UNIVERSITÀ DEGLI STUDI
DI GENOVA

Artificial Intelligence for Robotics 2

Assignment 1 Report

Ali Yousefi

Reza Taleshi

BOUAZZA EL MOUTAOUAKIL

Part1: Domain Model Description

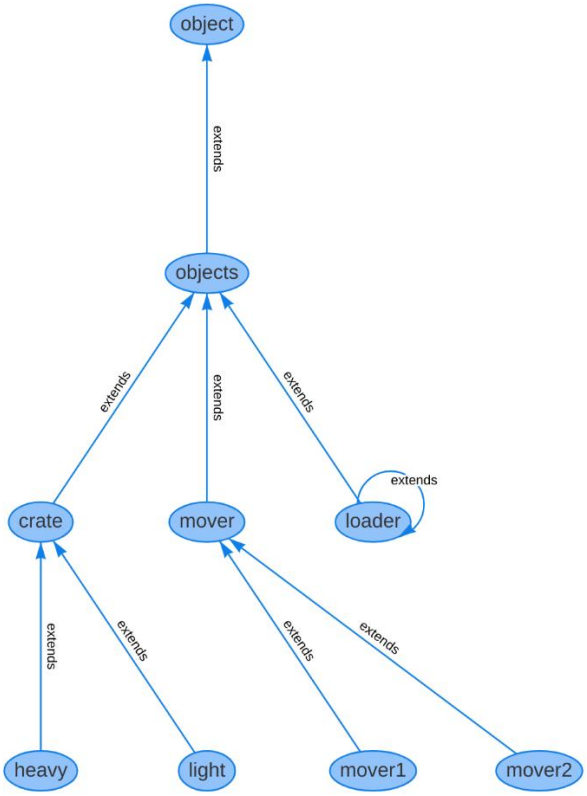
Considering the description of the automated warehouse scenario, we have numeric fluents such as crates weight and their distance from the loading bay, and also durative actions. So, we have to find a solution for this scenario using numeric and temporal planning.

1-1: Adding Requirements: *First step is to add the requirements for this kind of planning:*

```
(define (domain warehouse)
    (:requirements :strips :typing :negative-preconditions :durative-actions :numeric-fluents)
```

1-2: Defining Types: *We can define types for objects of the scenario to distinguish them, you can see the hierarchy of the objects used in this domain file: (generated by vscode)*

```
    (:types
        crate mover loader - objects
        heavy light - crate
        mover1 mover2 - mover
        loader - loader
    )
```



1-3: Defining Predicates: *Same as classical planning, we have to define some predicates to specify the condition of our objects in each state:*

```
(:predicates
    (carry-light-single ?m - mover ?c - light)
    (carry-light-double ?m1 - mover1 ?m2 - mover2 ?c - light)
    (carry-heavy ?m1 ?m2 - mover ?c - heavy)
    (loader-free ?ld - loader)
    (mover-empty ?m - mover)
    (mover-at-loadingBay ?m - mover)
    (mover-near-crate ?m - mover ?c - crate)
    (crate-at-warehouse ?c - crate)
    (crate-at-loadingBay ?c - crate)
    (crate-at-conveyorBelt ?c - crate)
)
```

1-4: Adding Numeric Fluents: *Now we can add numeric fluents into our domain model:*

```
(:functions
    (crate_distance ?c - crate)
    (crate_weight ?c - crate)
)
```

1-5: Durative Actions:

1-5-1: move-to-loadingBay-heavy*: First durative action is for the state when mover1 and mover2 carry a heavy crate to the loading bay together, the required precondition for this action is to both movers carrying a heavy crate and get into the loading bay before the loader is free. It takes a specific time as described in the assignment specification for movers to get to the loading bay and finally the effect of this action is at end both movers will be at the loading bay:*

```
(:durative-action move-to-loadingBay-heavy
    :parameters (?m1 - mover1 ?m2 - mover2 ?c - heavy ?ld - loader)
    :duration (= ?duration (/ (* (crate_distance ?c)(crate_weight ?c)) 100))
    :condition (and (at start (not (mover-at-loadingBay ?m1)))
                    (at start (not (mover-at-loadingBay ?m2)))
                    (over all (carry-heavy ?m1 ?m2 ?c))
                    (at end (loader-free ?ld)))
    :effect (and (at end  (mover-at-loadingBay ?m1))
                 (at end  (mover-at-loadingBay ?m2)))
)
```

1-5-2: move-to-loadingBay-light-single: *Second durative action is same as the first one but in this case we only have one mover carrying a light crate, the time which takes for this action can be derived with the same formula:*

```
(:durative-action move-to-loadingBay-light-single
    :parameters (?m - mover ?c - light ?ld - loader)
    :duration (= ?duration (/ (* (crate_distance ?c)(crate_weight ?c)) 100))
    :condition (and (at start (not (mover-at-loadingBay ?m)))
                    (over all (carry-light-single ?m ?c))
                    (at end (loader-free ?ld)))
    :effect (and (at end  (mover-at-loadingBay ?m)))
)
```

1-5-3: move-to-loadingBay-light-double: *Next durative action is for when two movers carrying a light crate to get to the loading bay faster, so the formula is different from the previous ones.*

```
(:durative-action move-to-loadingBay-light-double
    :parameters (?m1 - mover1 ?m2 - mover2 ?c - light ?ld - loader)
    :duration (= ?duration (/ (* (crate_distance ?c)(crate_weight ?c)) 150))
    :condition (and (at start (not (mover-at-loadingBay ?m1)))
                    (at start (not (mover-at-loadingBay ?m2)))
                    (over all (carry-light-double ?m1 ?m2 ?c))
                    (at end (loader-free ?ld)))
    :effect (and (at end  (mover-at-loadingBay ?m1))
                 (at end  (mover-at-loadingBay ?m2)))
)
```

1-5-4: move-to-crate-light: *This action considers the situation when a mover is free and it is located in the loading bay, so it decides to move toward a light crate to pick it up, after the duration is passed the mover will be near the light crate.*

```
(:durative-action move-to-crate-light
    :parameters (?m - mover ?c - light)
    :duration (= ?duration (/ (crate_distance ?c) 10))
    :condition (and (at start (mover-empty ?m))
                    (at start (mover-at-loadingBay ?m))
                    (at start (not (mover-near-crate ?m ?c)))
                    (over all (crate-at-warehouse ?c)))
    :effect (and (at end (not (mover-at-loadingBay ?m)))
                 (at end (mover-near-crate ?m ?c)))
)
```

1-5-5: move-to-crate-heavy*: This action is like the previous one but in this case both movers decide to move toward a heavy crate to pick it up together.*

```
(:durative-action move-to-crate-heavy
        :parameters (?m1 - mover1 ?m2 - mover2 ?c - heavy)
        :duration (= ?duration (/ (crate_distance ?c) 10))
        :condition (and (at start (mover-empty ?m1))
                        (at start (mover-empty ?m2))
                        (at start (mover-at-loadingBay ?m1))
                        (at start (not (mover-near-crate ?m1 ?c)))
                        (at start (mover-at-loadingBay ?m2))
                        (at start (not (mover-near-crate ?m2 ?c)))
                        (over all (crate-at-warehouse ?c)))
        :effect (and (at end (not (mover-at-loadingBay ?m1)))
                     (at end (mover-near-crate ?m1 ?c))
                     (at end (not (mover-at-loadingBay ?m2)))
                     (at end (mover-near-crate ?m2 ?c)))
    )
```

1-5-6: pick-light-single*: This action considers the situation in which a mover is near a light crate and it is empty, the effect of this action is mover at the end is not empty and carrying the crate, the duration for this action was not mentioned in the assignment description file so we decided it to be 1unit of time:*

```
    (:durative-action pick-light-single
        :parameters (?m - mover ?c - light)
        :duration (= ?duration 1)
        :condition (and (at start (mover-empty ?m))
                        (over all (mover-near-crate ?m ?c))
                        (at start (crate-at-warehouse ?c)))
        :effect (and (at end (not (mover-empty ?m)))
                     (at end (not (crate-at-warehouse ?c)))
                     (at end (carry-light-single ?m ?c)))
    )
```

1-5-7: pick-light-double: *Same as previous action we have a light crate but, in this case, the precondition is both movers have to be near the crate and empty, its effect is both of them won't be empty and carrying the crate:*

```
    (:durative-action pick-light-double
        :parameters (?m1 - mover1 ?m2 - mover2 ?c - light)
        :duration (= ?duration 1)
        :condition (and (at start (mover-empty ?m1))
                        (at start (mover-empty ?m2))
                        (over all (mover-near-crate ?m1 ?c))
                        (over all (mover-near-crate ?m2 ?c))
                        (at start (crate-at-warehouse ?c)))
```

```
    :effect (and (at end (not (mover-empty ?m1)))
                 (at end (not (mover-empty ?m2)))
                 (at end (not (crate-at-warehouse ?c)))
                 (at end (carry-light-double ?m1 ?m2 ?c)))
)
```

1-5-8: drop-light-single: *It consider the situation in which a mover is carrying a light crate and it is located in the loading bay, the effect of this action is the mover will be empty at the end and the crate will be located at the loading bay, the duration for this action was not mentioned in the assignment description file so we decided it to be 1unit of time:*

```
(:durative-action drop-light-single
    :parameters (?m - mover ?c - light)
    :duration (= ?duration 1)
    :condition (and (at start (carry-light-single ?m ?c))
                    (over all (mover-at-loadingBay ?m)))
    :effect (and (at start (not (carry-light-single ?m ?c)))
                    (at end (crate-at-loadingBay ?c))
                    (at end (mover-empty ?m)))
)
```

1-5-9: drop-light-double: *Same as previous action, but in this case both movers are carrying the light crate:*

```
(:durative-action drop-light-double
    :parameters (?m1 - mover1 ?m2 - mover2 ?c - light)
    :duration (= ?duration 1)
    :condition (and (at start (carry-light-double ?m1 ?m2 ?c))
                    (over all (mover-at-loadingBay ?m1))
                    (over all (mover-at-loadingBay ?m2)))
    :effect (and (at start (not (carry-light-double ?m1 ?m2 ?c)))
                    (at end (crate-at-loadingBay ?c))
                    (at end (mover-empty ?m1))
                    (at end (mover-empty ?m2)))
)
```

1-5-10: pick-heavy: *The preconditions for this action is both movers to be near the heavy crate and they are empty, the effect is both movers won't be empty anymore and the crate won't be at the warehouse, the duration for this action was not mentioned in the assignment description file so we decided it to be 1unit of time:*

```
(:durative-action pick-heavy
    :parameters (?m1 - mover1 ?m2 - mover2 ?c - heavy)
    :duration (= ?duration 1)
    :condition (and (at start (crate-at-warehouse ?c))
                    (over all (mover-near-crate ?m1 ?c))
                    (over all (mover-near-crate ?m2 ?c))
```

```
                    (at start (mover-empty ?m1))
                    (at start (mover-empty ?m2)))
        :effect (and (at end (not (mover-empty ?m1)))
                    (at end (not (mover-empty ?m2)))
                    (at end (carry-heavy ?m1 ?m2 ?c))
                    (at end (not (crate-at-warehouse ?c))))
    )
```

1-5-11: drop-heavy: *It consider the situation when the movers are carrying a heavy crate and they are located in the loading bay, effect of this action is both movers will be empty and heavy crate will be at the loading bay, the duration for this action was not mentioned in the assignment description file so we decided it to be 1unit of time:*

```
    (:durative-action drop-heavy
        :parameters (?m1 - mover1 ?m2 - mover2 ?c - heavy)
        :duration (= ?duration 1)
        :condition (and (at start (carry-heavy ?m1 ?m2 ?c))
                    (over all (mover-at-loadingBay ?m1))
                    (over all (mover-at-loadingBay ?m2)))
        :effect (and (at end (not (carry-heavy ?m1 ?m2 ?c)))
                    (at end (crate-at-loadingBay ?c))
                    (at end (mover-empty ?m1))
                    (at end (mover-empty ?m2)))
    )
```

1-5-12: load: *It consider the situation in which, a crate is located in the loading bay, it takes 4units of time to load a crate, and during the load process, the loader is no longer free:*

```
    (:durative-action load
        :parameters (?ld - loader ?c - crate)
        :duration (= ?duration 4)
        :condition (and (at start (crate-at-loadingBay ?c))
                    (at start (not (crate-at-conveyorBelt ?c))))
        :effect (and (at end (not (crate-at-loadingBay ?c)))
                    (at end (crate-at-conveyorBelt ?c))
                    (at start (not (loader-free ?ld)))
                    (at end (loader-free ?ld)))
    )
)
```

Part2: Analysis of the Planning Engine:

As it was mentioned in the assignment specification file, there are 4 problems, we used LPG to find solutions for these problems.

2-1: Problem1: *In the first problem we have three crates, one of them is heavy and the others are light, it can be detected that in time 0, both movers go toward a light crate, and get near them after 2 time units, and then pick them up which takes 1 unit of time, then only mover1 goes to the loading bay and mover2 waits for mover1 to drop the crate. Then, loader starts loading the first crate on the conveyor belt and in the same time mover2 goes toward the loading bay, so mover2 gets there exactly when the loader is free and drops the second crate on the loading bay, then both movers go toward the heavy crate and bring it to the loading bay and drop it there, 5 unit times after the loader is free, and finally loader puts the last crate on the conveyor belt in time unit of 23.*

```
; Version LPG-td-1.0
; Seed 16782574
; Problem warehouse-prob1.pddl
; Time 0.01
; Search time 0.00
; Parsing time 0.01
; Mutex time 0.00
; MakeSpan 27.00


0.0002:   (MOVE-TO-CRATE-LIGHT MOVER1 LIGHT1) [2.0000]
2.0004:   (PICK-LIGHT-SINGLE MOVER1 LIGHT1) [1.0000]
3.0006:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER1 LIGHT1 LOADER) [4.0000]
7.0008:   (DROP-LIGHT-SINGLE MOVER1 LIGHT1) [1.0000]
8.0010:   (LOAD LOADER LIGHT1) [4.0000]
0.0002:   (MOVE-TO-CRATE-LIGHT MOVER2 LIGHT2) [2.0000]
2.0004:   (PICK-LIGHT-SINGLE MOVER2 LIGHT2) [1.0000]
8.0012:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER2 LIGHT2 LOADER) [4.0000]
12.0014:   (DROP-LIGHT-SINGLE MOVER2 LIGHT2) [1.0000]
13.0016:   (LOAD LOADER LIGHT2) [4.0000]
13.0016:   (MOVE-TO-CRATE-HEAVY MOVER1 MOVER2 HEAVY) [1.0000]
14.0018:   (PICK-HEAVY MOVER1 MOVER2 HEAVY) [1.0000]
15.0020:   (MOVE-TO-LOADINGBAY-HEAVY MOVER1 MOVER2 HEAVY LOADER) [7.0000]
22.0022:   (DROP-HEAVY MOVER1 MOVER2 HEAVY) [1.0000]
23.0024:   (LOAD LOADER HEAVY) [4.0000]
```

2-2: Problem2: *In the second problem, we have four crates, two of them are heavy and the others are light, same as before at time instance 0, both movers go toward a light crate, but in this problem mover2 gets near the crate light2 sooner because it is more near to the loading bay, and then picks it up, but it waits until mover1 brings the crate light1 to the loading bay. After 1 time unit when loader starts loading crate light1, mover2 starts moving toward the loading bay so it can drop crate light2 right after loader is finished with the first one. Then loader starts loading the crate light2 and also both movers go toward the crate heavy1. They bring it to the loading bay and then go for the next heavy one, and loader puts the last one on the conveyor belt.*

```
; Version LPG-td-1.0
; Seed 43171466
; Problem warehouse-prob2.pddl
; Time 0.01
; Search time 0.01
; Parsing time 0.00
; Mutex time 0.00
; MakeSpan 47.00


0.0002:   (MOVE-TO-CRATE-LIGHT MOVER1 LIGHT1) [2.0000]
2.0004:   (PICK-LIGHT-SINGLE MOVER1 LIGHT1) [1.0000]
3.0006:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER1 LIGHT1 LOADER) [4.0000]
7.0008:   (DROP-LIGHT-SINGLE MOVER1 LIGHT1) [1.0000]
8.0010:   (LOAD LOADER LIGHT1) [4.0000]
0.0002:   (MOVE-TO-CRATE-LIGHT MOVER2 LIGHT2) [1.0000]
1.0004:   (PICK-LIGHT-SINGLE MOVER2 LIGHT2) [1.0000]
9.0012:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER2 LIGHT2 LOADER) [3.0000]
12.0014:   (DROP-LIGHT-SINGLE MOVER2 LIGHT2) [1.0000]
13.0016:   (LOAD LOADER LIGHT2) [4.0000]
13.0016:   (MOVE-TO-CRATE-HEAVY MOVER1 MOVER2 HEAVY1) [1.0000]
14.0018:   (PICK-HEAVY MOVER1 MOVER2 HEAVY1) [1.0000]
15.0020:   (MOVE-TO-LOADINGBAY-HEAVY MOVER1 MOVER2 HEAVY1 LOADER) [7.0000]
22.0022:   (DROP-HEAVY MOVER1 MOVER2 HEAVY1) [1.0000]
23.0024:   (LOAD LOADER HEAVY1) [4.0000]
23.0024:   (MOVE-TO-CRATE-HEAVY MOVER1 MOVER2 HEAVY2) [2.0000]
25.0026:   (PICK-HEAVY MOVER1 MOVER2 HEAVY2) [1.0000]
26.0028:   (MOVE-TO-LOADINGBAY-HEAVY MOVER1 MOVER2 HEAVY2 LOADER) [16.0000]
42.0030:   (DROP-HEAVY MOVER1 MOVER2 HEAVY2) [1.0000]
43.0032:   (LOAD LOADER HEAVY2) [4.0000]
```

2-3: Problem3: *In the third problem, we have one light crate and three heavy crates. This caused the planner outputs always to be the same, obviously because there are not to much other options for the movers to chose between them. So, it can be detected from the solution that in time instance 0, mover1 goes toward the light crate to bring it to the loading bay and then loader puts it on the conveyor belt. Then both movers got toward a heavy crate and drop it on the loading bay 14 unit of times after the loader is finished. The next heavy crate gets to the loading bay 16 unit of times after the second crate is loaded and the last one gets there 19 units of time after the third one is loaded, so it takes 67 units of time for the whole process to be finished.*

```
; Version LPG-td-1.0
; Seed 123884967
; Problem warehouse-prob3.pddl
; Time 0.01
; Search time 0.00
; Parsing time 0.01
; Mutex time 0.00
; MakeSpan 71.00


0.0002:   (MOVE-TO-CRATE-LIGHT MOVER1 LIGHT) [1.0000]
1.0004:   (PICK-LIGHT-SINGLE MOVER1 LIGHT) [1.0000]
2.0006:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER1 LIGHT LOADER) [3.0000]
5.0008:   (DROP-LIGHT-SINGLE MOVER1 LIGHT) [1.0000]
6.0010:   (LOAD LOADER LIGHT) [4.0000]
6.0010:   (MOVE-TO-CRATE-HEAVY MOVER1 MOVER2 HEAVY1) [2.0000]
8.0012:   (PICK-HEAVY MOVER1 MOVER2 HEAVY1) [1.0000]
9.0014:   (MOVE-TO-LOADINGBAY-HEAVY MOVER1 MOVER2 HEAVY1 LOADER) [14.0000]
23.0016:   (DROP-HEAVY MOVER1 MOVER2 HEAVY1) [1.0000]
24.0018:   (LOAD LOADER HEAVY1) [4.0000]
24.0018:   (MOVE-TO-CRATE-HEAVY MOVER1 MOVER2 HEAVY2) [2.0000]
26.0020:   (PICK-HEAVY MOVER1 MOVER2 HEAVY2) [1.0000]
27.0022:   (MOVE-TO-LOADINGBAY-HEAVY MOVER1 MOVER2 HEAVY2 LOADER) [16.0000]
43.0024:   (DROP-HEAVY MOVER1 MOVER2 HEAVY2) [1.0000]
44.0026:   (LOAD LOADER HEAVY2) [4.0000]
44.0026:   (MOVE-TO-CRATE-HEAVY MOVER1 MOVER2 HEAVY3) [3.0000]
47.0028:   (PICK-HEAVY MOVER1 MOVER2 HEAVY3) [1.0000]
48.0030:   (MOVE-TO-LOADINGBAY-HEAVY MOVER1 MOVER2 HEAVY3 LOADER) [18.0000]
66.0032:   (DROP-HEAVY MOVER1 MOVER2 HEAVY3) [1.0000]
67.0034:   (LOAD LOADER HEAVY3) [4.0000]
```

2-4: Problem4: *In the last problem we have 6 light crates, so this made the planner to pars different solutions for the problem file because there are more options for the movers to pick between the crates and work individually. As you can see in the best plan provided below, in the time instance 0, both movers go toward a crate and get there in the same time, so both of them pick a crate and one of them goes back to the loading bay. Once the loader starts working, mover1 goes toward another crate and also, mover2 goes toward the loading bay and gets there right after the loader is finished. Mover1 has been waiting for the loader to finish working on the second crate and starts moving to the loading bay and gets there right after the loader is finished. The whole process takes places as it was like in these steps, so it can be concluded the best solution is the one in which the loader waits less and the crates are provided right after it has finished.*

```
; Version LPG-td-1.0
; Seed 46487134
; Problem warehouse-prob4.pddl
; Time 0.00
; Search time 0.00
; Parsing time 0.00
; Mutex time 0.00
; MakeSpan 42.00


0.0002:   (MOVE-TO-CRATE-LIGHT MOVER2 LIGHT1) [2.0000]
2.0004:   (PICK-LIGHT-SINGLE MOVER2 LIGHT1) [1.0000]
3.0006:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER2 LIGHT1 LOADER) [6.0000]
9.0008:   (DROP-LIGHT-SINGLE MOVER2 LIGHT1) [1.0000]
10.0010:   (LOAD LOADER LIGHT1) [4.0000]
0.0002:   (MOVE-TO-CRATE-LIGHT MOVER1 LIGHT2) [2.0000]
2.0004:   (PICK-LIGHT-SINGLE MOVER1 LIGHT2) [1.0000]
10.0012:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER1 LIGHT2 LOADER) [4.0000]
14.0014:   (DROP-LIGHT-SINGLE MOVER1 LIGHT2) [1.0000]
15.0016:   (LOAD LOADER LIGHT2) [4.0000]
10.0010:   (MOVE-TO-CRATE-LIGHT MOVER2 LIGHT3) [1.0000]
11.0012:   (PICK-LIGHT-SINGLE MOVER2 LIGHT3) [1.0000]
16.0018:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER2 LIGHT3 LOADER) [3.0000]
19.0020:   (DROP-LIGHT-SINGLE MOVER2 LIGHT3) [1.0000]
20.0022:   (LOAD LOADER LIGHT3) [4.0000]
20.0022:   (MOVE-TO-CRATE-LIGHT MOVER2 LIGHT4) [2.0000]
22.0024:   (PICK-LIGHT-SINGLE MOVER2 LIGHT4) [1.0000]
23.0026:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER2 LIGHT4 LOADER) [4.0000]
27.0028:   (DROP-LIGHT-SINGLE MOVER2 LIGHT4) [1.0000]
28.0030:   (LOAD LOADER LIGHT4) [4.0000]
15.0016:   (MOVE-TO-CRATE-LIGHT MOVER1 LIGHT5) [3.0000]
18.0018:   (PICK-LIGHT-SINGLE MOVER1 LIGHT5) [1.0000]
23.0032:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER1 LIGHT5 LOADER) [9.0000]
32.0034:   (DROP-LIGHT-SINGLE MOVER1 LIGHT5) [1.0000]
33.0036:   (LOAD LOADER LIGHT5) [4.0000]
33.0036:   (MOVE-TO-CRATE-LIGHT MOVER1 LIGHT6) [1.0000]
34.0038:   (PICK-LIGHT-SINGLE MOVER1 LIGHT6) [1.0000]
35.0040:   (MOVE-TO-LOADINGBAY-LIGHT-SINGLE MOVER1 LIGHT6 LOADER) [2.0000]
37.0042:   (DROP-LIGHT-SINGLE MOVER1 LIGHT6) [1.0000]
38.0044:   (LOAD LOADER LIGHT6) [4.0000]
```