CSCI 561 - Foundation for Artificial Intelligence

Discussion Section (Week 5) Midterm-1 Review

PROF WEI-MIN SHEN WMSHEN@USC.EDU

Material covered by midterm1

- **Covers everything studied in class up to and including CSP and RL (not include “logic”)**
- **Lectures vs book: what to know?**

- If something is covered both in the book and the slides of lecture/discussion: use the slides.

- If something is covered in the book only and was not covered at all in the lecture/discussions: you do not need to know it.

- If something is covered in the book and in the slides of lecture/discussion but with additional details provided in the book: you need to know both, and use the slides for the overlapping parts.

Midterm 1 Instructions:

- **Sep 28, 2022, 5-6:50PM, join your Zoom meeting**
- **Maximum credits/points for this midterm: 100 points**
- **Credits/points for each question is indicated on the question**
- **Closed book**
- **No books or any other material are allowed**
- **<u>Please practice</u> in your sample exam-0 on DEN**
- **<u>Please get your camera checked by a TA</u>**
- **<u>Please test</u> your lockdown browser and make sure you know how to enter your answers**
- **No questions during the exam**
- **Be brief: a few words are often enough if they are precise and use the correct vocabulary studied in class**
- **Make sure your environment for the exam is quite/exclusive**

Sample exam Questions

1. **General AI**
2. **Search Concepts**
3. **Comparing Strategies**
4. **Game Playing**
5. **CSP**
6. **RL (use examples in the lecture)**

Note: The actual exam questions will be different or harder/easier then those distributed in the sample papers or here.

___F__ The Turing test defines the conditions under which a machine can be said to be "intelligent".

___F__ A* is an admissible algorithm.

___F__ DFS is faster than BFS.

___T__ DFS has lower asymptotic space complexity than BFS.

___F__ When using the correct temperature decrease schedule, simulated annealing is guaranteed to find the global optimum in finite time.

_____F_____ Alpha-beta pruning accelerates game playing at the cost of being an approximation to full minimax.

_____F_____ Hill-climbing is an entirely deterministic algorithm.

_____T_____ The exact evaluation function values do not affect minimax decision as long as the ordering of these values is maintained.

_____F_____ A perfectly rational backgammon-playing agent never loses

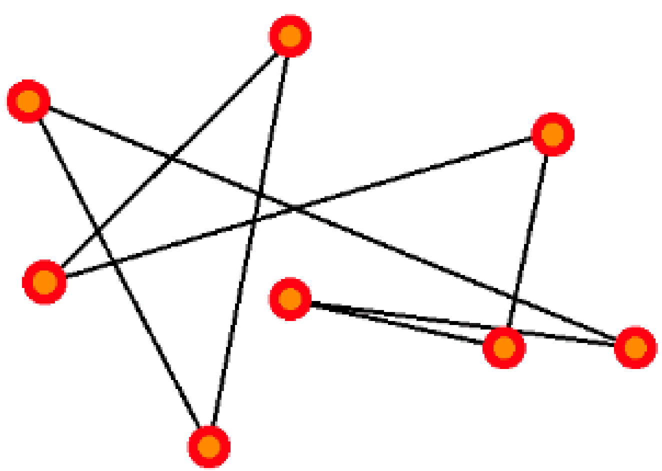
_____T_____ Hill climbing search is best used for problem domains with densely packed goals

- A suitable representation for states: permutation of all cities in the tour

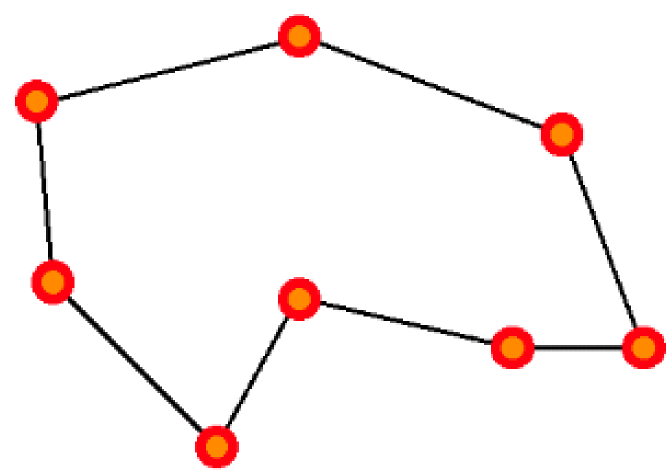
  <A, B, C, D, E>
- The initial state of the problem: random permutation of all cities
- A good goal test to use in this problem: minimize the distance travelled
- Good operators to use for search: permute 2 cities
- Which search algorithm would be the most appropriate to use here if we want to minimize the distance of the tour found?

Local Search - GA/SA/hill climbing, etc…

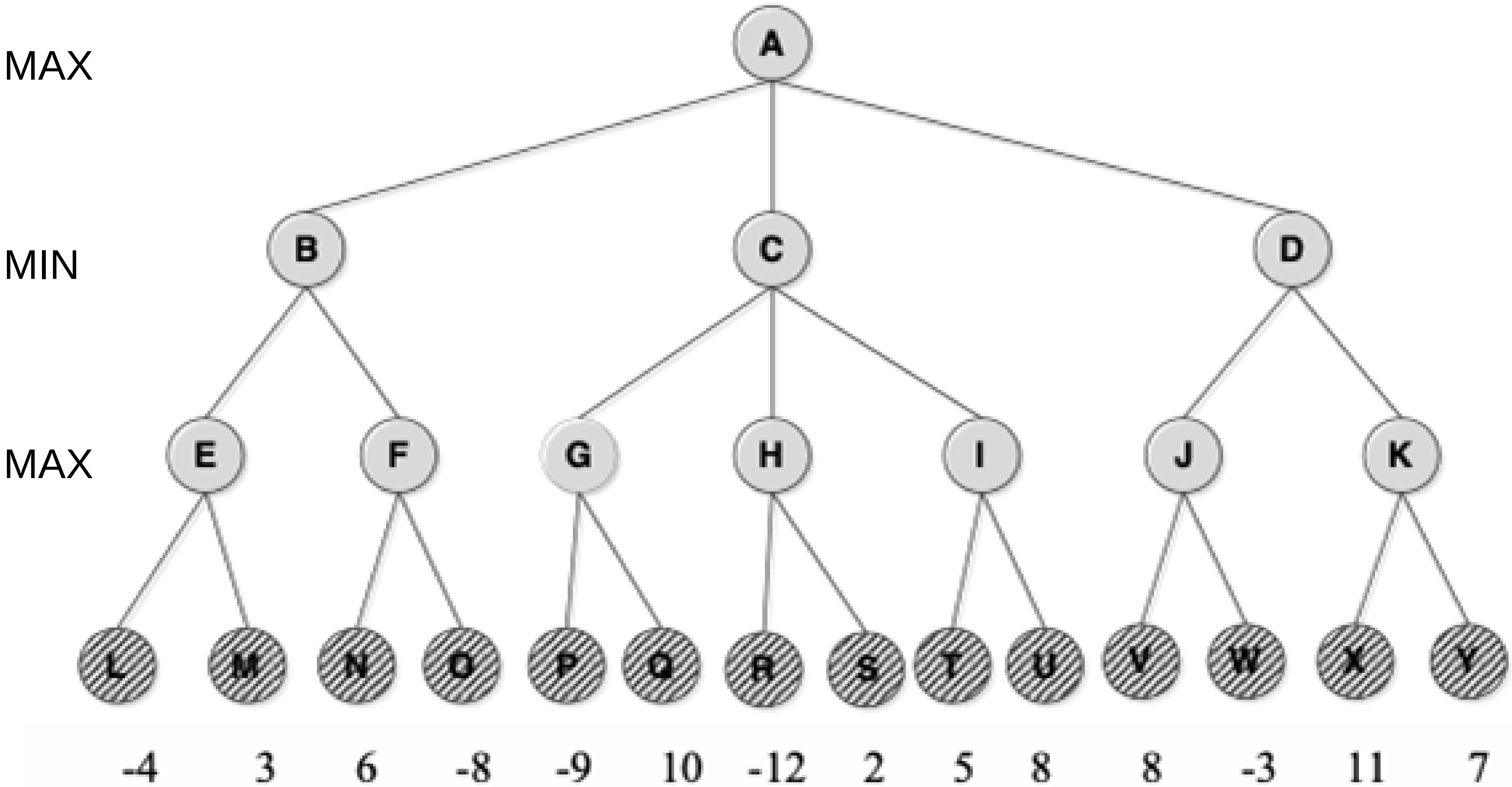
Suboptimal solution (long path)



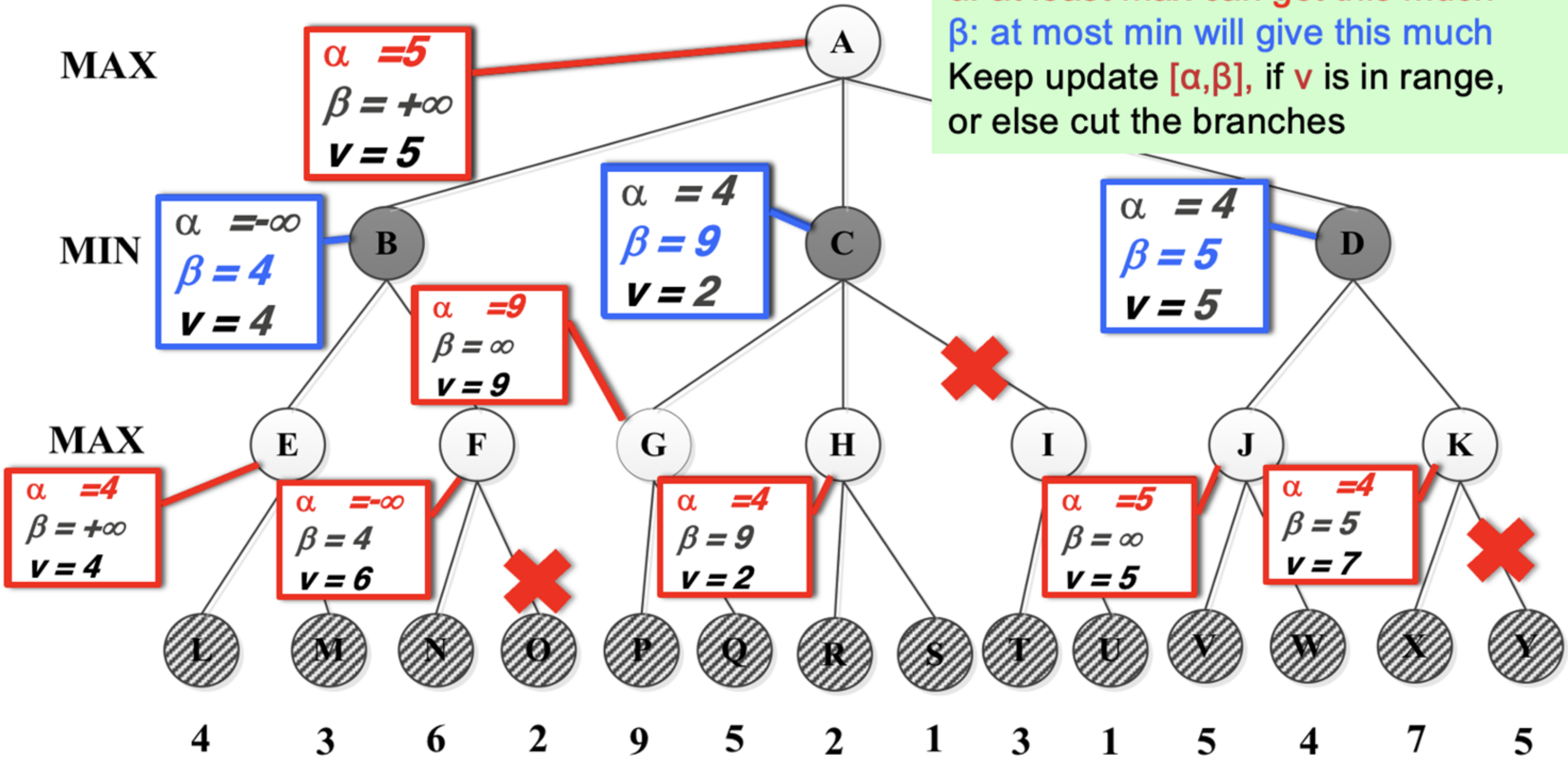
Optimal solution



Minimax


MAX
MIN
MAX
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
-4
3
6
-8
-9
10
-12
2
5
8
8
-3
11
7

α: at least max can get this much
β: at most min will give this much
Keep update [α,β], if v is in range, or else cut the branches



Max Node

for each a **in** ACTIONS($state$) **do**
  $v \leftarrow \text{MAX}(v, \text{MIN-VALUE}(\text{RESULT}(s,a), \alpha, \beta))$
  if $v \geq \beta$ **then return** v
  $\alpha \leftarrow \text{MAX}(\alpha, v)$
return v

Min Node

for each a **in** ACTIONS($state$) **do**
  $v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(\text{RESULT}(s,a), \alpha, \beta))$
  if $v \leq \alpha$ **then return** v
  $\beta \leftarrow \text{MIN}(\beta, v)$
return v

The schedules of the customers are:

Company 1: Webflix: 8:00-9:00am

Company 2: Anazon: 8:30-9:30am

Company 3: Pied Piper: 9:00-10:00am

Company 4: Hooli: 9:00-10:00am

Company 5: Gulu: 9:30-10:30am

The profiles of your engineers are:

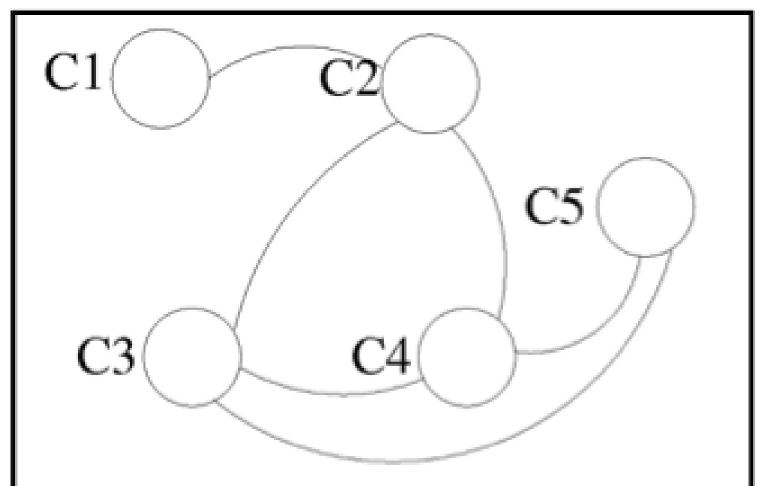
1) Albacore can maintain Pied Piper and Hooli.
2) Bosam can maintain all companies, but Webflix.
3) Coleslaw can maintain all companies.

Using Company as variable, formulate this problem as a CSP problem with variables, domains, and constraints. Constraints should be specified formally and precisely, but may be implicit rather than explicit.

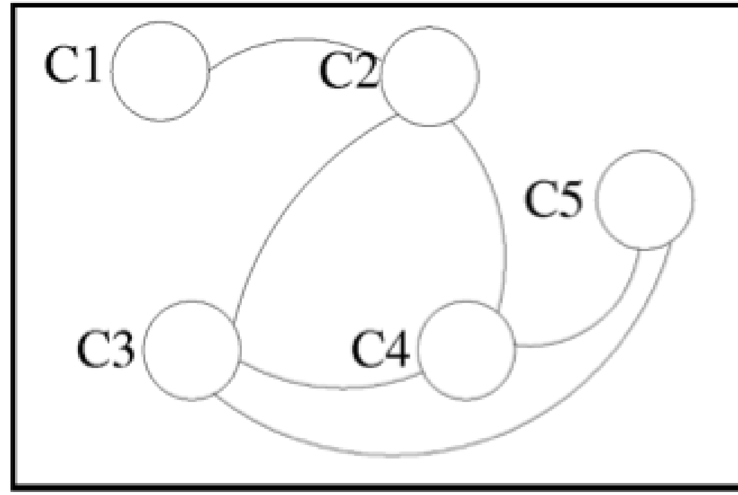
Draw the constraint graph associated with your CSP.

Variable	Domain
C1	C
C2	BC
C3	ABC
C4	ABC
C5	BC

Constraints:
$C1 \neq C2$, $C2 \neq C3$, $C3 \neq C4$, $C4 \neq C5$, $C2 \neq C4$, $C3 \neq C5$.



Variable	Domain
C1	C
C2	BC
C3	ABC
C4	ABC
C5	BC



Constraints:
C1 ≠ C2, C2 ≠ C3, C3 ≠ C4, C4 ≠ C5, C2 ≠ C4, C3 ≠ C5.

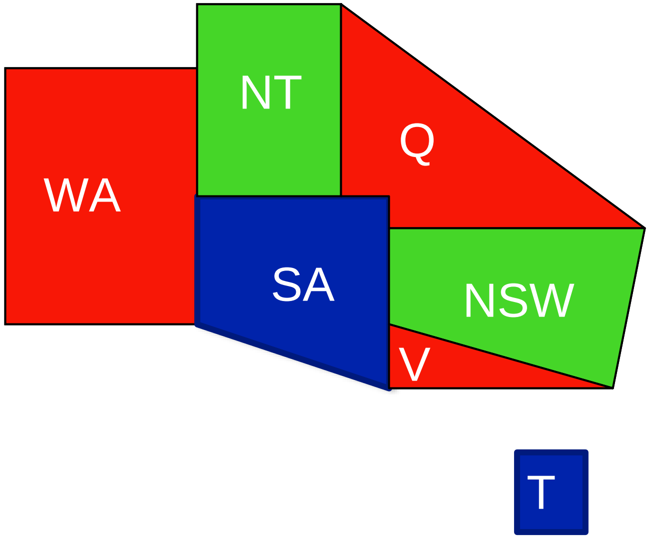
Show the domains of the variables after running arc-consistency on this initial graph (after having already enforced any unary constraints).

Give one solution to this CSP.

C1 = C, C2 = B, C3 = C, C4 = A, C5 = B.

Variable	Domain
C1	C
C2	B
C3	AC
C4	AC
C5	BC

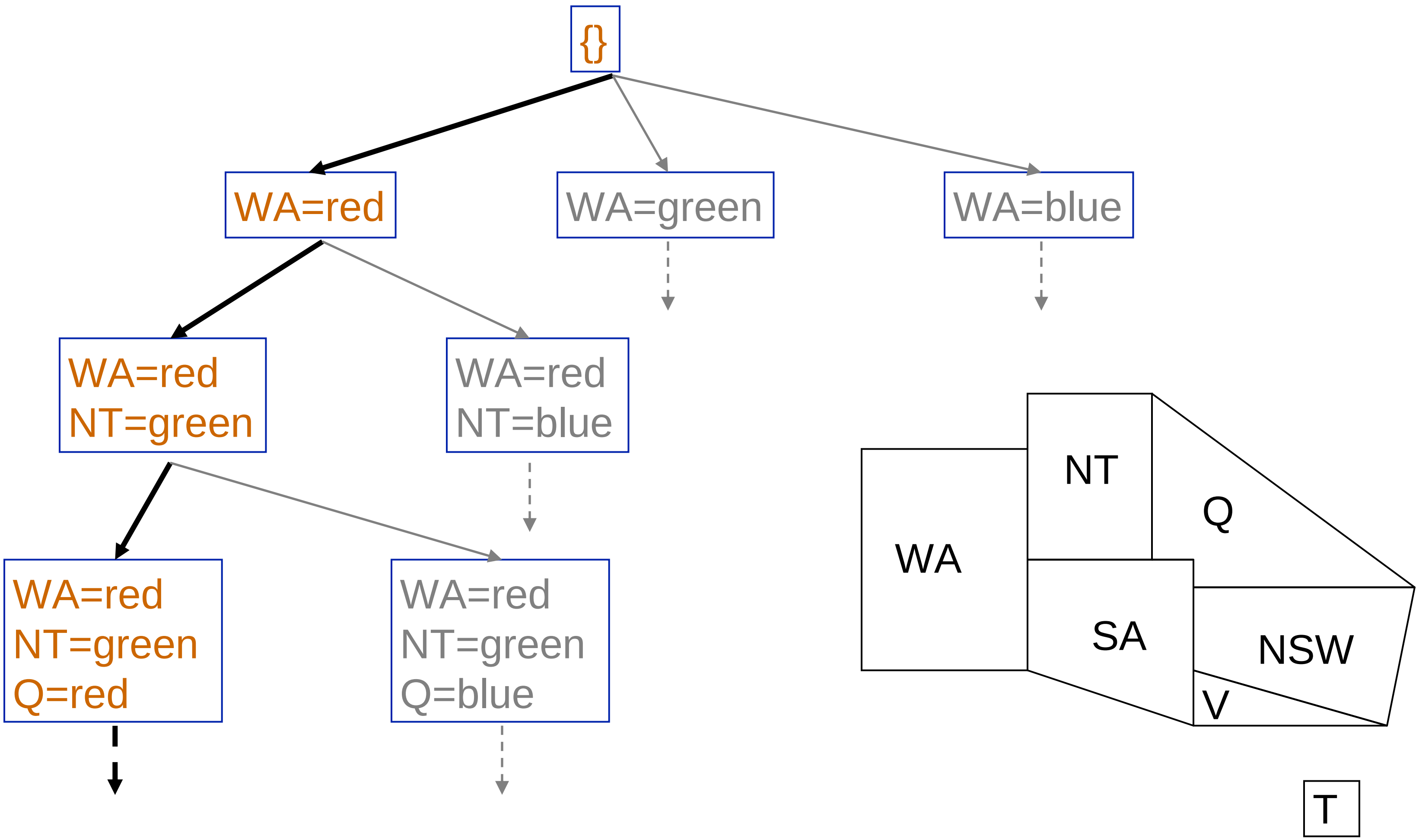
CSP Example: Map Coloring



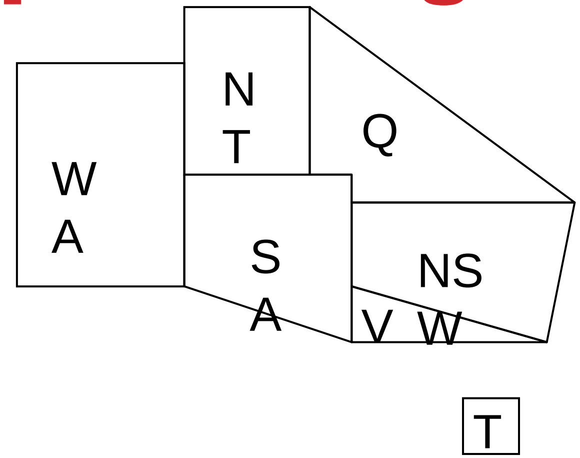
- 7 variables {WA,NT,SA,Q,NSW,V,T}
- Each variable has the same domain {red, green, blue}
- No two adjacent variables have the same value:

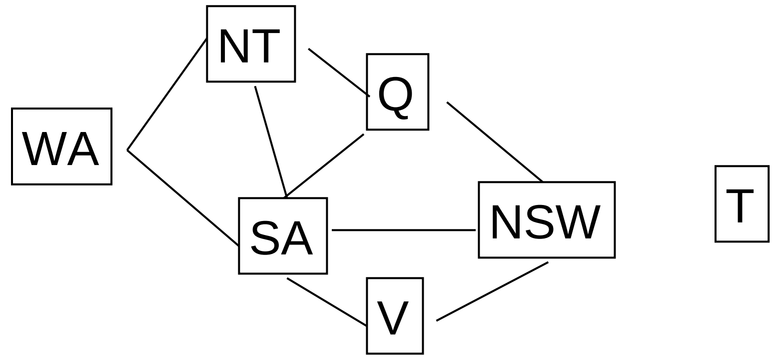
WA≠NT, WA≠SA, NT≠SA, NT≠Q, SA≠Q, SA≠NSW, SA≠V,Q≠NSW, NSW≠V

Backtracking Search: Map Coloring



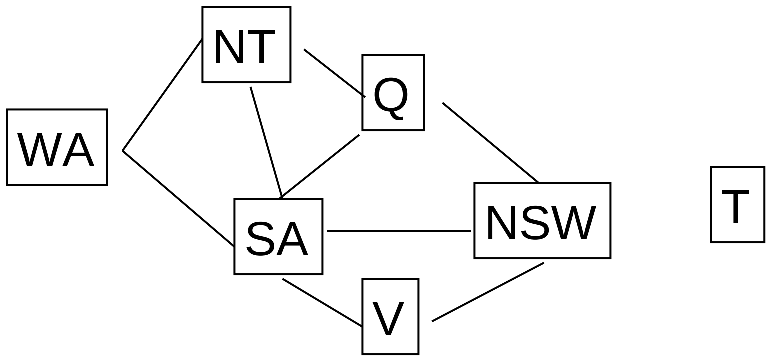
Map Coloring: Forward Checking





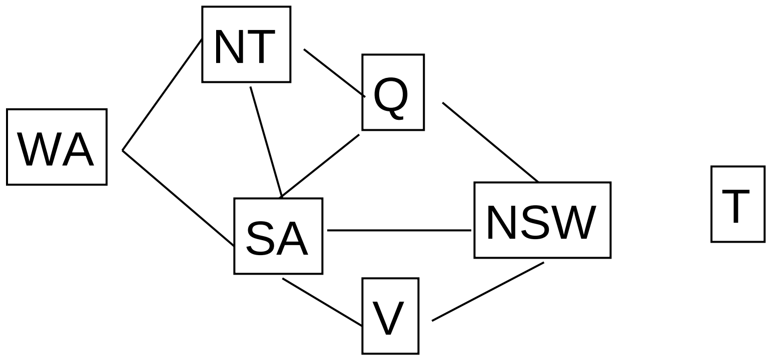
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
1: R	RGB	RGB	RGB	RGB	RGB	RGB

Map Coloring: Forward Checking



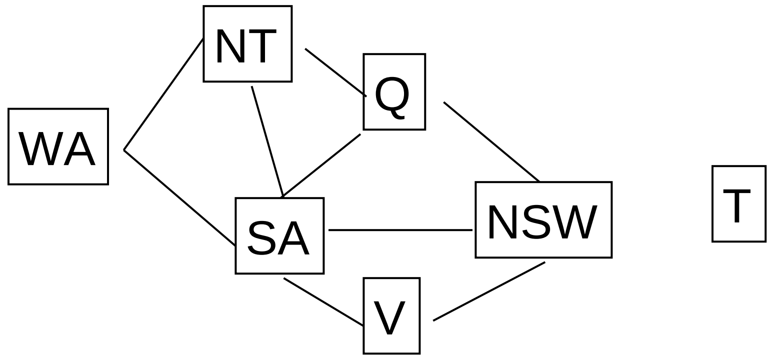
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
1: R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB

Map Coloring: FC



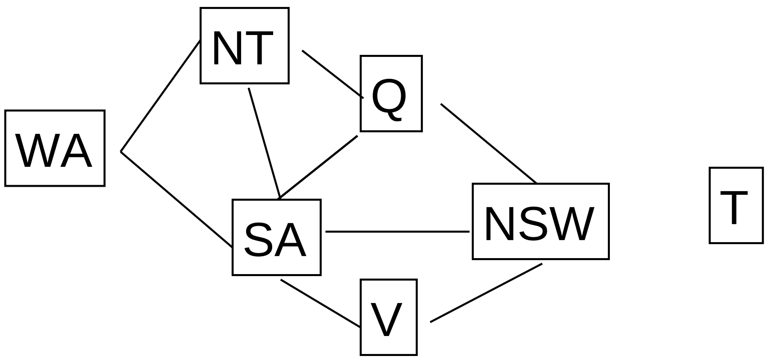
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~GB	2: G	RGB	RGB	~~R~~GB	RGB

Map Coloring: FC



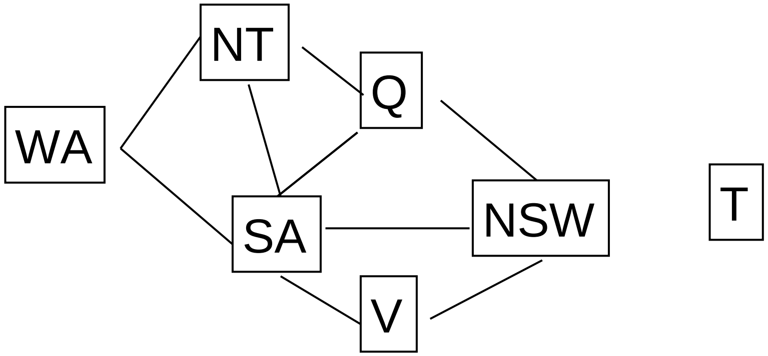
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~RG~~B	2: G	R~~G~~B	RGB	~~RG~~B	RGB

Map Coloring: FC



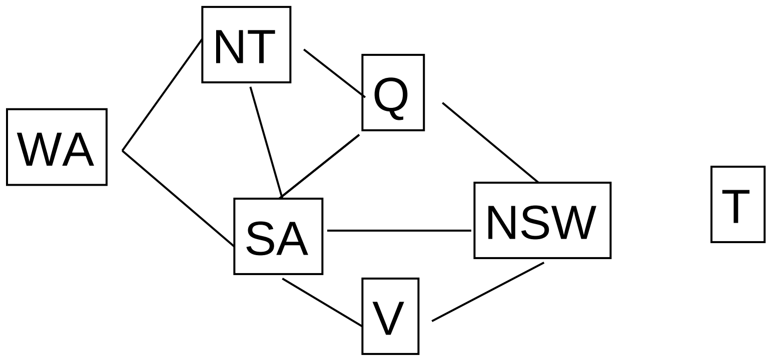
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~~~G~~B	G	R~~G~~B	RGB	~~R~~~~G~~B	RGB
R	~~R~~~~G~~B	G	R~~G~~B	3:B	~~R~~~~G~~B	RGB

Map Coloring: FC



WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~~~G~~B	G	R~~G~~B	RGB	~~R~~~~G~~B	RGB
R	~~R~~~~G~~B	G	R~~G~~~~B~~	3:B	~~R~~~~G~~~~B~~	RGB

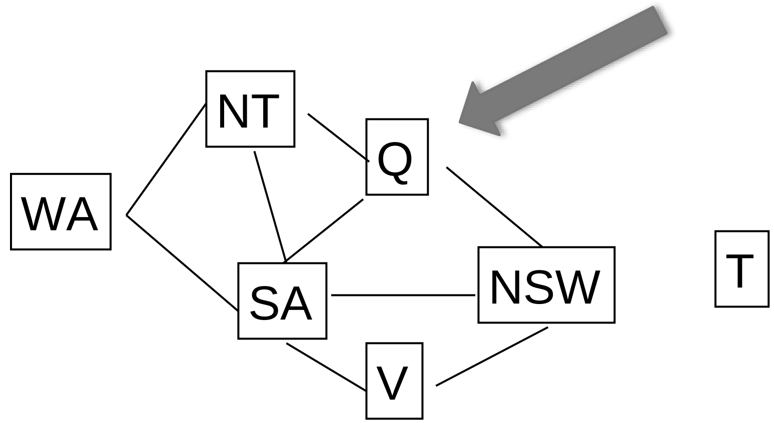
Other inconsistencies



Impossible assignments that forward checking does not detect

WA	NT	Q				
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~~~G~~B	G	R~~G~~B	RGB	~~R~~~~G~~B	RGB
R	~~R~~~~G~~B	G	R~~G~~~~B~~	3:B	~~R~~~~G~~~~B~~	RGB

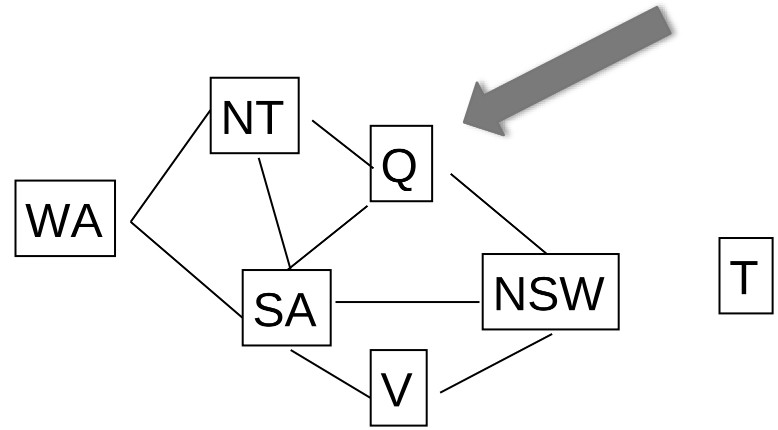
Map Coloring: Constraint Propagation



WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~GB	2: G	RGB	RGB	~~R~~GB	RGB

Go back to assigning "GREEN" to Queensland

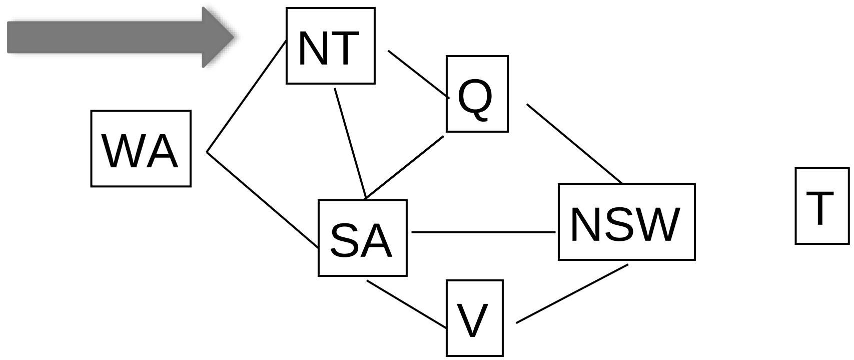
Map Coloring: CP



WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~~~G~~B	2: G	R~~G~~B	RGB	~~R~~~~G~~B	RGB

Immediate propagation removes GREEN for NSW, SA & NT

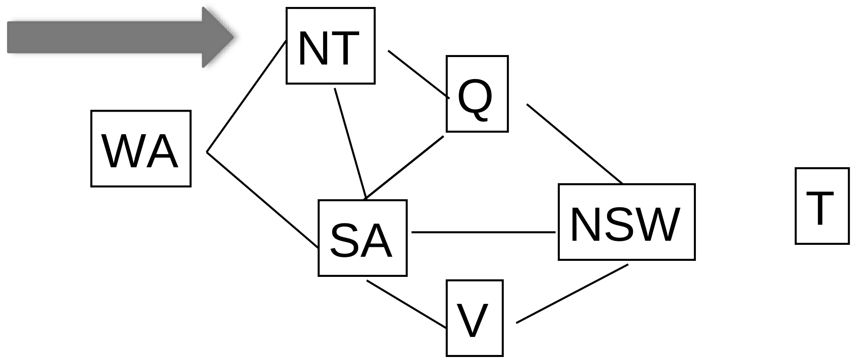
Map Coloring: CP



WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~~~G~~B	G	R~~G~~B	RGB	~~R~~~~G~~B	RGB

Since possible values for NT changed, continue to check arc consistency from NT

Map Coloring: CP



WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~ ~~G~~B	G	R~~G~~B	RGB	~~R~~ ~~G~~ ~~B~~	RGB

Constraint NT≠SA

Constraint violation with SA immediately detected

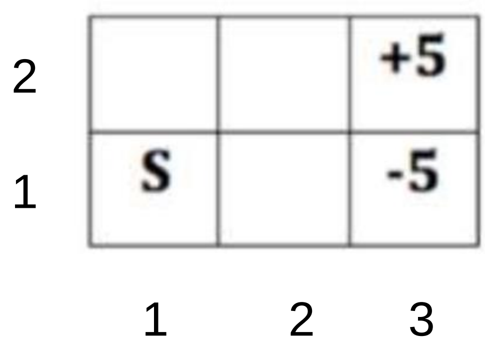
MDP and Q-Learning

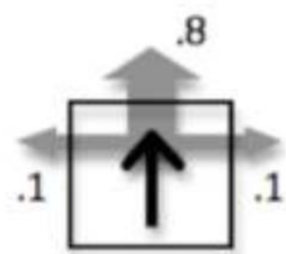
The grid world MDP shown below operates like to the one we saw in class.

The states are grid squares, identified by their row and column number (row first).

The agent always starts in state (1,1), marked with the letter S. There are two terminal goal states, (2,3) with reward +5 and (1,3) with reward -5. Rewards are 0 in non-terminal states. (The reward for a state is received as the agent moves into the state.)

The transition function is such that the intended agent movement (North, South, West, or East) happens with probability .8. With probability .1 each, the agent ends up in one of the states perpendicular to the intended direction. If a collision with a wall happens, the agent stays in the same state.





MDP and Q-Learning

9.a. [4%] Draw the optimal policy for this grid. Draw it directly on the grid world above.

9.b. [6%] Suppose the agent knows the transition probabilities. Give the first two rounds of value iteration updates for each state, with a discount of 0.9. (Assume V_0 is 0 everywhere and compute V_i for times i = 1, 2).

State	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)
V_0	0	0	0	0	0	0
V_1						
V_2						

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_a \sum_{s'} T(s,a,s')U_i(s')$$

MDP and Q-Learning

9.a. [4%] Draw the optimal policy for this grid. Draw it directly on the grid world above.

9.b. [6%] Suppose the agent knows the transition probabilities. Give the first two rounds of value iteration updates for each state, with a discount of 0.9. (Assume V_0 is 0 everywhere and compute V_i for times i = 1, 2).

State	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)
V_0	0	0	0	0	0	0
V_1	0	0	-5	0	0	+5
V_2						

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_a \sum_{s'} T(s,a,s') U_i(s')$$

MDP and Q-Learning

9.a. [4%] Draw the optimal policy for this grid. Draw it directly on the grid world above.

9.b. [6%] Suppose the agent knows the transition probabilities. Give the first two rounds of value iteration updates for each state, with a discount of 0.9. (Assume V_0 is 0 everywhere and compute V_i for times i = 1, 2).

State	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)
V_0	0	0	0	0	0	0
V_1	0	0	-5	0	0	+5
V_2	0	0	-5	0		+5

0 + 0.9 * max{
left 0.8*0+0.1*0+0.1*0,
up 0.1*0+0.8*0+0.1*5,
down 0.1*0+0.8*0+0.1*5,
right 0.1*0+0.1*0+0.8*5}
= 0 + 0.9 * {4.0} = 3.6

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_a \sum_{s'} T(s,a,s')U_i(s')$$