Artificial Intelligence I

Lab 2 - Winter Semester 2023 / 2024
https://moodle.haw-landshut.de/course/view.php?id=10282

1. For each of the following activities, give a PEAS description of the task environment and characterize it in terms of the properties of task environments (observable? deterministic? episodic? static? discrete? agents?)

- Playing soccer.
- Shopping for used AI books on the Internet.
- Knitting a sweater.
- Bidding on an item at an auction.

2. Give a complete problem formulation for each of the following problems. Choose a formulation that is precise enough to be implemented.

(a) There are six glass boxes in a row, each with a lock. Each of the first five boxes holds a key unlocking the next box in line; the last box holds a banana. You have the key to the first box, and you want the banana.

(b) You start with the sequence $ABABAECCEC$, or in general any sequence made from A, B, C, and E. You can transform this sequence using the following equalities: $AC = E$, $AB = BC$, $BB = E$, and $Ex = x$ for any x. For example, $ABBC$ can be transformed into AEC (using $BB = E$), and then AC (using $Ex = x$), and then E (using $AC = E$). Your goal is to produce the sequence E.

(c) There is an $n \times n$ grid of squares, each square initially being either unpainted floor or a bottomless pit. You start standing on an unpainted floor square, and can either paint the square under you or move onto an adjacent unpainted floor square. You want the whole floor painted.

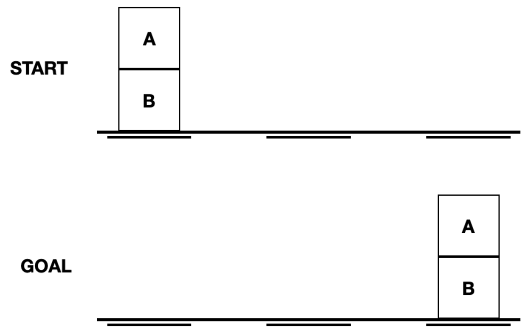
(d) A container ship is in port, loaded high with containers. There 13 rows of containers, each 13 containers wide and 5 containers tall. You control a crane that can move to any location above the ship, pick up the container under it, and move it onto the dock. You want the ship unloaded.

(e) Suppose two friends live in different cities on a map, such as the Romania map from the lecture. On every turn, we can simultaneously move each friend to a neighboring city on the map. The amount of time needed to move from city i to city j is equal to the road distance $d(i, j)$ between the cities, but on each turn the friend that arrives first must wait until the other one arrives (and calls the first on his/her cell phone) before the next turn can begin. We want the two friends to meet as quickly as possible.

3. Containers are restacked in a container warehouse. Three bays are available for this purpose. Containers A and B are to be transported from the left to the right bin, whereby the middle bin can be used. The containers can only be transported one at a time. With breadth-first search for the container problem, find a path from the starting node to the goal node.

In What order are my actions/nodes added to the frontier/fringe

A2, A3



Does it make sense to use depth-first search on the container-restacking-problem? How would you proceed if not only two containers are stacked on top of each other, but 10 or more?

1, a) Playing soccer

Performance: Goals, Players on field, Balltime
Enviroment: Gamefield, Players, Goal placement, Field Lines
Actuators: Run, Shoot, Pass, Throw
Sensors: Seeing, Speed, Location on the field, distance to goal, position of other players (my team / other team)

Observable: yes
Deterministic: no
Episodic: no
Static: no
Discrete: ?
Agents: multiagents

---

b) Knitting a Sweater.

P: Rows knitted, Parts knitted, Speed per knit?
E: Wool, knitting needles
A: Knitting Patterns, loops, movement, refill, join parts
S: Knits made count, Visuals

Observable: yes
Deterministic: yes
Episodic: yes
Static: yes
Discrete: yes
Agents: single

---

c) Shoping for used AI Books on the Internet

P: Numbers of bought books, price per book, time used, differance to original price.
E: WWW, difrent webpages, API's, Protocolls
A: Buy, Switch page, Login, Search, look into HTML, Query, regex search, code, ...
S: Screen image, HTTP responses, Network requests, email history/access,

observable: yes
Deterministic: no
Episodic: yes/no depends
Dynamic: yes
Discrete: continuous
Agents: Multiagents

---

d) Bidding on an Item on an auction.

P: Biddprice, Realprice differance, Won bidds, resellprice differance
E: Bidding place (webpage like ebay or reallife auction), Other people, auctioneer,
A: Bid, Observe
S: Visuals, Audio,

Observability: fully observable
Deterministic: stochastic
Episodic: yes
Static: Dynamic
Discrete: yes
Agents: multiagents,

2)

a) 1. check if you have the banana
2. open next box
3. take whats in the box

b)
```
ABAB AECCEC ⇒E      | AC → E  |
‾ABBC      CC       | BB → E  |
A   CCC             | Ex → x  |
      CC            | AB → BC |
‾
ABAB ABBC CEC
ABBC ABBC CEC
AEC  AEC  CEC
 E    AEBBCEC
 E    ABBC EC
 E    AEC EC
      ACC
       EC
        C
```

For this problem I would use a uninformed search algorithm.
Breadth-first-Search seams reasonable for me because there are not many results and It would be easy to run into infinite loops.

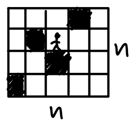
For this I define the problem as:
initial state: ABAB AECCEC
actions: can be used on every individual character pair.

```
AC → E
 E → AC
BB → E
 E → BB
AB → BC
 Ex → x
```

goal-test: is state "E"

A problem her is, that it is possible to find no solution.

c)



1. Paint the tile you stand on
2. Search for adjacent unpainted tiles and remember them.
3. If ther are none go to the last remembered free tile
4. If there are none remembered left, you are done.
5. Go to the next free tile from top in a couterclockwise rotation.
6. Repeat.

d)
```
While there are still containes on the boat do:
    If there is no more containes in the row move to the start of the next.
    If ther is no more row move back to the first one
    Pick the container
    Move to the dock
    Bring the crane to the next container in the row.
```

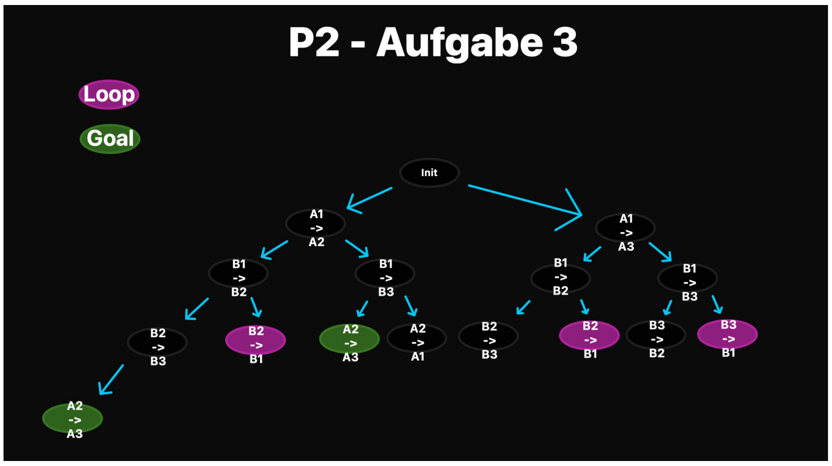
e) For every person, add all nearby citys to a LiFo que and sort the new added entries by:

$$\text{delta\_pathcost} = \text{abs}(p_1(i,j) - p_2(i,j))$$

$$\text{normalized\_delta\_pathcost} = \underline{\underline{\text{clamp}(\text{delta\_pathcost}, 0, 100, 0, 1)}}$$

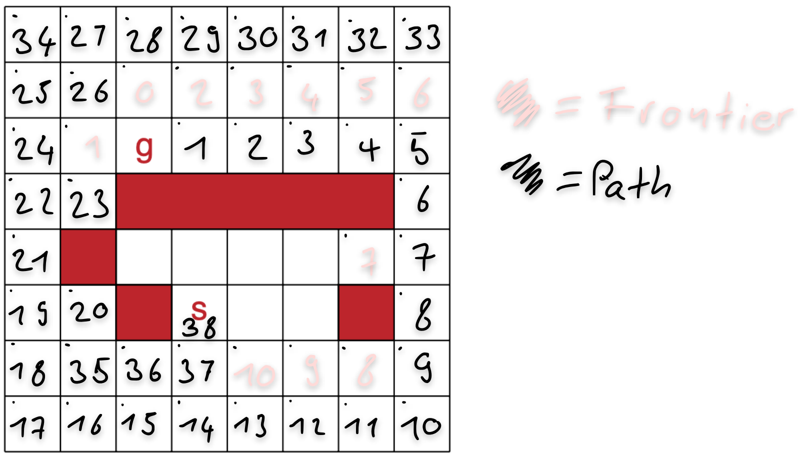
now we can take the best-first-search-algorithm and sort our prio-que by the normalized-delta-pathcost
The goal-test checks if both locations are the same.

3)



In this example a depth-first-search would be a good solution, because it would only take 4 steps to get to the goal or 2k steps where k is the number of stacked containers.
But we have to consider, that this is not the optimal case.

4. Consider the problem of finding a path in the grid shown below from the position s to the position g. The robot can move on the grid horizontally and vertically, one square at a time (each step has a cost of one). No step may be made into a forbidden red area.



On the grid, number the nodes in the order in which they are removed from the frontier in a depth-first search from s to g, given that the order of the actions you will test is: **up, left, right**, then **down**. Assume there is a cycle check.

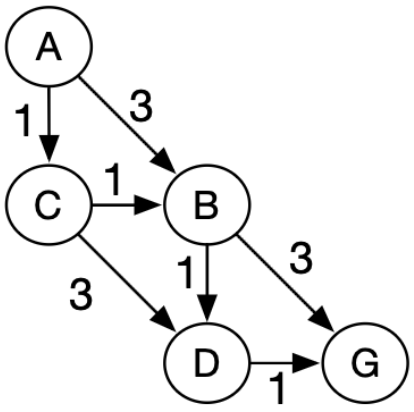
5. Consider the graph below where the problem is to find a path from start A to goal G. For each of the following algorithms, show the sequence of frontiers and give the path found.

(a) Depth-first search, where the neighbors are expanded in alphabetic ordering.

(b) Breadth-first search.

a)
Path-Taken: A-C-D-G
Frontier: B, G

b)
Path Taken: A-B-G
Frontier: C, D, G


A
B
C
D
G
1
3
1
1
3
3
1