ADL SOLUTION FOR FOX, GOOSE AND BAG OF BEANS PROBLEM

Domain:

(define (domain river-domain)

(:requirements :adl :typing)

(:predicates (onLeft ?x))

(:action crossRiver

:parameters (?x)

:precondition

(and

(or

(and

(onLeft ?x)

(onLeft farmer)

)

(and

(not(onLeft ?x))

(not(onLeft farmer))

```
)
)
(or
( = ?x goose)
(and
(= ?x fox)
(or
(and
(onLeft goose)
(not(onLeft beans))
)
(and
(not(onLeft goose))
(onLeft beans)
)
)
)
(and
(= ?x beans)
(or
```

```
(and
(onLeft fox)
(not(onLeft goose))
)
(and
(not(onLeft fox))
(onLeft goose)
)
)
)
(and
(= ?x farmer)
(or
(and
(onLeft goose)
(and
(not(onLeft fox))
(not(onLeft beans))
)
)
```

```
(and
(not(onLeft goose))
(and
(onLeft fox)
(onLeft beans)
)
)
)
)
)
)
:effect
(and
(when (onLeft ?x)
(and
(not (onLeft ?x))
(not (onLeft farmer)))
)
(when (not (onLeft ?x))
(and
```

```
(onLeft ?x)
(onLeft farmer))
)
)
)
)
```

Problem

```
(define (problem river)
(:domain river-domain)
(:objects
fox goose beans farmer)
(:init
(onLeft fox)
(onLeft goose)
(onLeft beans)
(onLeft farmer)
)
(:goal
(and
```

```
(not(onLeft fox))

(not(onLeft goose))

(not(onLeft beans))

(not(onLeft farmer))

)

)

)
```

Input

```
./ff -o domain.pddl -f problem.pddl
```

Output

```
Cueing down from goal distance:    3 into depth [1]

Enforced Hill-climbing failed !
switching to Best-first Search now.

advancing to distance :    3
                           2
                           0

ff: found legal plan as follows

step    0: CROSSRIVER GOOSE
        1: CROSSRIVER FARMER
        2: CROSSRIVER FOX
        3: CROSSRIVER GOOSE
        4: CROSSRIVER BEANS
        5: CROSSRIVER FARMER
        6: CROSSRIVER GOOSE


time spent:    0.00 seconds instantiating 0 easy, 14 hard action templates
               0.00 seconds reachability analysis, yielding 8 facts and 14 actions
               0.00 seconds creating final representation with 8 relevant facts
               0.00 seconds building connectivity graph
               0.00 seconds searching, evaluating 12 states, to a max depth of 1
               0.00 seconds total time
```

Part 2

A

Motion planning is, when given a movable object and a description of the environment, finding a sequence of valid configurations that moves it from the start to the goal. It allows intelligent robots to arrange their motions to accomplish a desired task (reach the destination, clean the table etc.)

B

Action Planning:

Action planning is a process that can decide which steps to take in order to achieve some particular goal using logic rules

Since the verbal definition of Motion planning is given in the part A, below is another definition of Motion Planning

Motion Planning:

Input:

Model of the robot, world & how they change

- mathematical euquations
- 3D Models
- Semantic knowledge
- Uncertainty models

Performance Objective:

- Avoid collisions
- Respect physical limits

- Minimize time
- Look human

Output:

High-Quality sequence of actions

- Continuous path
- Feedback control policy
- High-level task decompostion

Difference between Motion Planning and Action planning:

Action Planning is used to formulate strategies to achieve a certain goal. The output is a sequence of actions such that when executed in order, will lead to the desired goal.

Motion planning concerns only one particular aspect of planning which is movement. This particular problem is usually solved with a Path Finding algorithm such as A.

C)

1) Autonomous vehicles (avoiding collision in traffic jam)

Should avoid obstacles such as other moving cars and objects when also minimizing the time spend in the traffic jam

2) Surgeries (Prostate Brachytherapy)

Apply the necessary steps for the surgery in order without harming any other tissue (avoiding obstacles, finding a valid path)

3) Climbing robots

Inspect the environment options so that make climbing possible

Bonus: Protein Folding (deformation and field folding)

D

Supposing there are islands in the river, we can solve the same problem with the motion planning via:

1. Define the states as nodes in a graph.
2. The states we don't want to happen (for example, fox eating goose) can be implemented as an obstacle along the path.
3. So the motion planning will try to reach the goal node by avoiding obstacles, thus, nothing ate nothing in the end.