UCS1524 – Logic Programming

Problem Reduction

And/Or Graph

Session Meta Data

Author	**Dr. D. Thenmozhi**
Reviewer	
Version Number	1.2
Release Date	22 October 2022

Session Objectives

- Understanding problem reduction in Prolog.
- Learn about And /Or graph for problem reduction.

Session Outcomes

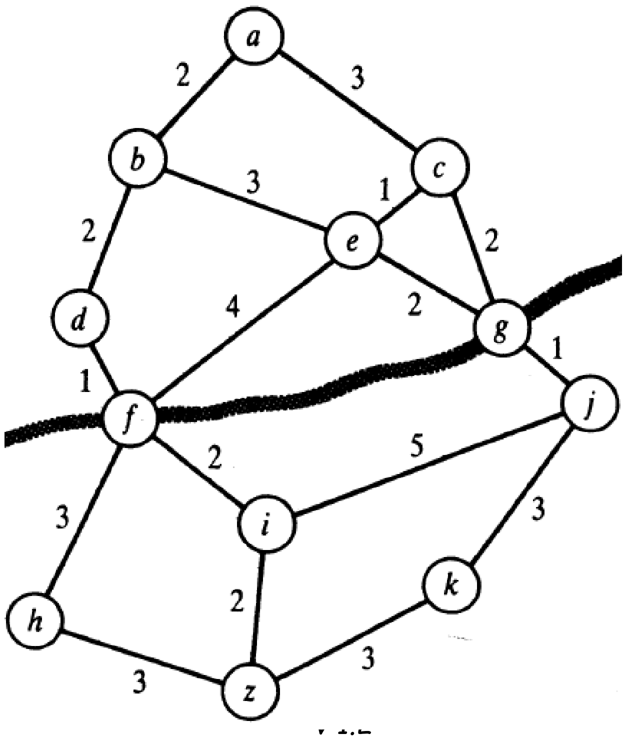
- At the end of this session, participants will be able to
  - Apply And/Or graph for problem reduction in Prolog.

Agenda

- Problem reduction
  - And/Or graph representation
  - Solution tree
  - Search in And/Or graph
  - Depth-first And/Or procedure

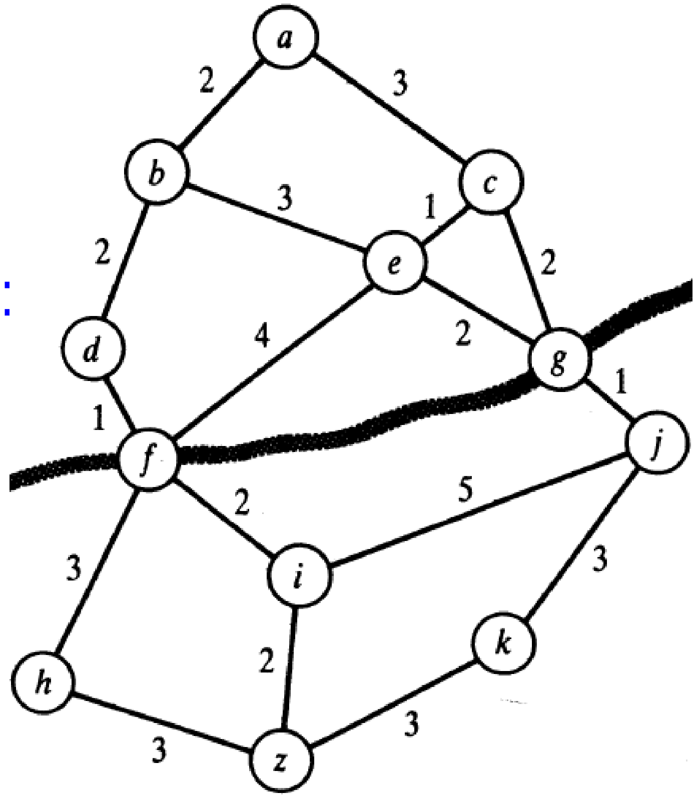
AND/OR graph representation of problems

- Technique for solving problems that can be decomposed into sub problems
- Finding a route from a to z in a road map. The river has to be crossed at f or g. An AND/OR representation of this problem



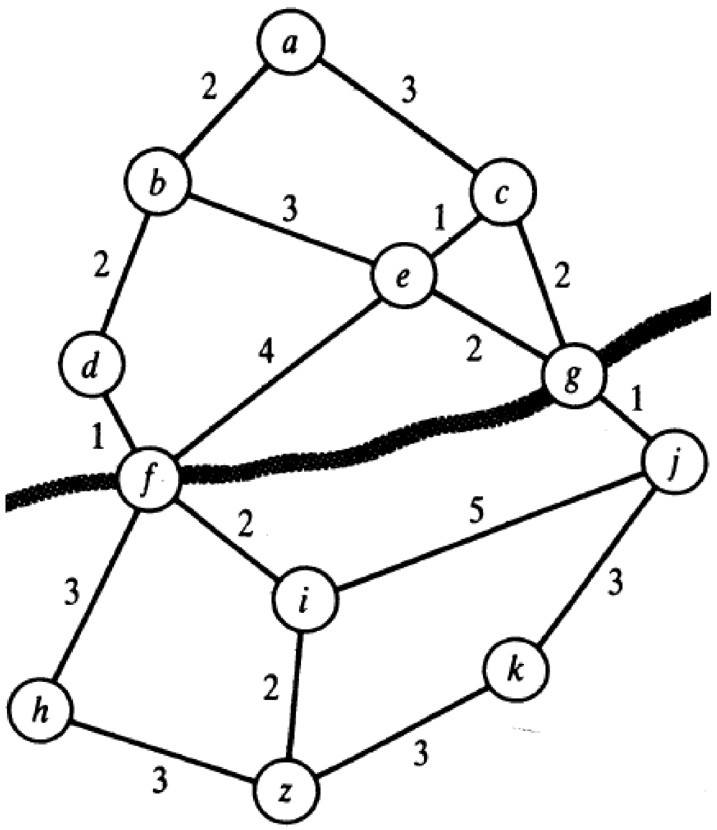
AND/OR graph representation of problems

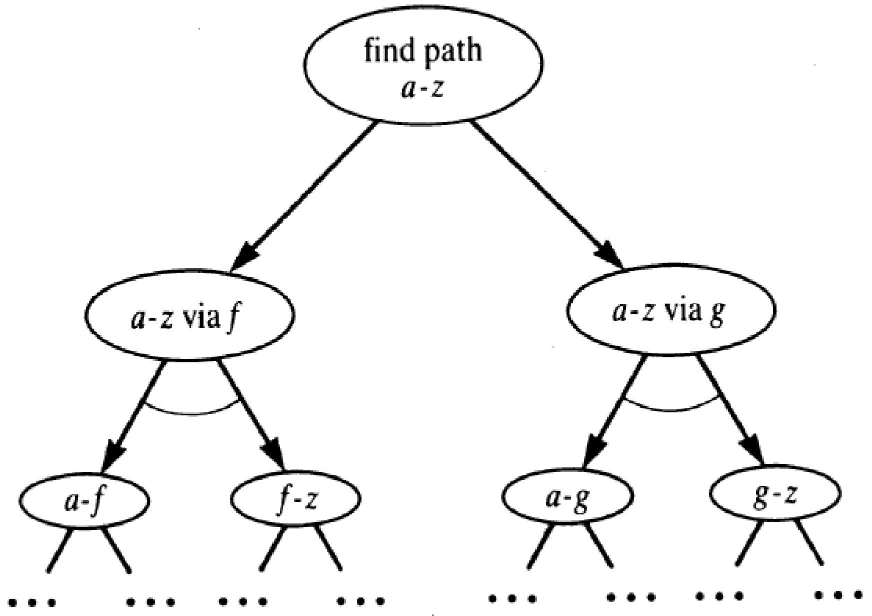
- This problem can be decomposed into
- (1) To find a path from a to z via f:
  - 1.1 find a path from a to f, and
  - I. 2 find a path from f to z
- (2) To find a path from a to z via g:
  - 2.1 find a path from a to g, and
  - 2.2 find a path from g to z.



AND/OR graph representation

- An AND/OR representation of the route-finding problem



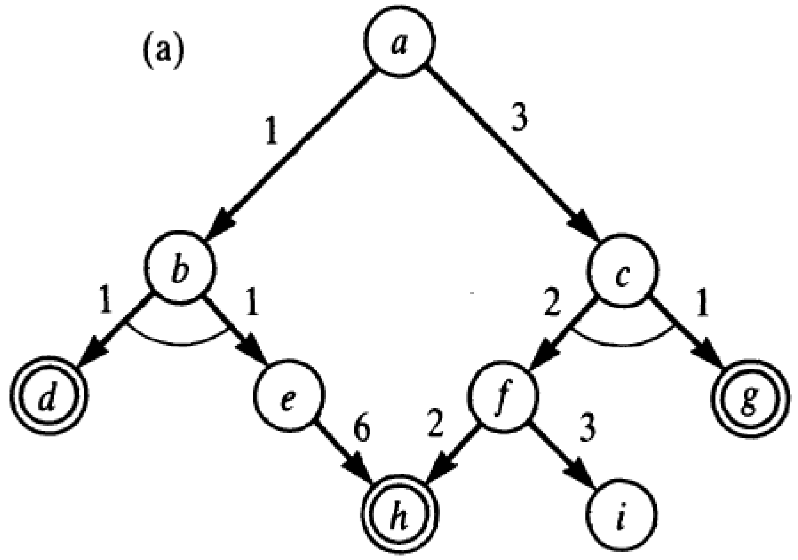


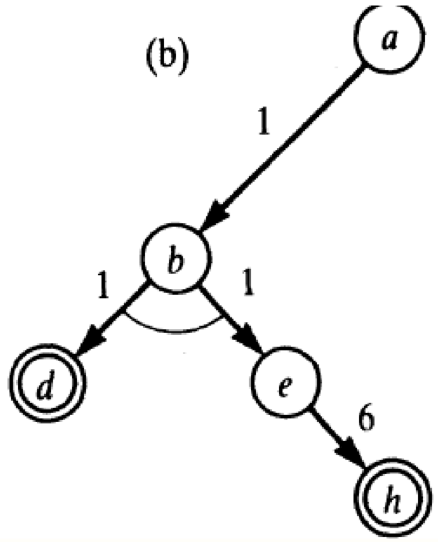
Curved arc indicates that all sub problems have to be solved

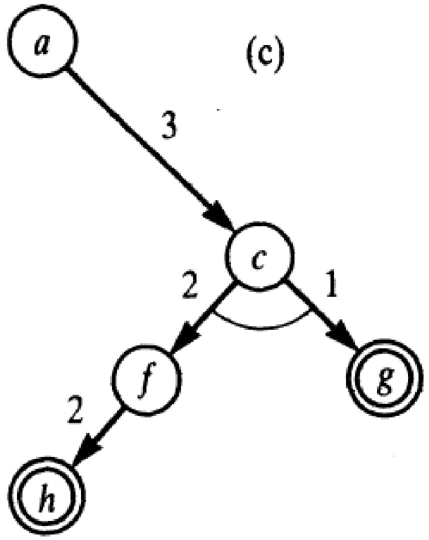
Solution Tree

- In the state-space representation, a solution to the problem was a path in the state space.
- In the AND/OR representation, a solution, of course, has to include all the sub problems of an AND node.
- Therefore the solution is not a path any more, but it is a tree. Such a solution tree, T, defined as follows:
  - the original problem, P, is the root node of T;
  - if P is an OR node then exactly one of its successors graph), together with its own solution tree, is in T;
  - if P is an AND node then all of its successors (in the together with their solution trees, are in T.

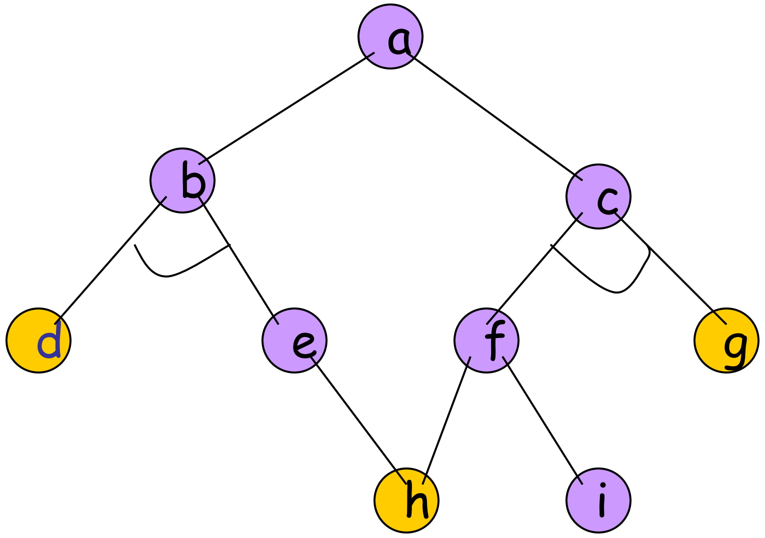
Solution Tree







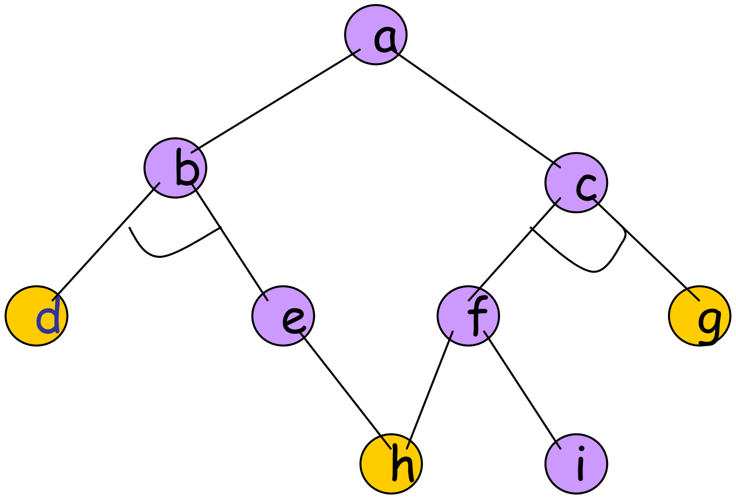
Search in and-or graphs



Search in And/Or graphs

- Use Prolog's own search mechanism
  - Only get answer yes or no, not solution tree.
  - Hard to extend to use cost as well
  - Infinite loop if there is a cycle

```
a :- b.
a :- c.
b :- d, e.
e :- h.
c :- f, g.
f :- h.
f :- i.
d.
g.
h.
```



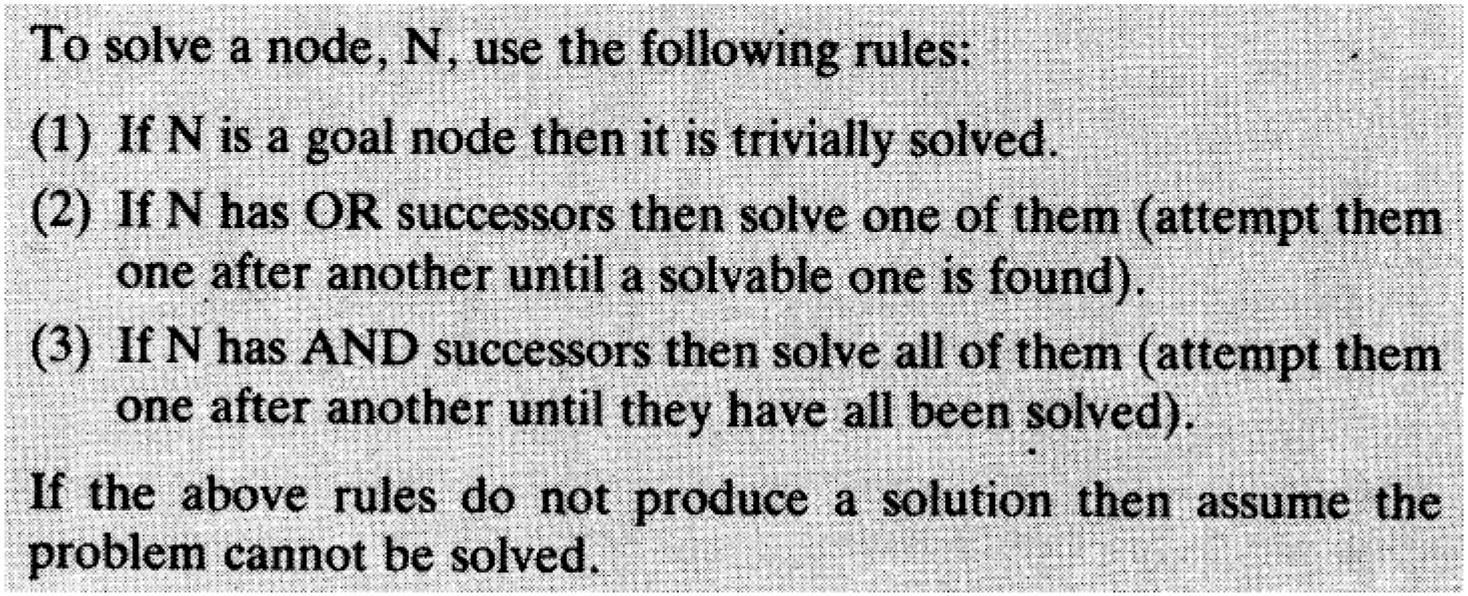
To ask whether problem **a** can be solved we can simply ask: ?- a.

Search in And/Or graphs

- Binary relation representation

```
:- op(600, xfx, --->).
:- op(500, xfx, :).
a ---> or : [b,c].
b ---> and : [d,e].
c ---> and : [f,g].
e ---> or : [h].
f ---> or : [h,i].
goal( d).
goal( g).
goal( h).
```

Depth-first And/Or procedure



To solve a node, N, use the following rules:

(1) If N is a goal node then it is trivially solved.

(2) If N has OR successors then solve one of them (attempt them one after another until a solvable one is found).

(3) If N has AND successors then solve all of them (attempt them one after another until they have all been solved).

If the above rules do not produce a solution then assume the problem cannot be solved.

- Disadvantage:
  - Does not produce a solution tree
  - Susceptible to infinite loops

Depth-first And/Or procedure

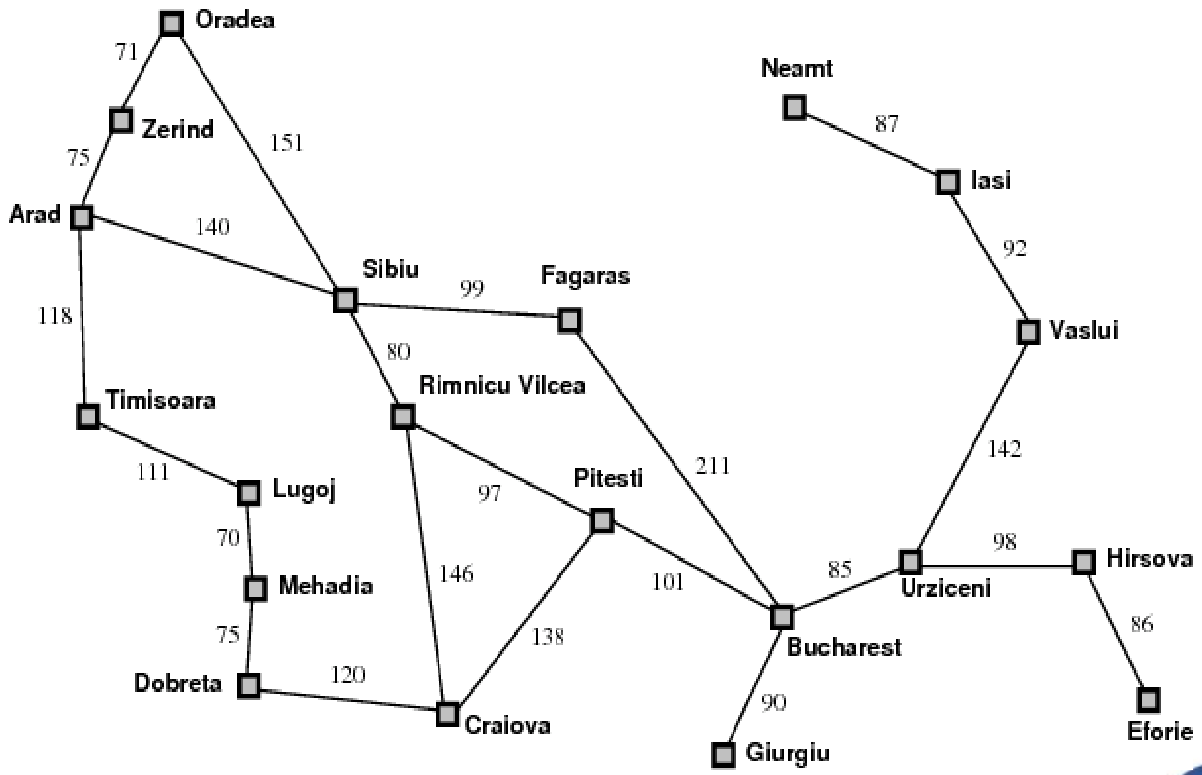
- solve(Node) :- goal(Node).
- solve(Node) :- Node ---> or : Nodes, member(Node1, Nodes), solve(Node1).
- solve(Node) :- Node ---> and : Nodes, solveall(Nodes).
- solveall([ ]).
- solveall([Node | Nodes]) :- solve(Node), solveall(Nodes).

Summary

- Problem reduction
  - And/Or graph representation
  - Solution tree
  - Search in And/Or graph
  - Depth-first And/Or procedure

Check your understanding

- Draw the solution trees of And/Or graph for the route planner to travel from Arad to Brucharest


Oradea
71
Zerind
151
75
Neamt
87
Iasi
Arad
140
92
Sibiu
99
Fagaras
118
Vaslui
80
Rimnicu Vilcea
Timisoara
142
211
111
Pitesti
97
Lugoj
98
70
146
85
101
Urziceni
Hirsova
Mehadia
86
138
Bucharest
75
120
Dobreta
90
Craiova
Eforie
Giurgiu