INF5390-2014 – Kunstig intelligens

Exercise 1 Solution

Roar Fjellheim

Exercise 1.1: Intelligent Agents (INF5390-02)

For every sentence below, state whether it is true or false, and support your reply with an example or counter-example:

a. An agent that only receives partial information on the environment cannot be rational.
   False. *Perfect rationality refers to the ability to make good decisions given the sensor information received.*

b. There exist environments where no pure reflex agent can be rational.
   True. *A pure reflex agent ignores previous percepts and cannot obtain an optimal state estimate in a partially observable environment.*

Exercise 1.1: Intelligent Agents (INF5390-02)

c. There is an environment where every agent is rational.
   True. *For example, in an environment with a single state, such that all actions have the same reward, it does not matter which action is taken.*

d. Input to the agent program is identical to input to the agent function.
   False. *The agent function, notionally speaking, takes as input the entire percept up to that point, while the agent program takes the current percept only.*

Exercise 1.1: Intelligent Agents (INF5390-02)

e. Assume that an agent selects actions at random. There exists an environment where this agent is rational.
   True*. This is a special case of c). If it does not matter which action is taken, selecting randomly is rational.*

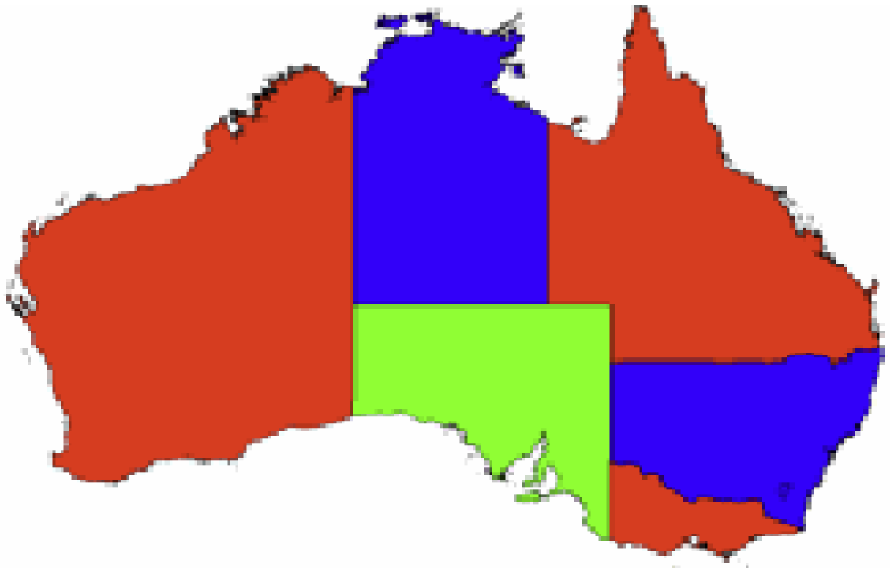
f. Every agent is rational in an unobservable environment.
   False*. Some actions are stupid (and the agent may know this if it has a model) even if it has no environment input.*

g. A perfectly rational poker-playing agent will never lose.
   False*. Unless it draws the perfect hand, the agent can lose if an opponent has better cards.*

Exercise 1.2: Solving Problems by Searching (INF5390-03)

The four colors problem can be defined as follows: With as few as possible and at most four colors*, color a map so that no neighboring regions have the same color.

The example shows 6 Australian (mainland) regions colored with 3 colors.



* That four colors are enough for any map was proven in 1976 as the first major theorem to be proved using a computer.

Exercise 1.2: Solving Problems by Searching (INF5390-03)

a. Give a precise specification of the task as a search problem.
b. Draw an in-principle diagram (not complete) of a search tree to find a solution.
c. Choose and justify an uninformed search algorithm for finding an optimal solution.
d. How would you characterize the efficiency of uninformed search to solve this problem?

Formulation of a problem

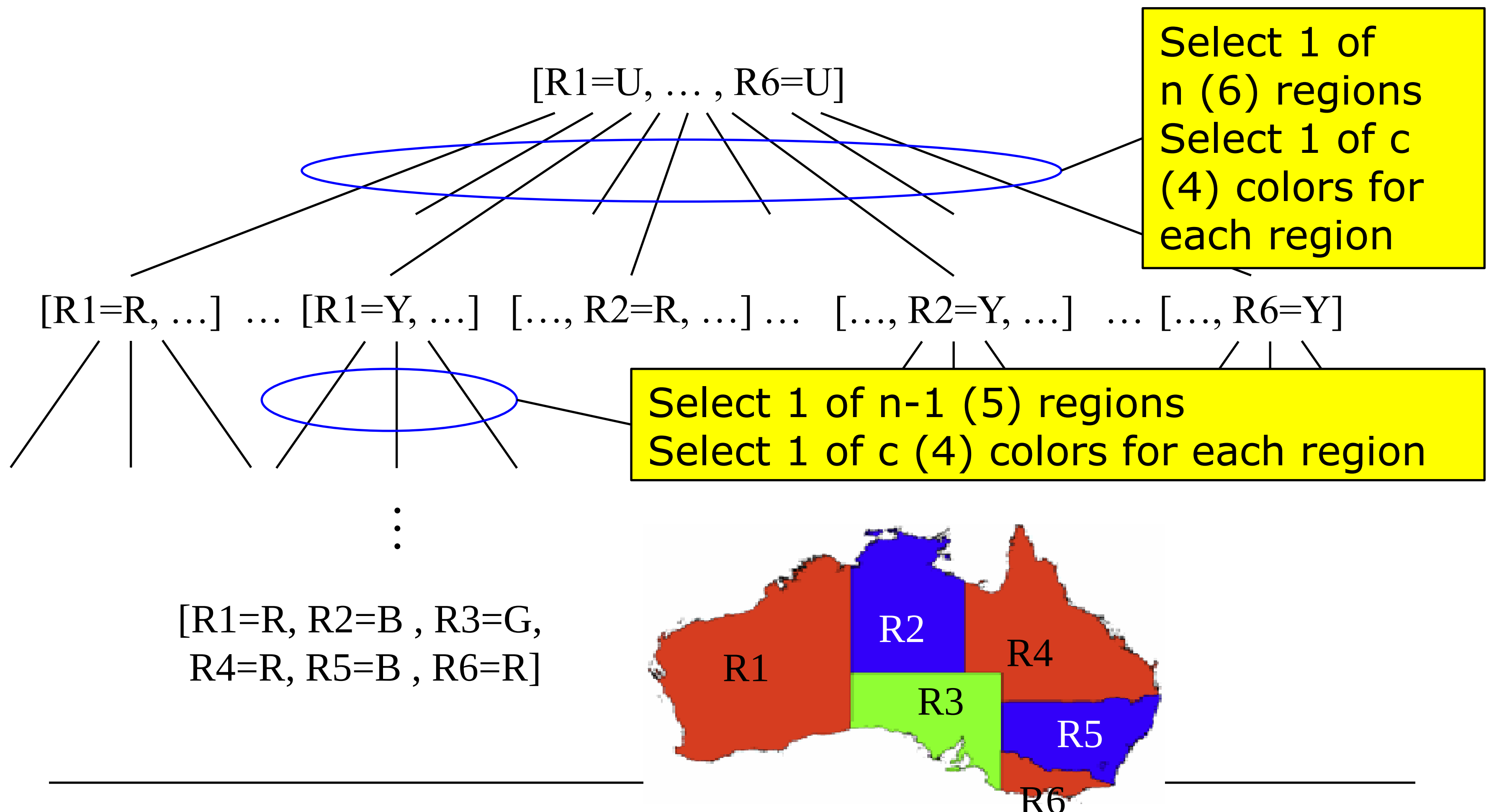
- Initial state
  - Initial state of environment
- Actions
  - Set of actions available to agent
- Path
  - Sequence of actions leading from one state to another
- Goal test
  - Test to check if a state is a goal state
- Path cost
  - Function that assigns cost to a path
- Solution
  - Path from initial state to a state that satisfies goal test

Defines the *state space*

States and actions

- Regions: R1, …, R6
- Colors: R (red), B (blue), G (green), Y (yellow),
  U (unassigned)
- State: [R1=c1, …, R6=c6] where ci is in Colors
- Initial state: [R1=U, … , R6=U]
- Actions: [… Rj=U …] ⇒ [… Rj=ci …] where ci ≠ U
- Goal test (target state): [R1=c1, …, R6=c6]
  where each ci ≠ U and
  no neighboring regions have the same color
- Cost function: +1 per assignment (but all goal paths have equal length, see later)

Uninformed search tree



Properties of the search tree

- Complexity
  $(n \times c) \times ((n-1) \times c) \times \ldots (1 \times c) = \mathbf{n! \times c^n}$
- But, there can only be c^n unique complete color assignments
- Many paths are equivalent (order of assignment is irrelevant)
- Many inconsistent partial assignments, cannot be corrected further down in tree
- All solutions at level n (here 6)
- Many consistent solutions (e.g. systematic exchange of colors)

n	c=4
1	4
2	32
3	384
4	6 144
5	122 880
6	2 949 120
7	82 575 360
8	2 642 411 520
9	95 126 814 720
10	3 805 072 588 800

Uninformed search algorithm

- Search path is limited to n = number of regions
  - √ Depth first search can be used
  - √ Could use breadth first, but high branching factor will lead to large memory requirement
- **Depth first search recommended**
  - √ Iterative deepening depth first could be considered, but in absence of checking for partial inconsistency, all goal paths will have length n

Recap: Complexity of depth-first search

- Depth-first has very low memory requirements, only needs to store one path from the root
- With branching factor *b* and depth *m*, space requirement is only *bm*.
  - √ For b=10, 100 bytes/node problem, memory increases from 100 bytes at depth 0 to 12 Kilobytes at depth 12
- Worst case time complexity is $O(b^m)$, but depth-first may find solution much quicker if there are many solutions (*m* may be much larger than *d* – the depth of the shallowest solution)

Properties and efficiency of selected algorithm

- **Complete**: If there as a solution, it will be found (finite size of tree and exhaustive search)
- **Optimal**: Any consistent solution found on level n is as good as any other, including the first found

- **Memory**: Low requirements
- **Time**: Goes as search tree complexity: $n! \times c^n$
  - Search generates many equivalent subtrees
  - Search generates many inconsistent subtrees
  - Unfeasible for large n

- Points to inadequacy of uninformed search for realistic problems

Constraint Satisfaction Problems (CSP)

- Represents states as variable=value pairs and conditions for solutions as **variable constraints**
- Starts out as classical search, but **propagates** constraints to eliminate entire subspaces
- Builds solution **incrementally** - Think of solving Sudoku, crosswords, etc.
- Many specialized techniques make CSP an efficient method for large **combinatorial** problems
- CSP solvers are widely applied to domains like scheduling, planning, configuring, timetabling, …
- For more on CSP: See AIMA Chapter 6