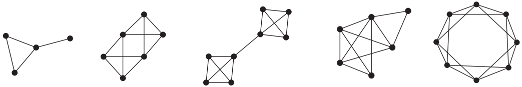


Clique Relaxations in Networks: Theory, Algorithms, and Applications

Sergiy Butenko (butenko@tamu.edu)

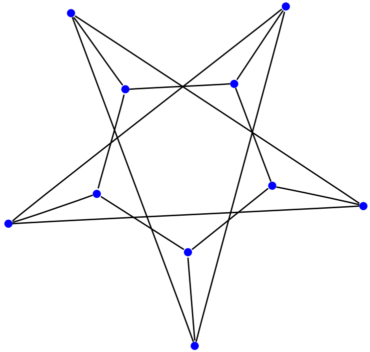
Department of Industrial and Systems Engineering
Texas A&M University
College Station, TX 77843-3131

Outline

Outline

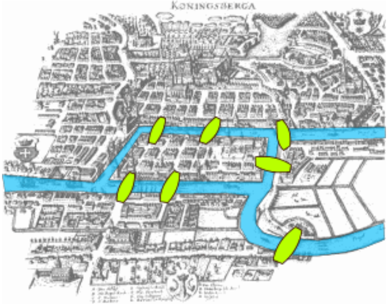
Graphs

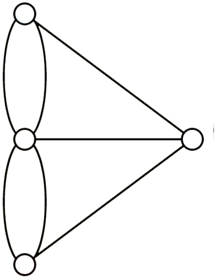
Graph is the mathematical term for a "network" - often visualized as vertices (points, nodes) connected by edges (lines, arcs)



Graph theory basics

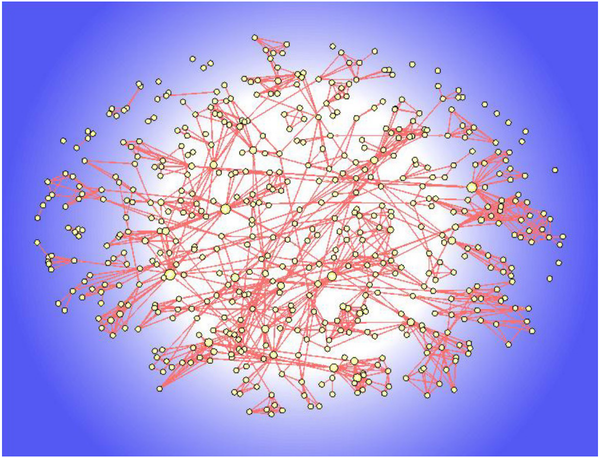
The origins of graph theory are attributed to the Seven Bridges of Königsberg problem solved by Leonhard Euler in 1735.





Social networks

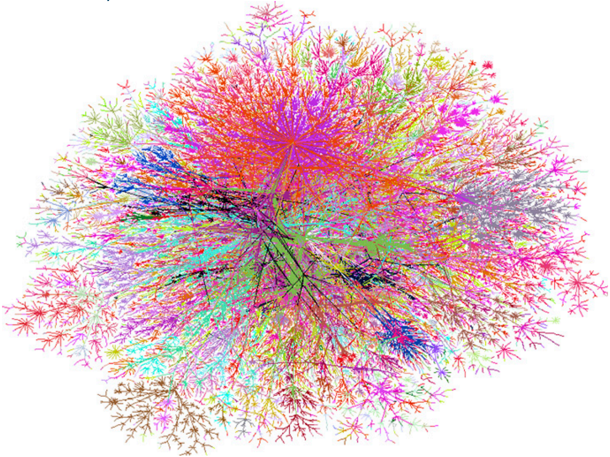
Science - co-authorship network



http://www.jeffkennedyassociates.com:16080/connections/concept/image.html

Telecommunication networks

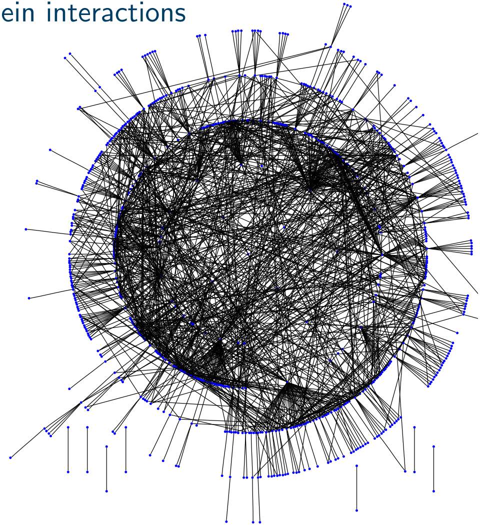
Internet colored by IP addresses



http://www.jeffkennedyassociates.com:16080/connections/concept/image.html

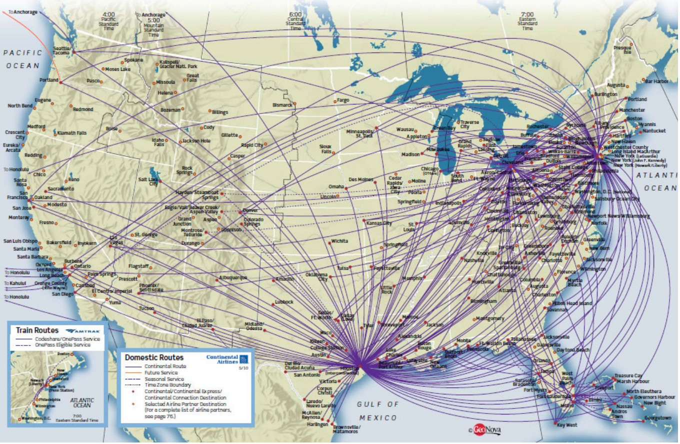
Biological networks

H. Pylori - protein interactions



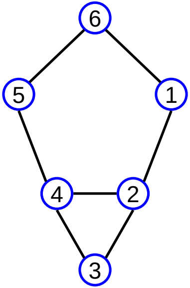
Transportation networks

Continental Airlines network, 2005



Graph theory basics

A *simple, undirected graph* is a pair $G = (V, E)$, where V is a finite set of vertices and $E \subseteq V \times V$ is a set of edges, with each edge defined on a pair of vertices.



$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(1, 2), (1, 6), (2, 3), (2, 4), (3, 4), (4, 5), (5, 6)\}$$

Graph theory basics

- If $(v, v') \in E$, the two vertices v and v' in G are called *adjacent* or *neighbors*, and the edge (v, v') is said to be *incident* to v and v'.
- The set of all neighbors of a vertex v in G is denoted by $N_G(v)$, and its cardinality $|N_G(v)|$ is called the degree of v in G and is denoted by $deg_G(v)$.
- The minimum and the maximum degree of a vertex in G are denoted by $\delta(G)$ and $\Delta(G)$, respectively.

Graph theory basics

- A *path* of length r between vertices v and v' in G is a subgraph of G defined by an alternating sequence of distinct vertices and edges $v \equiv v_0, e_0, v_1, e_1, \ldots, v_{r-1}, e_{r-1}, v_r \equiv v'$ such that $e_i = (v_i, v_{i+1}) \in E$ for all $1 \leq i \leq r-1$.
- Two vertices v and v' are *connected* in G if G contains at least one path between v and v'.
- A graph is *connected* if all its vertices are pairwise connected and *disconnected* otherwise.

Graph theory basics

- The *distance* between two connected vertices v and v' in G, denoted by $d_G(v, v')$, is the shortest length of a path between u and ν in G.
- The largest distance among the pairs of vertices in G defines the diameter of the graph, $diam(G) = \max_{v,v' \in V} d_G(v, v')$.
- The *connectivity* or *vertex connectivity* $\kappa(G)$ of G is given by the minimum number of vertices whose deletion yields a disconnected or a trivial graph.
- The *density* $\rho(G)$ of G is the ratio of the number of edges to the total number of possible edges, i.e., $\rho(G) = |E|/\binom{|V|}{2}$.

Graph theory basics

- A graph $G' = (V', E')$ is a *subgraph* of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$.
- Given a subset of vertices $S \subseteq V$, the *subgraph induced by* S, $G[S]$, is obtained by deleting all vertices in $V \setminus S$ and the edges incident to at least one of them.
- $\overline{G} = (V, \overline{E})$, is the *complement graph* of $G = (V, E)$, where $\overline{E} = \{(i, j) \mid i, j \in V,\ i \neq j \text{ and } (i, j) \notin E\}$.

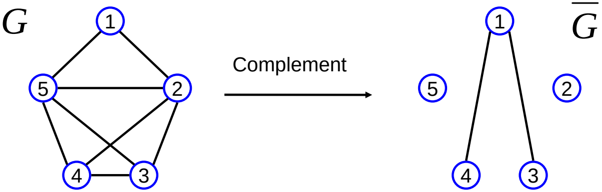
Cliques and independent sets

- A subset of vertices $C \subseteq V$ is called a clique if $G(C)$ is a complete graph.
- A subset $I \subseteq V$ is called an independent set (stable set, vertex packing) if $G(I)$ has no edges.
- C is a clique in G if and only if C is an independent set in $\bar{G}$.

Cliques and independent sets

- A clique (independent set) is said to be
  - maximal, if it is not a subset of any larger clique (independent set);
  - maximum, if there is no larger clique (independent set) in the graph.
- $\omega(G)$ – the clique number of G.
- $\alpha(G)$ – the independence (stability) number of G.

Cliques and independent sets



{1,2,5} : maximal clique

{1,4} : maximal independent set

{2,3,4,5} : maximum clique

{1,2,5} : maximal independent set

{1,4} : maximal clique

{2,3,4,5} : maximum independent set

Dominating sets

- A subset $D \subseteq V$ is called a dominating set if any vertex in V either belongs to D or has a neighbor in D.
- A dominating set is said to be minimum if there is no smaller dominating set in the graph.
- A minimum dominating set size is the domination number of G and is denoted by $\gamma(G)$.

Social networks

A social network is described by $G = (V, E)$ where V is the set of "actors" and E is the set of "ties".

- actors are people and a tie exists if two people know each other.
- actors are wire transfer database records and a tie exists if two records have the same *matching field*.
- actors are telephone numbers and a tie exists if calls were made between them.

Social network analysis

“Popular” social networks:

- Kevin Bacon Number
- Erdös Number
- Six degrees of separation and small world phenomenon in *acquaintance networks*

Cohesive subgroups

- *Cohesive subgroups* are "tightly knit groups" in a social network.
- *Social cohesion* is often used to explain and develop sociological theories.
- Members of a cohesive subgroup are believed to share information, have homogeneity of thought, identity, beliefs, behavior, even food habits and illnesses.

Applications

- **Acquaintance Networks** - criminal network analysis
- **Wire Transfer Database Networks** - detecting money laundering
- **Call Networks** - organized crime detection
- **Protein Interaction Networks** - predicting protein functions
- **Gene Co-expression Networks** - detecting network motifs
- **Stock Market Networks** - stock portfolios
- **Internet Graphs** - information search and retrieval
- **Wireless and telecommunication networks** - clustering and routing

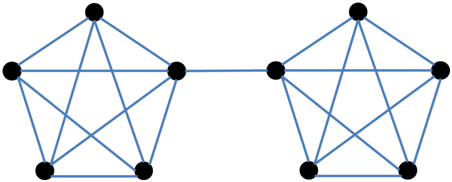
Properties of cohesive subgroups

Some desirable properties of a cohesive subgroup are:

- Familiarity (degree);
- Reachability (distance, diameter);
- Robustness (connectivity);
- Density (edge density).

Clique

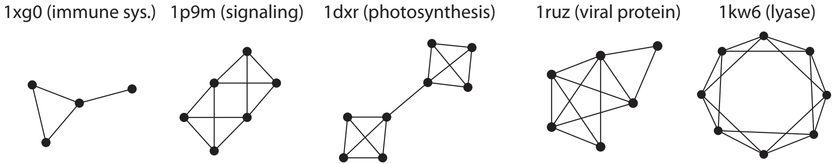
Clique is the earliest model of a cohesive subgroup.

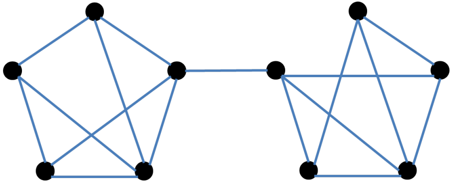


The "perfect cluster"

However ...

Perfect may mean impractical. Some examples:





Alternatives to clique

$G = (V, E)$. $S \subseteq V$ is

- s-**clique** if $d_G(v, v') \leq s$, for any $v, v' \in S$ (Luce 1950)
- s-**club** if $diam(G[S]) \leq s$ (Alba 1973, Mokken 1979)
- s-**plex** if $\delta(G[S]) \geq |S| - s$ (Seidman & Foster 1978)
- s-**defective clique** if $G[S]$ has at least $\binom{|S|}{2} - s$ edges (Yu et al. 2006)
- k-**core** if $\delta(G[S]) \geq k$ (Seidman 1983)
- k-**block** if $\kappa(G[S]) \geq k$ (Moody & White 2003)
- γ-**quasi-clique** if $\rho(G[S]) \geq \gamma$ (Abello et al. 2002)
- (λ, γ)-**quasi-clique** if $\delta(G[S]) \geq \lambda(|S| - 1)$ and $\rho(G[S]) \geq \gamma$ (Brunato et al. 2008)
- ...

Alternative clique definitions

(a) Vertices are **distance one** away from each other

(b) Vertices induce a subgraph of **diameter one**

(c) Every **one** vertex forms a **dominating set**

Alternative clique definitions

(d) **Degree:** Each vertex neighbors **all** vertices

(e) **Density:** Vertices induce a subgraph that has **all** possible edges

(f) **Connectivity:** need to be remove **all** vertices to obtain a disconnected induced subgraph

Defining clique relaxations

We can define clique relaxations by

(i) **restricting** a **violation** of an elementary clique-defining property or by

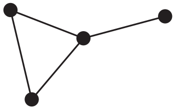
(ii) **ensuring** the **presence** of an *elementary clique-defining property*

(i) Restricting a violation

Pairs of vertices are **distance at most** s away from each other – s-**clique**

Induced subgraph is of **diameter at most** s – s-**club**

1xg0 (immune sys.)

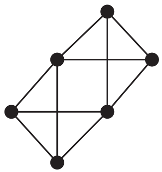


2-club

(i) Restricting a violation

Any set of size s ensures **domination** – ***s*-plex**

1p9m (signaling)



3-plex

(i) Restricting a violation

We replaced **one** with **at most** s in the alternative clique definitions

We can also replace **all** with **all but** s

(i) Restricting a violation

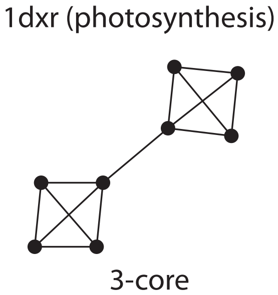
Degree: Each vertex neighbors **all but** s vertices – s-**plex** again

Density: Vertices induce a subgraph that has **all but** s possible edges – s-**defective clique**

Connectivity: need to be remove **all but** s vertices to obtain a disconnected induced subgraph – s-**bundle**

(ii) Ensuring a property

Each vertex has **degree at least** k – k-**core**

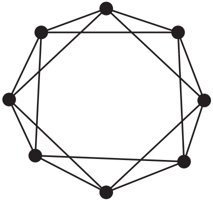


(ii) Ensuring a property

At least k vertices need to be removed

to disconnect the induced subgraph – k-**block**

1kw6 (lyase)

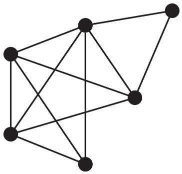


4-connected

Relative relaxations

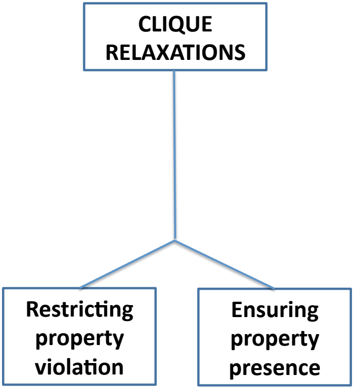
Vertices induce a subgraph that has **the fraction** γ of all possible edges – γ-**quasi-clique**

1ruz (viral protein)

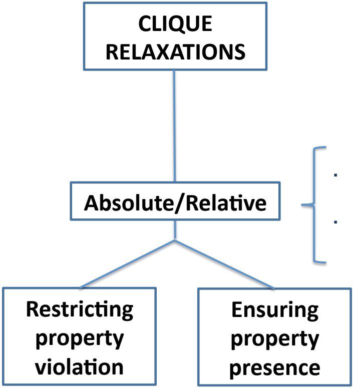


.7-quasiclique

Nature of a clique relaxation

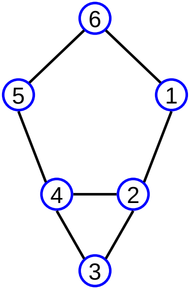


Nature of a clique relaxation

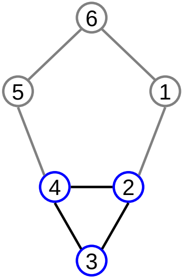


- Absolute (s,k)
  size independent
- Relative $\alpha|S|$
  size dependent

s-clique vs s-club

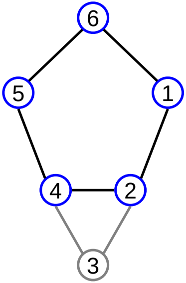


s-clique vs s-club



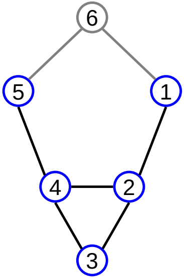
- $\{\mathbf{2,3,4}\}$ is a 1-club ... the "regular" clique

s-clique vs s-club



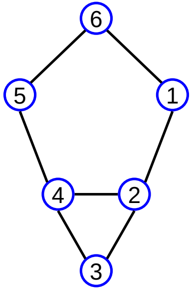
- $\{2,3,4\}$ is a 1-club … the "regular" clique
- $\{\mathbf{1,2,4,5,6}\}$ is a 2-club

s-clique vs s-club



- {2,3,4} is a 1-club ... the "regular" clique
- {1,2,4,5,6} is a 2-club
- {**1,2,3,4,5**} is a 2-clique but NOT a 2-club

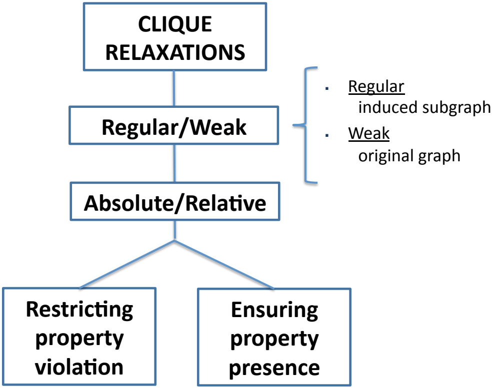
s-clique vs s-club



- $\{2,3,4\}$ is a 1-club ... the "regular" clique
- $\{1,2,4,5,6\}$ is a 2-club
- $\{1,2,3,4,5\}$ is a 2-clique but NOT a 2-club
- **maximality** of a 2-club is harder to test

s-clique appears to be a **weaker** cluster than s-club

Nature of a clique relaxation



Weak clique relaxations

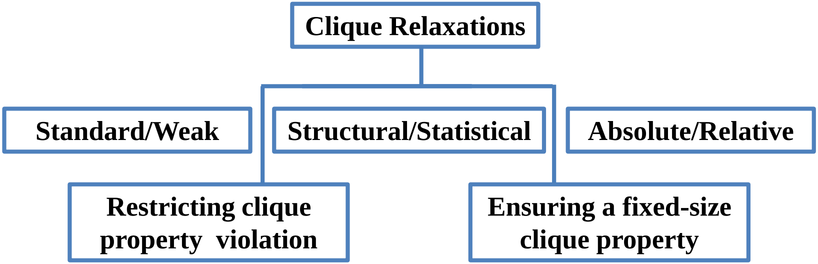
Distance-based: s-**clique (weak** s-**club)**

Vertices in S are **distance at most** s away from each other **in** G.

Connectivity-based: **weak** k-**block**

Any two vertices in **S** have **at least** k vertex-independent **paths** between them **in** G

Clique relaxations taxonomy



Order of a clique relaxation

It may be useful to relax **more than one** elementary clique-defining property

- **Clique** is the only clique relaxation of order 0
- Clique relaxations of **first order** relax **one** of the elementary properties (distance, diameter, ...) used to define clique
- Clique relaxations of **second order** relax **two** of the elementary clique-defining properties
- ...

Higher order clique relaxations

Simple Higher Order Relaxations: relaxing multiple elementary clique-defining properties simultaneously

(λ, γ)-*quasiclique*: Each vertex is connected to *at least* $\lambda(|S| - 1)$ vertices, and the induced subgraph has at least *the fraction* γ of all possible edges.

Robust Higher Order Relaxations: connectivity *embedded* into the definition (k-robustness/k-heredity)

k-robust s-club: The induced subgraph is not only an s-club, but also the removal of up to k vertices still preserves the s-club property.

Additional elementary clique-defining properties

A subset of vertices C is a clique in G if and only if one of the following conditions hold:

g) Independence number $\alpha(G[C]) = 1$;

h) Vertex cover number $\tau(G[C]) = |C| - 1$;

i) Chromatic number $\chi(G[C]) = |C|$;

j) Clique cover number $\bar{\chi}(G[C]) = 1$;

k) Edge connectivity number $\lambda(G[C]) = |C| - 1$.

Canonical clique relaxations

	Reachability	Familiarity		Composition	Robustness
	Diameter	Domination[1]	Density[2]	Degree[3]	Connectivity[4]
Clique	"one"	"one"	"all"	"all"	"all"
s-club	"at most s"				
s-plex		"s "			
γ-quasiclique			"at least γ"		
k-core				"at least k"	
k-connected					"at least k"

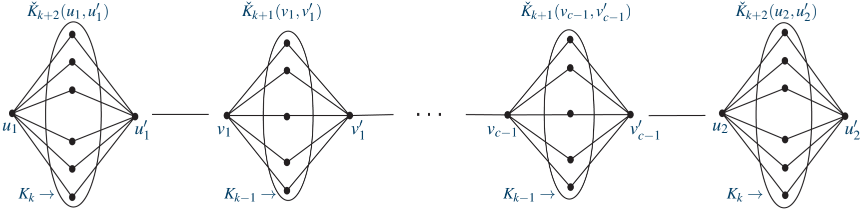
[1] *All vertices are dominated by*, [2] *Edges included*, [3] *Every vertex is connected to*, [4] *To disconnect, remove*

Diameter of k-cores

Let S be a k-core in G. If $G[S]$ is connected then $diam(G[S]) \leq d'_k$, where

$$d'_k = \max\left\{ \left\lceil \frac{|S|}{k+1} \right\rceil, 3\left(\left\lfloor \frac{|S|-z}{k+1} \right\rfloor - 1 \right) + z, z \in \{0,1,2\} \right\}$$

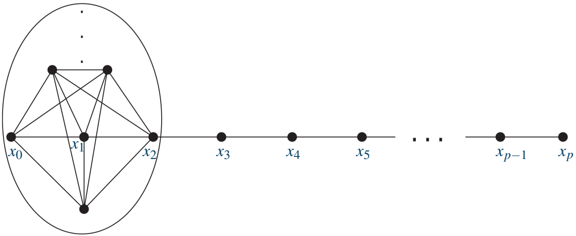
This bound is sharp



Diameter of γ-quasi-cliques

Let S be a γ-quasi-clique in G. If $G[S]$ is connected, then $diam(G[S]) \leq d_\gamma$, where

$$d_\gamma = \left\lfloor |S| + \frac{1}{2} - \sqrt{\gamma |S|^2 - (2+\gamma)|S| + \frac{17}{4}} \right\rfloor.$$



K_q without edge (x_0, x_2)

Cohesiveness properties

$S \subseteq V$	Diameter	Dominating Set	Minimum Degree	Edge Density	Connectivity
Clique	"one"	"one"	"all"	"one"	"all"
s-club	s	$\lvert S\rvert - 1$	1	$\frac{2}{\lvert S\rvert}$	1
s-plex	s	s	$\lvert S\rvert - s$	$1 - \frac{s-1}{\lvert S\rvert-1}$	$\lvert S\rvert - 2s + 2$
k-core	d'_k	$\lvert S\rvert - k$	k	$\frac{k}{\lvert S\rvert-1}$	$2k + 2 - \lvert S\rvert$
γ-quasi-clique	d_γ	$\lvert S\rvert$	$\left\lceil \gamma\binom{\lvert S\rvert}{2} - \binom{\lvert S\rvert-1}{2} \right\rceil$	γ	$\left\lceil \gamma\binom{\lvert S\rvert}{2} - \binom{\lvert S\rvert-1}{2} \right\rceil$
k-block	$\left\lfloor \frac{\lvert S\rvert-2}{k} + 1 \right\rfloor$	$\lvert S\rvert - k$	k	$\frac{k}{\lvert S\rvert-1}$	k

Optimization problems

Let RELAXED CLIQUE refer to a subset of vertices that satisfies the definition of an arbitrary clique relaxation concept.

Definition

A subset of vertices S is called a maximal RELAXED CLIQUE if it is a RELAXED CLIQUE and is not a proper subset of a larger RELAXED CLIQUE.

Definition

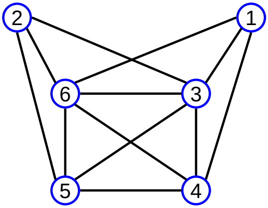
A subset of vertices S is called a maximum RELAXED CLIQUE if there is no larger RELAXED CLIQUE in the graph. The maximum RELAXED CLIQUE problem asks to compute a maximum RELAXED CLIQUE in the graph, and the size of a maximum RELAXED CLIQUE is called the RELAXED CLIQUE number.

Clique relaxations: s-plex

Definition

A subset of vertices S is said to be a s-plex if the minimum degree in the induced subgraph $\delta(G[S]) \geq |S| - s$

i.e. every vertex in $G[\mathbf{S}]$ has degree at least $|\mathbf{S}| - s$.

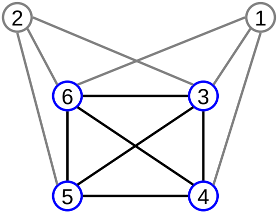


Clique relaxations: s-plex

Definition

A subset of vertices S is said to be a s-plex if the minimum degree in the induced subgraph $\delta(G[S]) \geq |S| - s$

i.e. every vertex in $G[\mathbf{S}]$ has degree at least $|\mathbf{S}| - s$.



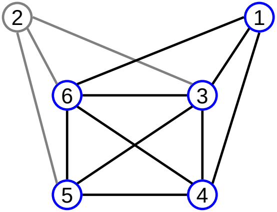
- $\{\mathbf{3,4,5,6}\}$ is a 1-plex ... the "regular" clique

Clique relaxations: s-plex

Definition

A subset of vertices S is said to be a s-plex if the minimum degree in the induced subgraph $\delta(G[S]) \geq |S| - s$

i.e. every vertex in $G[\mathbf{S}]$ has degree at least $|\mathbf{S}| - s$.



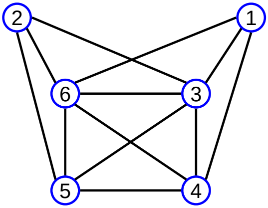
- $\{3,4,5,6\}$ is a 1-plex ... the "regular" clique
- $\{\mathbf{1,3,4,5,6}\}$ is a 2-plex (and NOT a 1-plex)

Clique relaxations: s-plex

Definition

A subset of vertices S is said to be a s-plex if the minimum degree in the induced subgraph $\delta(G[S]) \geq |S| - s$

i.e. every vertex in $G[\mathbf{S}]$ has degree at least $|\mathbf{S}| - s$.



- $\{3,4,5,6\}$ is a 1-plex ... the "regular" clique
- $\{1,3,4,5,6\}$ is a 2-plex (and NOT a 1-plex)
- $\{\mathbf{1,2,3,4,5,6}\}$ is a 3-plex (and NOT a 2-plex)

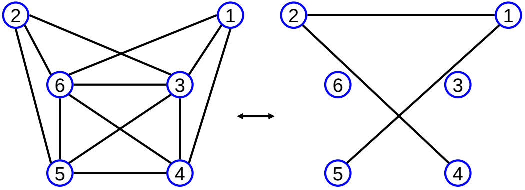
Complementary structure: co-s-plex

Definition

A subset of vertices S is a co-s-plex if the maximum degree in the induced subgraph $\Delta(G[S]) \leq s-1$.

i.e. degree of every vertex in $G[S]$ is at most $s-1$.

S is a co-s-plex in G *if and only if* S is an s-plex in the complement graph $\bar{G}$.



3-plex Co-3-plex

Structural Properties of an s-Plex

If G is an s-plex then

1. Every subgraph of G is a s-plex;
2. If $s < \frac{n+2}{2}$ then $diam(G) \leq 2$;
3. $\kappa(G) \geq n - 2s + 2$.
4. Any s vertices in G form a *dominating set* in G.

- s-plexes for "small" s values, guarantee reachability and connectivity while relaxing familiarity.

Structiural properties

Definition (Heredity)

A graph property Π is said to be *hereditary on induced subgraphs*, if for any graph G with property Π the deletion of any subset of vertices does not produce a graph violating Π.

Definition (Weak heredity)

A graph property Π is said to be *weakly hereditary*, if for any graph $G = (V, E)$ with property Π all subsets of V demonstrate the property Π in G.

Structiural properties

Definition (Quasi-heredity)

A graph property Π is said to be *quasi-hereditary*, if for any graph $G = (V, E)$ with property Π and for any size $0 \leq r < |V|$, there exists some subset $R \subset S$ with $|R| = r$, such that $G[S \setminus R]$ demonstrates property Π.

Definition (k-Heredity)

A graph property Π is said to be *k-hereditary on induced subgraphs*, if for any graph G with property Π the deletion of any subset of vertices with up to k vertices does not produce a graph violating Π.

Yannakakis theorem

- The *maximum* Π *problem* is to find the largest order induced subgraph that does not violate property Π
- Π is said to be *nontrivial* if it is true for a single vertex graph and is not satisfied by every graph
- Π is said to be *interesting* if there are arbitrarily large graphs satisfying Π

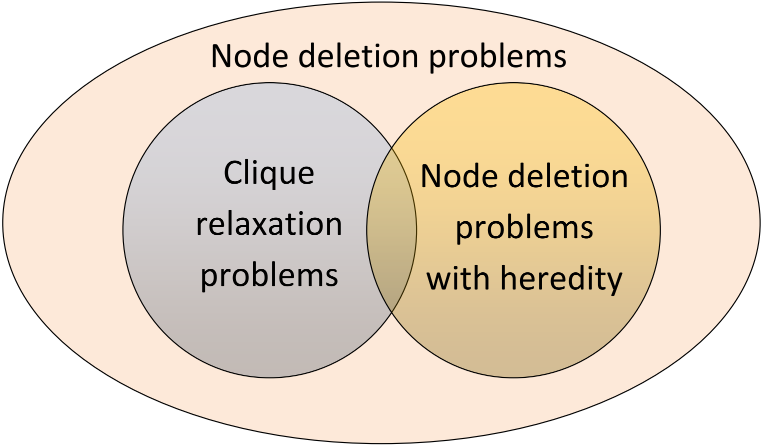
Theorem (Yannakakis, 1978)

The maximum Π *problem for nontrivial, interesting graph properties that are hereditary on induced subgraphs is NP-hard.*

Outline

Outline

Clique relaxation and node deletion problems



Max weight node deletion problem

Given

- a simple, undirected graph $G = (V, E)$,
- positive weights $w(v)$ for each $v \in V$,
- and a nontrivial, interesting and hereditary (on induced subgraphs) property Π.

Find

$$\nu(G) = \max\{w(P) : \; P \subseteq V, \; G[P] \text{ satisfies } \Pi\},$$

where $w(P) = \sum_{v \in P} w(v)$ and $G[P]$ is the subgraph induced by P.

Generalizing max clique algorithms

Some of the most practically effective algorithms for the maximum clique problem rely on the fact that clique is hereditary on induced subgraphs.

- Carraghan and Pardalos (1990) - used as DIMACS benchmark
- Östergård (2002)

We use this observation to develop a generalized algorithm for the minimum weight node deletion problem.

Generalized algorithm

- Order vertices $V = \{v_1, v_2, \cdots, v_n\}$, define $S_i = \{v_i, v_{i+1}, \cdots, v_n\}$
- We compute the function $c(i)$ that is the weight of the maximum induced subgraph with property Π in $G[S_i]$.
- Obviously, $c(n) = w(v_n)$ and $c(1) = \nu(G)$.
- For the unweighted case with $w(v_i) = 1, i = 1, \ldots, n$ we have

$$c(i) = \begin{cases} c(i+1) + 1, & \text{if the solution must contain } v_i \\ c(i+1), & \text{otherwise} \end{cases}$$

- For the weighted case, $c(i) > c(i+1)$ implies that v_i belongs to every optimal solution, and $c(i) \leq c(i+1) + w(v_i)$.

Generalized algorithm

- We compute the value of $c(i)$ starting from $c(n)$ and down to $c(1)$, and in each major iteration work with a current feasible solution P, a candidate set C, and an incumbent solution S.
- Pruning occurs when
  - $w(C) + w(P) < w(S)$ or
  - $c(i) + w(P) < w(S)$, where $i = \min\{j : v_j \in C\}$.
- Problem-specific features:
  - candidate set generation;
  - vertex ordering.

Outline

Outline

Edge-based clique relaxations

Given a graph $G = (V, E)$ and a subset of edges E', the *edge-induced subgraph* for E' is given by $G(E') = (V', E')$, where

$$V' = \{v \in V : \exists (u, v) \in E'\}$$

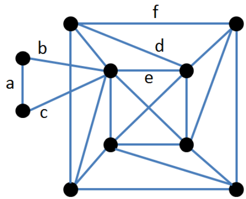
We will call a vertex v a neighbor of an edge $(u, u') \in E$ if v is adjacent to both u and u'

k-Community

Let $E' \subseteq E$ and $G(E') = (V', E')$. Then V' is called a k-community if every edge in E' has at least k neighboring vertices in $G(E')$.

Just like the maximum k-core, the maximum k-community can also be found in polynomial time using a simple iterative procedure.

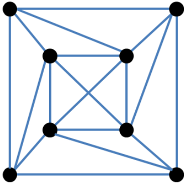
Examples



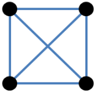
Original Graph

5-Core

3-Comm



4-Core



2-Comm

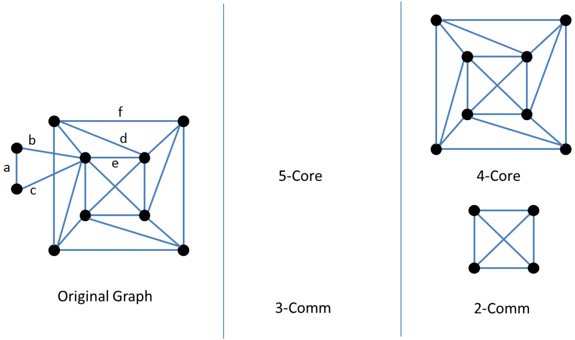
Useful properties

Properties:

1. A clique of size k is both a $(k-1)$-core and $(k-2)$-community.
2. If the k-core of G is empty, then $\omega(G) < k+1$.
3. If the k-community of G is empty, then $\omega(G) < k+2$.
4. A k-community of G is also a $(k+1)$-core of G.

Note that the converse is not true for properties 2 and 3.

Upper bounds



UB suggested by k-core: 5; by k-community: 4

Finding the upper bound

A simple binary search strategy for finding the least k that gives an empty graph:

Step 0 Set $k^u = n-2, k^l = 0, k = n-2$

Step 1 **If** k-Comm(G) is empty, $k^u = k$.
Else, $k^l = k$.

Step 2 **If** $k^u - k^l \leq 1$, set $k = k^u$, STOP.
Else, set $k = (k^u + k^l)/2$, go to Step 1.

Algorithm 1

Worst case complexity of the algorithm: $O(m^2 \Delta log(n))$.

Experiments with SNAP database

Network Type	Example
Social Networks	Epinions.com
	Slashdot
	Wikipedia votes
Communication	EU Research Inst emails
	Wikipedia communications
Citation Networks	US Patents
	Arxiv Citation Network
Web graphs	Stanford
	Google
Product Co-purchasing	Amazon
Internet P2P	Gnutella

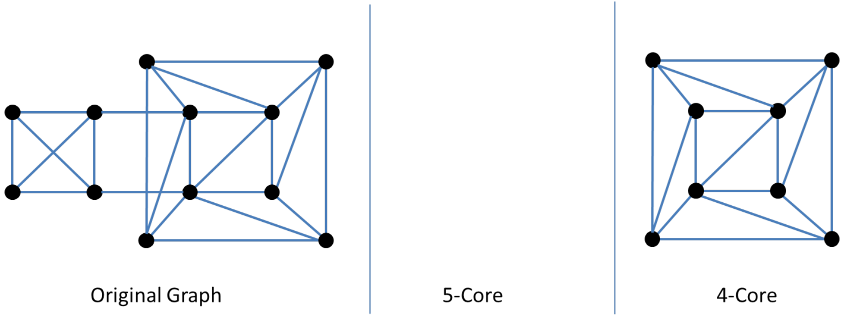
Upper bounds on SNAP graphs

Graph	n	m	k-Core UB		k-Comm UB	
			UB	n'	UB	n"
WikiTalk	2394385	4659565	132	700	52	237
cit-Patents	3774768	16518947	65	106	35	83
Email-EuAll	265214	364481	38	292	19	62
Cit-HepPh	34546	420877	31	40	24	36
Cit-HepTh	27770	352285	38	52	29	48
Slashdot0811	77360	469180	55	129	34	87
Slashdot0902	82168	504230	56	134	35	96
soc-Epinions1	75879	405740	68	486	32	61
Wiki-Vote	7115	100762	54	336	22	50
p2p-Gnutella31	62586	147892	7	1004	4	57
p2p-Gnutella04	10876	39994	8	365	4	12
p2p-Gnutella24	26518	65369	6	7480	4	41
p2p-Gnutella25	22687	54705	6	6091	4	25
p2p-Gnutella30	36682	88328	8	14	4	42
web-Stanford	281903	1992636	72	387	61	126
web-NotreDame	325729	1090108	156	1367	155	155
web-Google	875713	4322051	45	48	44	48
web-BerkStan	685230	6649470	202	392	201	392
Amazon0601	403394	2443408	11	32886	11	5361
Amazon0505	410236	2439437	11	32632	11	4878
Amazon0302	262111	899792	7	286	7	105
Amazon0312	400727	2349869	11	27046	11	4534

Upper bounds on SNAP graphs

Graph	n	m	k-Core UB		k-Comm UB	
			UB	n'	UB	n"
WikiTalk	2394385	4659565	132	700	52	237
cit-Patents	3774768	16518947	65	106	35	83
Email-EuAll	265214	364481	38	292	19	62
Cit-HepPh	34546	420877	31	40	24	36
Cit-HepTh	27770	352285	38	52	29	48
Slashdot0811	77360	469180	55	129	34	87
Slashdot0902	82168	504230	56	134	35	96
soc-Epinions1	75879	405740	68	486	32	61
Wiki-Vote	7115	100762	54	336	22	50
p2p-Gnutella31	62586	147892	7	1004	4	57
p2p-Gnutella04	10876	39994	8	365	4	12
p2p-Gnutella24	26518	65369	6	7480	4	41
p2p-Gnutella25	22687	54705	6	6091	4	25
p2p-Gnutella30	36682	88328	8	14	4	42
web-Stanford	281903	1992636	72	387	61	126
web-NotreDame	325729	1090108	156	1367	155	155
web-Google	875713	4322051	45	48	44	48
web-BerkStan	685230	6649470	202	392	201	392
Amazon0601	403394	2443408	11	32886	11	5361
Amazon0505	410236	2439437	11	32632	11	4878
Amazon0302	262111	899792	7	286	7	105
Amazon0312	400727	2349869	11	27046	11	4534

Finding a maximum clique



Finding a maximum clique in the residual graph is not enough!

Lower and upper bounds

Graph	n	m	LB	UB	Time(sec)
WikiTalk	2394385	4659565	26	52	1395.91
cit-Patents	3774768	1.7E+07	10	35	397.17
Email-EuAll	265214	364481	16	19	25.76
Cit-HepPh	34546	420877	18	24	11.65
Cit-HepTh	27770	352285	21	29	13.10
Slashdot0811	77360	469180	26	34	18.96
Slashdot0902	82168	504230	27	35	20.42
soc-Epinions1	75879	405740	23	32	19.65
Wiki-Vote	7115	100762	17	22	5.26
p2p-Gnutella31	62586	147892	4	4	2.39
p2p-Gnutella04	10876	39994	4	4	0.62
p2p-Gnutella24	26518	65369	4	4	1.12
p2p-Gnutella25	22687	54705	4	4	0.81
p2p-Gnutella30	36682	88328	4	4	1.39
web-Stanford	281903	1992636	61	61	500.48
web-NotreDame	325729	1090108	155	155	1939.41
web-Google	875713	4322051	44	44	160.09
web-BerkStan	685230	6649470	201	201	6111.66
Amazon0601	403394	2443408	11	11	55.68
Amazon0505	410236	2439437	11	11	53.83
Amazon0302	262111	899792	7	7	16.28
Amazon0312	400727	2349869	11	11	52.18

Finding a maximum clique

1. **Upper Bound** Use Algorithm 1 on graph G to obtain an upper bound and a residual graph G'.
2. **Lower Bound** Use an exact algorithm to find the max clique on the residual graph G' to get a LB l_ω.
3. **Scale Reduction** Find the l_ω-community of G.
4. **Max Clique** Use an exact algorithm to obtain the max clique of l_ω-community of G.

Overall Algorithm

Maximum clique on SNAP graphs

Graph	n	m	UB	n_{UB}	LB	n_{LB}	ω	Time(sec)
WikiTalk	2394385	4659565	52	237	26	1487	26	1668.07
cit-Patents	3774768	16518947	35	83	10	1324	**11**	522.98
Email-EuAll	265214	364481	19	62	16	139	16	42.65
Cit-HepPh	34546	420877	24	36	18	124	**19**	17.81
Cit-HepTh	27770	352285	29	48	21	177	**23**	19.78
Slashdot0811	77360	469180	34	87	26	156	26	31.15
Slashdot0902	82168	504230	35	96	27	157	27	36.56
soc-Epinions1	75879	405740	32	61	23	359	23	39.45
Wiki-Vote	7115	100762	22	50	17	379	17	9.03
p2p-Gnutella31	62586	147892	4	57	4	57	4	2.70
p2p-Gnutella04	10876	39994	4	12	4	12	4	0.67
p2p-Gnutella24	26518	65369	4	41	4	41	4	1.04
p2p-Gnutella25	22687	54705	4	25	4	25	4	0.87
p2p-Gnutella30	36682	88328	4	42	4	42	4	1.50
web-Stanford	281903	1992636	61	126	61	126	61	500.48
web-NotreDame	325729	1090108	155	155	155	155	155	1939.41
web-Google	875713	4322051	44	48	44	48	44	160.09
web-BerkStan	685230	6649470	201	392	201	392	201	6111.66
Amazon0601	403394	2443408	11	5361	11	5361	11	61.54
Amazon0505	410236	2439437	11	4878	11	4878	11	61.76
Amazon0302	262111	899792	7	105	7	105	7	17.54
Amazon0312	400727	2349869	11	4534	11	4534	11	58.65

Maximum clique on SNAP graphs

Graph	n	m	UB	n_{UB}	LB	n_{LB}	ω	Time(sec)
WikiTalk	2394385	4659565	52	237	26	1487	26	1668.07
cit-Patents	3774768	16518947	35	83	10	1324	**11**	522.98
Email-EuAll	265214	364481	19	62	16	139	16	42.65
Cit-HepPh	34546	420877	24	36	18	124	**19**	17.81
Cit-HepTh	27770	352285	29	48	21	177	**23**	19.78
Slashdot0811	77360	469180	34	87	26	156	26	31.15
Slashdot0902	82168	504230	35	96	27	157	27	36.56
soc-Epinions1	75879	405740	32	61	23	359	23	39.45
Wiki-Vote	7115	100762	22	50	17	379	17	9.03
p2p-Gnutella31	62586	147892	4	57	4	57	4	2.70
p2p-Gnutella04	10876	39994	4	12	4	12	4	0.67
p2p-Gnutella24	26518	65369	4	41	4	41	4	1.04
p2p-Gnutella25	22687	54705	4	25	4	25	4	0.87
p2p-Gnutella30	36682	88328	4	42	4	42	4	1.50
web-Stanford	281903	1992636	61	126	61	126	61	500.48
web-NotreDame	325729	1090108	155	155	155	155	155	1939.41
web-Google	875713	4322051	44	48	44	48	44	160.09
web-BerkStan	685230	6649470	201	392	201	392	201	6111.66
Amazon0601	403394	2443408	11	5361	11	5361	11	61.54
Amazon0505	410236	2439437	11	4878	11	4878	11	61.76
Amazon0302	262111	899792	7	105	7	105	7	17.54
Amazon0312	400727	2349869	11	4534	11	4534	11	58.65

Maximum clique on SNAP graphs

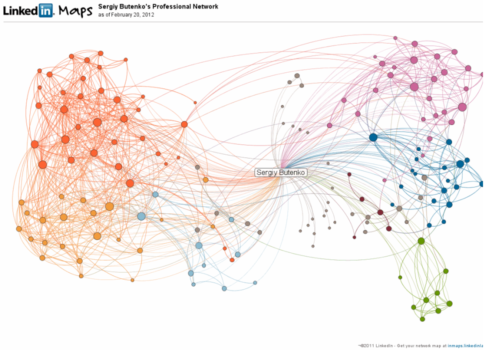
Graph	n	m	UB	n_{UB}	LB	n_{LB}	ω	Time(sec)
WikiTalk	2394385	4659565	52	237	26	1487	26	1668.07
cit-Patents	3774768	16518947	35	83	10	1324	**11**	522.98
Email-EuAll	265214	364481	19	62	16	139	16	42.65
Cit-HepPh	34546	420877	24	36	18	124	**19**	17.81
Cit-HepTh	27770	352285	29	48	21	177	**23**	19.78
Slashdot0811	77360	469180	34	87	26	156	26	31.15
Slashdot0902	82168	504230	35	96	27	157	27	36.56
soc-Epinions1	75879	405740	32	61	23	359	23	39.45
Wiki-Vote	7115	100762	22	50	17	379	17	9.03
p2p-Gnutella31	62586	147892	4	57	4	57	4	2.70
p2p-Gnutella04	10876	39994	4	12	4	12	4	0.67
p2p-Gnutella24	26518	65369	4	41	4	41	4	1.04
p2p-Gnutella25	22687	54705	4	25	4	25	4	0.87
p2p-Gnutella30	36682	88328	4	42	4	42	4	1.50
web-Stanford	281903	1992636	61	126	61	126	61	500.48
web-NotreDame	325729	1090108	155	155	155	155	155	1939.41
web-Google	875713	4322051	44	48	44	48	44	160.09
web-BerkStan	685230	6649470	201	392	201	392	201	6111.66
Amazon0601	403394	2443408	11	5361	11	5361	11	61.54
Amazon0505	410236	2439437	11	4878	11	4878	11	61.76
Amazon0302	262111	899792	7	105	7	105	7	17.54
Amazon0312	400727	2349869	11	4534	11	4534	11	58.65

Maximum clique on SNAP graphs

Graph	n	m	UB	n_{UB}	LB	n_{LB}	ω	Time(sec)
WikiTalk	2394385	4659565	52	237	26	1487	26	1668.07
cit-Patents	3774768	16518947	35	83	10	1324	**11**	522.98
Email-EuAll	265214	364481	19	62	16	139	16	42.65
Cit-HepPh	34546	420877	24	36	18	124	**19**	17.81
Cit-HepTh	27770	352285	29	48	21	177	**23**	19.78
Slashdot0811	77360	469180	34	87	26	156	26	31.15
Slashdot0902	82168	504230	35	96	27	157	27	36.56
soc-Epinions1	75879	405740	32	61	23	359	23	39.45
Wiki-Vote	7115	100762	22	50	17	379	17	9.03
p2p-Gnutella31	62586	147892	4	57	4	57	4	2.70
p2p-Gnutella04	10876	39994	4	12	4	12	4	0.67
p2p-Gnutella24	26518	65369	4	41	4	41	4	1.04
p2p-Gnutella25	22687	54705	4	25	4	25	4	0.87
p2p-Gnutella30	36682	88328	4	42	4	42	4	1.50
web-Stanford	281903	1992636	61	126	61	126	61	500.48
web-NotreDame	325729	1090108	155	155	155	155	155	1939.41
web-Google	875713	4322051	44	48	44	48	44	160.09
web-BerkStan	685230	6649470	201	392	201	392	201	6111.66
Amazon0601	403394	2443408	11	5361	11	5361	11	61.54
Amazon0505	410236	2439437	11	4878	11	4878	11	61.76
Amazon0302	262111	899792	7	105	7	105	7	17.54
Amazon0312	400727	2349869	11	4534	11	4534	11	58.65

Clustering

Partitioning a set of entities into 'natural groups' (clusters)



Fortunato's 2010 survey has over 2,300 citations according to Google Scholar.

k-Community clustering

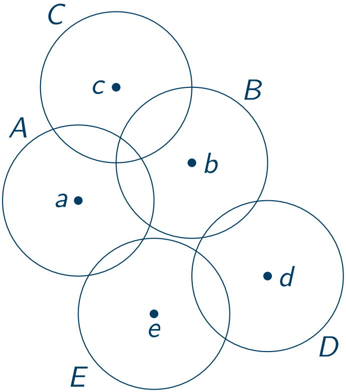
- Introduced a general purpose clustering algorithm based on clique relaxations.
- Do not aim to optimize any standard performance measure.
- Using *k*-community as a structure does well for a number of clustering quality measures.
- Enhancements to the basic algorithm can be designed according to requirements.

A. Verma and S. Butenko. Network clustering via clique relaxations: a community-based approach. In: *Graph Partitioning and Graph Clustering*. Ed. by D. A. Bader, H. Meyerhenke, P. Sanders, and D. Wagner. American Mathematical Society, 2013, pp.125–136.

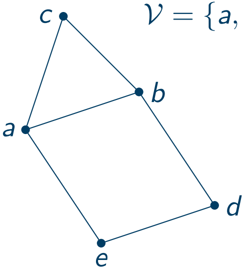
DIMACS Challenge

Unit disk graphs (UDGs)

A unit-disk graph (UDG) can be defined as the intersection graph of closed disks of equal (e.g., unit) diameter.

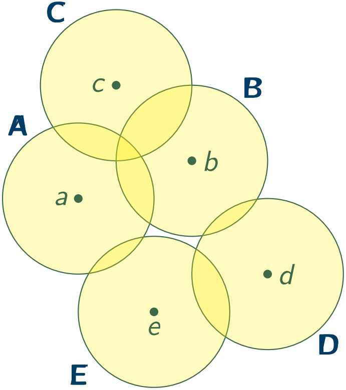


$$\mathcal{G} = (\mathcal{V}, \mathcal{E})$$
$$\mathcal{V} = \{a, b, c, d, e\}$$

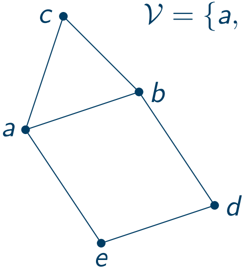


Unit disk graphs (UDGs)

A unit-disk graph (UDG) can be defined as the intersection graph of closed disks of equal (e.g., unit) diameter.



$$\mathcal{G} = (\mathcal{V}, \mathcal{E})$$
$$\mathcal{V} = \{a, b, c, d, e\}$$



Unit disk graphs (UDGs)

Many of the classical optimization problems on graphs remain NP-hard when restricted to UDGs

- maximum independent set
- minimum vertex cover
- graph coloring
- minimum dominating set
- minimum connected dominating set

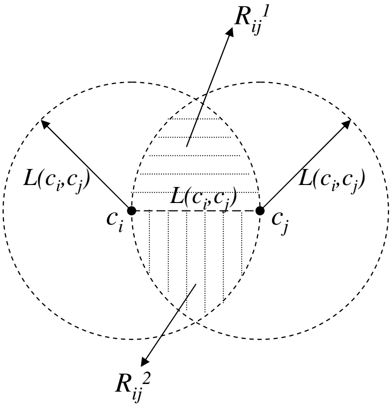
Unit disk graphs (UDGs)

Many of the classical optimization problems on graphs remain NP-hard when restricted to UDGs

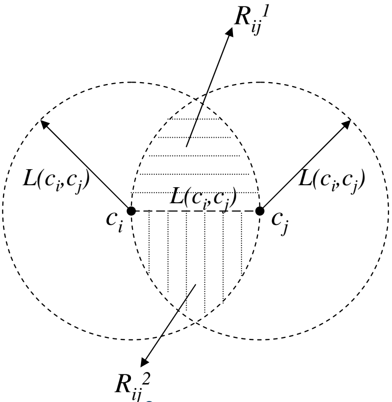
- maximum independent set
- minimum vertex cover
- graph coloring
- minimum dominating set
- minimum connected dominating set

The maximum clique problem is a notable exception.

Maximum clique in UDG



Maximum clique in UDG

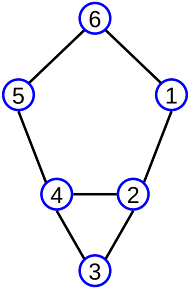


- Reduces to solving $O(|\mathcal{V}|^2)$ instances of the maximum independent set problem in bipartite graphs; $O(|\mathcal{V}|^{4.5})$ time.

Distance-based clique relaxations

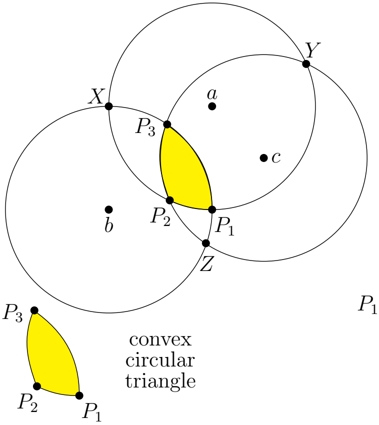
A k-clique is a subset of vertices C such that for every $i, j \in C$, $d(i,j) \leq k$.

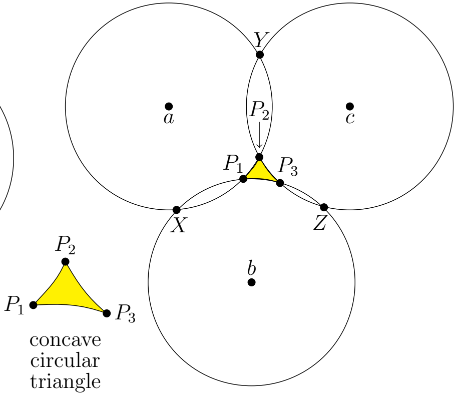
A k-club is a subset of vertices D such that $diam(G[D]) \leq k$.



- $\{2,3,4\}$ is a 1-club ... the "regular" clique
- $\{1,2,4,5,6\}$ is a 2-club
- $\{1,2,3,4,5\}$ is a 2-clique but NOT a 2-club
- maximality of a 2-club is harder to test

Circular triangles





Domination for 2-cliques in a UDG

We call a subset $\mathcal{S}$ of nodes k-dominated in $\mathcal{G}$ if there is a subset $\mathcal{D} \subseteq \mathcal{V}$ of at most k nodes such that any $u \in \mathcal{S} \setminus \mathcal{D}$ has a neighbor $v \in \mathcal{D}$.

Proposition

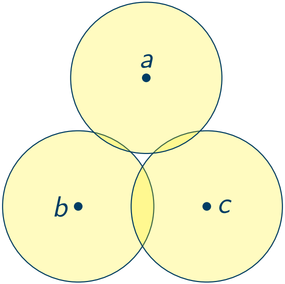
Any 2-clique in a UDG is 4-dominated.

Note: we do not require the elements in a dominating set to be members of the 2-clique.

Domination for 2-cliques in a UDG

Sketch of the proof. Let $\mathcal{K}$ be an arbitrary 2-clique in a UDG $\mathcal{G}$

Case 1: There exist **A**, **B**, and **C** in $\mathcal{K}$ such that $\mathbf{A} \cap \mathbf{B} \cap \mathbf{C} = \emptyset$



Consider **A**, **B**, and **C** in $\mathcal{K}$ that yield a concave circular triangle with the largest area. Then every other disk in $\mathcal{K}$ must overlap at least one entire lens $\mathbf{A} \cap \mathbf{B}$, $\mathbf{A} \cap \mathbf{C}$, or $\mathbf{B} \cap \mathbf{C}$; $\mathcal{K}$ is 3-dominated.

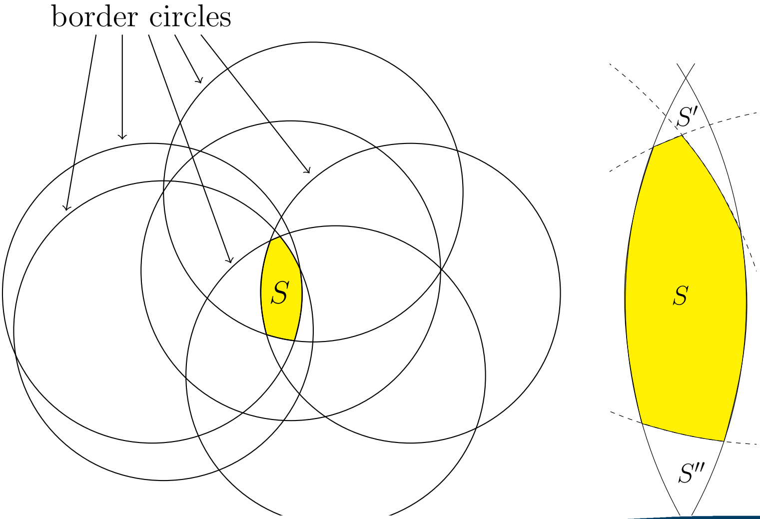
Domination for 2-cliques in a UDG

Case 2: $\mathbf{A} \cap \mathbf{B} \cap \mathbf{C} \neq \emptyset$ for any **A**, **B**, and **C** in $\mathcal{K}$.

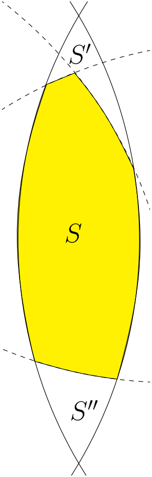
We use Helly's theorem in two dimensions: if $\mathcal{F}$ is a finite family of at least 3 convex sets on the 2-dimensional plane and every 3 members of $\mathcal{F}$ have a common point, then there is a point common to all members of $\mathcal{F}$.

- By Helly's theorem there exists a set of points S in $\mathbb{R}$ that are common for all members of $\mathcal{K}$.
- Clearly, if there is a node of $\mathcal{G}$ in S, the 2-clique is 1-dominated.
- Assume that S contains no nodes of $\mathcal{G}$.

Domination for 2-cliques in a UDG



Domination for 2-cliques in a UDG

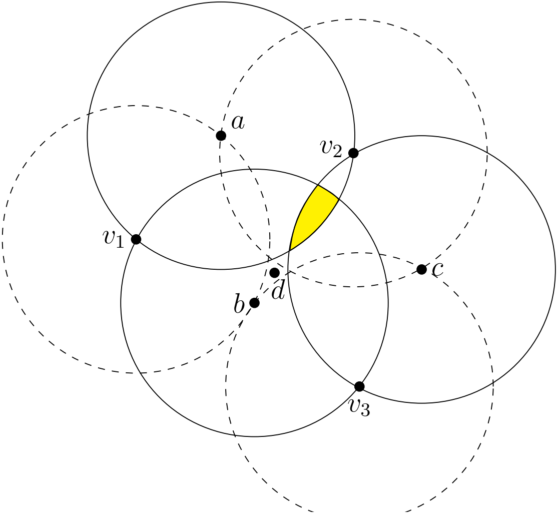


Consider an arbitrary pair of border disks **A** and **B** in $\mathcal{K}$ corresponding to non-consecutive pieces of the border of S.

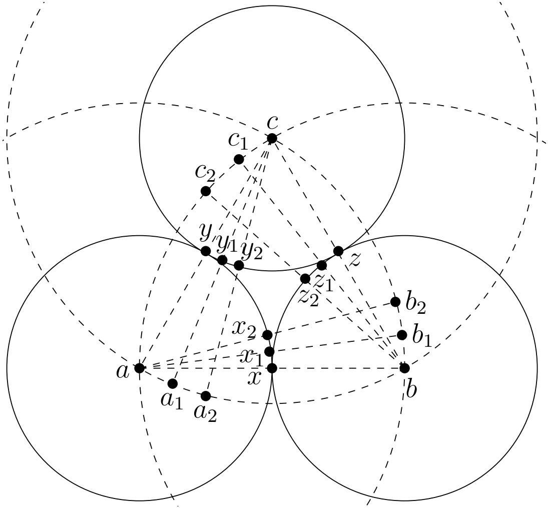
Since $\mathbf{A}, \mathbf{B} \in \mathcal{K}$, there must be a node p of $\mathcal{G}$ in $\mathbf{A} \cap \mathbf{B} \setminus S$.

- If S', S'' both contain the graph's nodes, p' and p'', respectively, then the 2-clique is 2-dominated by p' and p''.
- If only one of S', S'' contains nodes of $\mathcal{G}$, then we can find three border disks $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ such that $\mathbf{A} \cap \mathbf{B} \cap \mathbf{C}$ contains no nodes of $\mathcal{G}$.

Domination for 2-cliques in a UDG



2-Clique that is not 2-dominated



Domination for 2-clubs in a UDG

Corollary

Any 2-club in a UDG is 3-dominated.

This fact can potentially be used in designing exact algorithms for the maximum 2-club problem as follows.

- Instead of solving the problem for the original graph, we can solve it for induced subgraphs of all subsets of 3 vertices together with their neighbors.
- This may help solving instances where all such subgraphs are substantially smaller than the original graph.

A $\frac{1}{2}$-approximation algorithm for the maximum 2-clique problem in a UDG

Proposition

There exists a $\frac{1}{2}$-approx. algorithm for the maximum 2-clique problem in a UDG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that runs in $O(|\mathcal{V}|^{4.5})$ time.

- For a pair $\{v_1, v_2\}$ of nodes, $\mathcal{G}'(v_1, v_2)$, which is the subgraph of $\mathcal{G}^2$ induced by $N_{\mathcal{G}}[v_1] \cup N_{\mathcal{G}}[v_2]$, is a co-bipartite graph.
- We can identify the largest 2-clique $\mathcal{K}'$ dominated by 2 elements in $\mathcal{G}$ in $O(|\mathcal{V}|^{4.5})$ time.
- All 2-cliques are 4-dominated $\Rightarrow$ at least half of the nodes of a maximum 2-clique $\mathcal{K}^*$ in $\mathcal{G}$ must be dominated by 2 nodes.
- By weak heredity of 2-cliques, $|\mathcal{K}'| \geq \frac{1}{2}|\mathcal{K}^*|$.

The maximum 2-clique problem in a uniform random UDG

- In a sample set of experiments, we generated 3,500 uniform random UDGs of 50 nodes and 100 random UDGs of 100 nodes for each density in the range from .05 to 1 in increments of .05.
- In all 70,000 experiments with 50-node instances and all 2,000 experiments with 100-node instances, the size of the maximum 2-clique and the 2-clique found by the proposed approximation algorithm matched.

Minimum dominating set problem in graphs of diameter two

- Since all 2-clubs are 3-dominated, the minimum dominating set problem is polynomially solvable in UDGs of diameter two.
- In contrast, we show by reduction from Vertex Cover that Dominating Set is NP-complete when restricted to (general) graphs of diameter two.

Conclusion

- We provide a $\frac{1}{2}$-approximation algorithm for the maximum 2-clique problem in UDGs.
- The performance of the algorithm was explored in the context of uniform random UDGs.
- We have established that any diameter-two UDG has a dominating set of size at most 3, implying that the minimum dominating set problem is polynomially solvable in diameter-two UDGs.

Open questions

- What is the computational complexity of the maximum s-clique and s-club problems in UDGs? Is an efficient exact algorithm possible?
- Can one construct an example of a 2-clique that is not 3-dominated, or are all 2-cliques 3-dominated rather than 4-dominated, in which case the proposed algorithm becomes $\frac{2}{3}$-approximate?
- While the concepts of 2-cliques and 2-clubs are closely related, the proposed method does not directly extend to the maximum 2-club problem. Can one design an approximation algorithm for the maximum 2-club problem with a similar approximation ratio?

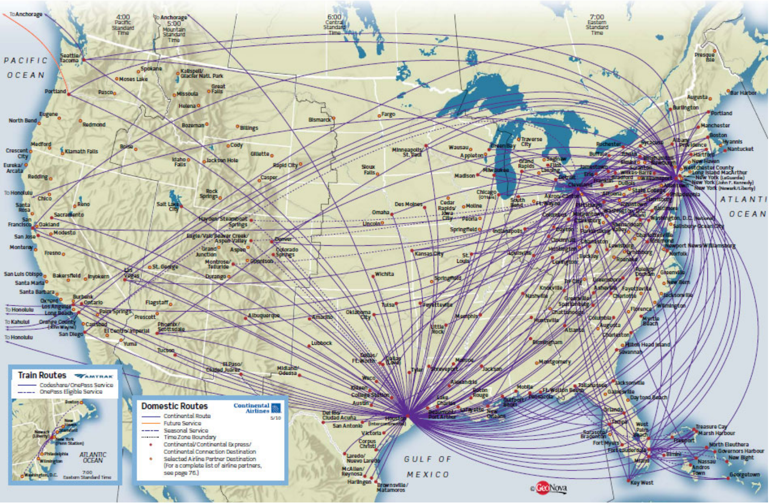
Hub-and-spoke model

- The hub-and-spoke structure provides passengers a convenient access (through hub cities) to a large number of destinations that could not possibly support point to point service.
- The wide range of services facilitated by the hub-and-spoke structure attracts a larger number of customers.
- The hub-and-spoke structure provides a 2-hop connectivity with the minimum possible total number of connections.

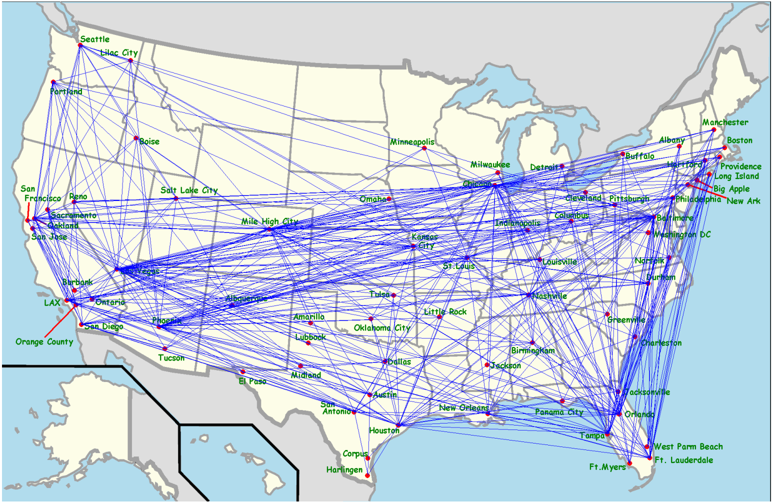
Disadvantages of hub-and-spoke

- Poor reliability: Removing just one hub node may completely disconnect the network.
- A high volume of passenger flows at hub airports creates inefficiencies (e.g., gate security check)
- Environmental concerns: An excessive number of flights results in airside and landside congestion, aircraft noise and emissions.

Hub-and-spoke network (Continental)



Point-to-point network (Southwest)



Need for restructuring

- Prediction: The advantages of the point-to-point operation will lead major airlines to reexamine their favorability towards the hub-and-spoke model and design more balanced connectivity structures.
- Hansson et al., 2002:

  "The airline business model - essentially designed to make anyone from anywhere to everywhere, seamlessly - was a great innovation, but is no longer economically sustainable in its current form."

Some issues to address

- The Southwest model proved effective for a small or medium size network, but would it work as well for larger carriers?
- Which connectivity properties of an airline network have the highest impact on quality and reliability of service provided by the airline?
- What are the minimum changes that need to be made to a current network in order to improve these connectivity properties?
- How to develop a network structure that will combine advantages of both hub-and-spoke and point-to-point approaches?

Desirable properties

(a) An airline network should have a low diameter in order to provide a fast and easy access between cities in the network.

(b) The total number of connections in the network should be considerably smaller than the maximum possible number of connections.

(c) An airline network should not contain a large group of nodes any two of which are distance > 2 or > 3 from each other.

(d) Removing one or several nodes or arcs from the network should not lead to a large increase in the network's diameter.

Airline networks data

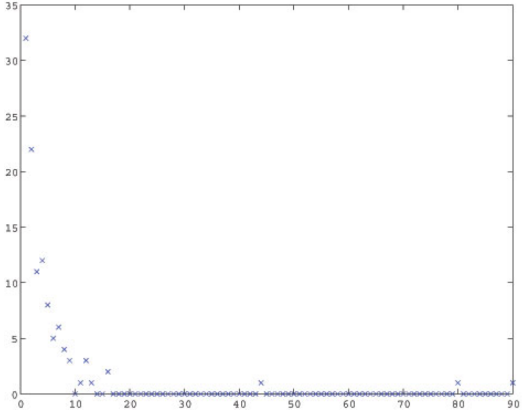
We used Bureau of Transportation Statistics data for July 2005 (including distances and passenger quantities). Flights with less than 100 passengers for the month were not considered.

Table : Characteristics of the airline networks

	AA	WN	DL	CO	NW	UA	US	ST	All
$\|V\|$	90	63	104	73	113	80	63	140	162
$\|E\|$	331	793	330	156	317	258	282	760	1944
E.D.	0.08	0.41	0.06	0.06	0.05	0.08	0.14	0.08	0.15
ω_2	77	57	100	64	91	67	61	102	103

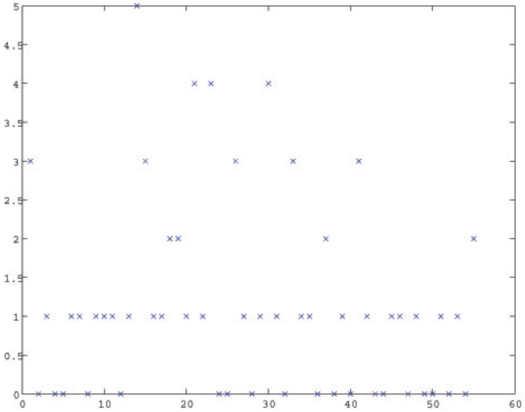
Degree distributions

Northwest Airlines



Degree distributions

Southwest Airlines



Maximum s-plex sizes

Table : Maximum s-plex sizes for the major airline networks

s	AA	WN	DL	CO	NW	UA	US	ST	All
1	7	14	7	5	7	7	7	11	20
2	9	18	9	6	8	9	8	12	25
3	10	20	10	7	9	10	10	15	28
4	11	22	11	8	10	11	11	17	30
5	12	23	12	9	11	11	12	19	32

k-Core based routing

- For a graph $G = (V, E)$, a subset S of vertices is called a k-core if the minimum degree of a vertex in $G(S)$ is k.

k-Core based routing

- For a graph $G = (V, E)$, a subset S of vertices is called a k-core if the minimum degree of a vertex in $G(S)$ is k.
- To design a k-core based routing system, we first consider a complete graph with vertices corresponding to airports. We assign a weight w_{ij} to each edge (i, j) as follows:

  $$w_{ij} = d_{ij}/p_{ij},$$

  where d_{ij} is the distance between i and j and p_{ij} is the number of passengers travelling between i and j during a certain time period (month).

k-Core based routing

- We solve the following problem:

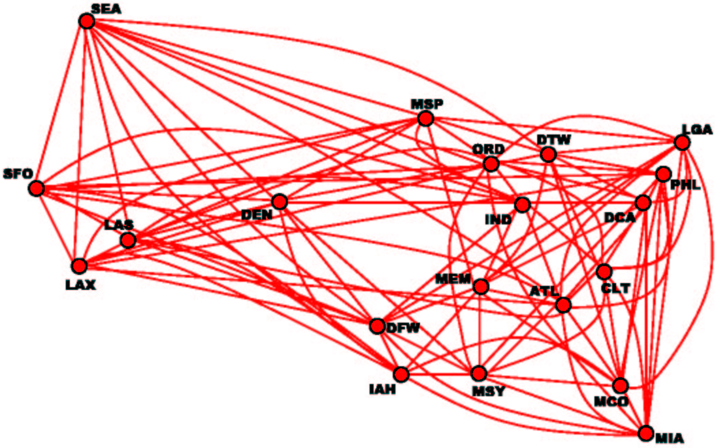
$$\min \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} w_{ij} x_{ij}$$

subject to

$$\sum_{j=1}^{|V|} x_{ij} \geq k,\ \forall i \in V;$$

$$x_{ij} \in \{0, 1\},\ i, j = 1, \ldots, |V|.$$

A 10-core network for 20 airports

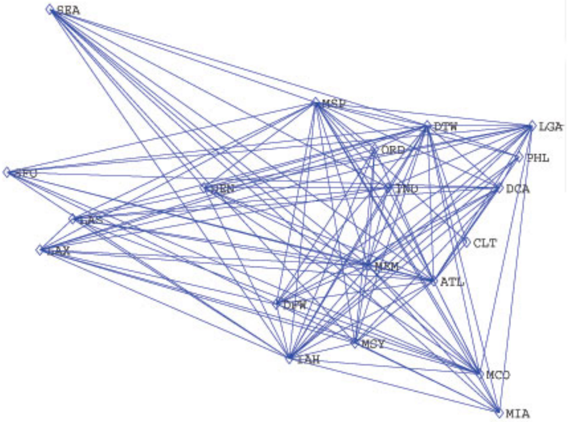


A 10-core network for 20 airports

Airport	Connections	Airport	Connections
IAH	10	ORD	14
IND	10	PHL	10
LAS	10	CLT	10
LAX	10	DCA	10
LGA	10	DEN	10
MCO	10	DFW	11
MEM	10	DTW	10
MIA	10	ATL	13
MSP	10	SEA	10
MSY	10	SFO	10

Total number of edges - 104.

Skyteam subnetwork



Skyteam subnetwork

Airport	Connections	Airport	Connections
IAH	18	ORD	8
IND	12	PHL	5
LAS	10	CLT	3
LAX	12	DCA	12
LGA	14	DEN	10
MCO	13	DFW	10
MEM	18	DTW	19
MIA	9	ATL	19
MSP	19	SEA	10
MSY	10	SFO	9

Total number of edges - 120.

Comparison

- We compare some of the major airline networks (restricted to the 20 airports) using the following measure:

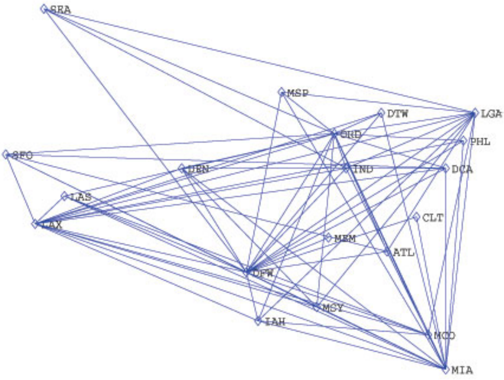
$$pd = \sum_{(i,j)\in E} d_{ij}^G p_{ij},$$

  where d_{ij}^G is the length of the shortest path between i and j in G.
- We denote by pd^* the minimum possible value for pd, which is achieved if the network is a complete point-to-point network.

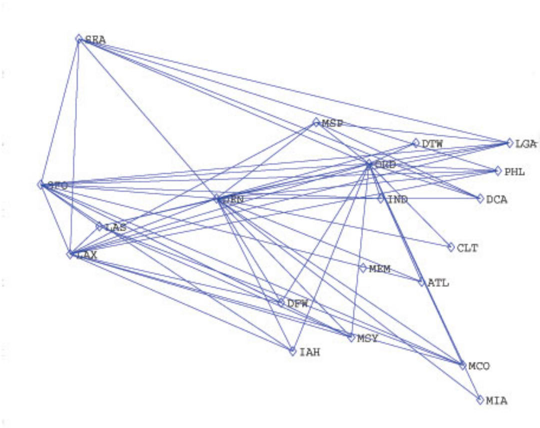
Comparison

Airline	Connections	pd/pd^*
AA	75	1.19
DL	35	1.61
NW	77	1.30
UA	62	4.58
US	49	1.75
CO	23	8.39
ST	120	1.15
10-core	104	1.02

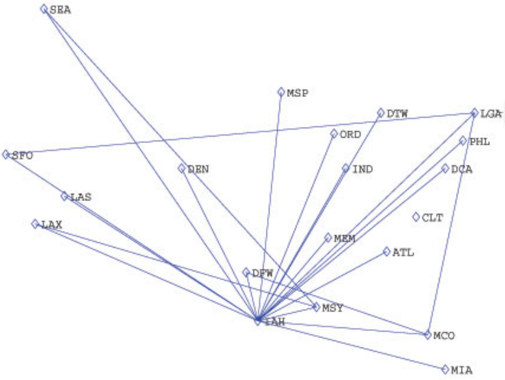
American Airlines subnetwork



United Airlines subnetwork



Continental Airlines subnetwork



Outline

Basics of the complexity theory

- Given a combinatorial optimization problem, a natural question is:

  is this problem "easy" or "hard"?

- How do we distinguish between "easy" and "hard" problems?

"Easy" problems

- By easy or tractable problems we mean the problems that can be solved in time polynomial with respect to their size.
- We also call such problems polynomially solvable and denote the class of polynomially solvable problems by $\mathcal{P}$.
  - Sorting
  - Minimum weight spanning tree
  - Linear programming

Defining "hard" problems

- ◇ How do we define "hard" problems?
- ◇ How about defining hard problems as all problems that are not easy, i.e., not in $\mathcal{P}$?
- ◇ Then some of the problems in such a class could be TOO hard – we cannot even hope to be able to solve them.
- ◇ We want to define a class of hard problems that we may be able to solve, if we are lucky (say, we may be able to guess the solution and check that it is indeed correct).

Three versions of optimization problems

Consider a problem

$$\min f(x) \text{ subject to } x \in X.$$

- Optimization version: find x from X that maximizes $f(x)$;
  Answer: x^* maximizes $f(x)$
- Evaluation version: find the largest possible $f(x)$;
  Answer: the largest possible value for $f(x)$ is f^*
- Recognition version: Given f^*, does there exist an x such that $f(x) \geq f^*$?
  Answer: "yes" or "no"

Recognition problems

◇ Recognition problems can still be undecidable.

Halting problem: Given a computer program with its input, will it ever halt?

◇ On the other hand, if we pick a random feasible solution and it happens to give "yes" answer, then we solved the problem in polynomial time.

Max clique: Randomly pick a solution (a clique). If its size is $\geq s$ *(which we can verify in polynomial time), then obviously the answer is "yes". This clique can be viewed as a certificate proving that this is indeed a yes instance of max clique.*

Class $\mathcal{NP}$

- ◇ We only consider problems for which any yes instance there exists a concise (polynomial-size) certificate that can be verified in polynomial time.
- ◇ We call this class of problems nondeterministic polynomial and denote it by $\mathcal{NP}$.

$\mathcal{P}$ vs $\mathcal{NP}$

- Note that any problem from $\mathcal{P}$ is also in $\mathcal{NP}$ (i.e., $\mathcal{P} \subseteq \mathcal{NP}$), so there are easy problems in $\mathcal{NP}$.
- So, are there "hard" problems in $\mathcal{NP}$, and if there are, how do we define them?
- We don't know if $P = NP$, but "most" people believe that $P \neq NP$.
- An "easy" way to make $1,000,000!
  http://www.claymath.org/millennium/P_vs_NP/
- We can call a problem hard if the fact that we can solve this problem would mean that we can solve any other problem in comparable amount of time.

Polynomial reducibility

- Reduce π_1 to π_2: if we can solve π_2 fast, then we can solve π_1 fast, given that the reduction is "easy".
- Polynomial reduction from π_1 to π_2 requires existence of polynomial-time algorithms
  1. A_1 converts an input for π_1 into an input for π_2;
  2. A_2 converts an output for π_2 into output for π_1.
- Transitivity: If π_1 is polynomially reducible to π_2 and π_2 is polynomially reducible to π_3 then π_1 is polynomially reducible to π_3.

NP-complete problems

- A problem π is called NP-complete if
  1. $\pi \in NP$;
  2. Any problem from NP can be reduced to π in polynomial time.
- A problem π is called NP-hard if any problem from NP can be reduced to π in polynomial time. (no $\pi \in NP$ requirement)
- Due to transitivity of polynomial reducibility, in order to show that a problem π is $\mathcal{NP}$-complete, it is sufficient to show that
  1. $\pi \in NP$;
  2. There is an NP-complete problem π' that can be reduced to π in polynomial time.

To use this observation, we need to know at least one $\mathcal{NP}$-complete problem...

Satisfiability (SAT) problem

- A Boolean variable x is a variable that can assume only the values true and false.
- Boolean variables can be combined to form Boolean formulas using the following logical operations:
  1. Logical AND ($\wedge$ or $\cdot$) -conjunction
  2. Logical OR ($\vee$ or $+$) - disjunction
  3. Logical NOT ($\bar{x}$)
- A clause is $C_j = \bigvee_{p=1}^{k_j} y_{j_p}$, where a *literal* y_{j_p} is x_r or $\bar{x}_r$ for some r.
- Conjunctive normal form (CNF):$F = \bigwedge_{j=1}^{m} C_j$, where C_j is a clause.

Satisfiability problem

- A CNF F is called satisfiable if there is an assignment of variables such that $F = 1$ ($TRUE$).
- Satisfiability (SAT) problem: Given m clauses $C_1, \ldots, C_m$ involving the variables $x_1, \ldots, x_n$, is the CNF

$$F = \bigwedge_{j=1}^{m} C_j,$$

satisfiable?

Theorem (Cook, 1971)

SAT is NP-complete.

“Best” approximation algorithms and heuristics

- For some problems there are hardness of approximation results stating that the problem is hard to approximate within a certain factor.
- For example, the k-center problem is hard to approximate within a factor better than 2.
- Then any polynomial-time algorithm approximating the k-center problem within the factor of 2 can be considered the “best” approximation algorithm for this problem.

"Best" approximation algorithms and heuristics

- However, some problems are even harder to approximate. For example, the maximum clique is hard to approximate within a factor $n^{1-\epsilon}$ for any positive ϵ.
- In this case, by the "best" heuristic we could mean a heuristic that cannot be provably outperformed by any other polynomial-time algorithm (unless $P = NP$).

Recognizing the gap between k-club and l-club numbers

Theorem

Let positive integer constants k and l, $l < k$ be given. The problem of checking whether $\bar{\omega}_l(G) = \bar{\omega}_k(G)$ is NP-hard.

Note that

$$\omega(G) \leq \Delta(G) + 1 \leq \bar{\omega}_k(G)$$

and observe that we can easily check whether $\omega(G) = \Delta(G) + 1$.

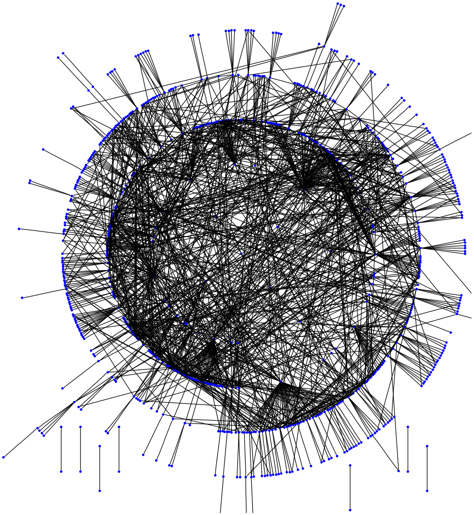
Hence, it is NP-hard to check whether $\bar{\omega}_k(G) = \Delta(G) + 1$.

"Best" heuristics for k-club/clique

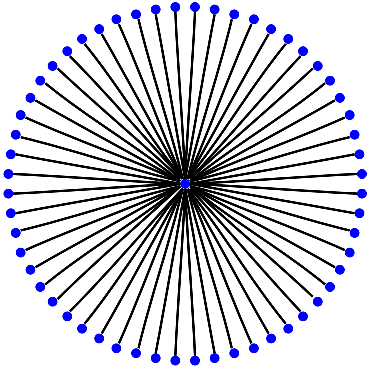
Corollary

Let k be a fixed integer, $k \geq 2$. Unless $P = NP$, there cannot be a polynomial time algorithm that finds a k-club of size greater than $\Delta(G) + 1$ whenever such a k-club exists in the graph.

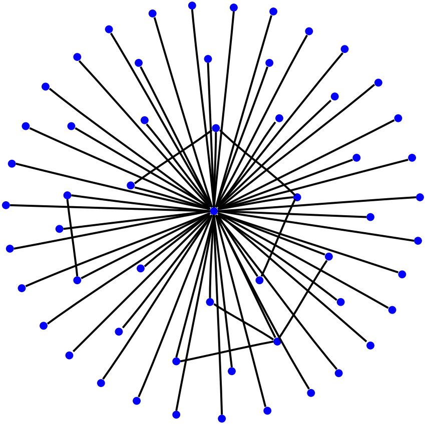
Protein Interaction Networks



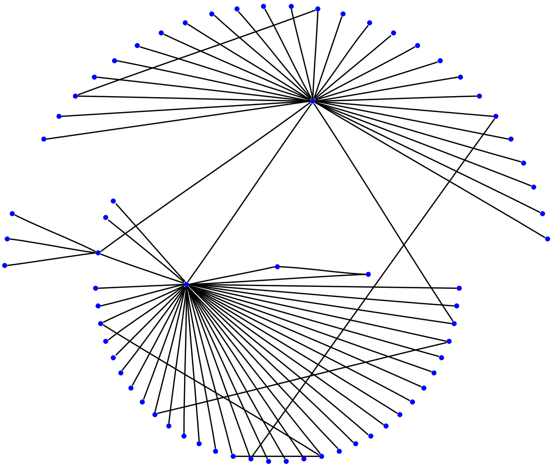
A max 2-club/clique of S. Cerevisiae.



A max 2-club/clique of H. Pylori.



A max 3-clique/club of S. Cerevisiae



Outline

Upper bound on the quasi-clique number

Proposition

The γ-clique number $\omega_\gamma(G)$ of a graph G with n vertices and m edges satisfies the following inequality:

$$\omega_\gamma(G) \leq \frac{\gamma + \sqrt{\gamma^2 + 8\gamma m}}{2\gamma}. \tag{1}$$

Moreover, if a graph G is connected then

$$\omega_\gamma(G) \leq \frac{\gamma + 2 + \sqrt{(\gamma+2)^2 + 8(m-n)\gamma}}{2\gamma}. \tag{2}$$

Relation between $\omega_\gamma(G)$ and $\omega(G)$

Proposition

The γ-clique number $\omega_\gamma(G)$ and the clique number $\omega(G)$ of graph G satisfy the following inequalities:

$$\frac{\omega(G)-1}{\omega(G)} \leq \frac{\omega_\gamma(G)-1}{\omega_\gamma(G)} \leq \frac{1}{\gamma}\frac{\omega(G)-1}{\omega(G)}. \quad (3)$$

Corollary

If $\gamma > 1 - \frac{1}{\omega(G)}$ then

$$\omega_\gamma(G) \leq \frac{\omega(G)\gamma}{1-\omega(G)+\omega(G)\gamma}. \quad (4)$$

Relation between $\omega_\gamma(G)$ and $\omega(G)$

Table : The value of upper bound (4) on γ-clique number with $\gamma = 0.95, 0.9, 0.85$ for graphs with small clique number.

$\omega(G)$	$1 - \frac{1}{\omega(G)}$	0.95	0.9	0.85
3	0.667	3.35	3.86	4.64
4	0.75	4.75	6	8.5
5	0.8	6.33	9	17
6	0.83	8.14	13.5	51
7	0.86	10.23	21	–
8	0.88	12.67	36	–
9	0.89	15.55	81	–
10	0.9	19	–	–

Outline

Probabilistic method

- A feasible solution of a discrete optimization problem (P) usually consists of a finite set of elements (e.g., vertices or edges of a graph) satisfying some property, and the objective is often to maximize/minimize the size of this set.
- Let $f^*(P)$ denote the optimal objective value of (P).
- In probabilistic method, with each such element i we associate its probability x_i of being included (randomly and independently) in some feasible (optimal) solution.

Probabilistic method

- If we compute the expected size $f_e(x)$ of the set of picked elements forming a feasible solution of (P), then we have

$$f^*(P) \leq f_e(x) \Rightarrow f^*(P) \leq \min_{x \in [0,1]^n} f_e(x).$$

- If on the other hand we can find $x^* \in [0,1]^n$ such that $f_e(x^*) = f^*(P)$, and the corresponding feasible solution $S(x^*)$ of problem (P) has size $f^*(P)$, then we have

$$f^*(P) = f_e(x^*) \geq \min_{x \in [0,1]^n} f_e(x).$$

Probabilistic method

- So, we obtain a formulation of (P) as a problem of minimizing a continuous function over the unit hypercube $[0, 1]^n$:

$$f^*(P) = \min_{x \in [0,1]^n} f_e(x).$$

- Next we illustrate this approach on the maximum independent set problem and the minimum dominating set problem.

Independence number

- Pick, randomly and independently, each vertex i of V with probability x_i.
- Let I be the set of picked vertices with no picked neighbors.
- $\Pr(i \in I) = x_i \prod_{j \in N(i)} (1 - x_j)$.
- Then the expected size of I is

$$E(|I|) = f(x) = \sum_{i=1}^{n} x_i \prod_{j \in N(i)} (1 - x_j).$$

Independence number

- Note that I is an independent set, thus,

$$\alpha(G) \geq \max_{x \in [0,1]^n} f(x) = \max_{x \in [0,1]^n} \sum_{i=1}^{n} x_i \prod_{j \in N(i)} (1 - x_j).$$

- On the other hand, for the characteristic vector x^* of a maximum independent set we have $f(x^*) = \alpha(G)$, so $\alpha(G) \leq \max_{x \in [0,1]^n} f(x)$, therefore $\alpha(G) = \max_{x \in [0,1]^n} f(x)$.

Domination number

- Pick, randomly and independently, each vertex i of V with probability x_i.
- Let X be the random set of all vertices picked and let Y be the random set of vertices that do not have any neighbor in X.
- The expected value of $|X|$ is $\sum_{i \in V} x_i$.

$$\begin{aligned} \forall i \in V,\ \Pr(i \in Y) &= \Pr(i \text{ and its neighbors are not in } X) \\ &= \prod_{j \in N[i]} (1 - x_j). \end{aligned}$$

- $E(|Y|) = \sum_{i \in V} \Pr(i \in Y) = \sum_{i \in V} \prod_{j \in N[i]} (1 - x_j)$.

Domination number

- $E(|X|) = \sum_{i \in V} x_i,\ E(|Y|) = \sum_{i \in V} \prod_{j \in N[i]} (1 - x_j)$, so
  $E(|X| + |Y|) = f(x) = \sum_{i \in V} \left(x_i + \prod_{j \in N[i]} (1 - x_j) \right)$
- Note that $X \cup Y$ is a dominating set, thus,

$$\gamma(G) \leq \min_{x \in [0,1]^n} f(x) = \min_{x \in [0,1]^n} \sum_{i \in V} \left(x_i + \prod_{j \in N[i]} (1 - x_j) \right)$$

  On the other hand, for the characteristic vector x^* of a minimum dominating set we have $f(x^*) = \gamma(G)$, so $\gamma(G) \geq \min_{x \in [0,1]^n} f(x)$, therefore $\gamma(G) = \min_{x \in [0,1]^n} f(x)$.

Math programming formulations

- Consider a simple undirected graph $G = (V, E)$ with n vertices
- Let $A = [a_{ij}]_{i,j=1}^{n}$ be the adjacency matrix of G
- Let $x = (x_1, \ldots, x_n)$ be a 0-1 vector with $x_i = 1$ if node i belongs to G_s, and $x_i = 0$ otherwise.

Math programming formulations

- G_S is a γ-clique if it has at least $\gamma|G_S|(|G_S|-1)/2$ edges
- We have: $|G_S| = \sum_{i=1}^{n} x_i$
- This number of edges can be expressed in terms of vector x as:

$$\frac{1}{2}\gamma \sum_{i=1}^{n} x_i \left(\sum_{i=1}^{n} x_i - 1 \right) = \frac{1}{2}\gamma \left(\sum_{i,j=1}^{n} x_i x_j - \sum_{i=1}^{n} x_i \right)$$

$$= \frac{1}{2}\gamma \left(\sum_{i,j=1; i\neq j}^{n} x_i x_j + \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i \right) = \frac{1}{2}\gamma \sum_{i,j=1; i\neq j}^{n} x_i x_j$$

- The number of edges in G_S is $\frac{1}{2} \sum_{i,j=1}^{n} a_{ij} x_i x_j$

Math programming formulations

We obtain the following 0-1 problem with one quadratic constraint:

$$\max \sum_{i=1}^{n} x_i$$

subject to:

$$\sum_{i,j=1}^{n} a_{ij} x_i x_j \geq \gamma \sum_{i,j=1;i \neq j}^{n} x_i x_j$$

Math programming formulations

Define $w_{ij} = x_i x_j$. The constraint $w_{ij} = x_i x_j$ is equivalent to

$$w_{ij} \leq x_i \, , w_{ij} \leq x_j \, , w_{ij} \geq x_i + x_j - 1 \, .$$

Linearized formulation: $\max \sum_{i=1}^{n} x_i$

$$\text{subject to} : \sum_{i,j=1}^{n} a_{ij} w_{ij} \geq \gamma \sum_{i,j=1; i \neq j}^{n} w_{ij}$$

$$w_{ij} \leq x_i \, , w_{ij} \leq x_j \, , w_{ij} \geq x_i + x_j - 1 \, .$$

$$w_{ij}, x_i \in \{0, 1\}, \ \forall i < j = 1, \ldots, n$$

$O(n^2)$ 0-1 variables, $O(n^2)$ constraints

References

- J. Pattillo, N. Youssef, and S. Butenko. On clique relaxation models in network analysis. *European Jour. of Oper. Res.*, 226: 9–18, 2013.
- B. Balasundaram, S. Butenko, and I. Hicks. Clique relaxations in social network analysis: the maximum k-plex problem. *Operations Research*, 59: 133–142, 2011.
- S. Trukhanov, C. Balasubramaniam, B. Balasundaram, and S. Butenko. Algorithms for detecting optimal hereditary structures in graphs, with application to clique relaxations. *Computational Optimization and Applications*, 56: 113–130, 2013.
- V. Boginski, S. Butenko, O. Shirokikh, S. Trukhanov, and J. Gil-Lafuente. A network-based data mining approach to portfolio selection via weighted clique relaxations. *Annals of Operations Research*, to appear.
- J. Pattillo, A. Veremyev, S. Butenko, V. Boginski. On the maximum quasi-clique problem. *Discrete Applied Math*, 161: 244–257, 2013
- A. Buchanan, J. S. Sung, V. Boginski, S. Butenko. On connected dominating sets of restricted diameter. *EJOR*, 236: 410–418, 2014

References

J. Pattillo, Y. Wang, and S. Butenko. Approximating 2-cliques in unit disk graphs. *Discrete Applied Math*, 166: 178–187, 2014

S. Shahinpour and S. Butenko.Algorithms for the maximum k-club problem in graphs.*J. of Combinatorial Optim*, 26: 520–554, 2013

S. Sethuraman and S. Butenko. The maximum ratio clique problem. *Computational Management Science*, to appear.

A. Verma, A. Buchanan, and S. Butenko. Solving the Maximum Clique and Vertex Coloring Problems on Very Large Sparse Networks. Under revision.

"The whole is more than the sum of its parts."
–Aristotle (384-322 BC)

Thank you!