Names:

Project 2: SATPlan and the Tower of Hanoi

CSCI 561

December 6, 2024

The *Tower of Hanoi* is a classic puzzle consisting of round disks stacked on pegs. One must move all disks to the final peg, subject to the following constraints:

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack.
3. No disk may be placed on top of a smaller disk.

Answer the following questions:

1. Create PDDL domains (operators and facts) for the following Tower of Hanoi instances (it is possible that the PDDL operators will be the same):
   (a) Two pegs and one disk.
   (b) Three pegs and two disks.
   (c) Three pegs and four disks.

   PDDL files shown below define the Tower of Hanoi problem and its domain, outlining the actions, objects, initial states, and goals for solving different instances of the problem. The domain file (domain.pddl) specifies the types of objects (e.g., disks, pegs), predicates to describe the relationships between them (e.g., on, clear, smaller), and two actions: moving a disk to a peg and moving a disk onto another disk. These actions include preconditions to ensure valid moves, such as checking if the destination is clear and if the disk being moved is smaller than the disk it is placed on. The problem files (problem-a.pddl, problem-b.pddl, and problem-c.pddl) define specific instances of the problem with varying numbers of pegs and disks as tasked in this assignment.

Domain File: `domain.pddl`

```
(define (domain hanoi)
  (:requirements :strips :typing)
  (:types
    place
    disk peg - place
  )
  (:predicates
    (on ?disk - disk ?place - place)
    (clear ?place - place)
    (smaller ?d1 - disk ?d2 - disk)
  )

  ;; Action to move a disk onto a peg
  (:action move-to-peg
    :parameters (?disk - disk ?from - place ?to - peg)
    :precondition (and
      (on ?disk ?from)
      (clear ?disk)
      (clear ?to)
      (not (= ?from ?to))
    )
    :effect (and
      (not (on ?disk ?from))
      (on ?disk ?to)
      (clear ?from)
      (not (clear ?to))
      (clear ?disk)
    )
  )

  ;; Action to move a disk onto another disk
  (:action move-to-disk
    :parameters (?disk - disk ?from - place ?to - disk)
    :precondition (and
      (on ?disk ?from)
      (clear ?disk)
      (clear ?to)
      (not (= ?from ?to))
      (smaller ?disk ?to)
    )
    :effect (and
      (not (on ?disk ?from))
      (on ?disk ?to)
      (clear ?from)
      (not (clear ?to))
      (clear ?disk)
    )
  )
)
```

Problem A: Two pegs and one disk

```
(define (problem hanoi-prob-a)
  (:domain hanoi)
  (:objects
    peg1 peg2 - peg
    disk1 - disk
  )
  (:init
    ;; Initial positions
    (on disk1 peg1)
    (clear disk1)
    (clear peg2)
  )
  (:goal
    (on disk1 peg2)
  )
)
```

Problem B: Three pegs and two disks

```
(define (problem hanoi-prob-b)
  (:domain hanoi)
  (:objects
    peg1 peg2 peg3 - peg
    disk1 disk2 - disk
  )
  (:init
    ;; Initial positions
    (on disk1 peg1)
    (on disk2 disk1)
    (clear disk2)
    (clear peg2)
    (clear peg3)

    ;; Size relationships
    (smaller disk2 disk1)
  )
  (:goal
    (and
      (on disk1 peg3)
      (on disk2 disk1)
    )
  )
)
```

Problem C: Three pegs and four disks

```
(define (problem hanoi-prob-c)
  (:domain hanoi)
  (:objects
    peg1 peg2 peg3 - peg
    disk1 disk2 disk3 disk4 - disk
  )
  (:init
    ;; Initial positions
    (on disk1 peg1)
    (on disk2 disk1)
    (on disk3 disk2)
    (on disk4 disk3)

    ;; Clear statuses
    (clear disk4)
    (clear peg2)
    (clear peg3)

    ;; Size relationships
    (smaller disk2 disk1)
    (smaller disk3 disk1)
    (smaller disk3 disk2)
    (smaller disk4 disk1)
    (smaller disk4 disk2)
    (smaller disk4 disk3)
  )
  (:goal
    (and
      (on disk1 peg3)
      (on disk2 disk1)
      (on disk3 disk2)
      (on disk4 disk3)
    )
  )
)
```

2. Download one or more of the following planners and use them to produce plans for your PDDL domains:

   - BlackBox: https://gitlab.com/HenryKautz/BlackBox/
   - Madagascar: http://research.ics.aalto.fi/software/sat/madagascar/
   - TMKit: http://tmkit.dyalab.org/

   What plans are produced by each planner for each instance (one, two, four disks)?

   For this assignment, we used Madagascar with MpC planner to produce plans which we ran on Linux. Figure 1 below shows the output we got for problem a, 1 disk and 2 pegs. Looking at the output, it found plan in 1 action only. The action is to move disk1 from peg1 to peg2 and that is what we expected as well. Time it took to get result is shown as "total time 0.00", which shows that we got results at least on the order of a millisecond.

```
zak@lenovo:~/Desktop/m/MADAGASCAR$ ./MpC -Q -P 0 ./domain.pddl problem-a.pddl
Madagascar 0.99999 25/02/2015 09:45:59 amd64 1-core (no VSIDS)
Options: -P 0 file:./domain.pddl file:problem-a.pddl
Domain: hanoi
Problem: hanoi-prob-a
Parser: 6 ground actions and 6 state variables
Invariants: 0 1  0.00 secs
Goal: conjunctive
Simplified: 2 ground actions and 4 state variables
Actions: STRIPS
Plan type: Sequential
                                Allocated 32 MB permanent (total 121 MB)
Horizon 5: 44 variables
SAT (4 decisions 0 conflicts)
PLAN FOUND: 5 steps
0 : (move-to-peg disk1 peg1 peg2)
1 actions in the plan.
total time 0.00 preprocess 0.00
total size 270.000 MB
max. learned clause length 0
t val conflicts decisions
5 1 0 4
```

Figure 1: Output for problem a, 1 disk and 2 pegs

Figure 2 below shows the output we got for problem b, 2 disks and 3 pegs. Looking at the output, it found plan in 3 action only, which was the expected number of actions. The action are to move disk2 from disk1 to peg2, then move disk1 from peg1 to peg3 and lastly move disk2 from peg2 to disk1 (same as moving to peg3). These actions as we verified manually seems correct. Time it took to get result is again at least on the order of a millisecond.

```
zak@lenovo:~/Desktop/m/MADAGASCAR$ ./MpC -Q -P 0 ./domain.pddl problem-b.pddl
Madagascar 0.99999 25/02/2015 09:45:59 amd64 1-core (no VSIDS)
Options: -P 0 file:./domain.pddl file:problem-b.pddl
Domain: hanoi
Problem: hanoi-prob-b
Parser: 35 ground actions and 16 state variables
Invariants: 0 1 2  0.00 secs
Goal: conjunctive
Simplified: 18 ground actions and 11 state variables
Actions: STRIPS
Plan type: Sequential
                              Allocated 32 MB permanent (total 121 MB)
Horizon 5: 181 variables
SAT (4 decisions 0 conflicts)
PLAN FOUND: 5 steps
0 : (move-to-peg disk2 disk1 peg2)
1 : (move-to-peg disk1 peg1 peg3)
2 : (move-to-disk disk2 peg2 disk1)
3 actions in the plan.
total time 0.00 preprocess 0.00
total size 271.000 MB
max. learned clause length 0
t val conflicts decisions
5 1 0 4
```

Figure 2: Output for problem b, 2 disks and 3 pegs

Figure 3 below shows the output we got for problem c, 4 disks and 3 pegs. Looking at the output, it found plan in 15 actions only, which was the expected minimum number of actions (best case). The actions are shown in the image below numbered from 0 to 14. We manually verified these actions to check it for correctness and they were correct. For reference, we have 2 types of actions, a disk can move to peg or another disk as long as another disk is larger. So "move-to-peg disk4 disk3 peg2" means that we move disk4 from disk3 to peg2. Also initially all disks are on peg1, disk4 is smallest(top), disk1 is largest(bottom) and goal is to have all disks on peg3 in same order. Time it took to get result is 0.01 (s).

```
zak@lenovo:~/Desktop/m/MADAGASCAR$ ./MpC -Q -P 0 ./domain.pddl problem-c.pddl
Madagascar 0.99999 25/02/2015 09:45:59 amd64 1-core (no VSIDS)
Options: -P 0 file:./domain.pddl file:problem-c.pddl
Domain: hanoi
Problem: hanoi-prob-c
Parser: 126 ground actions and 41 state variables
Invariants: 0 1 2  0.00 secs
Goal: conjunctive
Simplified: 68 ground actions and 24 state variables
Actions: STRIPS
Plan type: Sequential
                                Allocated 32 MB permanent (total 122 MB)
Horizon 5: 519 variables
5 UNSAT (0 decisions 0 conflicts)
Horizon 7: 717 variables
                                Allocated 32 MB (total 306 MB)
7 UNSAT (2 decisions 2 conflicts)
Horizon 10: 1014 variables
Horizon 14: 1410 variables
10 UNSAT (83 decisions 70 conflicts)
Horizon 20: 2004 variables
SAT (78 decisions 47 conflicts)
PLAN FOUND: 20 steps
0 : (move-to-peg disk4 disk3 peg2)
1 : (move-to-peg disk3 disk2 peg3)
2 : (move-to-disk disk4 peg2 disk3)
3 : (move-to-peg disk2 disk1 peg2)
4 : (move-to-disk disk4 disk3 disk1)
5 : (move-to-disk disk3 peg3 disk2)
6 : (move-to-disk disk4 disk1 disk3)
7 : (move-to-peg disk1 peg1 peg3)
8 : (move-to-disk disk4 disk3 disk1)
9 : (move-to-peg disk3 disk2 peg1)
10 : (move-to-disk disk4 disk1 disk3)
11 : (move-to-disk disk2 peg2 disk1)
12 : (move-to-peg disk4 disk3 peg2)
13 : (move-to-disk disk3 peg1 disk2)
14 : (move-to-disk disk4 peg2 disk3)
15 actions in the plan.
total time 0.01 preprocess 0.00
total size 609.000 MB
max. learned clause length 263
t val conflicts decisions
5 0 0 0
```

Figure 3: Output for problem c, 4 disk and 3 pegs

3. Encode the three-peg, two-disk instance as a Boolean formula using the SATPlan method. Please separately indicate the different parts of your encoding (start state, goal conditions, frame axioms, etc.).

1 SATPlan Boolean Formula Encoding for Tower of Hanoi

1.1 Variable Definitions

We define two types of Boolean variables for encoding the three-peg, two-disk Tower of Hanoi problem:

1.1.1 State Variables

For each disk $D \in \{D_1, D_2\}$, each peg $P \in \{P_1, P_2, P_3\}$, and each time $t \in \{0, 1, 2, 3\}$, we define:

`On_D_P_t` represents "disk D is on peg P at time t"

For example, `On_D1_P1_0 = True` means "Disk 1 is on Peg 1 at time 0."

1.1.2 Action Variables

For each disk D, distinct pegs $X, Y \in \{P_1, P_2, P_3\}$ $(X \neq Y)$, and time step $t \in \{1, 2, 3\}$, we define:

`Move_D_X_Y_t` represents "move disk D from peg X to Y at time t"

For example, `Move_D1_P1_P2_1 = True` means "At time 1, move Disk 1 from Peg 1 to Peg 2."

1.2 Constraint Categories

1.2.1 Initial State Constraints ($t = 0$)

$$\texttt{On\_D1\_P1\_0} = \texttt{True}$$
$$\texttt{On\_D2\_P1\_0} = \texttt{True}$$
$$\texttt{On\_D1\_P2\_0} = \texttt{False}$$
$$\texttt{On\_D1\_P3\_0} = \texttt{False}$$
$$\texttt{On\_D2\_P2\_0} = \texttt{False}$$
$$\texttt{On\_D2\_P3\_0} = \texttt{False}$$

1.2.2 Goal Conditions ($t = 3$)

$$\texttt{On\_D1\_P3\_3} = \texttt{True}$$
$$\texttt{On\_D2\_P3\_3} = \texttt{True}$$
$$\texttt{On\_D1\_P1\_3} = \texttt{False}$$
$$\texttt{On\_D1\_P2\_3} = \texttt{False}$$
$$\texttt{On\_D2\_P1\_3} = \texttt{False}$$
$$\texttt{On\_D2\_P2\_3} = \texttt{False}$$

1.2.3 Disk Position Uniqueness

For each disk $D \in \{D_1, D_2\}$ and time $t \in \{0, 1, 2, 3\}$:
Each disk must be on exactly one peg:

$$(\texttt{On\_D\_P1\_t} \vee \texttt{On\_D\_P2\_t} \vee \texttt{On\_D\_P3\_t})$$

Disk cannot be on multiple pegs simultaneously:

$$\neg(\texttt{On\_D\_P1\_t} \wedge \texttt{On\_D\_P2\_t})$$
$$\neg(\texttt{On\_D\_P1\_t} \wedge \texttt{On\_D\_P3\_t})$$
$$\neg(\texttt{On\_D\_P2\_t} \wedge \texttt{On\_D\_P3\_t})$$

1.2.4 Move Constraints

For each time step $t \in \{1, 2, 3\}$, possible moves are:

$$\texttt{Move\_D\_P1\_P2\_t}, \texttt{Move\_D\_P1\_P3\_t},$$
$$\texttt{Move\_D\_P2\_P1\_t}, \texttt{Move\_D\_P2\_P3\_t},$$
$$\texttt{Move\_D\_P3\_P1\_t}, \texttt{Move\_D\_P3\_P2\_t}$$

No simultaneous moves allowed:

$$\neg(\texttt{Move\_D\_P1\_P2\_t} \wedge \texttt{Move\_D\_P1\_P3\_t})$$
$$\neg(\texttt{Move\_D\_P1\_P2\_t} \wedge \texttt{Move\_D\_P2\_P1\_t})$$
$$\neg(\texttt{Move\_D\_P1\_P2\_t} \wedge \texttt{Move\_D\_P2\_P3\_t})$$
$$\neg(\texttt{Move\_D\_P1\_P2\_t} \wedge \texttt{Move\_D\_P3\_P1\_t})$$
$$\neg(\texttt{Move\_D\_P1\_P2\_t} \wedge \texttt{Move\_D\_P3\_P2\_t}).....$$

1.2.5 Action Preconditions and Effects

For small disk (D_1):

$$(\neg\texttt{Move\_D\_X\_Y\_t} \vee \texttt{On\_D\_X\_(t-1)})$$

For large disk (D_2):

$$(\neg\texttt{Move\_D2\_X\_Y\_t} \vee \texttt{On\_D2\_X\_(t-1)})$$
$$(\neg\texttt{Move\_D2\_X\_Y\_t} \vee \neg\texttt{On\_D1\_X\_(t-1)})$$

Effects of moves:

$$(\neg\texttt{Move\_D\_X\_Y\_t} \vee \texttt{On\_D\_Y\_t})$$
$$(\neg\texttt{Move\_D\_X\_Y\_t} \vee \neg\texttt{On\_D\_X\_t})$$

1.2.6 Frame Axioms

If a disk doesn't move, it stays in place:

$$(\neg\texttt{On\_D\_P\_(t-1)} \vee \texttt{Move\_D\_P\_Q\_t} \vee \texttt{Move\_D\_P\_R\_t} \vee \texttt{On\_D\_P\_t})$$

4. Find the satisfying assignments for the three-peg, two-disk Boolean formula using your DPLL implementation.
   (a) What is the satisfying assignment?
   (b) What is the corresponding plan?

Problem Statement

Using the SATPlan encoding from Problem 3, find the satisfying assignment for the Boolean variables and determine the corresponding plan that solves the Tower of Hanoi problem with three pegs and two disks.

Solution

1. Satisfying Assignment

Using a SAT solver, the satisfying assignment for the Boolean variables was determined. The assignment specifies the state variables and action variables required to move the disks from the initial state to the goal state.

Initial State (at $t = 0$):

$$\begin{aligned} &\text{On\_D1\_P1\_0} = \text{True}, \quad \text{On\_D2\_P1\_0} = \text{True} \\ &\text{On\_D1\_P2\_0} = \text{False}, \quad \text{On\_D1\_P3\_0} = \text{False} \\ &\text{On\_D2\_P2\_0} = \text{False}, \quad \text{On\_D2\_P3\_0} = \text{False} \end{aligned}$$

Goal State (at $t = 3$):

$$\begin{aligned} &\text{On\_D1\_P3\_3} = \text{True}, \quad \text{On\_D2\_P3\_3} = \text{True} \\ &\text{On\_D1\_P1\_3} = \text{False}, \quad \text{On\_D1\_P2\_3} = \text{False} \\ &\text{On\_D2\_P1\_3} = \text{False}, \quad \text{On\_D2\_P2\_3} = \text{False} \end{aligned}$$

Plan (Actions):

The satisfying assignment includes the following actions:

$$\text{Move\_D1\_P1\_P2\_1} = \text{True}, \quad \text{Move\_D2\_P1\_P3\_2} = \text{True}, \quad \text{Move\_D1\_P2\_P3\_3} = \text{True}$$

2. Corresponding Plan

The sequence of moves extracted from the satisfying assignment is:

(a) **At time** $t = 1$: Move Disk 1 from Peg 1 to Peg 2.
(b) **At time** $t = 2$: Move Disk 2 from Peg 1 to Peg 3.
(c) **At time** $t = 3$: Move Disk 1 from Peg 2 to Peg 3.

3. CNF Representation for the Plan

Using the SAT encoding provided in Problem 3, the corresponding CNF clauses for the actions and state variables are:

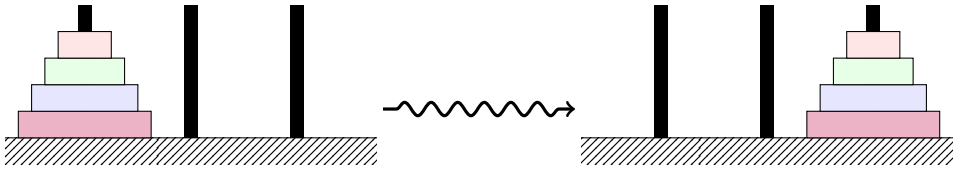

Figure 4: Tower of Hanoi Puzzle with 3 pegs and 4 disks.

Action Variables:

$$\text{Move\_D1\_P1\_P2\_1} \Rightarrow (\text{On\_D1\_P1\_0} \land \text{On\_D1\_P2\_1} \land \neg\text{On\_D1\_P1\_1})$$
$$\text{Move\_D2\_P1\_P3\_2} \Rightarrow (\text{On\_D2\_P1\_1} \land \text{On\_D2\_P3\_2} \land \neg\text{On\_D2\_P1\_2})$$
$$\text{Move\_D1\_P2\_P3\_3} \Rightarrow (\text{On\_D1\_P2\_2} \land \text{On\_D1\_P3\_3} \land \neg\text{On\_D1\_P2\_3})$$

Frame Axioms:

If no move occurs, the state remains unchanged:

$$\neg\text{Move\_D1\_P1\_P2\_t} \land \neg\text{Move\_D1\_P1\_P3\_t} \Rightarrow \text{On\_D1\_P1\_t+1}$$

Uniqueness Constraints:

Each disk can only be on one peg at a time:

$$\text{On\_D1\_P1\_t} \lor \text{On\_D1\_P2\_t} \lor \text{On\_D1\_P3\_t}$$

4. Verification

The plan satisfies all constraints:

- Only one disk moves at a time.
- No smaller disk is placed below a larger disk.
- The goal state is achieved at $t = 3$.

Conclusion

The satisfying assignment determines a valid plan for the Tower of Hanoi problem with two disks and three pegs. The plan adheres to all constraints and achieves the desired goal state.