Multi-Agent Path Finding

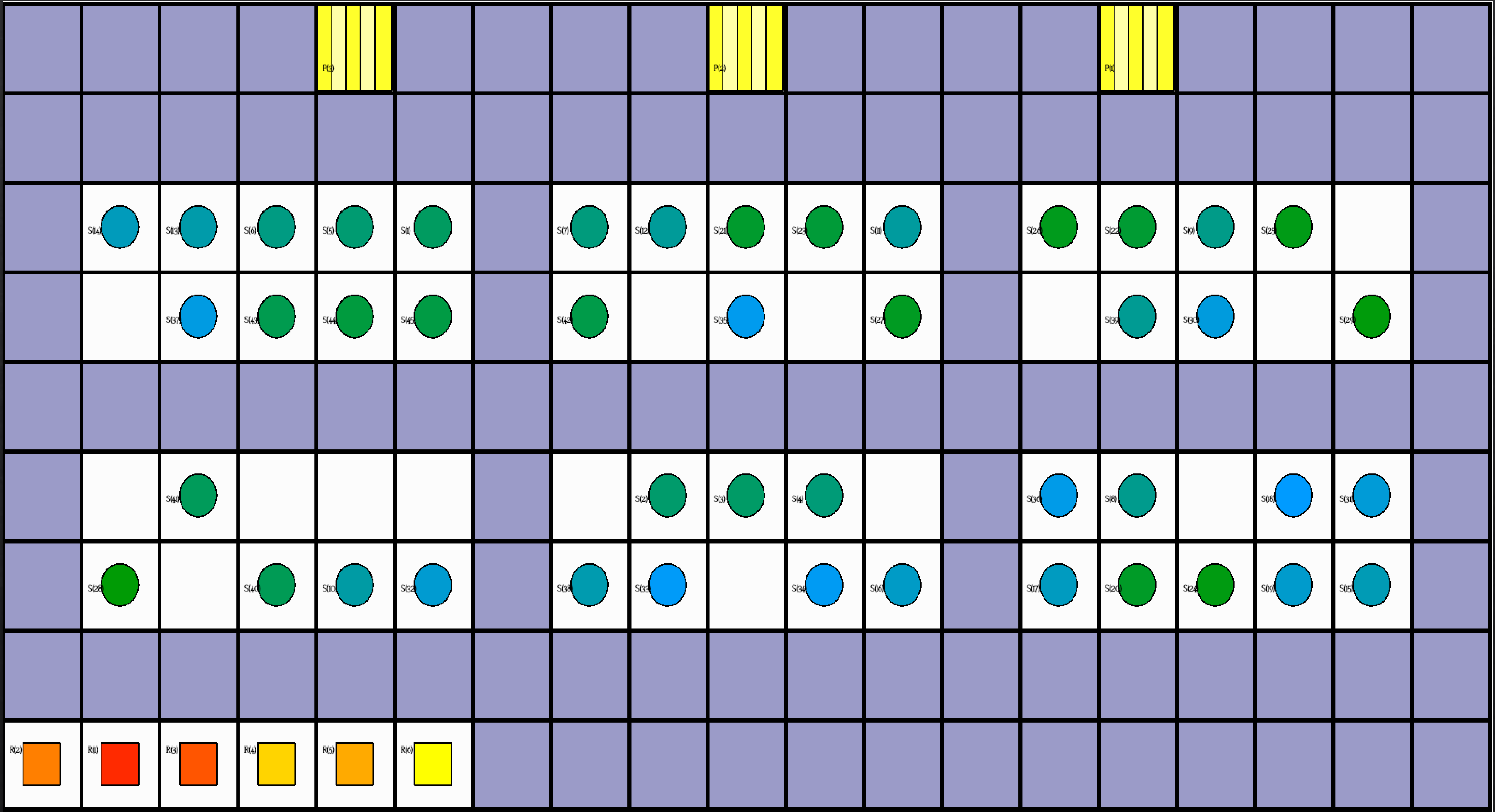
by Marcus Funke and Max Wiedenhöft

Table of Contents

- Explanation of asprilo and MAPF
- Details of the Merger
- Results and comparisons
- Conclusions
- Plans going forward

Explanation of asprilo and MAPF

- asprilo
  - Benchmark environment
    - Benchmark Generator
    - Solution Checker
    - Visualizer
    - Referencing Encodings


P(3)
P(2)
P(1)
S(14)
S(13)
S(6)
S(5)
S(1)
S(7)
S(12)
S(21)
S(23)
S(11)
S(26)
S(22)
S(9)
S(25)
S(37)
S(43)
S(44)
S(45)
S(42)
S(35)
S(27)
S(39)
S(30)
S(29)
S(41)
S(2)
S(3)
S(4)
S(36)
S(8)
S(18)
S(31)
S(28)
S(40)
S(10)
S(32)
S(38)
S(33)
S(34)
S(16)
S(17)
S(20)
S(24)
S(19)
S(15)
R(2)
R(1)
R(3)
R(4)
R(5)
R(6)

Explanation of asprilo and MAPF

- asprilo
  - Benchmark environment
  - Various problem Domains(A,B,C,M)

Explanation of asprilo and MAPF

- asprilo
  - Benchmark environment
  - Various problem Domains(A,B,C,M)
- **M**ulti **A**gent **P**ath **F**inding

Details of the Merger

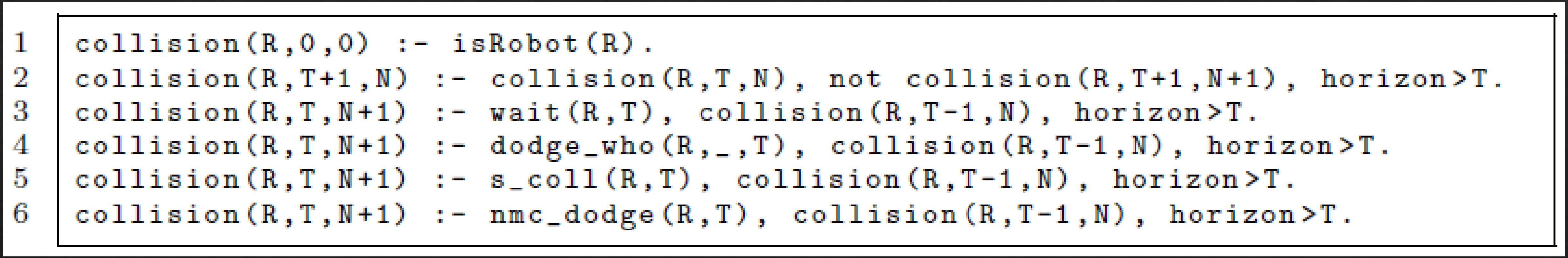
- 2.1 Basics

Basics

- plan_occurs/3 & step_move/4 & step_pickup/4 & step_putdown/4

Basics

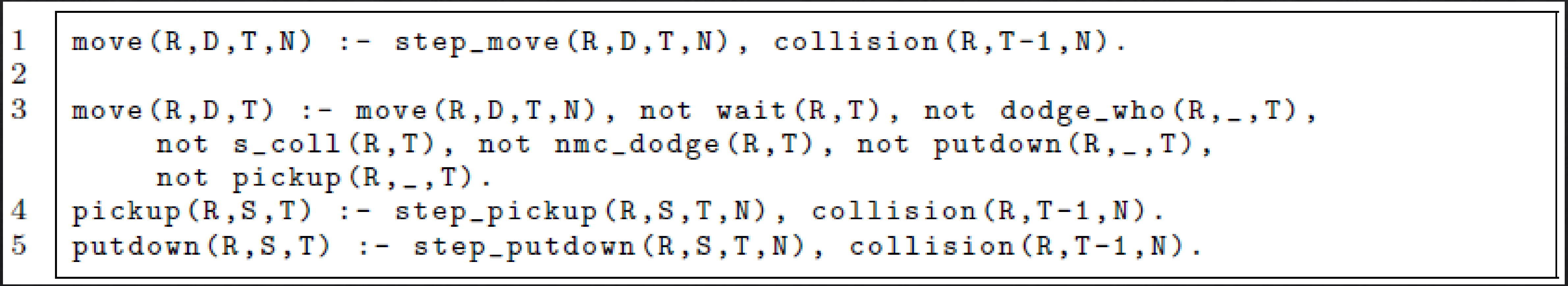
- plan_occurs/3 & step_move/4 & step_pickup/4 & step_putdown/4
- collision/3



```
collision(R,0,0) :- isRobot(R).
collision(R,T+1,N) :- collision(R,T,N), not collision(R,T+1,N+1), horizon>T.
collision(R,T,N+1) :- wait(R,T), collision(R,T-1,N), horizon>T.
collision(R,T,N+1) :- dodge_who(R,_,T), collision(R,T-1,N), horizon>T.
collision(R,T,N+1) :- s_coll(R,T), collision(R,T-1,N), horizon>T.
collision(R,T,N+1) :- nmc_dodge(R,T), collision(R,T-1,N), horizon>T.
```

Basics

- plan_occurs/3 & step_move/4 & step_pickup/4 & step_putdown/4
- collision/3
- move/4
- move/3 & pickup/3 & putdown/3



```
move(R,D,T,N) :- step_move(R,D,T,N), collision(R,T-1,N).

move(R,D,T) :- move(R,D,T,N), not wait(R,T), not dodge_who(R,_,T),
     not s_coll(R,T), not nmc_dodge(R,T), not putdown(R,_,T),
     not pickup(R,_,T).
pickup(R,S,T) :- step_pickup(R,S,T,N), collision(R,T-1,N).
putdown(R,S,T) :- step_putdown(R,S,T,N), collision(R,T-1,N).
```

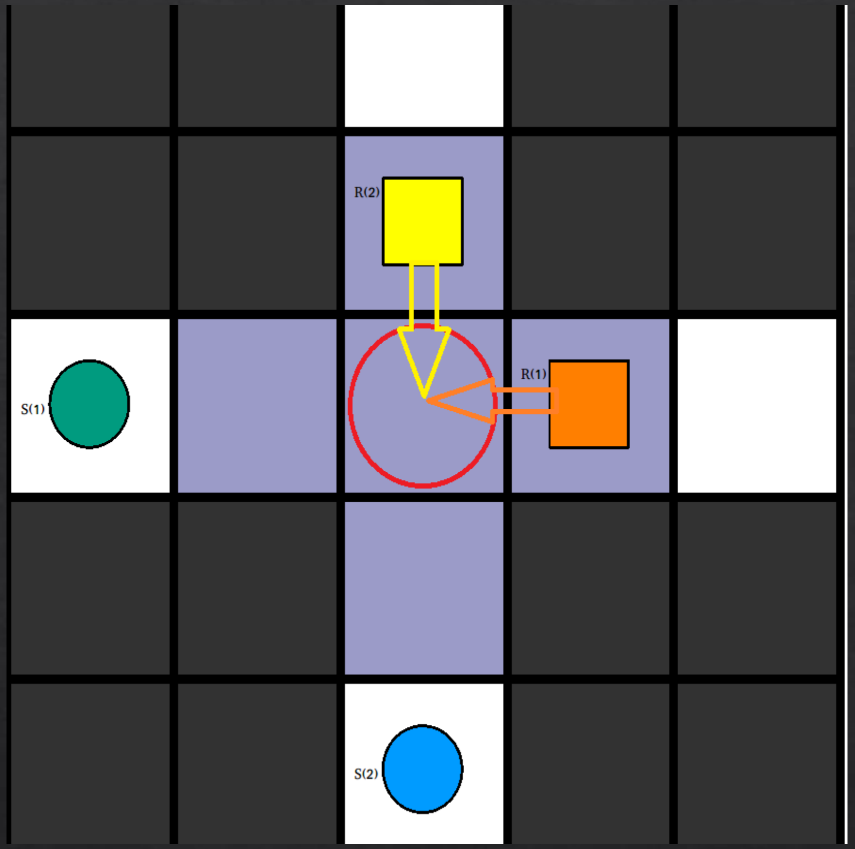
Basics

- plan_occurs/3 & step_move/4 & step_pickup/4 & step_putdown/4
- collision/3
- move/4
- move/3 & pickup/3 & putdown/3
- position/3 & carries/3

Details of the Merger

- 2.1 Basics
- 2.2 Use cases
    - 2.2.1 vertex collision

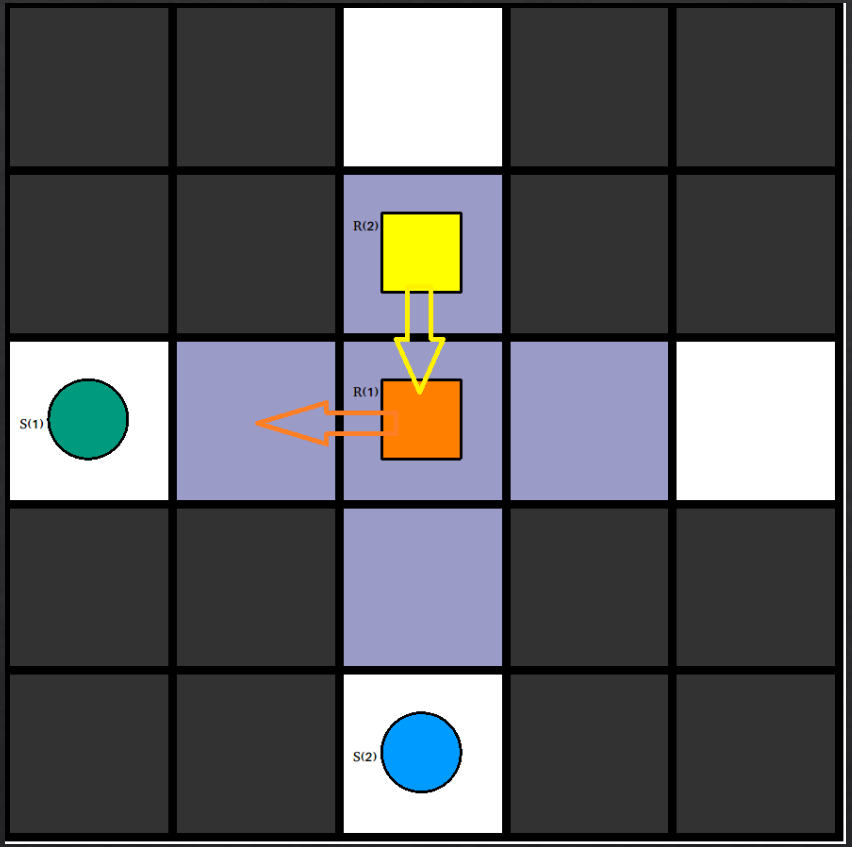
Vertex collision



- wait/2

```
wait(R2,T) :- move(R1,D1,T,N1), move(R2,D2,T,N2), position(R1,C1',T-1),
    N1>N2, nextto(C1',D1,C), position(R2,C2',T-1), nextto(C2',D2,C),
    not dodge_who(R2,_,T), not wait(R1,T).
wait(R2,T) :- move(R1,D1,T,N), move(R2,D2,T,N), position(R1,C1',T-1), R2>R1,
    nextto(C1',D1,C), position(R2,C2',T-1), nextto(C2',D2,C),
    not dodge_who(R2,_,T), not dodge_who(R1,_,T), not no_dodge(R2,T).
```

```
step_move(R,D,T1+1,N+1) :- step_move(R,D,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
step_pickup(R,S,T1+1,N+1) :- step_pickup(R,S,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
step_putdown(R,S,T1+1,N+1) :- step_putdown(R,S,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
```

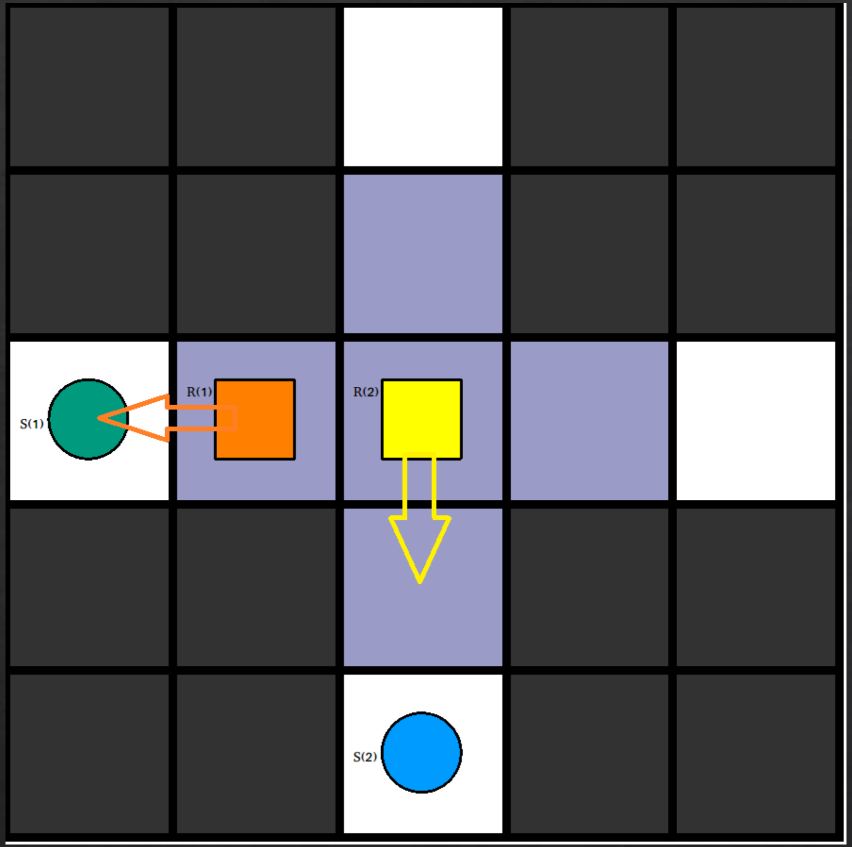


Vertex collision

- wait/2

```
wait(R2,T) :- move(R1,D1,T,N1), move(R2,D2,T,N2), position(R1,C1',T-1),
    N1>N2, nextto(C1',D1,C), position(R2,C2',T-1), nextto(C2',D2,C),
    not dodge_who(R2,_,T), not wait(R1,T).
wait(R2,T) :- move(R1,D1,T,N), move(R2,D2,T,N), position(R1,C1',T-1), R2>R1,
    nextto(C1',D1,C), position(R2,C2',T-1), nextto(C2',D2,C),
    not dodge_who(R2,_,T), not dodge_who(R1,_,T), not no_dodge(R2,T).
```

```
step_move(R,D,T1+1,N+1) :- step_move(R,D,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
step_pickup(R,S,T1+1,N+1) :- step_pickup(R,S,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
step_putdown(R,S,T1+1,N+1) :- step_putdown(R,S,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
```



Vertex collision

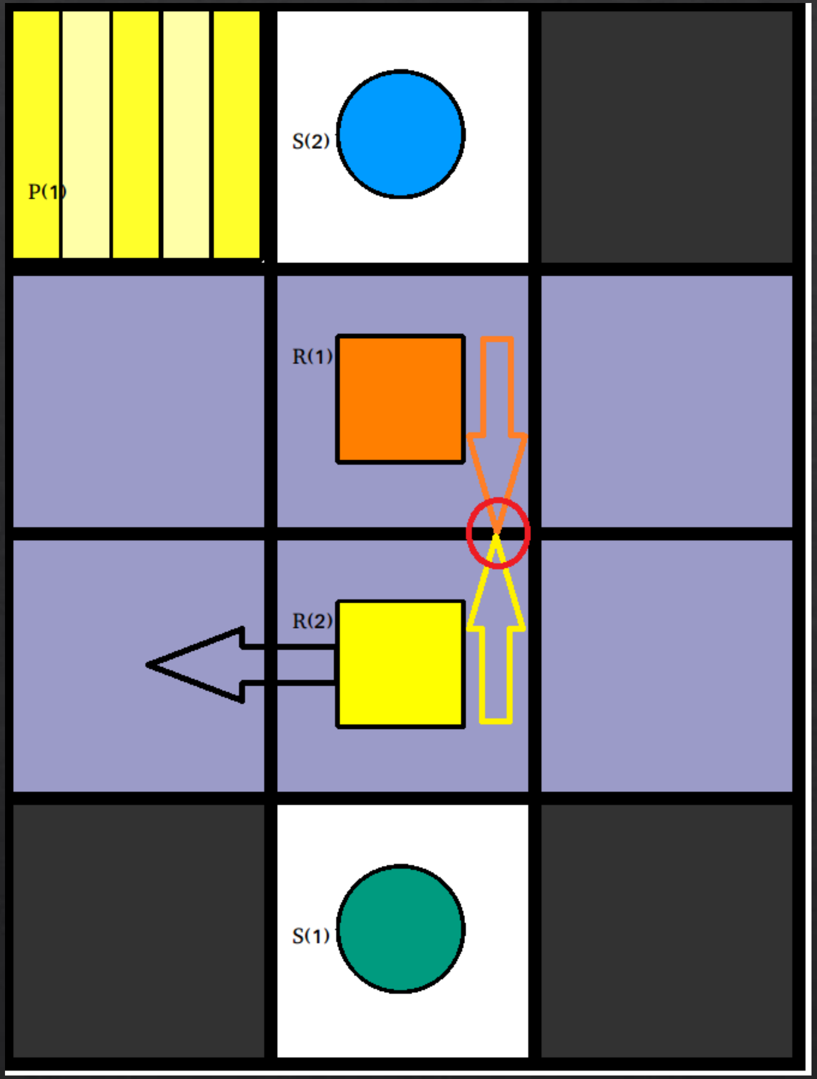
- wait/2

```
wait(R2,T) :- move(R1,D1,T,N1), move(R2,D2,T,N2), position(R1,C1',T-1),
    N1>N2, nextto(C1',D1,C), position(R2,C2',T-1), nextto(C2',D2,C),
    not dodge_who(R2,_,T), not wait(R1,T).
wait(R2,T) :- move(R1,D1,T,N), move(R2,D2,T,N), position(R1,C1',T-1), R2>R1,
    nextto(C1',D1,C), position(R2,C2',T-1), nextto(C2',D2,C),
    not dodge_who(R2,_,T), not dodge_who(R1,_,T), not no_dodge(R2,T).
```

```
step_move(R,D,T1+1,N+1) :- step_move(R,D,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
step_pickup(R,S,T1+1,N+1) :- step_pickup(R,S,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
step_putdown(R,S,T1+1,N+1) :- step_putdown(R,S,T1,N), wait(R,T),
    collision(R,T-1,N),(T1+1)>T, horizon>T1.
```

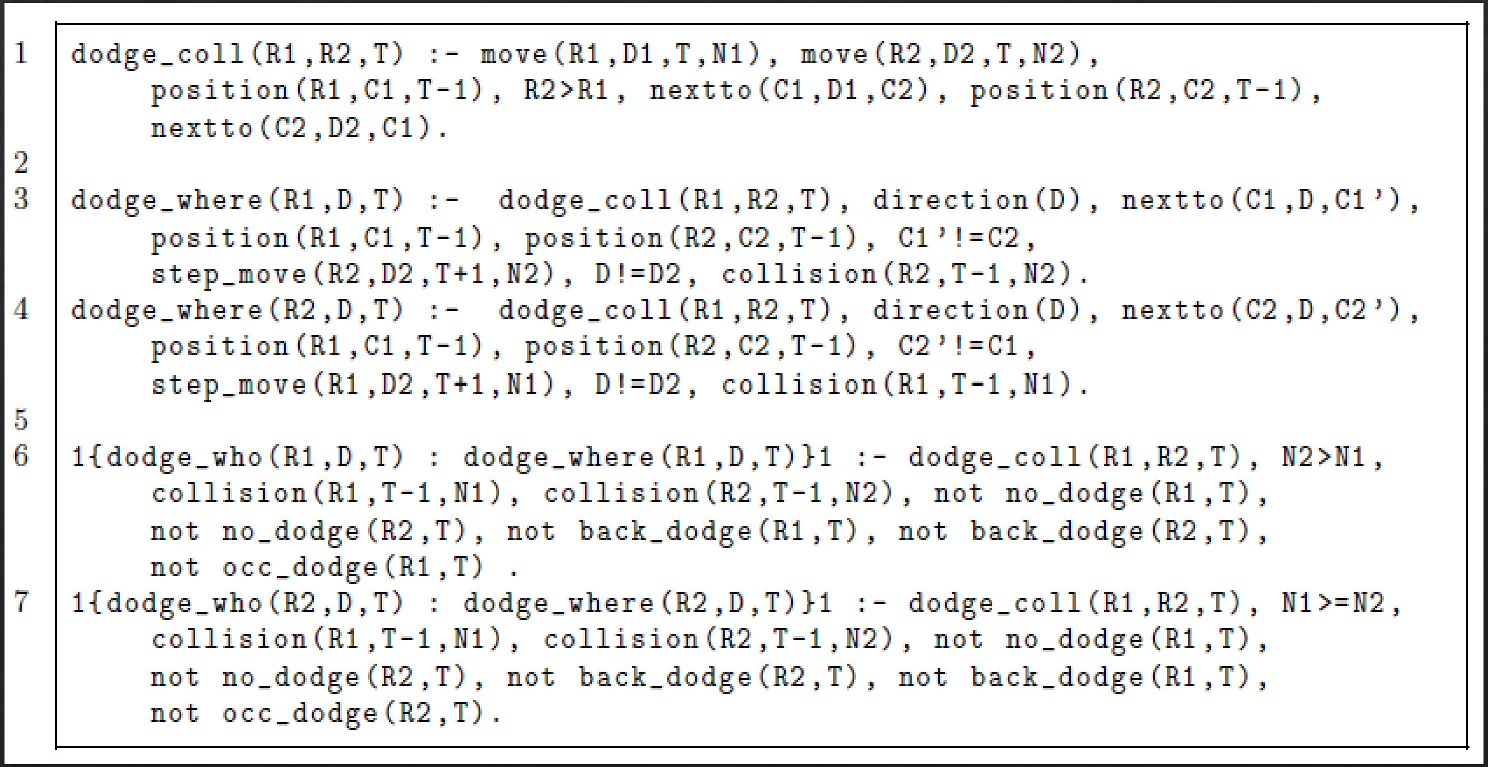
Details of the Merger

- 2.1 Basics
- 2.2 Use cases
  - 2.2.1 vertex collision
  - 2.2.2 edge collision

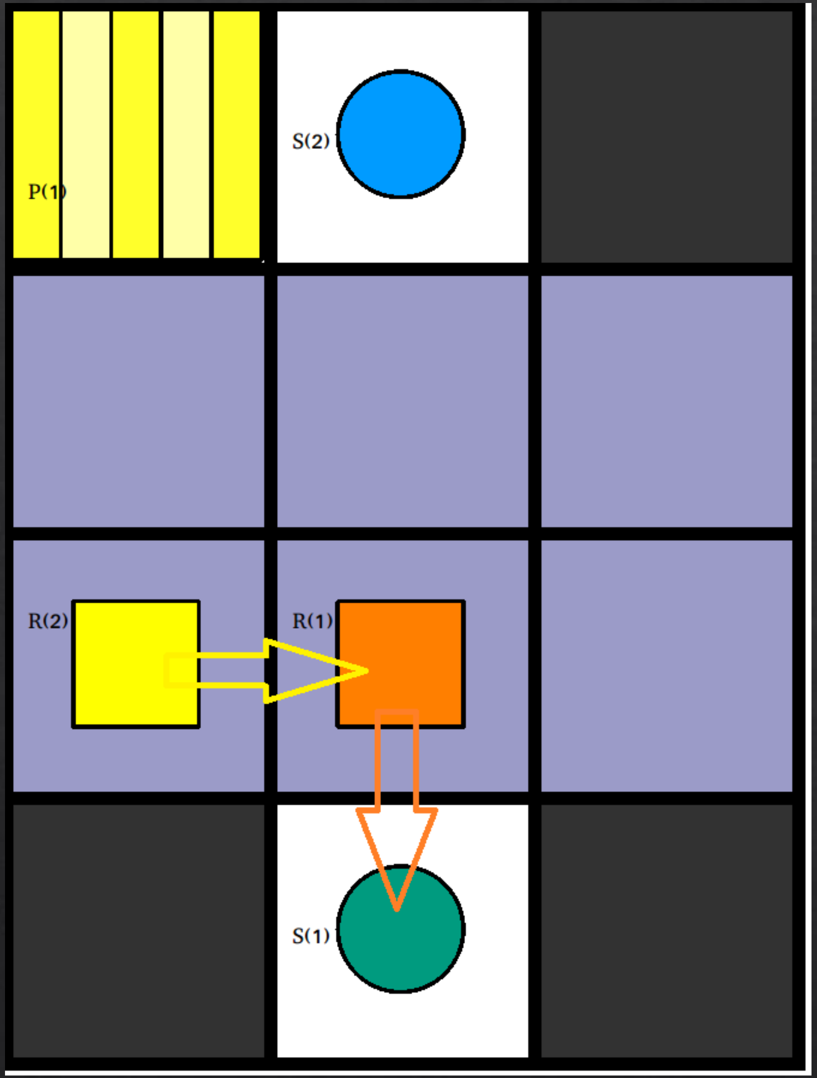


Edge collision

- dodge_coll/3
- dodge_where/3
- dodge_who/3

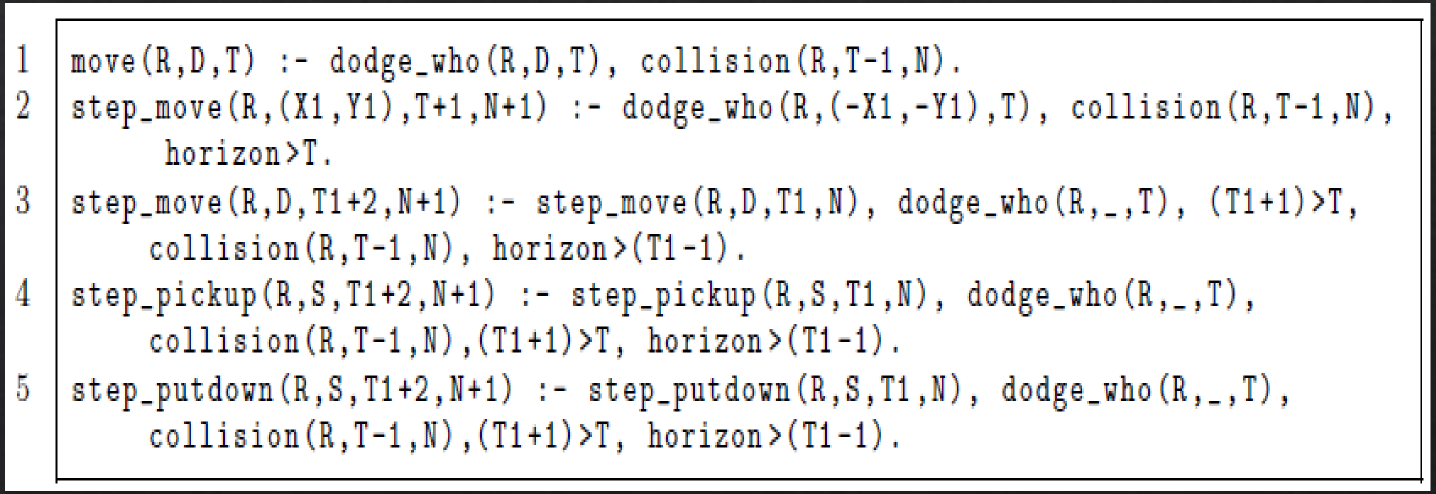


```
dodge_coll(R1,R2,T) :- move(R1,D1,T,N1), move(R2,D2,T,N2),
    position(R1,C1,T-1), R2>R1, nextto(C1,D1,C2), position(R2,C2,T-1),
    nextto(C2,D2,C1).

dodge_where(R1,D,T) :-  dodge_coll(R1,R2,T), direction(D), nextto(C1,D,C1'),
    position(R1,C1,T-1), position(R2,C2,T-1), C1'!=C2,
    step_move(R2,D2,T+1,N2), D!=D2, collision(R2,T-1,N2).
dodge_where(R2,D,T) :-  dodge_coll(R1,R2,T), direction(D), nextto(C2,D,C2'),
    position(R1,C1,T-1), position(R2,C2,T-1), C2'!=C1,
    step_move(R1,D2,T+1,N1), D!=D2, collision(R1,T-1,N1).

1{dodge_who(R1,D,T) : dodge_where(R1,D,T)}1 :- dodge_coll(R1,R2,T), N2>N1,
    collision(R1,T-1,N1), collision(R2,T-1,N2), not no_dodge(R1,T),
    not no_dodge(R2,T), not back_dodge(R1,T), not back_dodge(R2,T),
    not occ_dodge(R1,T) .
1{dodge_who(R2,D,T) : dodge_where(R2,D,T)}1 :- dodge_coll(R1,R2,T), N1>=N2,
    collision(R1,T-1,N1), collision(R2,T-1,N2), not no_dodge(R1,T),
    not no_dodge(R2,T), not back_dodge(R2,T), not back_dodge(R1,T),
    not occ_dodge(R2,T).
```

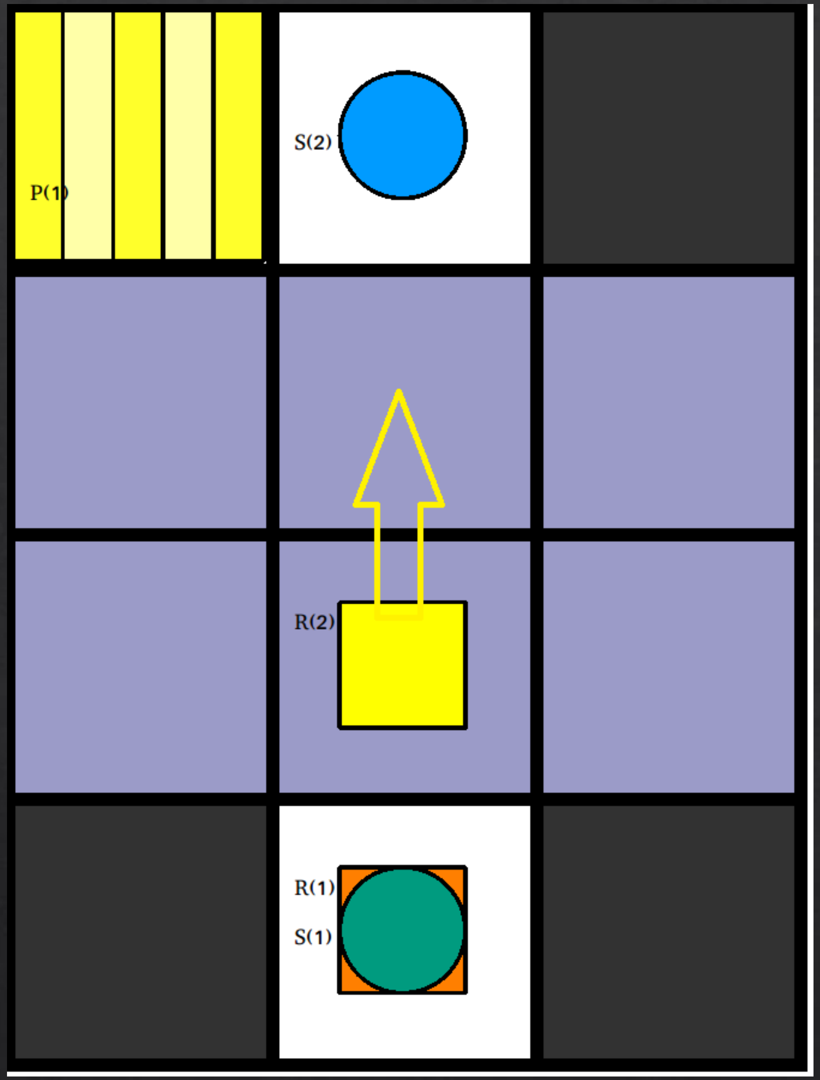


Edge collision

- dodge_coll/3
- dodge_where/3
- dodge_who/3

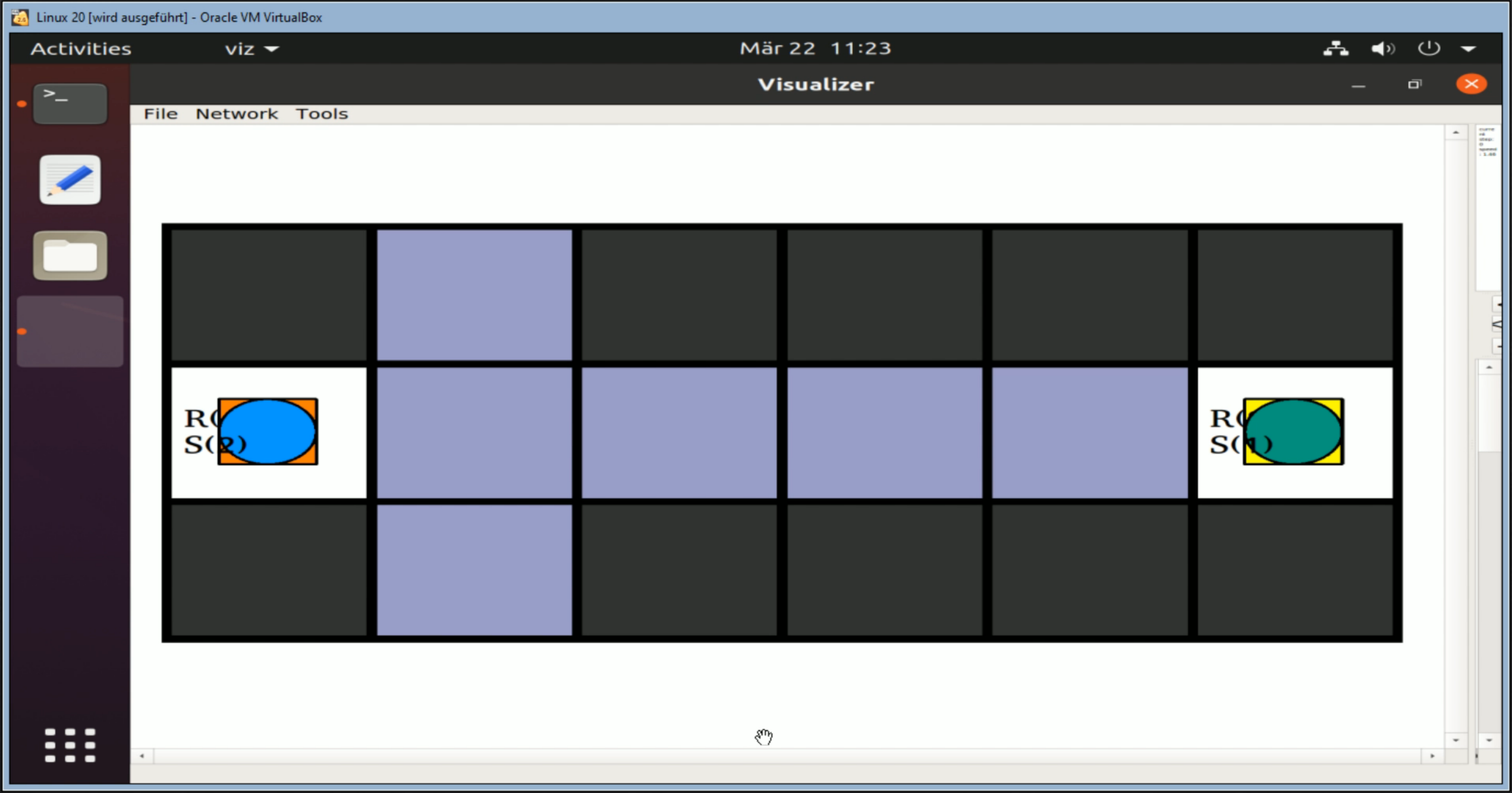


```
move(R,D,T) :- dodge_who(R,D,T), collision(R,T-1,N).
step_move(R,(X1,Y1),T+1,N+1) :- dodge_who(R,(-X1,-Y1),T), collision(R,T-1,N),
      horizon>T.
step_move(R,D,T1+2,N+1) :- step_move(R,D,T1,N), dodge_who(R,_,T), (T1+1)>T,
     collision(R,T-1,N), horizon>(T1-1).
step_pickup(R,S,T1+2,N+1) :- step_pickup(R,S,T1,N), dodge_who(R,_,T),
     collision(R,T-1,N),(T1+1)>T, horizon>(T1-1).
step_putdown(R,S,T1+2,N+1) :- step_putdown(R,S,T1,N), dodge_who(R,_,T),
     collision(R,T-1,N),(T1+1)>T, horizon>(T1-1).
```



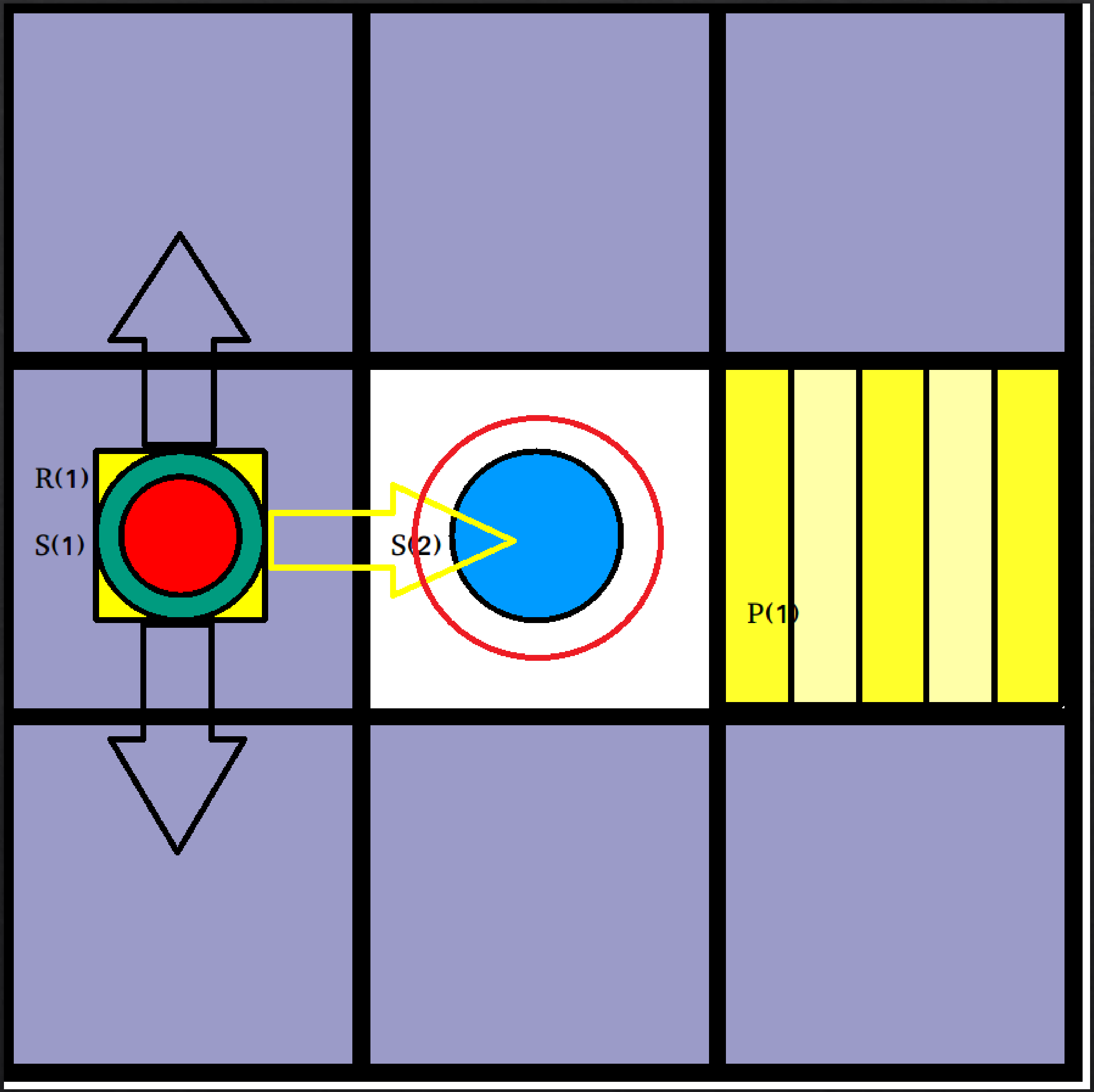
Edge collision

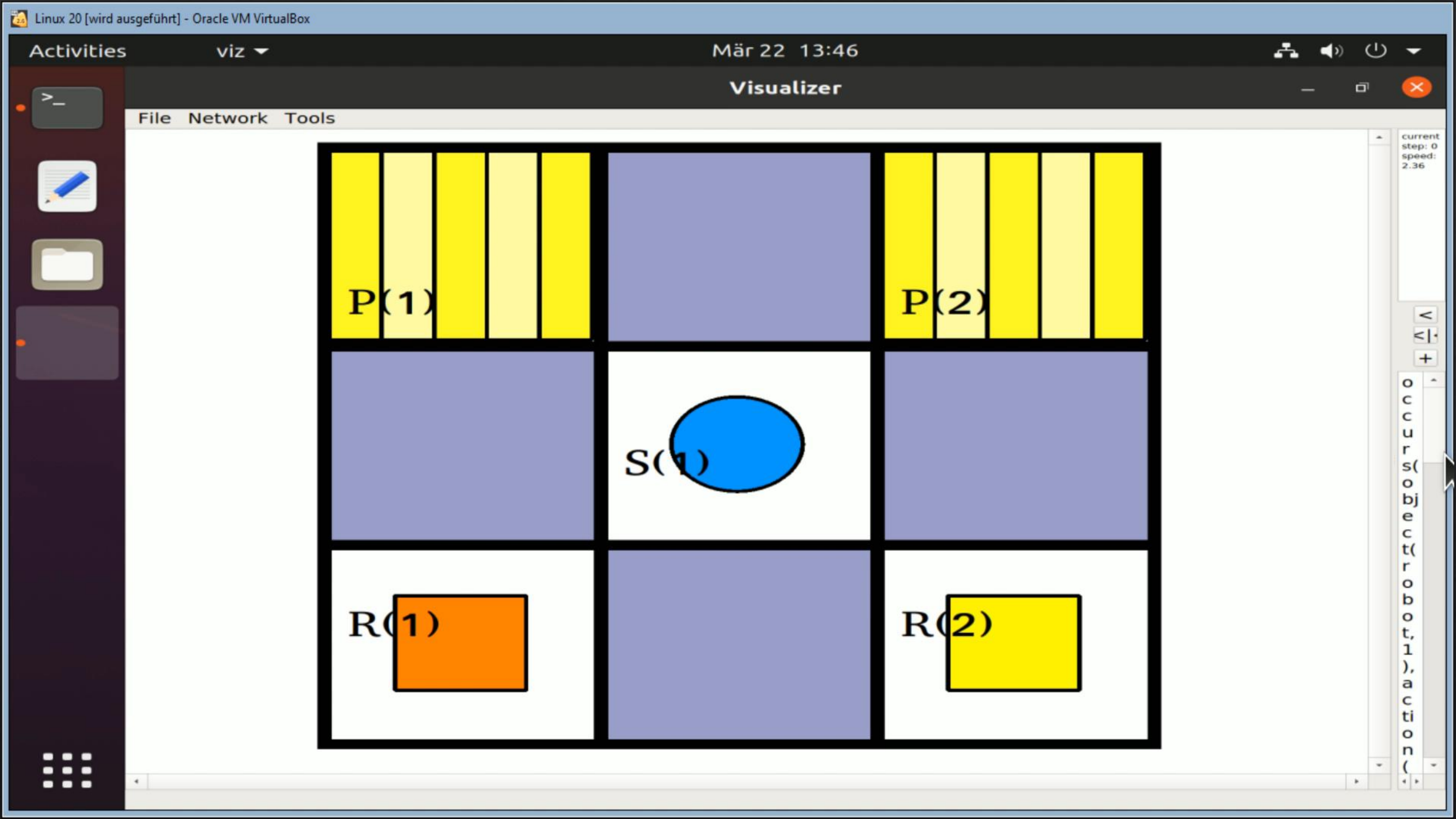
- dodge_coll/3
- dodge_where/3
- dodge_who/3


Linux 20 [wird ausgeführt] - Oracle VM VirtualBox
Activities
viz
Mär 22 11:23
Visualizer
File
Network
Tools
R(
S(2)
R(
S(1)

Details of the Merger

- 2.1 Basics
- 2.2 Use cases
  - 2.2.1 vertex collision
  - 2.2.2 edge collision
  - 2.2.3 issues of the B domain


R(1)
S(1)
S(2)
P(1)



```
current
step: 0
speed:
2.36
```

```
occurs(object(robot,1),action(
```

Results and Comparison

B. Group 1	Group 1 (Adrian)	Group 2 (Marcus)	Group 3 (Niklas)	Group 4 (Tarek)	Group 5 (Tom)	Horizon	#Robots
1	0.027s	0.457s	0.034s	0.058s	0.018s	5	2
2	0.125s	UNSAT	0.017s	UNSAT	0.016s	3	4
3	209.25	2.903s	0.072s	0.065s	0.094s	7-14	2
4	55.426s	KILLED	0.179	UNSAT	UNSAT	9	8

B. Group 2	Group 1 (Adrian)	Group 2 (Marcus)	Group 3 (Niklas)	Group 4 (Tarek)	Group 5 (Tom)	Horizon	#Robots
1	0.032s	0.972s	0.028s	0.029s	UNSAT	5-7	2
2	0.014s	0.105s	0.012s	0.029s	UNSAT	4	2
3	4.949s	6.666s	0.051s	0.041s	0.107s	6-9	4
4	UNSAT	4.578s	UNSAT	UNSAT	UNSAT	15	2

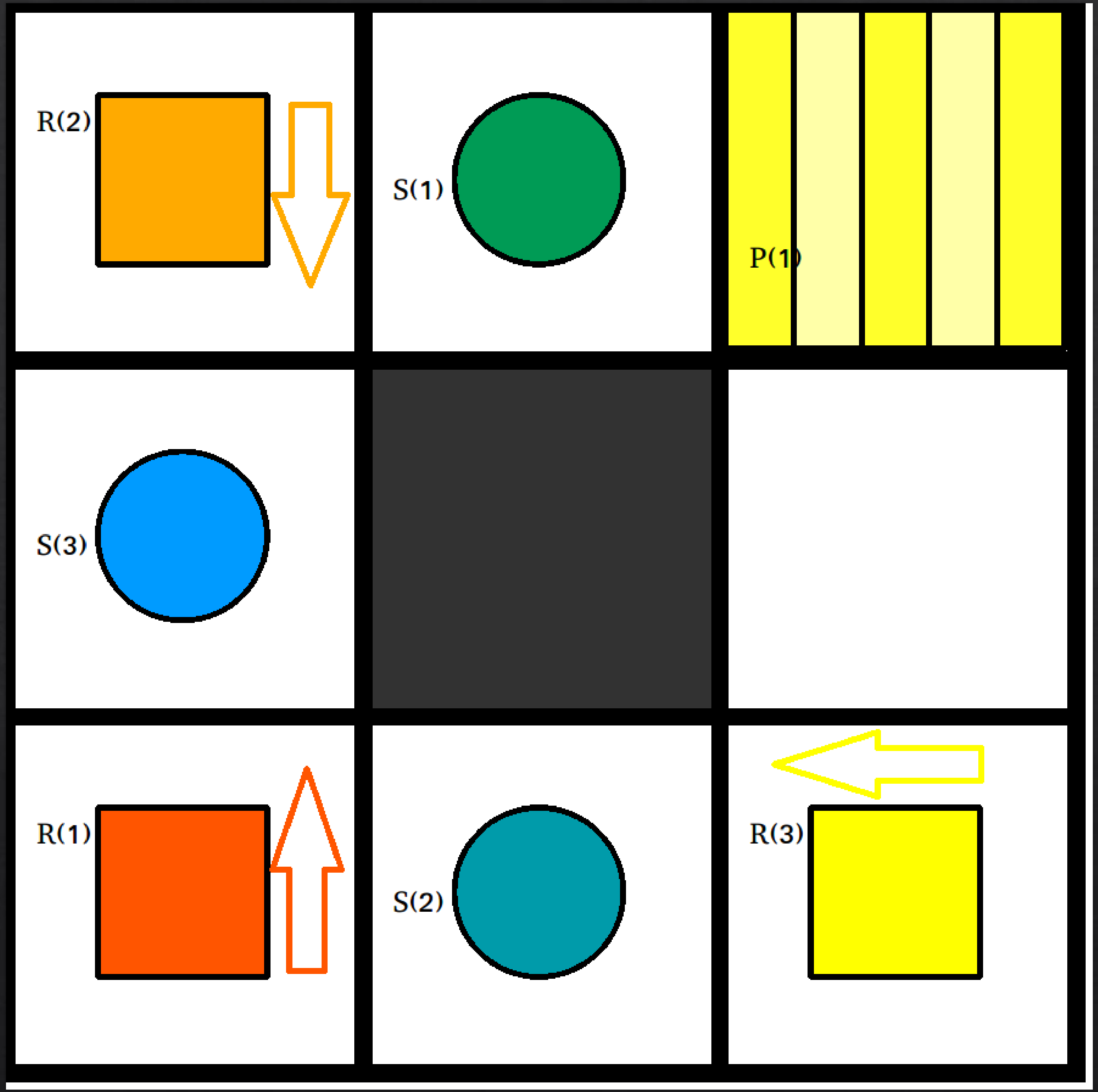
B. Group 3	Group 1 (Adrian)	Group 2 (Marcus)	Group 3 (Niklas)	Group 4 (Tarek)	Group 5 (Tom)	Horizon	#Robots
1	UNSAT	KILLED	0.902s	0.180s	0.309s	12-17	4
2	187.834s	KILLED	0.129s	0.418s	UNSAT	9	8
3	6.442s	KILLED	0.229s	0.450s	UNSAT	10(12)	5
4	419.762s	KILLED	0.229s	3.433s	UNSAT	21(23)	6

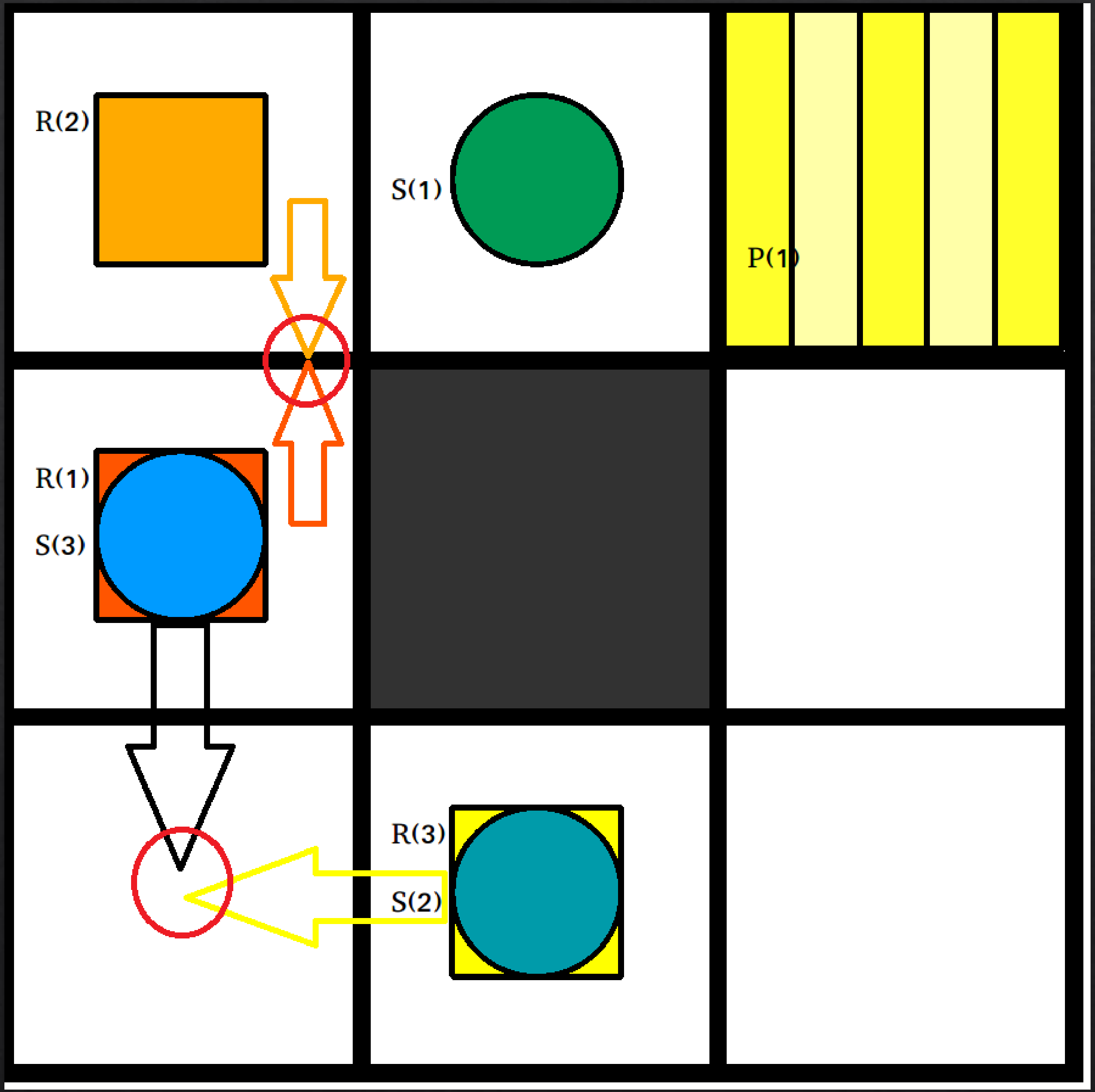
B. Group 4	Group 1 (Adrian)	Group 2 (Marcus)	Group 3 (Niklas)	Group 4 (Tarek)	Group 5 (Tom)	Horizon	#Robots
1	UNSAT	UNSAT	UNSAT	0.037s	UNSAT	5	3
2	KILLED	UNSAT	0.442s	0.352s	UNSAT	19	2
3	0.350s	UNSAT	0.058s	0.191s	0.191s	9(14)	3
4	1.065s	7.669s	0.084s	0.064	0.152s	15-16	2

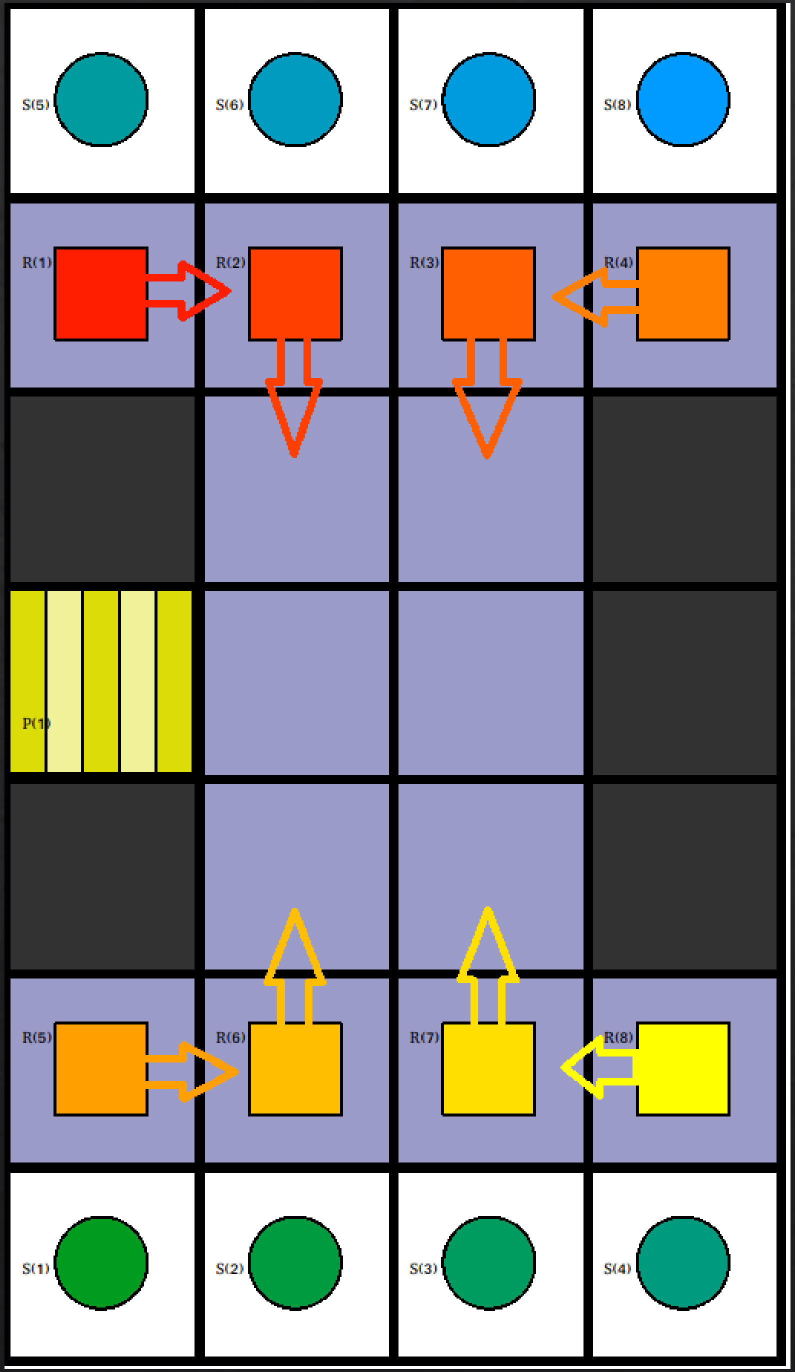
B. Group 5	Group 1 (Adrian)	Group 2 (Marcus)	Group 3 (Niklas)	Group 4 (Tarek)	Group 5 (Tom)	Horizon	#Robots
1	UNSAT	UNSAT	UNSAT	0.047s	0.127s	6-10	4
2	0.070s	3.000s	0.032s	0.024s	0.060s	4	3
3	KILLED	KILLED	KILLED	UNSAT	KILLED	40	50
4	KILLED	KILLED	KILLED	UNSAT	KILLED	100	30

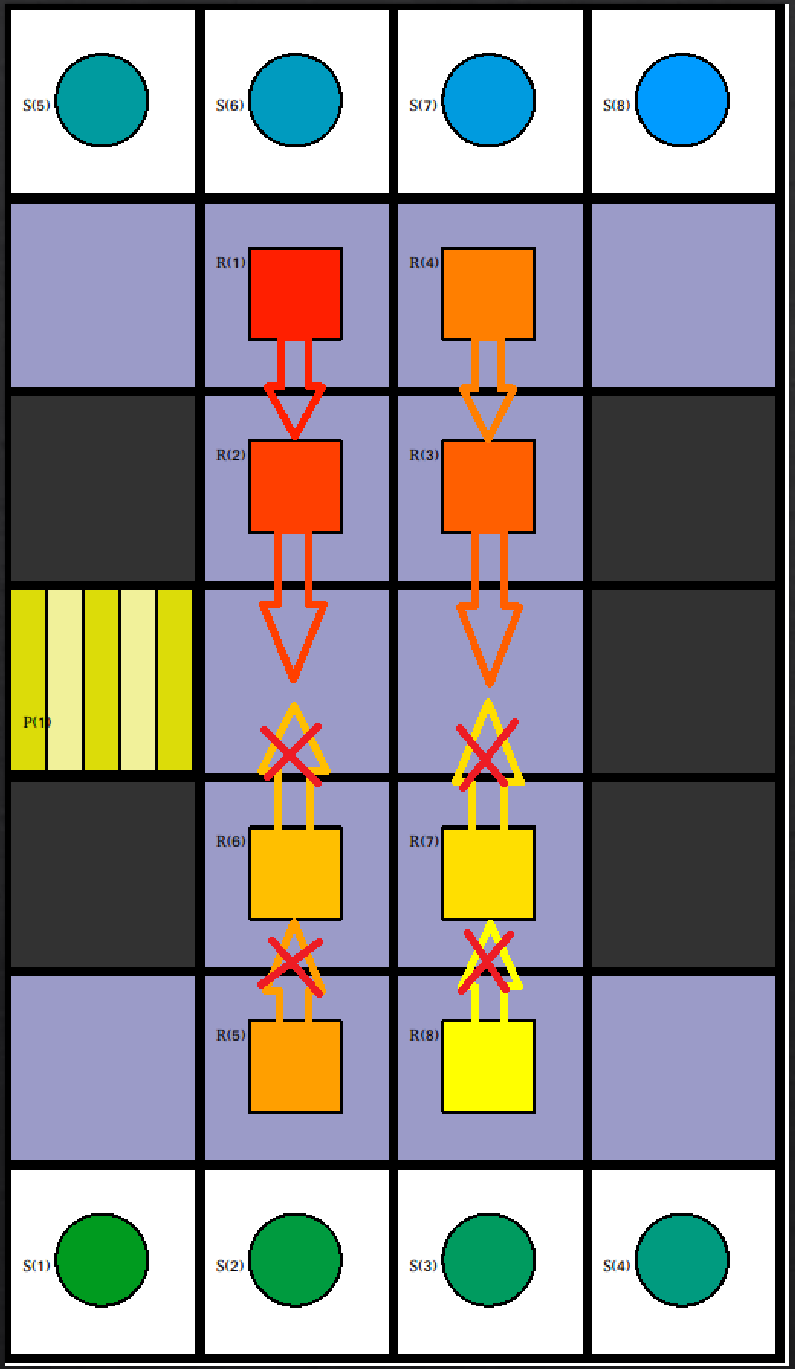
Conclusion

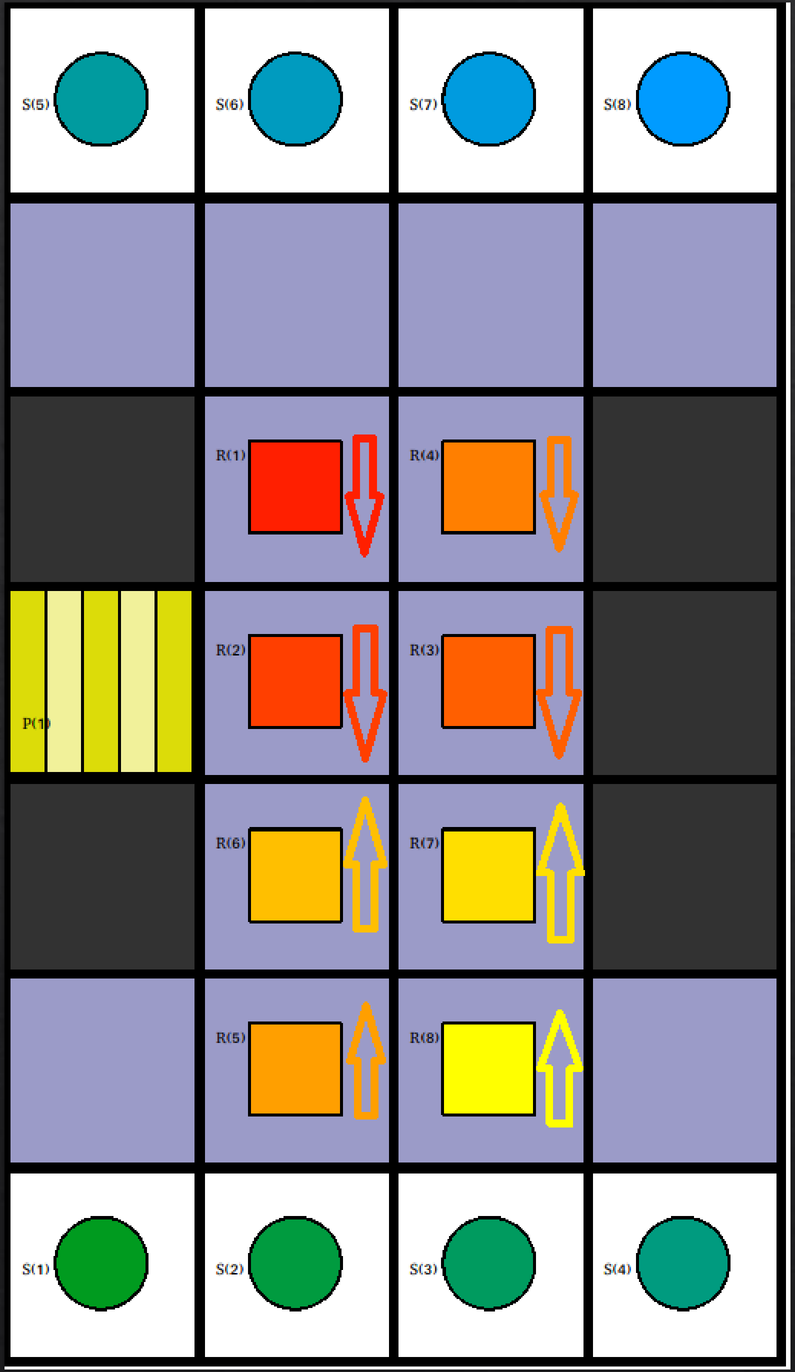
- General
- Special benchmarks and their problems


R(2)
S(1)
P(1)
S(3)
R(1)
S(2)
R(3)


R(2)
S(1)
P(1)
R(1)
S(3)
R(3)
S(2)


S(5)
S(6)
S(7)
S(8)
R(1)
R(2)
R(3)
R(4)
P(1)
R(5)
R(6)
R(7)
R(8)
S(1)
S(2)
S(3)
S(4)


S(5)
S(6)
S(7)
S(8)
R(1)
R(4)
R(2)
R(3)
P(1)
R(6)
R(7)
R(5)
R(8)
S(1)
S(2)
S(3)
S(4)


S(5)
S(6)
S(7)
S(8)
R(1)
R(4)
P(1)
R(2)
R(3)
R(6)
R(7)
R(5)
R(8)
S(1)
S(2)
S(3)
S(4)

Plans going forward

- Fixes for known “bugs”
- shelf collisions
- Adding new features

Questions?

References

- Group 2 (Marcus & Marcus) : https://github.com/Zard0c/ProjektMAPF
- Group 1 (Adrian) : https://github.com/salewsky/Plan-Merging
- Group 3 (Niklas & Marius) : https://github.com/NikKaem/mapf-project
- Group 4 (Tarek) : https://github.com/tramadan-up/asprilo-project
- Group 5 (Tom, Julian, Hannes) : https://github.com/tzschmidt/PlanMerger
- Experimenting with robotic intra-logistics domains : https://arxiv.org/abs/1804.10247
- Picture 1: Experimenting with robotic intra-logistics domains, page 7
- Picture 2-4 : Group 2, benchmark 5
- Picture 5-7 : Group 2, benchmark 6
- Picture 8 : small benchmark made for this presentation
- Picture 9-11: Group 4, shared benchmark 1
- Picture 12-14: Group 3, benchmark 6
- Table 1-5: Group 2 report paper, page 26 – 28
- Video 1: video on benchmark 16_mod2 from group 2
- Video 2: video on benchmark 17 from group 2