

Exercises: Artificial Intelligence

The farmer, fox, goose and grain

Representation

- States of the form $[\mathcal{L}|\mathcal{R}]$, where:
 - \mathcal{L} : *Items on left bank*
 - \mathcal{R} : *Items on right bank*
- \mathcal{L} and \mathcal{R} contain:
 - Fa: *Farmer*
 - Fo: *Fox*
 - Go: *Goose*
 - Gr: *Grain*

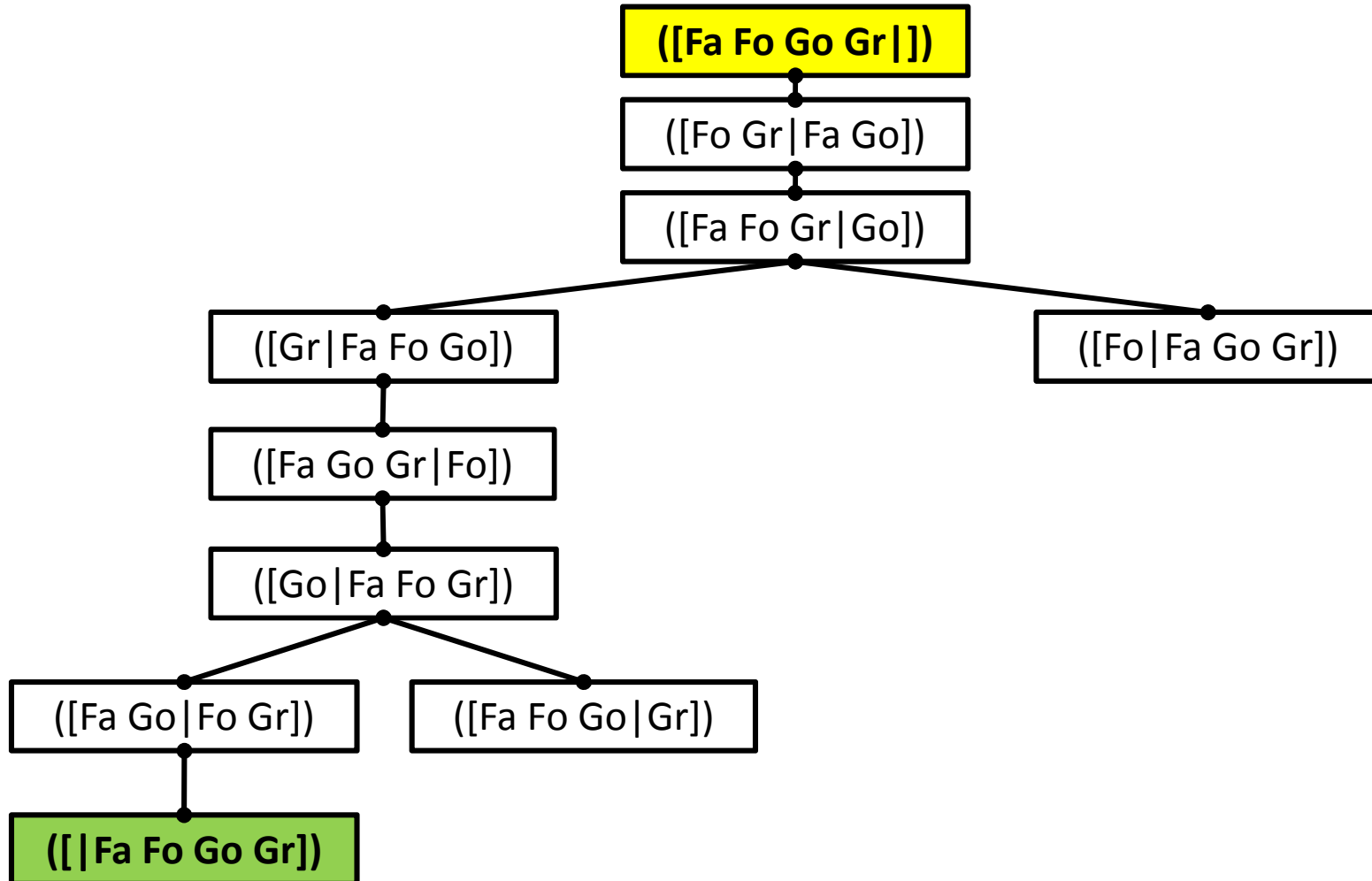
Representation

- Start: [Fa Fo Go Gr |]
- Goal: [| Fa Fo Go Gr]
- Rules:
 - $R_1: [Fa \mathcal{X} | \mathcal{Y}] \longrightarrow [\mathcal{X} | Fa \mathcal{Y}]$
 - $R_2: [\mathcal{X} | Fa \mathcal{Y}] \longrightarrow [Fa \mathcal{X} | \mathcal{Y}]$
 - $R_3: [Fa z \mathcal{X} | \mathcal{Y}] \longrightarrow [\mathcal{X} | Fa z \mathcal{Y}]$
 - $R_4: [\mathcal{X} | Fa z \mathcal{Y}] \longrightarrow [Fa z \mathcal{X} | \mathcal{Y}]$
 - No combination (Fo,Go) or (Go,Gr) on either bank, without the farmer.

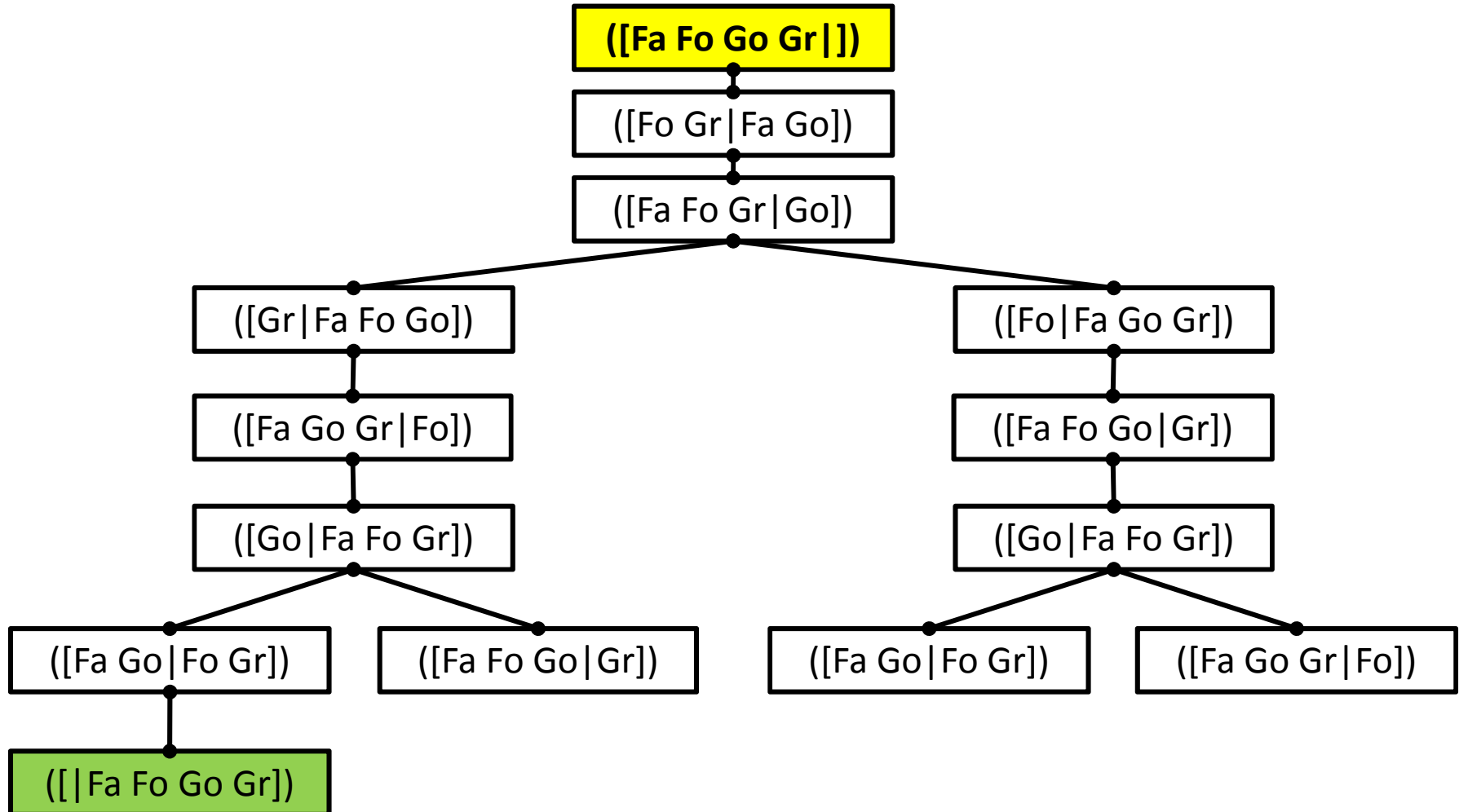
Depth-first search (queues)

- $S = (\langle [Fa\ Fo\ Go\ Gr\] \rangle)$
- $Q_1 = (\langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] \rangle)$
- $Q_2 = (\langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] \rangle)$
- $Q_3 = (\langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Gr\ | Fa\ Fo\ Go] \rangle, \langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Fo\ | Fa\ Go\ Gr] \rangle)$
- $Q_4 = (\langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Gr\ | Fa\ Fo\ Go] [Fa\ Go\ Gr\ | Fo] \rangle, \langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Fo\ | Fa\ Go\ Gr] \rangle)$
- $Q_5 = (\langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Gr\ | Fa\ Fo\ Go] [Fa\ Go\ Gr\ | Fo] [Go\ | Fa\ Fo\ Gr] \rangle, \langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Fo\ | Fa\ Go\ Gr] \rangle)$
- $Q_6 = (\langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Gr\ | Fa\ Fo\ Go] [Fa\ Go\ Gr\ | Fo] [Go\ | Fa\ Fo\ Gr] [Fa\ Go\ | Fo\ Gr] \rangle, \langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Gr\ | Fa\ Fo\ Go] [Fa\ Go\ Gr\ | Fo] [Go\ | Fa\ Fo\ Gr] [Fa\ Fo\ Go\ | Gr] \rangle, \langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Fo\ | Fa\ Go\ Gr] \rangle)$
- $G = (\langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Gr\ | Fa\ Fo\ Go] [Fa\ Go\ Gr\ | Fo] [Go\ | Fa\ Fo\ Gr] [Fa\ Go\ | Fo\ Gr] [Fa\ Fo\ Go\ Gr] \rangle, \langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Gr\ | Fa\ Fo\ Go] [Fa\ Go\ Gr\ | Fo] [Go\ | Fa\ Fo\ Gr] [Fa\ Fo\ Go\ | Gr] \rangle, \langle [Fa\ Fo\ Go\ Gr\] [Fo\ Gr\ | Fa\ Go] [Fa\ Fo\ Gr\ | Go] [Fo\ | Fa\ Go\ Gr] \rangle)$

Depth-first search (search tree)



Breadth-first search (search tree)



Exercises: Artificial Intelligence

Bidirectional Search

Bidirectional Search

PROBLEM 1: BREADTH-FIRST?

Other methods than 2 x breadth-first

- Bidirectional search is complete for each combination with at least one complete search-strategy.
 - 2 x Breadth-first
 - 2 x Depth-first
 - Breadth-first and Depth-first
- Not each combination benefits from searching at both ends.

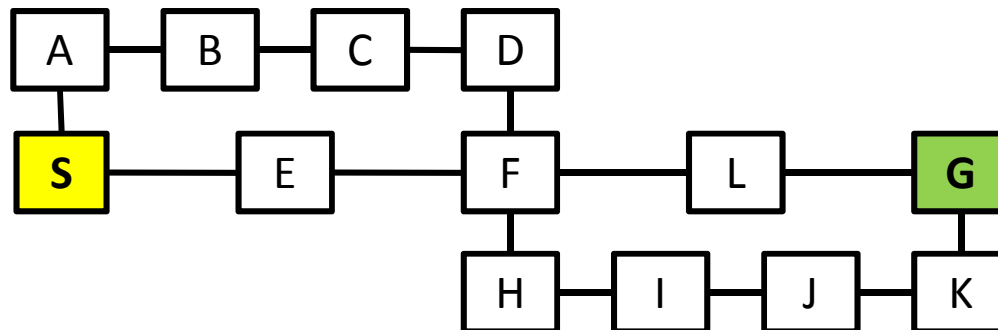
2 x Depth-first

- Forward:

- ($\langle S \rangle$) \rightarrow ($\langle SA \rangle, \langle SE \rangle$) \rightarrow ($\langle SAB \rangle, \langle SE \rangle$) \rightarrow ($\langle SABCD \rangle, \langle SE \rangle$)
 \rightarrow ($\langle SABCDF \rangle, \langle SE \rangle$)

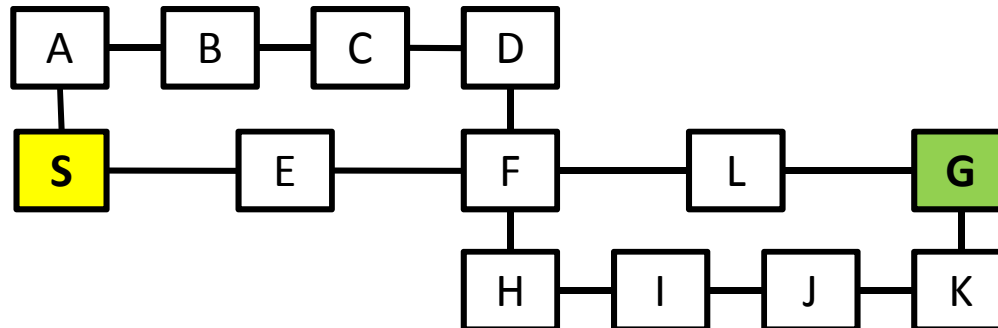
- Backward:

- ($\langle G \rangle$) \rightarrow ($\langle GK \rangle, \langle GL \rangle$) \rightarrow ($\langle GKJ \rangle, \langle GL \rangle$) \rightarrow ($\langle GKJI \rangle, \langle GL \rangle$)
 \rightarrow ($\langle GKJIH \rangle, \langle GL \rangle$) \rightarrow ($\langle **GKJIHF** \rangle, \langle **GL** \rangle$)



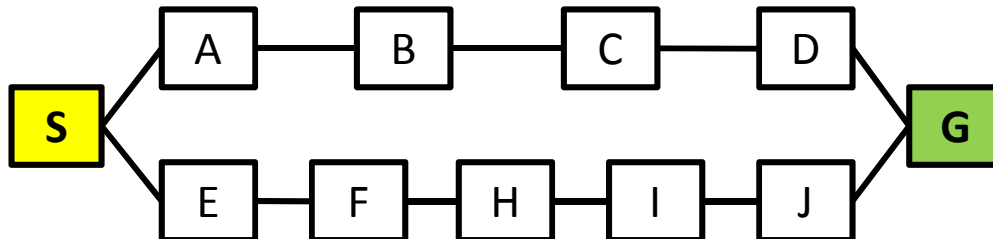
2 x Breadth-first

- Forward:
 - ($\langle S \rangle$) \rightarrow ($\langle SA \rangle, \langle SE \rangle$) \rightarrow ($\langle SE \rangle, \langle SAB \rangle$) \rightarrow ($\langle \mathbf{SAB} \rangle, \langle \mathbf{SEF} \rangle$)
- Backward:
 - ($\langle G \rangle$) \rightarrow ($\langle GK \rangle, \langle GL \rangle$) \rightarrow ($\langle GL \rangle, \langle GKJ \rangle$) \rightarrow ($\langle \mathbf{GKJ} \rangle, \langle \mathbf{GLF} \rangle$)



Breadth-first and Depth-first

- Forward (Breadth-first):
 - ($\langle S \rangle$) \rightarrow ($\langle SA \rangle, \langle SE \rangle$) \rightarrow ($\langle SE \rangle, \langle SAB \rangle$) \rightarrow ($\langle SAB \rangle, \langle SEF \rangle$)
 \rightarrow (**$\langle SEF \rangle$** , **$\langle SABC \rangle$**)
- Backward (Depth-first):
 - ($\langle G \rangle$) \rightarrow ($\langle GJ \rangle, \langle GD \rangle$) \rightarrow ($\langle GJI \rangle, \langle GD \rangle$) \rightarrow ($\langle GJIH \rangle, \langle GD \rangle$)
 \rightarrow (**$\langle GJIHF \rangle$** , **$\langle GD \rangle$**)

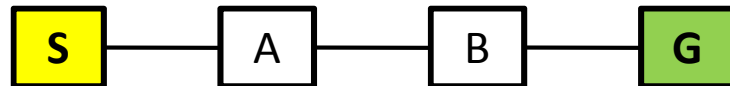


Bidirectional Search

PROBLEM 2: SHARED-STATE CHECK?

Replace shared-state check

- When only checking identical end-states, paths can cross each other unnoticed.
- Forward:
 - $(\langle S \rangle) \rightarrow (\langle SA \rangle) \rightarrow (\langle SAB \rangle) \rightarrow (\langle SABG \rangle)$
- Backward:
 - $(\langle G \rangle) \rightarrow (\langle GB \rangle) \rightarrow (\langle GBA \rangle) \rightarrow (\langle GBAS \rangle)$



Exercises: Artificial Intelligence

Beam Search

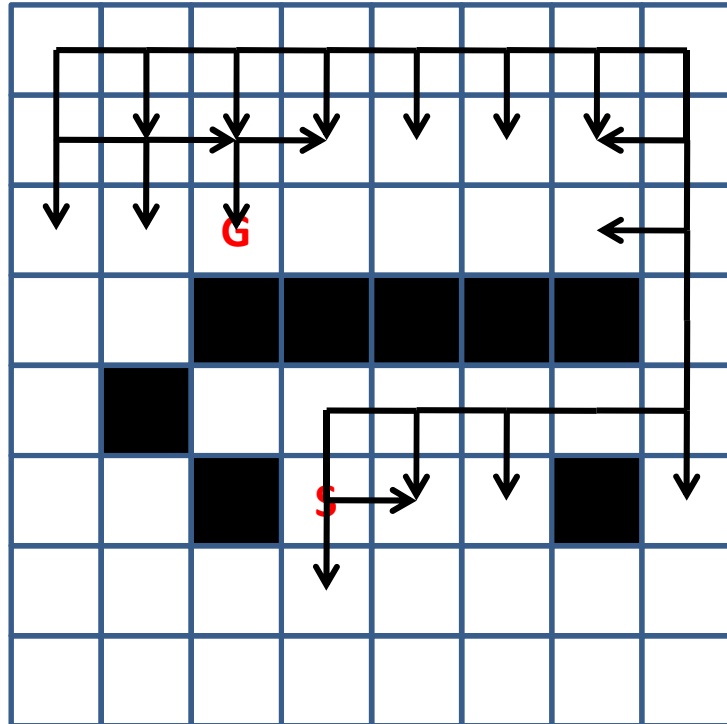
Beam Search

- ***Input:***
 - **QUEUE:** Path only containing root
 - **WIDTH:** Number
- ***Algorithm:***
 - **WHILE** (QUEUE not empty && goal not reached) **DO**
 - Remove ***all paths*** from QUEUE
 - Create paths to all children (of all paths)
 - Reject paths with loops
 - ***Sort new paths (according to heuristic)***
 - ***(Optimization: Remove paths without successor)***
 - Add WIDTH ***best paths*** to QUEUE
 - **IF** goal reached
 - **THEN** success
 - **ELSE** failure

Exercises: Artificial Intelligence

Path Search

Depth-first Search



17	16	15	14	13	12	11	10
18	19	20					9
		G					8
							7
		2	1	3	4	5	6
			S				

Heuristic: Manhattan Distance

4	3	2	3	4	5	6	7
3	2	1	2	3	4	5	6
2	1	0	1	2	3	4	5
3	2						6
4		2	3	4	5	6	7
5	4		4	5	6		8
6	5	4	5	6	7	8	9
7	6	5	6	7	8	9	10

Hill-climbing I Search

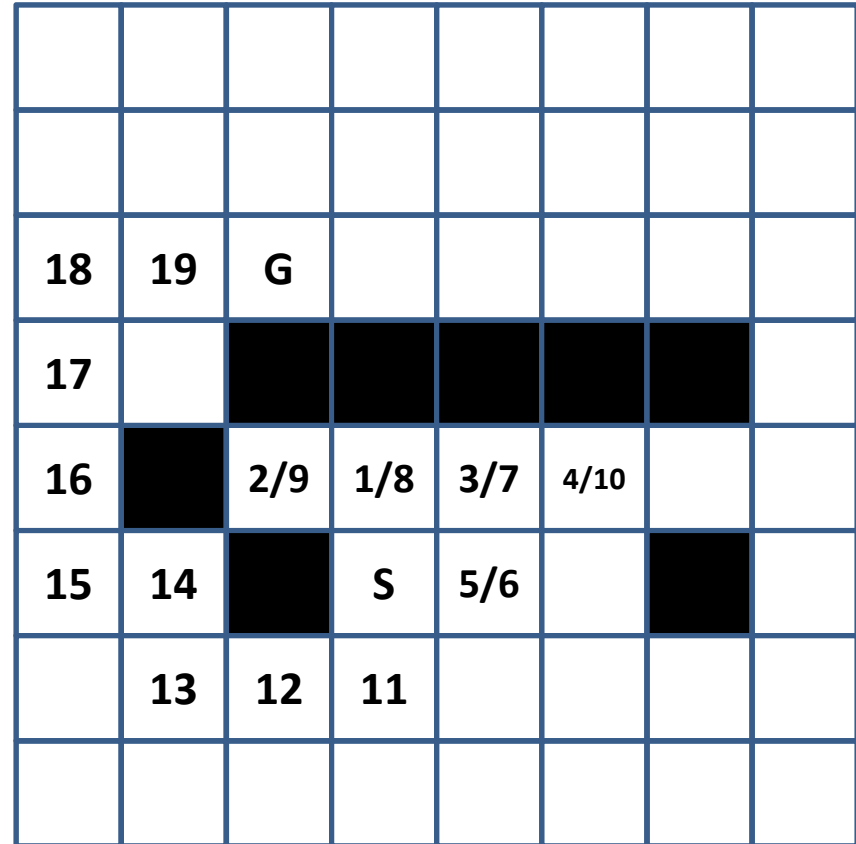
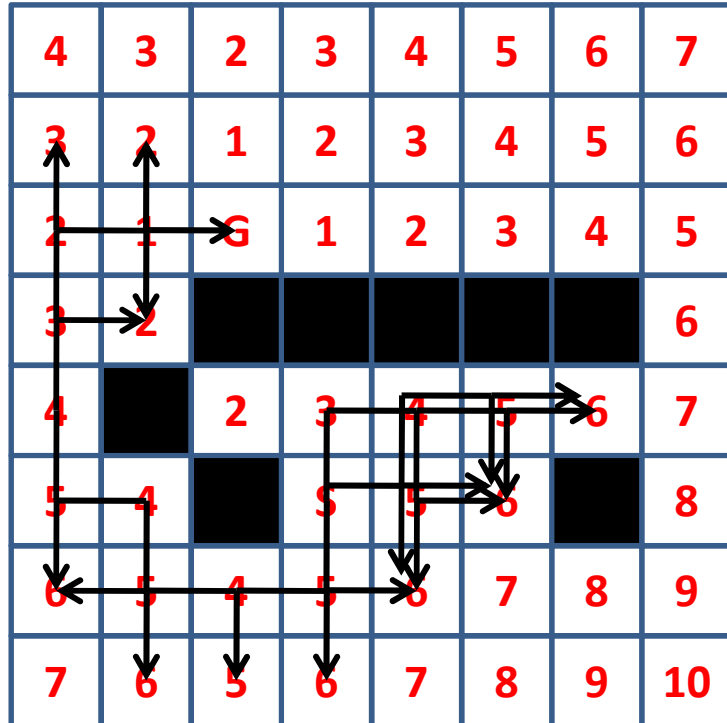
4	3	2	3	4	5	6	7
3	2	1	2	3	4	5	6
2	1	G	1	2	3	4	5
3	2						6
4		2	3	4	5	6	7
5	4		5	6			8
6	5	4	5	6	7	8	9
7	6	5	6	7	8	9	10

Diagram illustrating Hill-climbing I Search on a 2D grid. The grid contains numbers 1 through 10 and a goal state 'G'. The search path is indicated by arrows starting from 'G' and moving to adjacent cells with higher values. The path is: G (row 3, col 3) → (row 3, col 4) → (row 3, col 5) → (row 3, col 6) → (row 3, col 7) → (row 3, col 8) → (row 4, col 8) → (row 5, col 8) → (row 6, col 8) → (row 7, col 8) → (row 8, col 8) → (row 9, col 8) → (row 10, col 8).

		G	12	11	10	9	8	
							7	
			2	1	3	4	5	6
				S				

Diagram illustrating Hill-climbing I Search on a 2D grid. The grid contains numbers 1 through 12 and a goal state 'G'. The search path is indicated by arrows starting from 'S' and moving to adjacent cells with higher values. The path is: S (row 6, col 4) → (row 6, col 3) → (row 6, col 2) → (row 6, col 1) → (row 5, col 1) → (row 4, col 1) → (row 3, col 1) → (row 3, col 2) → (row 3, col 3) → (row 3, col 4) → (row 3, col 5) → (row 3, col 6) → (row 3, col 7) → (row 3, col 8).

Greedy Search



Exercises: Artificial Intelligence

Water Jugs

Representation

- States of the form $[x,y]$, where:
 - x : *contents of 4 liter jug*
 - y : *contents of 3 liter jug*
- Start: $[0,0]$
- Goal: $[2,0]$

Representation

- Rules:

- Fill x: $[x,y] \wedge x < 4 \longrightarrow [4,y]$

- Fill y: $[x,y] \wedge y < 3 \longrightarrow [x,3]$

- Empty x: $[x,y] \wedge x > 0 \longrightarrow [0,y]$

- Empty y: $[x,y] \wedge y > 0 \longrightarrow [x,0]$

- Fill x with y: $[x,y] \wedge x+y > 4 \wedge y > 0 \longrightarrow [4,(x+y-4)]$

- Fill x with y: $[x,y] \wedge x+y \leq 4 \wedge y > 0 \longrightarrow [(x+y),0]$

- Fill y with x: $[x,y] \wedge x+y > 3 \wedge x > 0 \longrightarrow [(x+y-3),3]$

- Fill y with x: $[x,y] \wedge x+y \leq 3 \wedge x > 0 \longrightarrow [0,(x+y)]$

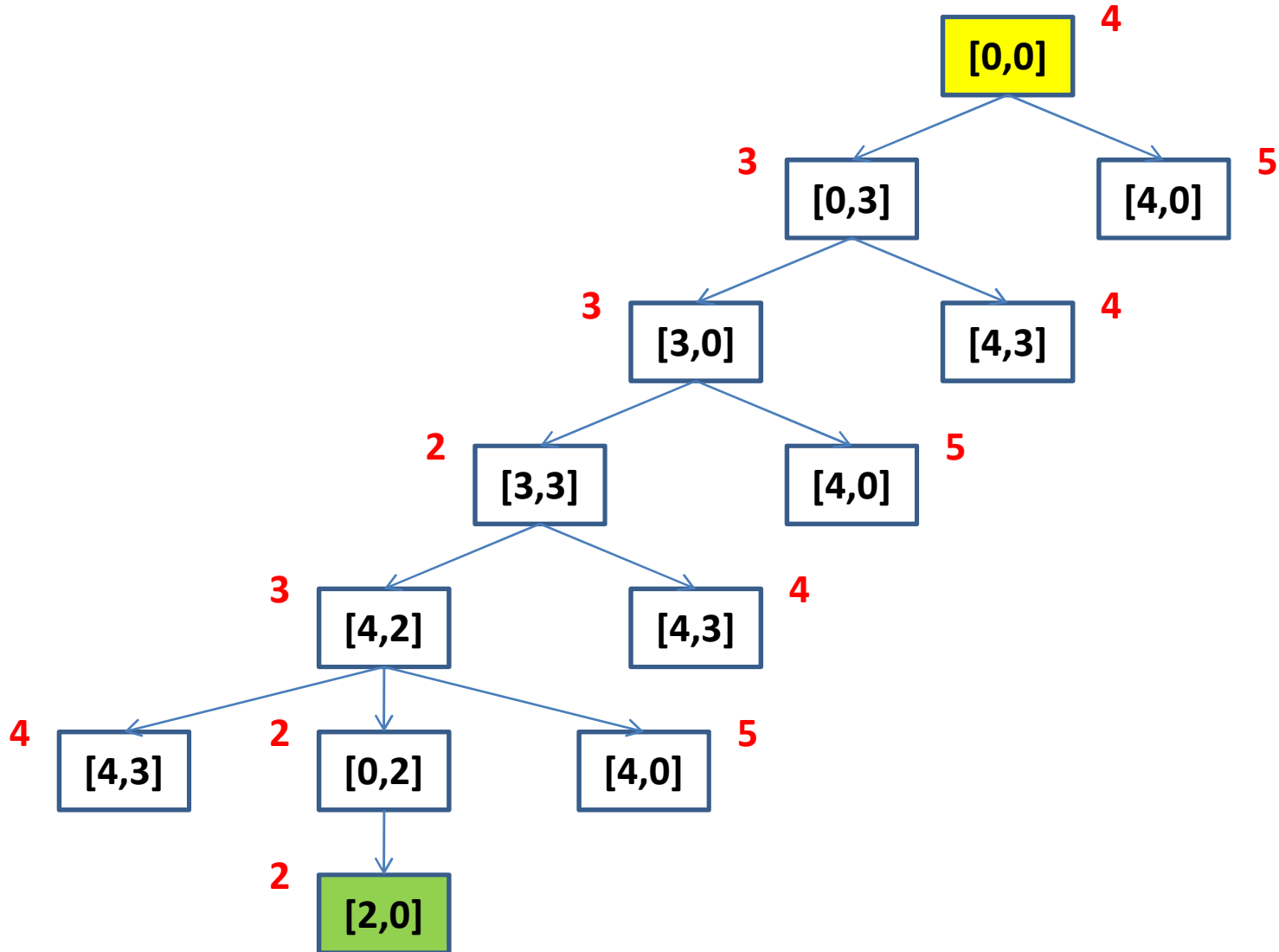
Heuristic

- $H([x,y]) = f(x) + f(y)$
- $f(x)$ is defined as follows:

x	0	1	2	3	4
f(x)	2	1	0	1	3

- We need a jug filled with 2 liter.
- To obtain a jug filled with 2 liter we need a jug filled with either 1 or 3 liter.
- We consider an empty jug better than a jug filled with 4 liter.

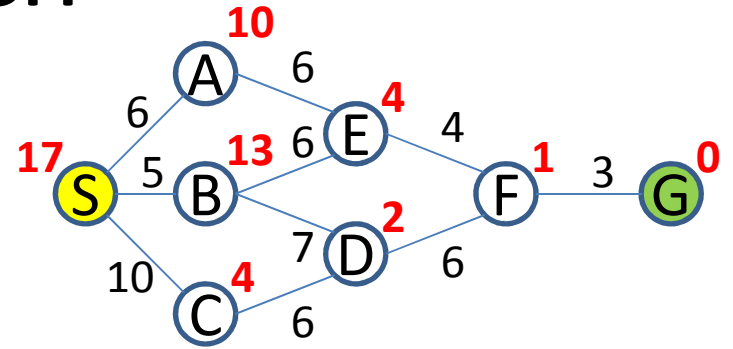
Hill-climbing II Search



Exercises: Artificial Intelligence

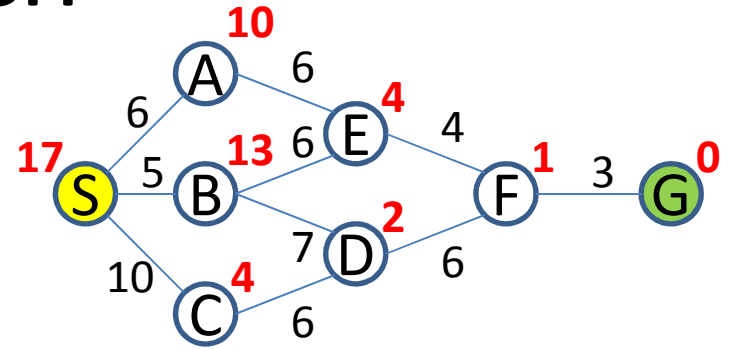
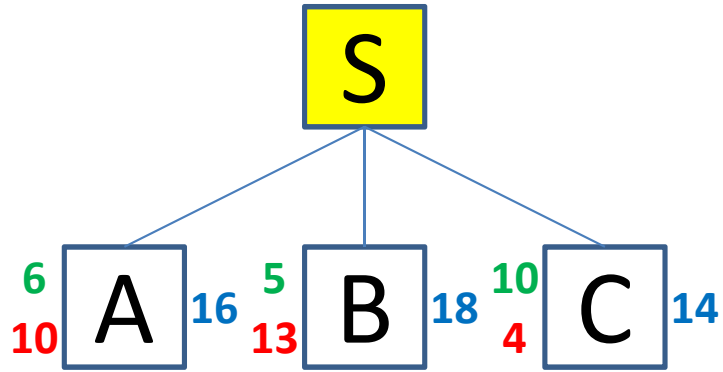
A*

A* Search



QUEUE:
S

A* Search



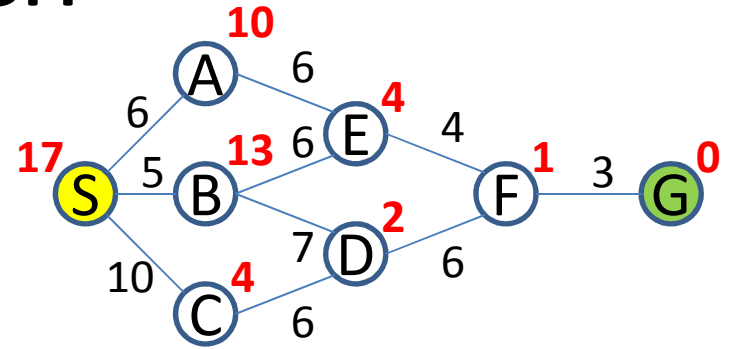
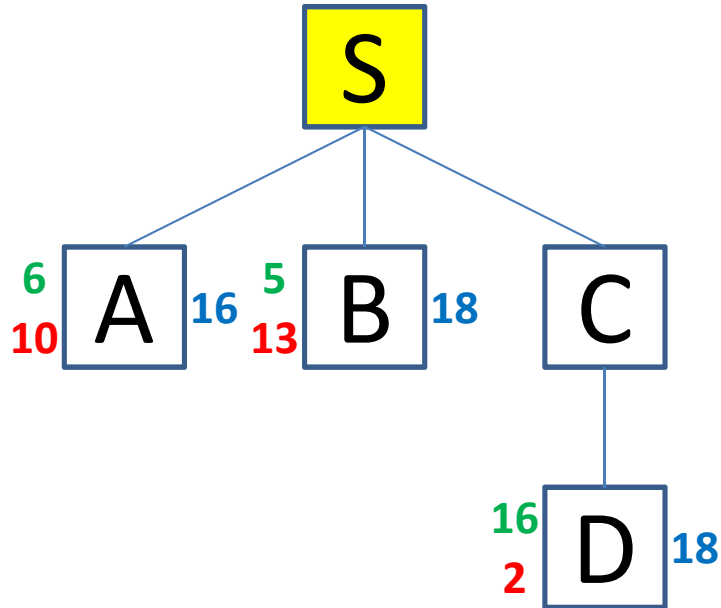
QUEUE:

SC

SA

SB

A* Search



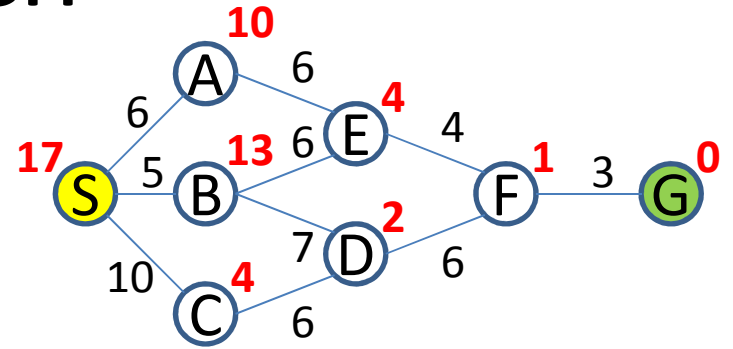
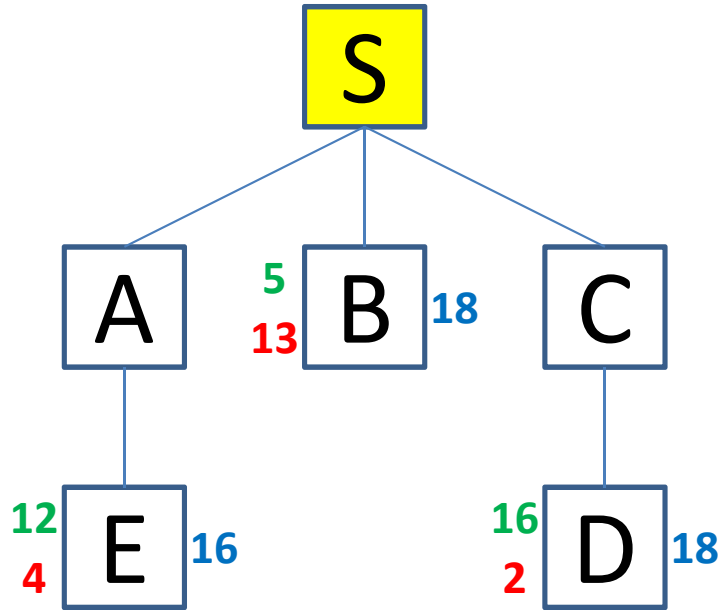
QUEUE:

SA

SCD

SB

A* Search



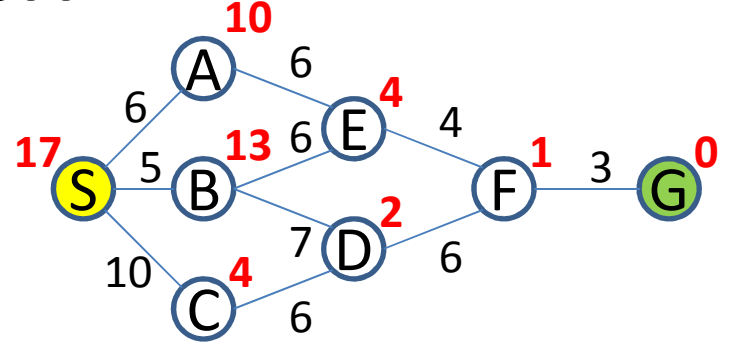
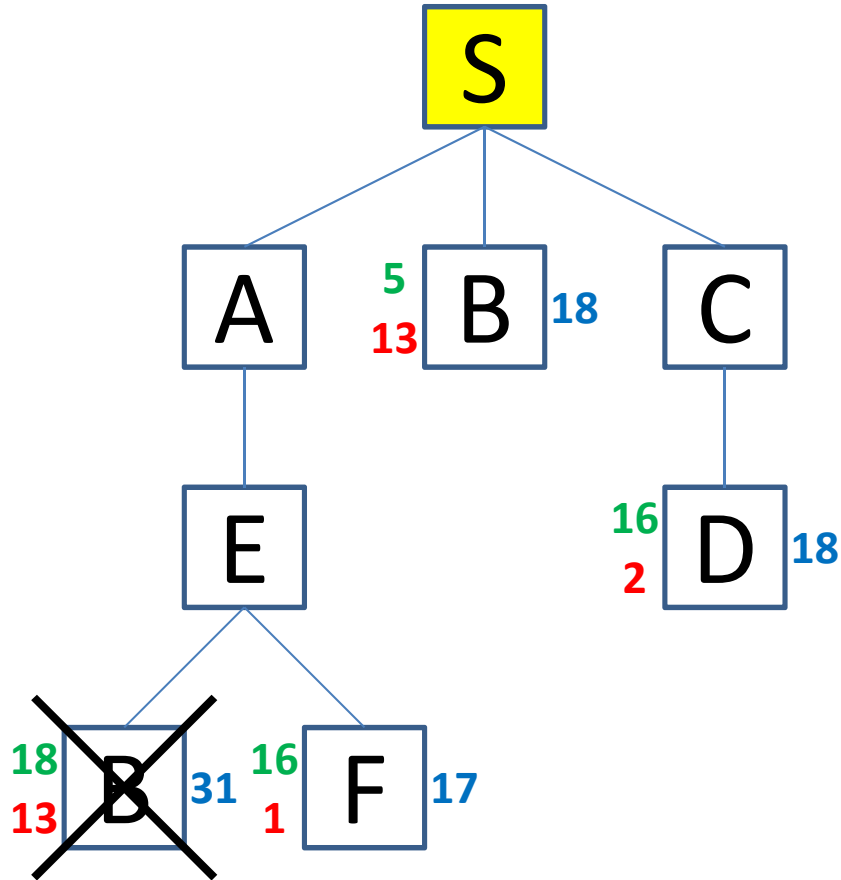
QUEUE:

SAE

SCD

SB

A* Search



QUEUE:

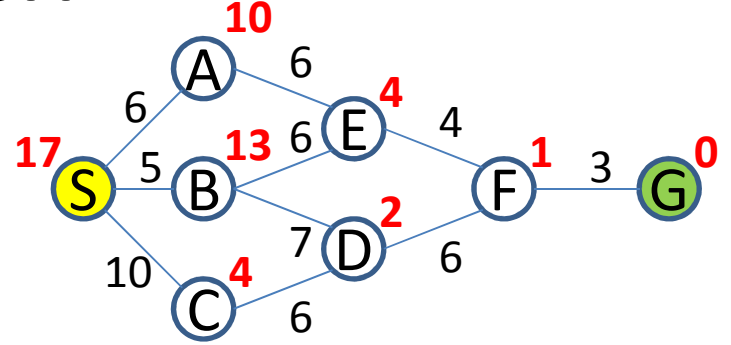
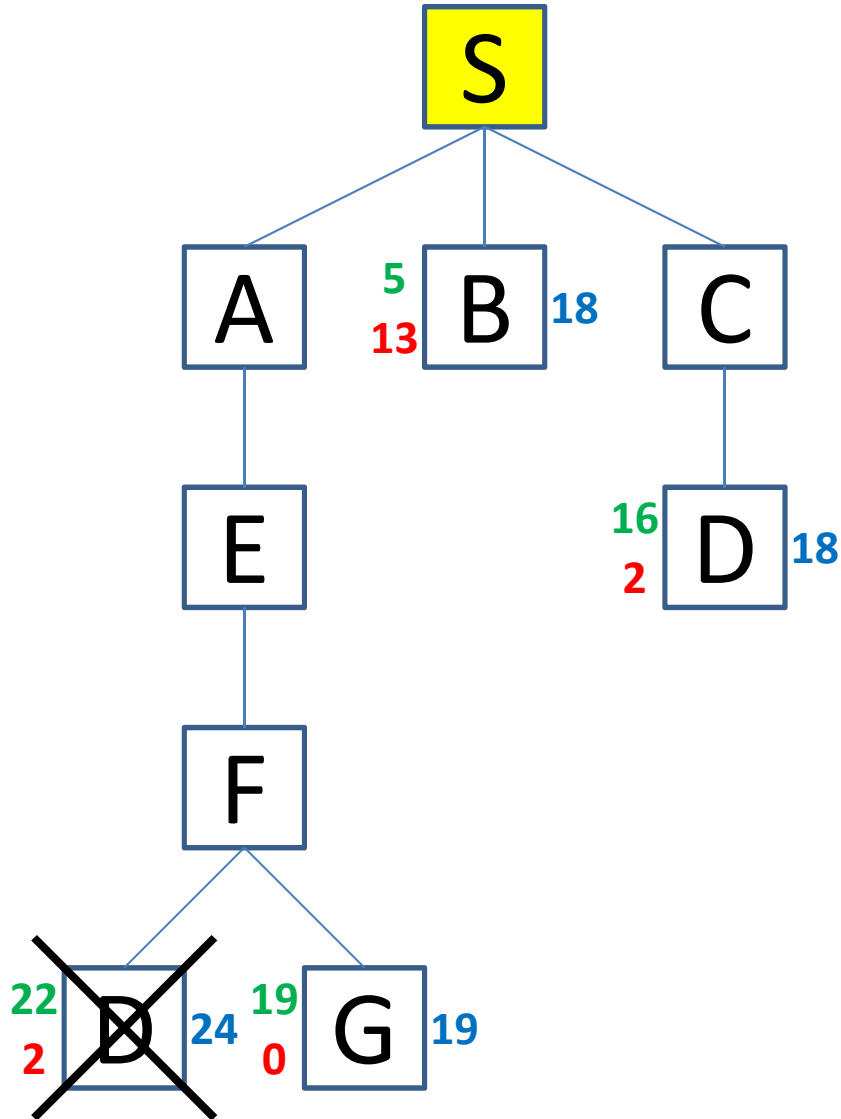
SAEF

SCD

SB

SAEB

A* Search



QUEUE:

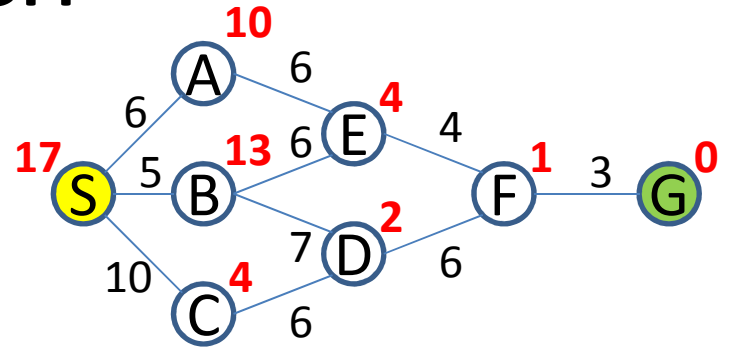
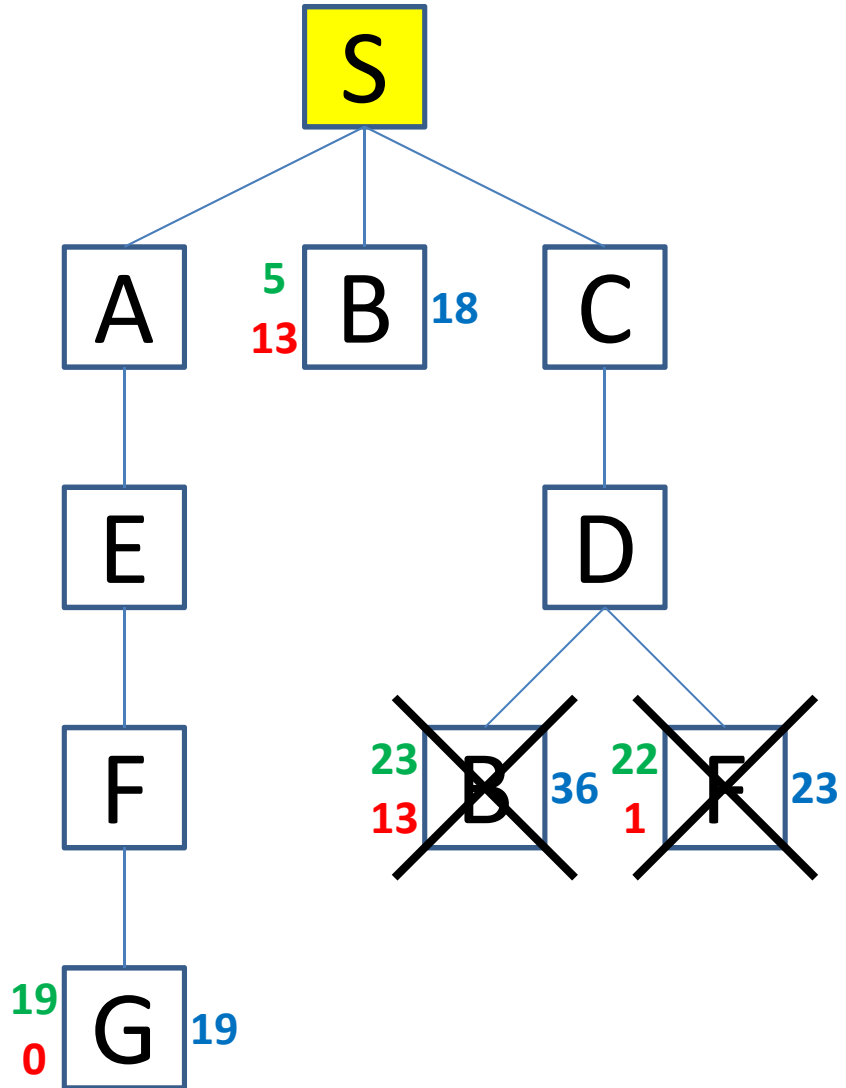
SCD

SB

SAEFG

SAEFD

A* Search



QUEUE:

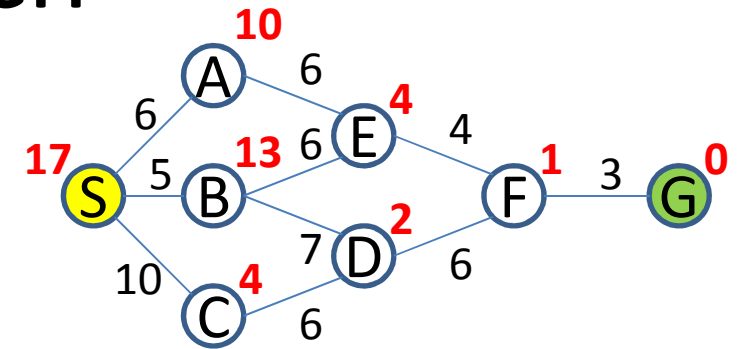
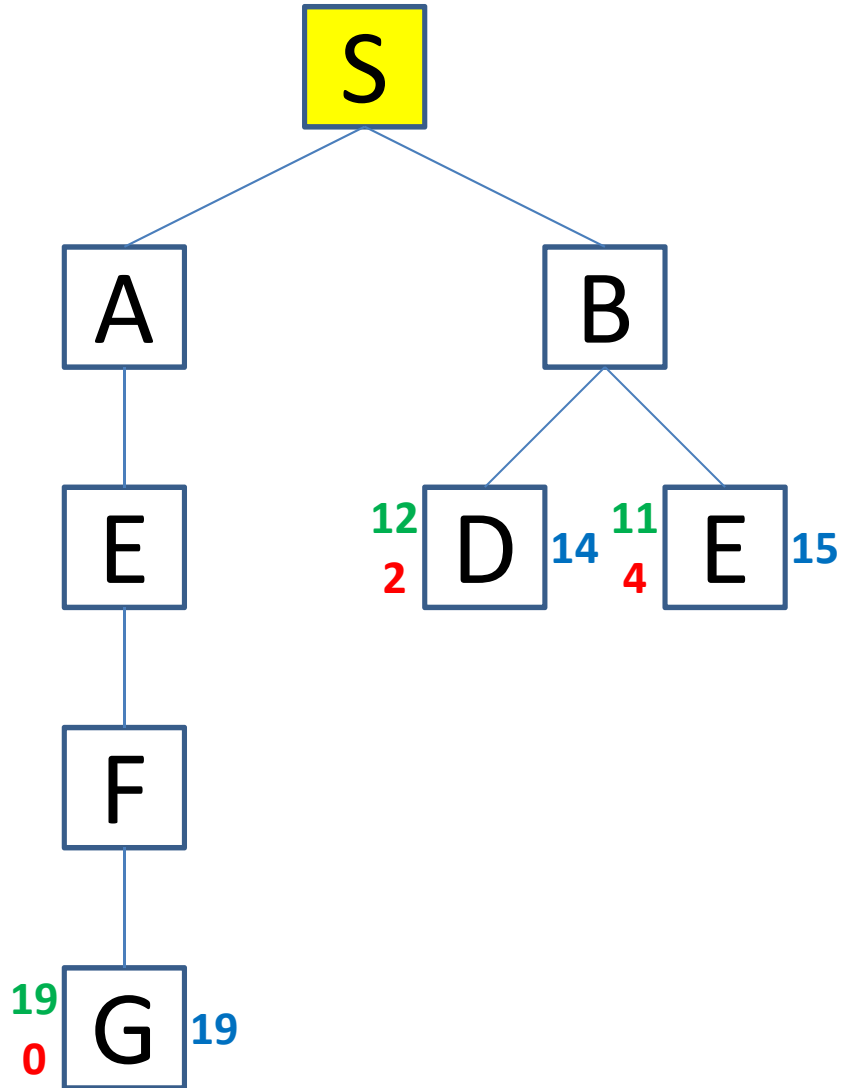
SB

SAEFG

SCDF

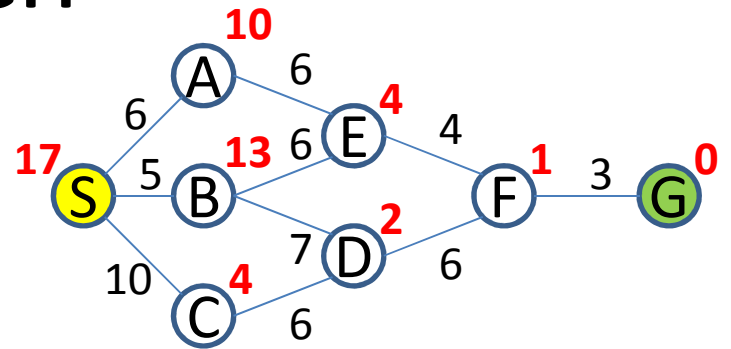
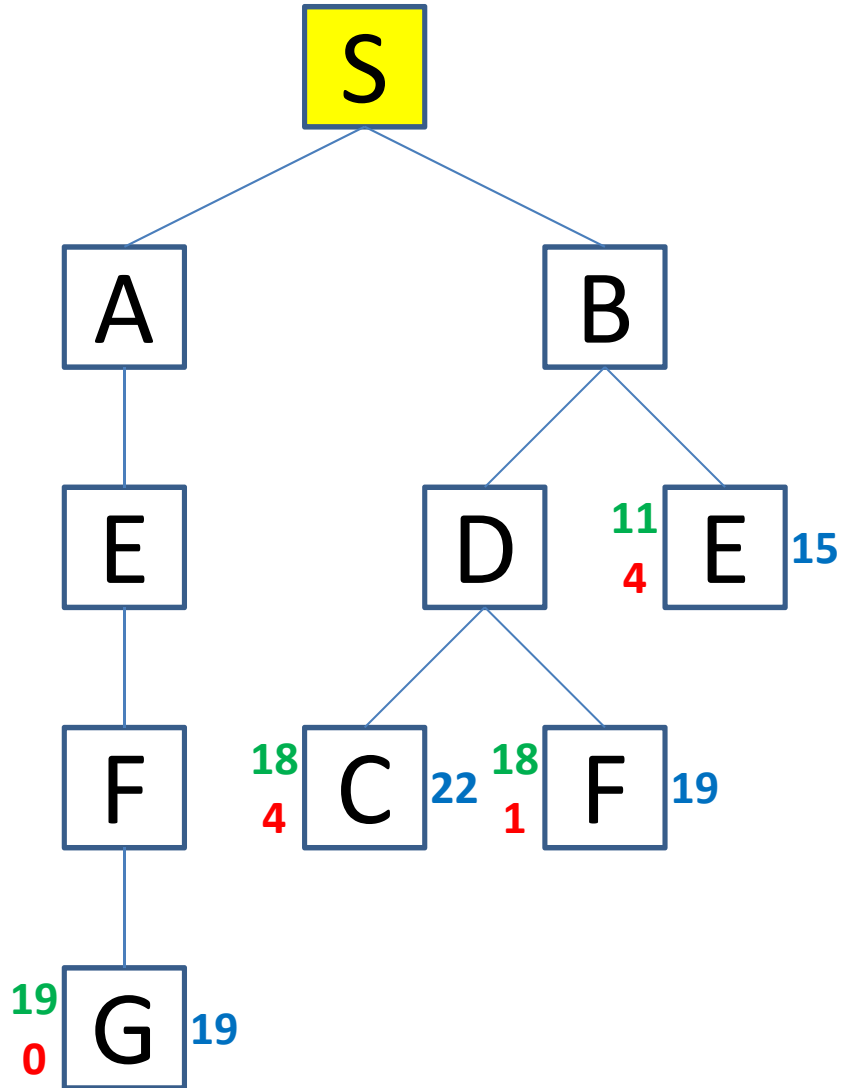
SCDB

A* Search



QUEUE:
SBD
SBE
SAEFG

A* Search



QUEUE:

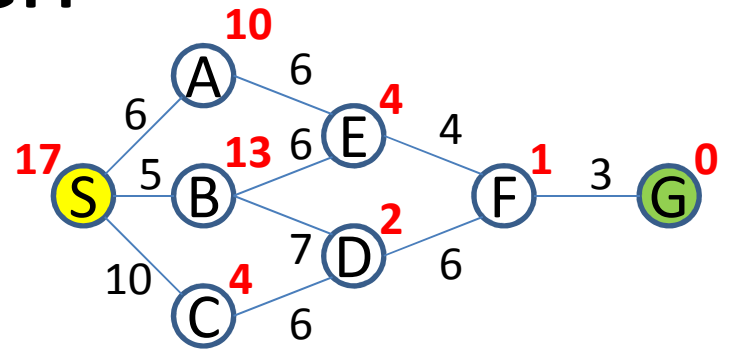
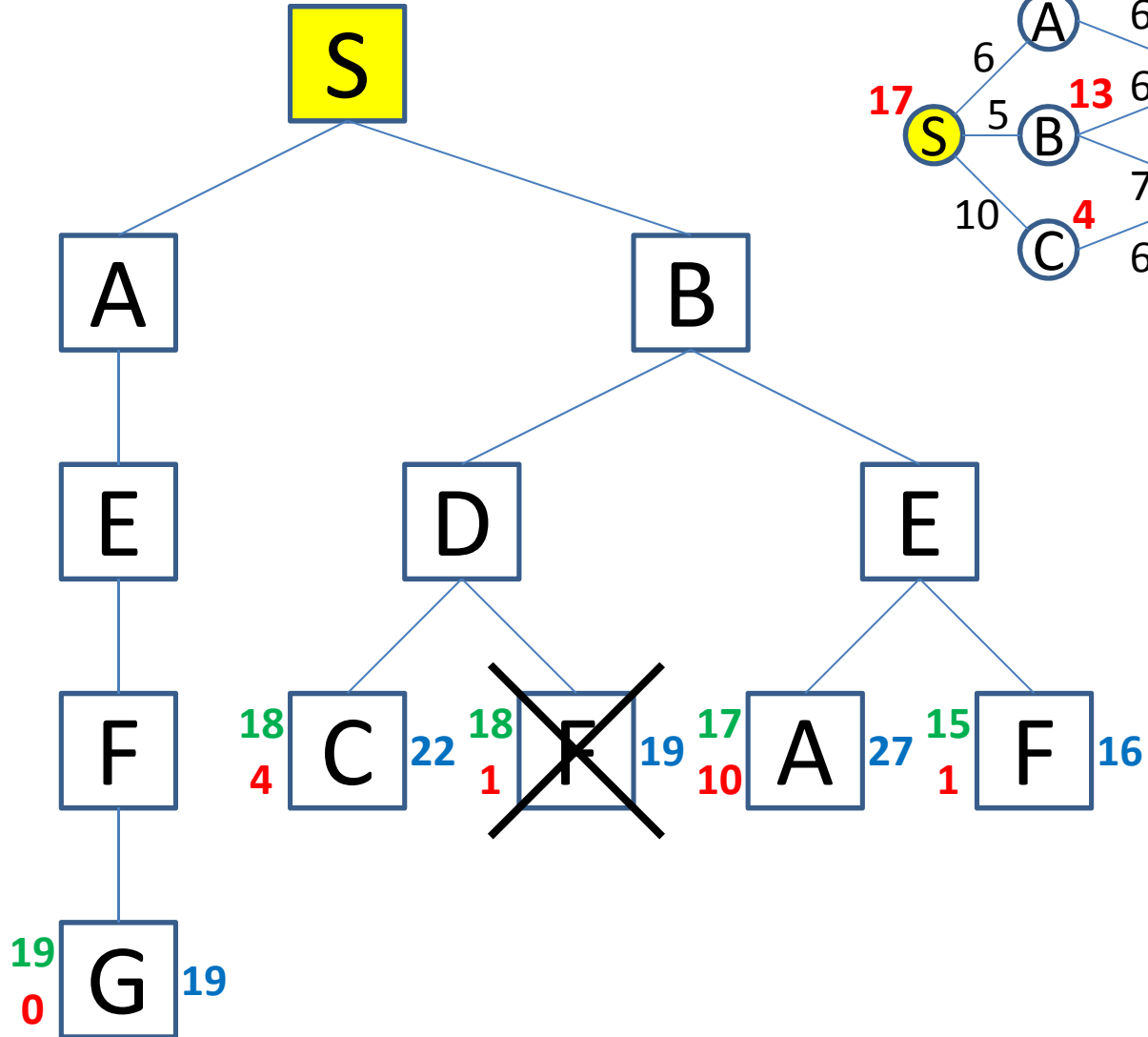
SBE

SBDF

SAEFG

SBDC

A* Search



QUEUE:

SBEF

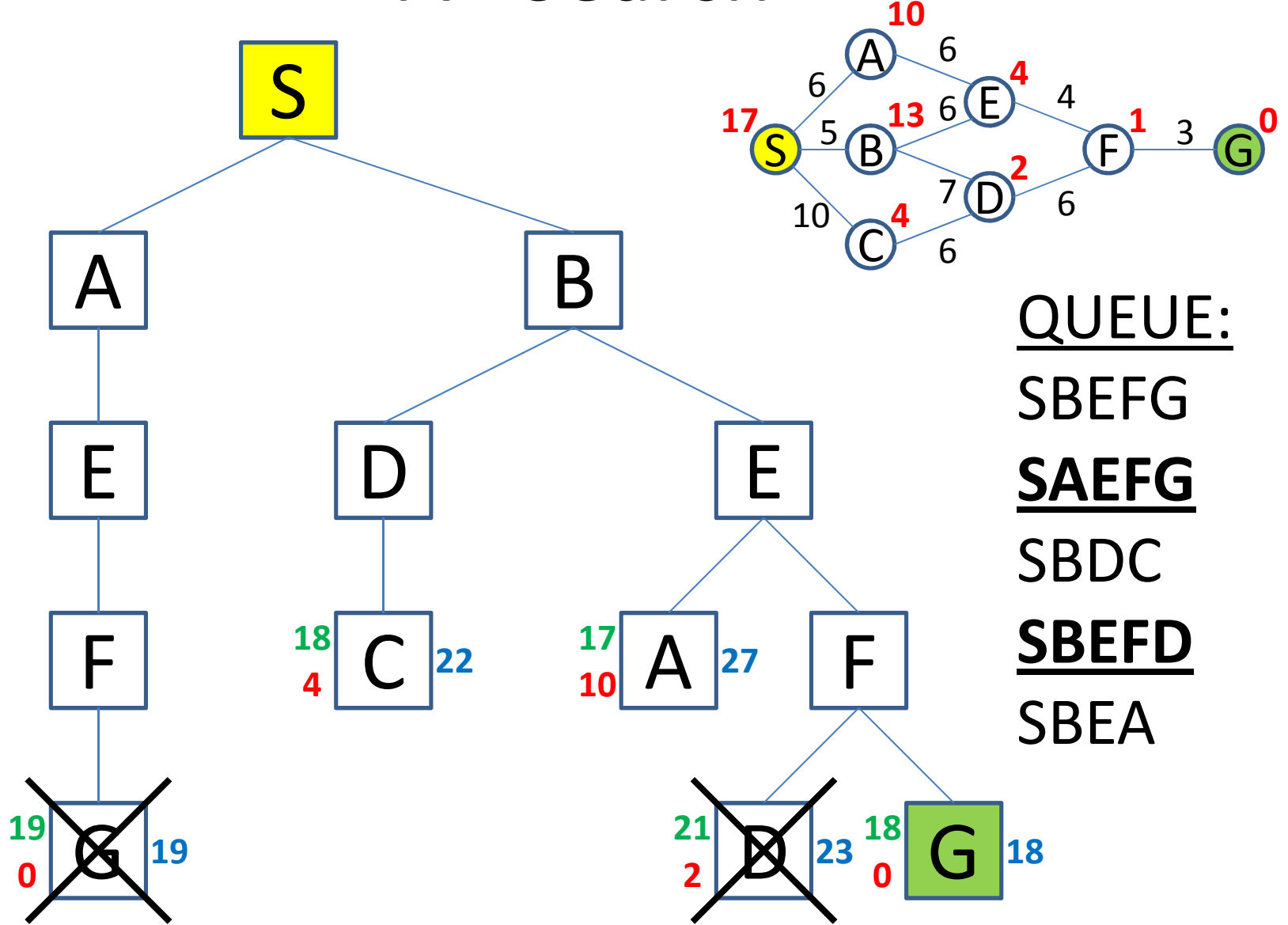
SAEFG

SBDF

SBDC

SBEA

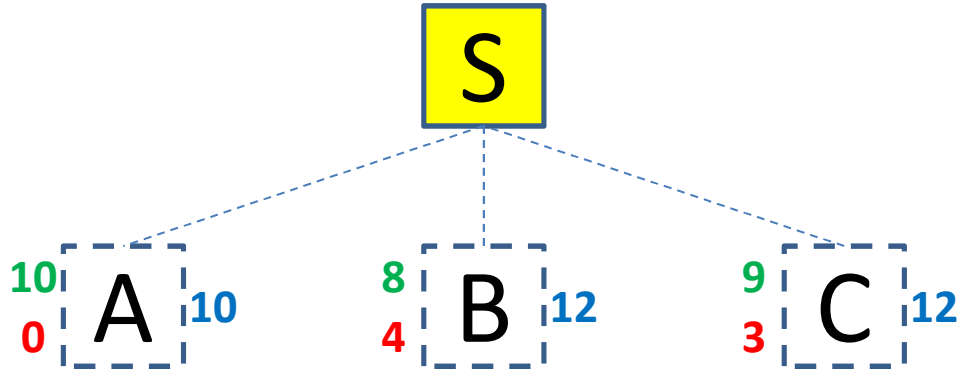
A* Search



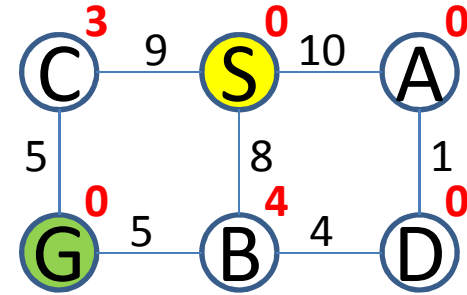
Exercises: Artificial Intelligence

Iterated Deepening A*

IDA* Search



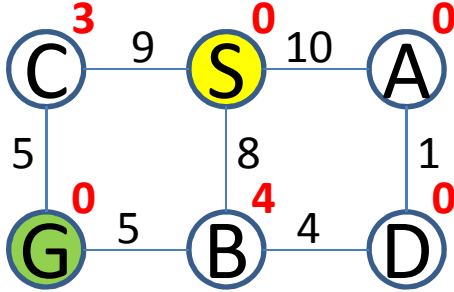
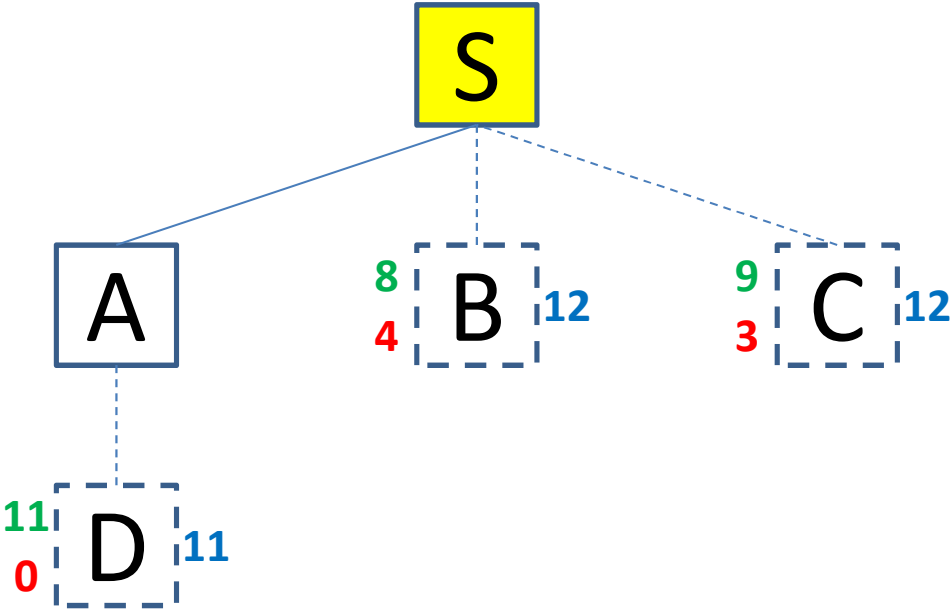
Children are explored depth-first!



f-bound = 0

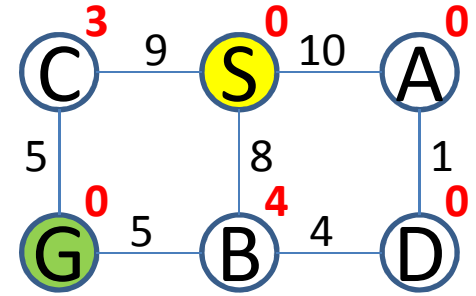
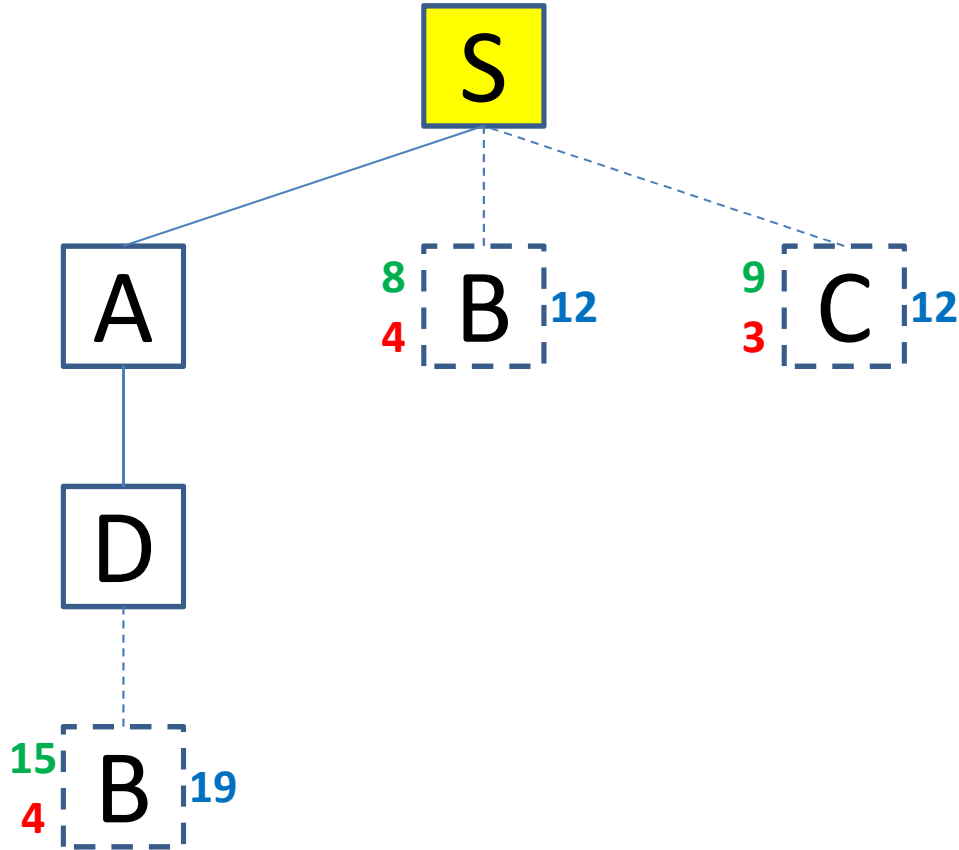
f-new = 10

IDA* Search



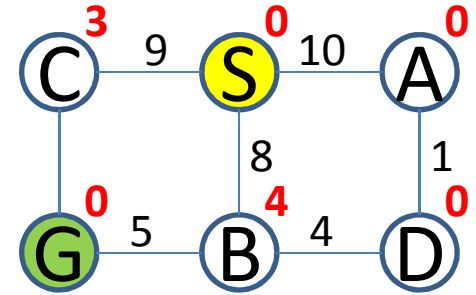
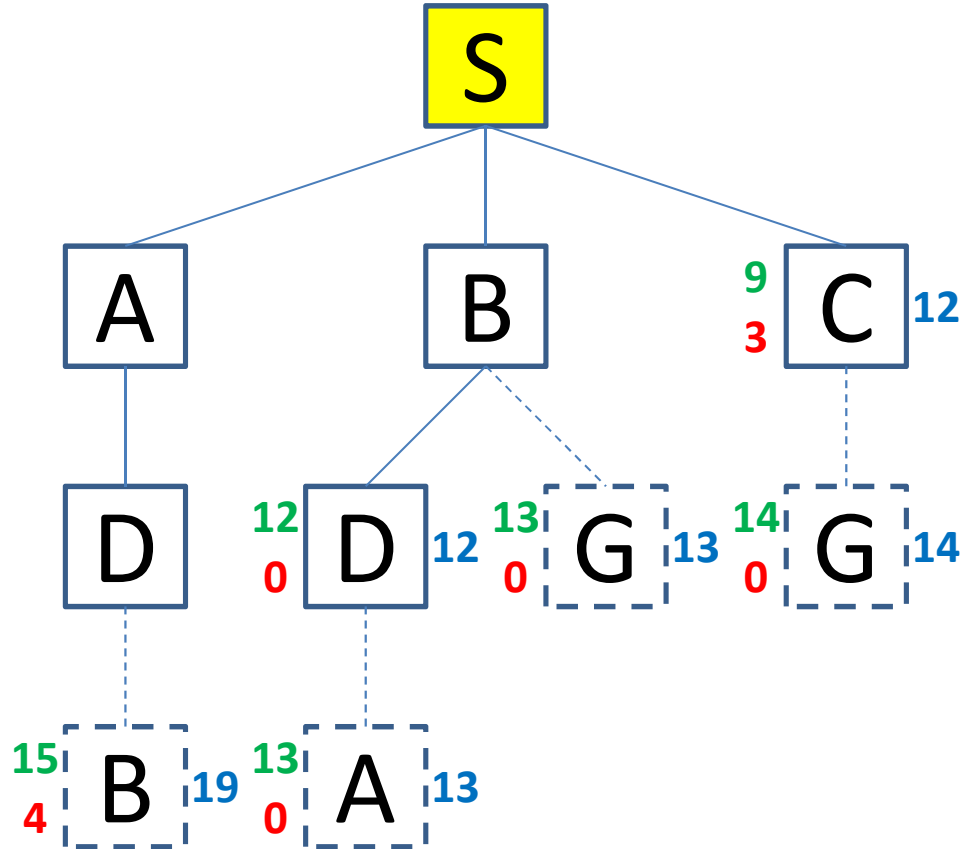
f-bound = 10
f-new = 11

IDA* Search



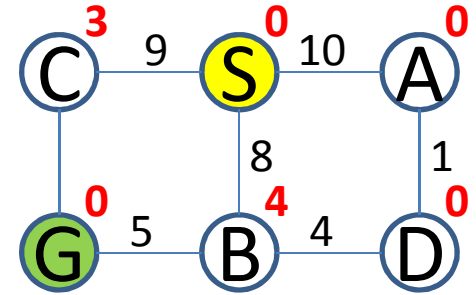
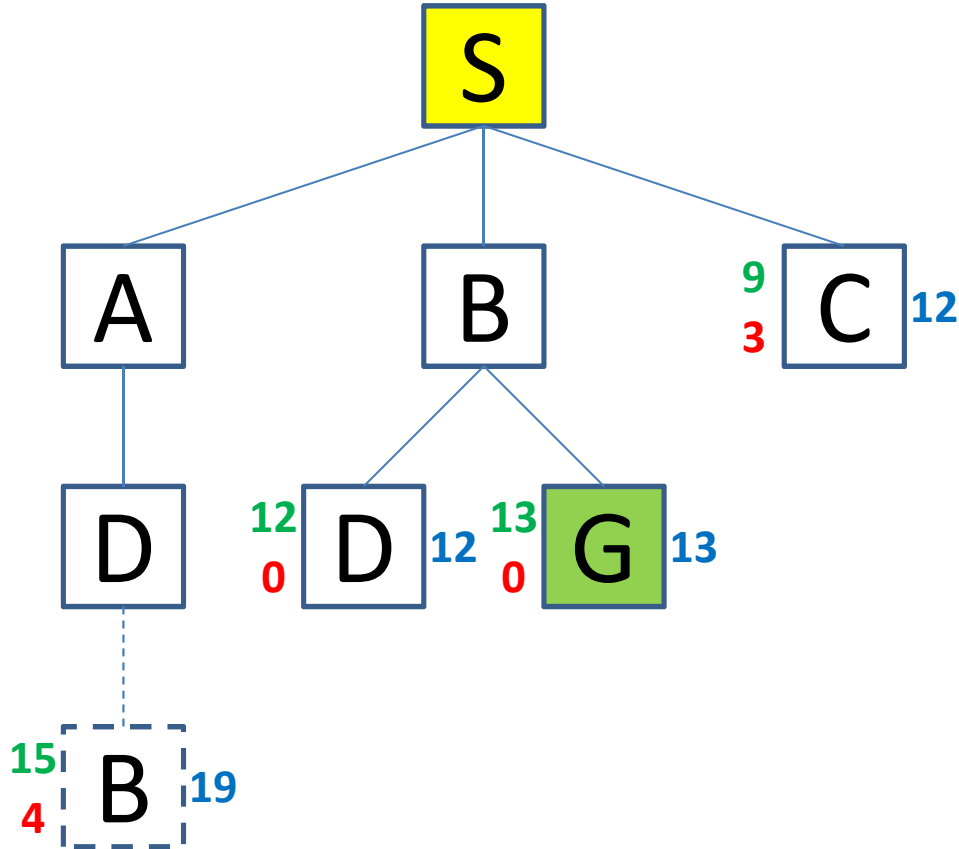
f-bound = 11
f-new = 12

IDA* Search



f-bound = 12
f-new = 13

IDA* Search



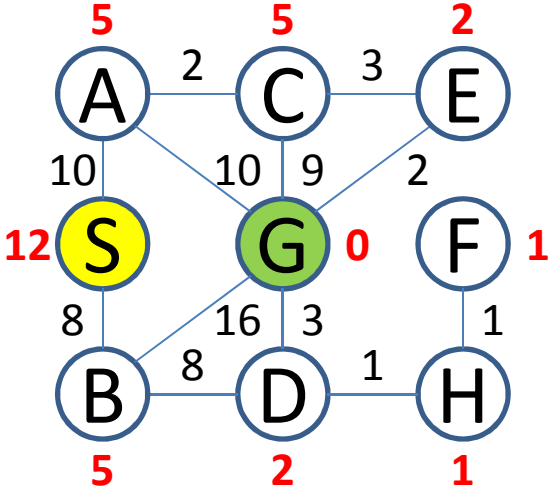
f-bound = 13

f-new = 19

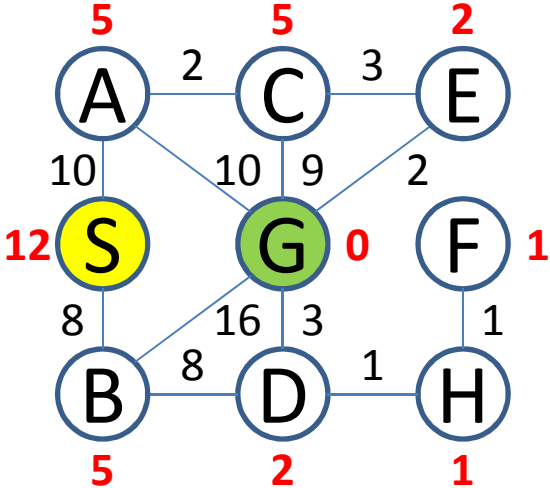
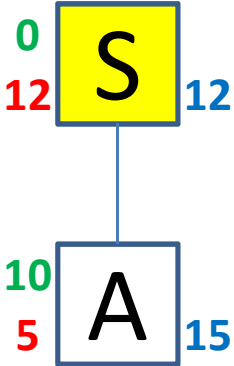
Exercises: Artificial Intelligence

Simplified Memory-bounded A*

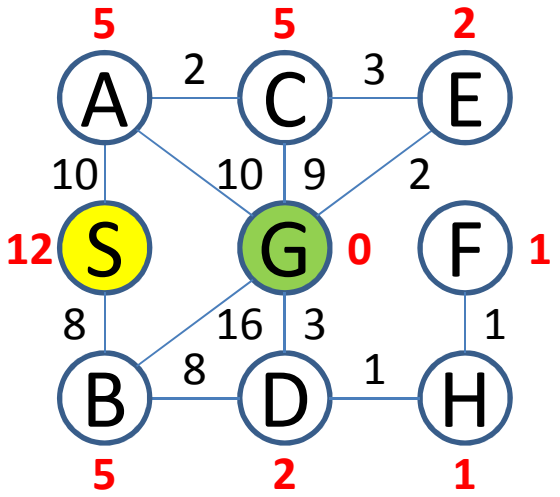
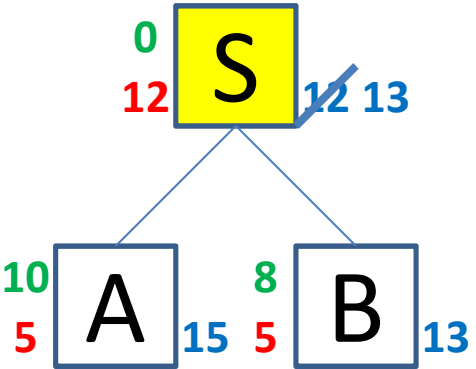
Problem



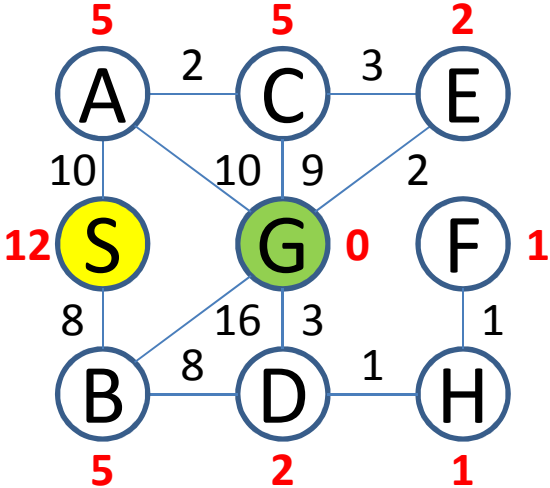
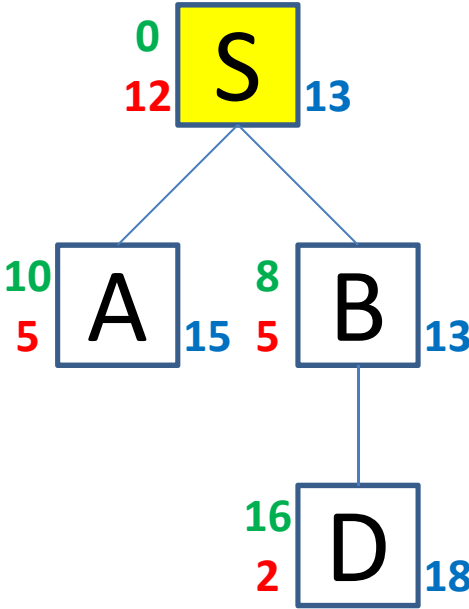
Problem



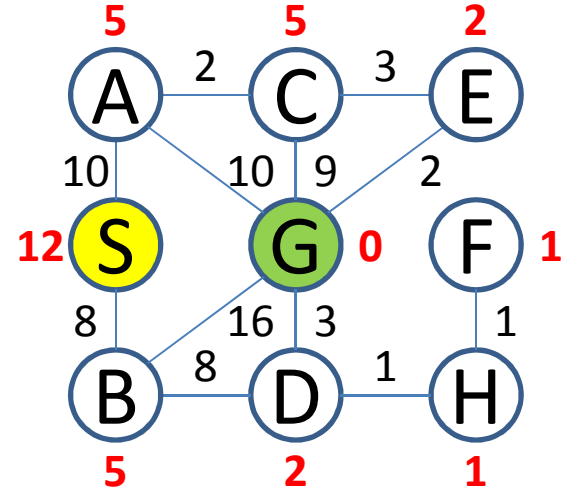
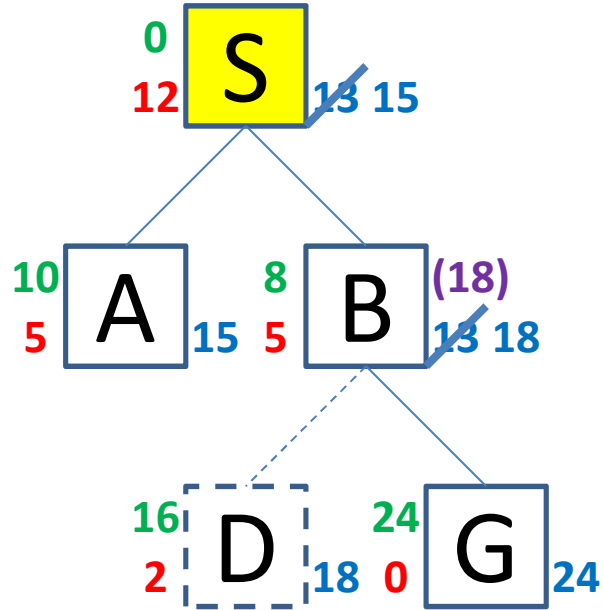
Problem



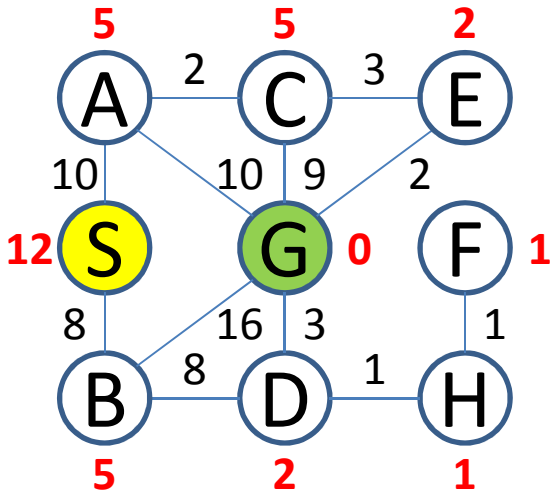
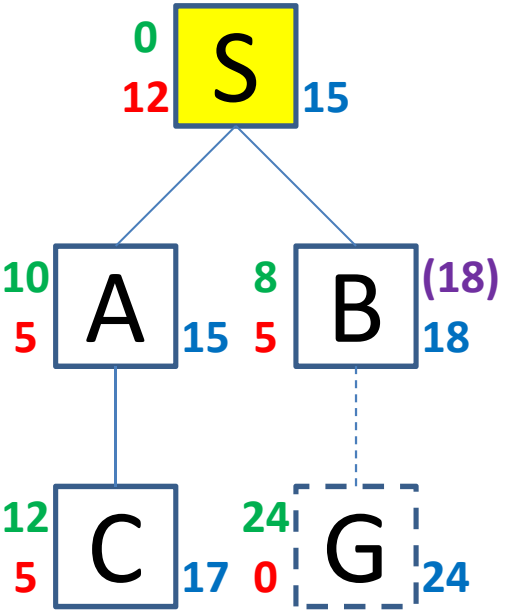
Problem



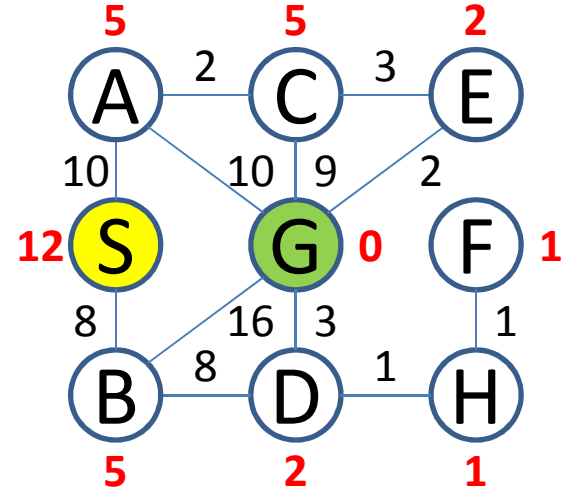
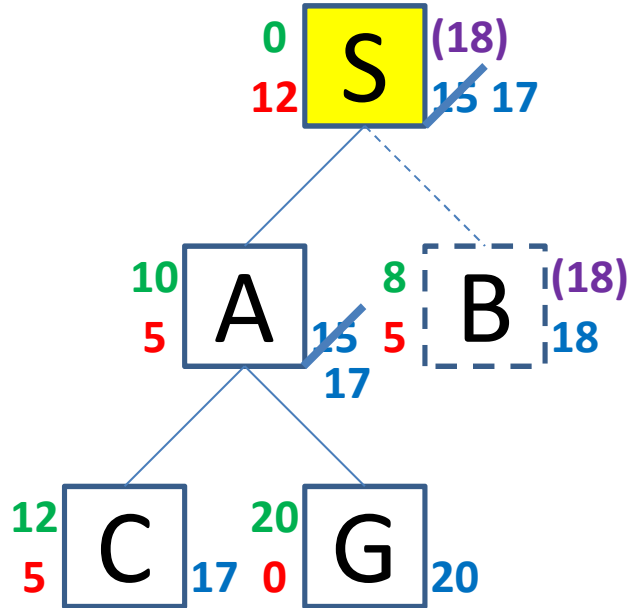
Problem



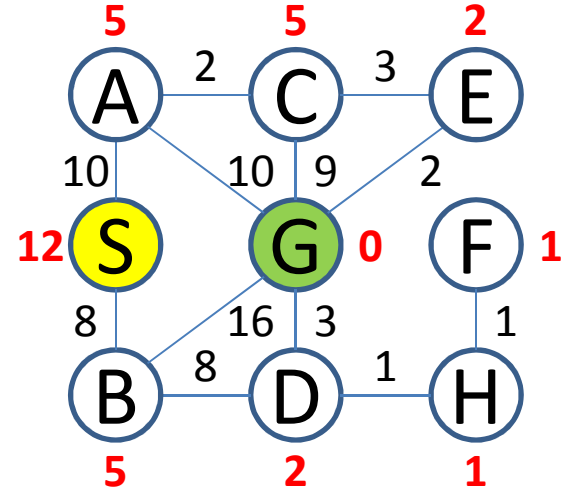
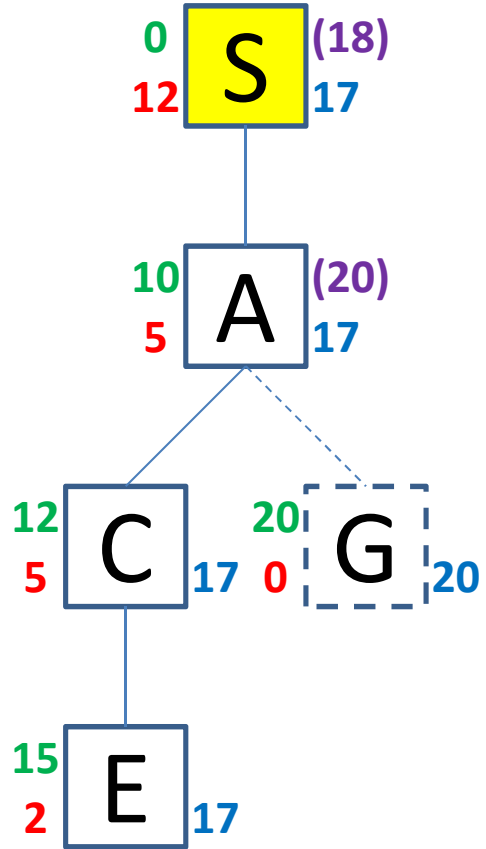
Problem



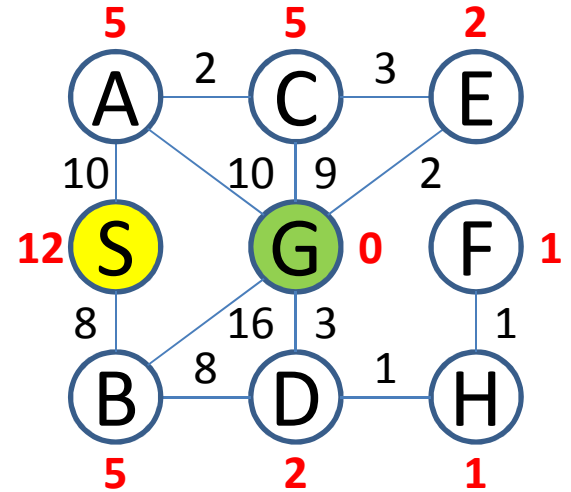
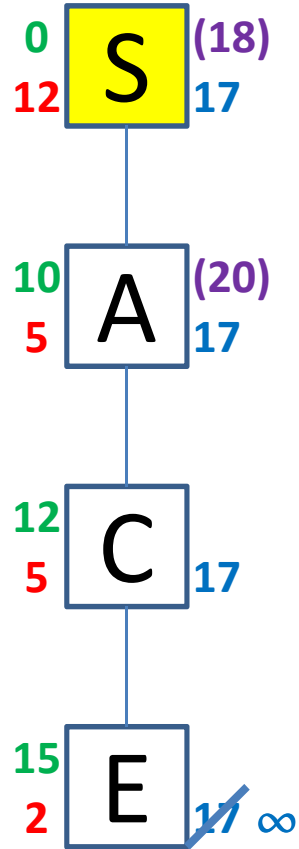
Problem



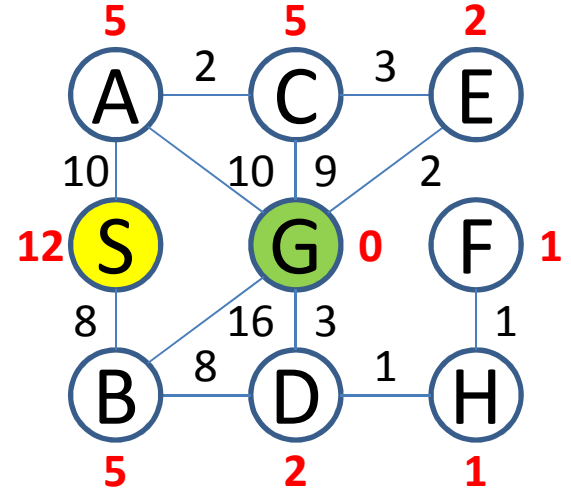
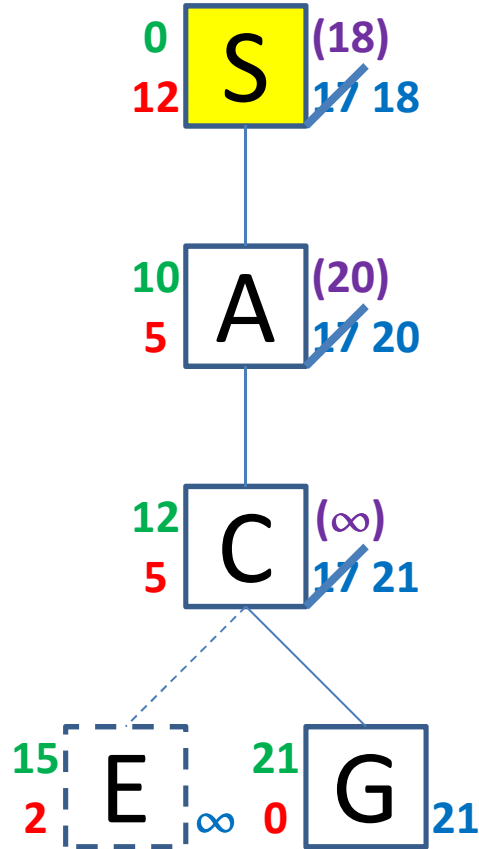
Problem



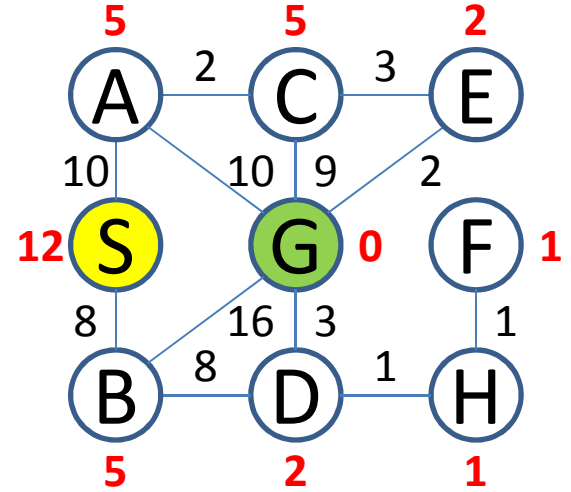
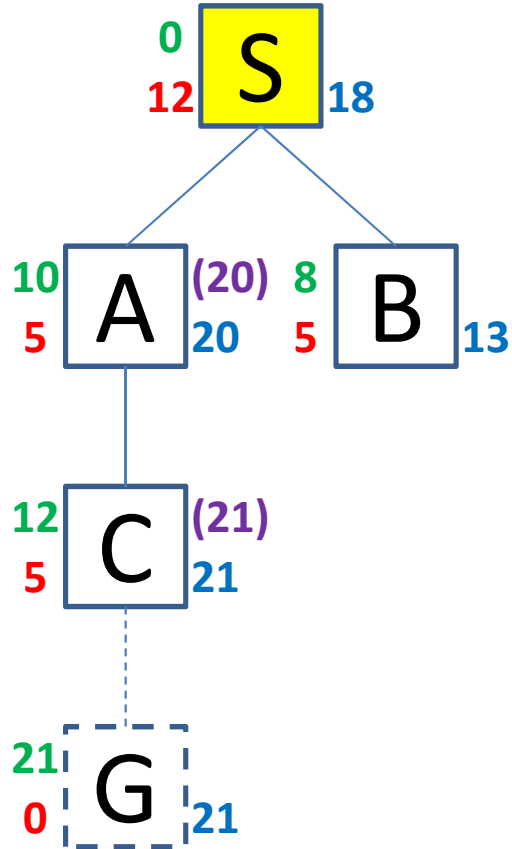
Problem



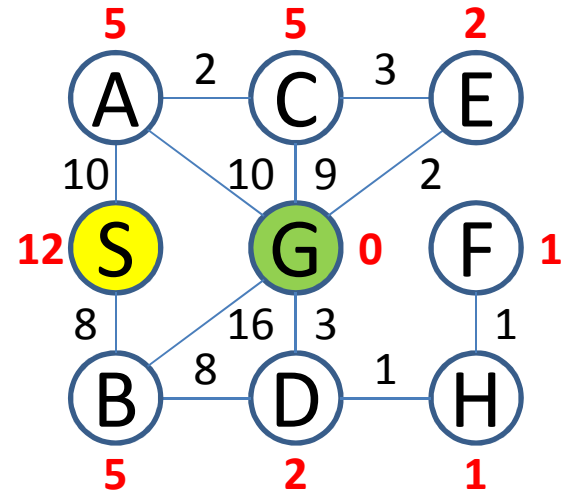
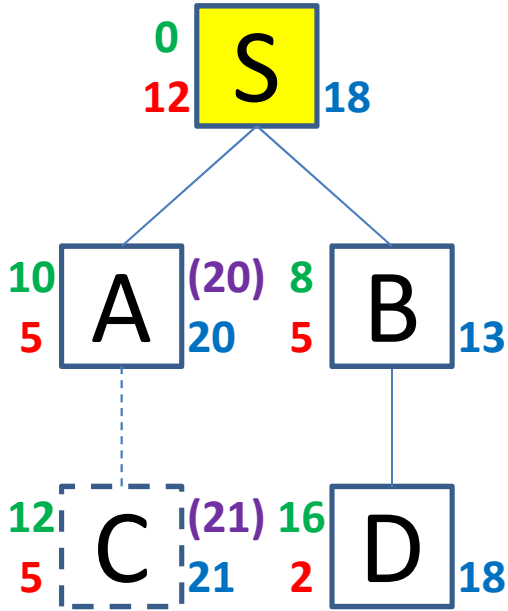
Problem



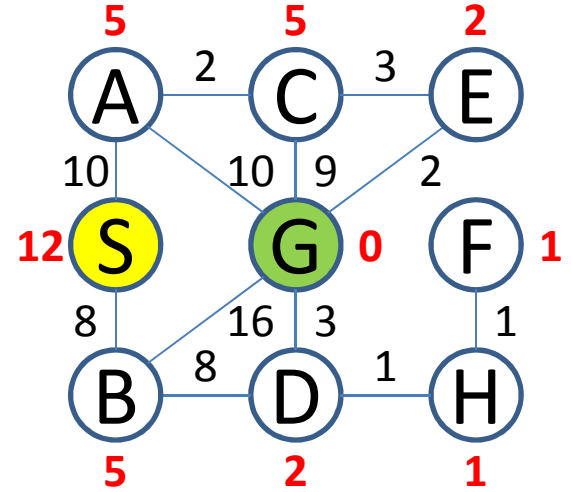
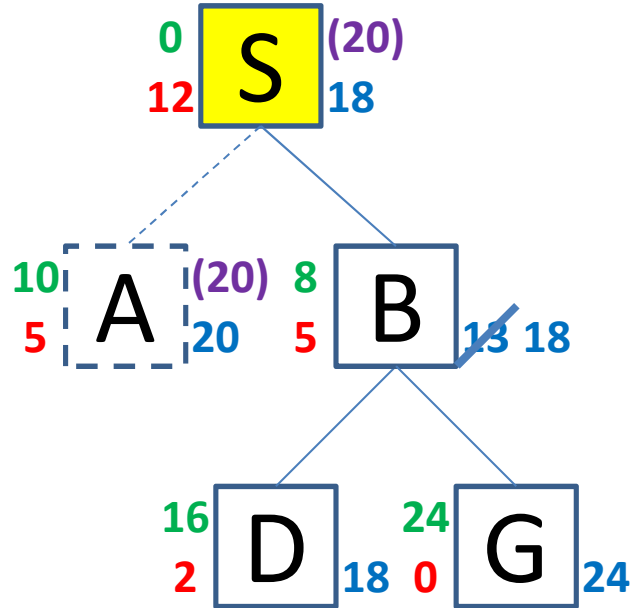
Problem



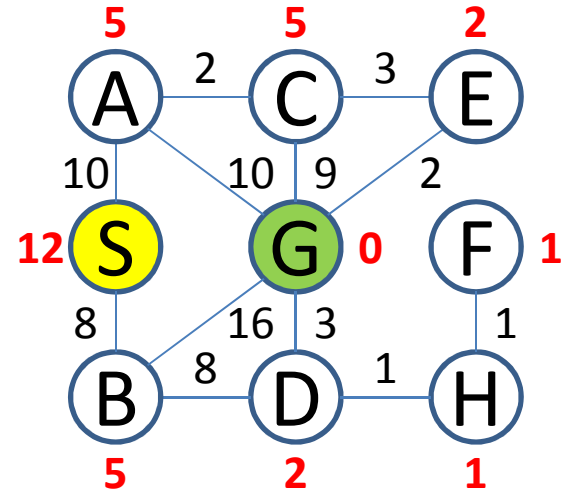
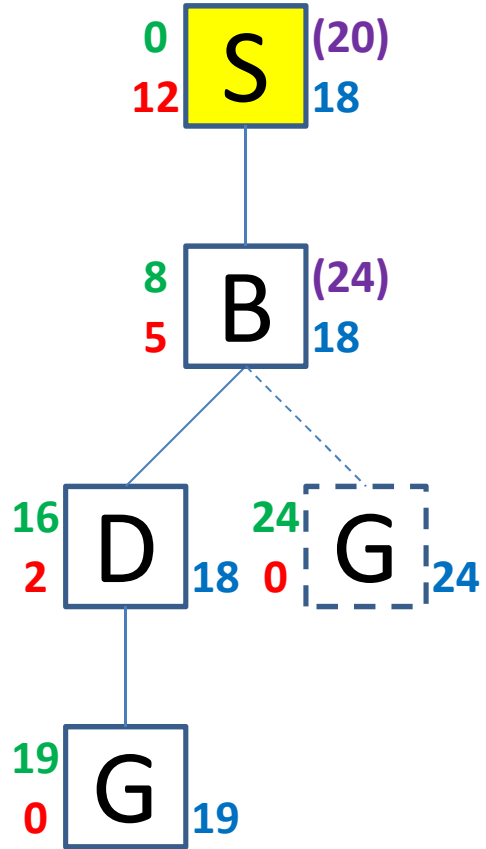
Problem



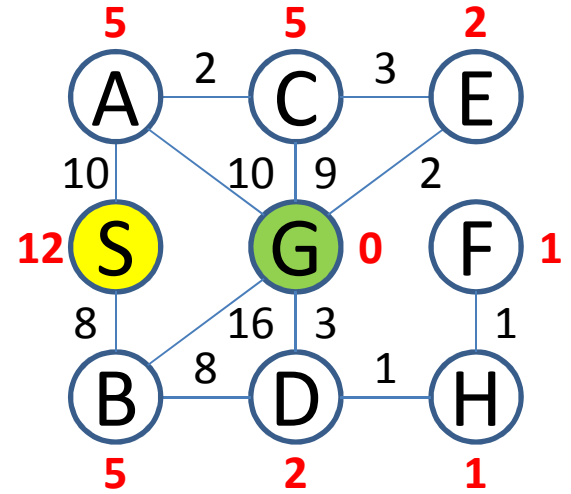
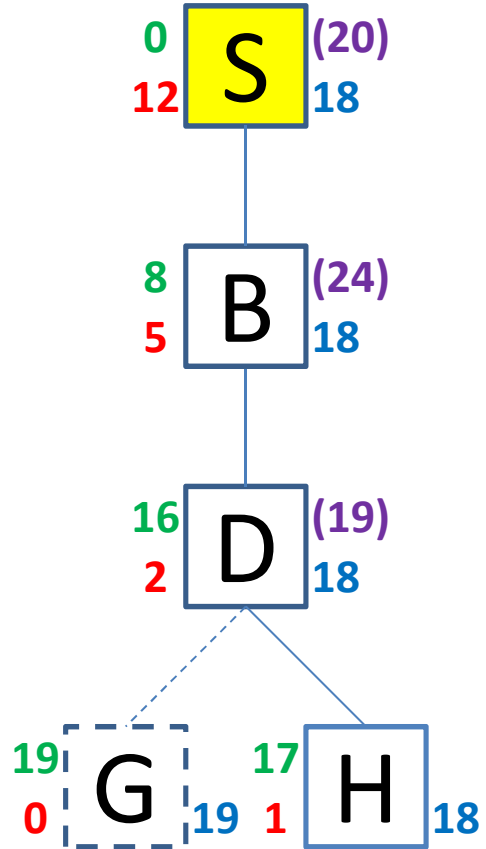
Problem



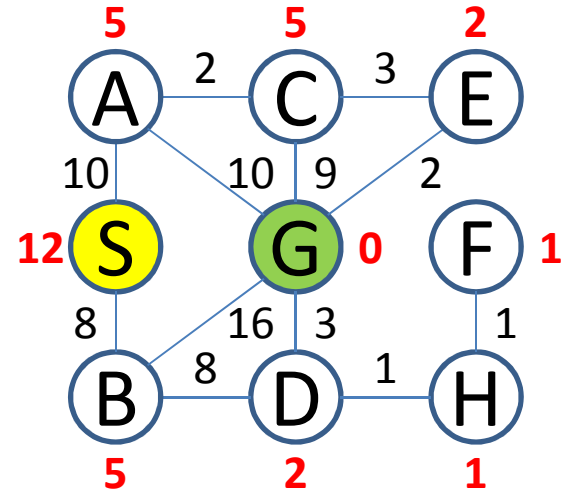
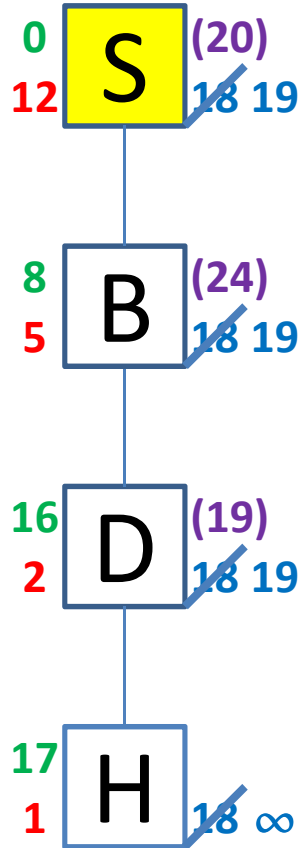
Problem



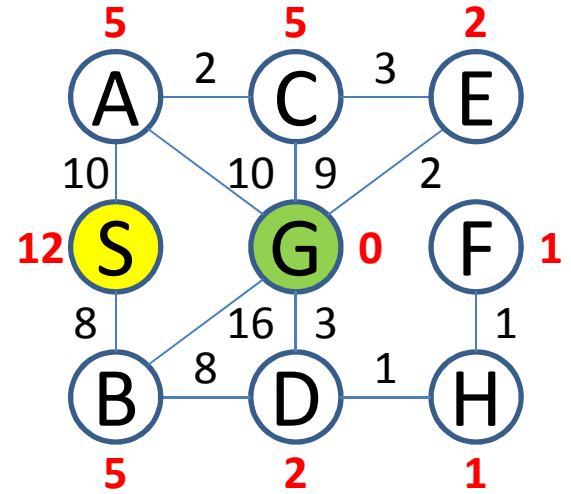
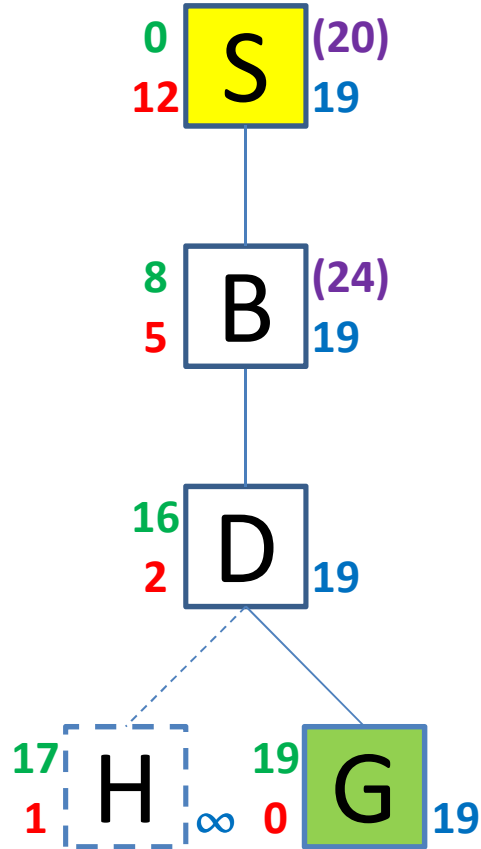
Problem



Problem



Problem



Exercises: Artificial Intelligence

Monotonicity 1

Problem

- Prove that:
 - **IF** a heuristic function h satisfies the *monotonicity restriction*
 - $h(x) \leq \text{cost}(x...y) + h(y)$
 - **THEN** f is *monotonously non-decreasing*
 - $f(s...x) \leq f(s...x...y)$

Monotonicity 1

- *Given:*
 - *h* satisfies the *monotonicity restriction*

- *Proof:*

$$\begin{aligned} f(S...A) &= \text{cost}(S...A) + h(A) \\ &\leq \text{cost}(S...A) + \mathbf{\text{cost}(A...B)} + h(B) \\ &\leq \text{cost}(S...A...B) + h(B) \\ &\leq \mathbf{f(S...A...B)} \end{aligned}$$

Exercises: Artificial Intelligence

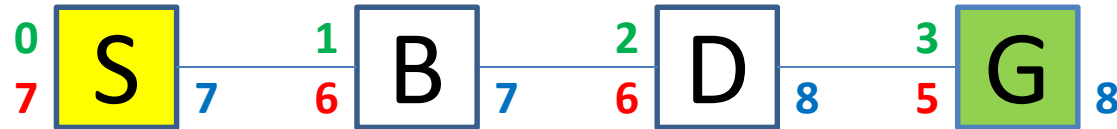
Monotonicity 2

Problem

- Prove or refute:
 - **IF** f is *monotonously non-decreasing*
 - $f(s...x) \leq f(s...xy)$
 - **THEN** h is an *admissible heuristic*
 - h is an underestimate of the remaining path to the goal with the smallest cost
- Can an extra constraint on h change this?

Monotonicity 2

- *Given:*
 - *f* is monotonously non-decreasing
- *Proof (Counter-example):*



f is monotonously non-decreasing,
yet *h* is not an admissible heuristic.

Monotonicity 2

- *Given:*

- f is monotonously non-decreasing

- Extra constraint: $h(G) = 0$

- *Proof:*

$$\underline{f(S...A) \leq f(S...AB) \leq \dots \leq f(S...AB...G)} \Leftrightarrow$$

$$f(S...A) \leq f(S...G) \Leftrightarrow$$

$$\text{cost}(S...A) + h(A) \leq \text{cost}(S...G) + h(G) \Leftrightarrow$$

$$\underline{\text{cost}(S...A)} + h(A) \leq \underline{\text{cost}(S...A)} + \text{cost}(A...G) + h(G) \Leftrightarrow$$

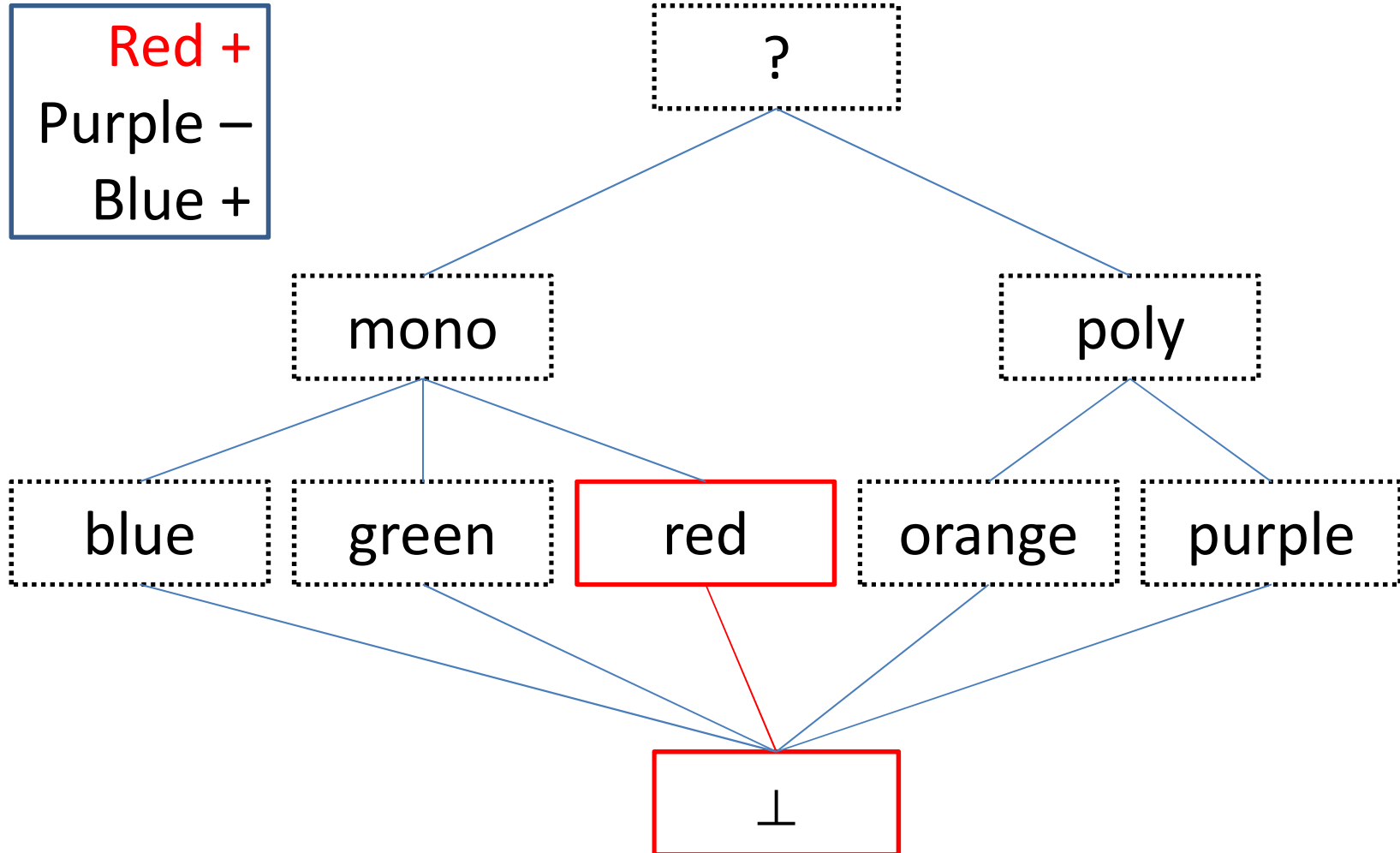
$$h(A) \leq \text{cost}(A...G) + \underline{h(G)} \Leftrightarrow$$

$$h(A) \leq \text{cost}(A...G)$$

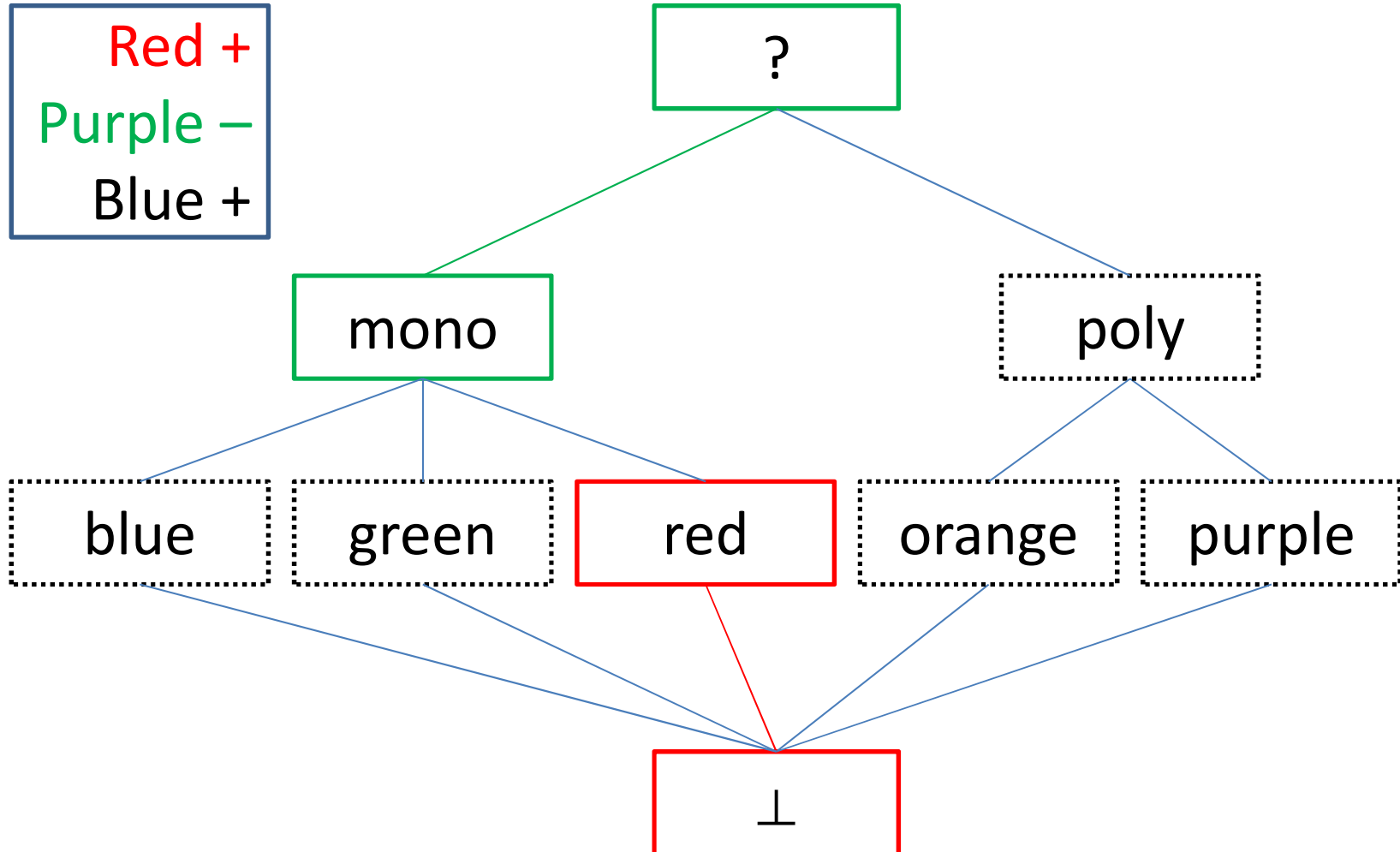
Exercises: Artificial Intelligence

Version Spaces: Colors

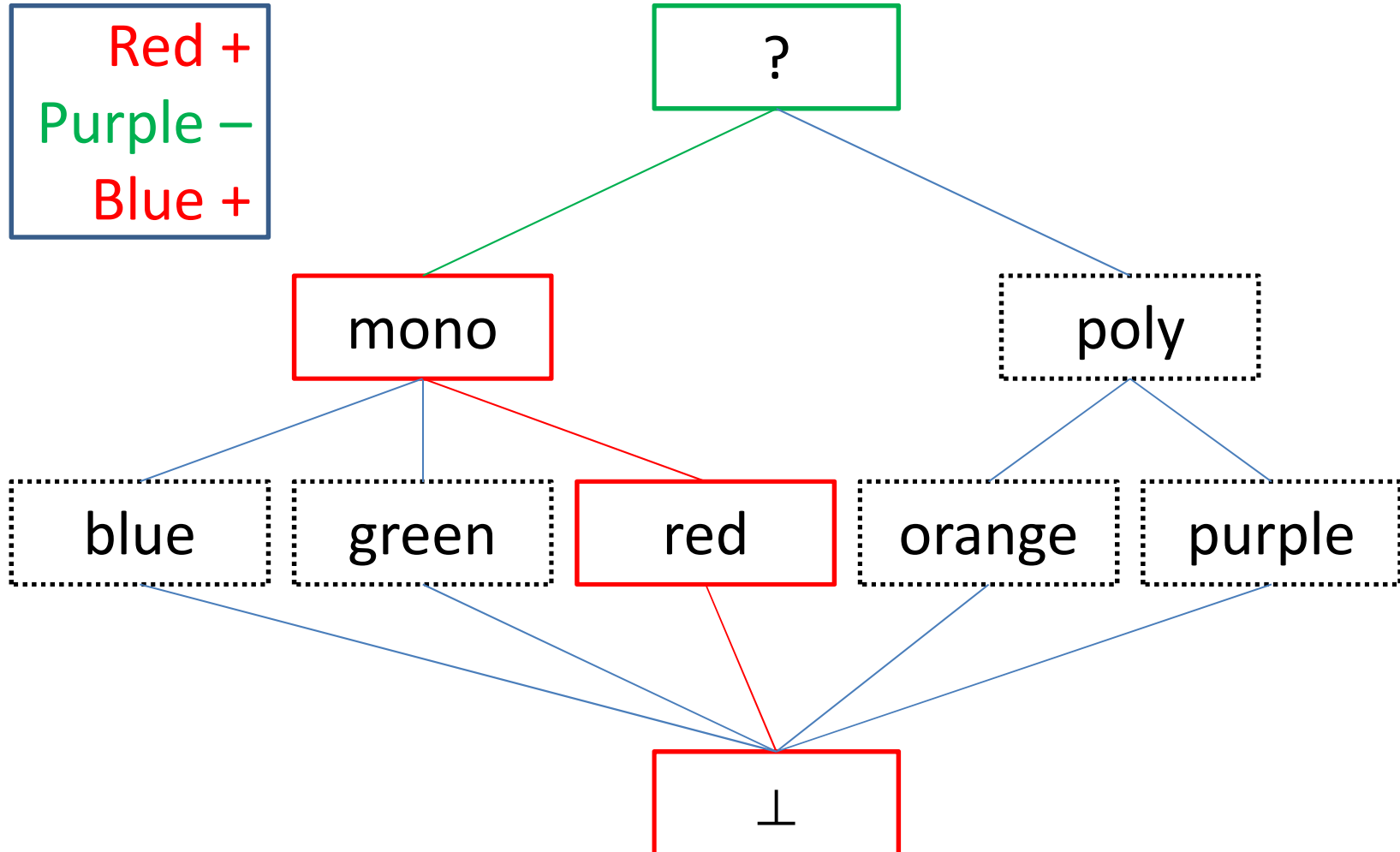
Version-Spaces Algorithm



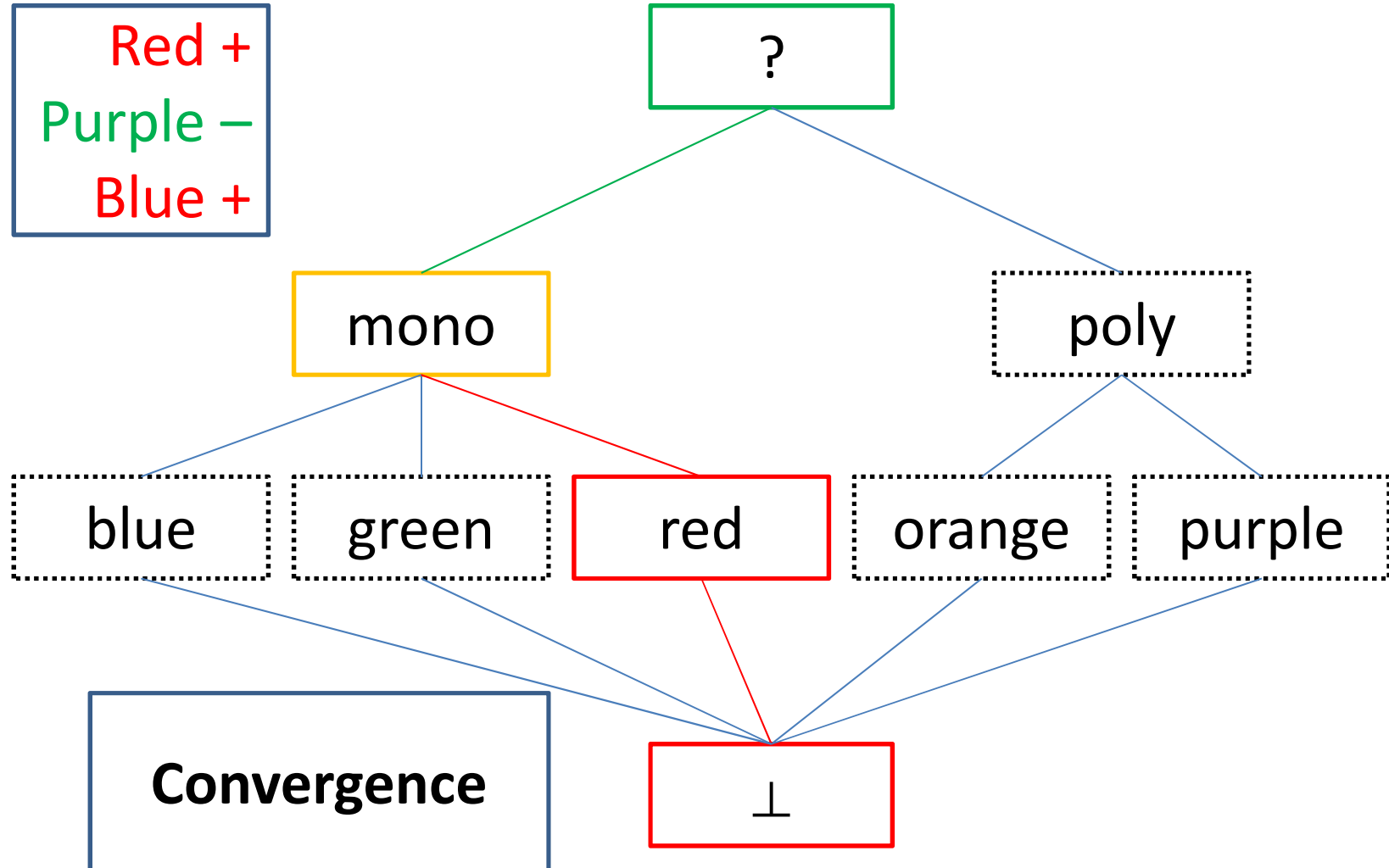
Version-Spaces Algorithm



Version-Spaces Algorithm



Version-Spaces Algorithm



Exercises: Artificial Intelligence

Version Spaces: Playing Cards

Version-Spaces Algorithm

[7,D] +
[A,C] -
[Q,H] -
[9,H] +
[8,C] -

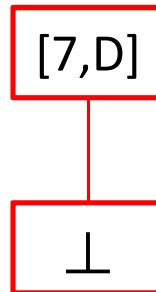
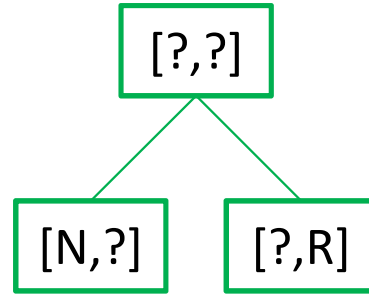
[?,?]

[7,D]

⊥

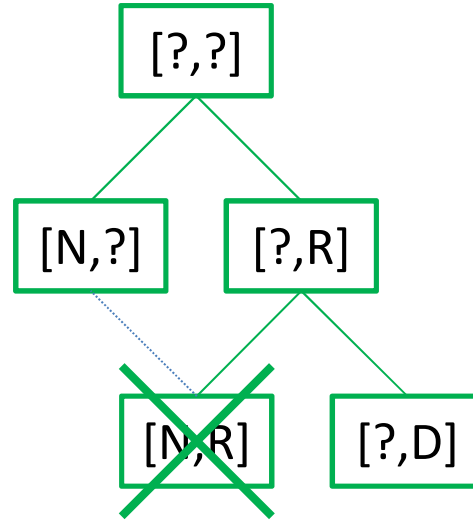
Version-Spaces Algorithm

[7,D] +
[A,C] -
[Q,H] -
[9,H] +
[8,C] -

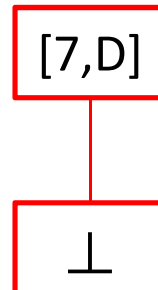


Version-Spaces Algorithm

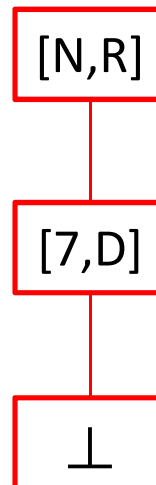
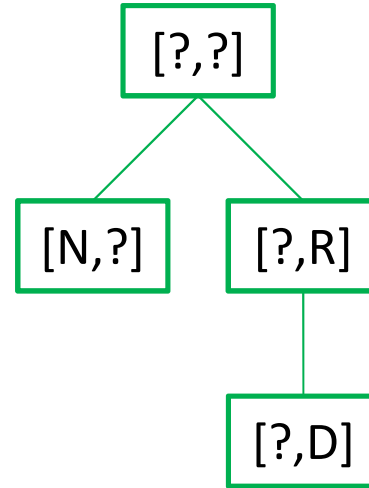
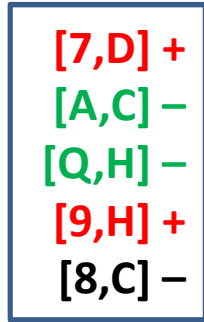
[7,D] +
[A,C] -
[Q,H] -
[9,H] +
[8,C] -



**Redundant
Hypotheses**

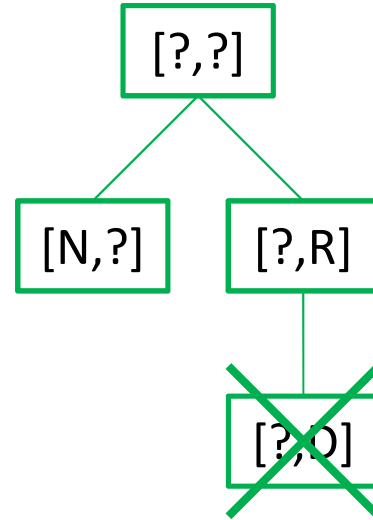


Version-Spaces Algorithm

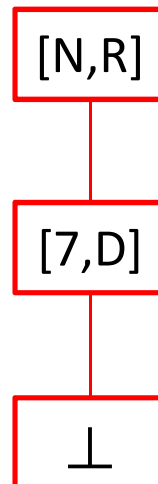


Version-Spaces Algorithm

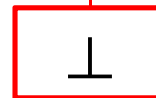
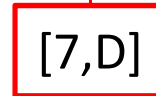
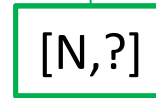
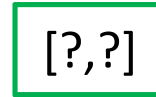
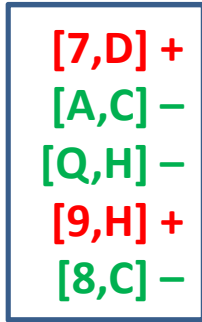
[7,D] +
[A,C] -
[Q,H] -
[9,H] +
[8,C] -



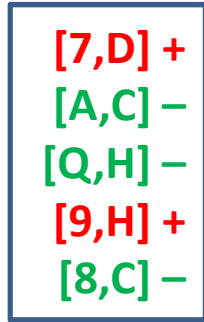
**does not cover
last positive
example**



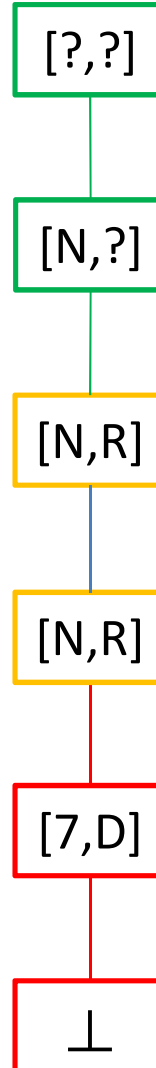
Version-Spaces Algorithm



Version-Spaces Algorithm



Convergence



Exercises: Artificial Intelligence

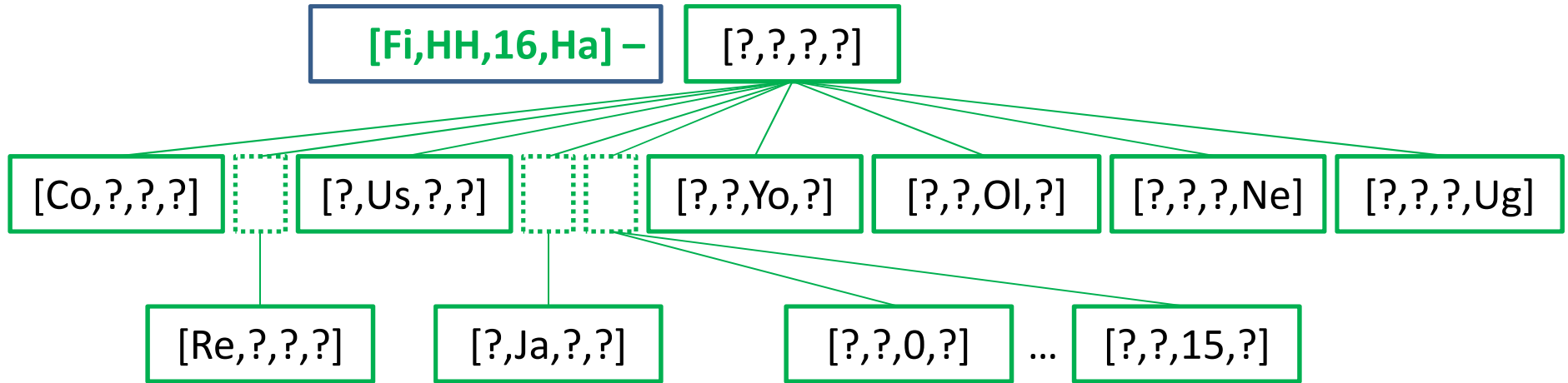
Version Spaces: Ex-exam

Version-Spaces Algorithm

[?,?,?,?]

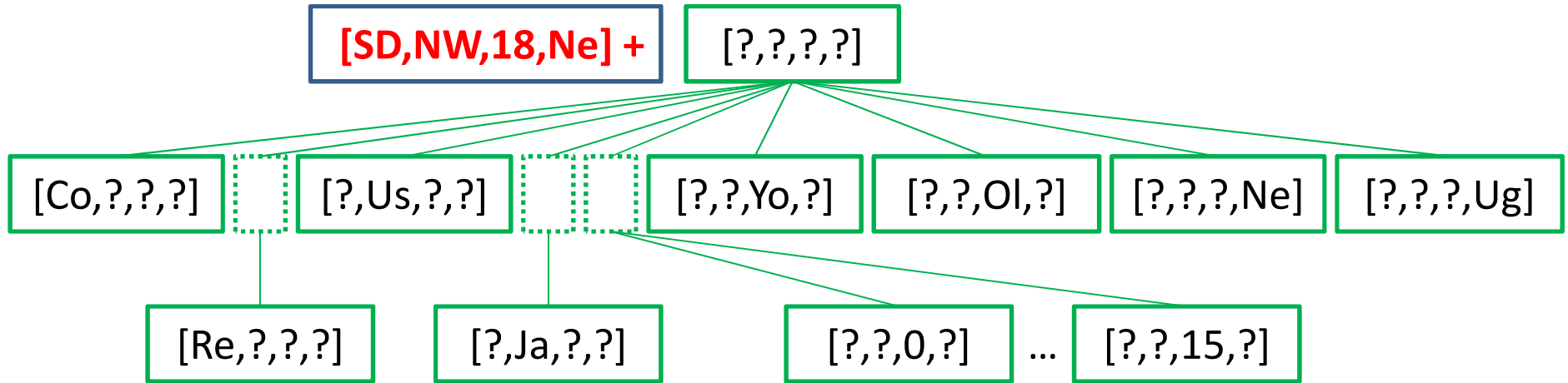
⊥

Version-Spaces Algorithm



⊥

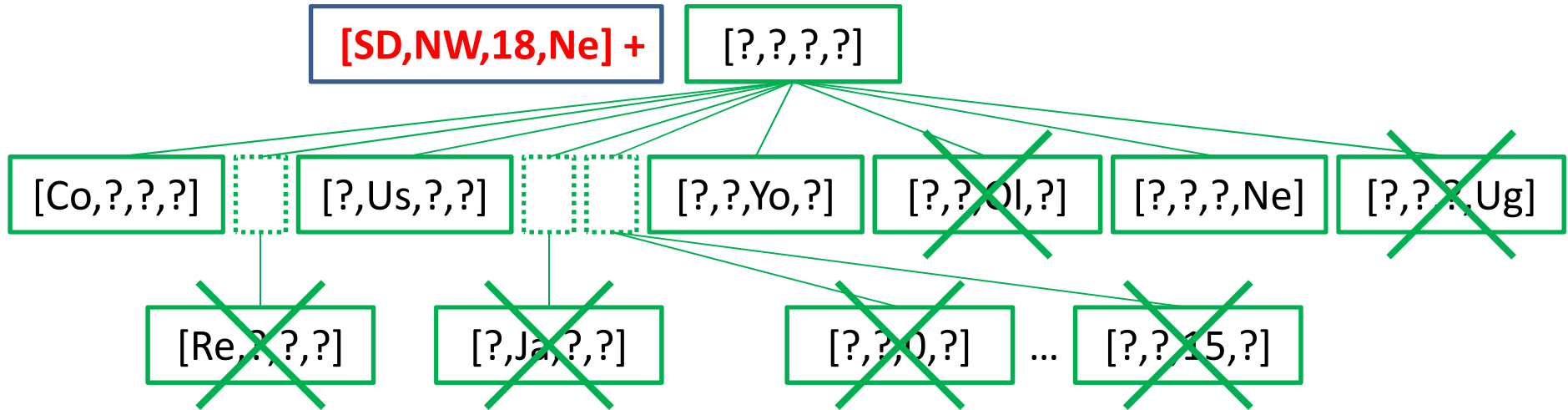
Version-Spaces Algorithm



$[SD, NW, 18, Ne]$

\perp

Version-Spaces Algorithm

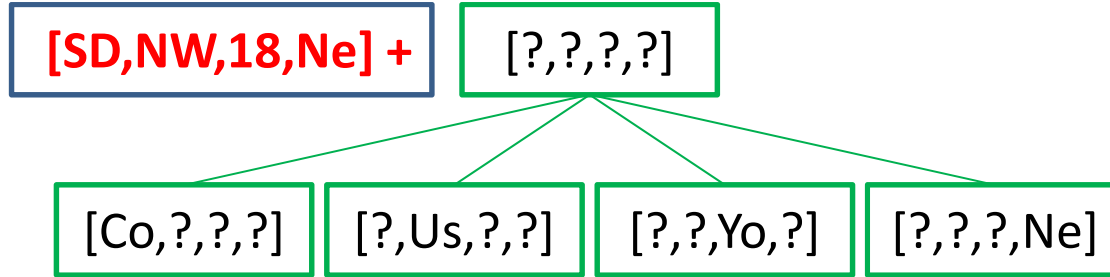


19 out of 23 do not cover last positive example

[SD,NW,18,Ne]

⊥

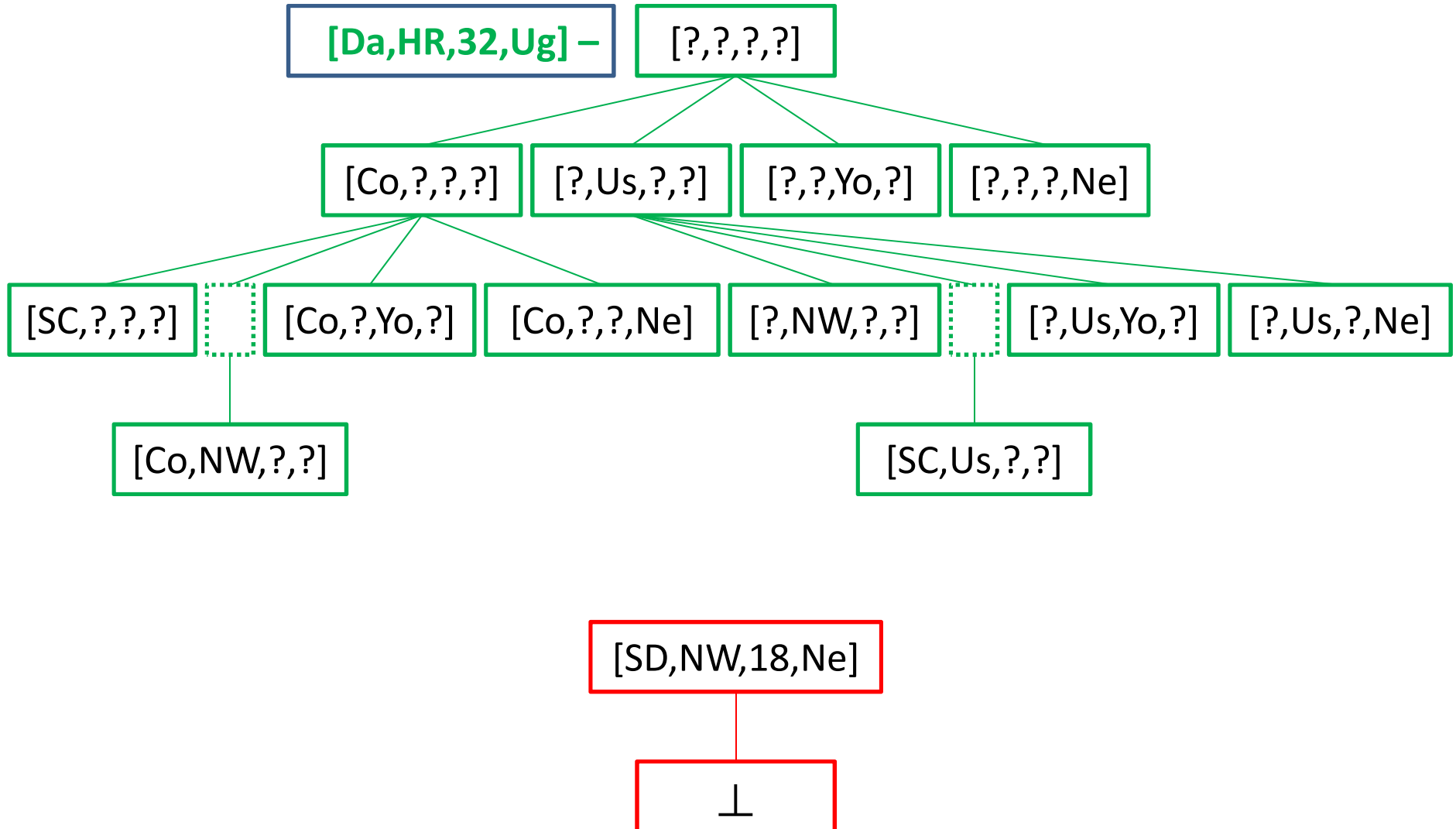
Version-Spaces Algorithm



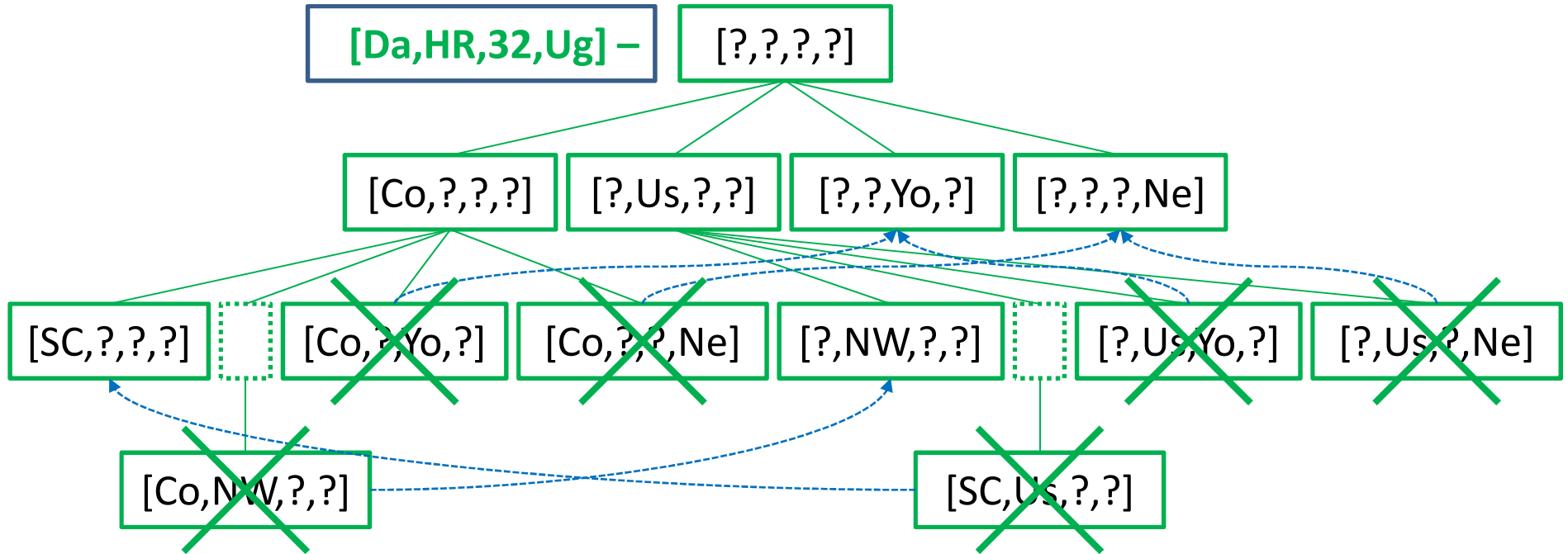
[SD,NW,18,Ne]

⊥

Version-Spaces Algorithm



Version-Spaces Algorithm

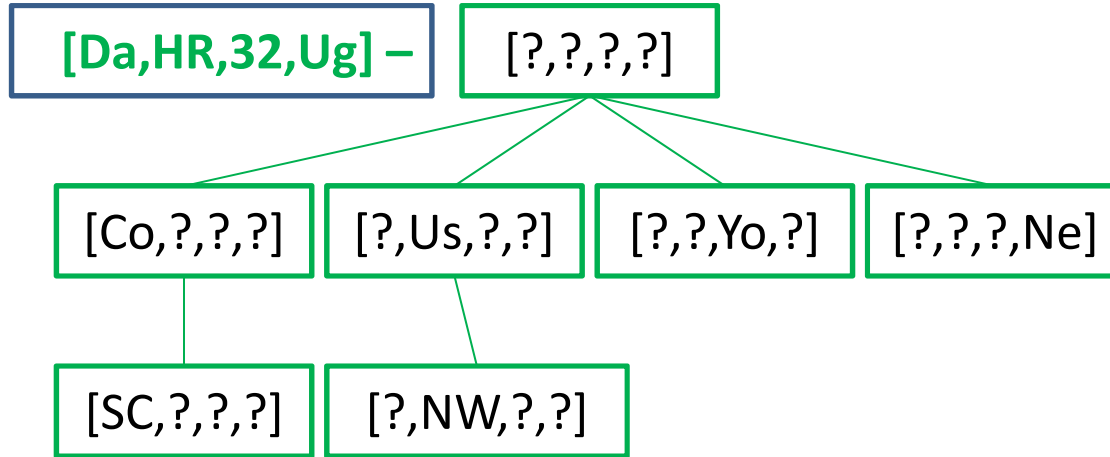


Redundant hypotheses

[SD,NW,18,Ne]

⊥

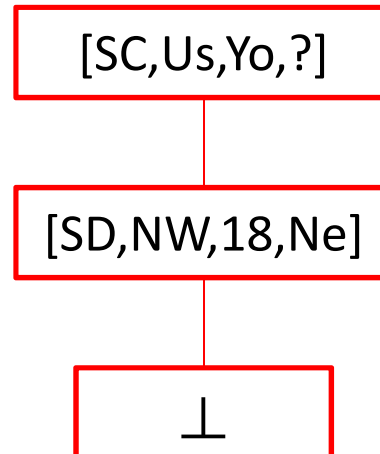
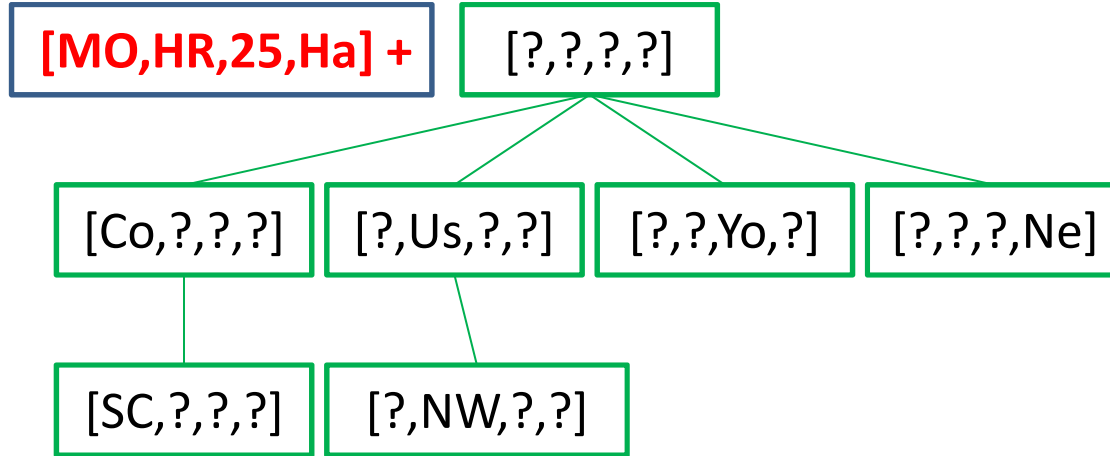
Version-Spaces Algorithm



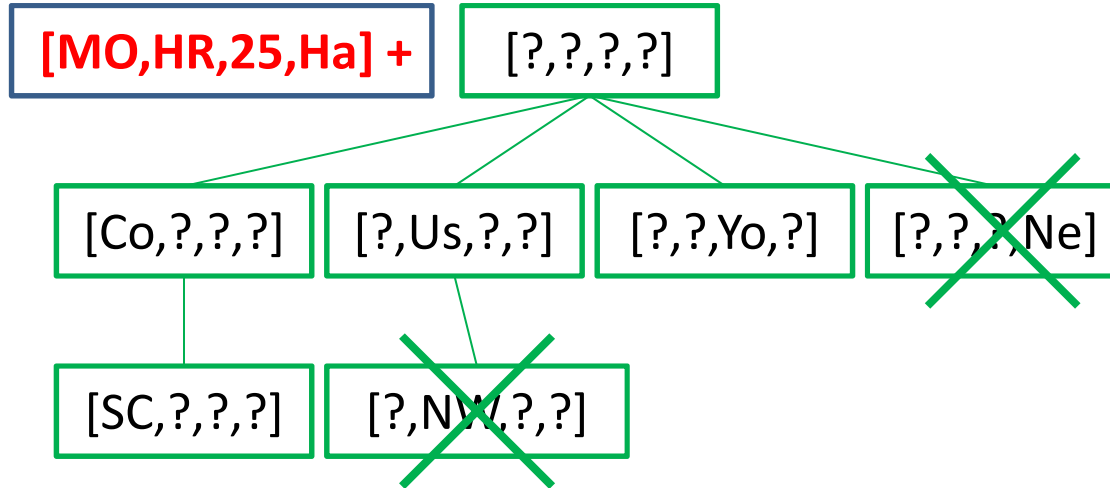
[SD,NW,18,Ne]

⊥

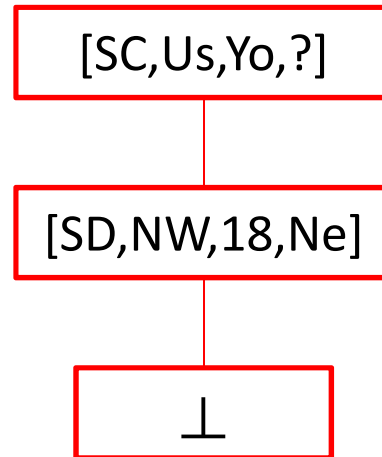
Version-Spaces Algorithm



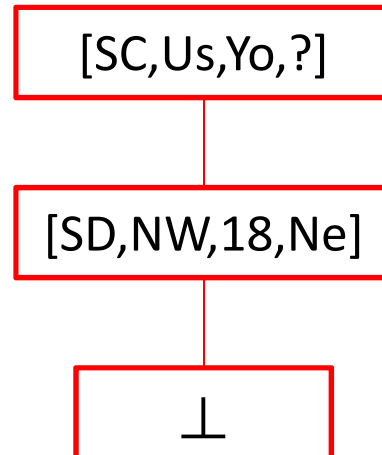
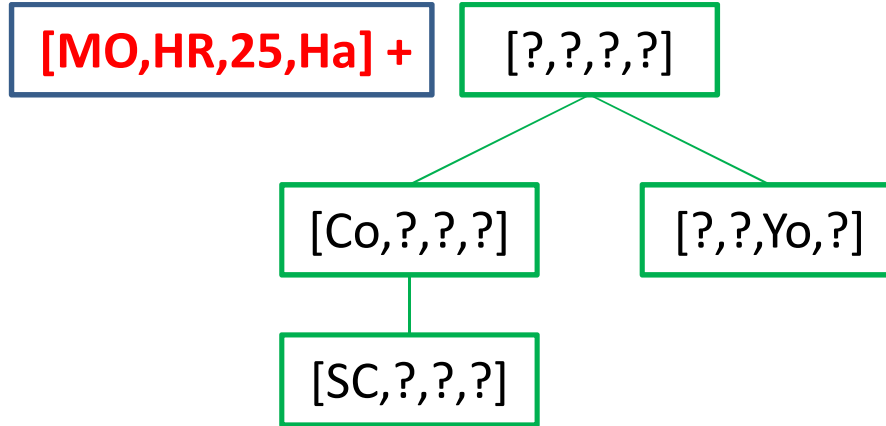
Version-Spaces Algorithm



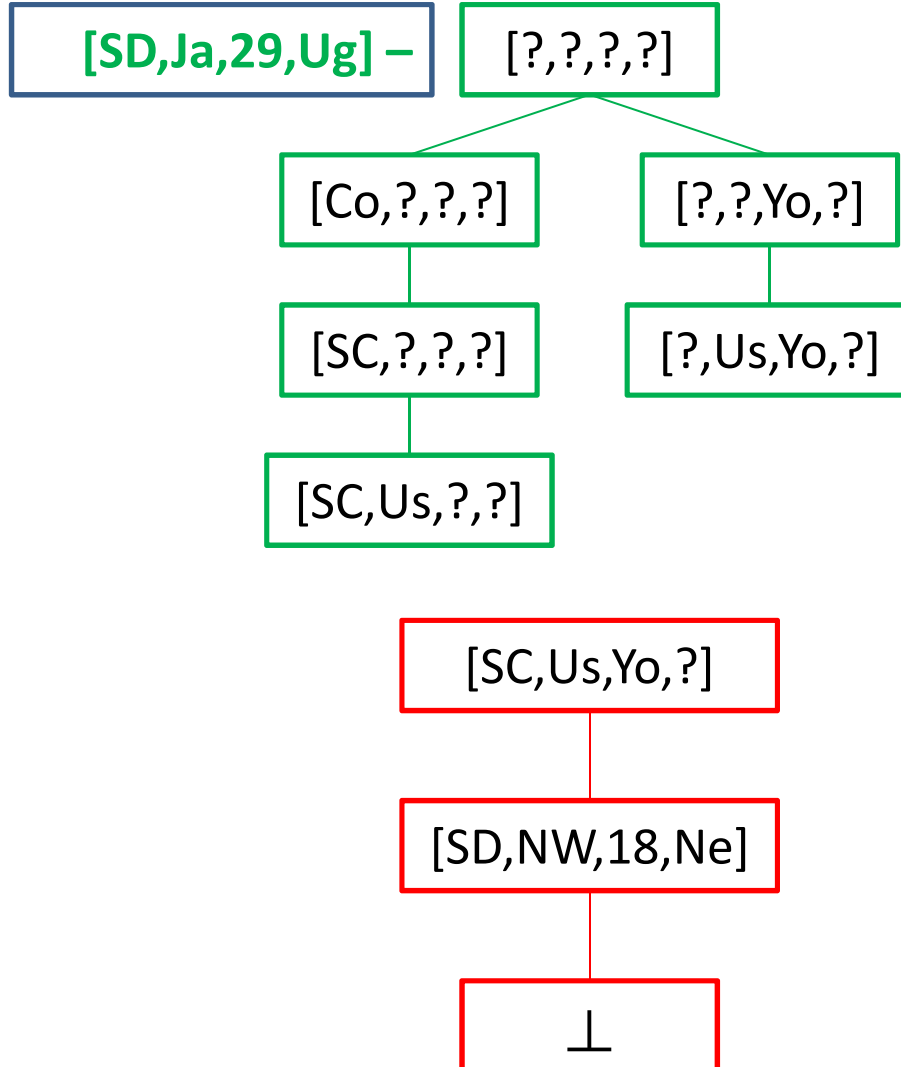
2 out of 4 do not cover last positive example



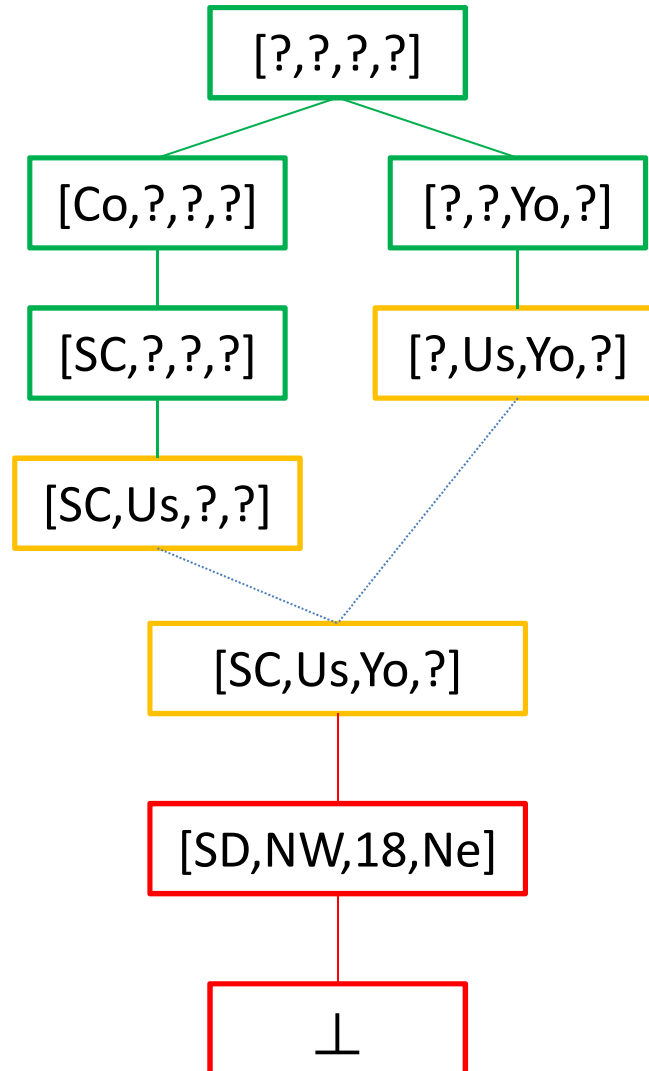
Version-Spaces Algorithm



Version-Spaces Algorithm

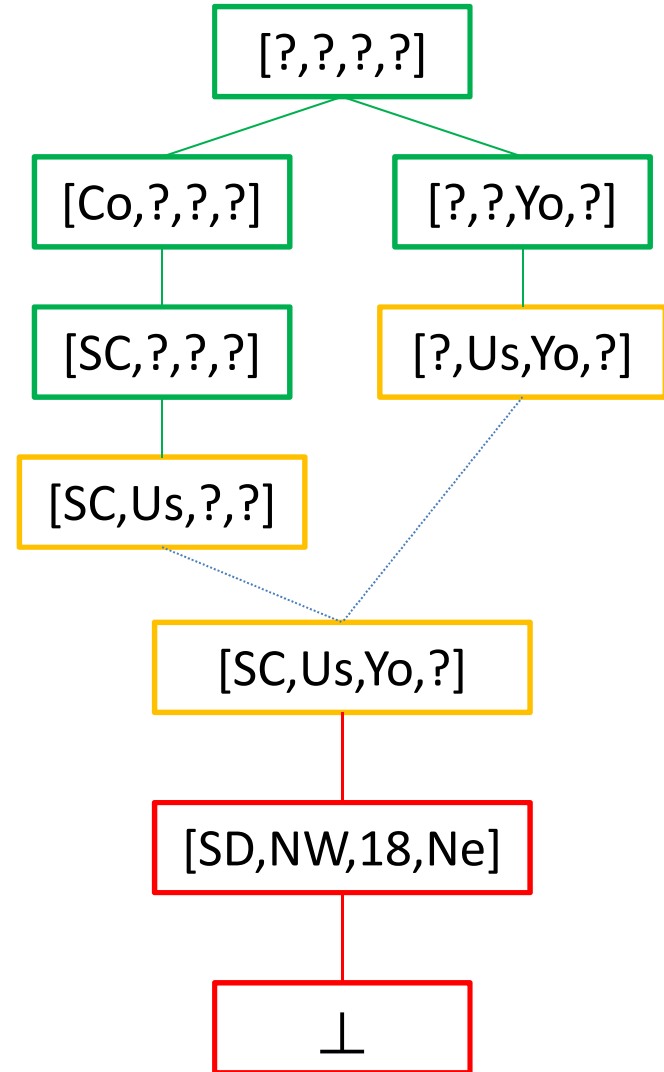


Version-Spaces Algorithm



Using the result

- [MO,HR,32,Ha]: **Maybe**
 - More Specific than [SC,Us,?,?]
 - Not more Specific than [SC,Us,Yo,?]
- [SD,HH,18,Ne]: **NO**
 - Not More Specific than [SC,Us,?,?]
 - Not More Specific than [?,Us,Yo,?]
- [Da,NW,22,Ug]: **Maybe**
 - More Specific than [?,Us,Yo,?]
 - Not more Specific than [SC,Us,Yo,?]



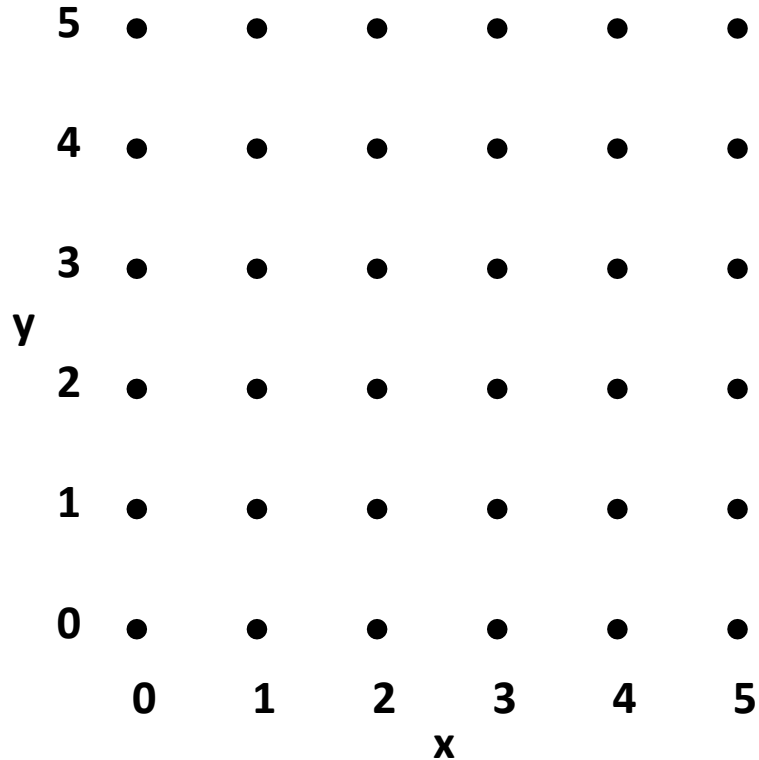
Exercises: Artificial Intelligence

Version Spaces: Computer Screen

Version-Spaces Algorithm

$G = \{[(0,0),5], \text{white}\}$

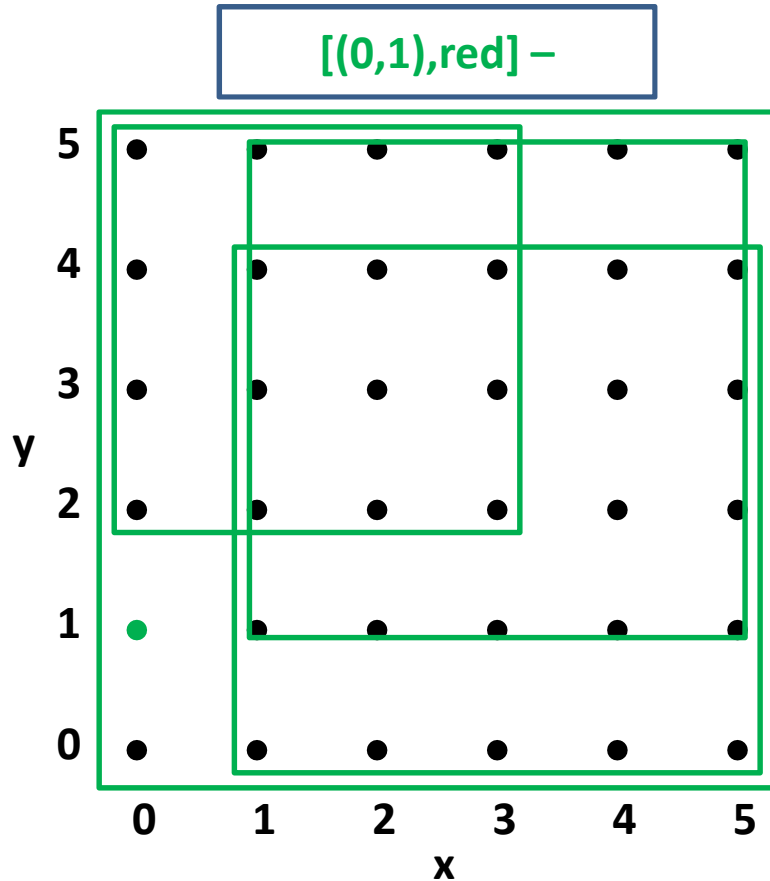
$S = \{\perp\}$



Version-Spaces Algorithm

$G = \{[(0,0),5], \text{white}\}$

$S = \{\perp\}$



$G = \{$

$[(0,2),3], \text{white},$

$[(1,0),4], \text{white},$

$[(1,1),4], \text{white},$

$[(0,0),5], \text{cyan}$

$\}$

Redundant:

$[(0,0),5], \text{green}$

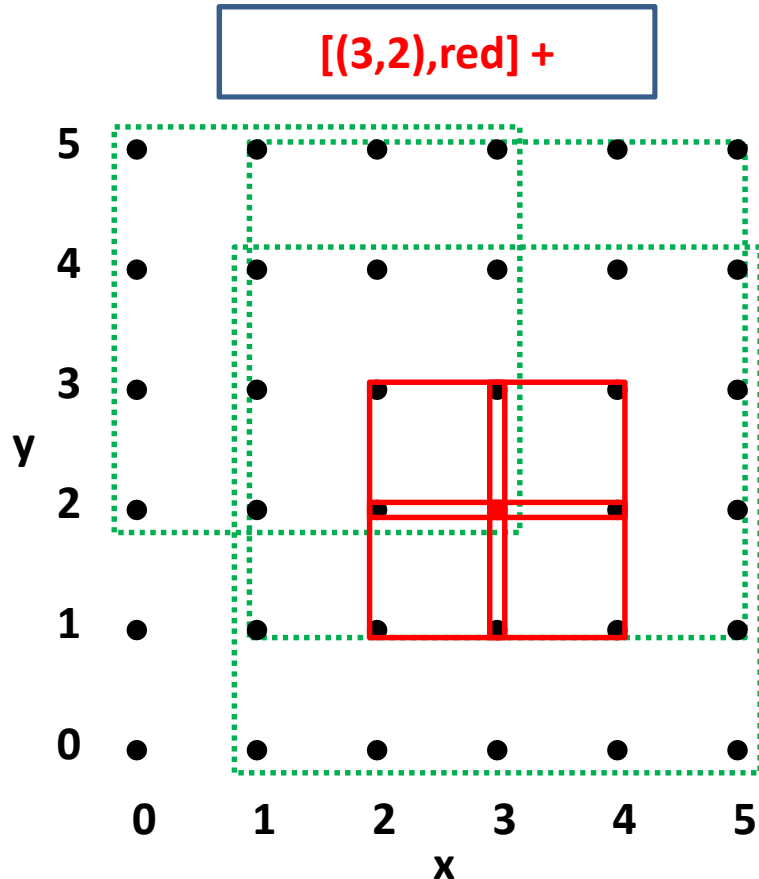
$[(0,0),5], \text{blue}$

$S = \{\perp\}$

Version-Spaces Algorithm

$G = \{ [((0,2),3),white], [((1,0),4),white], [((1,1),4),white], [((0,0),5),cyan] \}$

$S = \{\perp\}$



$G = \{$
 $[((0,2),3),white],$
 $[((1,0),4),white],$
 $[((1,1),4),white]$
 $\}$

Removed:

$[((0,0),5),cyan]$

$S = \{$

$[((2,1),1),red],$

$[((2,2),1),red],$

$[((3,1),1),red],$

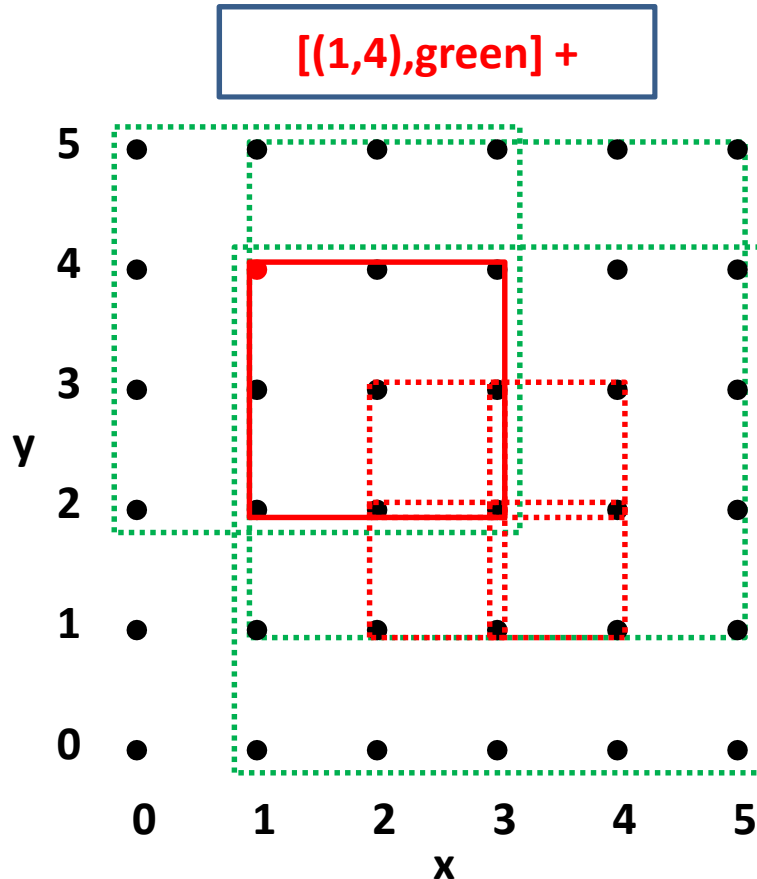
$[((3,2),1),red]$

$\}$

Version-Spaces Algorithm

$G = \{ [((0,2),3),\text{white}], [((1,0),4),\text{white}], [((1,1),4),\text{white}] \}$

$S = \{ [((2,1),1),\text{red}], [((2,2),1),\text{red}], [((3,1),1),\text{red}], [((3,2),1),\text{red}] \}$



$G = \{$

$[((0,2),3),\text{white}],$
 $[((1,0),4),\text{white}],$
 $[((1,1),4),\text{white}]$

$\}$

$S = \{$

$[((1,2),2),\text{yellow}]$

$\}$

Redundant:

$[((0,2),3),\text{yellow}]$

$[((1,2),3),\text{yellow}]$

$[((1,1),3),\text{yellow}]$

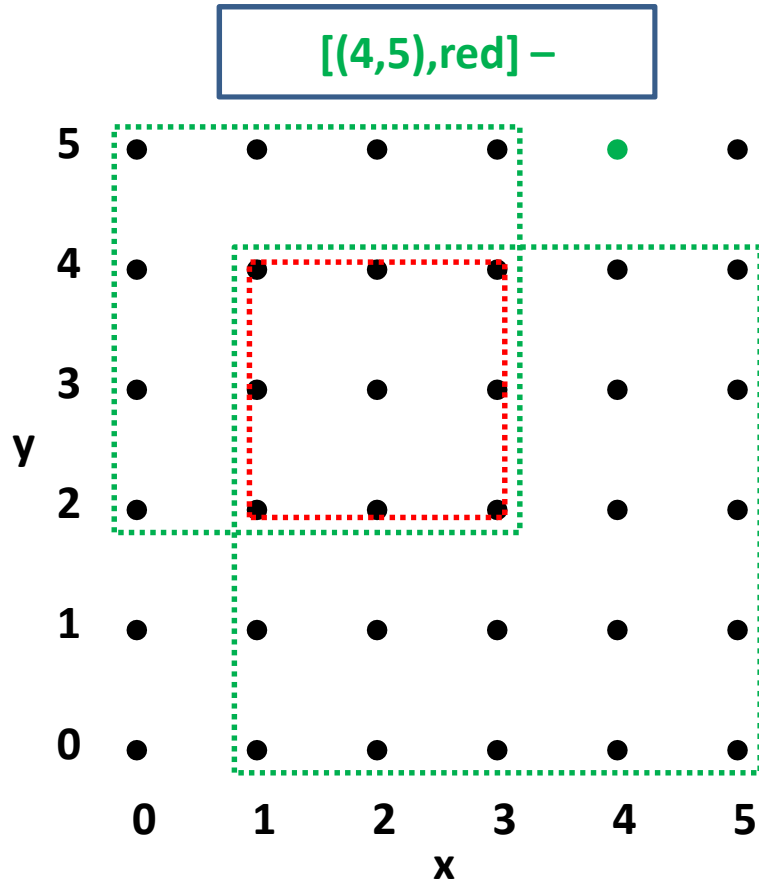
$[((1,1),4),\text{yellow}]$

$[((1,0),4),\text{yellow}]$

Version-Spaces Algorithm

$G = \{ [((0,2),3),\text{white}], [((1,0),4),\text{white}], [((1,1),4),\text{white}] \}$

$S = \{ [((1,2),2),\text{yellow}] \}$



$G = \{$
 $[((0,2),3),\text{white}],$
 $[((1,0),4),\text{white}]$
 $\}$

Redundant:

$[((1,1),3),\text{white}]$

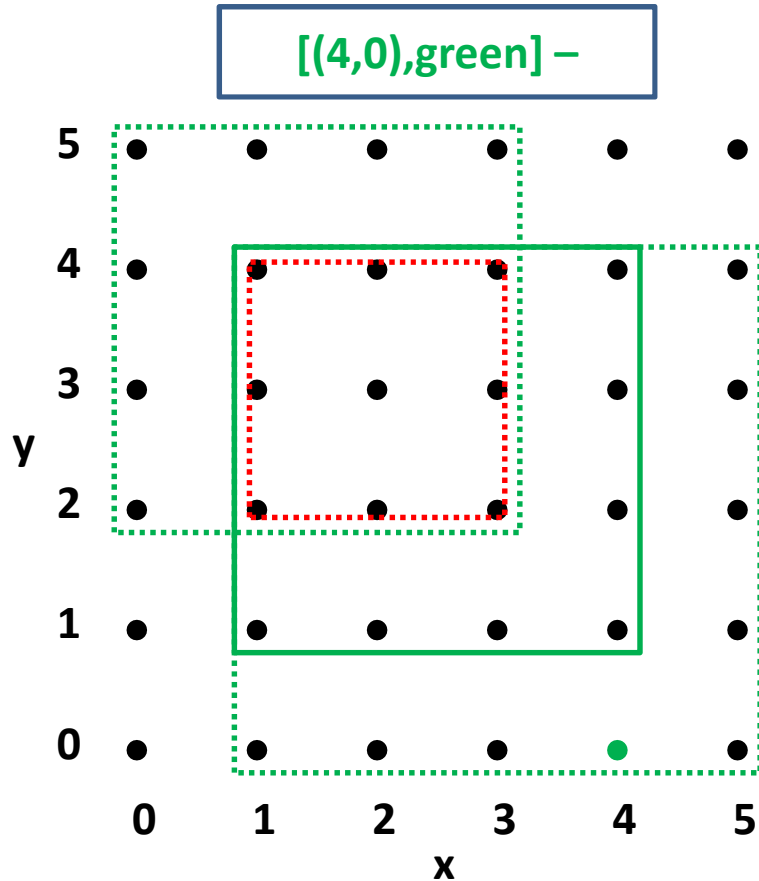
Others don't generalize S

$S = \{$
 $[((1,2),2),\text{yellow}]$
 $\}$

Version-Spaces Algorithm

$G = \{ [((0,2),3),\text{white}], [((1,0),4),\text{white}] \}$

$S = \{ [((1,2),2),\text{yellow}] \}$

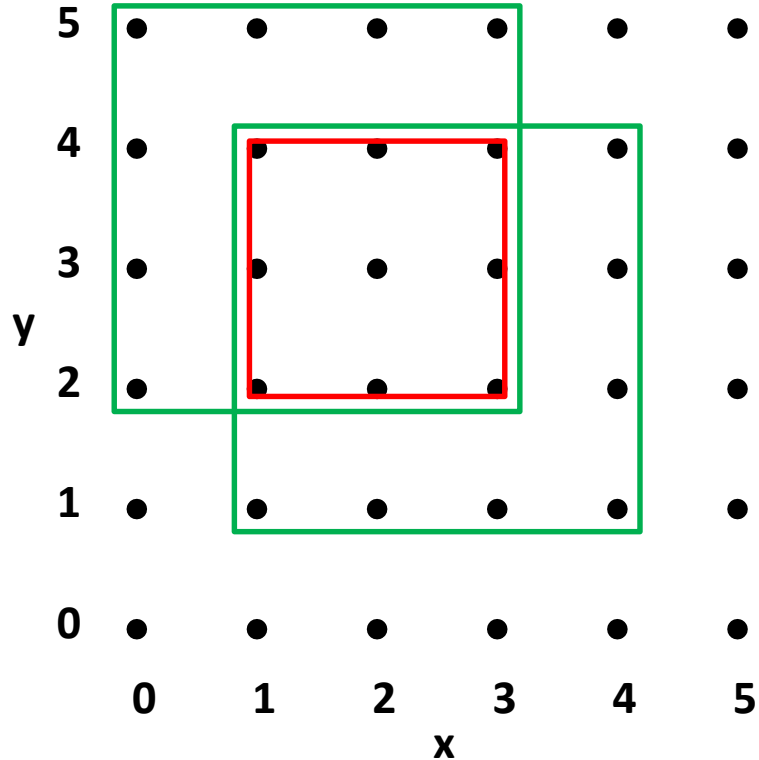


$G = \{$
 $[((0,2),3),\text{white}],$
 $[((1,1),3),\text{white}]$
 $\}$
Others don't generalize S
 $S = \{$
 $[((1,2),2),\text{yellow}]$
 $\}$

Version-Spaces Algorithm

$G = \{[(0,2),3,white],[(1,1),3,white]\}$

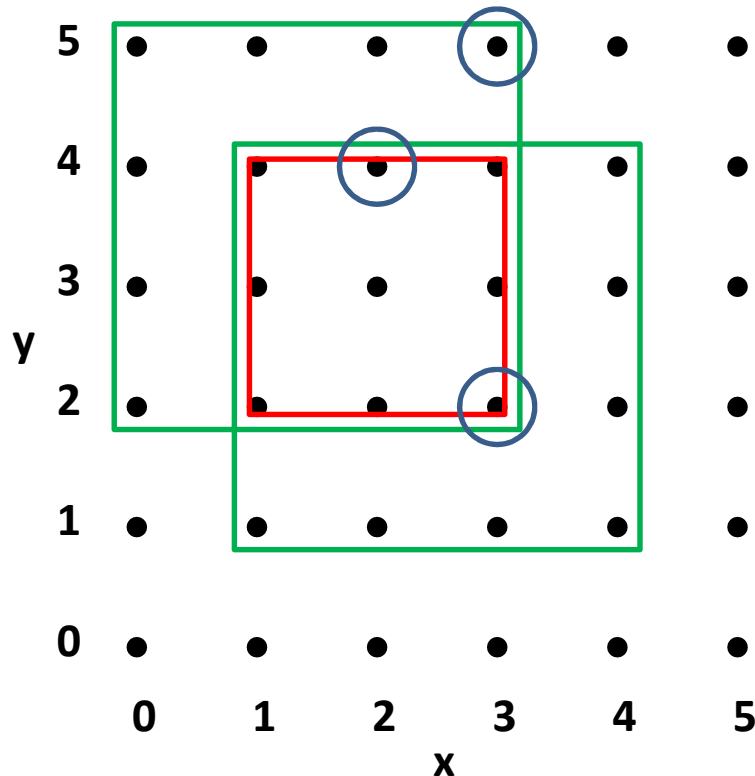
$S = \{[(1,2),2,yellow]\}$



Using the Result

$G = \{[(0,2),3,white],[(1,1),3,white]\}$

$S = \{[(1,2),2,yellow]\}$



$[(3,2),green]$

Yes

$[(2,4),red]$

Yes

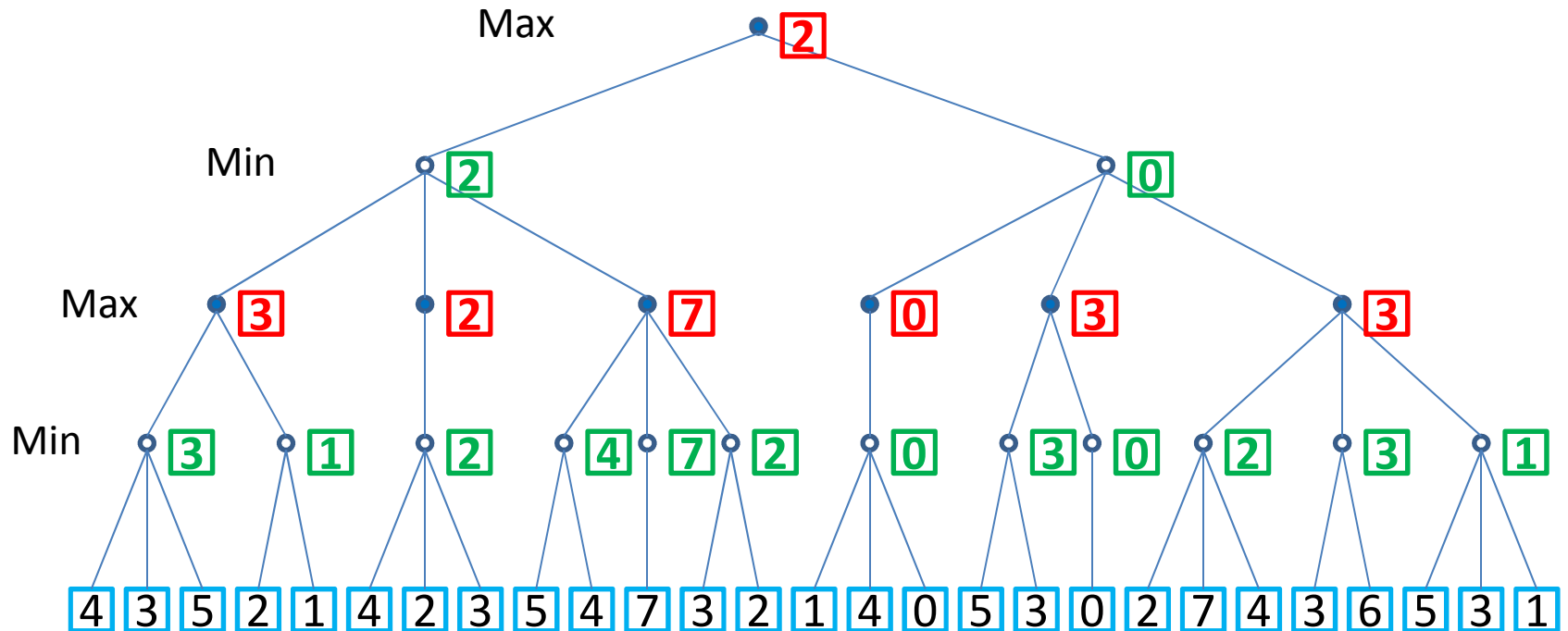
$[(3,5),blue]$

Maybe

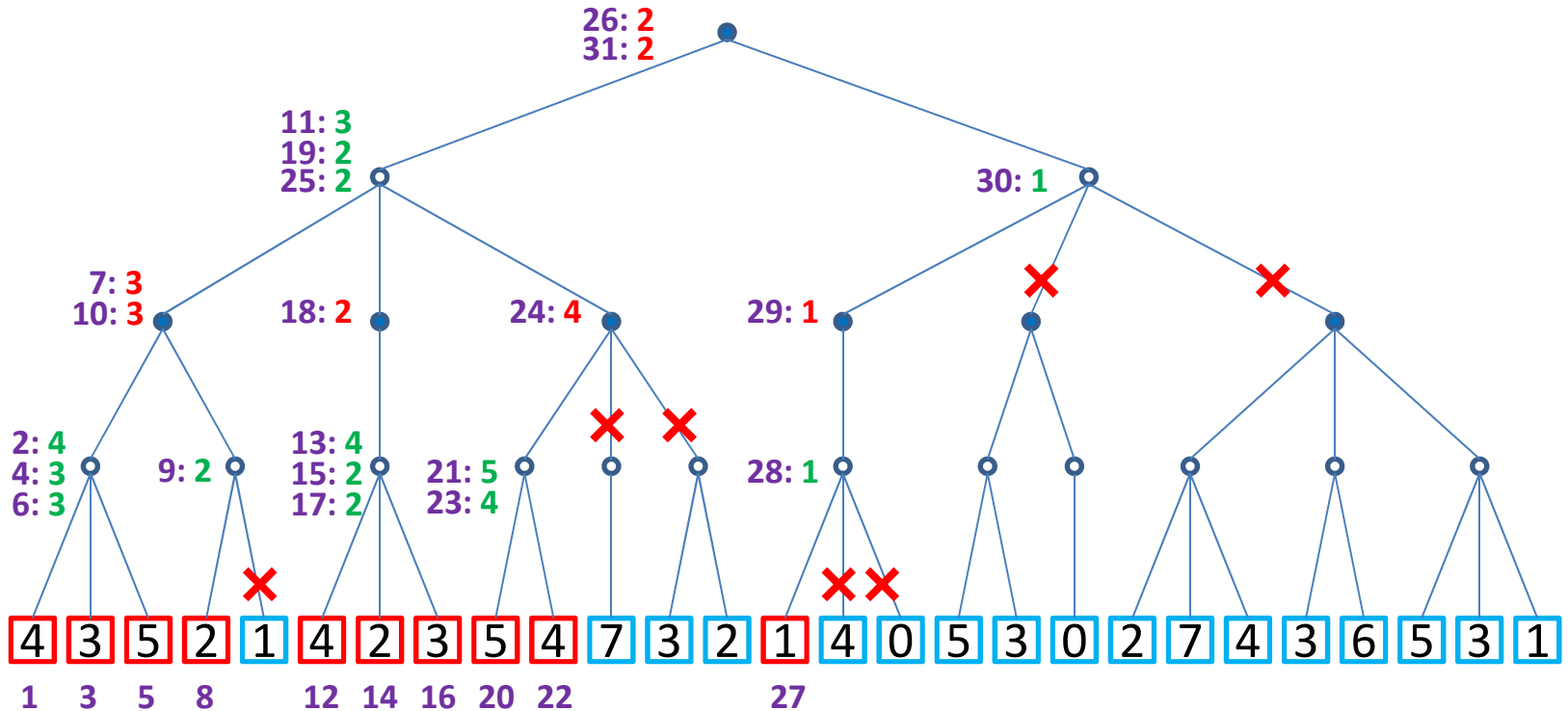
Exercises: Artificial Intelligence

MiniMax & Constraint Processing:
MiniMax Algorithm

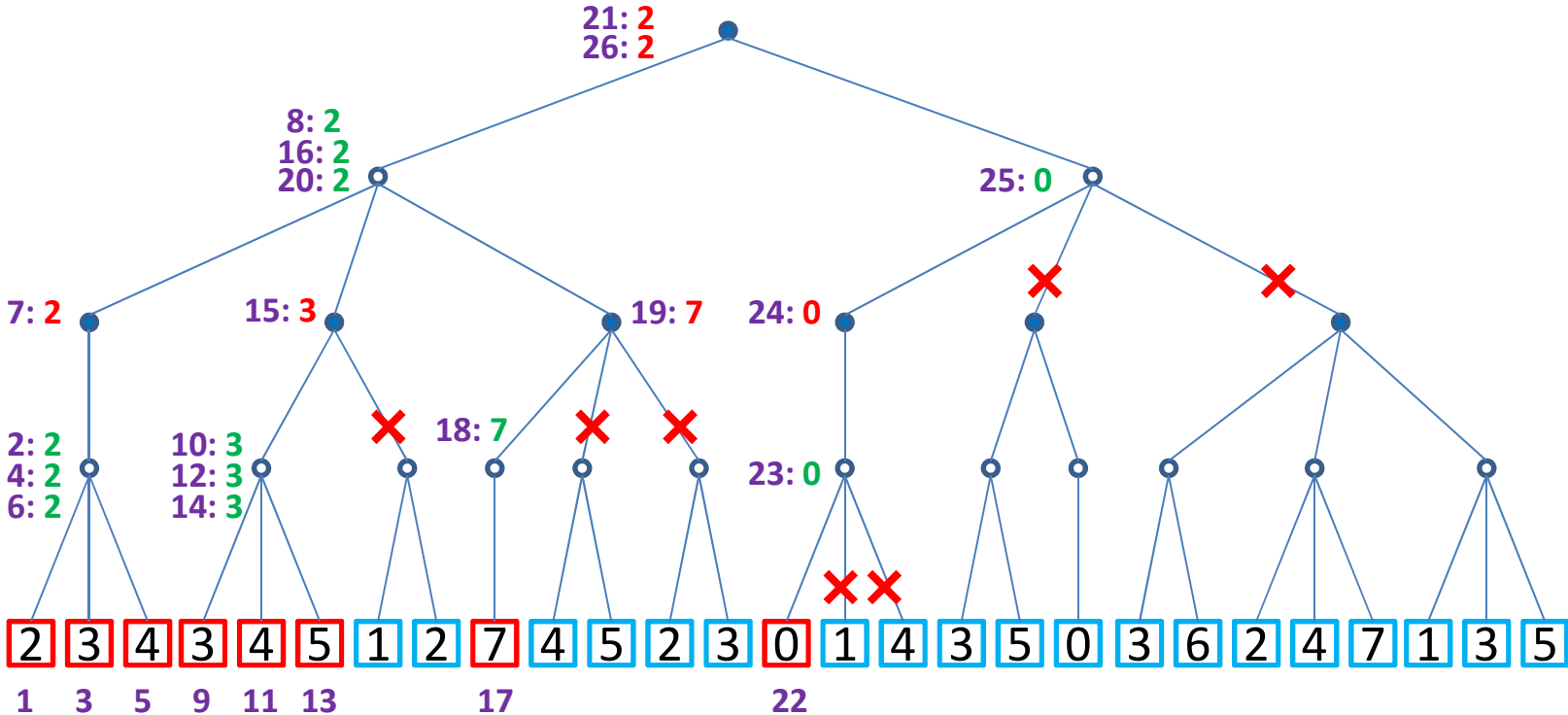
MiniMax without $\alpha\beta$ -pruning



MiniMax with $\alpha\beta$ -pruning



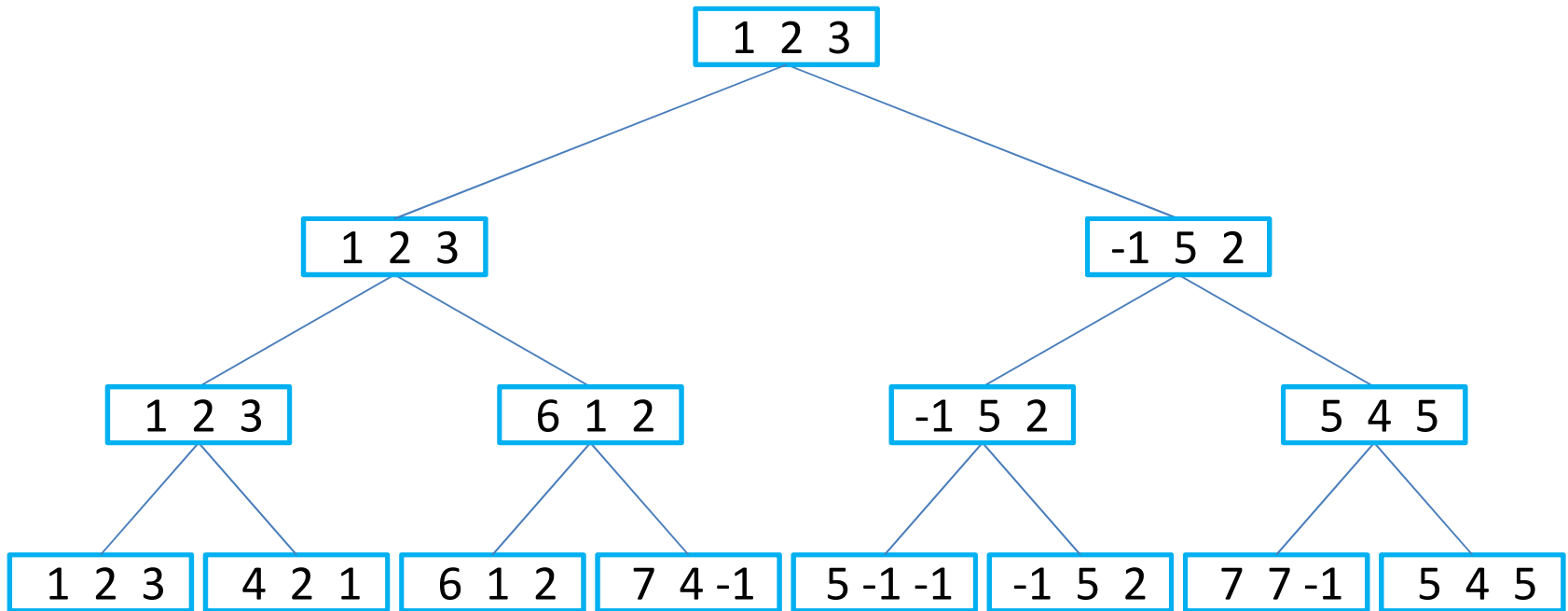
Reordering, MiniMax with $\alpha\beta$ -Pruning



Exercises: Artificial Intelligence

MiniMax & Constraint Processing:
MiniMax Algorithm for 3 Players

MiniMax For 3 Players

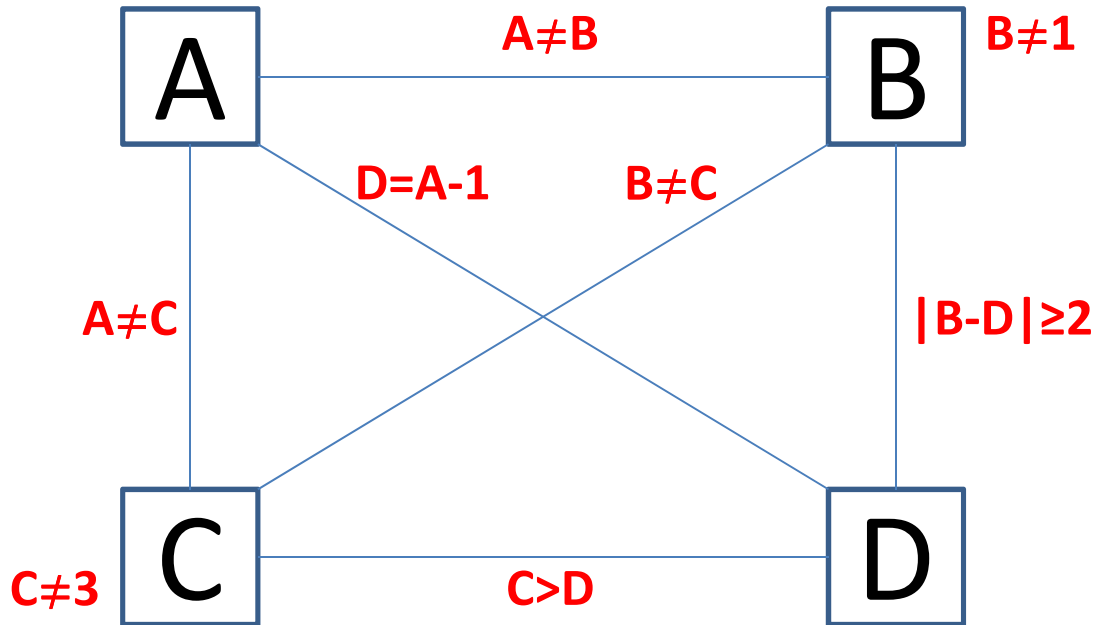


Exercises: Artificial Intelligence

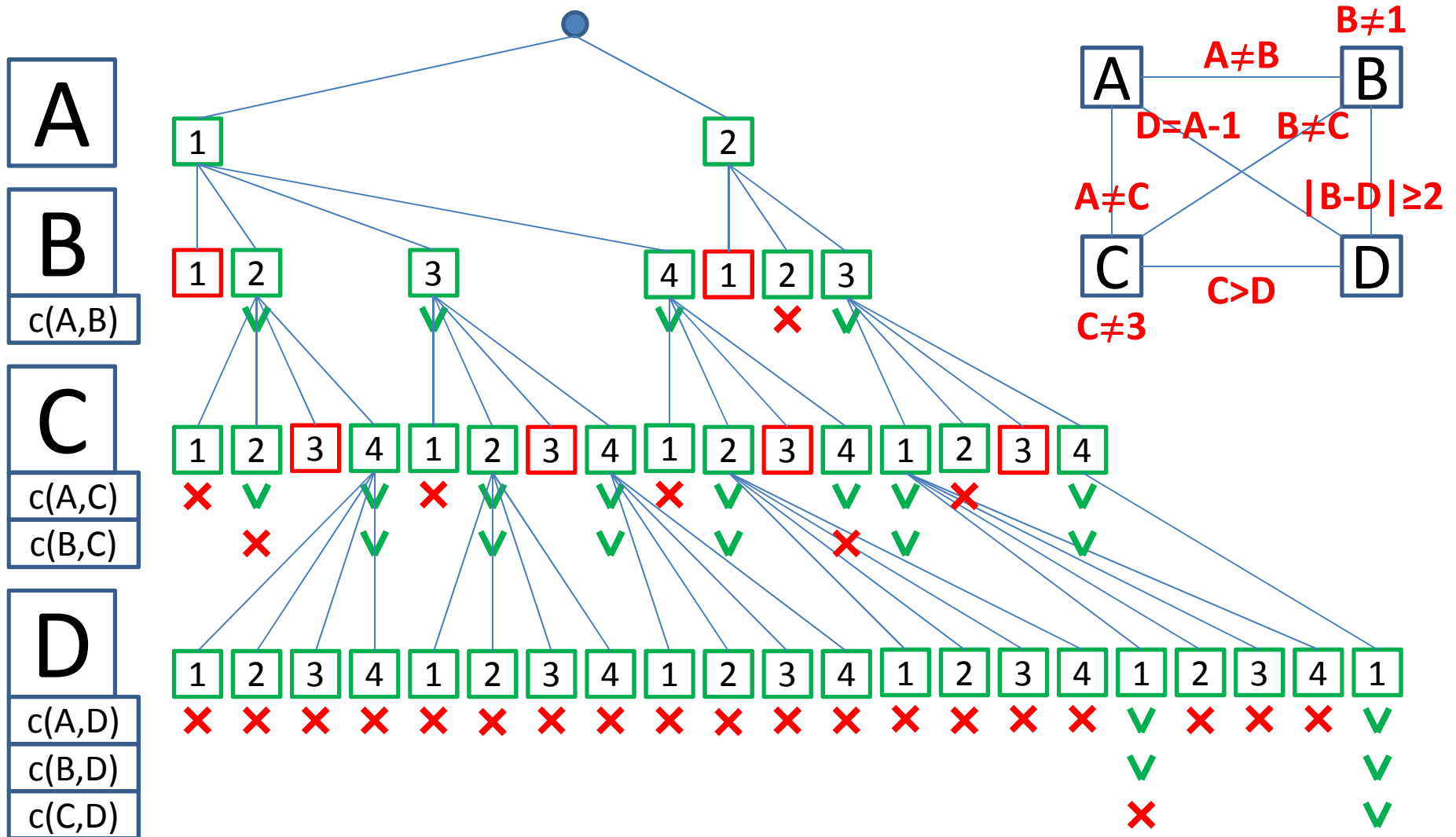
MiniMax & Constraint Processing:
The 4 Houses problem

Constraint Processing

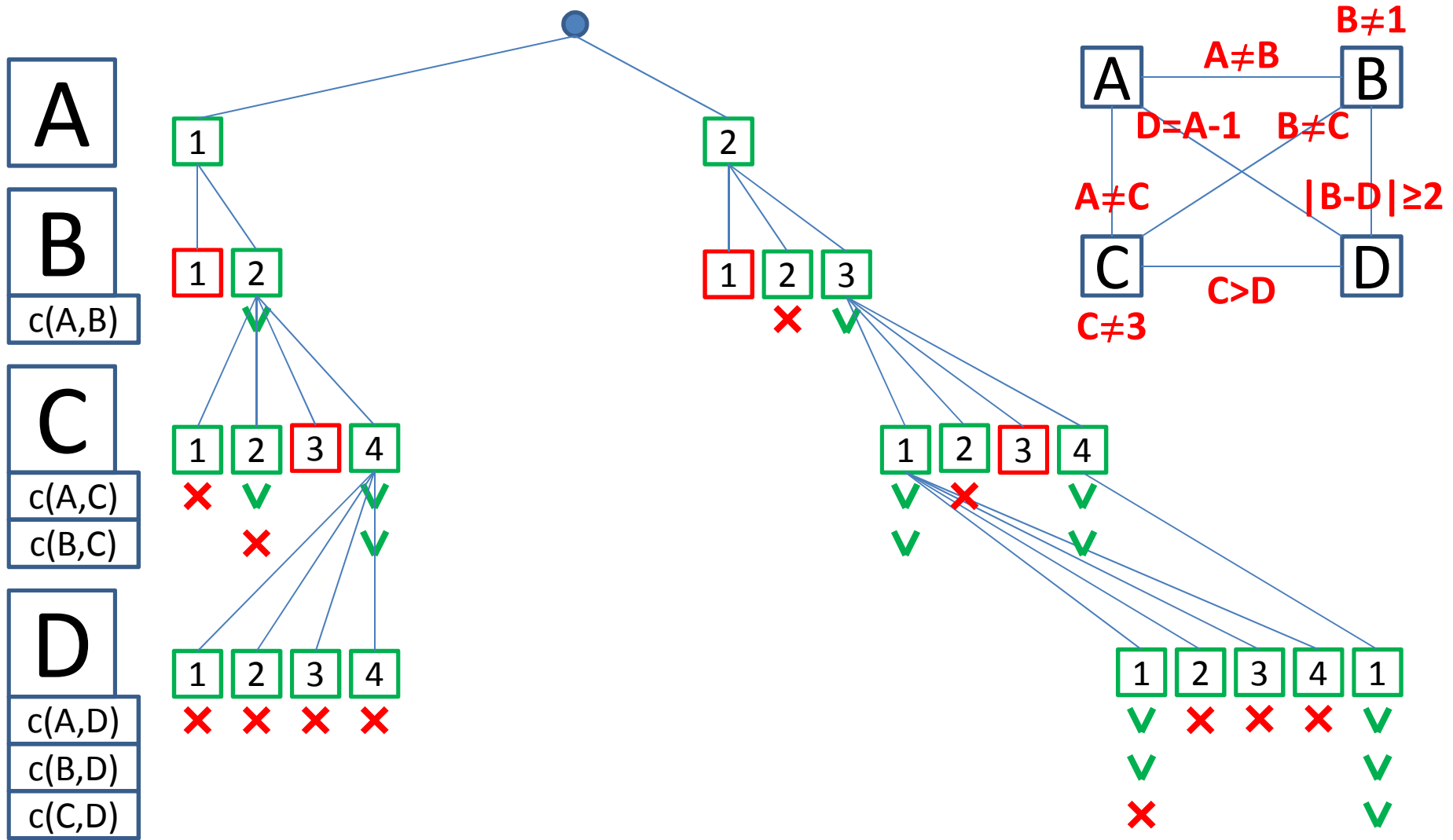
- Problem representation:



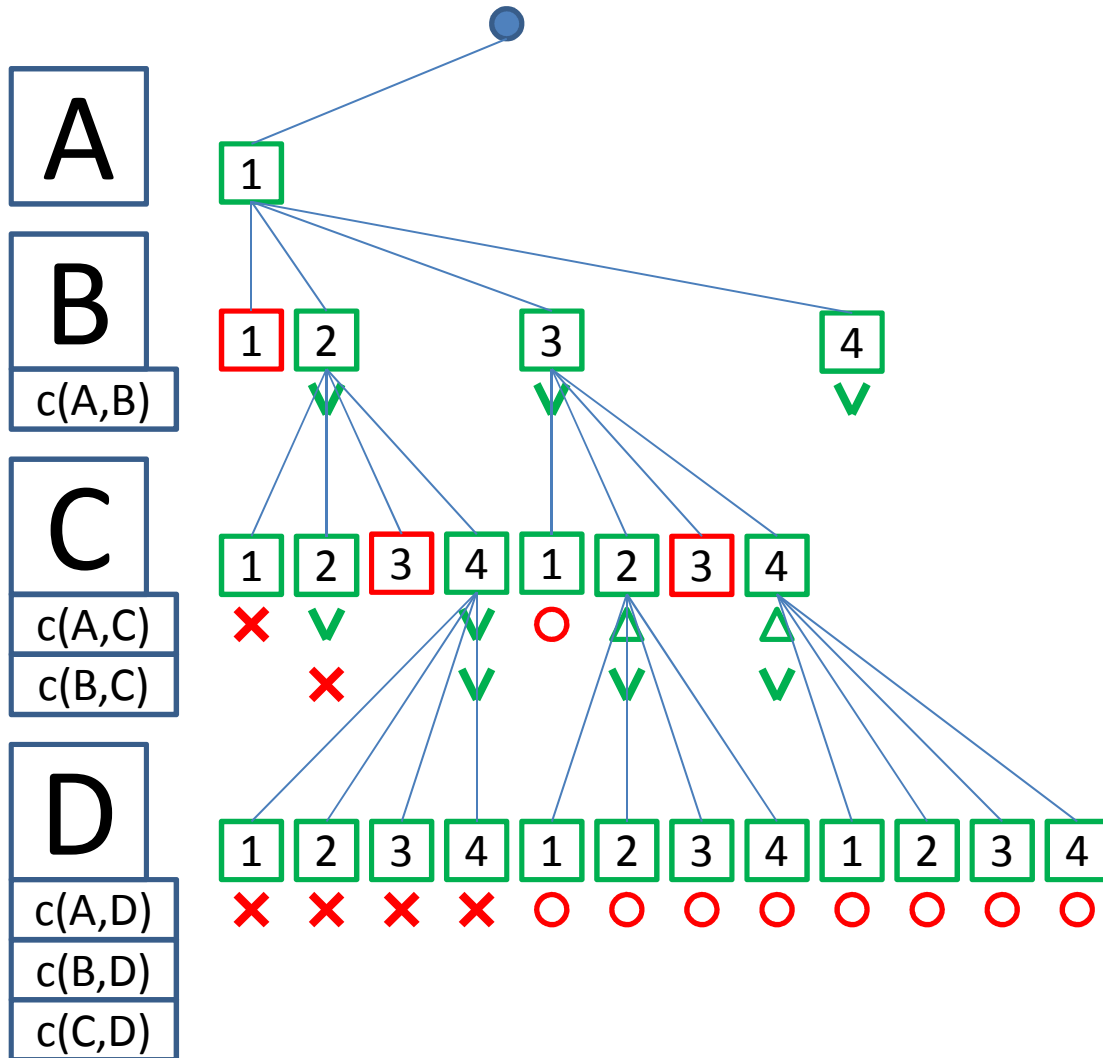
Constraint Processing: Backtracking



Constraint Processing: Backjumping



Constraint Processing: Backmarking



	1	2	3	4	Backup
A	0	1	1	1	1
B	0	1	1	1	1
C	1	2	0	2	2
D	1	1	1	1	2

Constraint Processing: Backmarking

A

B

$c(A,B)$

C

$c(A,C)$

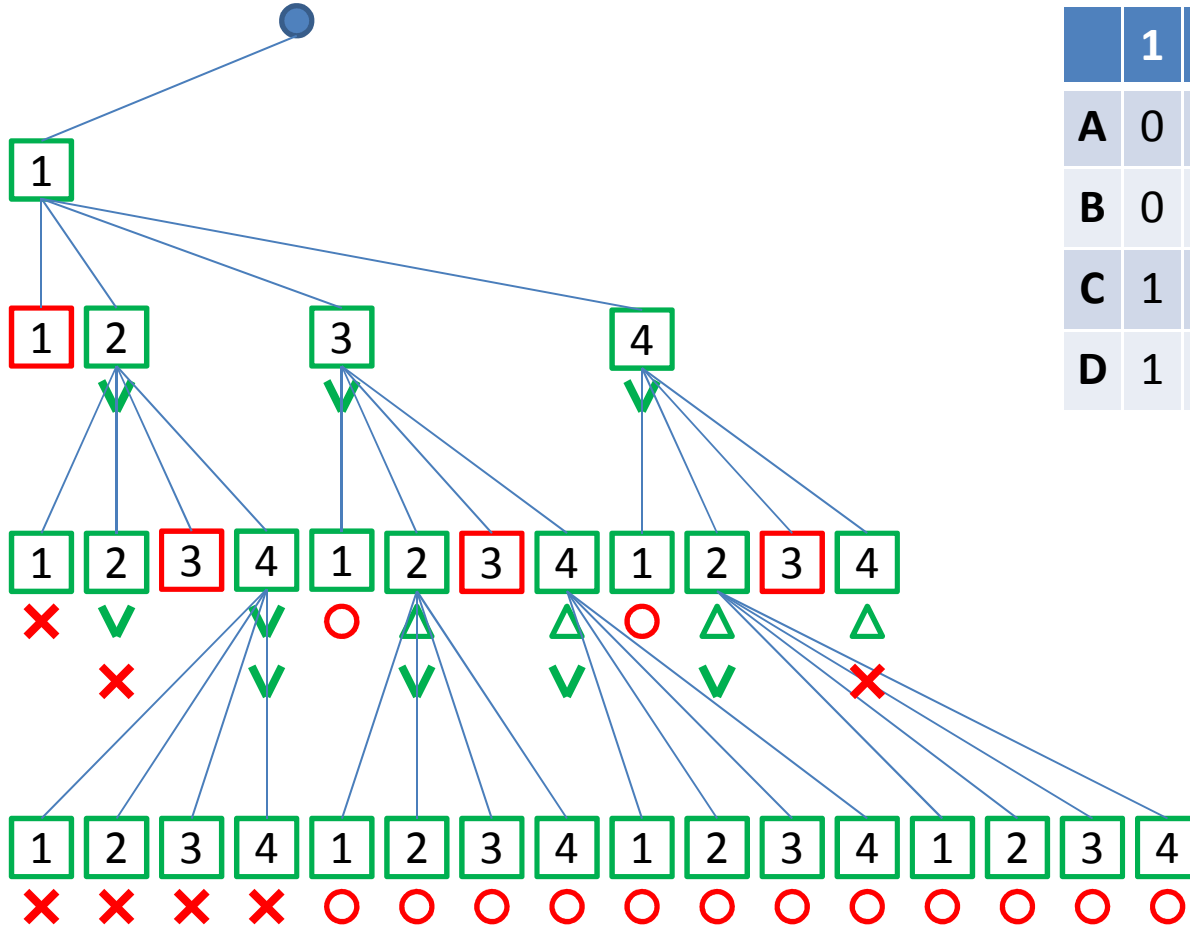
$c(B,C)$

D

$c(A,D)$

$c(B,D)$

$c(C,D)$



	1	2	3	4	Backup
A	0	1	1	1	1
B	0	1	1	1	1
C	1	2	0	2	2
D	1	1	1	1	3

Constraint Processing: Backmarking

A

B

$c(A,B)$

C

$c(A,C)$

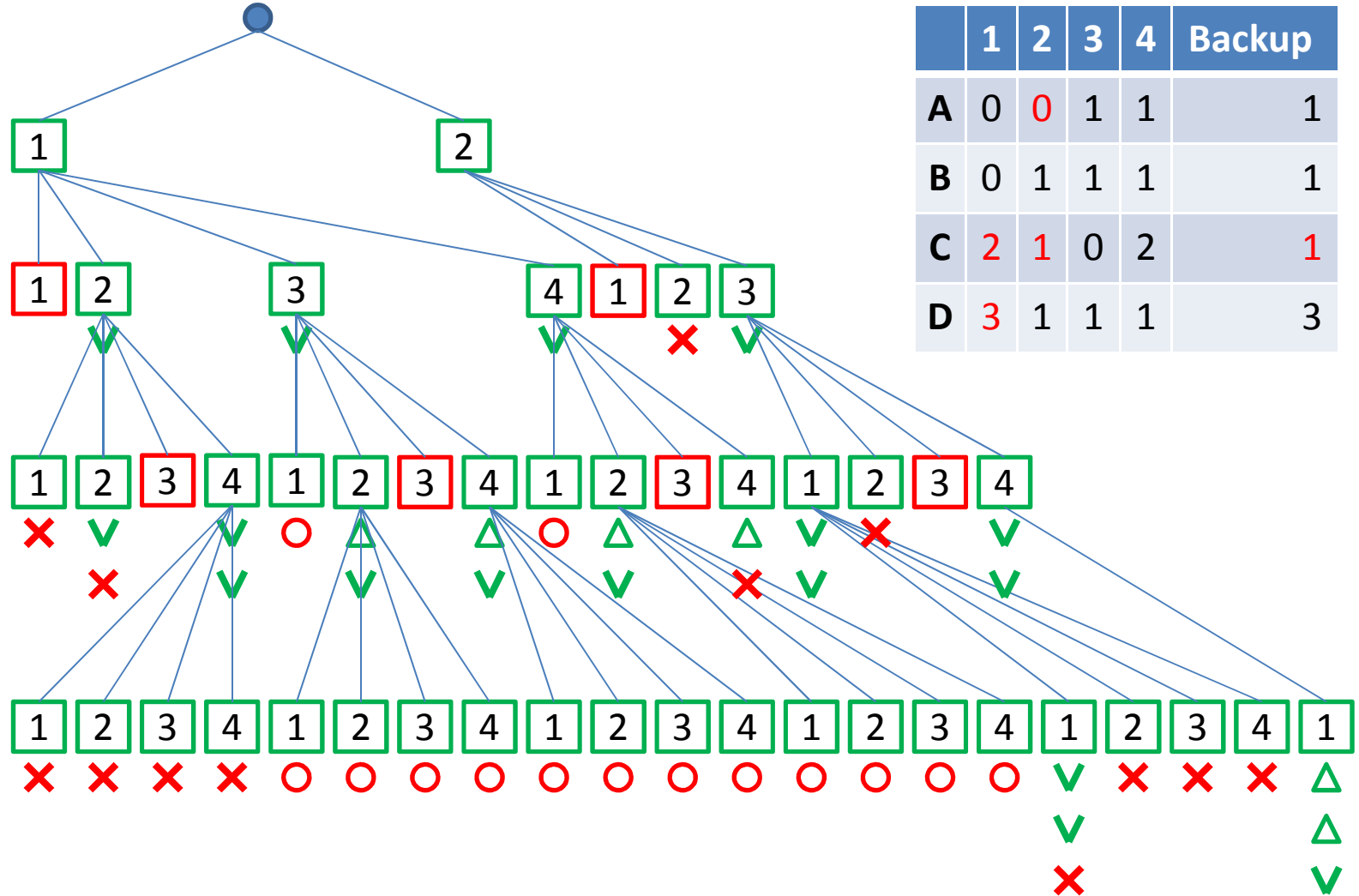
$c(B,C)$

D

$c(A,D)$

$c(B,D)$

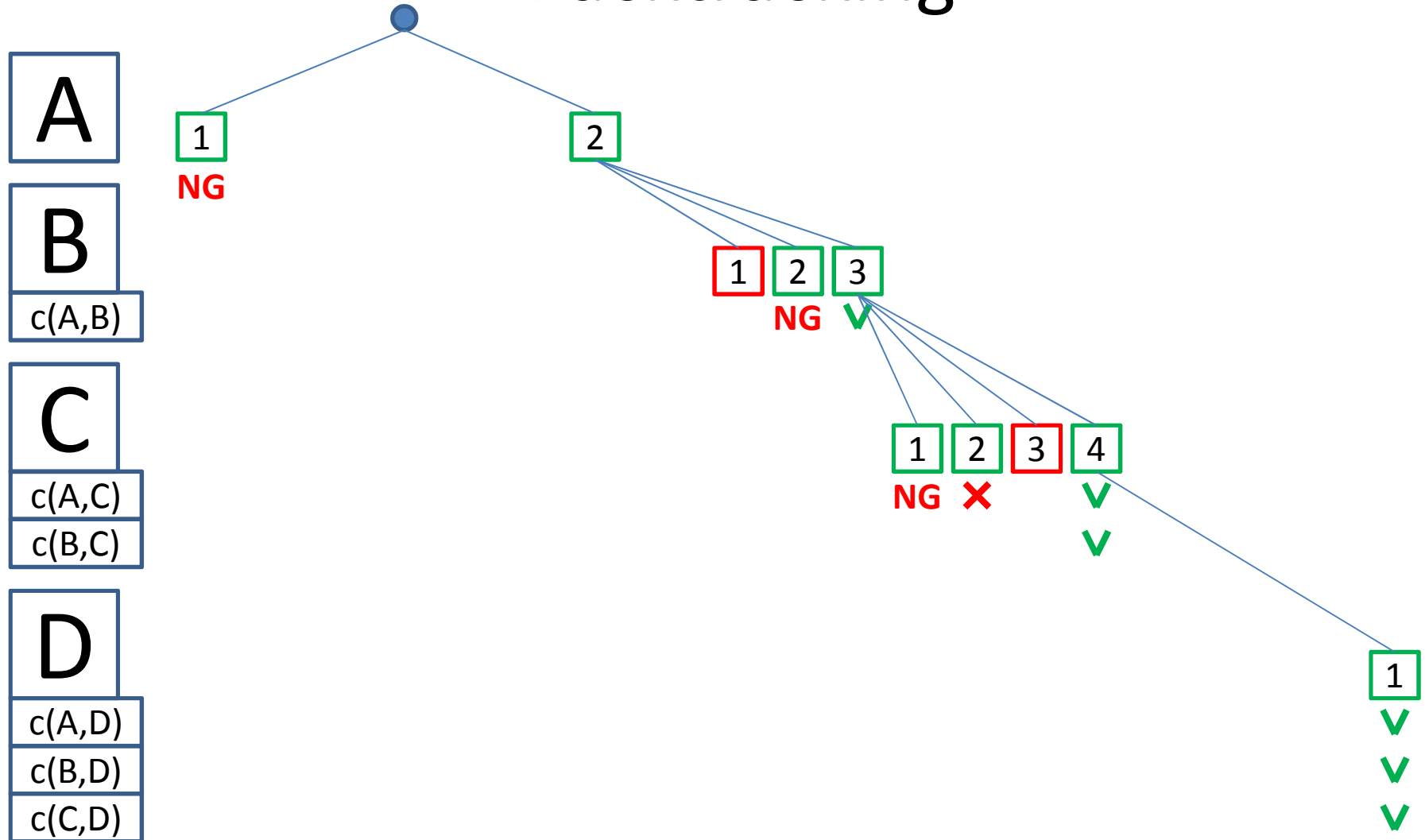
$c(C,D)$



Constraint Processing: No-goods

- $\{A=1\}$: No-good
 - No value for D such that $A = D + 1$
- $\{A=2, B=2\}$: No-good
 - A and B should have different houses
- $\{A=2, B=3\}$: Not a no-good: $\{A=2, B=3, C=4, D=1\}$
- $\{A=2, B=3, C=1\}$: No-good
 - $A = D + 1$, thus $D = 1$, but $C = 1$
- $\{A=2, B=4\}$: No-good
 - $A = D + 1$, thus $D = 1$, thus $C = 3$, but C cannot be 3

Constraint Processing: Intelligent Backtracking



Efficiency

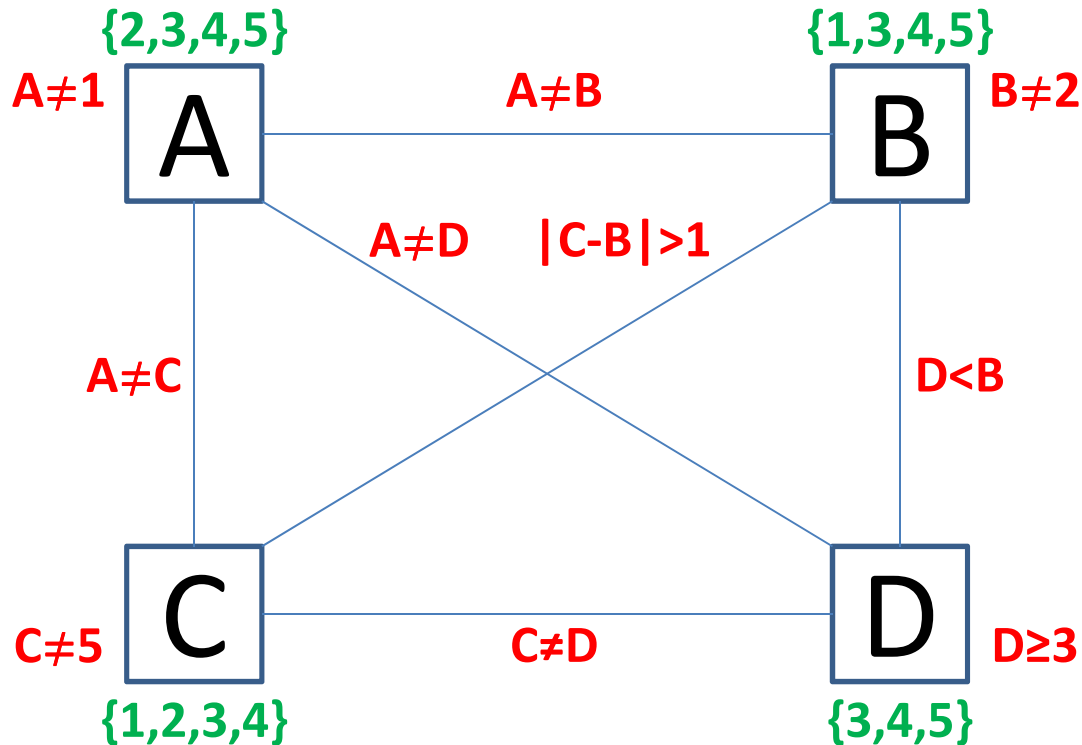
<i>All (One solution)</i>	Opened Nodes	Checks
Standard Backtracking	28 (13)	142 (56)
Backjumping	21 (8)	93 (30)
Backmarking	28 (13)	79 (34)
Intelligent Backtracking	6 (4)	16 (9) + NG

Exercises: Artificial Intelligence

Constraint Processing II & Waltz: The
4 Teachers problem

Problem Optimization

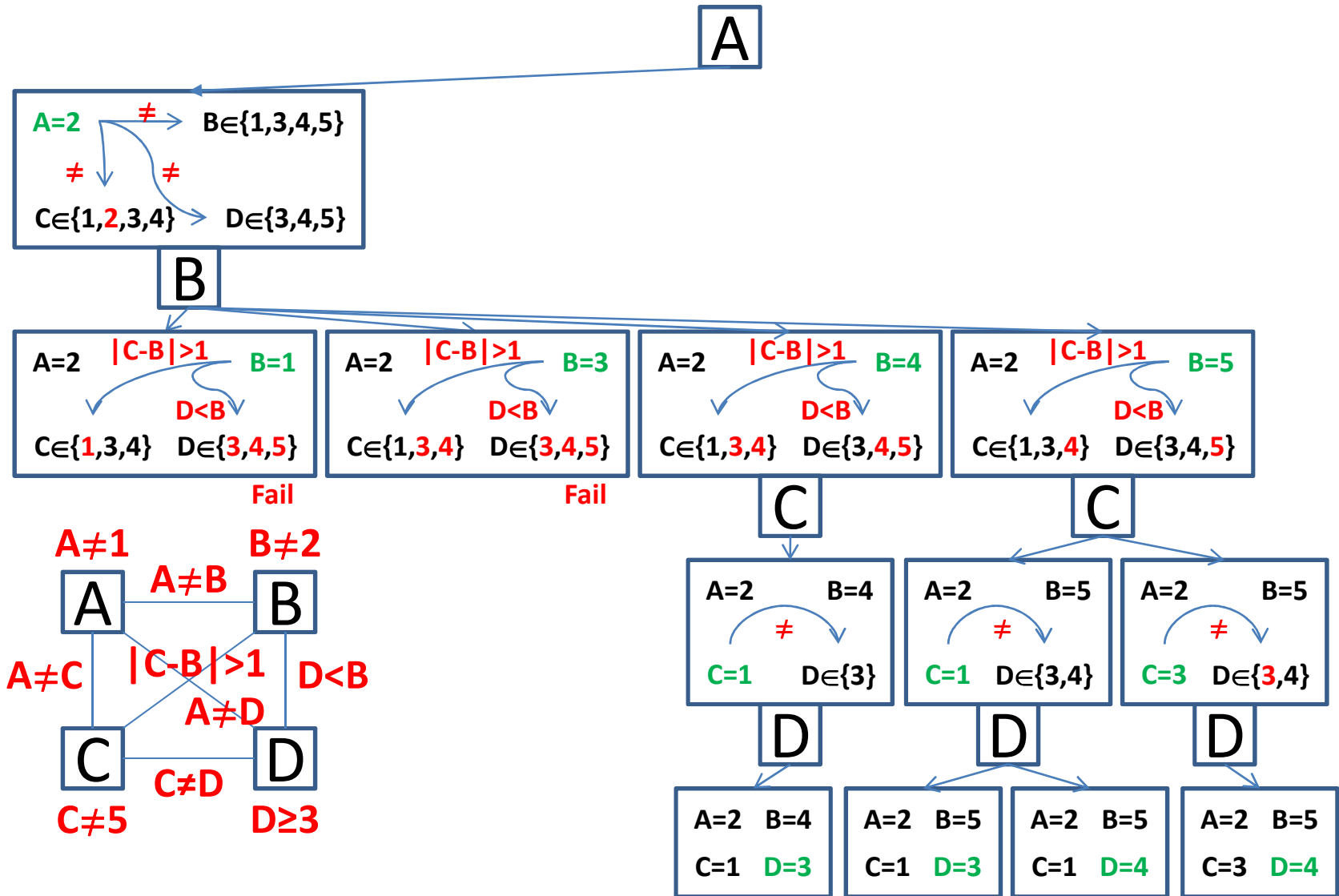
- Problem optimization:



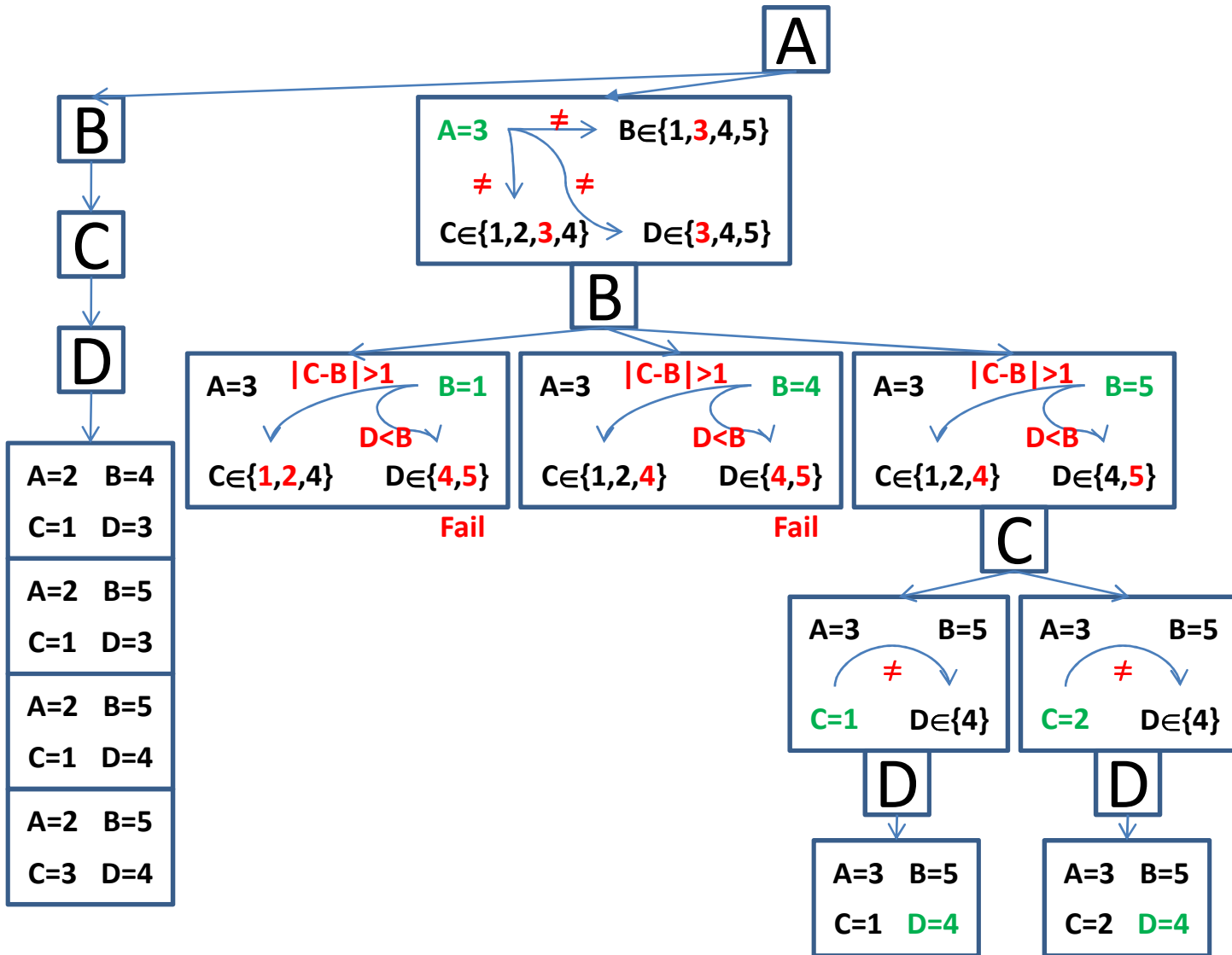
MiniMax & Constraint Processing: The 4 Houses problem

CONSTRAINT PROCESSING: FORWARD CHECKING

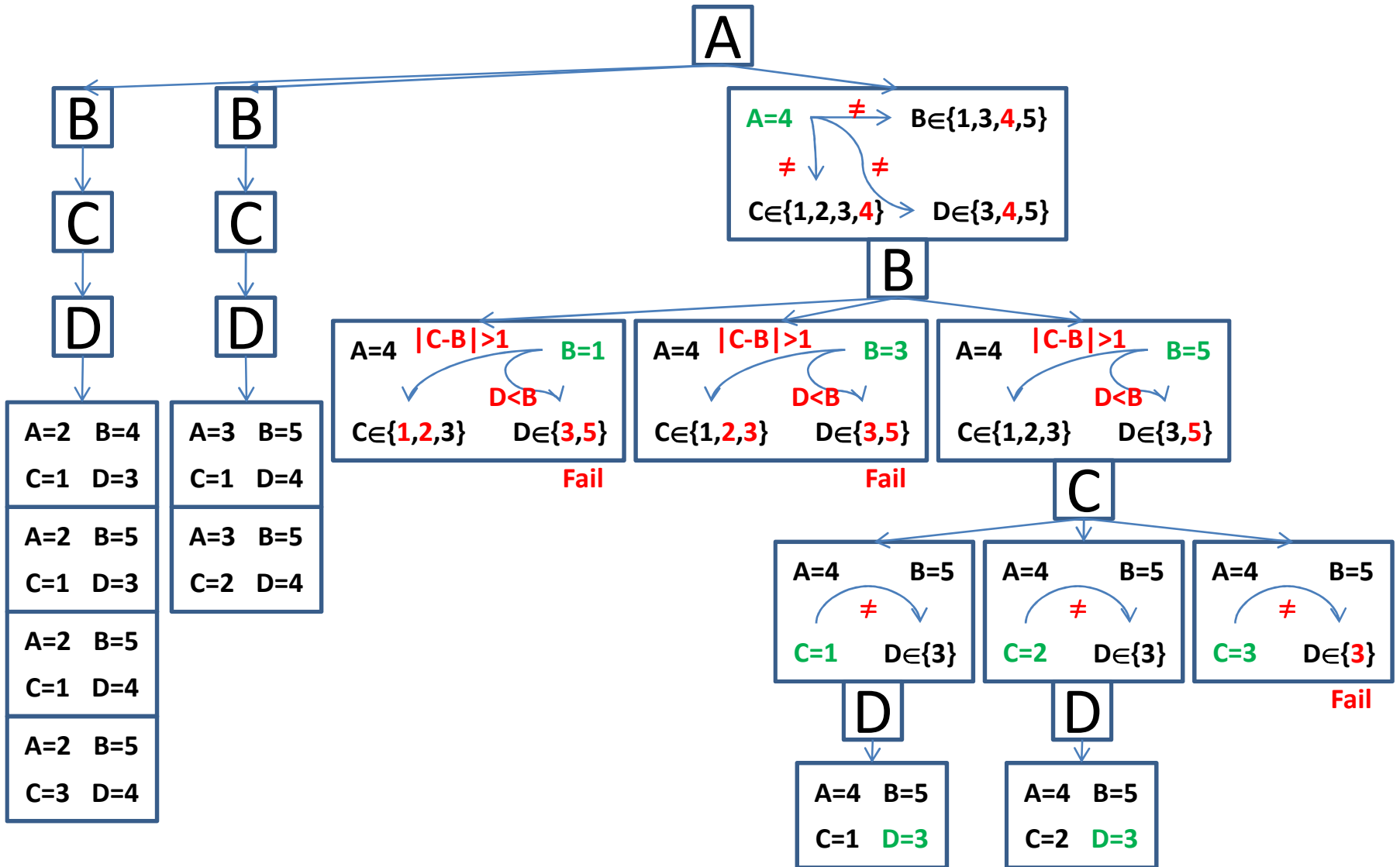
Forward Checking



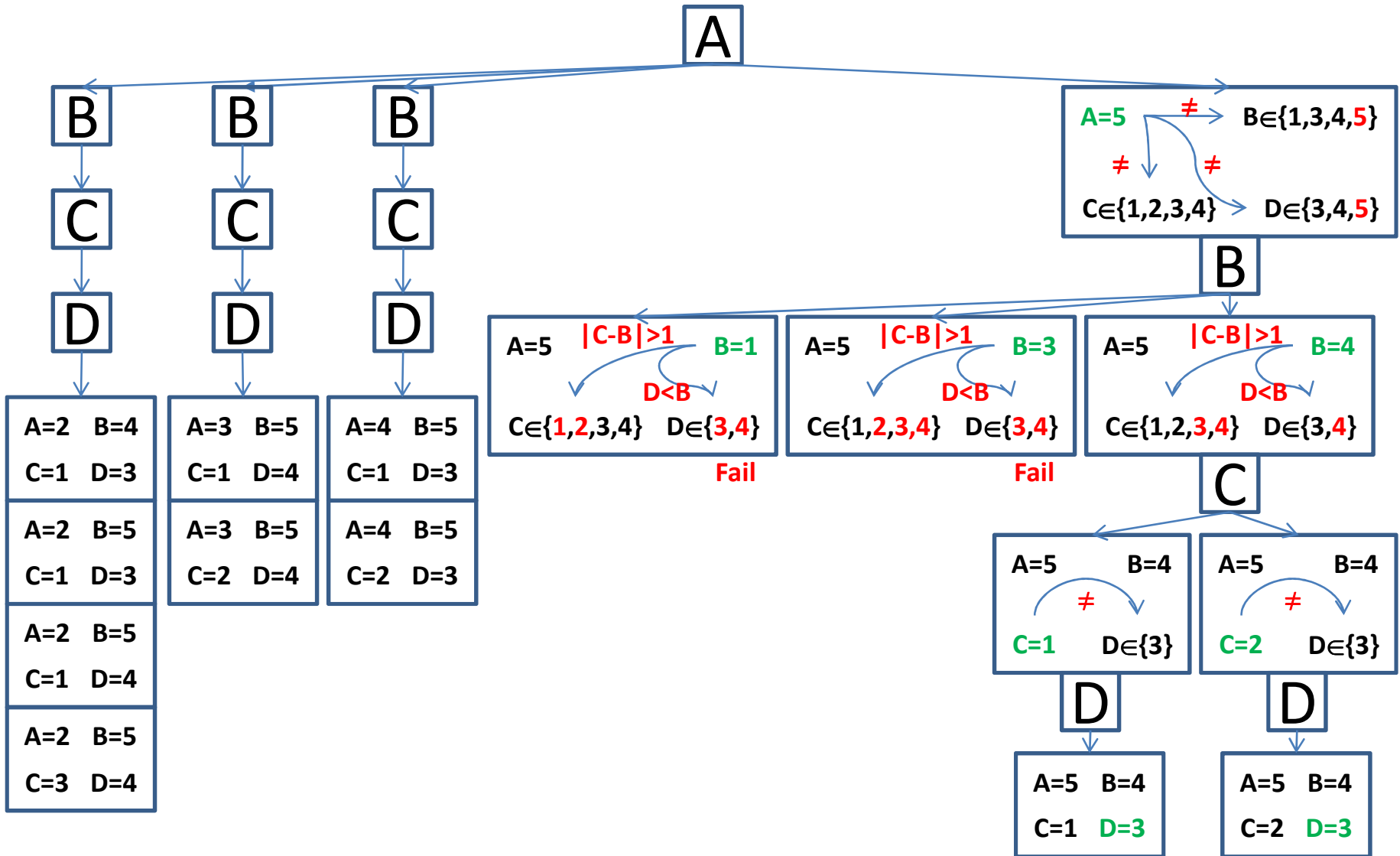
Forward Checking



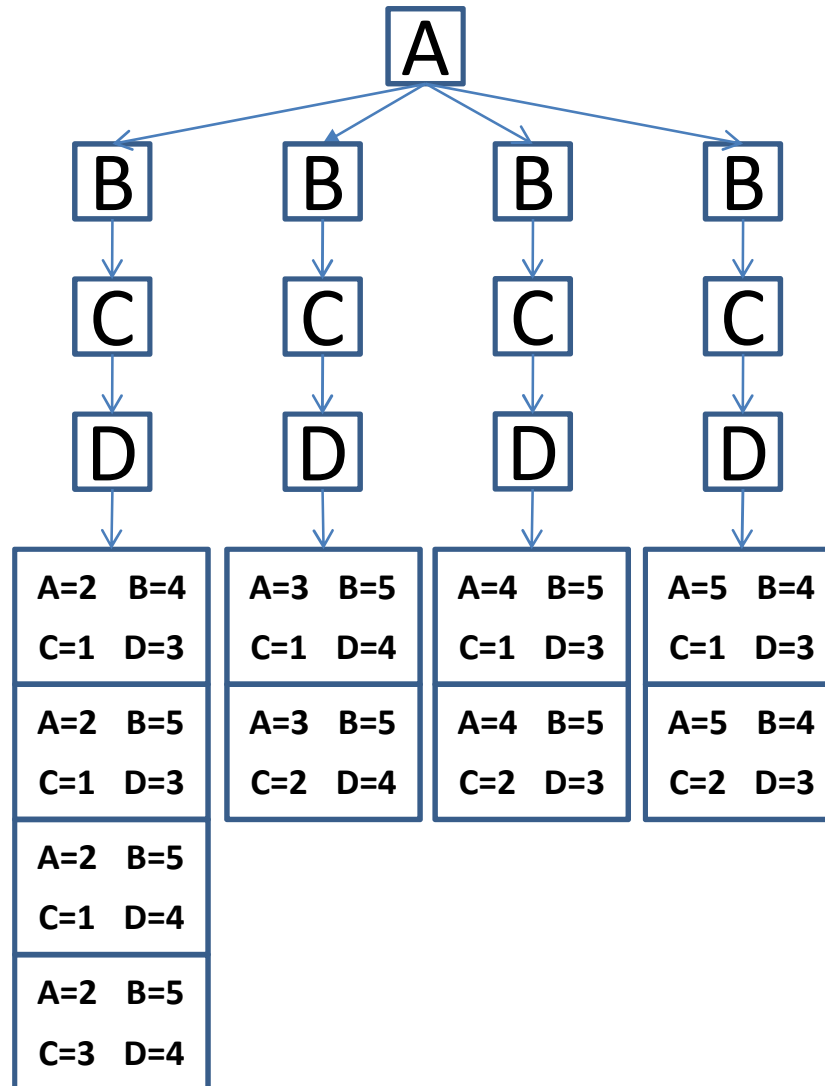
Forward Checking



Forward Checking



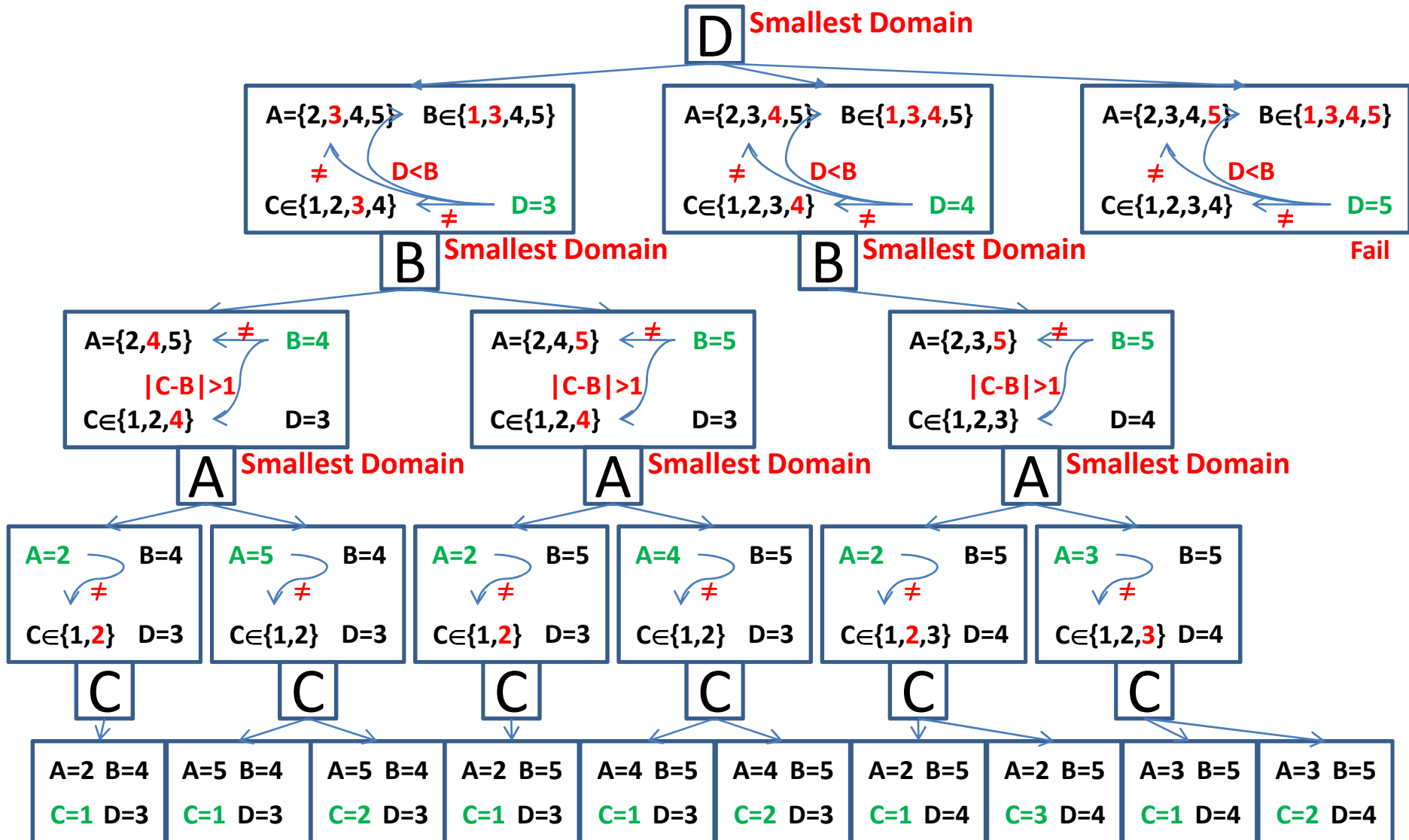
Forward Checking



MiniMax & Constraint Processing: The 4 Houses problem

CONSTRAINT PROCESSING: DYNAMIC SEARCH REARRANGEMENT FC

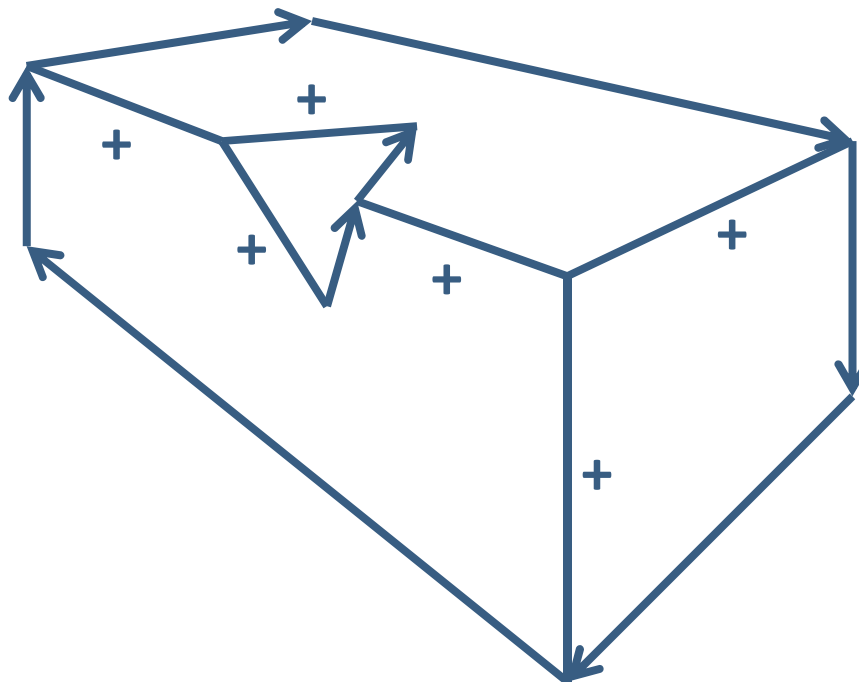
Dynamic Search Rearrangement FC



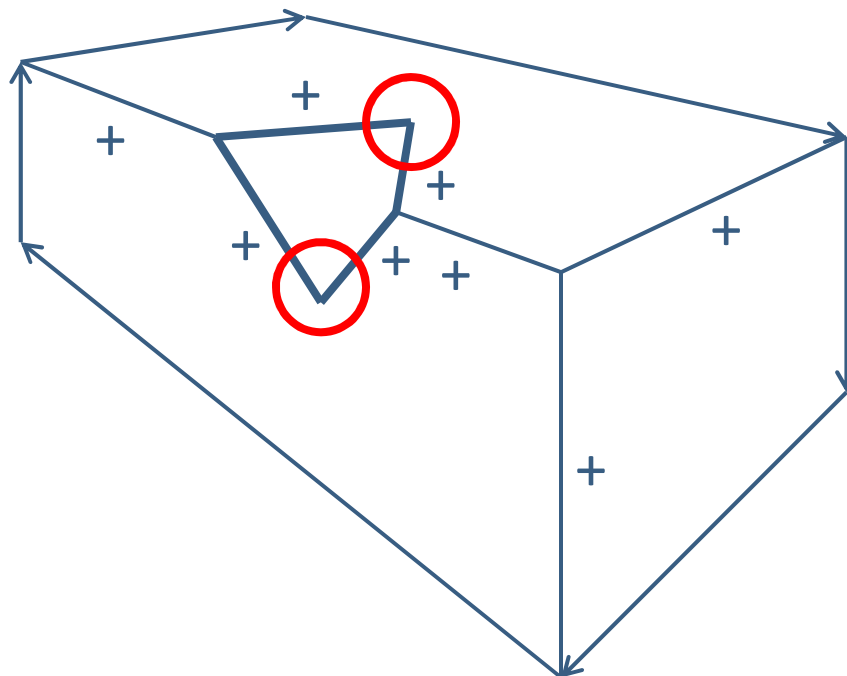
Exercises: Artificial Intelligence

Constraint Processing II & Waltz:
Waltz I

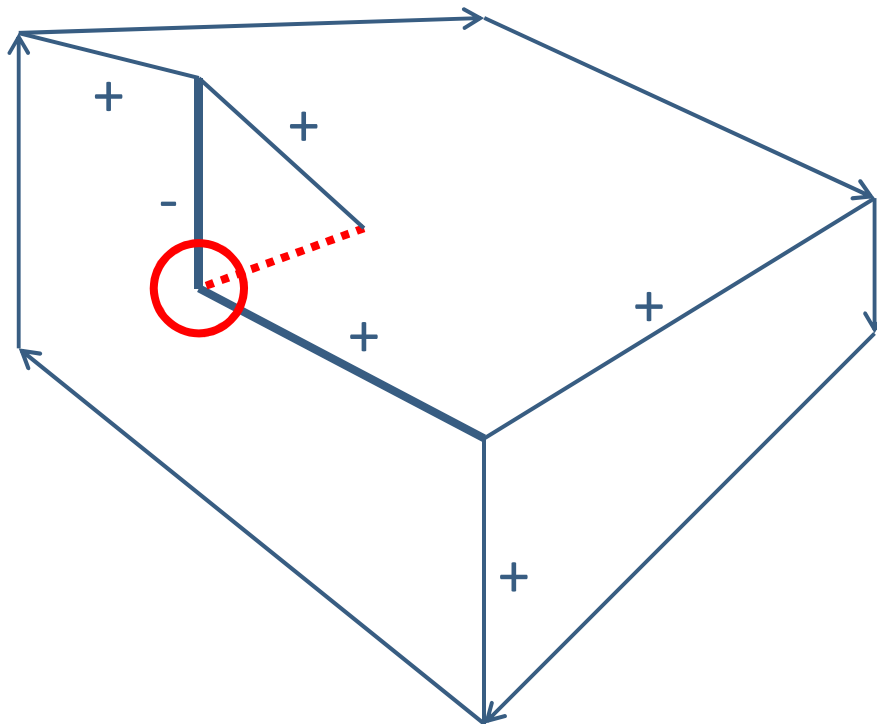
Solution



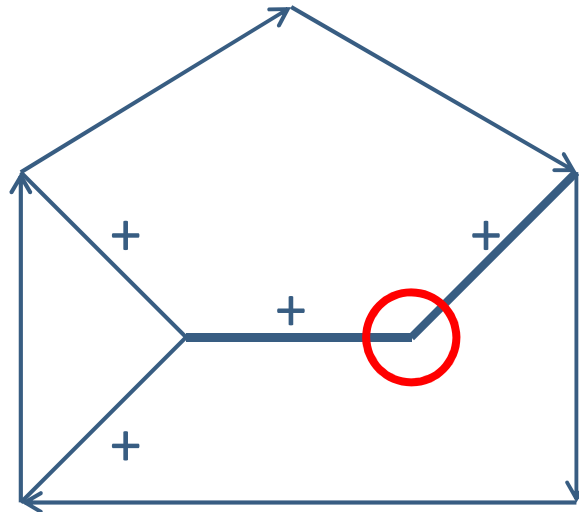
Solution



Solution

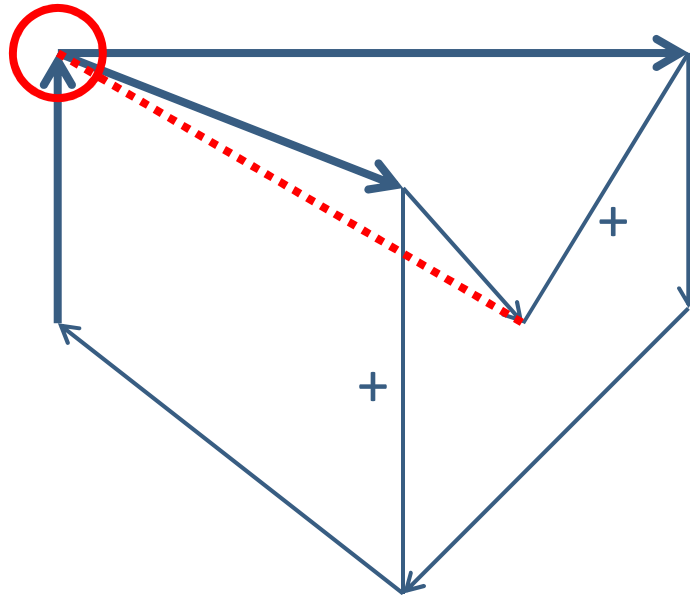


Solution



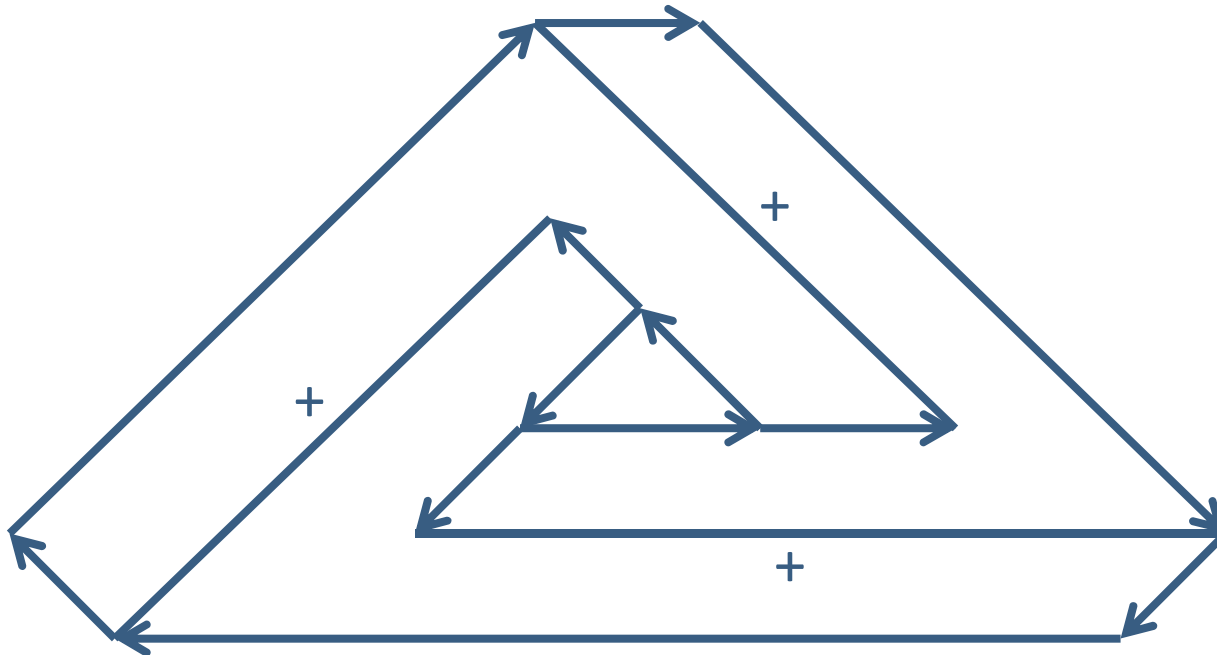
Solution

Line Drawing NOT allowed: 3-faced vertices!!



Solution

Drawing is locally correct, but is globally impossible.
Waltz procedure is local, thus, cannot detect this!

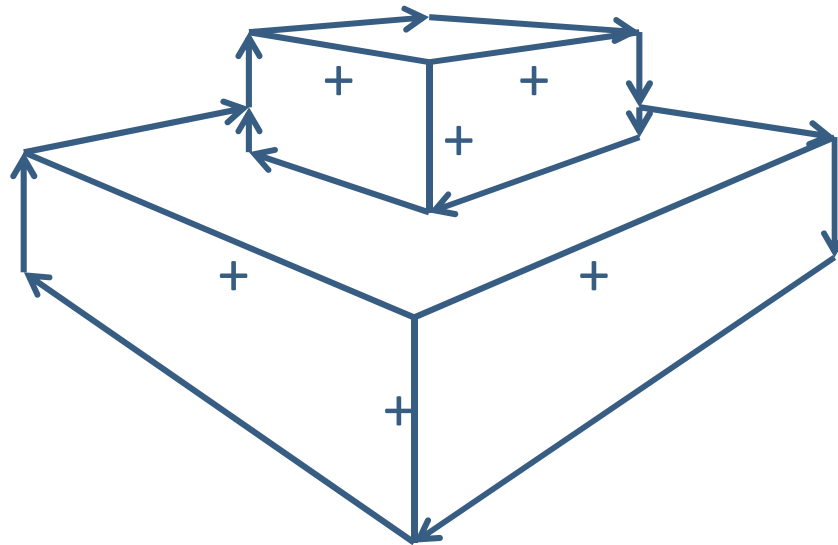


Exercises: Artificial Intelligence

Constraint Processing II & Waltz:
Waltz II

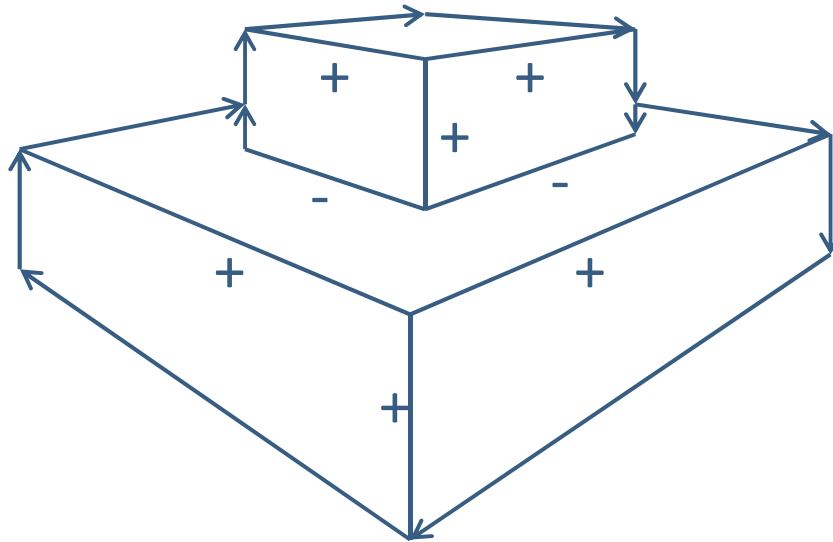
Solution

- Solution 1: Floating cube



Solution

- Solution 2: Sitting cube



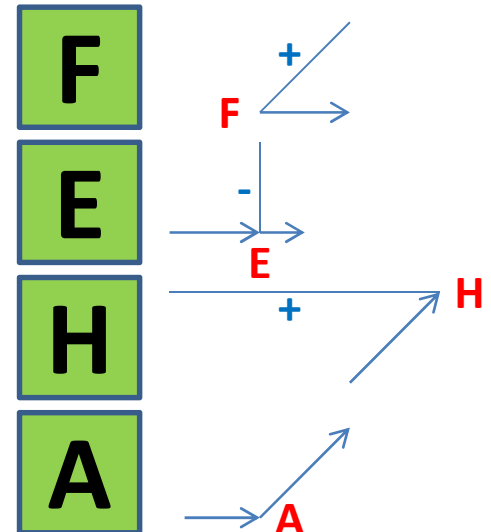
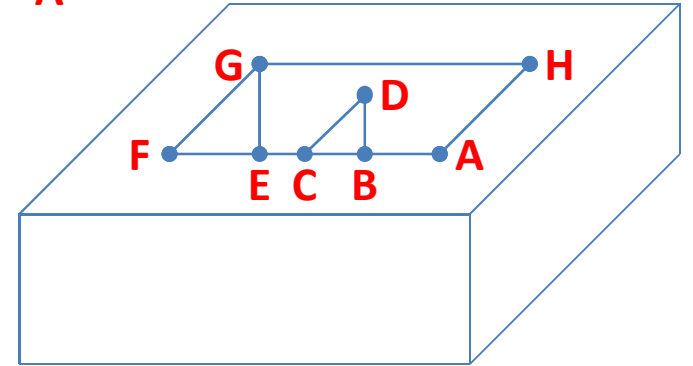
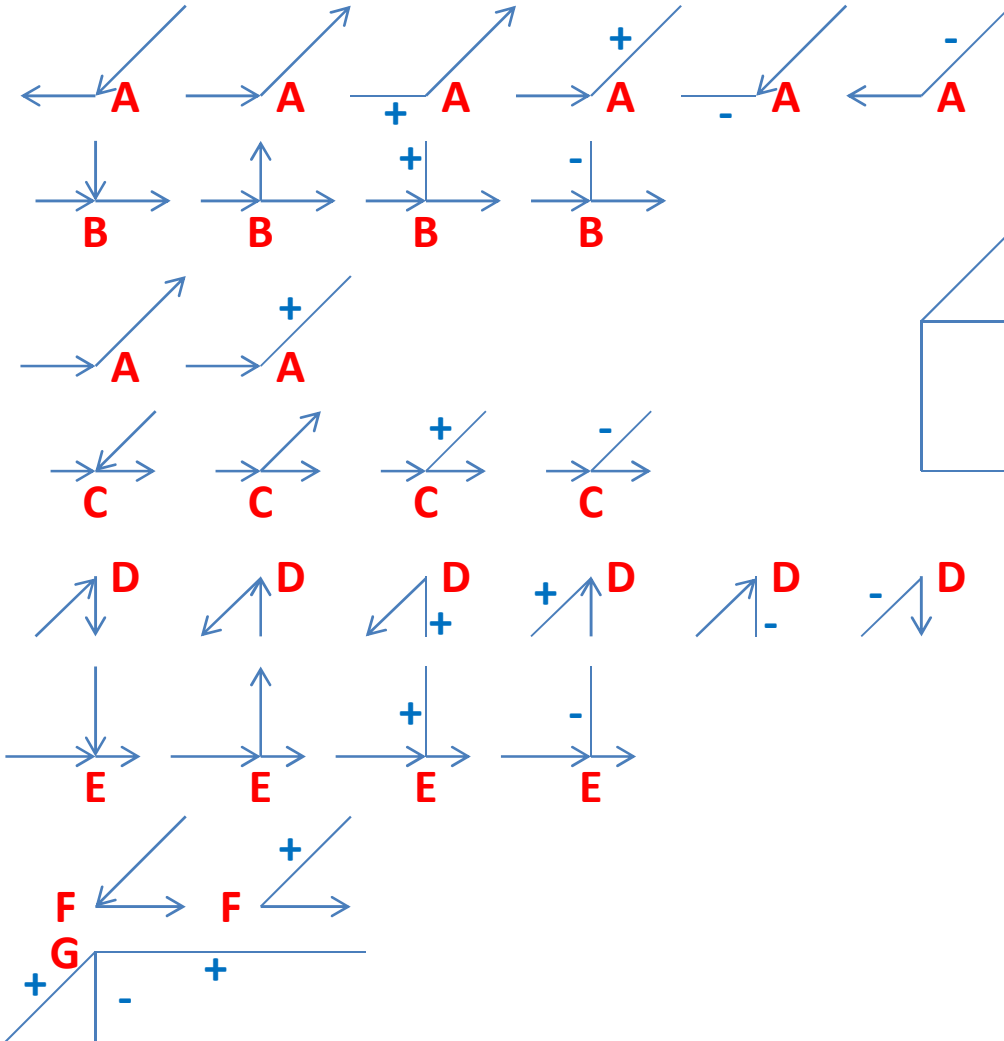
Exercises: Artificial Intelligence

Constraint Processing II & Waltz:
Waltz III

Exercises: Artificial Intelligence

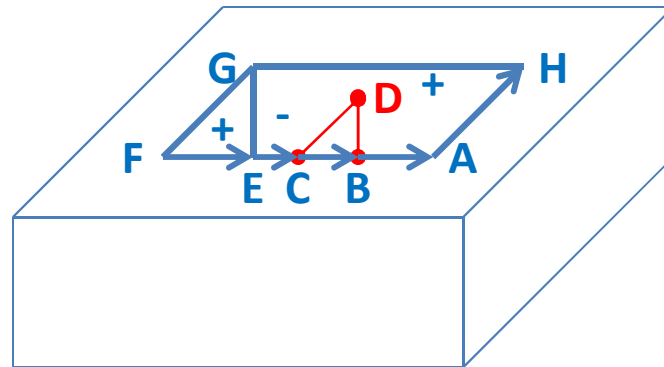
Constraint Processing II & Waltz:
Waltz IV

Solution



Solution

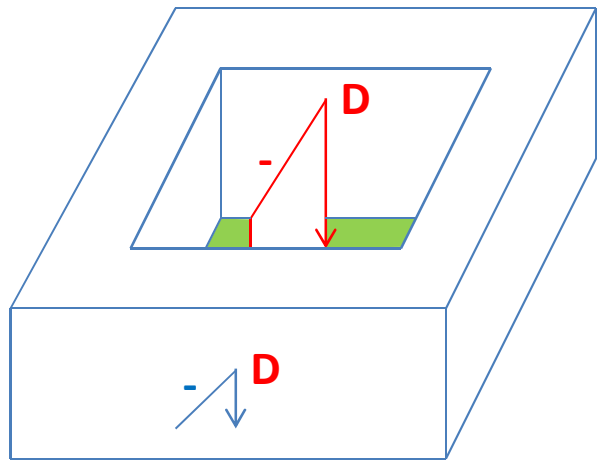
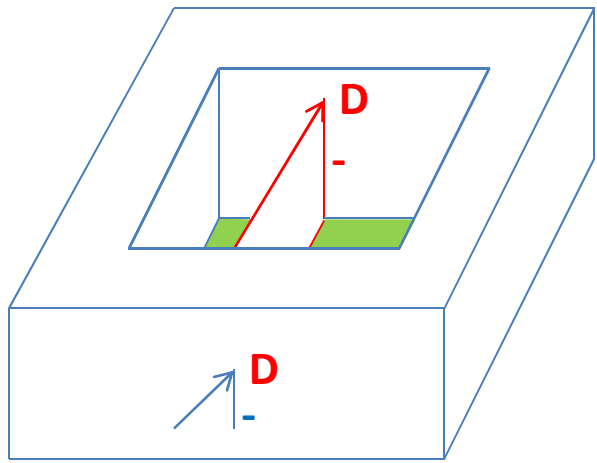
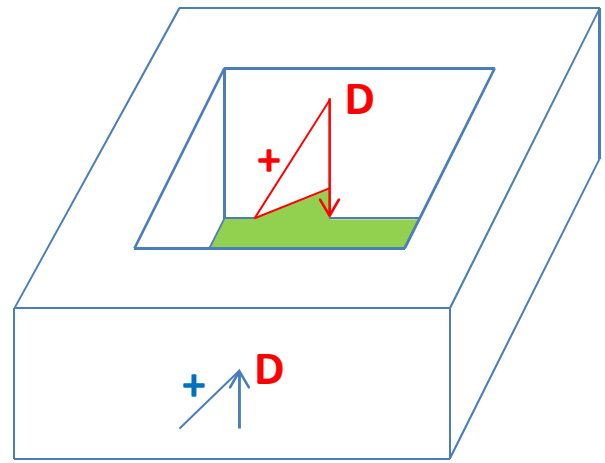
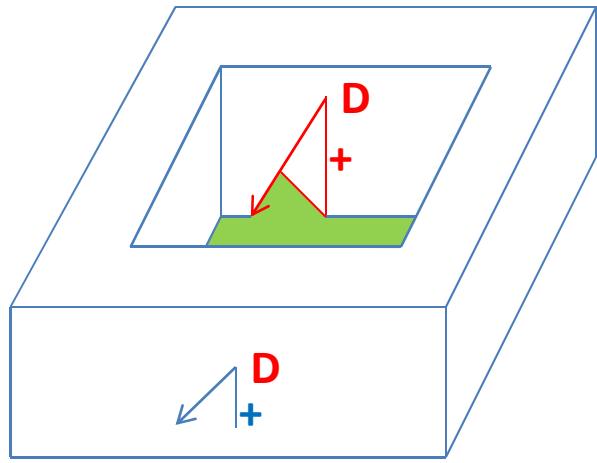
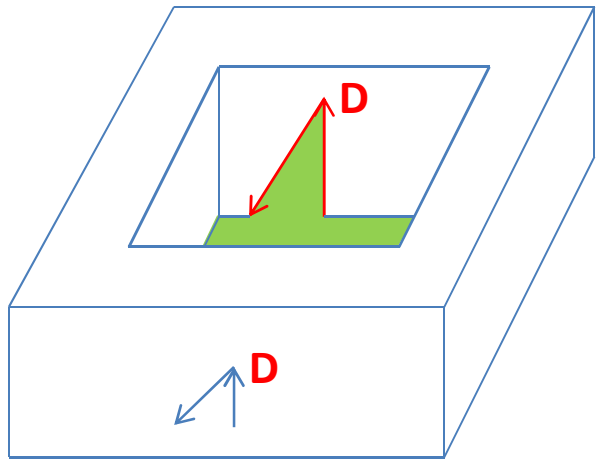
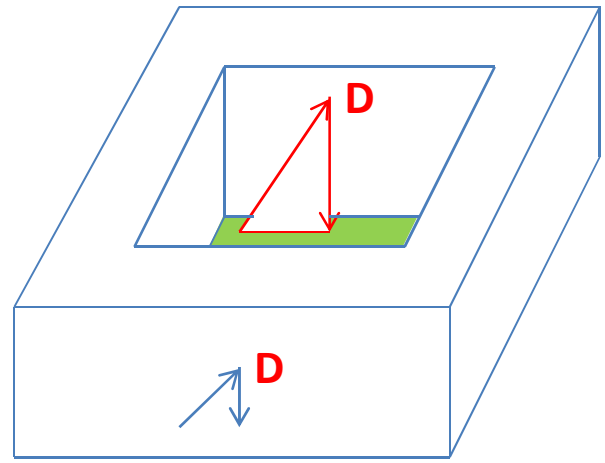
- We can determine **all nodes except for D**:



- **D** can still take **6 interpretations**:



Solution



Exercises: Artificial Intelligence

Constraint Processing II & Waltz:
Waltz V

Termination Waltz

- ***Waltz's procedure terminates if***
 - No possibilities for some vertex **OR**
 - No reduction of junction piles
- ***Waltz's procedure does not terminate if***
 - Only non-empty piles **AND**
 - Reduction of piles possible
- **BUT**
 - Piles are finite \implies Number of iterations finite
 \implies **Waltz's procedure terminates**

Exercises: Artificial Intelligence

Planning & Logic: Blocks world

STRIPS

I	
If	
Add	on(A,T)
	on(B,T)
	on(C,T)
	on(D,T)
	clear(A)
	clear(B)
	clear(C)
	clear(D)
Del	

O31 (x/B,z/A)	
If	
If	on(B,T)
	clear(A)
	clear(B)
Add	on(B,A)
Del	on(B,T) clear(A)

F	
If	
If	on(A,T)
	on(B,A)
	on(C,B)
	on(D,C)
	clear(D)
Add	
Del	

B(I,F)

B(I,O31)

B(I,O31)

B(I,O31)

B(O31,F)



Establishes



Threatens

Before constraint:

B(x,y)

STRIPS

I	
If	
Add	on(A,T)
	on(B,T)
	on(C,T)
	on(D,T)
	clear(A)
	clear(B)
	clear(C)
	clear(D)
Del	

O31	(x/B,z/A)
-----	-----------

O32	(x/C,z/B)
-----	-----------

If	on(C,T)
	clear(B)
	clear(C)

Add	on(C,B)
-----	---------

Del	on(C,T)
	clear(B)

F	
If	on(A,T)
	on(B,A)
	on(C,B)
	on(D,C)
	clear(D)
Add	
Del	



Establishes



Threatens

$B(O31, O32) \vee B(O32, I)$ ← Loop!

Before constraint:

$B(x,y)$

$B(I,F)$

$B(O31,F)$

$B(O32,F)$

$B(I,O31)$

$B(I,O32)$

STRIPS

I	
If	
Add	on(A,T)
	on(B,T)
	on(C,T)
	on(D,T)
	clear(A)
	clear(B)
	clear(C)
	clear(D)
Del	

F	
If	
If	on(A,T)
	on(B,A)
	on(C,B)
	on(D,C)
Add	
Del	

O31 (x/B,z/A)

O32 (x/C,z/B)

O33 (x/D,z/C)

If	on(D,T)
	clear(C)
	clear(D)

Add	on(D,C)
-----	---------

Del	on(D,T)
	clear(C)



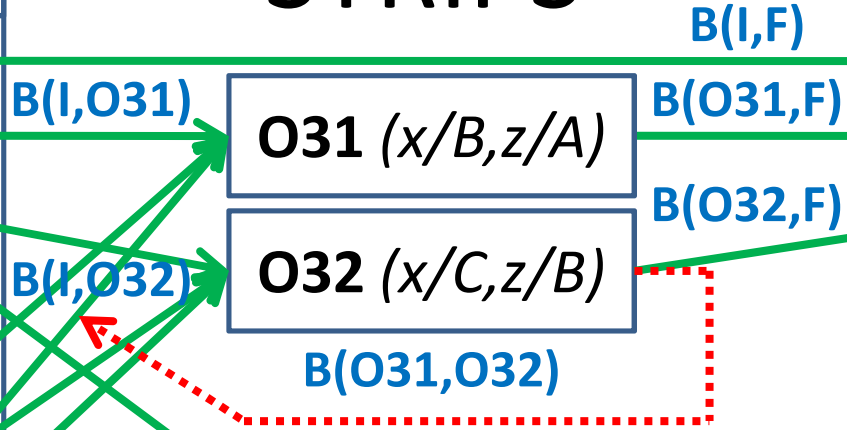
Establishes



Threatens

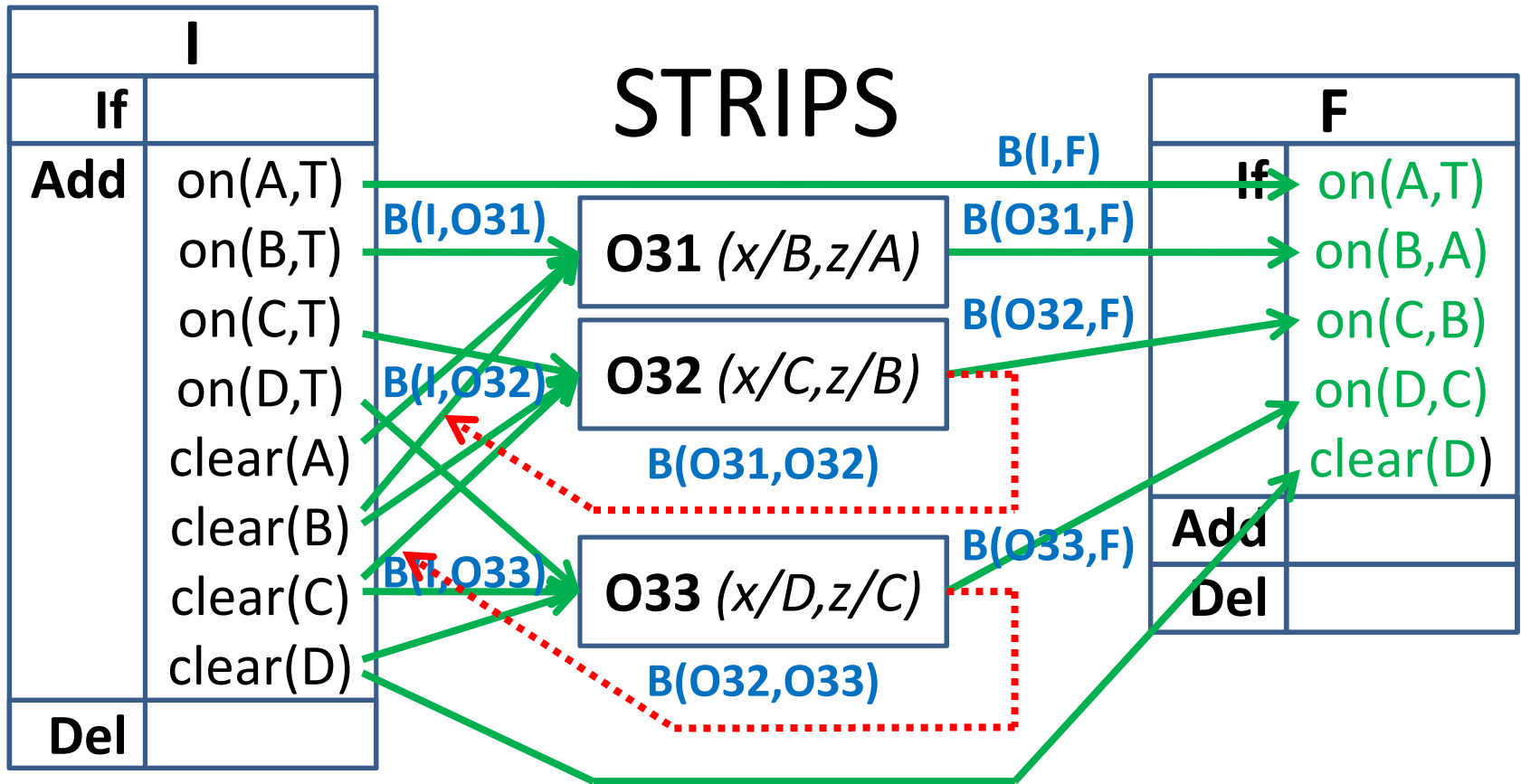
Before constraint:

$B(x,y)$



$B(O32,O33) \vee B(O33,I)$ ← Loop!

STRIPS



Establishes

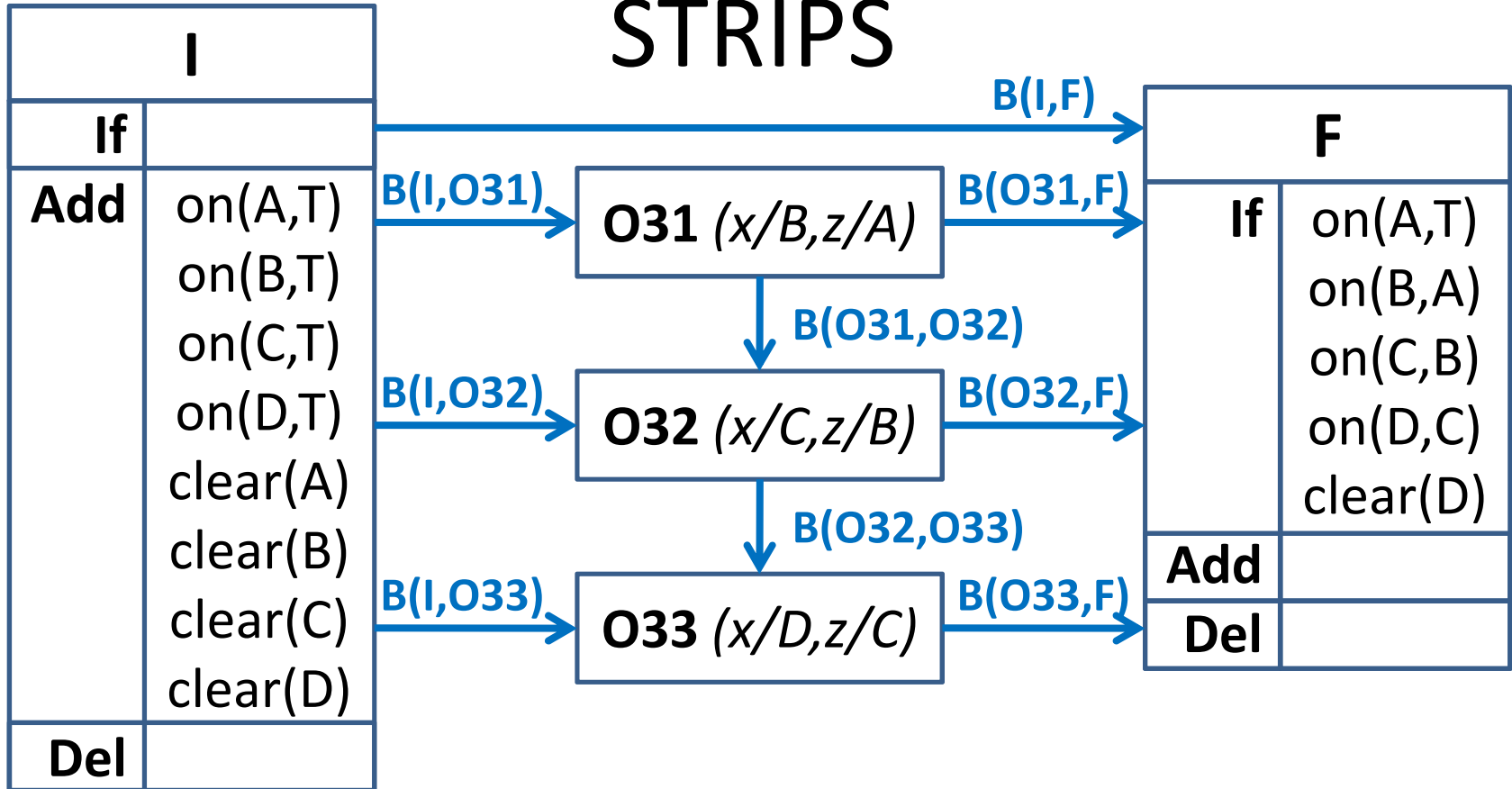


Threatens

Before constraint:

$B(x,y)$

STRIPS



Are the before constraints satisfiable?

YES:

→ **O31** → **O32** → **O33** →

Exercises: Artificial Intelligence

Planning & Logic: Buying milk

STRIPS

	If	
I	Add	at(home)
	Del	

Operator 2		
If	→	at(home)
Add		at(shop)
Del	...	at(home)

Operator 3		
If		at(shop) ←
Add		have(milk)
Del		

F		
If	→	have(milk) at(home)
Add		
Del		

$B(I,O2)$

$B(O2,I) \vee B(F,O2)$

2 x Loop!

$B(I,F)$

$B(O2,O3)$

$B(O3,F)$



Establishes



Threatens

Before constraint:

$B(x,y)$

STRIPS

	If	
I	Add	at(home)
	Del	

Operator 2		
If	at(home)	
Add	at(shop)	
Del	at(home)	

Operator 3		
If	at(shop)	
Add	have(milk)	
Del		

Operator 4		
If	paid at(shop)	
Add	at(home)	
Del	at(shop)	

F		
If	have(milk) at(home)	
Add		
Del		

$B(I,O2)$

$B(O2,O3)$

$B(O2,O4) \vee B(F,O2)$

$B(O3,F)$

$B(O4,F)$

Establishes

Threatens

Before constraint:

$B(x,y)$



STRIPS

I	If	
	Add	at(home)
	Del	

Operator 2		
If	→	at(home)
Add		at(shop)
Del		at(home)

Operator 3		
If		at(shop) ←
Add		have(milk)
Del		

Operator 4		
If		paid at(shop)
Add		at(home)
Del		at(shop)

F		
If	→	have(milk) at(home)
Add		
Del		

$B(I, O2)$

$B(O4, O2) \vee B(O3, O4)$

$B(O2, O3)$

$B(O2, O4)$

$B(O3, F)$

$B(O4, F)$



Establishes



Threatens

Before constraint:

$B(x, y)$

STRIPS

	If	
I	Add	at(home)
	Del	

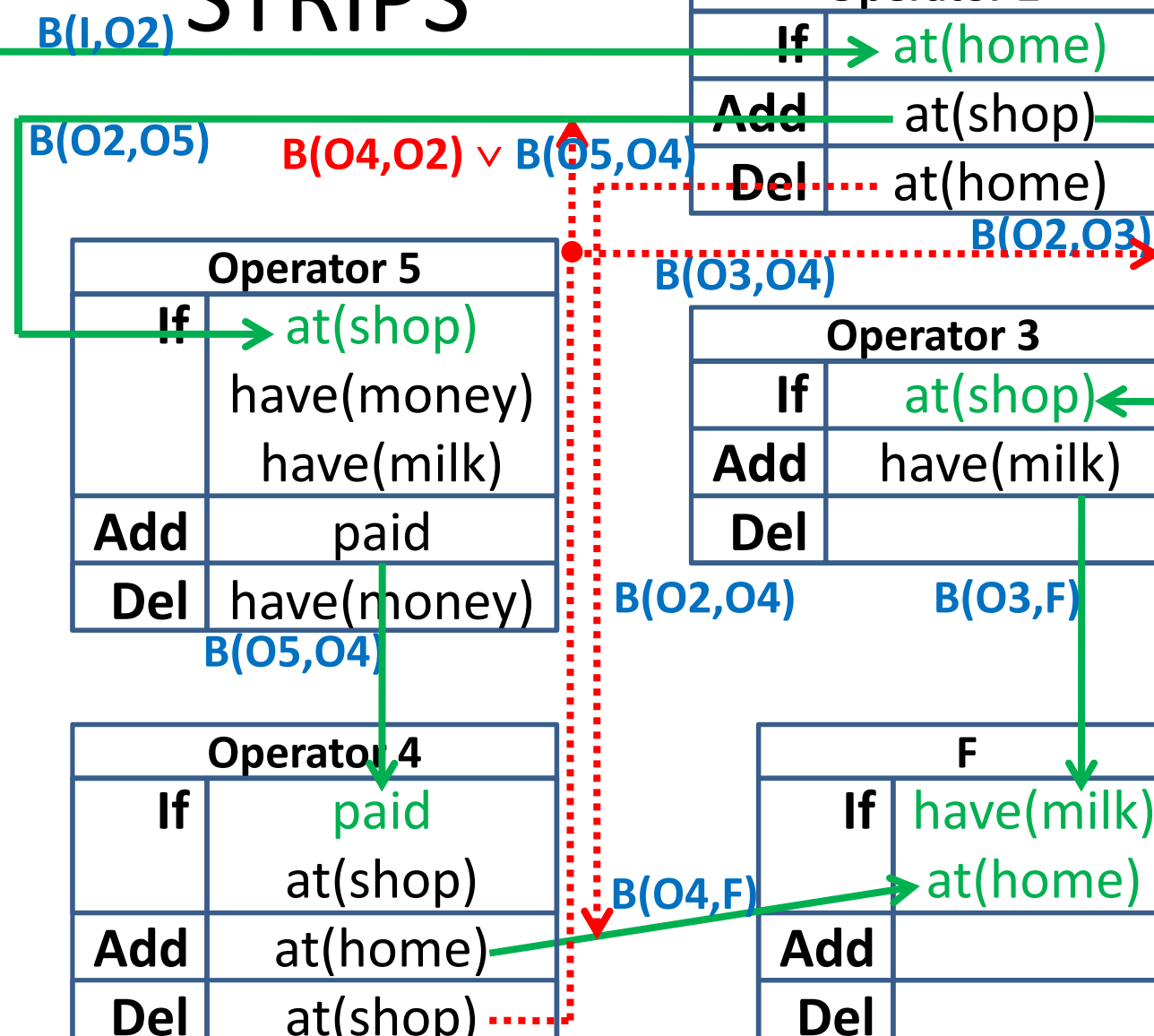
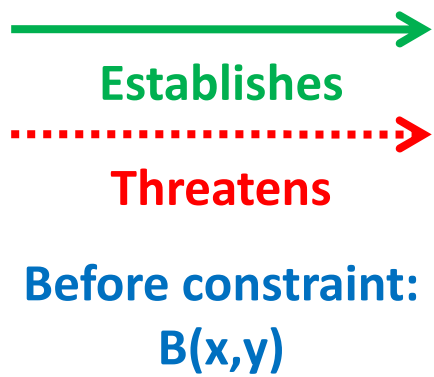
Operator 2	
If	→ at(home)
Add	→ at(shop)
Del	→ at(home)

Operator 5	
If	→ at(shop)
	have(money)
	have(milk)
Add	paid
Del	have(money)

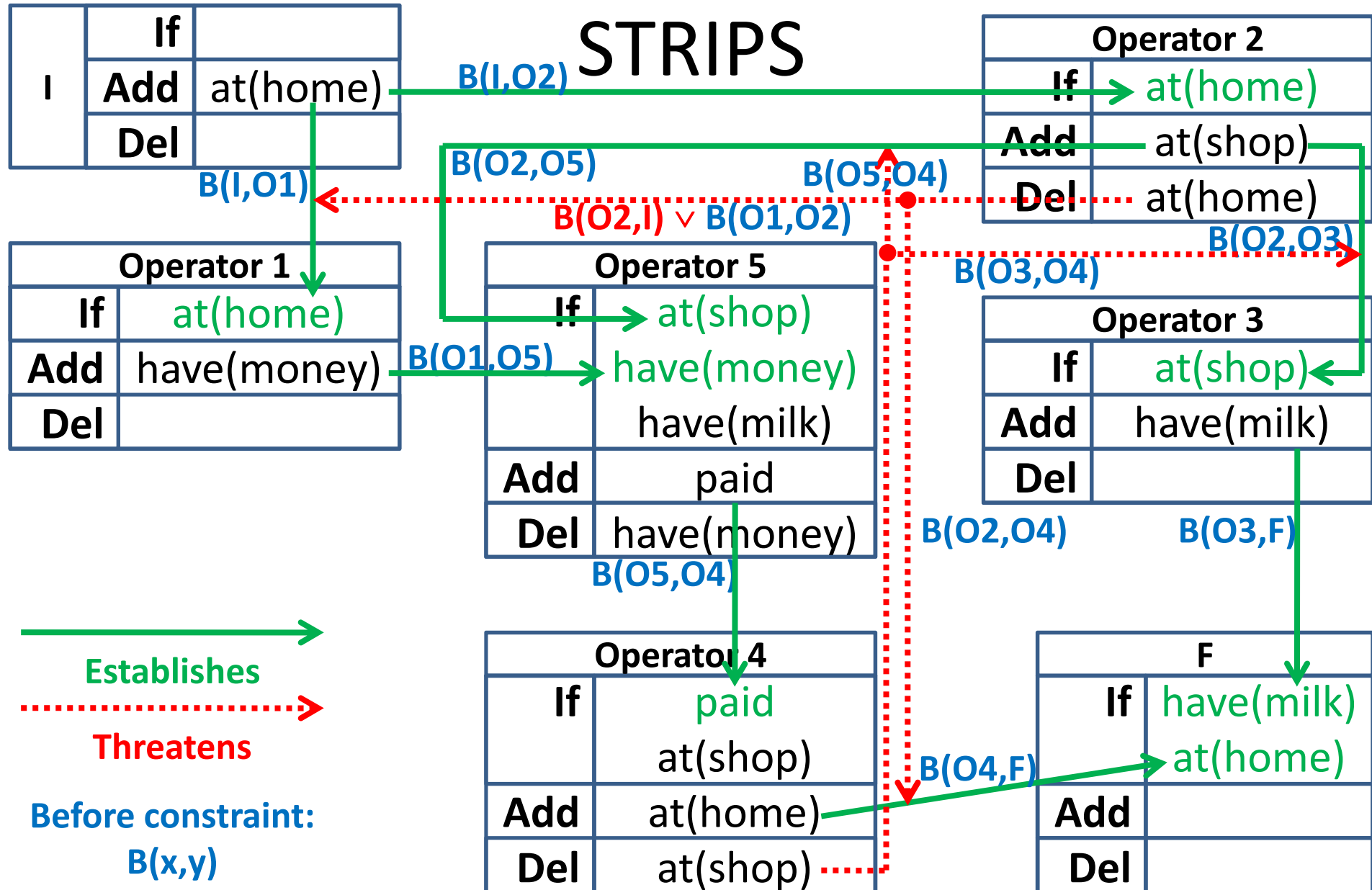
Operator 3	
If	← at(shop)
Add	have(milk)
Del	

Operator 4	
If	← paid
	at(shop)
Add	at(home)
Del	at(shop)

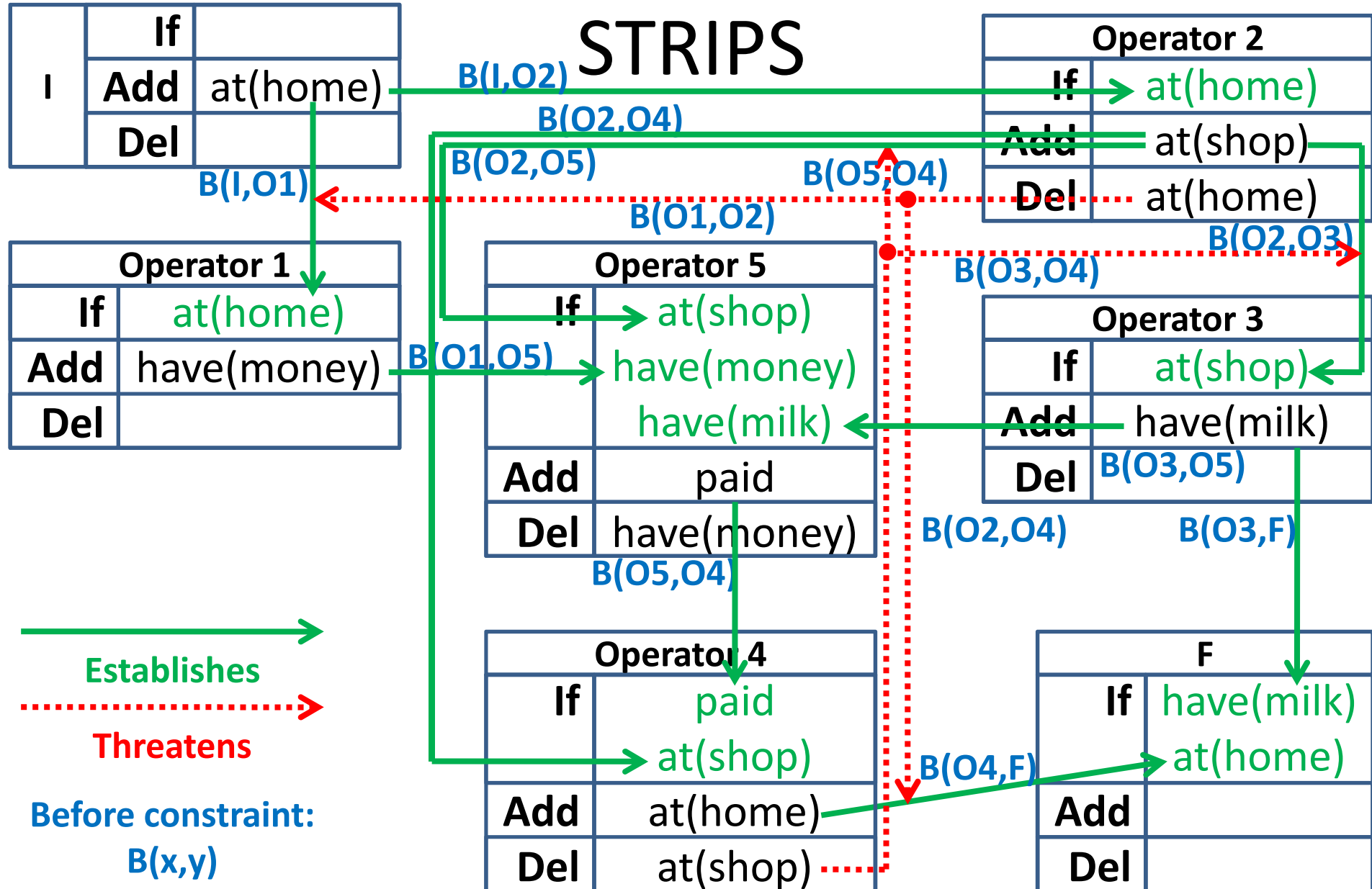
F	
If	← have(milk)
	at(home)
Add	
Del	



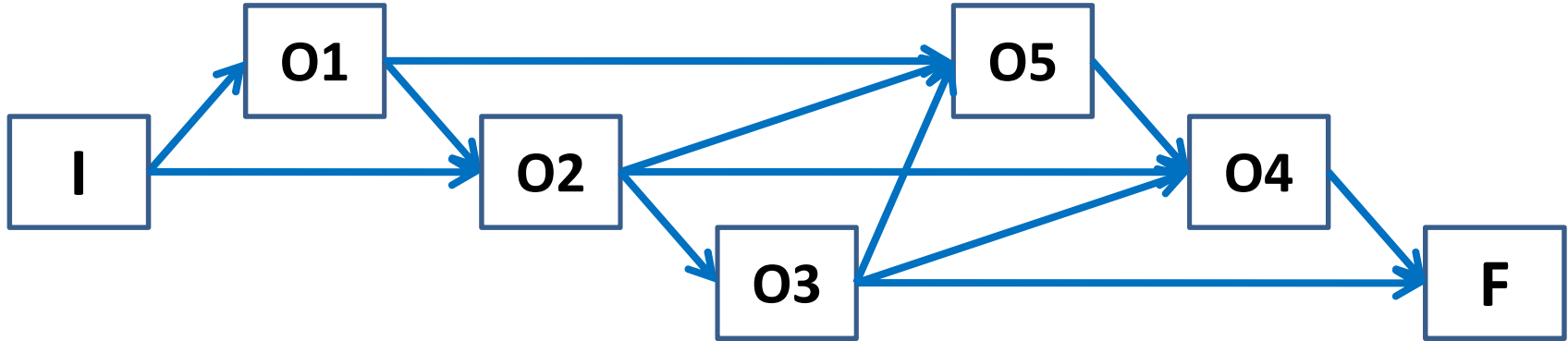
STRIPS



STRIPS



STRIPS



Are the before constraints satisfiable?

YES:

→ O1 → O2 → O3 → O5 → O4 →

Exercises: Artificial Intelligence

Planning & Logic: English to Logic

Problem & Solution

- *Not all students take both history and biology*

$$\neg \forall x [\text{student}(x) \Rightarrow \text{takes}(x, \text{hist}) \wedge \text{takes}(x, \text{bio})]$$

$$\Leftrightarrow [A \Rightarrow B \Leftrightarrow \neg A \vee B]$$

$$\neg \forall x [\neg [\text{student}(x)] \vee [\text{takes}(x, \text{hist}) \wedge \text{takes}(x, \text{bio})]]$$

$$\Leftrightarrow [\neg \forall x (F) \Leftrightarrow \exists x (\neg F)]$$

$$\exists x \neg [\neg [\text{student}(x)] \vee [\text{takes}(x, \text{hist}) \wedge \text{takes}(x, \text{bio})]]$$

$$\Leftrightarrow [\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B], [\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B]$$

$$\exists x [\text{student}(x) \wedge [\neg \text{takes}(x, \text{hist}) \vee \neg \text{takes}(x, \text{bio})]]$$

Problem & Solution

- *No person likes a smart vegetarian*

$$\forall x \forall y [\text{person}(x) \wedge \text{vegetarian}(y) \wedge \text{smart}(y) \Rightarrow \neg \text{likes}(x,y)]$$

$$\Leftrightarrow [A \Rightarrow B \Leftrightarrow \neg A \vee B]$$

$$\forall x \forall y [\neg[\text{person}(x) \wedge \text{vegetarian}(y) \wedge \text{smart}(y)] \vee \neg \text{likes}(x,y)]$$

$$\Leftrightarrow [\neg A \vee \neg B \Leftrightarrow \neg(A \wedge B)]$$

$$\forall x \forall y \neg[\text{person}(x) \wedge \text{vegetarian}(y) \wedge \text{smart}(y) \wedge \text{likes}(x,y)]$$

$$\Leftrightarrow [\forall x \neg(F) \Leftrightarrow \neg \exists x (F)]$$

$$\neg \exists x \exists y [\text{person}(x) \wedge \text{vegetarian}(y) \wedge \text{smart}(y) \wedge \text{likes}(x,y)]$$

Problem & Solution

- *There is a woman who likes all men who are not vegetarians.*

$$\exists x[\text{woman}(x) \wedge [\forall y [\text{man}(y) \wedge \neg \text{vegetarian}(y) \Rightarrow \text{likes}(x,y)]]]$$

Problem & Solution

- *The best score in history was better than the best score in biology.*

$\forall x \forall y [\text{bestscore}(\text{hist},x) \wedge \text{bestscore}(\text{bio},y) \Rightarrow \text{better}(x,y)]$

Problem & Solution

- *Every person who dislikes all vegetarians is smart.*

$$\forall x [\text{person}(x) \wedge [\forall y [\text{vegetarian}(y) \Rightarrow \neg \text{likes}(x,y)]] \Rightarrow \text{smart}(x)]$$

Problem & Solution

- *There is a barber who shaves all men in town who do not shave themselves.*

$$\exists x [\text{barber}(x) \wedge [\forall y [\text{townsman}(y) \wedge \neg \text{shaves}(y,y) \Rightarrow \text{shaves}(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x [\text{barber}(x) \wedge [\forall y [\neg [\text{townsman}(y) \wedge \neg \text{shaves}(y,y)] \vee \text{shaves}(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x [\text{barber}(x) \wedge [\forall y \neg [\text{townsman}(y) \wedge \neg \text{shaves}(y,y) \wedge \neg \text{shaves}(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x [\text{barber}(x) \wedge [\neg \exists y [\text{townsman}(y) \wedge \neg \text{shaves}(y,y) \wedge \neg \text{shaves}(x,y)]]]$$

Problem & Solution

- *No person likes a professor unless the professor is smart.*

$$\forall x \forall y [\text{person}(x) \wedge \text{professor}(y) \Rightarrow [\text{likes}(x,y) \Rightarrow \text{smart}(y)]] \Leftrightarrow$$

$$\forall x \forall y [\text{person}(x) \wedge \text{professor}(y) \Rightarrow [\neg \text{likes}(x,y) \vee \text{smart}(y)]] \Leftrightarrow$$

$$\forall x \forall y [\neg [\text{person}(x) \wedge \text{professor}(y)] \vee [\neg \text{likes}(x,y) \vee \text{smart}(y)]] \Leftrightarrow$$

$$\forall x \forall y [\neg [\text{person}(x) \wedge \text{professor}(y)] \vee \neg [\text{likes}(x,y) \wedge \neg \text{smart}(y)]] \Leftrightarrow$$

$$\forall x \forall y \neg [\text{person}(x) \wedge \text{professor}(y) \wedge \text{likes}(x,y) \wedge \neg \text{smart}(y)] \Leftrightarrow$$

$$\neg \exists x \exists y [\text{person}(x) \wedge \text{professor}(y) \wedge \text{likes}(x,y) \wedge \neg \text{smart}(y)]$$

Problem & Solution

- *Only one person failed both history and biology.*

$$\exists!x \text{ student}(x) \wedge \text{failed}(x, \text{hist}) \wedge \text{failed}(x, \text{bio})$$

Note that: $\exists!x p(x) \Leftrightarrow \exists x p(x) \wedge [\forall y [p(y) \Rightarrow x=y]]$

Problem & Solution

- *Politicians can fool some of the people all the time, and they can fool all of the people some of the time, but they can't fool all the people all of the time.*

$\forall x [\text{politician}(x) \Rightarrow [\exists y \text{ people}(y) \wedge [\forall t \text{ time}(t) \Rightarrow \text{fool}(x,y,t)]]]$

$\forall x [\text{politician}(x) \Rightarrow [\exists t \text{ time}(t) \wedge [\forall y \text{ people}(y) \Rightarrow \text{fool}(x,y,t)]]]$

$\forall x [\text{politician}(x) \Rightarrow \neg[\forall y \forall t [\text{people}(y) \wedge \text{time}(t)] \Rightarrow \text{fool}(x,y,t)]]]$

Exercises: Artificial Intelligence

Planning & Logic: And-Or-If

Problem & Solution

- *One more outburst like that and you are in contempt of court.*

outburst \Rightarrow court

NOT: outburst \wedge court

Problem & Solution

- *Either the Red Sox win or I'm out ten dollars.*

redSoxWin \Leftrightarrow \neg outTenDollars

NOT: redSoxWin \vee outTenDollars

Problem & Solution

- *Maybe I'll come to the party and maybe I won't.*

maybeComeToParty \vee \neg maybeComeToParty

NOT: maybeComeToParty \wedge \neg maybeComeToParty

Exercises: Artificial Intelligence

Planning & Logic: Weird Logic

Problem & Solution

- *I don't jump off the Empire State Building implies if I jump off the Empire State Building, then I float safely to the ground.*
 - *Translating the meaning of the sentence is not possible*

$$\begin{aligned} & \neg \text{jumpESB} \Rightarrow [\text{jumpESB} \Rightarrow \text{floatTTGround}] \Leftrightarrow \\ & \neg \text{jumpESB} \Rightarrow [\neg \text{jumpESB} \vee \text{floatTTGround}] \Leftrightarrow \\ & \text{jumpESB} \vee \neg \text{jumpESB} \vee \text{floatTTGround} \end{aligned}$$

Problem & Solution

- *It is not the case that if you attempt this exercise you will get an F. Therefore, you will attempt this exercise.*
 - *Translating the meaning of the sentence is not possible*

$$\neg[\text{attempt} \Rightarrow \text{getF}] \Rightarrow \text{attempt} \Leftrightarrow$$

$$\neg[\neg\text{attempt} \vee \text{getF}] \Rightarrow \text{attempt} \Leftrightarrow$$

$$\neg\text{attempt} \vee \text{getF} \vee \text{attempt}$$

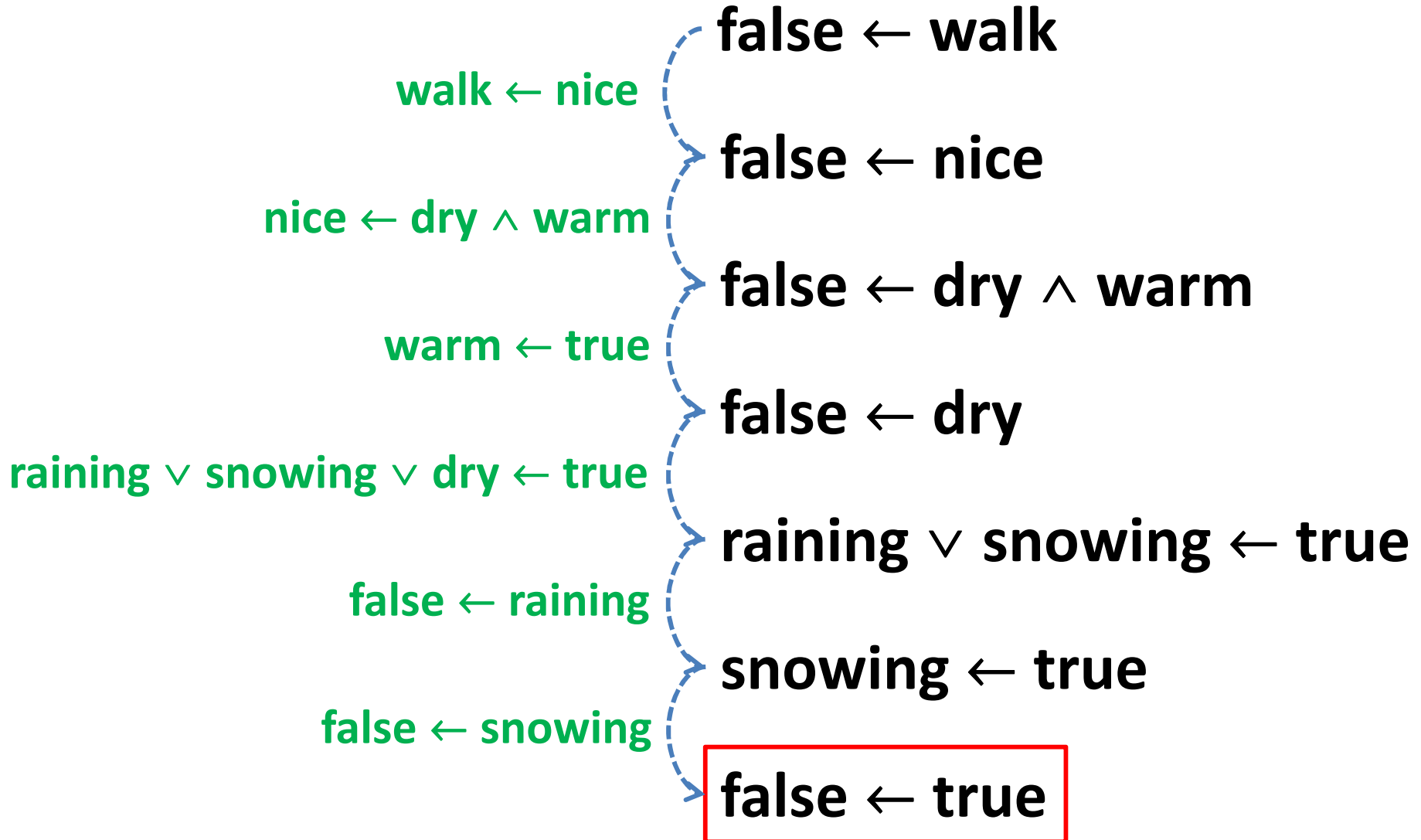
Exercises: Artificial Intelligence

Automated Reasoning: Good to walk

Solution

- *We assume that it is not good to walk:*
 - **false** \leftarrow **walk**
- *Given:*
 - raining \vee snowing \vee dry (\leftarrow true)
 - warm (\leftarrow true)
 - false \leftarrow raining
 - false \leftarrow snowing
 - walk \leftarrow nice
 - nice \leftarrow dry \wedge warm

Solution



Exercises: Artificial Intelligence

Automated Reasoning: MGU

Solution

MGU: $\{x/f(A), w/f(A), y/A\}$

Result: $p(f(A), f(A), g(z, A))$

- *What is the m.g.u. of: $p(f(y), w, g(z, y)) = p(x, x, g(z, A))$*
 - *Init: $p(f(y), w, g(z, y)) = p(x, x, g(z, A))$*
 - *Case 5: $f(y) = x, w = x, g(z, y) = g(z, A)$*
 - *Case 1: $x = f(y), w = x, g(z, y) = g(z, A)$*
 - *Case 4: $x = f(y), w = f(y), g(z, y) = g(z, A)$*
 - *Case 5: $x = f(y), w = f(y), z = z, y = A$*
 - *Case 2: $x = f(y), w = f(y), y = A$*
 - **Case 4: $x = f(A), w = f(A), y = A$**

Solution

- *What is the m.g.u. of: $p(A, x, f(g(y))) = p(z, f(z), f(A))$*
 - *Init: $p(A, x, f(g(y))) = p(z, f(z), f(A))$*
 - *Case 5: $A = z, x = f(z), f(g(y)) = f(A)$*
 - *Case 1: $z = A, x = f(z), f(g(y)) = f(A)$*
 - *Case 4: $z = A, x = f(A), f(g(y)) = f(A)$*
 - *Case 5: $z = A, x = f(A), g(y) = A$*
 - ***Case 5: stop := true***

Solution

- *What is the m.g.u. of: $q(x,x) = q(y,f(y))$*
 - *Init: $q(x,x) = q(y,f(y))$*
 - *Case 5: $x = y, x = f(y)$*
 - *Case 4: $x = y, y = f(y)$*
 - ***Case 3: stop := true***

Solution

MGU: $\{x/g(f(a),f(a)), u/f(a), v/f(a)\}$

Result: $f(g(f(a),f(a)),g(f(a), f(a)))$

- *What is the m.g.u. of: $f(x,g(f(a),u)) = f(g(u,v),x)$*
 - *Init: $f(x,g(f(a),u)) = f(g(u,v),x)$*
 - *Case 5: $x = g(u,v), g(f(a),u) = x$*
 - *Case 4: $x = g(u,v), g(f(a),u) = g(u,v)$*
 - *Case 5: $x = g(u,v), f(a) = u, u = v$*
 - *Case 1: $x = g(u,v), u = f(a), u = v$*
 - *Case 4: $x = g(f(a),v), u = f(a), f(a) = v$*
 - *Case 1: $x = g(f(a),v), u = f(a), v = f(a)$*
 - **Case 4: $x = g(f(a),f(a)), u = f(a), v = f(a)$**

Exercises: Artificial Intelligence

Automated Reasoning: Resolution

Solution

- *Assumption: Peter has no mother-in-law*
 - $\text{false} \leftarrow \text{mother-in-law}(x, \text{Peter})$
- *Given:*
 - $\text{mother-in-law}(x, y) \leftarrow \text{mother}(x, z) \wedge \text{married}(z, y)$
 - $\text{mother}(x, y) \leftarrow \text{female}(x) \wedge \text{parent}(x, y)$
 - $\text{female}(\text{An}) \leftarrow \text{true}$
 - $\text{parent}(\text{An}, \text{Maria}) \leftarrow \text{true}$
 - $\text{married}(\text{Maria}, \text{Peter}) \leftarrow \text{true}$

Solution

- $\text{false} \leftarrow \text{mother-in-law}(x, \text{Peter})$
 - $\text{mother-in-law}(x', y') \leftarrow \text{mother}(x', z') \wedge \text{married}(z', y')$
 - $\{x'/x, y'/\text{Peter}\}$
- $\text{false} \leftarrow \text{mother}(x, z') \wedge \text{married}(z', \text{Peter})$

Solution

- $\text{false} \leftarrow \text{mother-in-law}(x, \text{Peter})$
- $\text{false} \leftarrow \text{mother}(x, z') \wedge \text{married}(z', \text{Peter})$
 - $\text{mother}(x', y') \leftarrow \text{female}(x') \wedge \text{parent}(x', y')$
 - $\{x'/x, y'/z'\}$
- $\text{false} \leftarrow \text{female}(x) \wedge \text{parent}(x, z') \wedge \text{married}(z', \text{Peter})$

Solution

- $\text{false} \leftarrow \text{mother-in-law}(x, \text{Peter})$
- $\text{false} \leftarrow \text{mother}(x, z') \wedge \text{married}(z', \text{Peter})$
- $\text{false} \leftarrow \text{female}(x) \wedge \text{parent}(x, z') \wedge \text{married}(z', \text{Peter})$
 - $\text{female}(An)$
 - $\{x/An\}$
- $\text{false} \leftarrow \text{parent}(An, z') \wedge \text{married}(z', \text{Peter})$

Solution

- $\text{false} \leftarrow \text{mother-in-law}(x, \text{Peter})$
- $\text{false} \leftarrow \text{mother}(x, z') \wedge \text{married}(z', \text{Peter})$
- $\text{false} \leftarrow \text{female}(x) \wedge \text{parent}(x, z') \wedge \text{married}(z', \text{Peter})$
- $\text{false} \leftarrow \text{parent}(\text{An}, z') \wedge \text{married}(z', \text{Peter})$
 - $\text{parent}(\text{An}, \text{Maria})$
 - $\{z' / \text{Maria}\}$
- $\text{false} \leftarrow \text{married}(\text{Maria}, \text{Peter})$

Solution

$\{x/An\}$

- false \leftarrow mother-in-law(x,Peter)
- false \leftarrow mother(x,z') \wedge married(z',Peter)
- false \leftarrow female(x) \wedge parent(x,z') \wedge married(z',Peter)
- false \leftarrow parent(An,z') \wedge married(z',Peter)
- false \leftarrow married(Maria,Peter)
 - married(Maria,Peter)
- false \leftarrow true (\square)

Solution

- *Assumption: “There is no valid colouring”*
 - $false \leftarrow nb(b,g),nb(g,n),nb(n,b)$
- *Given:*
 - $c(R) \leftarrow true$
 - $c(G) \leftarrow true$
 - $c(B) \leftarrow true$
 - $nb(x,y) \leftarrow c(x), c(y), diff(x,y)$
 - $diff/2$ succeeds when arguments cannot be unified

Solution

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
 - $\text{nb}(x',y') \leftarrow c(x'), c(y'), \text{diff}(x',y')$
 - $\{x'/b, y'/g\}$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$

Solution

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
 - $\text{nb}(x',y') \leftarrow c(x'), c(y'), \text{diff}(x',y')$
 - $\{x'/g, y'/n\}$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$

Solution

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$
 - $\text{nb}(x',y') \leftarrow c(x'), c(y'), \text{diff}(x',y')$
 - $\{x'/n, y'/b\}$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,b)$

Solution

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,b)$
 - $c(R)$
 - $\{b/R\}$
- $\text{false} \leftarrow c(g) \wedge \text{diff}(R,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,R)$

Solution

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,b)$
- $\text{false} \leftarrow c(g) \wedge \text{diff}(R,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,R)$
 - $c(G)$
 - $\{g/G\}$
- $\text{false} \leftarrow \text{diff}(R,G) \wedge c(n) \wedge \text{diff}(G,n) \wedge \text{diff}(n,R)$

Solution

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,b)$
- $\text{false} \leftarrow c(g) \wedge \text{diff}(R,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,R)$
- $\text{false} \leftarrow \text{diff}(R,G) \wedge c(n) \wedge \text{diff}(G,n) \wedge \text{diff}(n,R)$
 - $c(B)$
 - $\{n/B\}$
- $\text{false} \leftarrow \text{diff}(R,G) \wedge \text{diff}(G,B) \wedge \text{diff}(B,R)$

Solution

$\{b/R, g/G, n/B\}$

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,b)$
- $\text{false} \leftarrow c(g) \wedge \text{diff}(R,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,R)$
- $\text{false} \leftarrow \text{diff}(R,G) \wedge c(n) \wedge \text{diff}(G,n) \wedge \text{diff}(n,R)$
- $\text{false} \leftarrow \text{diff}(R,G) \wedge \text{diff}(G,B) \wedge \text{diff}(B,R)$
 - Built-in $\text{diff}/2$: succeeds for different arguments
- $\text{false} \leftarrow \text{true} (\square)$

Alternative solution

b/B, g/G, n/R

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,b)$
- $\text{false} \leftarrow c(g) \wedge \text{diff}(\underline{\mathbf{B}},g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,\underline{\mathbf{B}})$
- $\text{false} \leftarrow \text{diff}(\underline{\mathbf{B}},G) \wedge c(n) \wedge \text{diff}(G,n) \wedge \text{diff}(n,\underline{\mathbf{B}})$
- $\text{false} \leftarrow \text{diff}(\underline{\mathbf{B}},G) \wedge \text{diff}(G,\underline{\mathbf{R}}) \wedge \text{diff}(\underline{\mathbf{R}},\underline{\mathbf{B}})$
 - Built-in diff/2: succeeds for different arguments
- $\text{false} \leftarrow \text{true} (\square)$

Or consistency = Continue search

- $\text{false} \leftarrow \text{nb}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge \text{nb}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{nb}(n,b)$
- $\text{false} \leftarrow c(b) \wedge c(g) \wedge \text{diff}(b,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,b)$
- $\text{false} \leftarrow c(g) \wedge \text{diff}(R,g) \wedge c(n) \wedge \text{diff}(g,n) \wedge \text{diff}(n,R)$
- $\text{false} \leftarrow \text{diff}(R,\underline{R}) \wedge c(n) \wedge \text{diff}(\underline{R},n) \wedge \text{diff}(n,R)$
- $\text{false} \leftarrow \text{diff}(R,\underline{R}) \wedge \text{diff}(\underline{R},B) \wedge \text{diff}(B,R)$
 - $\text{diff}(R,R)$ is false
- $\text{false} \leftarrow \text{false}$

Exercises: Artificial Intelligence

Automated Reasoning: Predicate
Resolution

Solution

- Formula in implicative normal form:

$$- \forall x p(x) \vee \neg r(f(x))$$

- $p(x) \leftarrow r(f(x))$

$$- \forall x \forall y r(f(x)) \vee r(f(f(y)))$$

- $r(f(x)) \vee r(f(f(y))) (\leftarrow \text{true})$

- Assumption

$$\neg [\forall x \exists y p(f(x)) \wedge r(y)] \Leftrightarrow \exists x \forall y \neg [p(f(x)) \wedge r(y)] \Leftrightarrow$$

$$\forall y \neg [p(f(A)) \wedge r(y)] \Leftrightarrow \text{false} \leftarrow p(f(A)) \wedge r(y)$$

Solution

- $\text{false} \leftarrow p(f(A)) \wedge r(y)$
 - $p(x') \leftarrow r(f(x'))$
 - $\{x'/f(A)\}$
- $\text{false} \leftarrow r(f(f(A))) \wedge r(y)$

Solution

- $\text{false} \leftarrow p(f(A)) \wedge r(y)$
- $\text{false} \leftarrow r(f(f(A))) \wedge r(y)$
 - Factoring: $\text{mgu}(r(f(f(A))) = r(y)) = \{y/f(f(A))\}$
- $\text{false} \leftarrow r(f(f(A))) \wedge r(f(f(A)))$

Solution

$$\{y/f(f(A))\}$$

- $\text{false} \leftarrow p(f(A)) \wedge r(y)$
- $\text{false} \leftarrow r(f(f(A))) \wedge r(y)$
- $\text{false} \leftarrow r(f(f(A))) \wedge r(f(f(A)))$
 - $r(f(x')) \vee r(f(f(y')))) (\leftarrow \text{true})$
 - Factoring: $\text{mgu}(r(f(x')) = r(f(f(y')))) = \{x'/f(y')\}$
 - $r(f(f(y')))) (\leftarrow \text{true})$
 - $\{y'/A\}$
- $\text{false} \leftarrow \text{true} (\square)$

Exercises: Artificial Intelligence

Automated Reasoning:
Movable Objects

Solution: Movable Objects

- ***English to logic***
- ***Logic to implicative normal form***
 - *Model*
 - *Assumption to prove*
- ***Apply resolution***
 - *Derive inconsistency:*
 - *Model + negated assumption*

Solution: Model to logic

- If all movable objects are blue, then all non-movable objects are green.
 - $(\forall x \text{ mov}(x) \rightarrow \text{blue}(x)) \rightarrow (\forall y \neg \text{mov}(y) \rightarrow \text{green}(y))$
- If there exists a non-movable object, then all movable objects are blue.
 - $(\exists x \neg \text{mov}(x)) \rightarrow (\forall y \text{ mov}(y) \rightarrow \text{blue}(y))$
- D is a non-movable object.
 - $\neg \text{mov}(D)$

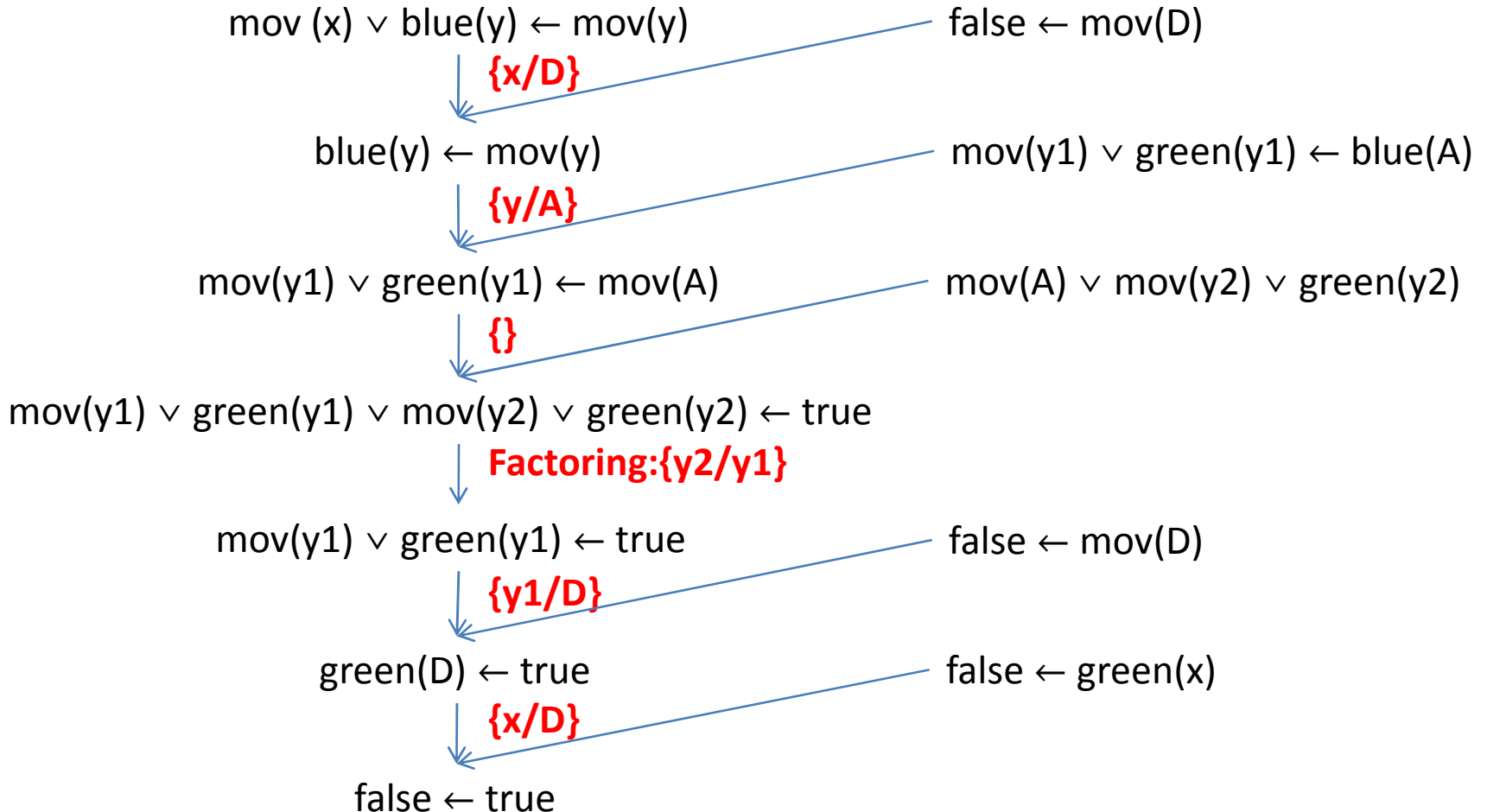
Solution: Implicative normal form

- $(\forall x \text{ mov}(x) \rightarrow \text{blue}(x)) \rightarrow (\forall y \neg \text{mov}(y) \rightarrow \text{green}(y))$
 - $\text{mov}(A) \vee \text{mov}(y) \vee \text{green}(y) \leftarrow \text{true}$
 - $\text{mov}(y) \vee \text{green}(y) \leftarrow \text{blue}(A)$
- $(\exists x \neg \text{mov}(x)) \rightarrow (\forall y \text{ mov}(y) \rightarrow \text{blue}(y))$
 - $\text{mov}(x) \vee \text{blue}(y) \leftarrow \text{mov}(y)$
- $\neg \text{mov}(D)$
 - $\text{false} \leftarrow \text{mov}(D)$
- Negated assumption: $\neg \exists x \text{ green}(x) \leftrightarrow \forall x \neg \text{green}(x)$
 - $\text{false} \leftarrow \text{green}(x)$

Solution: Implicative normal form

- **Prove** using resolution:
 - Assumption: **false** \leftarrow **green(x)**
- **Model:**
 - $\text{mov}(A) \vee \text{mov}(y) \vee \text{green}(y) (\leftarrow \text{true})$
 - $\text{mov}(y) \vee \text{green}(y) \leftarrow \text{blue}(A)$
 - $\text{mov}(x) \vee \text{blue}(y) \leftarrow \text{mov}(y)$
 - $\text{false} \leftarrow \text{mov}(D)$

Solution: Resolution



Exercises: Artificial Intelligence

Automated Reasoning: Politicians

Problem: Politicians

- ***Given:***

- If a poor politician exists, then all politicians are male.
- If people are friends with a politician, then this politician is poor and female.
- Lazy people have no friends.
- People are either male or female, but not both.
- If Joel is not lazy, then he is a politician.

- ***Proof by resolution:***

- There exists no person who is a friend of Joel.

Solution: English to logic

- $(\exists x \text{ pol}(x) \wedge \text{poor}(x)) \rightarrow (\forall y \text{ pol}(y) \rightarrow \text{male}(y)).$
- $\forall x (\text{pol}(x) \wedge (\exists y \text{ fr}(y,x))) \rightarrow \text{poor}(x) \wedge \text{fem}(x).$
- $\forall x \text{ lazy}(x) \rightarrow (\neg(\exists y \text{ fr}(y,x))).$
- $\forall x (\text{male}(x) \vee \text{fem}(x)) \wedge (\neg(\text{male}(x) \wedge \text{fem}(x))).$
- $\neg \text{lazy}(\text{Joel}) \rightarrow \text{pol}(\text{Joel}).$

Solution: Implicative normal form

- $\text{male}(y) \leftarrow \text{pol}(x) \wedge \text{poor}(x) \wedge \text{pol}(y)$
- $\text{poor}(x) \leftarrow \text{pol}(x) \wedge \text{fr}(y,x)$
- $\text{fem}(x) \leftarrow \text{pol}(x) \wedge \text{fr}(y,x)$
- $\text{false} \leftarrow \text{lazy}(x) \wedge \text{fr}(y,x)$
- $\text{male}(x) \vee \text{fem}(x)$
- $\text{false} \leftarrow \text{male}(x) \wedge \text{fem}(x)$
- $\text{lazy}(\text{Joel}) \vee \text{pol}(\text{Joel})$

Solution: Implicative normal form

- Prove:
 - There exists no person who is a friend of Joel
 - $\neg \exists x \text{ fr}(x, \text{Joel}) \leftrightarrow \forall x \neg \text{fr}(x, \text{Joel})$
- Negate assumption:
 - There exists a person who is a friend of Joel
 - $\exists x \text{ fr}(x, \text{Joel})$
 - Call the friend S
 - $\text{fr}(S, \text{Joel})$

Solution: Implicative normal form

- $\text{male}(y) \leftarrow \text{pol}(x) \wedge \text{poor}(x) \wedge \text{pol}(y)$
- $\text{poor}(x) \leftarrow \text{pol}(x) \wedge \text{fr}(y,x)$
- $\text{fem}(x) \leftarrow \text{pol}(x) \wedge \text{fr}(y,x)$
- $\text{false} \leftarrow \text{lazy}(x) \wedge \text{fr}(y,x)$
- $\text{male}(x) \vee \text{fem}(x)$
- $\text{false} \leftarrow \text{male}(x) \wedge \text{fem}(x)$
- $\text{lazy}(\text{Joel}) \vee \text{pol}(\text{Joel})$
- **$\text{fr}(\text{S}, \text{Joel})$**

Solution: Apply Resolution

- $\text{male}(y1) \leftarrow \text{pol}(x1) \wedge \underline{\text{poor}(x1)} \wedge \text{pol}(y1)$
 - $\underline{\text{poor}(x2)} \leftarrow \text{pol}(x2) \wedge \text{fr}(y2,x2)$
 - RESOLUTION: $\{x2/x1\}$
- $\text{male}(y1) \leftarrow \underline{\text{pol}(x1)} \wedge \underline{\text{pol}(y1)} \wedge \text{fr}(y2,x1)$
 - FACTORING: $\{y1/x1\}$
- $\text{male}(x1) \leftarrow \text{pol}(x1) \wedge \text{fr}(y2,x1)$
 - *'Politicians who have friends must be male'*

Solution: Apply Resolution

- **male(x1) \leftarrow pol(x1) \wedge fr(y2,x1)**
 - false \leftarrow male(x3) \wedge fem(x3)
 - RESOLUTION: {x3/x1}
- **false \leftarrow pol(x1) \wedge fr(y2,x1) \wedge fem(x1)**
 - *'Politicians who have friends cannot be female'*

Solution: Apply Resolution

- **false** \leftarrow **pol(x1)** \wedge **fr(y2,x1)** \wedge **fem(x1)**
 - **fem(x4)** \leftarrow **pol(x4)** \wedge **fr(y4,x4)**
 - RESOLUTION: {x4/x1}
- **false** \leftarrow **pol(x1)** \wedge **fr(y2,x1)** \wedge **pol(x1)** \wedge **fr(y4,x1)**
 - FACTORING: {}
- **false** \leftarrow **pol(x1)** \wedge **fr(y2,x1)** \wedge **fr(y4,x1)**
 - FACTORING: {y4/y2}
- **false** \leftarrow **pol(x1)** \wedge **fr(y2,x1)**
 - *'Politicians do not have friends'*

Solution: Apply Resolution

- **false** \leftarrow **pol(x1)** \wedge **fr(y2,x1)**
 - lazy(Joel) \vee pol(Joel)
 - RESOLUTION: {x1/Joel}
- **lazy(Joel)** \leftarrow **fr(y2,Joel)**
 - *'If Joel has friend, then he must be lazy'*

Solution: Apply Resolution

- **lazy(Joel) \leftarrow fr(y2,Joel)**
 - false \leftarrow lazy(x5) \wedge fr(y5,x5)
 - RESOLUTION: {x5/Joel}
- false \leftarrow **fr(y2,Joel) \wedge fr(y5,Joel)**
 - FACTORING: {y5/y2}
- false \leftarrow fr(y2,Joel)
 - *'Joel does not have any friends'*

Solution: Apply Resolution

- **false** \leftarrow **fr(y2,Joel)**
 - fr(S,Joel)
 - RESOLUTION: {y2/S}
- **false** \leftarrow **true**