

Homework 4: Planning

Due Nov. 2 11:59pm

Please read the rules for assignments on the course web page (<http://www2.cs.duke.edu/courses/fall21/compsci570/>). In particular, you must **write your own code from scratch** for this assignment. Please let us know if you have questions about what you can use. Please use Ed Discussion for questions (use private questions if your question is likely to reveal part of the answer to others). Use Gradescope to turn this in.

1 Introduction

In this brief homework assignment, we will consider another block stacking domain. Unlike the ones considered before, in this domain, a block needs to be stacked on *two* other blocks, and those two blocks must be touching each other. (Or, the block can be put on the table.) Thus, we no longer just consider whether a block is clear; we must consider whether the left side of its top is clear (`left-clear ?block`), and whether the right side of its top is clear (`right-clear ?block`). Furthermore we need to keep track of whether two blocks are adjacent (`touching ?left-block ?right-block`), as well as whether there is currently nothing to the left or right of a block (`nothing-to-left ?block`), (`nothing-to-right ?block`).

Because a block may be touching another block, we can no longer use the same kind of gripper that we used before. Instead, we use one that can grab a block *from the top right*. This means that for us to be able to pick up a block, it needs to have nothing on top of it, *and* there needs to be nothing adjacent to the right of it.

Consider the example in Figure 1. The arm is currently holding block G. It could place it on top of E and F; it could place it on top of C and D, so that F will be touching it; it could place it on the table so D will be touching it; or it could place it on the table in a new stack so that nothing will be touching it. It *cannot* put G next to A, because it has to come in from the top right.

If the arm weren't currently holding G, it could pick up either F or D. It *cannot* pick up E, because it has to come in from the top right. We have to consider a number of different cases, all corresponding to different actions. For picking up a block, it matters whether it is touching a block to the left (in which case we need to update that for the block to the left) or not; it also matters whether it is on top of other blocks (in which case we need to update that for the blocks below) or not. In the former case we add `-touching` to the action and in the latter case `-stacked`. E.g., action `pick-up-from-top-right-touching-stacked` is picking up a block that is on top of other blocks *and* touching another block on the left. The same is true for when we put blocks down.

2 Submission Instructions and Testing

In the domain file (`block-domain.pddl`) you're given, the predicates and some of the actions have already been defined and you need to implement the remaining action schemas:

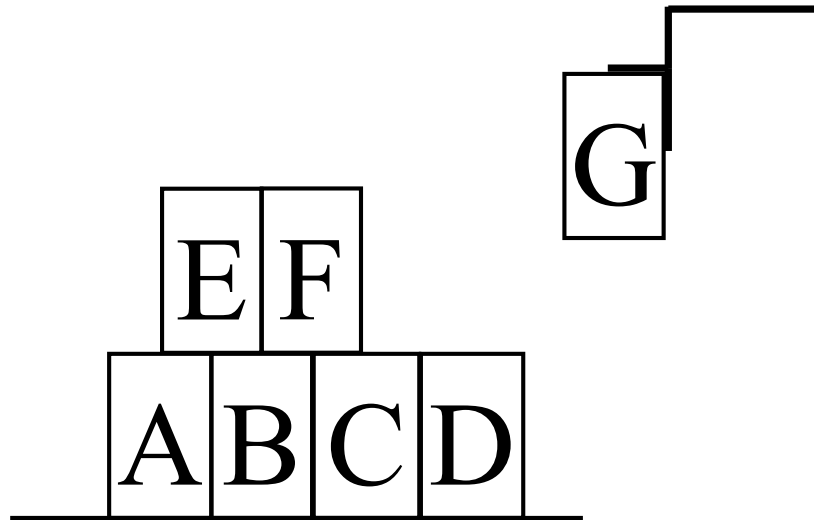


Figure 1: Example Configuration

- take-from-bag
- pick-up-from-top-right-stacked
- pick-up-from-top-right-touching
- pick-up-from-top-right
- put-down-from-top-right-stacked
- put-down-from-top-right-touching
- put-down-from-top-right

You should **not** define new predicates and actions. You should submit `block-domain.pddl` to the Gradescope assignment for this homework, and you should upload it directly without zipping.

You can use the fast-forward solver (`ff`), the executable file included in the helper code, to test your implementation. You need to run `ff` on a Linux OS where you can use editors such as `vim` or `emacs` to modify files. Alternatively (less recommended), you can run an Ubuntu image using [Docker](#) containers on Mac or Windows. Once you download and install Docker, you can run it from the command line:

```
$ docker run -v [/path/to/hw/directory]:/home -t -i ubuntu
$ cd home
$ ./ff -o block-domain.pddl -f block-test-1.pddl
```

You also need to give Docker the permission to access the directory.

```
alperkamil@akb-mac:~$ docker run -v /Users/alperkamil/Downloads/hw4_helper_code:/home -t -i ubuntu
root@802e3707dd5b:/# cd home
root@802e3707dd5b:/home# ./ff -o block-domain.pddl -f block-test-1.pddl

ff: parsing domain file
domain 'STACKING-DOMAIN' defined
... done.
ff: parsing problem file
problem 'BLOCK-TEST-2' defined
... done.
```

Figure 2: Example Execution on Mac

```
root@c14b1d84b70c:/home
Microsoft Windows [Version 10.0.17134.648]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\alper>docker run -v "D:\Downloads\hw4_helper_code":/home -t -i ubuntu
root@c14b1d84b70c:/# cd home
root@c14b1d84b70c:/home# ./ff -o block-domain.pddl -f block-test-1.pddl

ff: parsing domain file
domain 'STACKING-DOMAIN' defined
... done.
ff: parsing problem file
problem 'BLOCK-TEST-1' defined
... done.
```

Figure 3: Example Execution on Windows

We have given you three instance files to test your code with; we have kept others secret for testing purposes. Please note that the planner is *not* guaranteed to give an *optimal* plan, and as you can see it comes up with some pretty strange plans. That is OK.

- block-test-1.pddl

– Reverse the order of blocks in a row.

Initial State:

B1--B2--B3--B4

Goal:

B4--B3--B2--B1

– Plan:

- step 0: PICK-UP-FROM-TOP-RIGHT-TOUCHING B4 B3
- 1: PUT-DOWN-FROM-TOP-RIGHT B4
- 2: PICK-UP-FROM-TOP-RIGHT-TOUCHING B3 B2
- 3: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B3 B4
- 4: PICK-UP-FROM-TOP-RIGHT-TOUCHING B2 B1
- 5: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B2 B3
- 6: PICK-UP-FROM-TOP-RIGHT B1
- 7: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B1 B2

- block-test-2.pddl

– Construct a pyramid using the blocks in the bag.

Goal:

B3

XX--B2

XX--XX--B1

– Plan:

step

- 0: TAKE-FROM-BAG B1
- 1: PUT-DOWN-FROM-TOP-RIGHT B1
- 2: TAKE-FROM-BAG B2
- 3: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B2 B1
- 4: TAKE-FROM-BAG B3
- 5: PUT-DOWN-FROM-TOP-RIGHT B3
- 6: PICK-UP-FROM-TOP-RIGHT-TOUCHING B2 B1
- 7: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B2 B3
- 8: PICK-UP-FROM-TOP-RIGHT B1
- 9: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B1 B2
- 10: TAKE-FROM-BAG B6
- 11: PUT-DOWN-FROM-TOP-RIGHT B6
- 12: PICK-UP-FROM-TOP-RIGHT-TOUCHING B1 B2
- 13: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B1 B6
- 14: PICK-UP-FROM-TOP-RIGHT-TOUCHING B2 B3
- 15: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B2 B1
- 16: PICK-UP-FROM-TOP-RIGHT B3
- 17: PUT-DOWN-FROM-TOP-RIGHT-STACKED B3 B1 B2
- 18: TAKE-FROM-BAG B5
- 19: PUT-DOWN-FROM-TOP-RIGHT B5
- 20: TAKE-FROM-BAG B4
- 21: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B4 B5
- 22: PICK-UP-FROM-TOP-RIGHT-STACKED B3 B1 B2
- 23: PUT-DOWN-FROM-TOP-RIGHT B3
- 24: PICK-UP-FROM-TOP-RIGHT-TOUCHING B2 B1
- 25: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B2 B3
- 26: PICK-UP-FROM-TOP-RIGHT-TOUCHING B1 B6
- 27: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B1 B4
- 28: PICK-UP-FROM-TOP-RIGHT-TOUCHING B2 B3
- 29: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B2 B1
- 30: PICK-UP-FROM-TOP-RIGHT B3
- 31: PUT-DOWN-FROM-TOP-RIGHT-STACKED B3 B1 B2
- 32: PICK-UP-FROM-TOP-RIGHT B6
- 33: PUT-DOWN-FROM-TOP-RIGHT-STACKED B6 B5 B4
- 34: PICK-UP-FROM-TOP-RIGHT-STACKED B3 B1 B2
- 35: PUT-DOWN-FROM-TOP-RIGHT B3
- 36: PICK-UP-FROM-TOP-RIGHT-TOUCHING B2 B1
- 37: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING-STACKED B2 B6 B4 B1
- 38: PICK-UP-FROM-TOP-RIGHT B3
- 39: PUT-DOWN-FROM-TOP-RIGHT-STACKED B3 B6 B2

- block-test-3.pddl

– Destruct a pyramid.

Initial State:

B6

B4--B5

B1--B2--B3

Goal:

B6--B5--B4--B3--B2--B1

– Plan:

step 0: PICK-UP-FROM-TOP-RIGHT-STACKED B6 B4 B5
1: PUT-DOWN-FROM-TOP-RIGHT B6
2: PICK-UP-FROM-TOP-RIGHT-TOUCHING-STACKED B5 B4 B2 B3
3: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B5 B6
4: PICK-UP-FROM-TOP-RIGHT-STACKED B4 B1 B2
5: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B4 B5
6: PICK-UP-FROM-TOP-RIGHT-TOUCHING B3 B2
7: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B3 B4
8: PICK-UP-FROM-TOP-RIGHT-TOUCHING B2 B1
9: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B2 B3
10: PICK-UP-FROM-TOP-RIGHT B1
11: PUT-DOWN-FROM-TOP-RIGHT-TOUCHING B1 B2