CSCI 561 - Foundation for Artificial Intelligence

Discussion Section (Week 5)
Midterm-1 Review

PROF WEI-MIN SHEN SHEN@ISI.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:

- Feb 19, 2020, 5-6:30PM, SGM123
- Maximum credits/points for this midterm: 100 points
- Credits/points for each question is indicated on the question
- Closed book
- <u>No books</u> or any other material are allowed
- Draw your number and go to your seat
- Leave your bags/phones in front of the room
- Write down your name and student ID
- 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
- Just bring pens/erasers and your seat number.
- Nothing else on table.

Sample exam Questions

1. General AI
2. Search Concepts
3. Comparing Strategies
4. Game Playing
5. CSP
6. RL (self-review examples in lecture)

Note: The sample questions posted on DEN may be different or harder/easier then the actual questions in Midterm1 exam (90 min)

____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

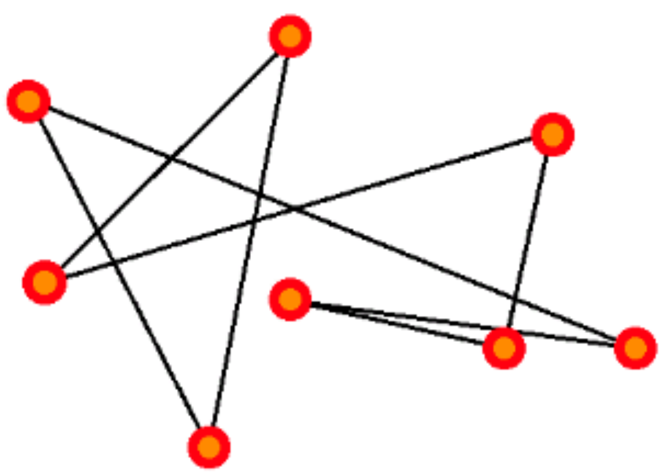
__T__ Hill climbing search is best used for problem domains with densely packed goals

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

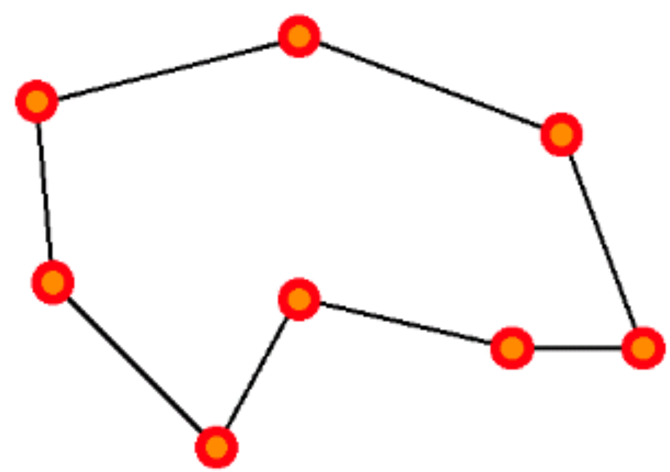
  <A, B, C, D, E>
- The initial state of the problem: random permutation of all cities
- A good goal test to use in this problem: minimize the distance travelled
- Good operators to use for search: permute 2 cities
- Which search algorithm would be the most appropriate to use here if we want to minimize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

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

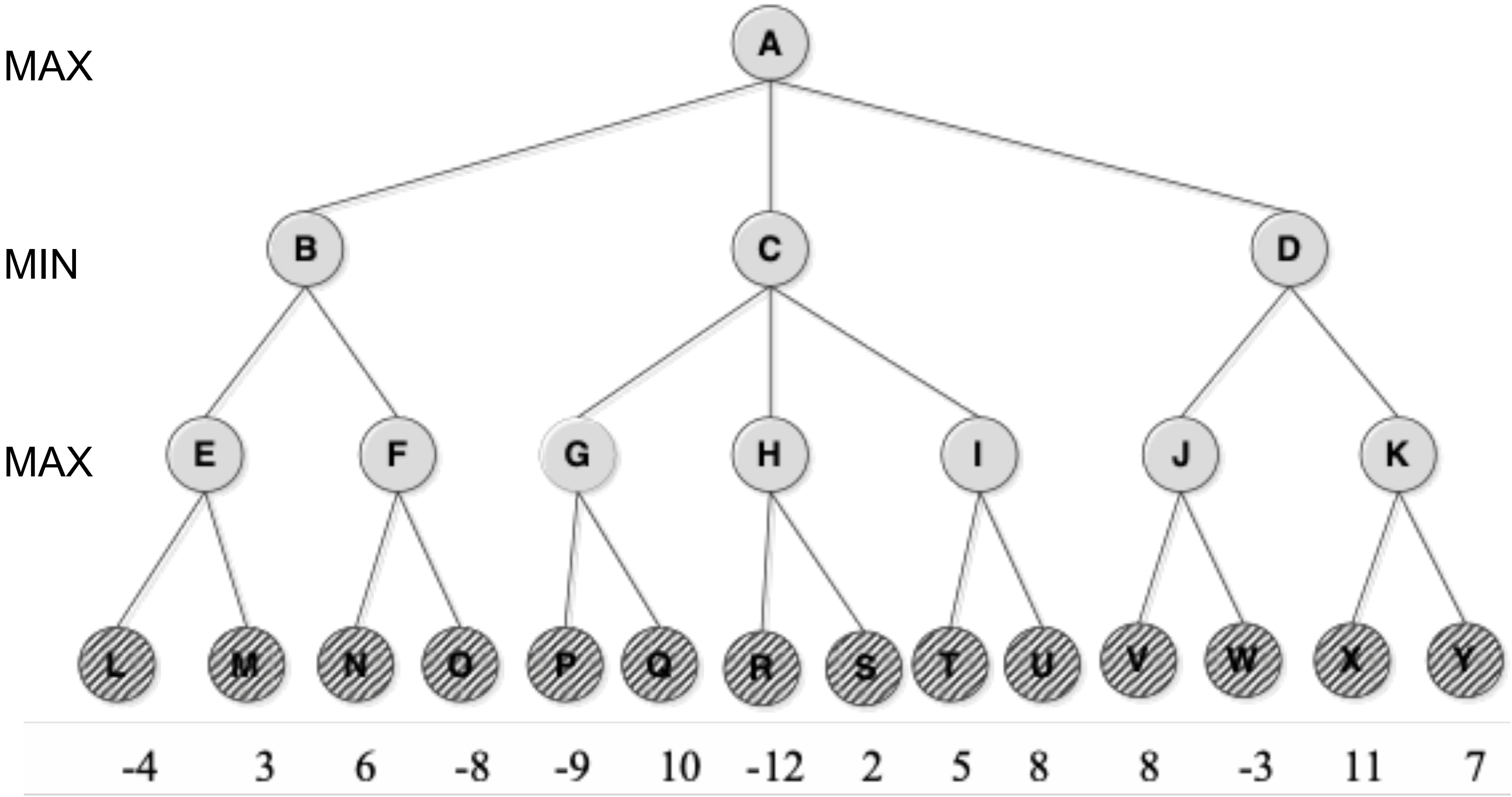
Suboptimal solution (long path)



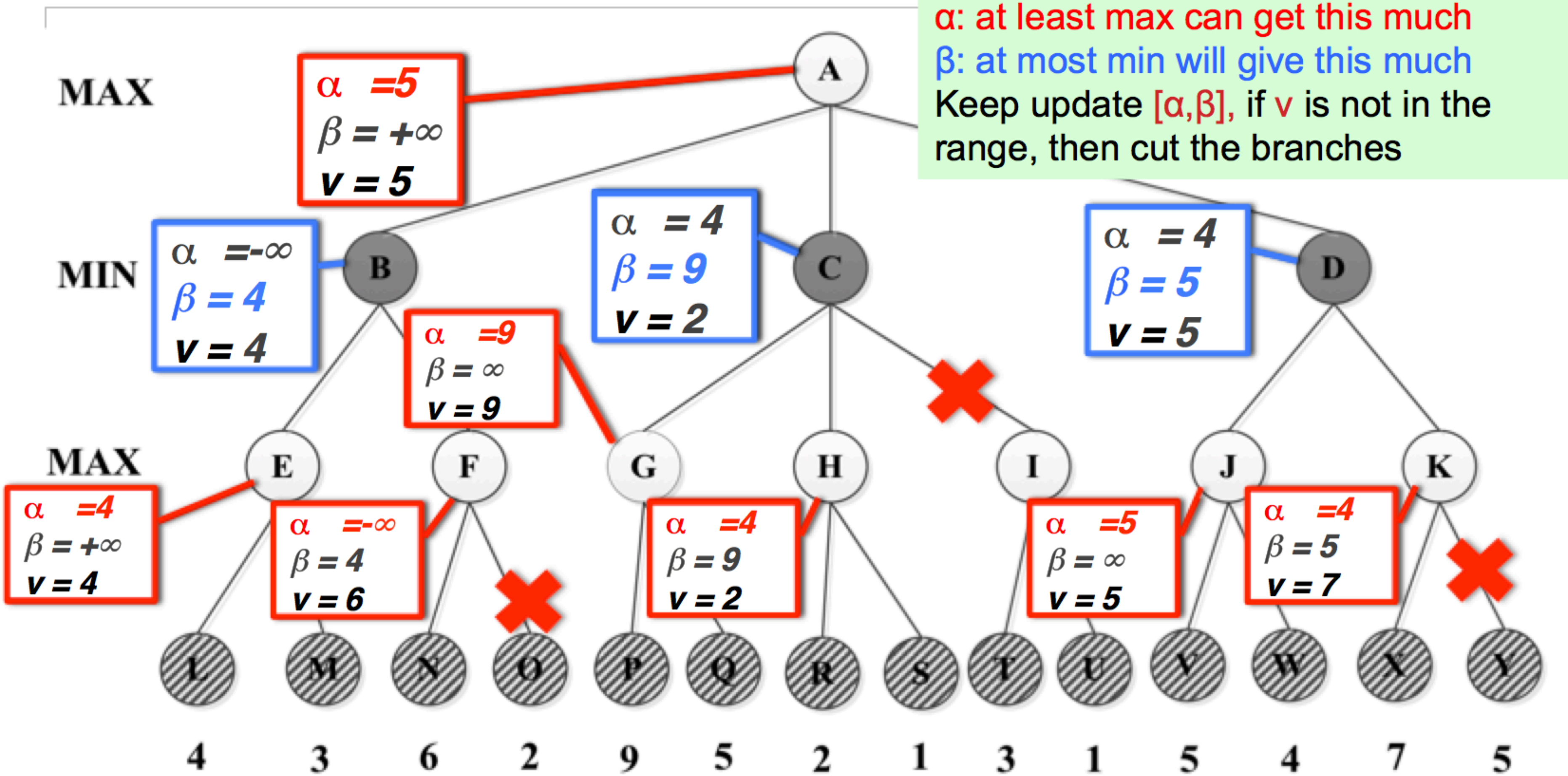
Optimal solution



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 not in the range, then 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 $\boxed{v \geq \beta}$ **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 $\boxed{v \leq \alpha}$ **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

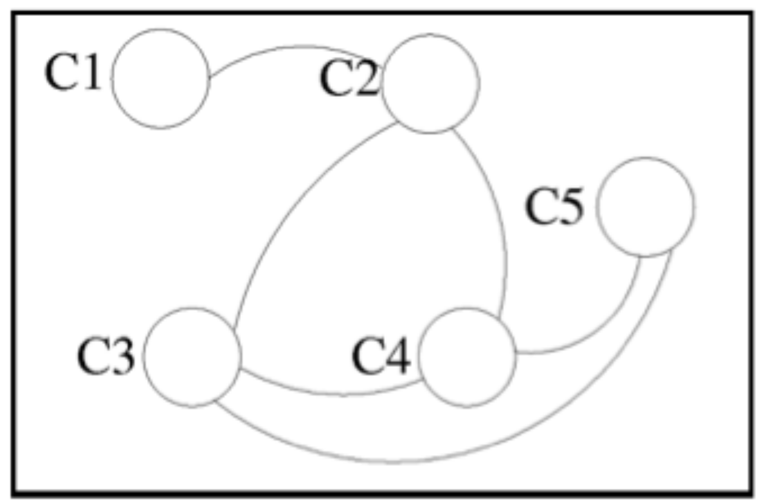
The profiles of your engineers are:

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

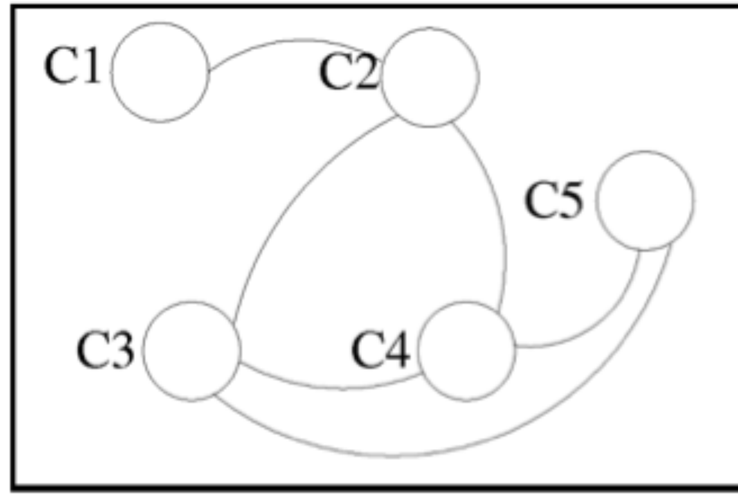
- Using Company as variable, formulate this problem as a CSP problem with variables, domains, and constraints. Constraints should be specified formally and precisely, but may be implicit rather than explicit.
- Draw the constraint graph associated with your CSP.

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

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



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



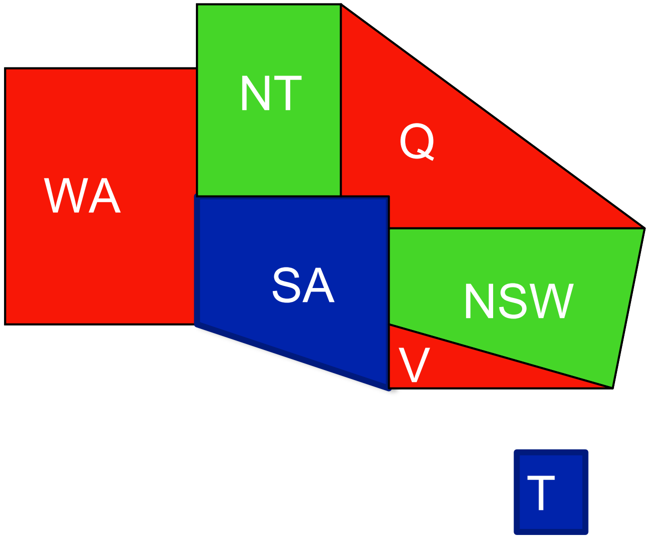
Constraints:
C1 ≠ C2, C2 ≠ C3, C3 ≠ C4, C4 ≠ C5, C2 ≠ C4, C3 ≠ C5.

- 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

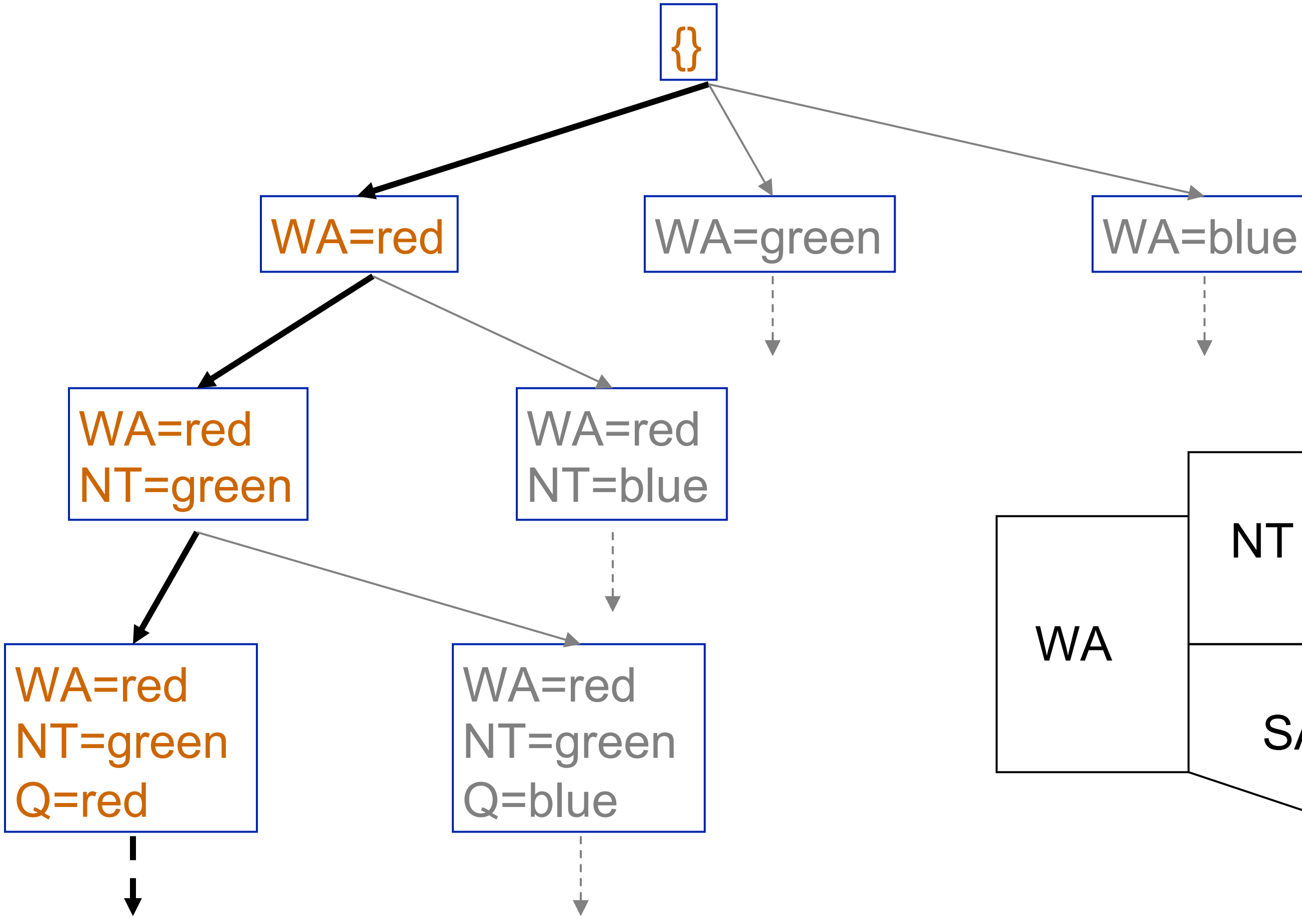
CSP Example: Map Coloring

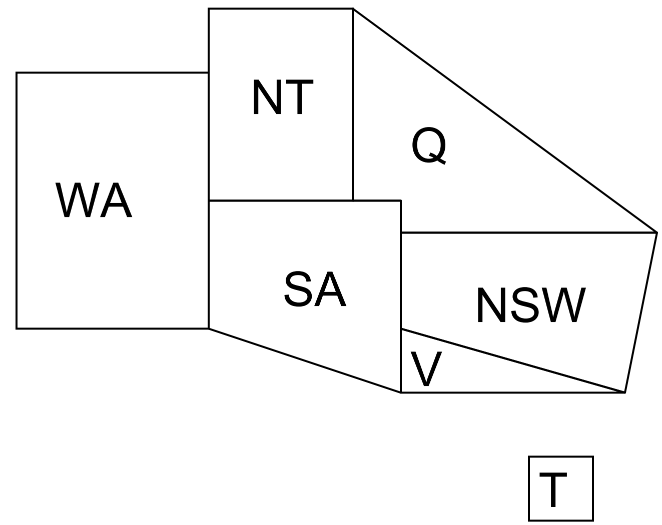


- 7 variables {WA,NT,SA,Q,NSW,V,T}
- Each variable has the same domain {red, green, blue}
- No two adjacent variables have the same value:

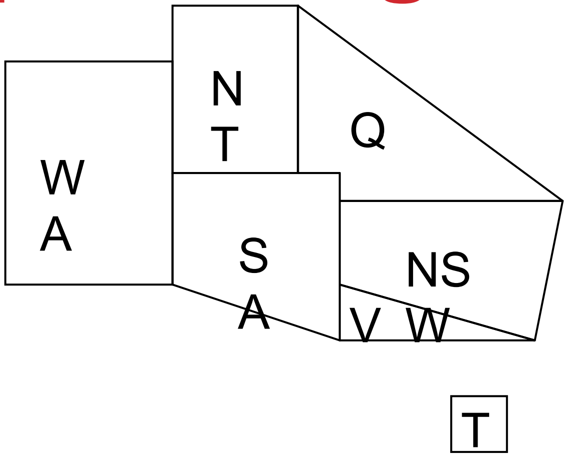
  WA≠NT, WA≠SA, NT≠SA, NT≠Q, SA≠Q, SA≠NSW, SA≠V,Q≠NSW, NSW≠V

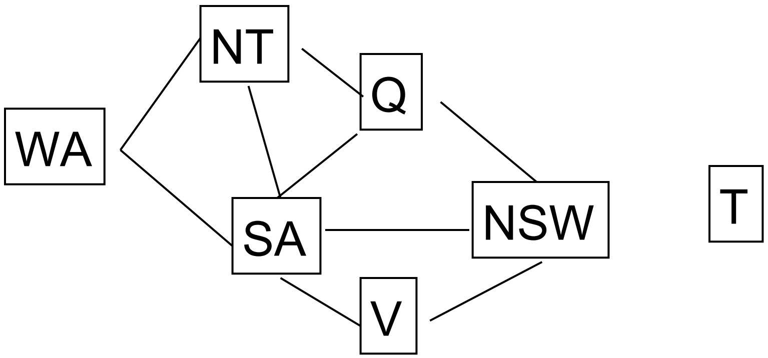
Backtracking Search: Map Coloring





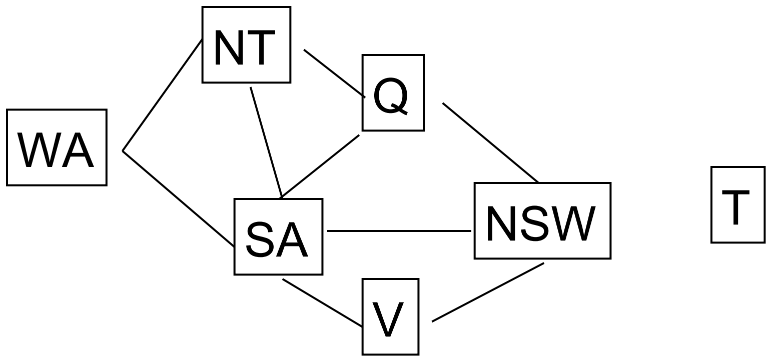
Map Coloring: Forward Checking





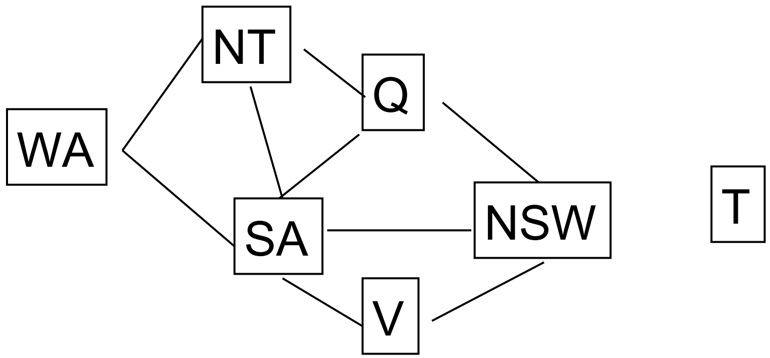
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
1: R	RGB	RGB	RGB	RGB	RGB	RGB

Map Coloring: Forward Checking



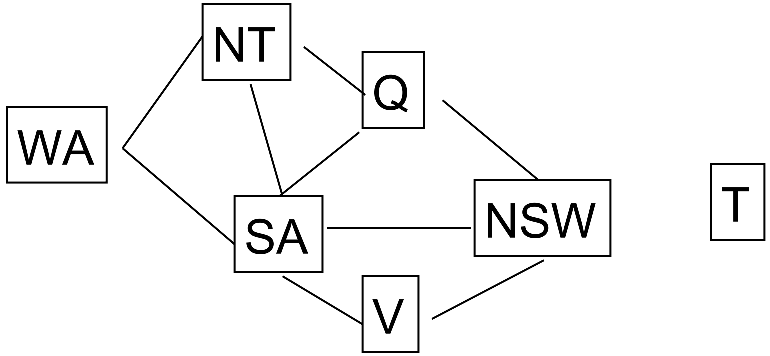
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
1: R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB

Map Coloring: FC



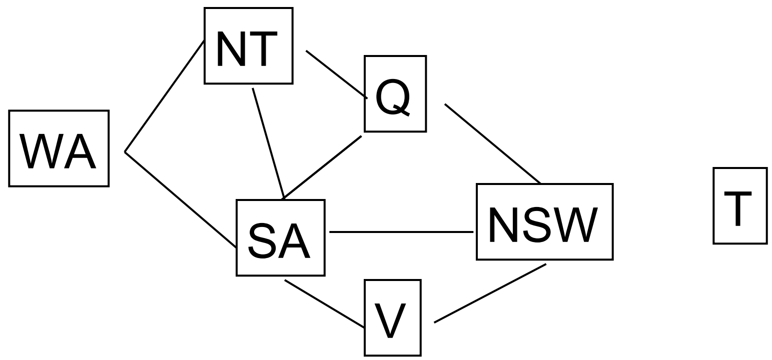
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~GB	2: G	RGB	RGB	~~R~~GB	RGB

Map Coloring: FC



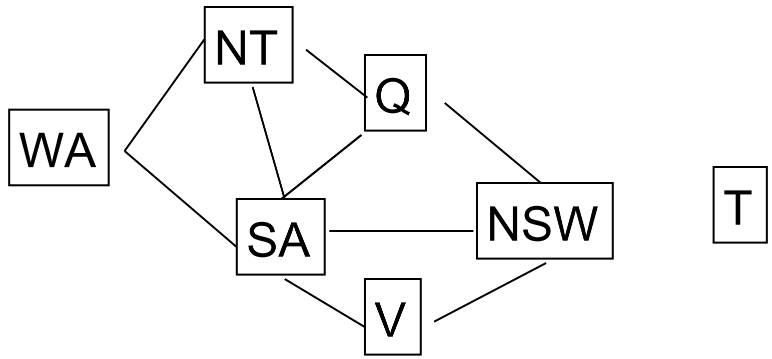
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~RG~~B	2: G	R~~G~~B	RGB	~~RG~~B	RGB

Map Coloring: FC



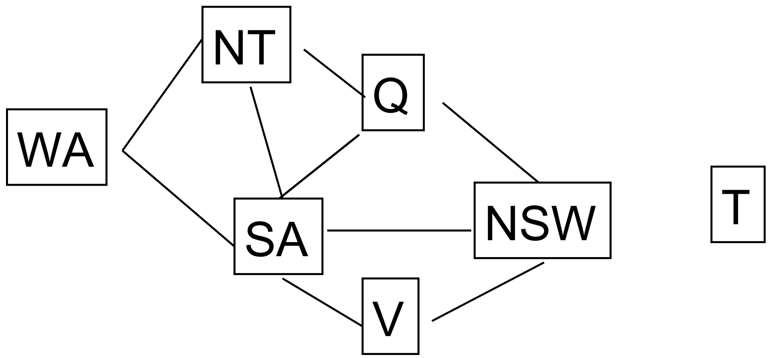
WA	NT	Q	NSW	V	SA	T
RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	~~R~~GB	RGB	RGB	RGB	~~R~~GB	RGB
R	~~R~~~~G~~B	G	R~~G~~B	RGB	~~R~~~~G~~B	RGB
R	~~R~~~~G~~B	G	R~~G~~B	3:B	~~R~~~~G~~B	RGB

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

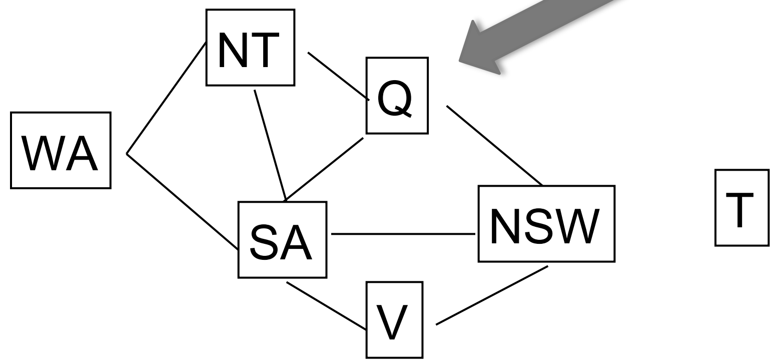
Other inconsistencies



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	~~RG~~B	G	R~~G~~B	RGB	~~RG~~B	RGB
R	~~RGB~~	G	R~~GB~~	3:B	~~RGB~~	RGB

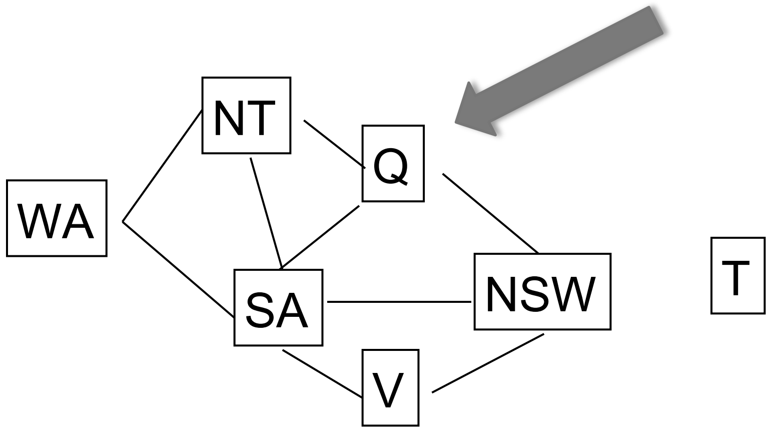
Map Coloring: Constraint Propagation



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

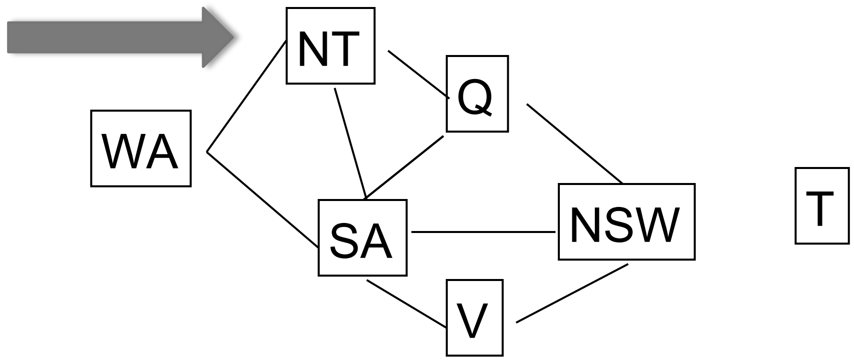
Map Coloring: CP



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

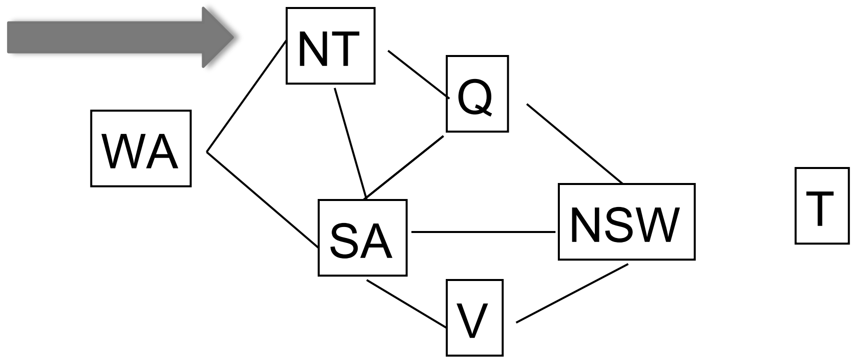
Map Coloring: CP



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