Practice of Answer Set Programming

Graph Problems in ASP (I)

Ira A. Fulton Schools of Engineering
Arizona State University

Objectives

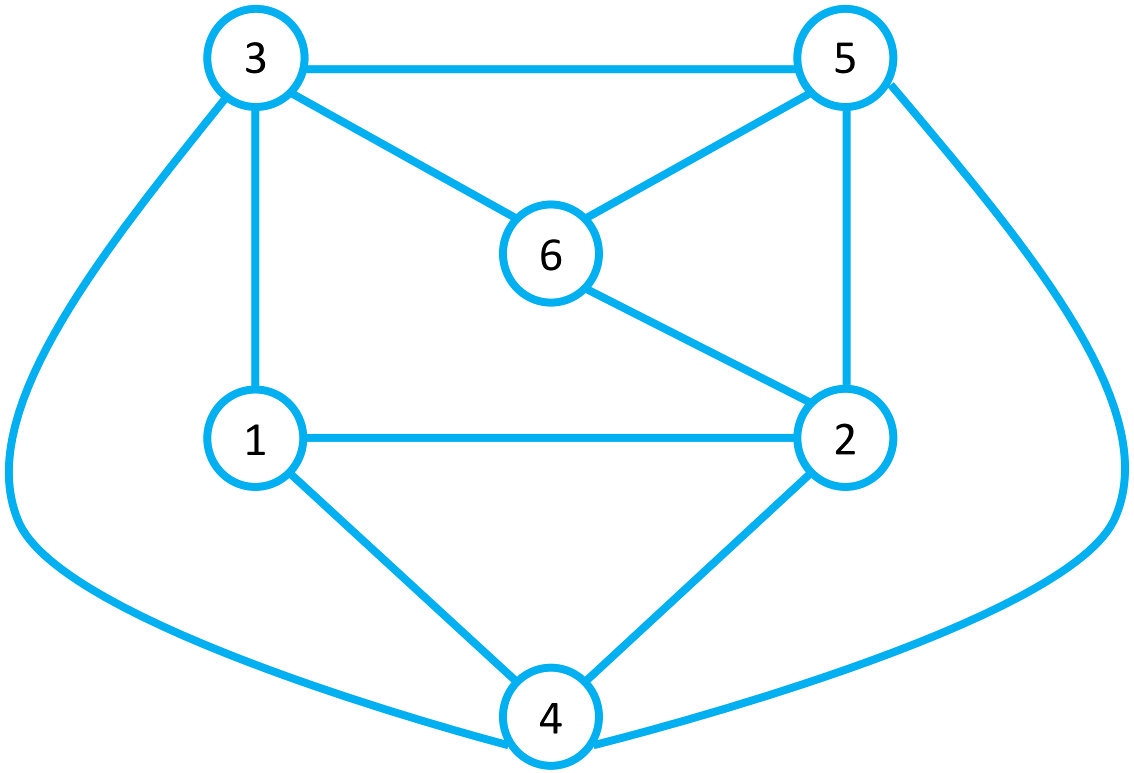
Objective

Use ASP to solve graph-related problems

Graph Coloring Problem

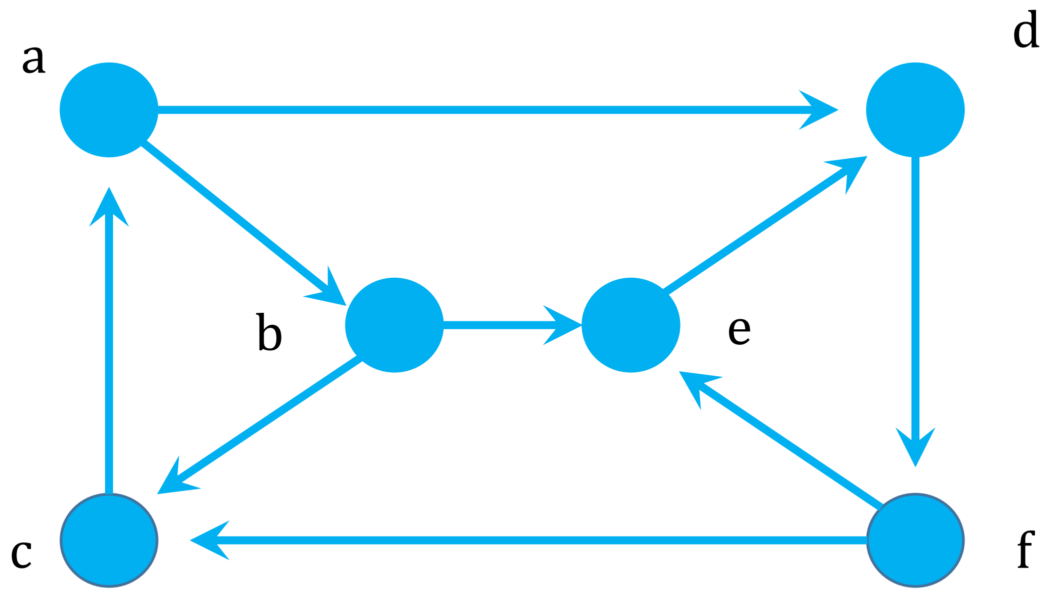
Graph 1

```
% Nodes
vertex(1..6).


% Edges
edge(1,(2;3;4)). edge(4,(1;2)).

edge(2,(4;5;6)). edge(5,(3;4;6)).

edge(3,(1;4;5)). edge(6,(2;3;5)).
```



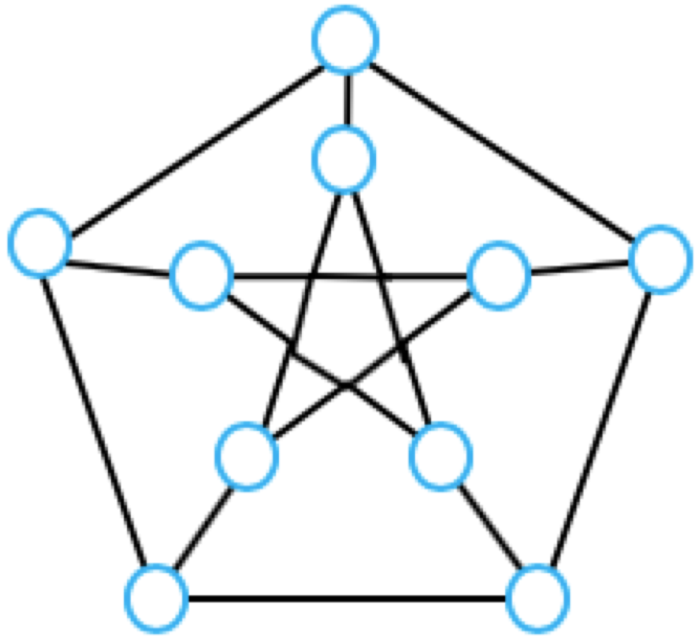
Graph 2

```
vertex(a; b; c; d; e; f).

edge(a,b; b,c; c,a; d,f; f,e; e,d; a,d; f,c; b,e).
```



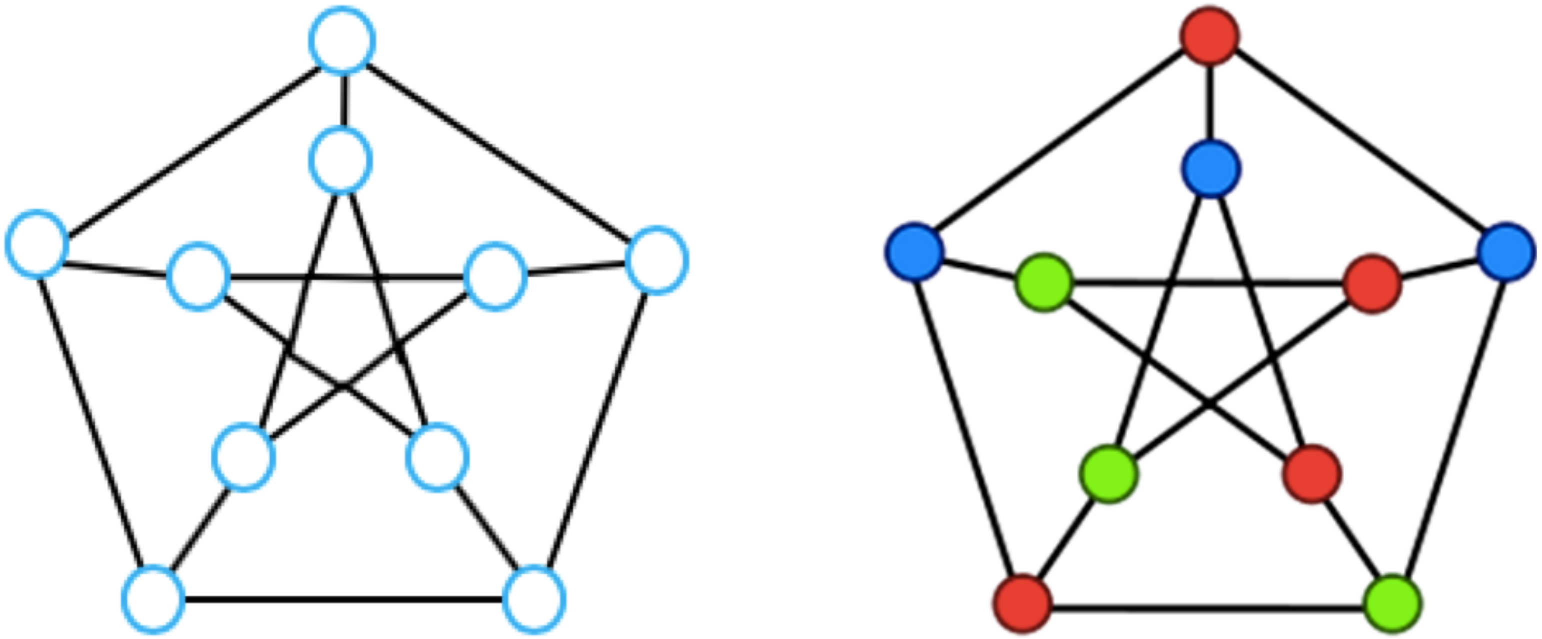
Graph Coloring

An n-coloring of a graph G is a function f from its set of vertices to $\{1, . . . , n\}$ such that $f(x) \neq f(y)$ for every pair of adjacent vertices x, y.



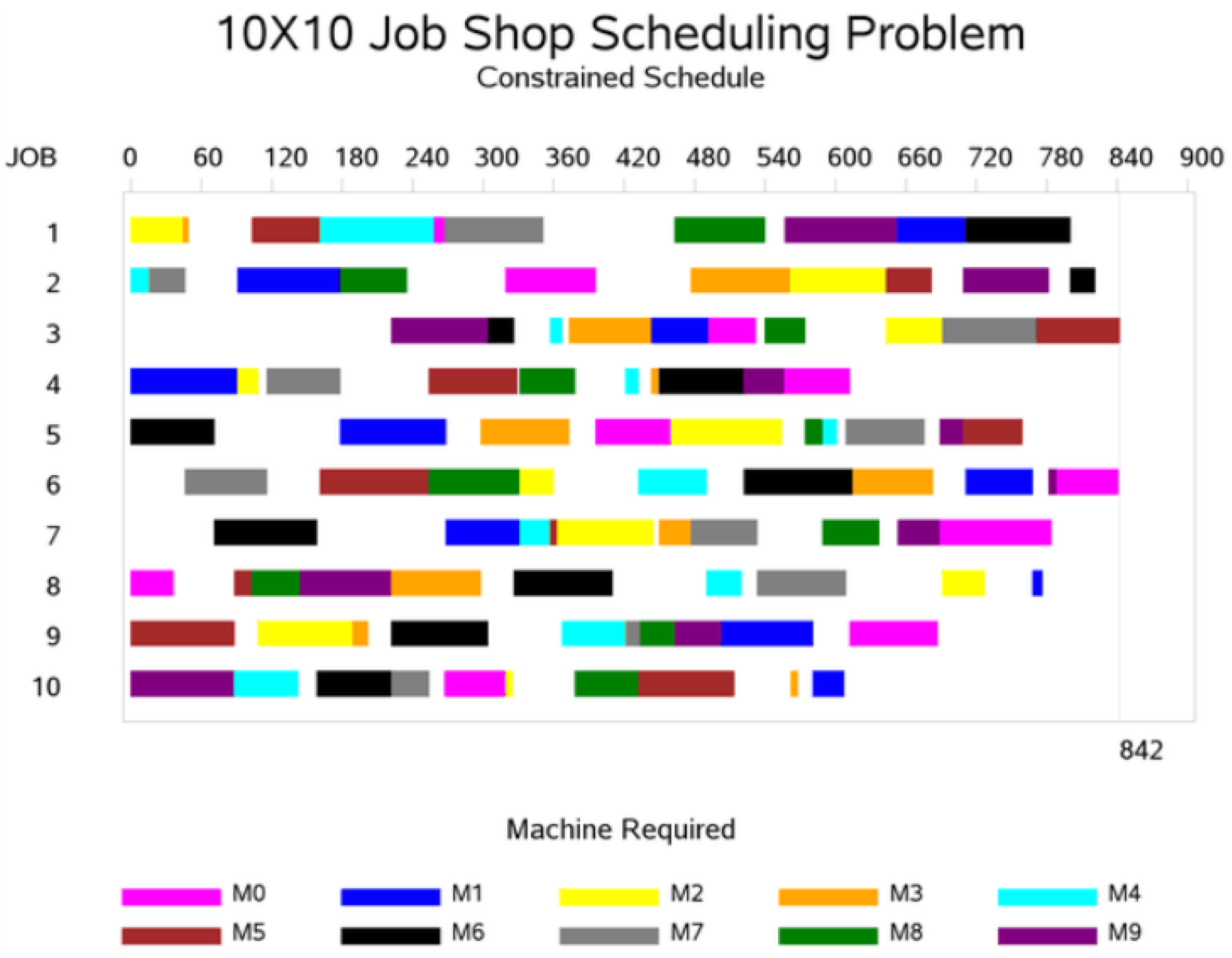
Graph Coloring

An n-coloring of a graph G is a function f from its set of vertices to $\{1, . . . , n\}$ such that $f(x) \neq f(y)$ for every pair of adjacent vertices x, y.



Graph Coloring Application: Job Scheduling

- A given set of jobs need to be assigned to time slots, each job requires one such slot.
- Jobs can be scheduled in any order, but pairs of jobs may be in *conflict* in the sense that they may not be assigned to the same time slot, for example because they both rely on a shared resource.
- The corresponding graph contains a vertex for every job and an edge for every conflicting pair of jobs.



http://support.sas.com/documentation/cdl/en/orcpug/66101/HTML/default/viewer.htm#orcpug_clp_examples14.htm

Graph Coloring Application: Exam Scheduling

- **Suppose we want to make an exam schedule for a university. We have a list of different subjects and students enrolled in every subject.**
  - *How do we schedule the exam so that no two exams with a common student are scheduled at same time?*
  - *How many minimum time slots are needed to schedule all exams?*
- **This problem can be represented as a graph where every vertex is a subject and an edge between two vertices mean there is a common student.**

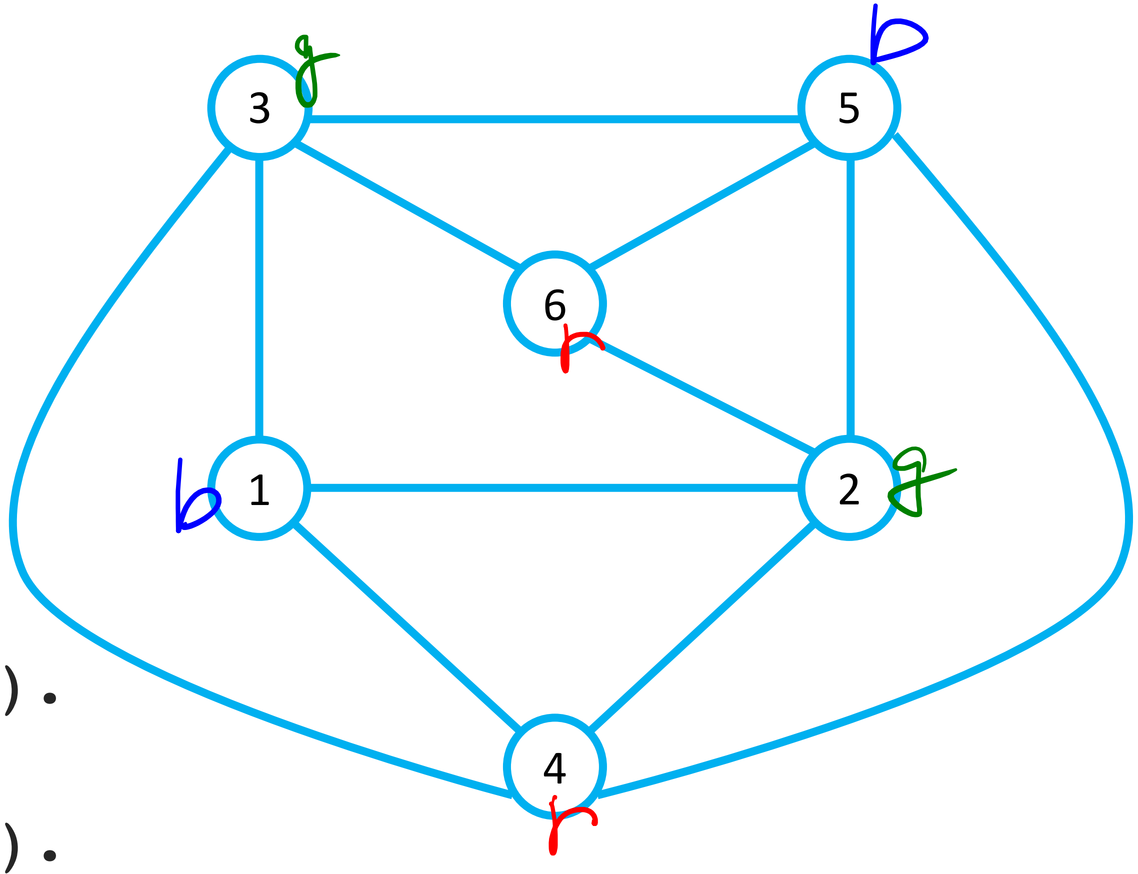
Graph Coloring in ASP

- **The stable models of the following program are in a 1–1 correspondence with the n-colorings**

```
% File 'color'

% each vertex is mapped to a color
{color(X,C): C=1..n}=1 :- vertex(X).

% the adjacent vertices should be of different color
:- edge(X,Y), color(X,C), color(Y,C).
```

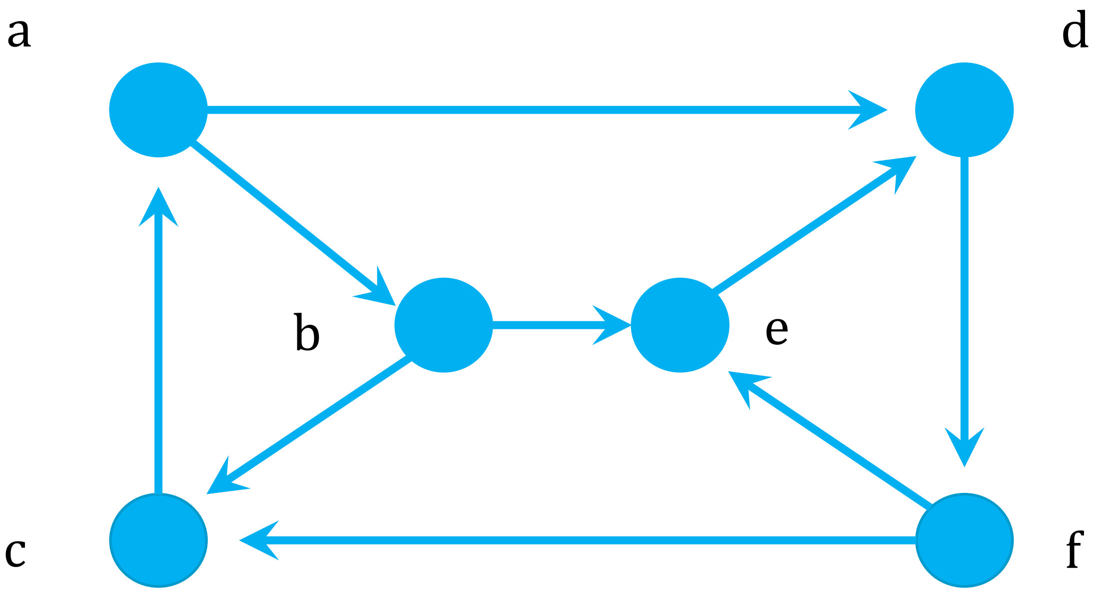
Graph 1

```
% Nodes
vertex(1..6).

% Edges
edge(1,(2;3;4)). edge(4,(1;2)).

edge(2,(4;5;6)). edge(5,(3;4;6)).

edge(3,(1;4;5)). edge(6,(2;3;5)).
```



Graph 2

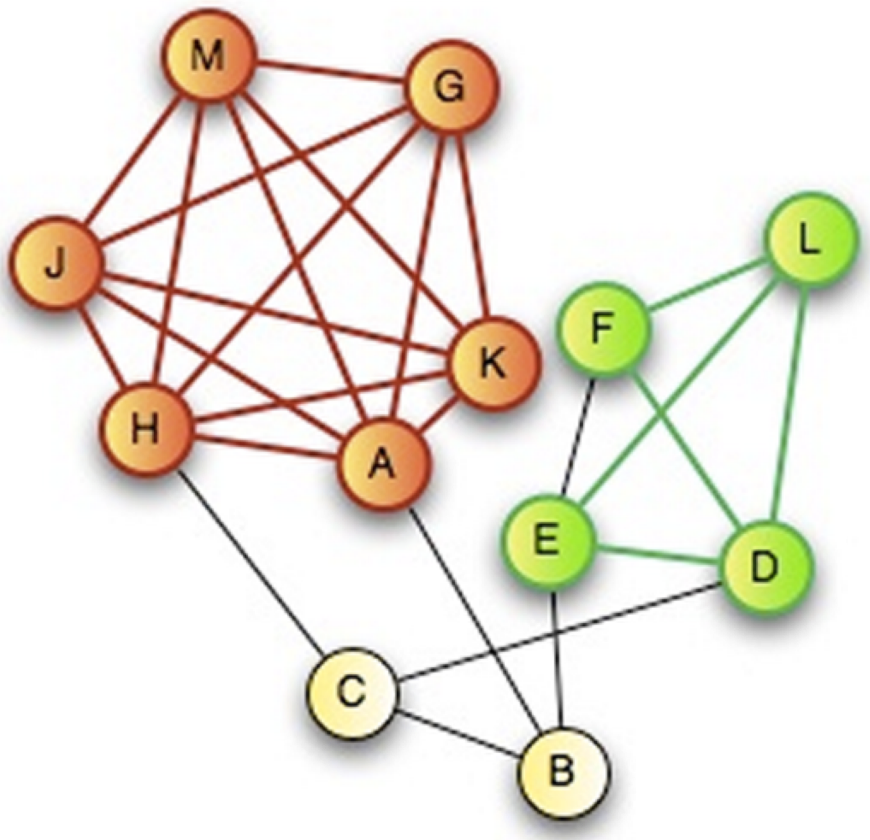
```
vertex(a; b; c; d; e; f).

edge(a,b; b,c; c,a; d,f; f,e; e,d; a,d; f,c; b,e).
```



Clique

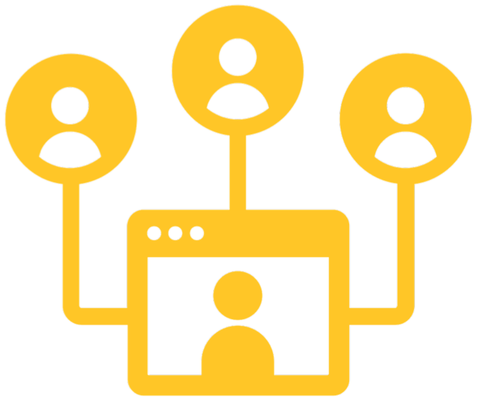
Definition

- A **clique** in a graph G is a set of pairwise adjacent vertices of G



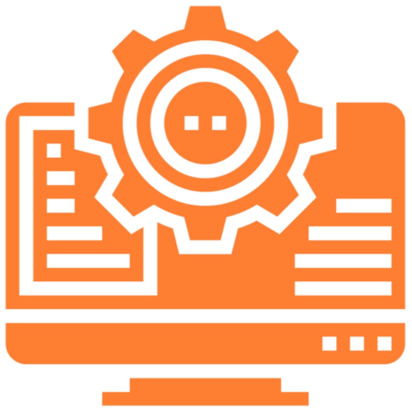
Application of (Maximum) Clique – 1 of 2

- Social network analysis: “Clique models idealize three important structural properties that are expected of a cohesive subgroup, namely, familiarity (each vertex has many neighbors and only a few strangers in the group), reachability (a low diameter, facilitating fast communication between the group members) and robustness (high connectivity, making it difficult to destroy the group by removing members).”
- Patterns in telecommunications traffic

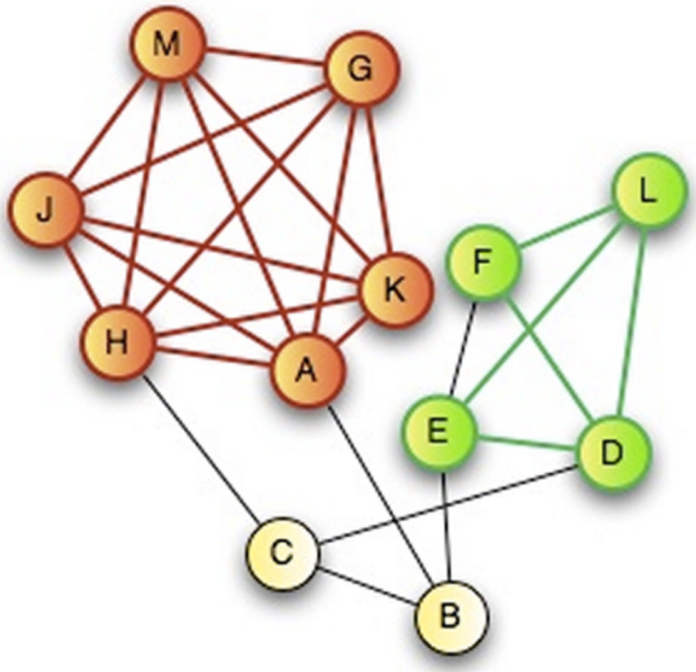


Application of (Maximum) Clique – 2 of 2

- Bioinformatics: “identify common substructures between a collection of molecules known to possess certain pharmacological properties”
- Fault diagnosis on large multiprocessor systems
- Computer vision and pattern recognition



Cliques in ASP

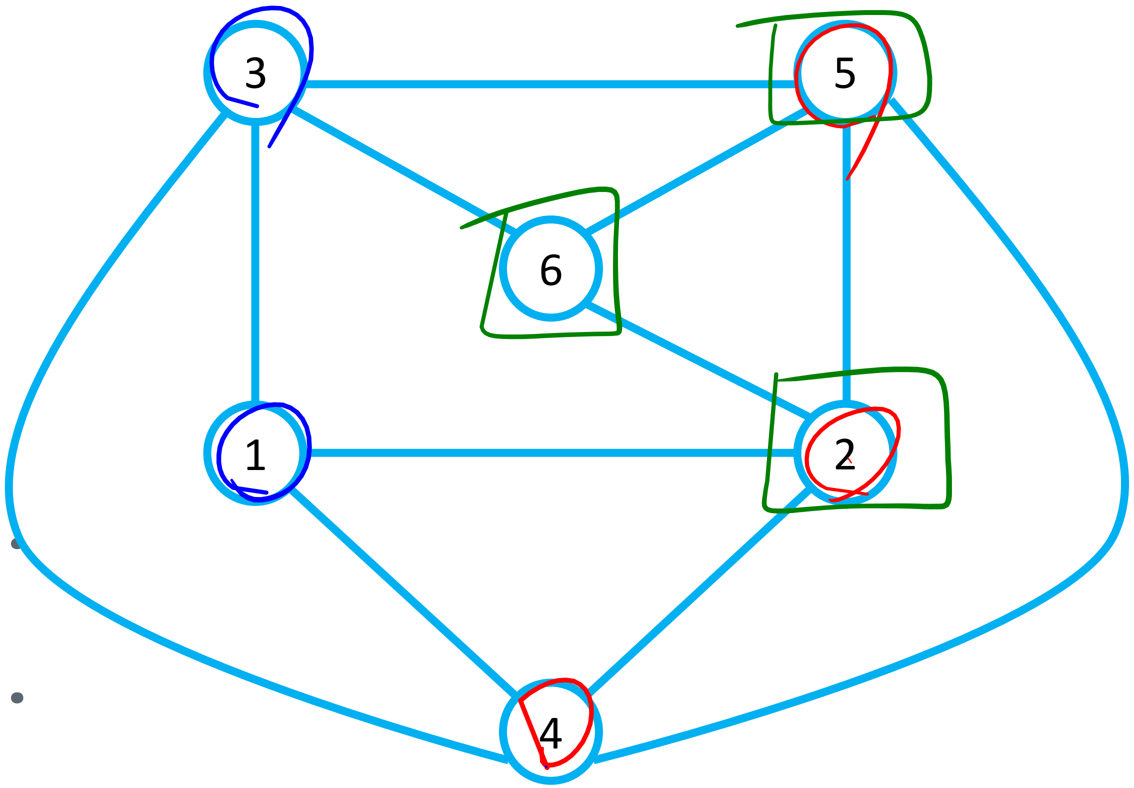


The stable models of the following program are in a 1–1 correspondence with cliques of cardinalities n.

```
% File 'clique'

% in/1 is a set consisting of n vertices of G
{in(X): vertex(X)} = n.

% there must be an edge between the chosen vertices
:- in(X), in(Y), X!=Y, not edge(X,Y), not edge(Y,X).
```

Graph 1

```
% Nodes
vertex(1..6).

% Edges
edge(1,(2;3;4)). edge(4,(1;2)).

edge(2,(4;5;6)). edge(5,(3;4;6)).

edge(3,(1;4;5)). edge(6,(2;3;5)).
```



Wrap-Up