More on the implementation of the BDI Control Loop

Autonomous Software Agents

A.A. 2022-2023

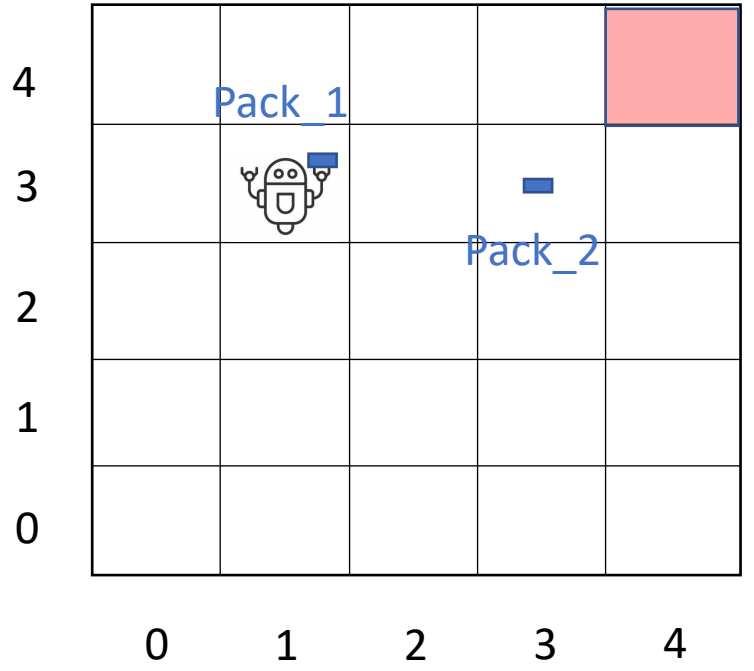
Prof. Paolo Giorgini

Dr. Marco Robol

UNIVERSITY OF TRENTO - Italy

Department of Information and Communication Technology

Intentions



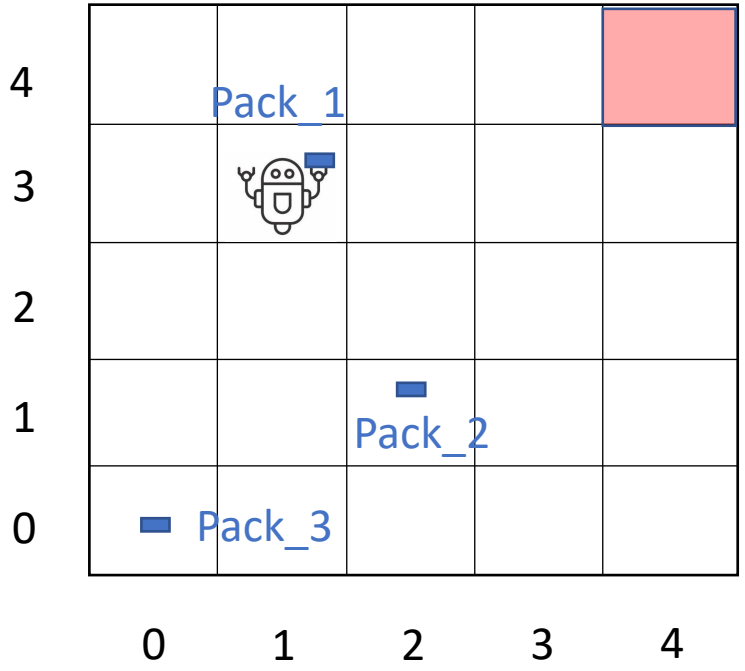

```
I = [in(pack_1,4,4)]
```

```
if (in(Pack,X,Y) ^ arm(FREE)) -> carry(Pack)
if (carry(Pack) ^ del_zone(X,Y)) -> in(Pack,X,Y)
```

```
I = [carry(pack_2), in(pack_1,4,4)]                    # order is important
P = [move(RIGHT),move(RIGHT),pick_up(pack_2)]

Completed P -> I = [in(pack_1,4,4), in(pack_2,4,4)]    # order is not important
               P = [move(RIGHT),move(UP),put_down(pack_1)]
Completed P -> I = [in(pack_2,4,4)]
               P = [put_down(pack_2)]
```

Options and Intention Revision



I = [in(pack_1,4,4)]

Options = {carry(pack_2), carry(pack_3))

Filtering

carry(pack_2) and carry(pack_3) are not compatible

Revision

I=[carry(pack_2),in(pack_1,4,4)]

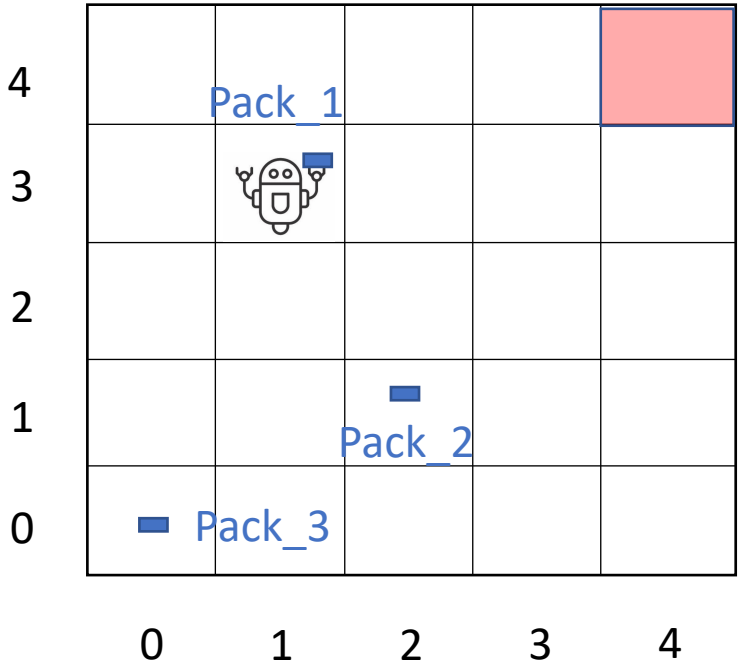
I=[carry(pack_3),in(pack_1,4,4)]

What is the best?

Utility for an intention

- C_i = cost to achieve an Intention I_i
  - It depends on the cost of the plan that will be applied
  - Given I_i and P the set of all possible plans to achieve I_i , C_i is the min cost of plans in P
- R_i = reward to achieve an Intention I_i
- U_i = utility to achieve an Intention I_i

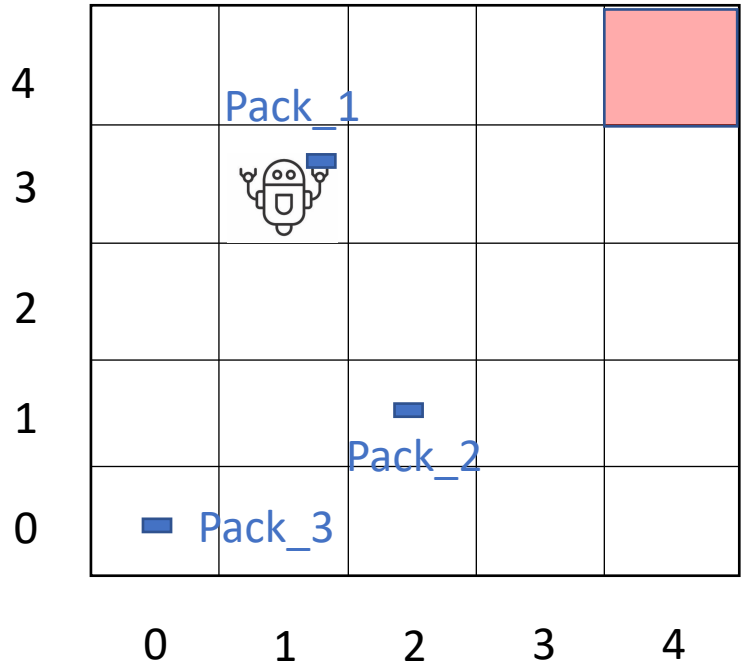
$$U_i = R_i - C_i$$



	R	C
in(pack_1,4,4)	9	5
carry(pack_2)	2	4
carry(pack_3)	2	5

They includes pick_up and put_down actions

It could be a bit more complex




```
I=[carry(pack_2),in(pack_1,4,4)]
```

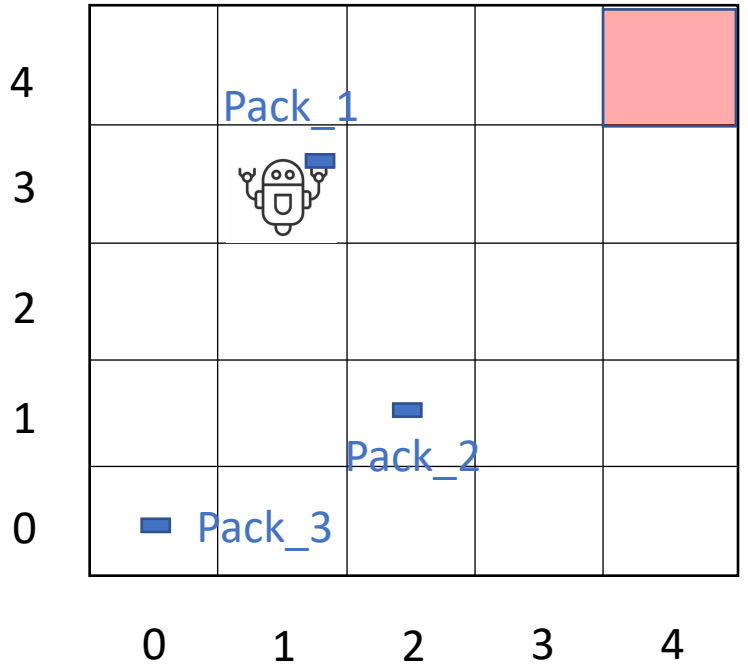
	R	C
in(pack_1,4,4)	9	5
carry(pack_2)	2	4
carry(pack_3)	2	5

If the agent decide for carry(pack_2) the agent will move to the position (2,1) and the cost of moving in (4,4) will be 6 and not 5 as at the beginning

	R	C
carry(pack_2),in(pack_1,4,4)	2+9 = 11	4+6 = 10
carry(pack_3),in(pack_1,4,4)	2+9 = 11	4+9 = 13

Priority of intentions

	R	C
carry(pack_2),in(pack_1,4,4)	2+9 = 11	4 + 6 = 10
carry(pack_3),in(pack_1,4,4)	2+9 =11	4+9 = 13



What is the best order for `I = {carry(pack_2),in(pack_1,4,4)}`

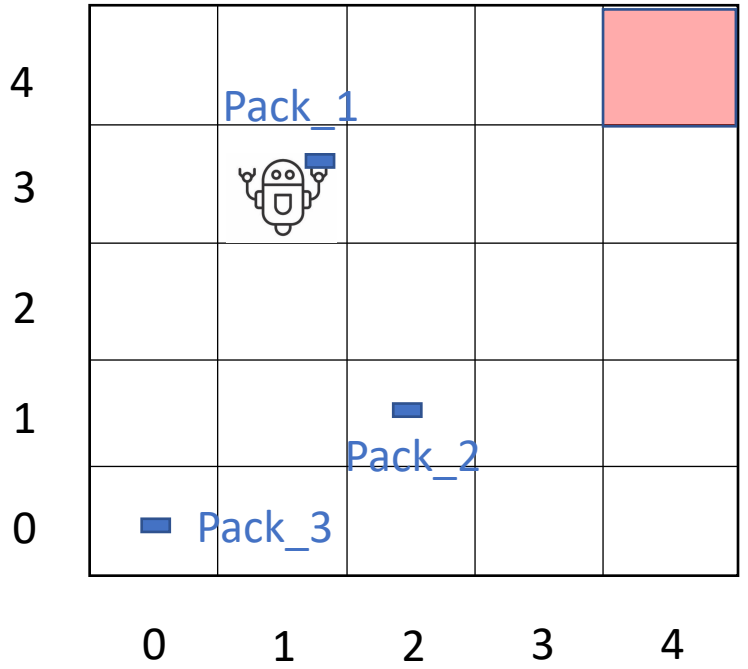
If rewards do not change over time

	R	C
carry(pack_2),in(pack_1,4,4)	2+9 = 11	4+6 =10
in(pack_1,4,4),carry(pack_2)	9+2 = 11	5+6 = 11

… but if they change

- Like in the Deliveroojs

At the time of the decision



	R	C
in(pack_1,4,4)	9	5
carry(pack_2)	10	4
carry(pack_3)	10	5

	Points
carry(pack_2),in(pack_1,4,4)	6 + 0 = 6
in(pack_1,4,4),carry(pack_2)	4 + 1 = 5

In the case you have also to deliver pack_2 to earn points

	Points
carry(pack_2),in(pack_1,4,4)	0 + 0 = 0
in(pack_1,4,4),carry(pack_2)	4 + 0 = 4