Learning Objectives

- To define a classical planning problem with symbols and predicates

- To run BFS search, linear planning, and GraphPlan on the problem

A robot arm (yellow) can pick up and put down blocks to form stacks. It cannot pick up a block that has another block on top of it. It cannot pick up more than one block at a time. Any number of blocks can sit on the table.

Q1. Problem definition

(a) Suppose we have predicates: In-Hand(block), On-Table(block), On-Block(blockOver,blockUnder), Clear(block), and HandEmpty(). Write the instances in this problem.

> **Instances:**
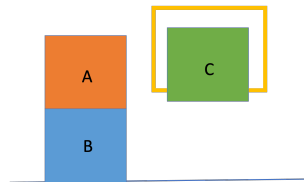
(b) Write the state of this configuration:

Figure 1: Block Configuration 1

> **State:**

(c) If instead we had the predicates: In-Hand(block), On(blockOver,loc), Clear(block), and Hand-Empty(). What would be the instances and state of Block Configuration 1 now?

> **Instances and State:**

Q2. Operators

Use the instances and state from Q1a and Q1b to answer the following questions.

Define the following operators:

Pickup_Table(block):

    Preconditions: HandEmpty() $\wedge$ Clear(block) $\wedge$ On-Table(block)

    Add: In-Hand(block)

    Delete: HandEmpty() $\wedge$ On-Table(block)

Putdown_Table(block):

    Preconditions: In-Hand(block)

    Add: HandEmpty() $\wedge$ On-Table(block)

    Delete: In-Hand(block)

(a) Use the patterns above to write the operators Pickup_fromBlock(block,block) and Putdown_onBlock(block,block)

Pickup_fromBlock(block,fromBlock)

Putdown_onBlock(block,onBlock)

Q3. Planning

Start State →

Goal State →

(a) Write the start state and goal state.

> **Start and Goal states**

(b) Perform BFS by matching operators to the current state, and adding an edge in the search tree for each matching operator. What is the plan found?

> **Plan**

(c) Perform linear planning by solving for the goal in the following stack order (pop the rightmost item first): HandEmpty(), On-Table(A), On-Table(B), On-Block(C, A), Clear(B), Clear(C). What is the plan found?

> **Plan**

(d) What if we tried to achieve our goals in the opposite order (leftmost first): HandEmpty(), On-Table(A), On-Table(B), On-Block(C, A), Clear(B), Clear(C). What is the plan found?

> **Plan**

(e) Draw the first 2 layers of the GraphPlan graph.

Graphplan Graph

Q4. Challenge Problem: Word Puzzle

Here's a puzzle for you. You start with a 4 letter word, and there is a goal 4 letter word. You can take one action change_one_letter which transforms the word to a new valid word (cannot be made up). The goal is to determine what sequence of transformations changes the start word to the goal word the fastest.

For example: Suppose I'm given NECK and the goal is SOAP. I can perform the following actions:

$$NECK - DECK - DOCK - SOCK - SOAK - SOAP$$

You decide to model the problem using classical planning. You're given:

- a dictionary of words,

- a predicate one_change(startword, endword) which returns true if the startword and endword are 1 letter apart and false otherwise.

- a predicate same_word(word1, word2) which returns true if the two words are exactly the same and false otherwise.

Write an initial state and goal state that represents this problem. You might need to create other predicates.

Initial State:

Goal State:

Write an operator that makes use of the information above to find a sequence of words:

Operator: