Graph Classes

www.graphclasses.org

In blue I have highlighted some of the classes that play a part in these lectures. In short, interval plays a part in “1D maxcon” problems. Their intersections with split and threshold seem to be useful sub-structuring.

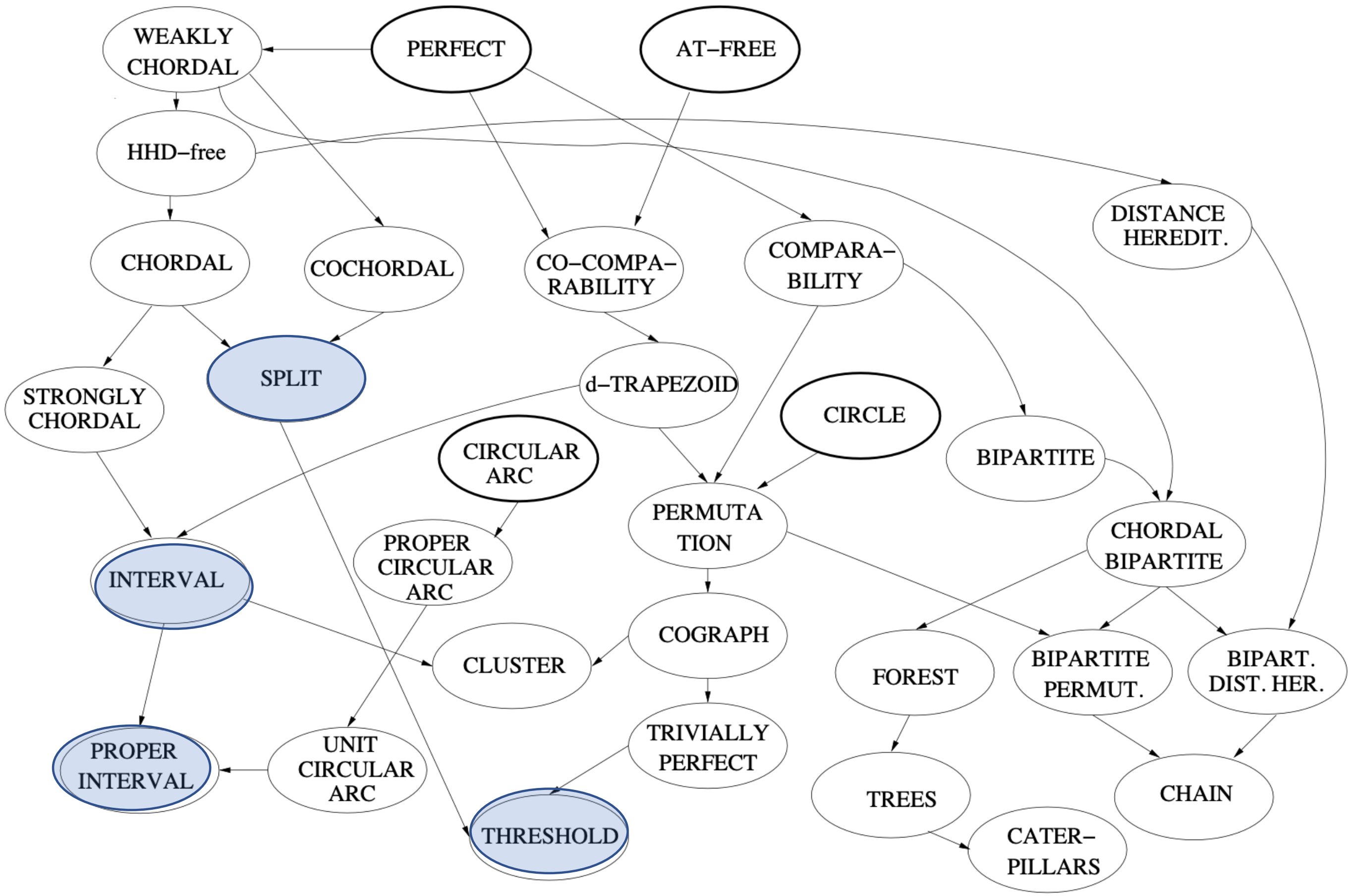


Figure 2.2: This diagram represents the inclusion relations among the graph classes we described in Chapter 2.1. An arrow from class A to class B means that $B \subset A$, and if there are arrows from two classes A and B to a class C, it means that $C \subseteq A \cap B$. The thicker ovals, represents the classes that are not perfect.

From Mancini thesis – this author prefers arrows drawn downwards rather than upwards as I use in Hasse Diagrams

I expect (generalizations of) Chordal (to hypergraphs) will play a role in describing ”all” MaxCon problems…..

Graphs (and slightly less so, Hypergraphs) are so well studied…they provide a “ready” taxonomy/structure to a huge array of problems.

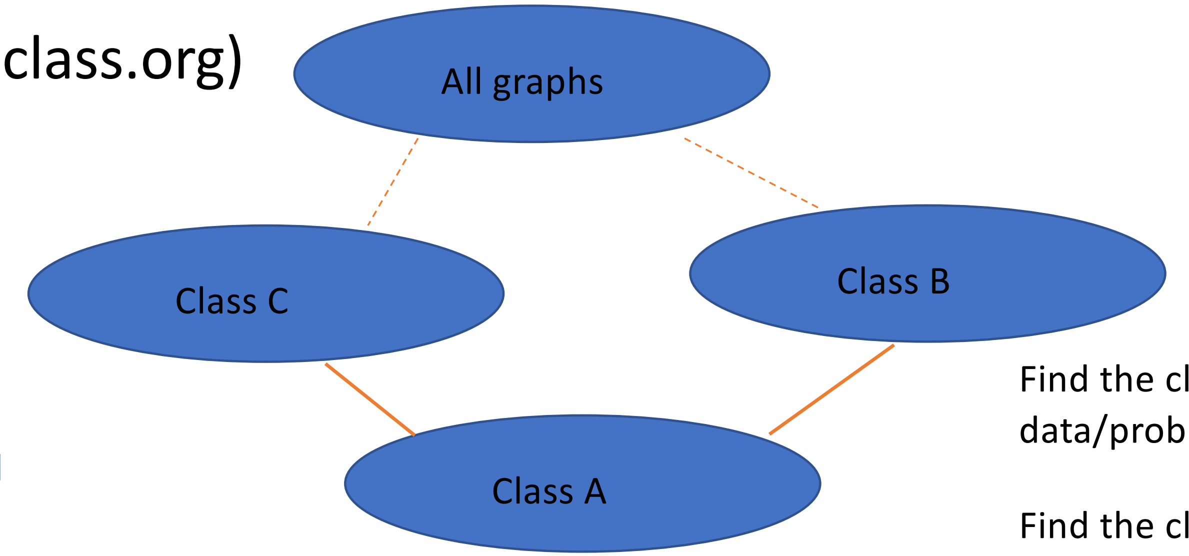
Graph taxonomy

(e.g., graphclass.org)

Class A is smaller than class C or B

But inherits all the good properties of C or B

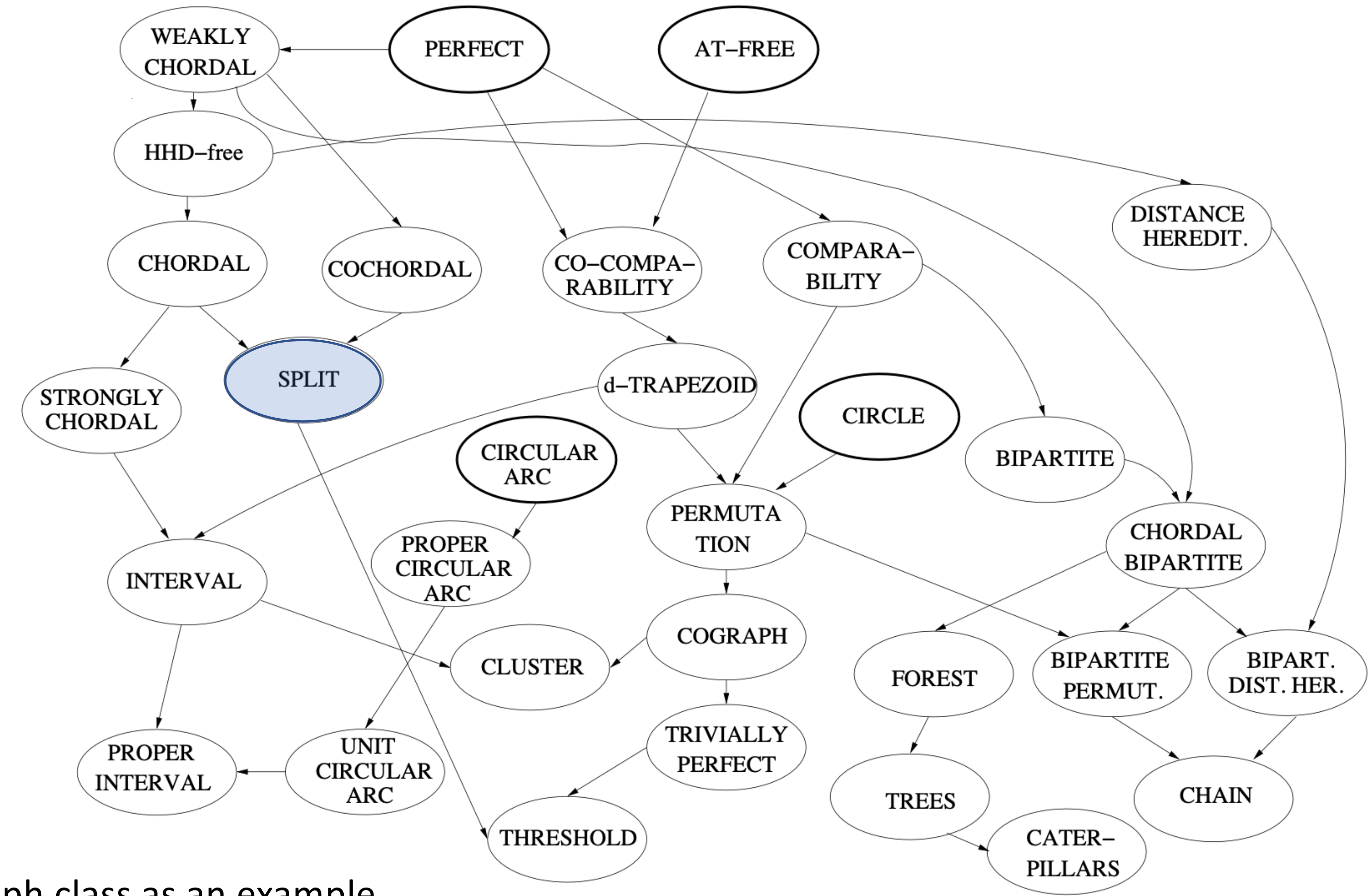
Plus, may have extra good properties because of the specialized nature of this class



Find the class that fits all your data/problem instance.

Find the class(es) that nearly fits all your data/problem instances

Find the class(es) that approximately fits some data instances and is VERY NICE



From Mancini thesis

Take split graph class as an example.
Tracing upwards, we can see every split graph is a chordal graph – but not visa versa. Continuing up, every chordal graph is a perfect graph…..etc.
Going downwards – we can see that *some* split graphs are threshold graphs (have extra properties that the “other” split graphs don’t have. Of course, all threshold graphs are split graphs.
The above is a TINY part of the of “atlas” of graph classes.

Graphclass: perfect

Definition:

A graph is perfect if for all induced subgraphs H: \chi(H) = \omega(H), where \chi is the chromatic number and \omega is the size of a maximum clique.

Unweighted problems

3-Colourability [?]	Polynomial	[+]Details
Clique [?]	Polynomial	[+]Details
Clique cover [?]	Polynomial	[+]Details
Colourability [?]	Polynomial	[+]Details
Domination [?]	NP-complete	[+]Details
Feedback vertex set [?]	NP-complete	[+]Details
Graph isomorphism [?]	GI-complete	[+]Details
Hamiltonian cycle [?]	NP-complete	[+]Details
Hamiltonian path [?]	NP-complete	[+]Details
Independent dominating set [?]	NP-complete	[+]Details
Independent set [?]	Polynomial	[+]Details
Maximum cut [?]	NP-complete	[+]Details
Monopolarity [?]	Unknown to ISGCI	[+]Details
Polarity [?]	Unknown to ISGCI	[+]Details
Recognition [?]	Polynomial	[+]Details

Weighted problems

Weighted clique [?]	Polynomial	[+]Details
Weighted feedback vertex set [?]	NP-complete	[+]Details
Weighted independent dominating set [?]	NP-complete	[+]Details
Weighted independent set [?]	Polynomial	[+]Details
Weighted maximum cut [?]	NP-complete	[+]Details

Graphclass: perfect

Definition:

A graph is perfect if for all induced subgraphs H: \chi(H) = \omega(H), where \chi is the chromatic number and \omega is the size of a maximum clique.

Unweighted problems

3-Colourability [?]	Polynomial	[+]Details
Clique [?]	Polynomial	[+]Details
Clique cover [?]	Polynomial	[+]Details
Colourability [?]	Polynomial	[+]Details
Domination [?]	NP-complete	[+]Details
Feedback vertex set [?]	NP-complete	[+]Details
Graph isomorphism [?]	GI-complete	[+]Details
Hamiltonian cycle [?]	NP-complete	[+]Details
Hamiltonian path [?]	NP-complete	[+]Details
Independent dominating set [?]	NP-complete	[+]Details
Independent set [?]	Polynomial	[+]Details
Maximum cut [?]	NP-complete	[+]Details
Monopolarity [?]	Unknown to ISGCI	[+]Details
Polarity [?]	Unknown to ISGCI	[+]Details
Recognition [?]	Polynomial	[+]Details

Weighted problems

Weighted clique [?]	Polynomial	[+]Details
Weighted feedback vertex set [?]	NP-complete	[+]Details
Weighted independent dominating set [?]	NP-complete	[+]Details
Weighted independent set [?]	Polynomial	[+]Details
Weighted maximum cut [?]	NP-complete	[+]Details

Graphclass: split

Definition:

A graph is a split graph if it can be partitioned in an independent set and a clique.

Unweighted problems

3-Colourability [?]	Linear	[+]Details
Clique [?]	Linear	[+]Details
Clique cover [?]	Polynomial	[+]Details
Colourability [?]	Linear	[+]Details
Domination [?]	NP-complete	[+]Details
Feedback vertex set [?]	Polynomial	[+]Details
Graph isomorphism [?]	GI-complete	[+]Details
Hamiltonian cycle [?]	NP-complete	[+]Details
Hamiltonian path [?]	NP-complete	[+]Details
Independent dominating set [?]	Linear	[+]Details
Independent set [?]	Linear	[+]Details
Maximum cut [?]	NP-complete	[+]Details
Monopolarity [?]	Linear	[+]Details
Polarity [?]	Linear	[+]Details
Recognition [?]	Linear	[+]Details

Weighted problems

Weighted clique [?]	Polynomial	[+]Details
Weighted feedback vertex set [?]	Polynomial	[+]Details
Weighted independent set [?]	Linear	[+]Details
Weighted maximum cut [?]	NP-complete	[+]Details

Wouldn’t it be nice if your data/instances were split graphs?

- Or some large number of your data instances…
- Or some identifiable and “important” set of your problem instances…
- Or if you’re a useful/important set of your data instances were close (in some sense) to split graphs…
- Well some of my interests are 

Graphclass: split

Definition:

A graph is a split graph if it can be partitioned in an independent set and a clique.

Unweighted problems

3-Colourability [?]	Linear	[+]Details
Clique [?]	Linear	[+]Details
Clique cover [?]	Polynomial	[+]Details
Colourability [?]	Linear	[+]Details
Domination [?]	NP-complete	[+]Details
Feedback vertex set [?]	Polynomial	[+]Details
Graph isomorphism [?]	GI-complete	[+]Details
Hamiltonian cycle [?]	NP-complete	[+]Details
Hamiltonian path [?]	NP-complete	[+]Details
Independent dominating set [?]	Linear	[+]Details
Independent set [?]	Linear	[+]Details
Maximum cut [?]	NP-complete	[+]Details
Monopolarity [?]	Linear	[+]Details
Polarity [?]	Linear	[+]Details
Recognition [?]	Linear	[+]Details

Weighted problems

Weighted clique [?]	Polynomial	[+]Details
Weighted feedback vertex set [?]	Polynomial	[+]Details
Weighted independent set [?]	Linear	[+]Details
Weighted maximum cut [?]	NP-complete	[+]Details

Graphclass: threshold

Unweighted problems

3-Colourability [?]	Linear	[+]Details
Clique [?]	Linear	[+]Details
Clique cover [?]	Linear	[+]Details
Colourability [?]	Linear	[+]Details
Domination [?]	Linear	[+]Details
Feedback vertex set [?]	Linear	[+]Details
Graph isomorphism [?]	Linear	[+]Details
Hamiltonian cycle [?]	Linear	[+]Details
Hamiltonian path [?]	Linear	[+]Details
Independent dominating set [?]	Linear	[+]Details
Independent set [?]	Linear	[+]Details
Maximum cut [?]	Polynomial	[+]Details
Monopolarity [?]	Linear	[+]Details
Polarity [?]	Linear	[+]Details
Recognition [?]	Linear	[+]Details

Weighted problems

Weighted clique [?]	Linear	[+]Details
Weighted feedback vertex set [?]	Linear	[+]Details
Weighted independent dominating set [?]	Linear	[+]Details
Weighted independent set [?]	Linear	[+]Details
Weighted maximum cut [?]	NP-complete	[+]Details

But….``not so fast’’…linear in what? Also cost of constructing /reading graph?

Sometimes you don’t have the graph itself……..cost of deriving the graph….

Moreover, maybe linear **in #edges** which is typically **quadratic** in **#vertices**….unless sparse graph….

Yet even with those caveats, if you can show your data/problem instances are in the “special” class with “more nice properties” then maybe you can still devise more efficient algorithms than is currently known (because no-one else has realized these problem instances come from that special class).

Or maybe YOUR data/situation comes with even special “powers” in addition to the graph class/structure….

E.g., a cheap oracle for X (where a group of vertices is a clique or not…for example)

But.... ``not so fast''...these are results for graphs – what about hypergraphs...

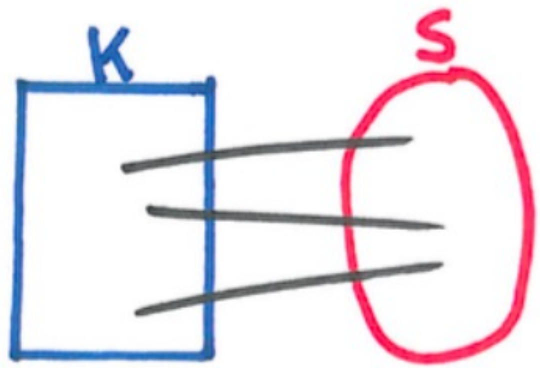
A k-uniform hypergraph is a graph when k=2.

For k>2, generally the situation is more complex and the "attractive properties" don't always hold in the "hypergraph generalization**(s)**"

However the basic principle of course apply – a hierarchy of hypergraph classes with more favourable characteristics (but less data/instance coverage) as one moves down the hierarchy (to more specialized classes).

Definition: A *split graph* is a graph G whose vertex set can be partitioned as $V(G) = K \cup S$ where K is a clique and S is a stable set.

Such a partition is a KS-partition.



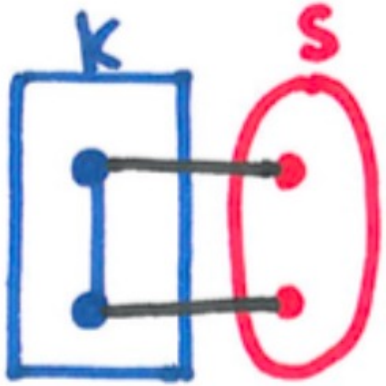
Definition: A KS-partition of a split graph G is *K-max* if $|K| = \omega(G)$ and *S-max* if $|S| = \alpha(G)$.

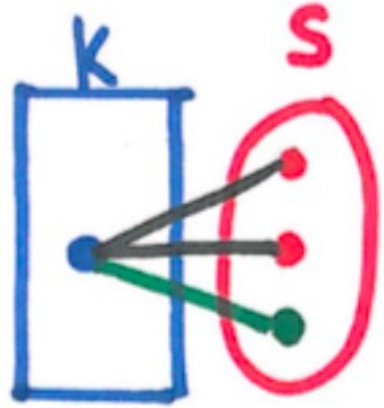
(From Trenk DIMACS talk)

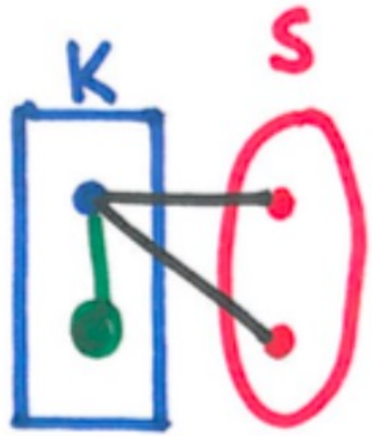
Why might split graphs be of interest?

- MANY reasons (discovered in the 1970's when studying optimization and Linear programming in particular).
- Complex to explain some of these reasons…but look it up if interested!
- For my (our?) purposes – they describe a "perfect community". The vertices (people) in the (maximal)clique are perfectly connected to each other. The vertices (people) not in the clique are not connected to each other (but may be connected to some of the people in the clique). More generally – perfect cluster…
- In MaxCon terms for feasible graph – K are inliers and S are outliers – for the infeasible graph K are outliers and S are inliers.

P_4 $\qquad\qquad$ $K_{1,3}$







P_4 has a unique KS-partition
It is both K-max and S-max.

$K_{1,3}$ has two KS-partitions
One is S-max, the other is K-max.

(From Trenk DIMACS talk)

Two kinds of split graphs

Theorem (Hammer, Simeone: 1977) For any KS-partition of a split graph G, exactly one of the following holds.

1. $|K| = \omega(G)$ and $|S| = \alpha(G)$. (K-max, S-max)
2. $|K| = \omega(G) - 1$ and $|S| = \alpha(G)$. (S-max)
3. $|K| = \omega(G)$ and $|S| = \alpha(G) - 1$. (K-max)

Moreover, in
(1.) the partition is unique, in
(2.) there exists $s \in S$ so that $K \cup \{s\}$ is complete, and in
(3.) there exists $k \in K$ so that $S \cup \{k\}$ is a stable set.

Theorem (Cheng, Collins, Trenk: 2016) Let G be a split graph with degree sequence $d_1 \geq d_2 \geq \cdots \geq d_n$ and let $m = \max\{i : d_i \geq i - 1\}$. Then G is unbalanced if $d_m = m - 1$ and balanced if $d_m > m - 1$.

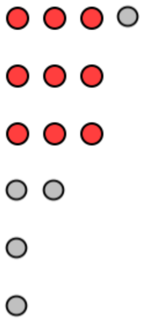
(From Trenk DIMACS talk)

Splittance

- A similar type of calculation on degrees determines whether a graph is split or, when not, yields the "splittance" (distance from being split: number of edges that need to be added or removed to make it a splirt graph)
- Thus checking for splittance is relatively cheap/fast
- (Aside – part of the key to this result and that of balanced/unbalanced) is the "Durfee Square".

Examples [edit]

The partition 4 + 3 + 3 + 2 + 1 + 1:



has a Durfee square of side 3 (in red) because it contains 3 parts that are ≥ 3, but does not contain 4 parts that are ≥ 4. Its Durfee symbol consists of the 2 partitions 1 and 2+1+1.

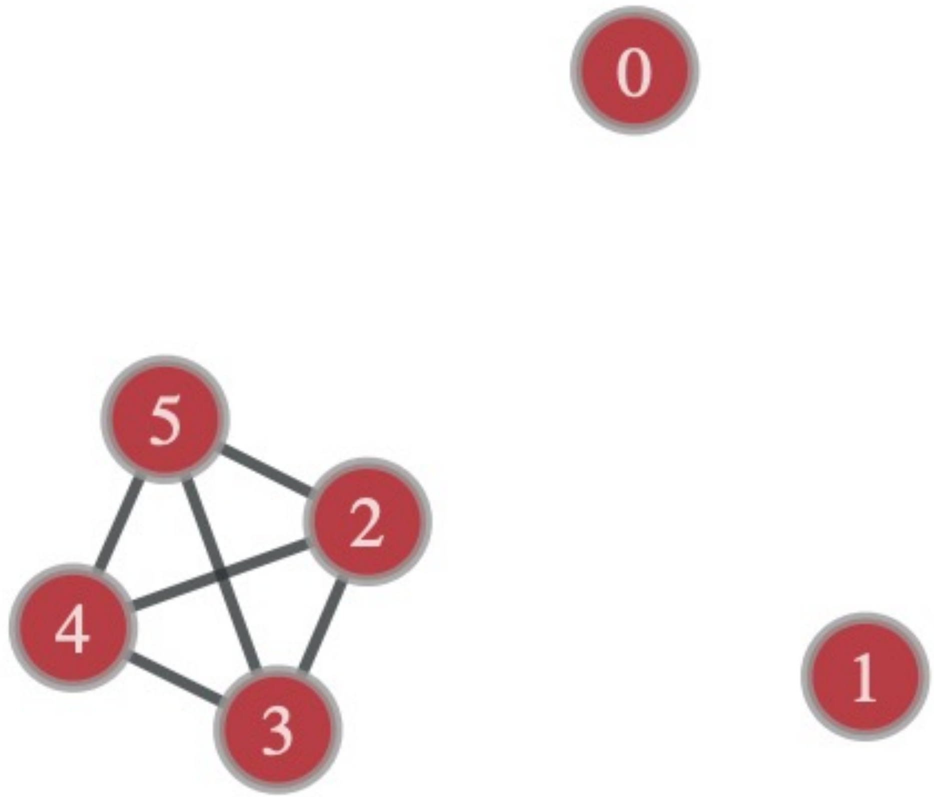
History [edit]

Durfee squares are named after William Pitt Durfee, a student of English mathematician James Joseph Sylvester. In a letter to Ar Cayley in 1883, Sylvester wrote:[4]

> "*Durfee's square is a great invention of the importance of which its author has no conception.*"

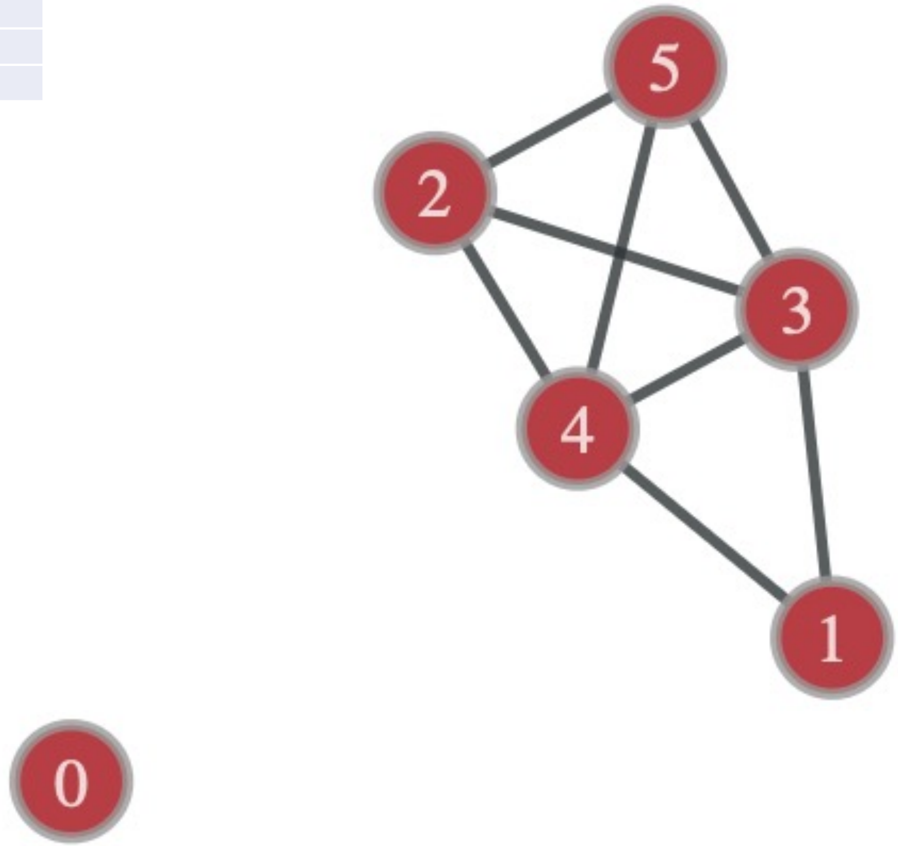
(Wikipedia)

Degree Sequence	3	3	3	3	0	0
i-1	0	1	2	3	4	5
m	3					
m(m-1)	6					
sum_di_to_m	9					
sum_di_m+1_to_n	3					
splittance	0					



“UnBalanced” – all members of clique are swing vertices

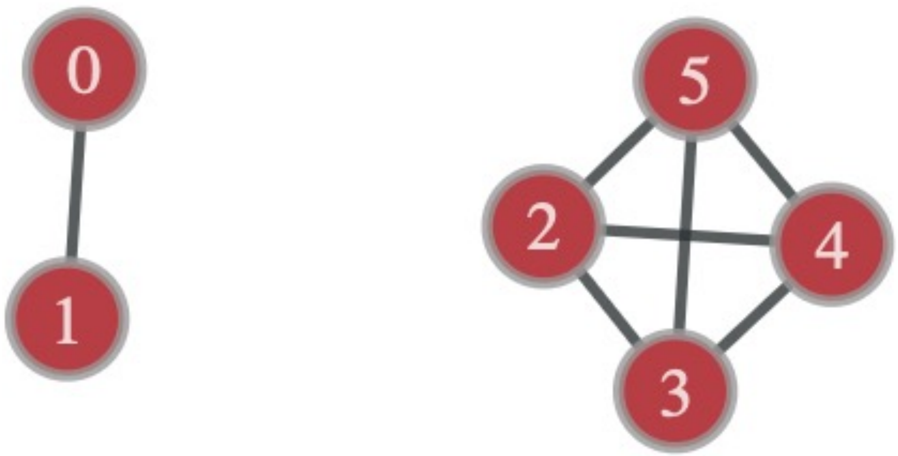
Degree Sequence	4	4	3	3	2	0
i-1	0	1	2	3	4	5
m	3					
m(m-1)	6					
sum_di_to_m	11					
sum_di_m+1_to_n	5					
splittance	0					



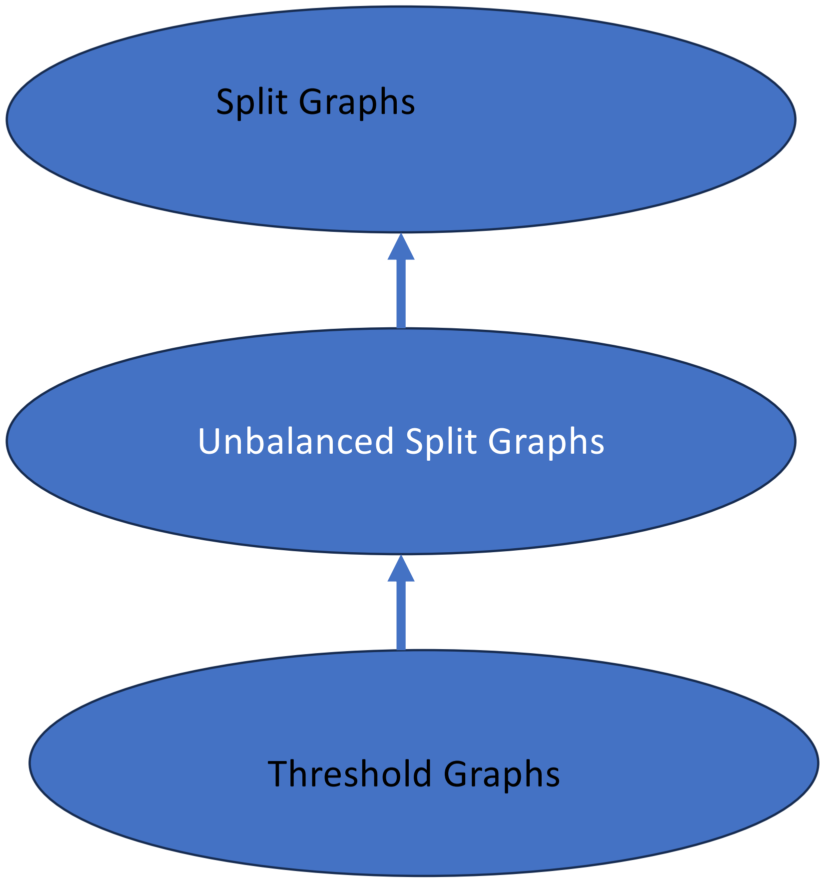
Input

Unbalanced – members 2 and 5 of the clique are swing vertices

Degree Sequence	3	3	3	3	1	1	
i-1	0	1	2	3	4	5	
m	3						
m(m-1)	6						
sum_di_to_m	9						
sum_di_m+1_to_n	5						
splittance	1						



All threshold graphs are unbalanced split graphs (but not visa versa)



{0,1}* constructible graphs

- Threshold graphs have a fun/simple description – one that says how they can be constructed.
- Order your vertices – then, one by one, in order, add these vertices choosing either:
  - Add an isolated vertex (totally unconnected to any previous vertex) – 0 in construction string
  - Add a dominating vertex (totally connected to every previous vertex) – 1 in construction string
- Being {0,1}* constructible is ONE of the many ways to define threshold graphs
- Each way to define threshold graphs leads to a different generalization to hypergraphs – {0,1}* is one of those classes of hypergraphs