CS 6364 QUIZ 1

Rohith Peddi, RXP190007

September 3, 2020

1 KITTY WORLD

Read section 2.4.7 from the Textbook and represent the kitty world as:

1. Atomic representation.
2. Factored representation.
3. Structured representation

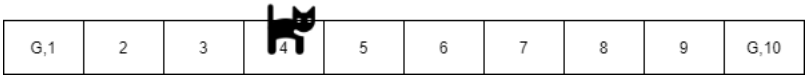
1.1 Description

Describe each of the representations (5 points for each representation)

1.1.1 Atomic representation

1. Here each state of the world is indivisible and has no internal structure associated.
2. The algorithms underlying search and game-playing work with atomic representations.

Let us associate a search problem to the kitty world with multiple goal states.

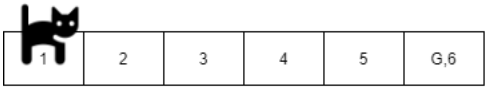


State space	Possible Percepts	Possible Actions
Positions(10)	Clap, Pet, Bump	Walk forward, Walk backward, Purr, Meow, Blink

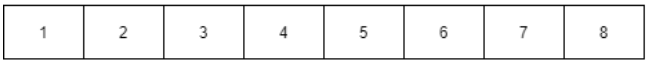
Here we consider two goal states and each position can be represented as a node. As kitty moves in a **partially observable environment** it tries to build a model of the environment it traverses. The problem is devised with two goal states to make it apt for a Utility based agent. Here **utility function of a state decreases with increase in distance from goal**.

1.1.2 Factored representation

1. Splits up each state into a fixed set of variables or attributes each of which can have a value
2. Areas where factored representation is used Constraint Satisfaction Problems, Planning, Bayesian Networks



POSITION



BATTERY

ACTIONS

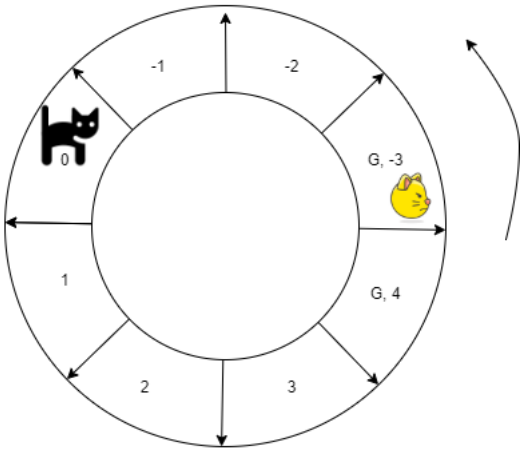
State space	Possible Percepts	Possible Actions
Positions(6), Battery(8), Actions(10)	Clap, Pet, Bump	Walk, Purr, Meow

So each state is made of three variables : Position(P), Battery(B), Actions(A) = S(P, B, A) $\implies$ *total* :(6)(8)(10) = 480

Action	Variables effected
Walk	Position, Battery, Actions
Meow	Actions, Battery
Purr	Actions

1.1.3 Structured representation

1. Objects and their varying relationships are described explicitly
2. Structured representation underlie relational databases, first order logic and much of natural language processing



State space	Possible Percepts	Possible Actions
Objects(robot kitty, random kitty, $P_{(-3\|-2\|-1\|0\|1\|2\|3\|4)}$)	Clap, Pet, Bump	Walk forward Walk backward, Purr, Meow

Relations :

1. In(*Robot*, P_x)
2. In(Robot Kitty, P_x) and In(Random Kitty, P_y) $\implies$ ($x \neq$ y)

Object Attributes :

1. Robot Kitty(Number of times bumped, Direction)
2. Random Kitty(Direction)

Environment Properties :

1. Robot Kitty receives percept Pet when it reaches Goal state (-3 or 4)
2. Whenever Robot Kitty and Random Kitty shall reach same position then environment sends out percept bump
3. In every other position environment sends out percept clap
4. Random Kitty can move in any of the two directions with equal probability

1.2 Pseudo code

Write the pseudo-code of a utility-based agent for each of these representations (5 points for each representation)

1. For fully observable and fully known environments goal and utility based agents compute a policy which is a mapping between states and actions

2. For partially observable environments agent can compute a conditional plan that specifies sequence of actions as a function of agent's perception.

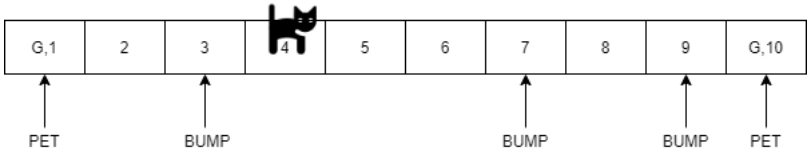
GENERIC PSEUDO CODE FOR UTILITY BASED AGENTS

```
function UTILITY–BASED–AGENT(percept) returns an action
    persistent: state, the agent's current conception of the world state
                model, describes how the next state depends on current state
                utility−function, describes agent's utility function
                plan, a sequence of actions to take, empty initially
                action, most recent action, none initially

    state = UPDATE–STATE(state, action, percept, model)
    if plan is empty then
        plan = PLAN(state, utility−function, model)
    action = FIRST(plan)
    plan = REST(plan)
return action
```

Update functions for each representation are written below. Plan consisting of sequence of actions generated depends on state(affected by percepts), utility function, model(gives information about actions and corresponding change in state variables)

1.2.1 Atomic Representation



If above picture describes the percepts sent by the environment when kitty reaches that position. Kitty being a Utility based agent traverses in the following manner.

Assuming Utility function = -$min(|P-1|, |P-10|)$

```
State = UPDATE-STATE(state, action, percept, model)
Position - 1 = UPDATE-STATE(Position, Walk backward)
Position + 1 = UPDATE-STATE(Position, Walk forward)
Position = UPDATE-STATE(Position, Meow | Purr)
```

As plan is a function of current state, utility function and model (Includes information about different possible actions).

U(state) just depends on the position. Can be calculated by taking just the value of position into consideration.

```
Step - 1: Action generated = Walk backward, U(4) = -3
Step - 2: Action generated = Meow, Walk backward, U(3) = -2
Step - 3: Action generated = Walk backward, U(2) = -1
Step - 4: Action generated = Purr, U(1) = 0
```

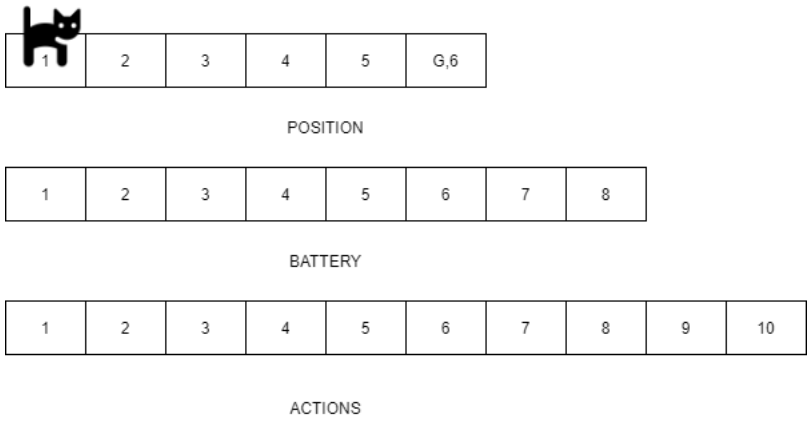
Utility Maximised

Possible actions in Plan computed based on utility function maximization (Assuming kitty performs action Meow when bumped)

```
if percept == clap :
    if |P-1|<|P-10|:
        Walk backward
    else:
        Walk forward
if percept = pet :
    Purr
if percept = bump :
    Meow, Walk backward
```

1.2.2 Factored Representation

Considering the state space of the kitty world in the below picture



Following can be the state update parameters for each percept and action

```
State = UPDATE-STATE(state, action, percept, model)
Position + 1 = UPDATE-STATE(Position, Walk)
```

```
Battery - 1 = UPDATE-STATE(Battery, Walk | Meow)
Actions + 1 = UPDATE-STATE(Actions, Walk | Meow | Purr)
Battery - 1 = UPDATE-STATE(Battery, Bump)
Battery + 1 = UPDATE-STATE(Battery, Clap | Pet)
```

Utility is maximised when kitty reaches goal state within less number of actions and More battery life. A parallel to self driving cars that have goals of faster and safer way of reaching destination.

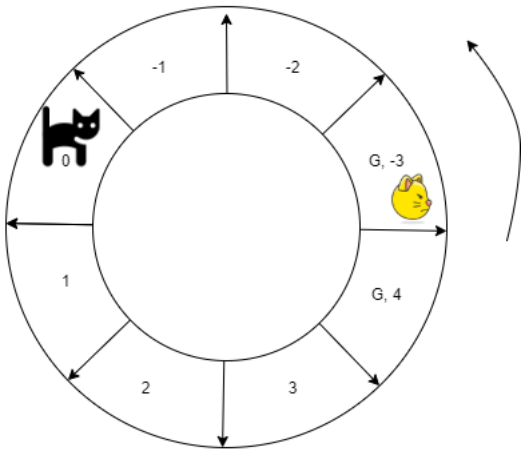
A possible utility function : U(P, B, A) = $-|P-6|+Battery+(10-Actions)$

```
plan = PLAN(state, utility-function, model)
```

Sequence of actions generated depends on the percept sequence and above utility function.

1.2.3 Structured Representation

Considering a parallel situation of not being able to use a factored representation of TruckAheadBackingIntoDairyFarmDrivewayBlockedByLooseCow mentioned in the textbook.



Following can be state update parameters for each percept and action

```
State = UPDATE-STATE(state, action, percept, model)
In(Object, Position + 1) = UPDATE-STATE(In(Object, Position), Walk forward)
In(Object, Position - 1) = UPDATE-STATE(In(Object, Position), Walk backward)
-(direction) = UPDATE-STATE(Object, Bump)
(Number of times bumped + 1) = UPDATE-STATE(Object, Bump)
In(Object, Position - 1(direction)) = UPDATE-STATE(In(Object, Position), bump)
In(Object, Position) = UPDATE-STATE(In(Object, Position), Purr)
```

Utility is maximised when kitty reaches the goal state i,e In(Kitty, $P_{(-3,4)}$)

Mapping each tuple in the relation to a real number on scale based on distance from goal gives us following:

Udist(In(Kitty Robot, P_x)) = $-min(|x+3|, |x-4|)$
Utility = -(Number of times bumped) + Udist(x)

Possible actions computed using above utility function in generation of plan:

```
if percept == clap :
    if number of times bumped < 3 :
        if direction = 1:
            Walk forward
        else:
            Walk backward
    else
        direction = -(direction)
        if direction = 1:
            Walk forward
        else:
            Walk backward
if percept == bump:
    Meow
if percept == pet:
    Purr
```

Assuming reaching environment generated percept Pet in goal state.