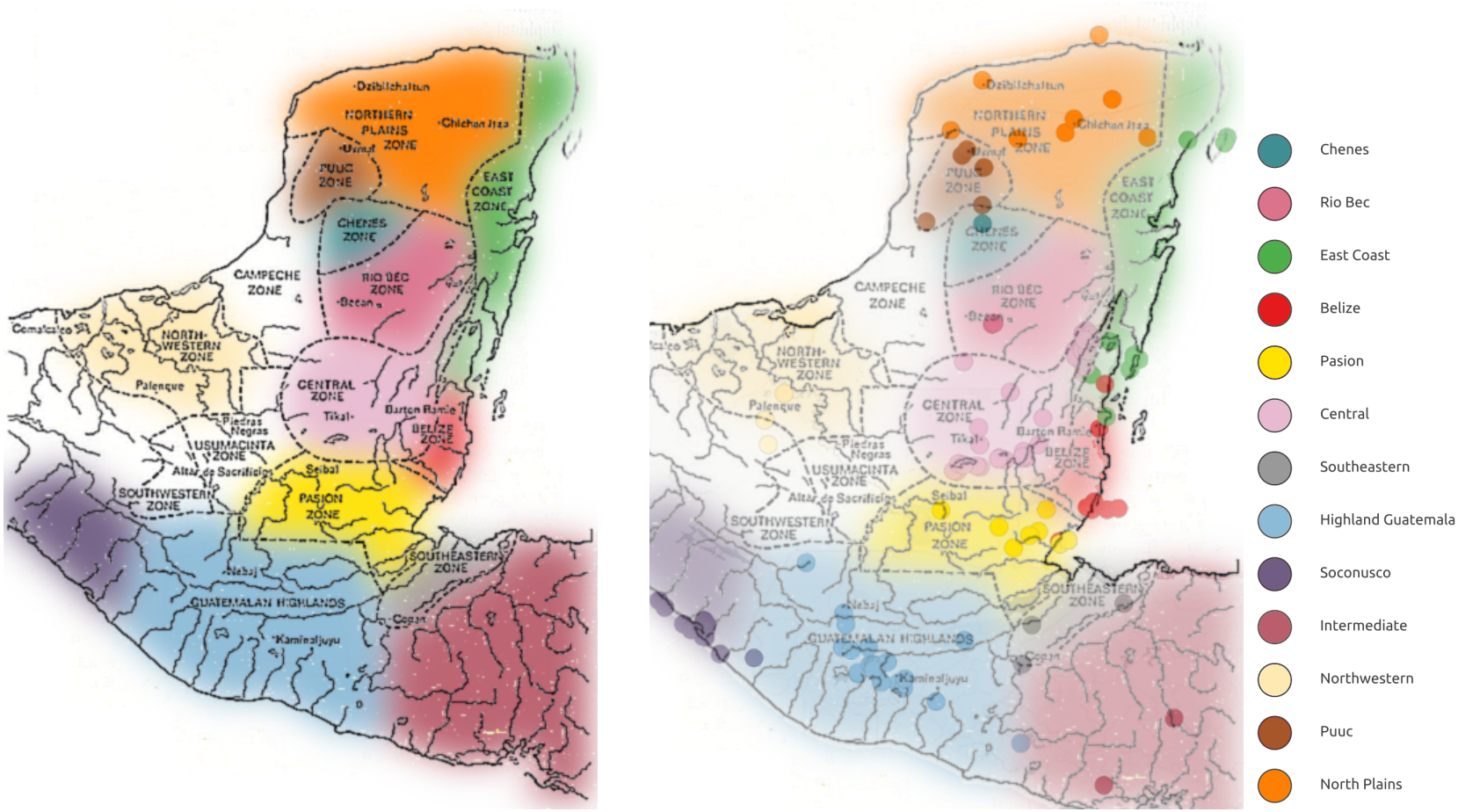
Introduction



Maya sites in Mesoamerica: mountainous / plain / coastal regions

Site Coloring

Introduction

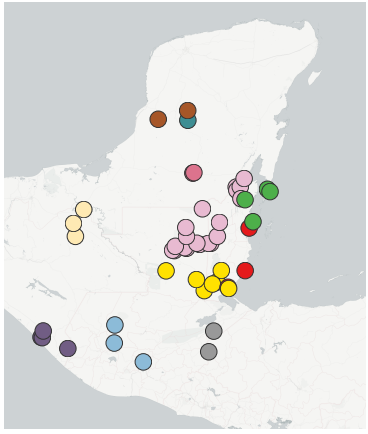


Coloring by Geographic Zones[13]

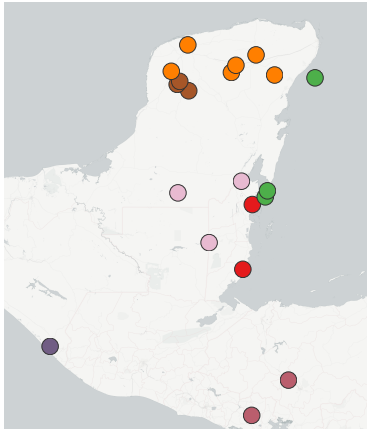
[13] Adams/Culbert: The origins of civilization in the Maya lowlands (1977).

Chronological Overview

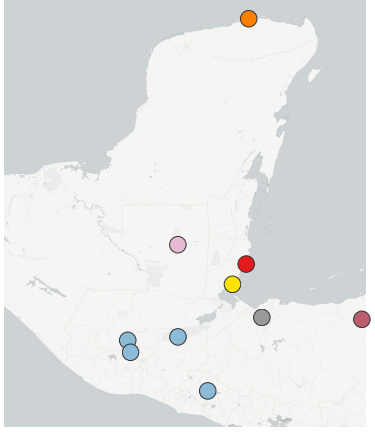
Introduction



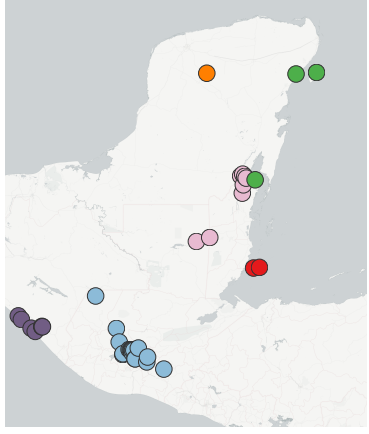
Classic [50 sites]



Early Postclassic [10 sites]



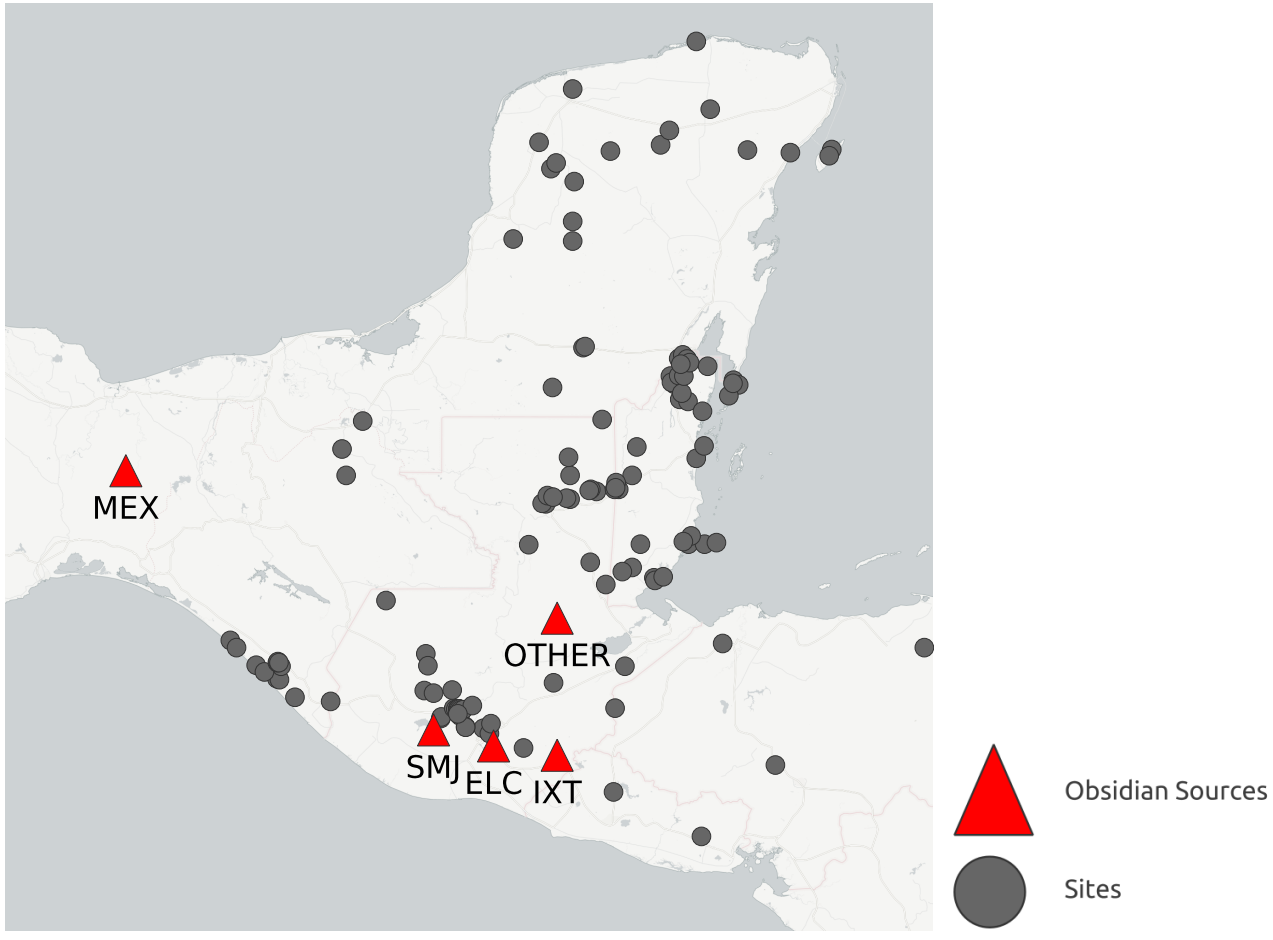
Terminal Classic [20 sites]



Late Postclassic [40 sites]

Obsidian Sources

Introduction



Obsidian Source Location (MEX and OTHER are aggregations)

Generalization of the Data

Introduction

Given

- a set of geographic locations L,
- a set of discrete temporal units T,
- a set of classes of artifacts C,

define *geo-temporal frequencies* as three-dimensional tensor

$$X \in \mathbb{N}^{|L| \times |T| \times |C|} ,$$

i.e. the number $X_{l,t,c}$ of pottery sherds of a certain ware $c \in C$ found on site $l \in L$ for temporal unit $t \in T$.

Network Abstraction

Replication

Choose nodes $\mathcal{N}$ and affiliations $\mathcal{A}$ from tensor dimensions:

$$\mathcal{N}, \mathcal{A} \in \{L, T, C\}$$

e.g. site-site relations:

$$x : L \times L \to \mathcal{W}$$

But other combinations are possible, too!

(Normalized) Brainerd-Robinson Similarity

Replication

Similarity of two sites a and b according to their relative distribution of obsidian sources[14]:

$$\sigma_{BR}(a,b) = 1 - \frac{\sum_{c \in C} |D_{a,c} - D_{b,c}|}{2}$$

with $D \in \mathbb{R}^{|L| \times |C|}$, $D_{l,c} = \frac{X_{l,c}}{||X_{l,\cdot}||_1}$, $||D_{l,\cdot}||_1 = 1$

and $l, a, b \in L, c \in C$

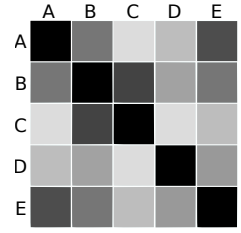
thus $\sigma_{BR} \in [0,1]$

[14] Brainerd: The place of chronological ordering in archaeological analysis (1951); Robinson: A method for chronologically ordering archaeological deposits (1951); Cowgill: Why Pearson's r is not a good similarity coefficient for comparing collections (1990).

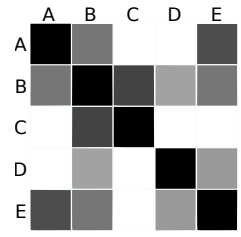
Steps

Replication

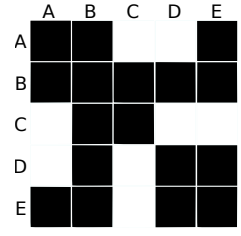
1. **Network Abstraction**
   Edges encode Brainerd-Robinson Similarity between sites.



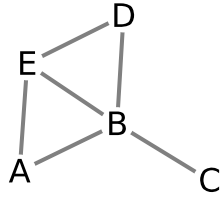
2. **Thresholding**
   Delete least similar edges as long as network stays connected.



3. **Binarization**
   Set similarity of all remaining edges to 1.

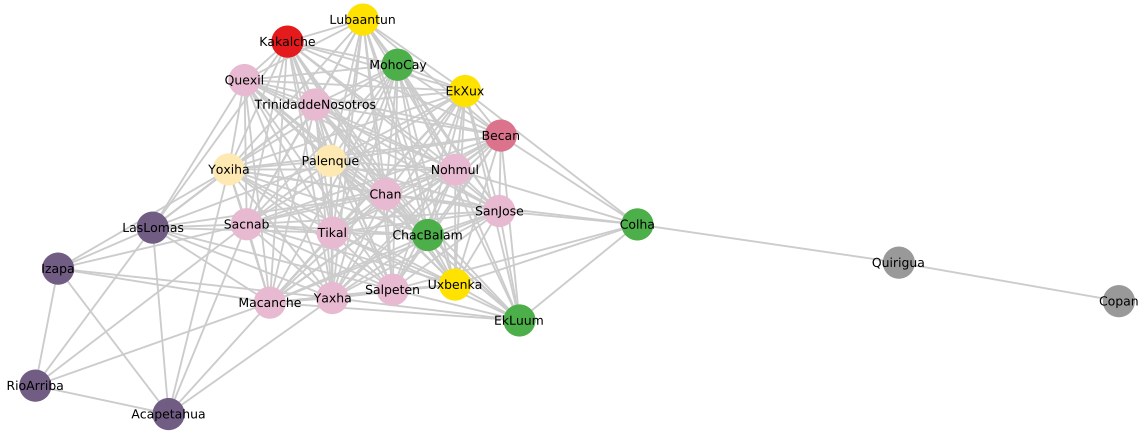


4. **Layout**
   Determine node positions by Spring embedder.

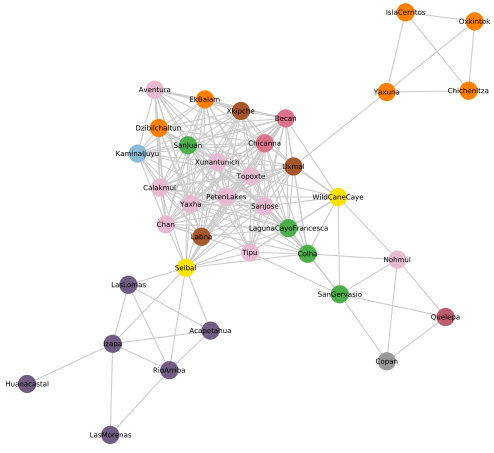


Results

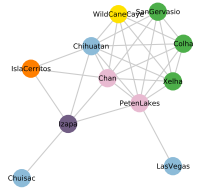
Replication



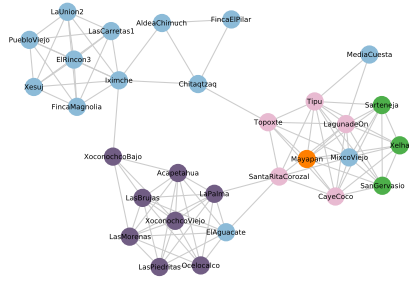
Classic Period



Terminal Classic



Early Postclassic



Late Postclassic

Layout for Valued Networks

Refining the method

1. **Network Abstraction**
   Edges encode Brainerd-Robinson Similarity between sites.
2. **Thresholding**
   Delete least similar edges as long as network stays connected.
3. **Binarization**
   Set similarity of all remaining edges to 1.
4. **Layout**
   Determine node positions by Spring embedder.

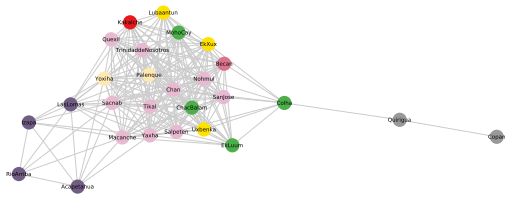
Layout for valued networks

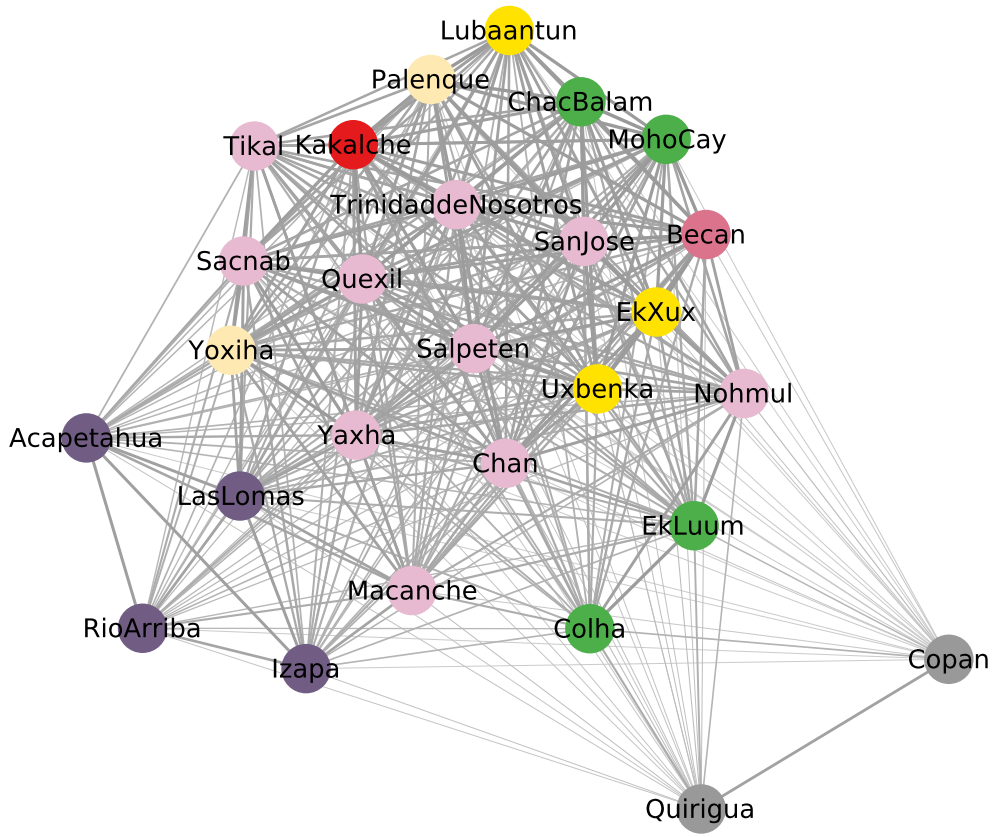
Refining the method

1. **Network Abstraction**
   Edges encode Brainerd-Robinson Similarity between sites.
× **Thresholding**
   Delete least similar edges as long as network stays connected.
× **Binarization**
   Set similarity of all remaining edges to 1.
2. **Layout for valued networks**
   Determine node positions by Multidimensional Scaling (MDS).

Classic Period

Refining the method

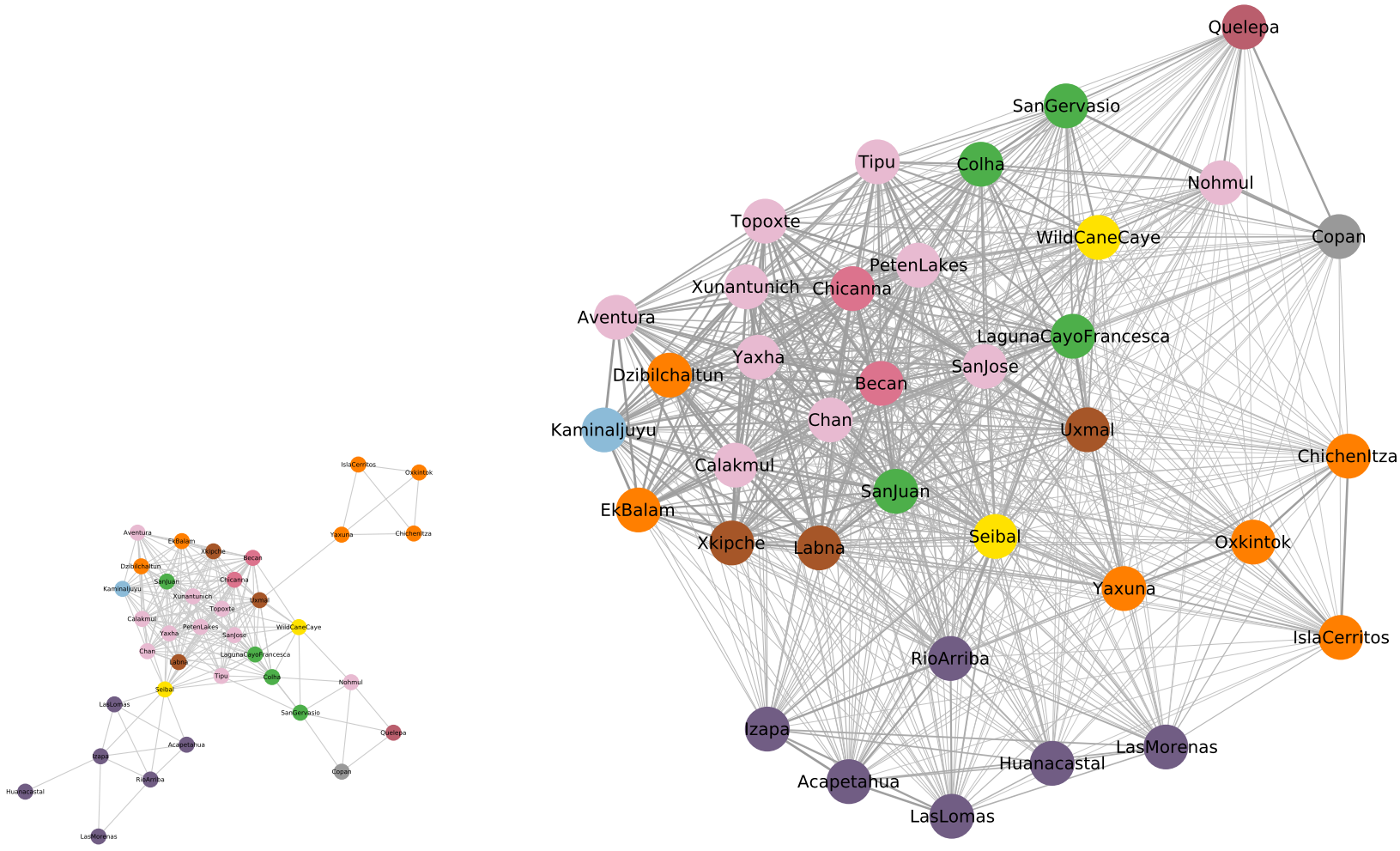




Copan is not only connected through Quirigua.

Terminal Classic Period

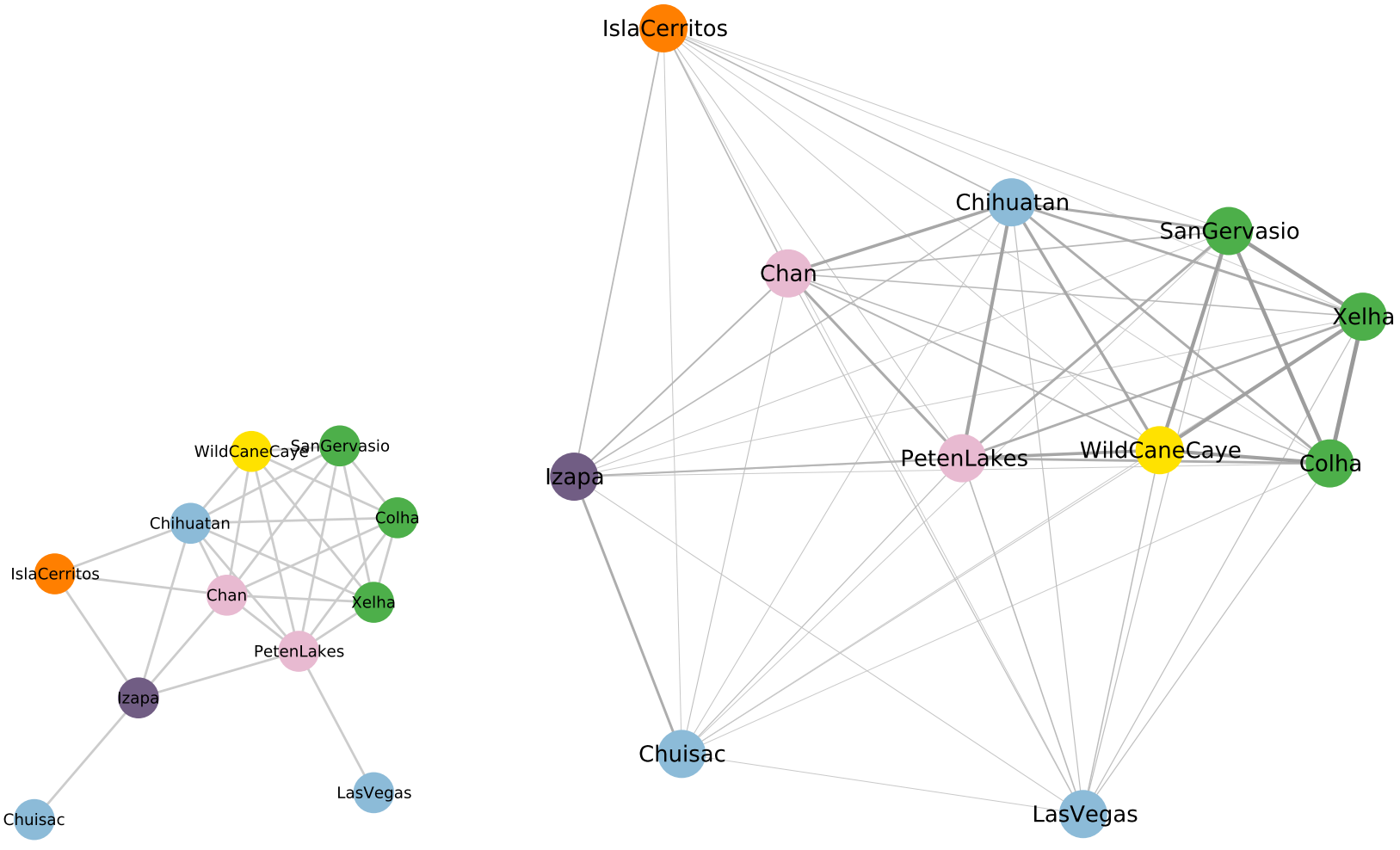
Refining the method



Seibal (yellow) becomes *broker* for orange cluster, too.

Early Postclassic Period

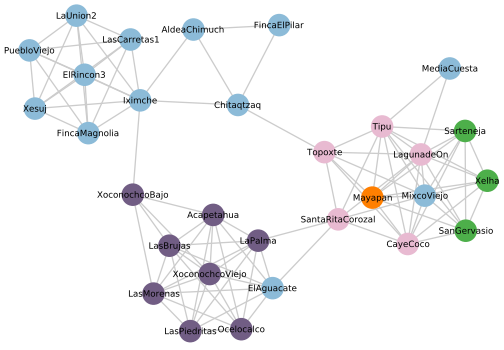
Refining the method

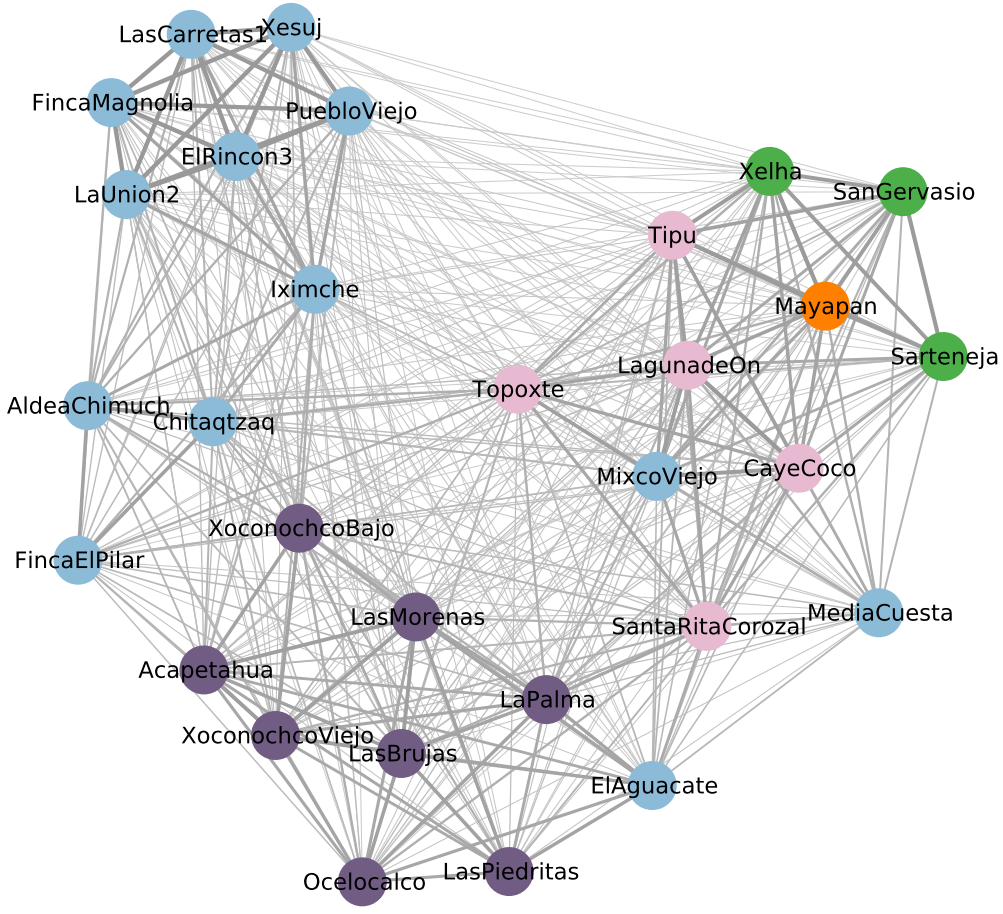


Overall distances to green sites increase.

Late Postclassic Period

Refining the method

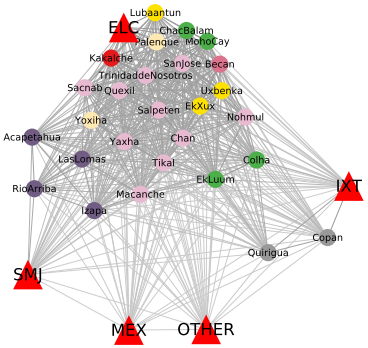




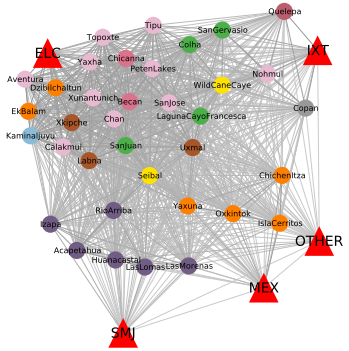
MediaCuesta is dissimilar from the blue cluster.

Including Sources as Reference Points[15]

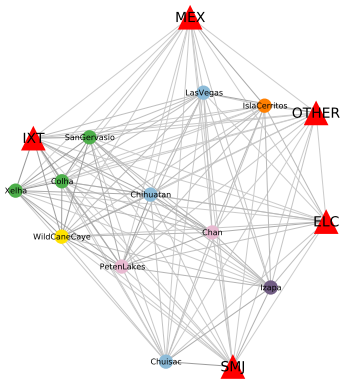
Extending the method



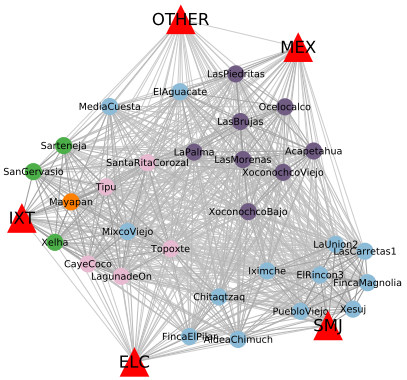
Classic



Terminal Classic



Early Postclassic



Late Postclassic

[15] Weidele et al.: On graphical representations of similarity in geo-temporal frequency data (2016).

Homework Assignment:

- Implement random layout *10 pts.*
- Implement node-local overlap metric *10 pts.*

Contact:

Daniel Weidele
dkweidel@us.ibm.com