# CS 480 NoteSheet

## Chapter 2

### PEAS

* The *Performance* measure
* The *Environment* in which the agent will operate
* The *Actuators* that the agent will use to affect the environment
* The *Sensors* that the agent will use to perceive the environment

### Env prop

* Fully vs partially observable (can be unobservable too)
* Single agent vs multiagent
* multiagent: competitive vs. cooperative
* Deterministic vs. nonderministic (stochastic)
  + nonderministic: next state is NOT completely determined by the current state and agent action
* Episodic vs. sequential
  + sequential: current decision / action COULD affect all future decisions / actions
* Static vs. dynamic
  + Static: environment CANNOT change while the agent is taking its time to decide
* Discrete vs. continuous
  + continuous: time changes are continuous
* Known vs. unknown
  + known: agent knows all outcomes to its actions
  + unknown: learning and exploration can be necessary

### State representations

* Atomic
  + state representation has NO internal structure
* Factored
  + state representation includes fixed attributes (which can have values)
* Structured
  + state representation includes objects and their relationships

### Typical agent arch

* Simple reflex agent
  + uses condition-action rules
* Model-based reflex agent
  + keeps track of the unobserved parts of the environment by maintaing internal state:
  + “how the world works”: state transition model
  + how percepts and environment is related: sensor model
* Goal-based reflex agent
  + maintains the model of the world and goals to select decisions (that lead to goal)
* Utility-based reflex agent
  + maintains the model of the world and utility function to select PREFERRED decisions (that lead to the best expected utility: avg (EU \* p))

### Search problem: Dracula’s Roadtrip

State Space: a map of Romania  
Initial State: Arad  
Goal State: Bucharest  
Actions: ACTIONS(Arad) = {ToSibiu,ToTimisoara,ToZerind}  
Transition Model: RESULT(Arad, ToZerind) = Zerind  
Action Cost Function [ActionCost(Scurrent, a, Snext)]: ActionCost(Arad, ToSibiu, Sibiu) = 140

## Chapter 3

### Search perf

* Completeness: Is the algorithm guaranteed to find a solution when there is one, and to correctly report failure when there is not?
* Cost optimality: Does it find a solution with the lowest path cost of all solutions?

### Informed Search and Heuristics

Informed search relies on domain-specific knowledge / hints that help locate the goal state

An *admissible heuristics* is guaranteed to give you the optimal solution

Every *consistent heuristics*, heuristics that only makes the estimate better, is admissible heuristics, but not the other way around

### Greedy

Single heuristic (eg. distance to goal)

### A\*

heuristic and total path cost (eg. dist. to goal + path cost from initial node)

## Chapter 5

### Min-Max

I don’t know what move my opponent will choose, but I am going to ASSUME that it is going to be the best / optimal option

* At every leaf node the MinMax value (utility at leaf node) is calculated,
* For every MAX Player node, the current LARGEST child MinMax value is saved in
* For every MIN Player node, the current SMALLEST child MinMax value is saved in
* If at a MIN Player node m the current value , then the search at node m can end. Here is the LARGEST value of a MAX Player node in the path from the root to node m,
* If at a MAX Player node n the current value , then the search at node n can end. Here is the SMALLEST value of a MIN Player node in the path from the root to node n.

## Chapter 6

### Constraint Satisfaction Problem (CSP)

* a set of variables
* a set of domains
* a set of constraints C that specify allowable combinations of value
* If NO constraints violated: consistent assignment
* If ALL variables have a value: complete assignment
* If SOME variables have NO value: partial assignment
* SOLUTION: consistent and complete assignment
* PARTIAL SOLUTION: consistent and partial assignment

#### Variable Types

* Domains can be
  + finite, for example: {1, 2, 3, 5, 8, 20} (simpler)
  + infinite, for example: a set of all integers
* Variables can be:
  + discrete, for example: X = {X1, …, Xn} (simpler)
  + continuous, for example: R+
* Constraints can be:
  + unary (involve single variable), for example: X1 = 5
  + binary (involve two variables), for example: X1 = X2
  + higher order (involve > 2 variables), for example: X1 = X2 \* X3

#### local consistency

Remove inconsistent values from variable domains as we go as they would make certain assignments inconsistent later anyway

* Node consistency
  + a single variable is node-consistent (in a constraint graph) if all the values in its domain satisfy variable unary constraints
* Arc consistency
  + a single variable is arc-consistent (in a constraint graph) if all the values in its domains satisfy ALL its binary constraints
* Path consistency
  + two variable set {Xi, Xj} is path-consistent (in a constraint graph) with respect to a third variable Xm if for EVERY assignment {Xi = a, Xj = b} there is an assignment to Xm (between Xi and Xj) that satisfies constraints on {Xi, Xm} and {Xm, Xj}.

## Chapter 7

### Logical Entailment

A set of sentences (called premises) logically entails a sentence (called a conclusion) if and only if every truth assignment that satisfies the premises also satisfies the conclusion

PREMISES CONCLUSION

### Conjunctive Normal Form CNF

A sentence is in CNF if and only if consists of conjunction: of clauses.  
A clause Ki consists of a disjunction of literals

eg.