Experimental report for COM1005 Assignment 3: The Warrier's Planning Game

Ruiqing Xu

May 29, 2020

1 Description of Strips operators

- In the diagram below: WR = Warrior, LR = Ladder, TC = Treasure chest. There are four StripsApplyOperators and these have four actions in order: move, carry, climb down, lift up. The goal state of no-snake situation is that: Warrior at L, Ladder at P, Treasure Chest at L.

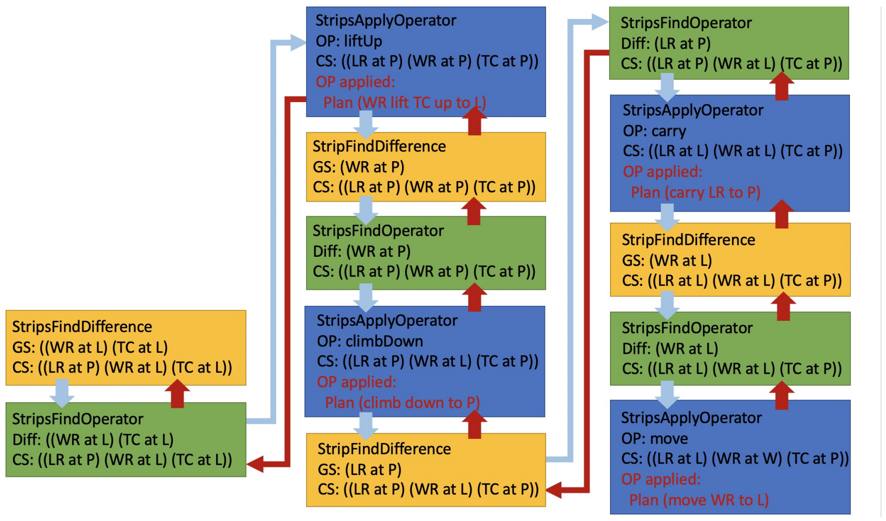

Figure 1: no-snake diagram

- In the diagram below: WR = Warrior, HK = Hook, RP = Rope, TC = Treasure chest. There are five StripsApplyOperators and these have five actions in order: move, carry1, attach, carry2, lift. The goal state of snake situation is that: Warrior at R, Hook at R, Rope at R, Treasure Chest at R.

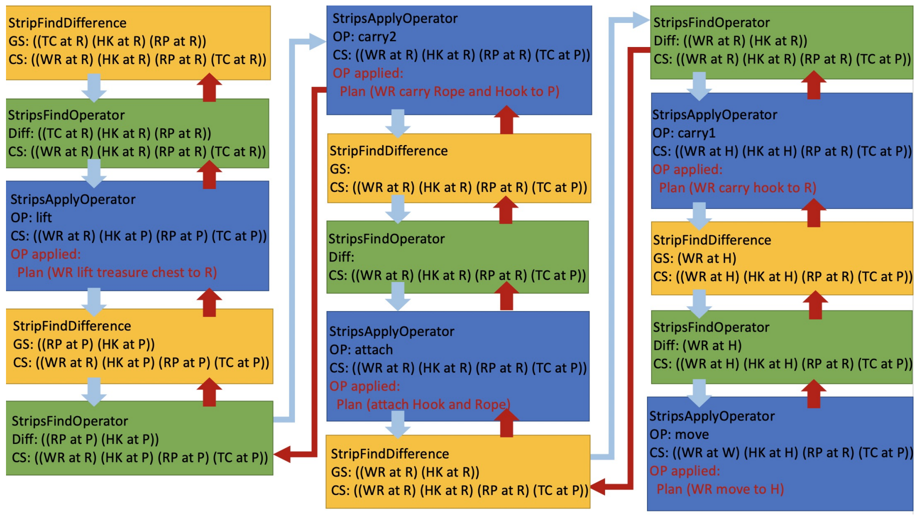
StripFindDifference
GS: ((TC at R) (HK at R) (RP at R))
CS: ((WR at R) (HK at R) (RP at R) (TC at R))
StripsFindOperator
Diff: ((TC at R) (HK at R) (RP at R))
CS: ((WR at R) (HK at R) (RP at R) (TC at R))
StripsApplyOperator
OP: lift
CS: ((WR at R) (HK at P) (RP at P) (TC at P))
OP applied:
Plan (WR lift treasure chest to R)
StripFindDifference
GS: ((RP at P) (HK at P))
CS: ((WR at R) (HK at P) (RP at P) (TC at P))
StripsFindOperator
Diff: ((RP at P) (HK at P))
CS: ((WR at R) (HK at P) (RP at P) (TC at P))
StripsApplyOperator
OP: carry2
CS: ((WR at R) (HK at R) (RP at R) (TC at P))
OP applied:
Plan (WR carry Rope and Hook to P)
StripFindDifference
GS:
CS: ((WR at R) (HK at R) (RP at R) (TC at P))
StripsFindOperator
Diff:
CS: ((WR at R) (HK at R) (RP at R) (TC at P))
StripsApplyOperator
OP: attach
CS: ((WR at R) (HK at R) (RP at R) (TC at P))
OP applied:
Plan (attach Hook and Rope)
StripFindDifference
GS: ((WR at R) (HK at R))
CS: ((WR at R) (HK at R) (RP at R) (TC at P))
StripsFindOperator
Diff: ((WR at R) (HK at R))
CS: ((WR at R) (HK at R) (RP at R) (TC at P))
StripsApplyOperator
OP: carry1
CS: ((WR at H) (HK at H) (RP at R) (TC at P))
OP applied:
Plan (WR carry hook to R)
StripFindDifference
GS: (WR at H)
CS: ((WR at H) (HK at H) (RP at R) (TC at P))
StripsFindOperator
Diff: (WR at H)
CS: ((WR at H) (HK at H) (RP at R) (TC at P))
StripsApplyOperator
OP: move
CS: ((WR at W) (HK at H) (RP at R) (TC at P))
OP applied:
Plan (WR move to H)

Figure 2: snake diagram

1.1 Discussion of snake and no-snake scenarios

- No-snake scenario: Firstly, the warrior moves to the point L to carries the ladder, then lowers it into the pit(point P). After that, He uses the ladder to climb down to get the treasure chest at point T, and finally lifted it up. (see Figure 3)

- Snake scenario: At the beginning, the warrior moves to point H to carry the hook, then he moves to the R with it. After that, he carries the rope at R, then he attaches hook and rope together at R. Finally, he threw hook and rope into the pit(point P), and pulled the treasure chest up. (see Figure 4)

1.2 Description of suggested improvement

- Conflict resolving mechanism: Strips need to be manually distributed in the sequence and assign labels of actions, or the marked actions should be processed in parallel.

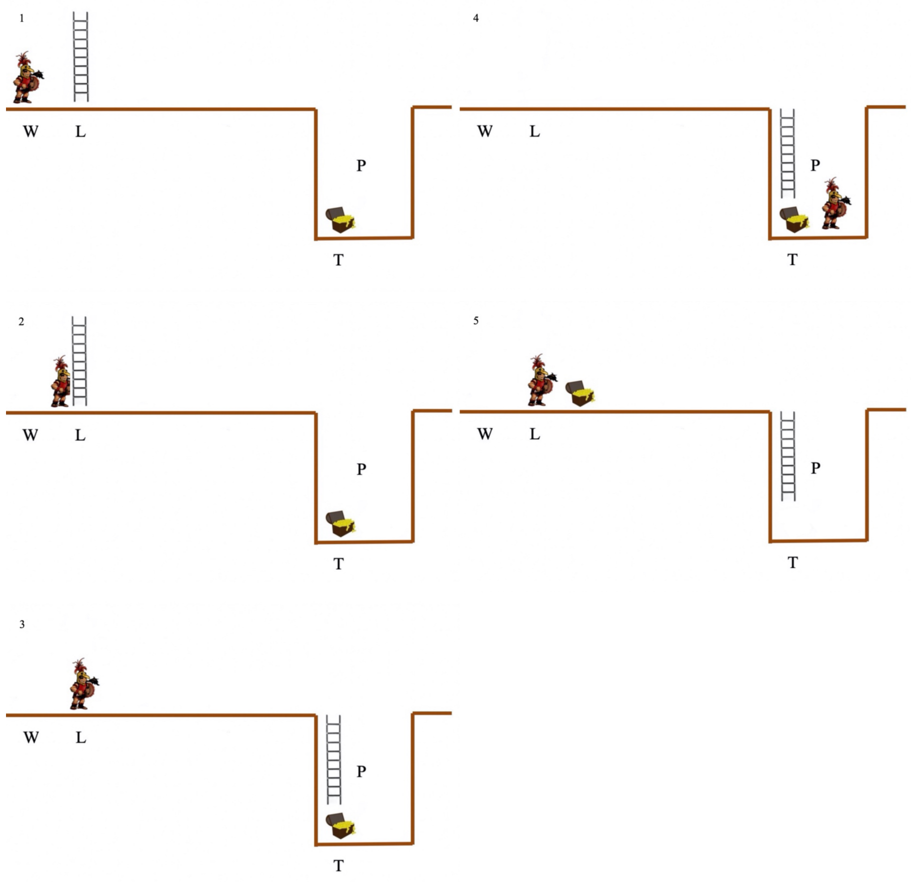

Figure 3: no-snake

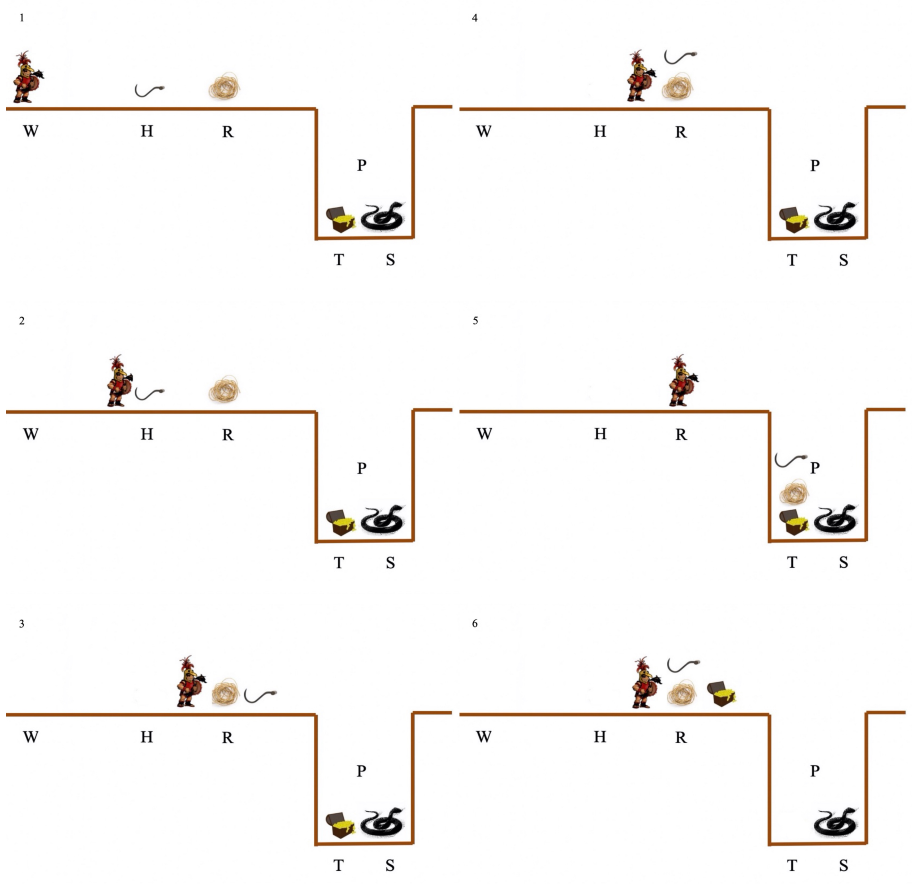

Figure 4: snake

- Strips should use open space instead of close space, so that it can handle lots of problems: First, Strips can handle many parameters and states, even if the states and parameters are really similar, and the system will not break. Second, Strips can judge the order of action, if there are multiple irrelevant actions.

- If Strips still retain close space, in programming, sometimes use invariant to replace variables if the program does not work. It means instead of matching variables, the system is able to find the data one-to-one, which avoids the problem of getting confused by too many variables.

2 Limitation

- There is close space in Strips which are not as easy to operate as open space. For example, Strips cannot handle multiple types of parameters and conditions, and when there are two similar modules that need to operate, Strips cannot judge the order of actions, which means there is no conflict resolution mechanism. In the warrior game also encountered this problem:

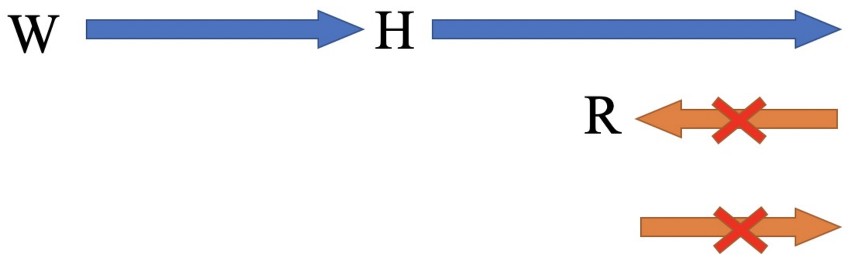

Figure 5: conflict

- when there are too many irrelevant actions in a planning problem, the agent cannot handle them. It is because states are represented simply as a conjunction of positive literals. Also, it is difficult to find a good heuristic function.

- In many cases, Strips cannot use problem decomposition, it means they cannot completely decompose the planning problem into sub-problems. Although in theory, it is difficult to solve these difficulties, but in classical planning theory, in order to avoid these problems, people directly ignore or add some assumptions, because in practical application, problems are often limited to a limited scope, and we only need to use some special techniques to solve these problems.

3 Conclusions

In conclusion, Strips will cause problem when there are too much actions and variables. There are several ways that can improve. First, use open space, Second, do not make the states represent simply as a conjunction of positive literals.

4 Printout of my system's workings

4.1 Case where there is no snake in the pit

```
-----------------
Strips1
currrent state
Warrior at W
Ladder at L
TreasureChest at T
goal state
Ladder at P
Warrior at L
TreasureChest at L
working on goal Ladder at P
------------------
StripsOperatorFinder
working on Ladder at P
calling StripsApplyOperator to apply operator
carry ?obj1 to P
back from Strips3
-------------------
StripsApplyOperator
current state
```

```
Warrior at W
Ladder at L
TreasureChest at T
attempting to use operator
carry ?obj1 to P
in context {?obj1=Ladder}
precond not met-  Warrior at L
calling Strips1 for goal Warrior at L
-----------------
Strips1
currrent state
Warrior at W
Ladder at L
TreasureChest at T
goal state
Warrior at L
working on goal Warrior at L
------------------
StripsOperatorFinder
working on Warrior at L
calling StripsApplyOperator to apply operator
move from W to L
back from Strips3
-------------------
StripsApplyOperator
current state
Warrior at W
Ladder at L
TreasureChest at T
attempting to use operator
move from W to L
in context {}
StripsApplyOperator: Applying op [move from W to L]
[Warrior at W]
[Warrior at L]
New state
Ladder at L
TreasureChest at T
Warrior at L
-----------------
```

```
Strips1
currrent state
Ladder at L
TreasureChest at T
Warrior at L
goal state
Warrior at L
all goals met
-----------------
StripsApplyOperator: Applying op [carry Ladder to P]
[Ladder at L]
[Ladder at P]
New state
TreasureChest at T
Warrior at L
Ladder at P
-----------------
Strips1
currrent state
TreasureChest at T
Warrior at L
Ladder at P
goal state
Ladder at P
Warrior at L
TreasureChest at L
working on goal TreasureChest at L
------------------
StripsOperatorFinder
working on TreasureChest at L
calling StripsApplyOperator to apply operator
lift up from P
back from Strips3
-------------------
StripsApplyOperator
current state
TreasureChest at T
Warrior at L
Ladder at P
attempting to use operator
```

```
lift up from P
in context {?obj2=TreasureChest}
precond not met-  Warrior at P
calling Strips1 for goal Warrior at P
-----------------
Strips1
currrent state
TreasureChest at T
Warrior at L
Ladder at P
goal state
Warrior at P
working on goal Warrior at P
------------------
StripsOperatorFinder
working on Warrior at P
calling StripsApplyOperator to apply operator
carry ?obj1 to P
back from Strips3
-------------------
StripsApplyOperator
current state
TreasureChest at T
Warrior at L
Ladder at P
attempting to use operator
carry ?obj1 to P
in context {?obj1=Warrior}
StripsApplyOperator: Applying op [carry Warrior to P]
[Warrior at L]
[Warrior at P]
New state
TreasureChest at T
Ladder at P
Warrior at P
-----------------
Strips1
currrent state
TreasureChest at T
Ladder at P
```

```
Warrior at P
goal state
Warrior at P
all goals met
-----------------
StripsApplyOperator: Applying op [lift up from P]
[TreasureChest at T, Warrior at P]
[TreasureChest at L, Warrior at L]
New state
Ladder at P
TreasureChest at L
Warrior at L
-----------------
Strips1
currrent state
Ladder at P
TreasureChest at L
Warrior at L
goal state
Ladder at P
Warrior at L
TreasureChest at L
all goals met
-----------------
Result is true
Plan is   [move from W to L, carry Ladder to P,
carry Warrior to P, lift up from P]
```

4.2 Case where there is a snake in the pit

```
-----------------
Strips1
currrent state
Warrior at W
Hook at H
Rope at R
TreasureChest at T
goal state
Warrior at R
Hook at R
```

```
Rope at R
TreasureChest at R
working on goal Warrior at R
------------------
StripsOperatorFinder
working on Warrior at R
calling StripsApplyOperator to apply operator
carry1 Hook from H to R
back from Strips3
-------------------
StripsApplyOperator
current state
Warrior at W
Hook at H
Rope at R
TreasureChest at T
attempting to use operator
carry1 Hook from H to R
in context {}
precond not met-  Warrior at H
calling Strips1 for goal Warrior at H
-----------------
Strips1
currrent state
Warrior at W
Hook at H
Rope at R
TreasureChest at T
goal state
Warrior at H
working on goal Warrior at H
------------------
StripsOperatorFinder
working on Warrior at H
calling StripsApplyOperator to apply operator
move from W to H
back from Strips3
-------------------
StripsApplyOperator
current state
```

```
Warrior at W
Hook at H
Rope at R
TreasureChest at T
attempting to use operator
move from W to H
in context {}
StripsApplyOperator: Applying op [move from W to H]
[Warrior at W]
[Warrior at H]
New state
Hook at H
Rope at R
TreasureChest at T
Warrior at H
-----------------
Strips1
currrent state
Hook at H
Rope at R
TreasureChest at T
Warrior at H
goal state
Warrior at H
all goals met
-----------------
StripsApplyOperator: Applying op [carry1 Hook from H to R]
[Hook at H, Warrior at H]
[Warrior at R, Hook at R]
New state
Rope at R
TreasureChest at T
Warrior at R
Hook at R
-----------------
Strips1
currrent state
Rope at R
TreasureChest at T
Warrior at R
```

```
Hook at R
goal state
Warrior at R
Hook at R
Rope at R
TreasureChest at R
working on goal TreasureChest at R
------------------
StripsOperatorFinder
working on TreasureChest at R
calling StripsApplyOperator to apply operator
carry1 Hook from H to R
back from Strips3
-------------------
StripsApplyOperator
current state
Rope at R
TreasureChest at T
Warrior at R
Hook at R
attempting to use operator
carry1 Hook from H to R
in context {?obj1=TreasureChest}
precond not met-  ?obj1 at H
calling Strips1 for goal TreasureChest at H
-----------------
Strips1
currrent state
Rope at R
TreasureChest at T
Warrior at R
Hook at R
goal state
TreasureChest at H
working on goal TreasureChest at H
------------------
StripsOperatorFinder
working on TreasureChest at H
calling StripsApplyOperator to apply operator
lift TreasureChest to R
```

```
back from Strips3
-------------------
StripsApplyOperator
current state
Rope at R
TreasureChest at T
Warrior at R
Hook at R
attempting to use operator
lift TreasureChest to R
in context {?obj3=TreasureChest}
precond not met-  ?obj2 at P
calling Strips1 for goal ?obj2 at P
-----------------
Strips1
currrent state
Rope at R
TreasureChest at T
Warrior at R
Hook at R
goal state
?obj2 at P
working on goal ?obj2 at P
------------------
StripsOperatorFinder
working on ?obj2 at P
calling StripsApplyOperator to apply operator
carry2 Hook and Rope to P
back from Strips3
-------------------
StripsApplyOperator
current state
Rope at R
TreasureChest at T
Warrior at R
Hook at R
attempting to use operator
carry2 Hook and Rope to P
in context {}
StripsApplyOperator: Applying op [carry2 Hook and Rope to P]
```

```
[Rope at R, Rope at R]
[Rope at P, Rope at P]
New state
TreasureChest at T
Warrior at R
Hook at R
Rope at P
Rope at P
-----------------
Strips1
currrent state
TreasureChest at T
Warrior at R
Hook at R
Rope at P
Rope at P
goal state
?obj2 at P
all goals met
-----------------
precond not met-  ?obj1 at P
calling Strips1 for goal ?obj1 at P
-----------------
Strips1
currrent state
TreasureChest at T
Warrior at R
Hook at R
Rope at P
Rope at P
goal state
?obj1 at P
all goals met
-----------------
StripsApplyOperator: Applying op [lift TreasureChest to R]
[?obj2 at P, ?obj1 at P]
[TreasureChest at R, ?obj2 at R, ?obj1 at R]
New state
TreasureChest at T
Warrior at R
```

```
Hook at R
Rope at P
Rope at P
TreasureChest at R
?obj2 at R
?obj1 at R
-----------------
Strips1
currrent state
TreasureChest at T
Warrior at R
Hook at R
Rope at P
Rope at P
TreasureChest at R
?obj2 at R
?obj1 at R
goal state
Warrior at R
Hook at R
Rope at R
TreasureChest at R
all goals met
-----------------
Result is true
Plan is   [move from W to H, carry1 Hook from H to R,
carry2 Hook and Rope to P, lift TreasureChest to R]
```