Faculty of Informatics
Cognitive Systems Exercises
Summer Semester 2020

Ge Li (ge.li@kit.edu)
Maximilian Hüttenrauch (m.huettenrauch@kit.edu)
Felix Schneider (felix.schneider@kit.edu)
Christian Schmidt (christian.schmidt@kit.edu)

Homework 4

Language Models, Search, and Dynamic Programming

Hand in online in ILIAS by 1 July 2020, 17:00
Each homework is worth 100 points. In the exam, each full 100 points across all homeworks is worth 1 point on the exam (max 6 of 60).

Exercise 1. Language Models (15 Points)

Consider a 3-gram language model with the vocabulary $V = \{\text{Karlsruhe}, \text{Institute}, \text{of}, \text{Technology}\}$. The probabilities are given as follows:

w_{i-2}	w_{i-1}	$w_i =$ Karlsruhe	Institute	of	Technology	</s>
n/a	<s>	0.5	0.25	0.1	0.1	0.05
<s>	Karlsruhe	0	0.8	0.05	0.05	0.1
<s>	Institute	1	0	0	0	0
<s>	of	0	0.9	0	0	0.1
<s>	Technology	0	0.7	0.2	0	0.1
Karlsruhe	Karlsruhe	0.2	0.2	0.2	0.2	0.2
Karlsruhe	Institute	0	0	0.5	0.5	0
Karlsruhe	of	0	0.6	0.2	0.1	0.1
Karlsruhe	Technology	0	0.4	0.4	0.1	0.1
Institute	Karlsruhe	0	0	0.4	0.4	0.2
Institute	Institute	0.2	0.2	0.2	0.2	0.2
Institute	of	0.1	0.1	0.1	0.1	0.6
Institute	Technology	0.05	0.05	0.6	0.1	0.2
of	Karlsruhe	0.1	0.6	0.1	0.1	0.1
of	Institute	0.3	0	0.1	0.2	0.4
of	of	0	0	0	0	0
of	Technology	0.2	0.2	0.2	0.2	0.2
Technology	Karlsruhe	0	0.4	0.1	0.1	0.4
Technology	Institute	0.4	0	0	0	0.6
Technology	of	0.1	0.4	0	0	0.5
Technology	Technology	0.2	0.2	0.2	0.2	0.2

Here, <s> represents the beginning of sentence symbol and </s> the end of sentence symbol. Calculate the probabilities for the following sentences under the given language model as well as the language model perplexity for each sentence.

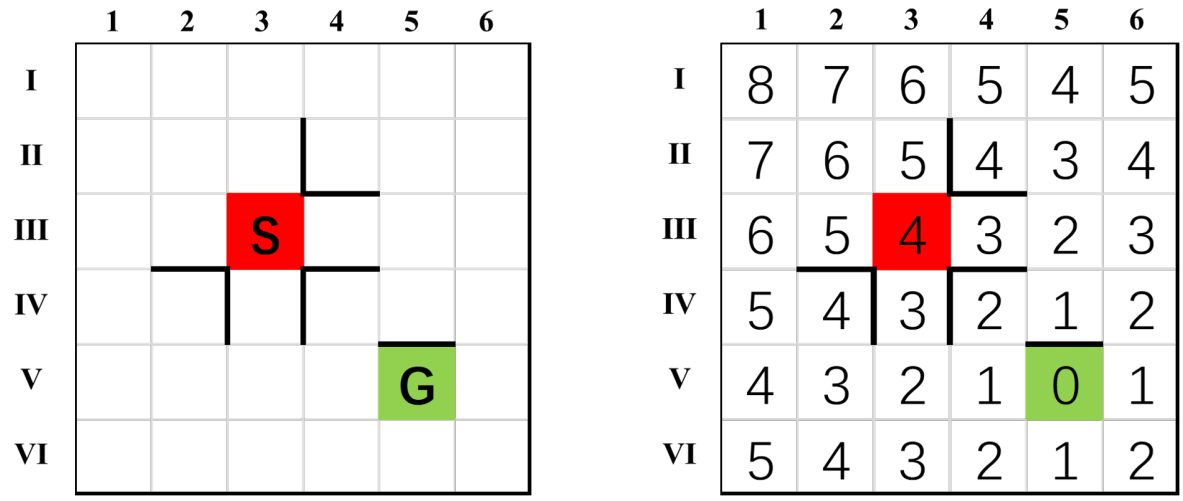
a) “Karlsruhe Institute of Technology” 5 P.

b) “Institute Karlsruhe Technology of” 5 P.

c) “of Institute Institute Technology” 5 P.

Exercise 2. Search (30 Points)

A robot is searching the shortest path to escape a 6x6 GridWorld from S (start) to G (goal). It has at most 4 possible moving directions in each cell: East (right), South (down), West (left), and North (up), with 1 cell in each step. If a wall (thick black line) lies in between two cells, the robot cannot move across. Each step incurs a uniform cost of 1. Figure 1a shows layout of the GridWorld, and Figure 1b shows the heuristic value estimates of each cell.



(a) Layout (b) Heuristic of Grid World

Figure 1: Grid World

All tasks below should be answered in the given Jupyter Notebook:

a. Code

Read the code in notebook and fill in the missing steps, to finish the implementations of DFS, BFS and A* search algorithms. 14 P.

b. DFS

Perform DFS algorithm to find a path from S to G. Write down the path and compute how many nodes are expanded in DFS search. 2 P.

c. BFS

Perform BFS algorithm to find a path from S to G. Write down the path and compute how many nodes are expanded in BFS search. 2 P.

d. A*

Given the heuristic function h(n) in Figure 1b, perform A* algorithm to find a path from S to G. Write down the path and compute how many nodes are expanded in A* search. 2 P.

e. Heuristic

What is the definition of an admissible heuristic? Is the heuristic from Figure (b) admissible? Why? 3 P.

f. Adversarial Search

Given a Minimax Search Tree in Figure 5, perform Alpha-Beta pruning and answer these questions:

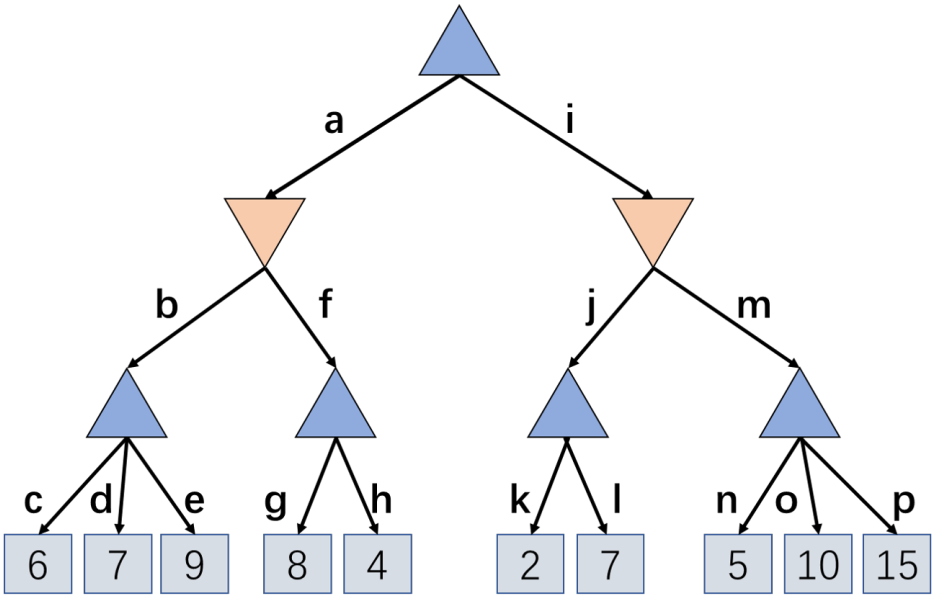


Figure 2: Alpha - Beta Pruning

- What is the value of the root? 3 P.
- Which path(s) among *a* - *p* will be pruned after search? Note if a "parent" path is pruned, then its children paths will be ignored. 4 P.

Exercise 3. TamagotchAI (Pen and Paper, 20 Points)

On the last episode of Storage Wars, you won the auction for a storage unit previously owned by a Japanese business man. In a box of old stuff, you find a small plastic toy with a display. Instantly, you realize it's a prototype of Tamagotchi [1] and, given you recently heard about all the cool AI algorithms, start thinking about calculating an optimal policy for playing with it.

In its initial state, Filibert (not sure why you named your digital pet like that) is *happy* and, independently of the state, you can choose from two actions. You can let Filibert *play* or make him *sleep*.

While being *happy*, *playing* yields a reward of $+20$ and, with a 75 % chance, Filibert will stay *happy*. Choosing to make Filibert *sleep* when he's happy will not alter his state and give 0 reward.

After a long day of playing, Filibert becomes *tired*. To be more precise, *playing* while being *happy* has a 25 % chance of making him *tired*, still resulting in a reward of $+20$. When being *tired*, the obvious thing would of course be to *sleep* making Filibert *happy* again every time. But *sleeping* is boring and therefore results in a reward of 0. Instead, you can again choose to *play* which results in the reduced reward of $+10$ due to his lack of energy. *Playing* while being *tired* will make Filibert more and more *tired* resulting in *death* caused by sleep deprivation in 10 % of the times. *Death* on this early prototype of a Tamagotchi is a terminal state and results in a reward of -100.

Remarks: In this exercise, the rewards are given for choosing a certain action in a certain state, not for the transition. Start with a value function $V_0(s) = 0 \forall s$. For $t > 0$, assign the value $V(s = \textit{dead}) = r(dead)$. The time dependent optimal policy for the problem with finite horizon H is given by $\pi_t^*(s) = \arg\max_a Q_{H-t}^*(s, a)$.

a. MDP

8 P.

Given the instructions above, draw the MDP of Filibert's states, similar to the racing example in the lecture. Draw a circle for each state (*happy*, *tired*, *dead*) and arrows between the states corresponding to the actions (*play*, *sleep*). Additionally, note the transition probabilities and the reward assigned for the respective transition.

b. Value Iteration

12 P.

Now that we figured out the underlying MDP, we can start calculating the optimal policy for a finite horizon. Recall the definition of the value function

$$V^*(s) = \max_a Q^*(s, a)$$

with

$$Q^*(s, a) = r(s, a) + \gamma \sum_{s'} P(s' \mid s, a)\, V^*(s').$$

Set $\gamma = 1$ and compute two iterations of Value Iteration. Write down the optimal policy for $t = 0$ and $t = 1$.

[1] `https://en.wikipedia.org/wiki/Tamagotchi`

Exercise 4. Optimal Policy Learning in a Grid World (35 Points)

You recently acquired a robot for cleaning you apartment but you were not happy with its performance and you decided to reprogram it using the latest AI algorithms. As a side consequence, the robot became self-aware and, whenever you are away, it prefers to play with toys rather than cleaning the apartment. Only the cat has noticed the strange behavior and attacks the robot. The robot is about to start its day and his current perception of the environment can be seen in Figure 3. The black squares marked with *O* denote obstacles, e.g., walls, tables, etc. The robot really wants to avoid crashing into them to protect his valuable sensors. The reward of such a state is set to $r_{\text{obstacle}} = -1000$. While the robot is waterproof, it developed a phobia of water *W*, imitating the cat *C*. The reward of $r_{\text{water}} = -1$. The robot doesn't appreciate the cat's behavior so it set its reward to $r_{\text{cat}} = -30$ for the state with the cat. The state containing a toy *T* is given $r_{\text{toy}} = 30$ as the robot enjoys playing with it. Some of the initial specification is still in the robot's reward function, which gives $r_{\text{dirt}} = 2$ to states with dirt *D*. The reward can be collected for *every* time-point the robot is at that state. The robot has the following actions: Down, right, up, left, and stay, with that – somehow strange – priority. In our system we represent the actions with the IDs zero to four, while the grid is indexed as `{row, column}`. The robot cannot leave the apartment as it is surrounded by walls. These walls are made of soft, yet impenetrable, material which does no harm to the robot.

Remark: Please answer all text questions of Exercise 4 in the corresponding cells of the Jupyter Notebook.

a. Implementing Value Iteration

20 P.

Implement the Value Iteration algorithm in the provided Jupyter Notebook. Carefully read the pseudocode and the hints given.

b. Finite Horizon Problem

4 P.

We first consider the finite horizon problem with horizon $T = 15$ steps. The goal of the robot is to maximize the expected return

$$J_\pi = \mathbb{E}_\pi \left[\sum_{t=1}^{T-1} r_t(s_t, a_t) + r_T(s_T) \right] \tag{1}$$

according to policy π, state s, action a, reward r, and horizon T. Since the rewards in our case are independent of the action and the next state, and the actions are, for now, deterministic, Equation 1 becomes

$$J_\pi = \sum_{t=1}^{T} r_t(s_t).$$

Determine the optimal policy using the Value Iteration algorithm and plot the resulting policy. Is the robot avoiding the cat? If not, why is this the case?

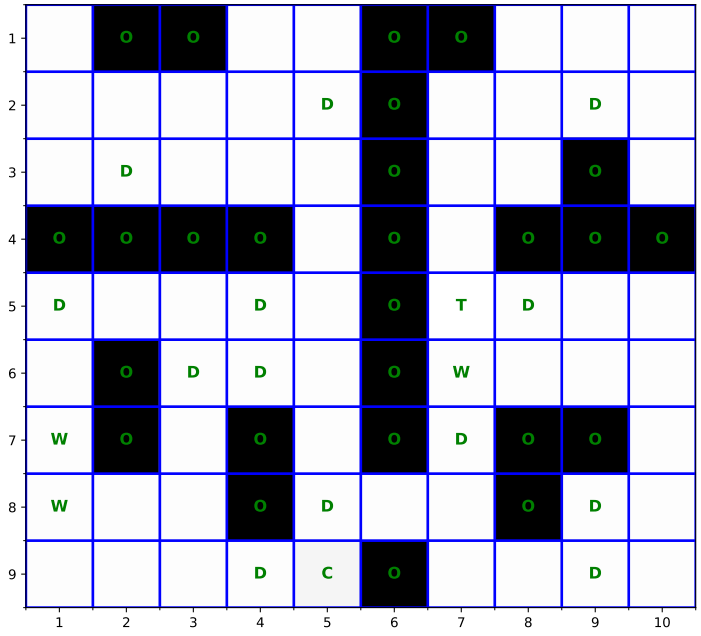


Figure 3: State of the apartment

c. Infinite Horizon Formulation (Pen and Paper)

3 P.

We now consider the infinite horizon problem where $T = \infty$. Rewrite Equation 1 for the infinite horizon case and add a discount factor γ. Briefly explain, why the discount factor is needed.

d. Infinite Horizon Problem

4 P.

Using the infinite horizon formulation, calculate the optimal policy for the robot. Use a discount factor of $\gamma = 0.8$ and plot the resulting policy. Comment on the new policy, what has changed?

e. Finite Horizon Problem with Stochastic Transition Function

4 P.

Due to the robot and cat fighting, your wooden floor deteriorated and you decided to polish it. The robot now slips and experiences problems with its controls. For each of the actions *down, right, up, left*, the robot has a probability 0.7 for achieving it and a probability of 0.1 to go an adjacent state, but

not the opposite direction. For example, if the action is *up* the robot would go *up* 70% of the time, and *left* or *right* 10% of the time. Additionally, the action can fail and the robot will remain on the same state with probability 0.1. For the action *stay*, the robot does not apply any controls so it is not affected and stays with probability 1.0. Using the finite horizon formulation, calculate the optimal policy for the robot. Use a time horizon of $T = 15$ steps. Comment on the policy and explain the behavior between states (2, 3) and (3, 3).

Übungsblatt 4

Klassifkation und Machine Learning

Abgabe online auf ILIAS bis 1 July 2020, 17:00
Auf jedem Übungsblatt können bis zu 100 Punkte gesammelt werden. Für jede vollen 100 Punkte in Summe erhalten Sie einen Bonuspunkt in der Klausur (max. 6 von 60).

Aufgabe 1. Sprachmodelle (15 Punkte)

Gegeben sei ein 3-gramm Sprachmodell mit dem Vokabular $V = \{\text{Karlsruhe}, \text{Institut}, \text{für}, \text{Technologie}\}$. Die Wahrscheinlichkeiten sind wie folgt:

		$w_i =$				
w_{i-2}	w_{i-1}	Karlsruhe	Institut	für	Technologie	</s>
n/a	<s>	0,5	0,25	0,1	0,1	0,05
<s>	Karlsruhe	0	0,8	0,05	0,05	0,1
<s>	Institut	1	0	0	0	0
<s>	für	0	0,9	0	0	0,1
<s>	Technologie	0	0,7	0,2	0	0,1
Karlsruhe	Karlsruhe	0,2	0,2	0,2	0,2	0,2
Karlsruhe	Institut	0	0	0,5	0,5	0
Karlsruhe	für	0	0,6	0,2	0,1	0,1
Karlsruhe	Technologie	0	0,4	0,4	0,1	0,1
Institut	Karlsruhe	0	0	0,4	0,4	0,2
Institut	Institut	0,2	0,2	0,2	0,2	0,2
Institut	für	0,1	0,1	0,1	0,1	0,6
Institut	Technologie	0,05	0,05	0,6	0,1	0,2
für	Karlsruhe	0,1	0,6	0,1	0,1	0,1
für	Institut	0,3	0	0,1	0,2	0,4
für	für	0	0	0	0	0
für	Technologie	0,2	0,2	0,2	0,2	0,2
Technologie	Karlsruhe	0	0,4	0,1	0,1	0,4
Technologie	Institut	0,4	0	0	0	0,6
Technologie	für	0,1	0,4	0	0	0,5
Technologie	Technologie	0,2	0,2	0,2	0,2	0,2

Hierbei steht <s> für das Satzanfangsszeichen und </s> für das Satzendzeichen.

Berechnen Sie die Wahrscheinlichkeiten für die folgenden Sätze gegeben dem Sprachmodell sowie die Perplexität für jeden Satz.

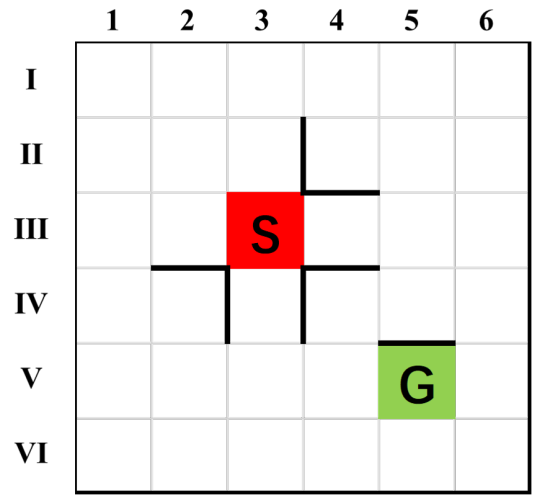
a) “Karlsruhe Institut für Technologie” 5 P.

b) “Institut Karlsruhe Technologie für” 5 P.

c) “für Institut Institut Technologie” 5 P.

Aufgabe 2. Suche in einer Gitterwelt (30 Punkte)

Ein Roboter in einer 6x6 Gitterwelt sucht den kürzesten Pfad von *S* (Start) nach *G* (Goal). Er kann sich pro Zug ein Feld in eine von vier Richtungen bewegen, Ost (rechts), Süd (runter), West (links), Nord (hoch), außer das Feld grenzt an eine Wand (dicke schwarze Linie), welche er nicht überwinden kann. Jeder Schritt hat Kosten von 1. Abbildung 4a zeigt einen Plan der Gitterwelt, Abbildung 4b die zugehörige Heuristik.



(a) Grid World

	1	2	3	4	5	6
I	8	7	6	5	4	5
II	7	6	5	4	3	4
III	6	5	4	3	2	3
IV	5	4	3	2	1	2
V	4	3	2	1	0	1
VI	5	4	3	2	1	2

(b) Heuristic of Grid World

a. Code

14 P.

Implementieren Sie die fehlenden drei Schritte im bereitgestellten Jupyter Notebook um die Implementierung von DFS, BFS und A* zu vervollständigen.

b. DFS

2 P.

Führen Sie den DFS Algorithmus aus, um einen Pfad von *S* nach *G* zu finden. Geben Sie den Pfad aus und berechnen Sie wie viele Knoten im DFS Suchbaum erweitert werden.

c. BFS

2 P.

Führen Sie den BFS Algorithmus aus, um einen Pfad von *S* nach *G* zu finden. Geben Sie den Pfad aus und berechnen Sie wie viele Knoten im BFS Suchbaum erweitert werden.

d. A*

2 P.

Führen Sie den A* Algorihmus mit der in Abbildung 4b gegebenen Heuristik Funktion $h(n)$ aus, um einen Pfad von *S* nach *G* zu finden. Geben Sie den Pfad aus und berechnen Sie, wie viele Knoten im A* Suchbaum erweitert werden.

e. Heuristik

3 P.

Definieren Sie eine zulässige Heuristik. Ist die Heuristik in Abbildung 4b zulässig? Wenn ja, warum? Wenn nein, warum nicht?

f. Alpha-Beta Pruning

Gegeben ist der Minimax Such Baum in Abbildung 5. Führen Sie Alpha-Beta pruning aus und beantworten Sie folgende Fragen:

- Was ist der Wert des Ausgangsknoten? 4 P.
- Welche der Pfade *a* bis *b* werden durch die Suche abgeschnitten? Anmerkung: Wenn ein "parent"Knoten abgschnitten wird, werden seine "child"Knoten ignoriert. 3 P.

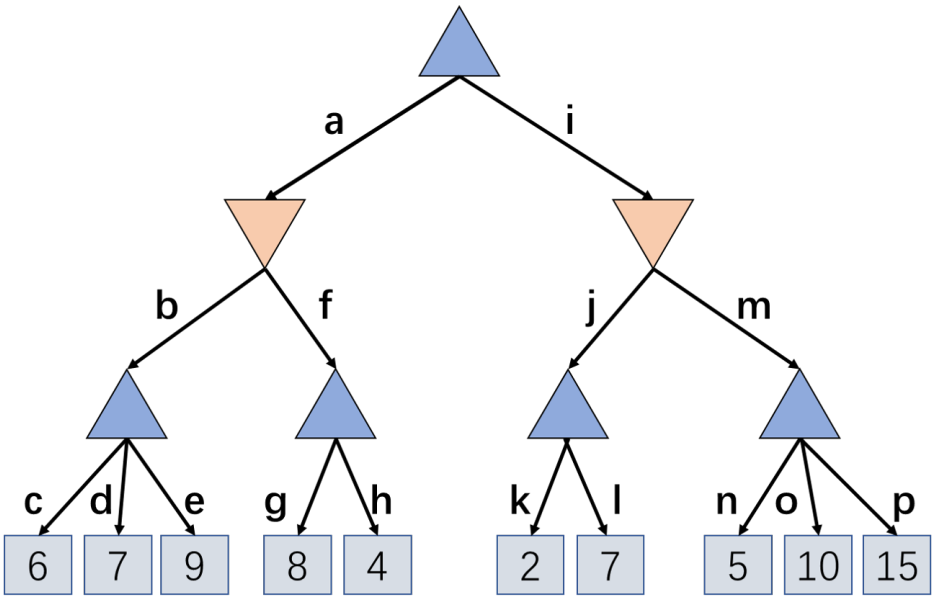


Abbildung 5: Alpha - Beta Pruning

Aufgabe 3. TamagotchAI (Pen and Paper, 20 Punkte)

In der letzten Episode von Storage Wars hast du den Zuschlag für einen Lagerraum bekommen, der zuvor einem japanischen Geschäftsmann gehörte. In einer Kiste mit alten Sachen findest du ein kleines Spielzeug mit einem Bildschirm. Sofort erkennst du, dass es sich um einen Prototyp eines Tamagotchi [2] handelt. Da du vor kurzem von duften KI Algorithmen gehört hast, fängst du an, an die Berechnung einer optimalen Strategie zu denken.

Im Ausgangszustand ist Filibert (wie auch immer du auf diesen Namen kamst) *glücklich*. Unabhängig von seinem Zustand kannst du zwei Aktionen ausführen. Du kannst Filibert *spielen* lassen, oder ihn *schlafen* schicken.

So lange Filibert *glücklich* ist bringt *spielen* eine Belohnung von $+20$ und mit einer Wahrscheinlichkeit von 75 % bleibt er *glücklich*. Filibert *schlafen* zu legen während er *glücklich* ist ändert seinen Zustand nicht und gibt eine Belohnung von 0.

Nach einem anstrengenden Tag voller Spiele wird Filibert *müde*. Genauer gesagt hat *spielen* eine 25% Chance Filibert *müde* zu machen, was jedoch immer noch eine Belohnung von $+20$ bedeutet. Im *müden* Zustand wäre das naheliegendste zu *schlafen*, was Filibert jedes mal wieder *glücklich* macht. Allerdings ist *schlafen* langweilig und bringt daher eine Belohnung von 0. Stattdessen kannst du immer noch entscheiden Filibert spielen zu lassen, was müdigkeitsbedingt nur noch eine Belohnung von $+10$ bringt. Im *müden* Zustand zu *spielen* macht Filibert mehr und mehr müde, resultierend in *Tod* durch völlige Verausgabung in 10 % der Fälle. *Tot* zu sein bei diesem frühen Prototyp ist ein endgültiger Zustand und ergibt eine Belohnung von -100.

Anmerkung: In dieser Aufgabe werden Belohnung für das Wählen einer bestimmten Aktion in einem bestimmten Zustand vergeben, nicht für den Zustand in den übergegangen wird. Starte mit einer Value Funktion $V_0(s) = 0 \, \forall s$. Setze $V_t(s = dead) = r(dead)$ für $t > 0$. Die zeitabhängige optimale Strategie für ein Problem mit endlichem Horizont ist gegeben durch $\pi_t^*(s) = \arg\max_a Q_{H-t}^*(s, a)$.

a. MDP

8 P.

Entwerfe mit den oben gegebenen Hinweisen das MDP welches die Zustände, Transitionen und Belohnungen von Filibert beschreibt (ähnlich zum Racing Beispiel aus der Vorlesung). Zeichne für jeden Zustand (*glücklich*, *müde*, *tot*) einen Kreis und verbinde sie mit den zu den Aktionen (*spielen*, *schlafen*) passenden Pfeilen um die Transitionen auszudrücken. Notiere zusätzlich die Übergangswahrscheinlichkeiten und die zugehörigen Belohnungen.

b. Value Iteration

12 P.

Nachdem du nun das dem Problem zugrunde liegende MDP herausgefunden hast, kannst du mit dem Berechnen der optimalen Strategie für einen endlichen Horizont beginnen. Erinnere dich an die Defintion der Value Funktion

$$V^*(s) = \max_a Q^*(s, a)$$

[2] `https://de.wikipedia.org/wiki/Tamagotchi`

mit

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a)\, V^*(s').$$

Setze $\gamma = 1$ und berechne zwei Iterationen von Value Iteration.

Aufgabe 4. Lernen einer optimalen Strategie in einer Gitterwelt (35 Punkte)

Du hast vor kurzem einen Reinigungsroboter für deine Wohnung gekauft. Da du mit seiner Leistung nicht zufrieden warst, hast du dich entschlossen ihn mit den heißesten KI Algorithmen neu zu programmieren. Infolge dessen hat dein Roboter ein Bewusstsein erlangt und bevorzugt es, wann immer du nicht anwesend bist, mit Spielsachen zu spielen, statt die Wohnung zu reinigen. Nur der Katze ist dieses seltsame Verhalten aufgefallen und sie attackiert den Roboter.

Der Roboter beginnt seinen Tag und Abbildung 6 zeigt seine aktuelle Wahrnehmung der Wohnung. Die mit einem *O* markierten schwarzen Quadrate sind Hindernisse wie Wände, Tische etc. und der Roboter will unter allen Umständen verhindern mit ihnen zu kollidieren. Die Belohnung für einen solchen Zustand ist $r_{\text{obstacle}} = -1000$

Obwohl der Roboter wasserdicht ist hat er, vermutlich die Katze *C* imitierend, eine Angst vor Wasser *W* entwickelt und bekommt in diesem Zustand eine Belohnung von $r_{\text{water}} = -1$. Desweiteren mißfällt ihm das Verhalten der Katze, so daß er bei einem Antreffen eine Belohnung von $r_{\text{cat}} = -30$ bekommt. Der Zustand mit dem Spielzeug *T* hingegen ist eine Belohnung von $r_{\text{toy}} = 30$ wert, da ihm das Spielzeug sehr gut gefällt. Trotz der Neuprogrammierung ist ein Teil seines ursprünglichen Verhaltens noch in seiner Belohnungsfunktion enthalten, so daß Zustände mit Dreck *D* eine Belohnung von $r_{\text{dirt}} = 2$ bringen. Die Zustände der Wohnung sind permanent, eine Belohnung kann also zu jedem Zeitpunkt in dem sich der Roboter in einem Zustand befindet erhalten werden. Der Roboter kann die folgenden Aktionen ausführen: Runter, rechts, hoch, links und stehen bleiben – priorisiert in exakt dieser Reihenfolge, wenn er sich nicht entscheiden kann. Die Aktionen werden mit den Zahlen 0 bis 4 kodiert, die Gitterwelt per Zeilen- und Spaltenummer. Das Apartment kann nicht verlassen werden, da es von Wänden umgeben ist. Diese Wände sind aus einem weichen, dennoch undurchdringlichen, Material welches dem Roboter keinen Schaden zufügt.

a. Value Iteration Implementierung

20 P.

Implementiere Value Iteration nach dem Pseudocode im bereitgestellten Jupyer Notebook. Achte auf die gegebenen Hinweise.

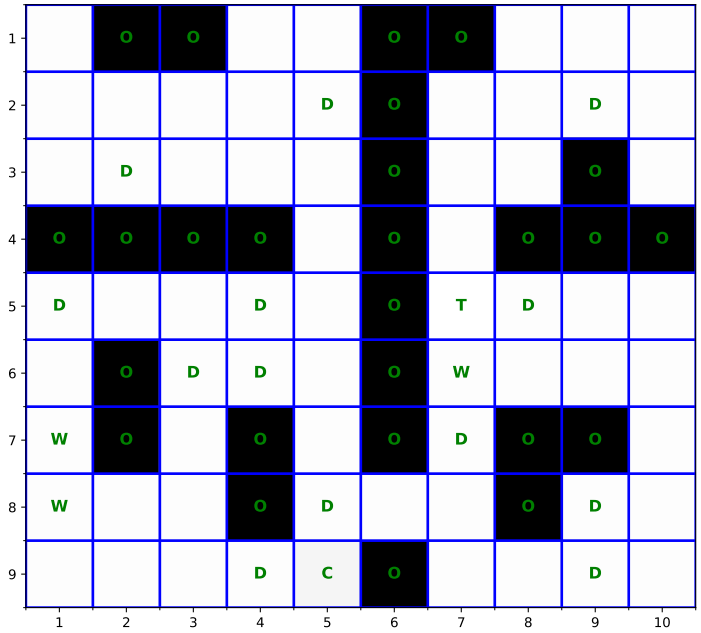


Abbildung 6: State of the apartment

b. Problem mit endlichem Horizont

4 P.

In der ersten Aufgabe schauen wir auf das Problem mit einem endlichen Horizont von $T = 15$ Schritten. Das Ziel des Roboters ist, den erwarteten Return

$$J_\pi = \mathbb{E}_\pi \left[\sum_{t=1}^{T-1} r_t(s_t, a_t, s_{t+1}) + r_T(s_T) \right] \tag{2}$$

über die unter der Strategie π besuchten Zustände s, ausgeführten Aktionen a und Belohnungen r innerhalb des Horizonts T zu maximieren. Da die Belohnungen in unserem Fall unabhängig von der Aktion und des nächsten Zustandes sind, und Aktionen zunächst deterministisch ausgeführt werden lässt siche Gleichung 2 umformen zu

$$J_\pi = \sum_{t=1}^{T} r_t(s_t).$$

Bestimme die optimale Strategie und plotte sie. Wird der Katze aus dem Weg gegangen? Falls nicht, warum nicht?

c. Problem mit unendlichem Horizont (pen and paper)

3 P.

Jetzt schauen wir auf das Problem mit unendlichem Horizont mit $T = \infty$. Formuliere Gleichung 2 für diesen Fall und füge einen Discount Faktor γ ein. Gebe eine kurze Erklärung, warum der Discount Faktor notwendig ist.

d. Problem mit unendlichem Horizont

4 P.

Starte Value Iteration mit der Formulierung für den Fall mit unendlichem Horizont und berechne die optimale Strategie für den Roboter. Benutze hierfür einen Discount Faktor von $\gamma = 0.8$ und plotte die Strategie. Beschreibe die neue Strategie, was hat sich geändert?

e. Problem mit endlichem Horizont und stochastischer Übergangsfunktion

4 P.

Nach einigen Scharmützeln zwischen dem Roboter und der Katze ist der Parkettboden so mitgenommen, daß du ihn hast polieren lassen. Dies hat zur Folge, dass der Roboter seine Aktionen auf dem glatten Boden nicht mehr genau kontrollieren kann. Jede der Aktionen *runter, rechts, hoch, links* des Roboters hat eine Wahrscheinlichkeit von 0.7 korrekt ausgeführt zu werden, mit einer Wahrscheinlichkeit von 0.1 landet er in einem der angrenzenden Zustände, außer dem Zustand in entgegengesetzter Richtung. Beispielsweise wird die Aktion *hoch* in 70 % der Fälle korrekt ausgeführt, in jeweils 10 % der Fälle wird die "verrauschteÄktion *links* bzw. *rechts* ausgeführt. Zusätzlich kann die Aktion in 10 % der Fälle fehlschlagen und der Roboter verbleibt im gleichen Zustand. Da die Aktion *stehenbleiben* keine Bewegung zur Folge hat bleibt sie von der Glätte des Bodens unbeeinflusst.

Berechne die optimale Strategie mit der Formulierung für endliche Horizonte. Benutz wieder einen Horizont von $T = 15$ Schritten. Beschreibe die Strategie und erkläre das Verhalten zwischen den Zuständen (2, 3) und (3, 3).