ARTIFICIAL INTELLIGENCE FOR ROBOTICS II

ASSIGNMENT 1

AI Planning

Group I

Ankur Kohli (5160903),
Awais Tahir (5174335),
Subhransu Sourav Priyadarshan (5302091)

Submitted to
Prof. Mauro Vallati

Preface

The work is based on AI Planning which involves the demonstration of an automated warehouse for a company. An automated warehouse helps in order management in a precise way which automatically boost the storage capacity.

Acknowledgements

We take this opportunity to express our sincere thanks to Prof. Mauro Vallati for their guidance and support. We would also like to express our gratitude towards him for showing confidence in us. It was a privilege to have a great experience working under him in a cordial environment. We are very much thankful to the University of Genoa, for providing us the opportunity of pursue M.Sc in Robotics Engineering in a peaceful environment with ample resources. In the end, we would like to acknowledge our parents, friends. Without their support, this work would not have been possible.

Abstract

This assignment is about AI Planning in which the task is to model an automated warehouse scenario in an efficient manner. We presented with some techniques for handling planning problem throughout the course such as PDDL+/ Numeric planning, Numeric and Temporal planning, and so on. For this assignment, we used Numeric and Temporal planning using LPG 1.2 Planner. A planner is used to design an application domain and problem (defines the initial state and goal state), and also to generates a plan.

Contents

1 Introduction 5

2 Related Work 6

3 Methodology 7

4 Results 12

5 Conclusion 15

6 References 16

1 Introduction

The task is to make sure that the crates are loaded and unloaded with the help of three robots (1 "loader robot" and 2 "mover robot"). Mover robots will carry the crates from warehouse and drop at the loading bay. At the end loader robot will place the crates on the conveyor belt. We also have to keep in mind that the weight of the crates is not the same. Some are light-weight, some are fragile while others are heavy-weight. Light crates, (<50kg) can be moved around by a single mover robot while heavy crates (>50kg) need two robots for obvious reasons. Also, light crates can be moved by two mover robots. The time required to move the crate depends on the weight. Hence the formula is used for the time calculation of light crates using single mover robot distance * weight / 100 (for light crates). For heavy crates the time calculation is as same of light crates. As mentioned above light crates can also be carried by two mover robots for this case the time calculation formula will change and calculated by using distance * weight / 150 (for heavy crates). There are also fragile crates. Fragile crates can be heavy or light it's depend. These fragile crates need extra care to carry from the warehouse to the loading bay. But fragile crates will be carried by two mover robots whether the weight of a fragile crates is less than 50 Kg or more. Also, the time calculation formula for these crates will be same as of heavy and light crates. The only change in the speed of the mover robot will effect such as if the mover robot takes 4 times unit to carry the other type of crates and it will take 6 time units instead of 4 time units to carry fragile ones. When there are no crates, mover robots cover 10 distance units per time unit. Initially, the mover robots are both at the loading bay. It takes the loader robot 4 time units to load a crate on the conveyor belt, and it can load a single crate at a time. Some of the prerequisites are that the loading bay must be kept free while the loader is in the process of loading a crate on the belt. In layman's language, no crates can be put on the loading bay during the 4 time units as aforementioned.

2 Related Work

Fox M. at al discussed the the syntax and modelling style of pddl+, showing that the language makes the modelling of complex time-dependent effects an effective one [1].

Cushing, W. demonstrated by way of compilation methods and analysis of algorithmic properties such as completeness that the more immediately pressing computational obstacles can be dealt with in a more efficient manner [2].

Li H. at al discussed that the Autonomous underwater vehicles (AUV) paradigm, however, requires robust operation that is cost effective and responsive to the environment and its major challenge is to sense remotely [3].

Penberthy S. at al present ZENO, a least commitment planner that handles actions occurring over extended in- tervals of time. Deadline goals, metric precon- ditions, metric effects, and continuous change are supported [4].

Vallati M. at al discussed about the ParLPG planning system is based on the idea of using a generic algorithm configuration procedure – here, the well-known ParamILS framework – to optimise the performance of a highly parametric planner on a set of problem instances representative of a specific planning domain [5].

Long D. at al analyzed the resultsfrom several perspectives, in order to address the questions of comparative performance between planners, comparative difficulty of domains, the degree of agreement between planners about the relative difficulty of individual problem instances and the question of how well planners scale relative to one another over increasingly difficult problems [6].

McDermott at al developed a first-order temporal logic in which it is possible to prove and name things about facts and events.To be specific, the logic provides analyses of casualty [7].

Smith D. at al discusse on the paper about TGP, a new algorithm for temporal planning. TGP operates by incrementally expanding a compact planning graph representation that handles actions of differing duration [8].

Vila L. discussed the Temporal Reasoning. Then, the most important representational issues which determine a Temporal Reasoning approach are introduced: the logical form on which the approach is based, the ontology and the concepts related with reasoning about action [9].

Bacchus F. at al discussed how domain-dependent search control knowledge can be represented in a temporal logic, and then utilized to effectively control a forward-chaining planner[10].

3 Methodology

This section will describes the methodology and the approach used for this assignment to accomplish a goal. We produces a single PDDL domain file and the corresponding planning problem files. We used Numeric and temporal Planning technique to achieve the goal. In each case we are given with the certain number of crates such as heavy, light, and fragile. We are also given that the crates are distinguish between groups for extension 1. In detail, the groups (A, B, and so on) are to indicate one of the optimal extensions that we need into consideration while planning the model. Crates from the same group have to be loaded one after the other and vice versa.

To design a plan we used a LPG Planner. LPG (Local search for Planning Graphs) ia a planner which is based on local search and planning graphs. These will grasps PDDL 2.1 domains. This PDDL 2.1 domains associates with numerical quantities and durations. This planner able to solve not only plan generation but also plan adaptation problems. The evaluation function uses some heuristics to estimate the "search cost" and the "execution cost" of achieving a (possibly numeric) precondition. Action durations and numerical quantities (e.g., fuel consumption) are represented in the actions graphs, and are modeled in the evaluation function. In temporal domains, actions are ordered using a "precedence graph" that is maintained during search, and that takes into account the mutex relations of the planning graph. The system can produce good quality plans in terms of one or more criteria. This is achieved by an anytime process producing a sequence of plans, each of which is an improvement of the previous ones in terms of its quality. LPG is integrated with a best-first algorithm similar to the one used by FF. The system can automatically switch to best-first search after a certain number of search steps and "restarts" have been performed. Finally, LPG can be used as a pre-processor to produce a quasi-solution that is then repaired by ADJ, a plan-analysis technique for fast plan-adaptation (*source from: https://lpg.unibs.it/lpg/*).

Next, we describes the syntactic foundations of our code. For this, we created single domain file and comparable four problem files. A "domain file" in PDDL depicts the "universal" features of a problem. These universal features will not interchange anyway of what the case we are trying to solve. In domain file, there are mainly requirements, object types, predicates, functions, and actions which can exist within the model design. General structure of a domain is as follows:

```
  (define (domain domainname)
 (:requirements [:strips] [:typing] [:adl])
 (:predicates (predicatename11 [?X1 ?X2 ... ?Xn])
 (predicatename2 [?Y1 ?Y2 ... ?Yn])
 ...)
 (:action actionname

 [:parameters (?L1 ?L2 ... ?Ln)]
 [:precondition preconditionformula]
 [:effect effectformula]
)
...)
```

The "problem file" in PDDL describes the other part of the planning. The problem consists of objects, init (initial stage), and goal. General structure of problem file is as follows:

```
  (define (problem porblemname)
 (:domain domainname)
 (:objects O1 O2 ..... On)
 (:init I1 I2 ..... In)
 (:goal conditionformula)
 )
```

In our assignment, we worked on LPG Planner which is based on PDDL 2.1 and PDDL 2.1 was language of the 2002 AIPS Competition and construct on the syntax defined for PDDL 1.2 (*source from: https://planning.wiki/ref/pddl21*).

Domain file in PDDL 2.1, is the extended version of PDDL 1.2 which contains two new attributes such as durative-actions and functions. These two new features are raise to numeric fluent. General structure of domain in PDDL 2.1 is as follows:

```
  (define (domain domain-name)
 (:requirements :durative-actions :fluents :duration-inequalities)
 (:types t1 t2 .....tn)
 (:predicates
 ...
 ) v (:functions
 (functions - f1)
 (functions- f2)
 .....
 (functions -fn) v )
 (:durative-action actionname
 :parameters (argument)
 :duration (= ?duration durationnumber)
 :condition (expression of condition)
 :effect (expression of effect)
 )
 )
```

Problem file in PDDL 2.1, is the expanded in conductive to complement the syntax of the domain file. In PDDL 2.1, metric is the additional features is added to the problem file. Also, metric feature acts like an optimal function role which depicts the cost value for a designed plan.

```
  (define
 (problem problem-name)
 (:domain domain-name)
 (:objects
 O1 O2 .....On - objects
 )
 (:init
 (initial state)
 ...
 ) v (:goal (goal state
 ...
 )
 )
 (:metric (numericexpreation)
 )
 )
 )
```

In this part we will describe about a description of the meaning of each component of the model used in our code.

Domain File: It comprises of

(i) Requirements: Added the requirements for numeric and temporal planning.

(:requirements :strips :typing :negative-preconditions :durative-actions :numeric-fluents)

(ii)Types: Defined the types of objects to differentiate them such as crates, mover, and loader. Also, distinguish among the crates like heavy, light, and fragile.

(iii)Predicates: This will highlights predicates based on our task such as:

(a) carrying-lightcrate/heavycrate/fragilecrate is used to crate the carates on mover top.
(b) idleloader is used to check the state of loader robot.
(c) empty-top-mover is also used to check the mover robot head is empty or not.
(d) mover-placed-loading is the predicate which defines the mover robots location at the loading bay.
(e) mover-at-crate is the another predicate to check mover robots is at the crate position or not.
(f) crate-placed-loadingbay/convyrbelt these predicates are used to drop the crates at the loading bay and placed the crates on the conveyor belt.

(iv)Functions: This feature is used to the numeric fluents fro the crates such as distance of crate (dis-of-crate) and weight of crate (wt-ofcrate).

(v)Durative-Actions: This will defines the actions taken by the robots throughout the plan such as:

(a) carry-heavycrate/light/fragile-towards-bay; these actions define mover robots are carrying heavy, light, and fragile crates towards loading bay.

(b) mover-towards-heavy/light/fragile-crate; these actions depict mover robots are moving towards heavy, fragile, and light crates.

(c) mover-pickup-heavycrate/lightcrate/fragilecrate; these actions highlight the mover robots is picking up the different types of the crates from the warehouse.

(d) drop-heavycrate/lighcrate/fragilecrate-at-loadingbay; these actions will shows that mover robots will drops crates at the loading bay.

(e)loading-crate-on-conveyor-belt; this the last action defined in the designing of plan. This action will express that loader robot is placing the crates on the conveyor belt.

Problem File: This part consists of

(i)Objects: The objects which are defined in the problem file represents that what objects to be encoded in the problem file. There are different variables declared to distinguish among the objects such as:

Problem 1: In problem 1, we defined h for heavy crate, l1 for light crate, f for fragile crate, mvr1 and mvr2 for mover robots, l for loader robot.

Problem 2: In problem 2, we declared h1 and h2 for heavy crates, l1 and l2 for light crates, mvr1 and mvr2 for mover robots, l for loader robots.

Problem 3: In problem 3, we illustrate variables h1 and h2 for heavy crates, l1 for light crates, f for fragile crates, mvr1 and mvr2 for mover robots, l for loader robot.

Problem 4: In problem 4, we elucidate that h is for heavy crate, l1 and l2 for light crates, f1, f2, f3, and f4 are for fragile crates, mvr1 and mvr2 for mover robots, and lat but not the least l is for loader robot.

(ii)Init: This section is the initial state of the plan. This will highlights the initial position of the objects such as

Problem 1: In this problem, there are 3 crates. Also,
(a) mvr1 and mvr2 i.e. two mover robots tops is empty. Also, defined by "empty-top-mover mvr1/2".

(b) Loader robot is free and it is declares as "idleloader l".

(c) Position of the mover robots is at the loading bay. Declared by "mover-placed-loadingbay-mvr1/2".

(d) Crates at the initial position and also distinguish the crates groups for extension 1. In this case, fragile (20 Kg) and light (20 Kg) crate is grouped into same group name as Group A and heavy crate (70 Kg) is not defined into any group, declared as "crate-initial-position f/l1/h".

(e) Defining the crate distance and weight such as "dis-of-crate f/l1/h" and "wt-of-crate f/l1/h".

Problem 2: In this problem initial state is similar to Problem 1 initial state but there is a slight difference in crates weight and distance such as heavy crate h1 (70 Kg) and h2 (80 Kg) are categorize in group A for extension 1 and light crates l1 (20 Kg) and l2 (30 Kg) are labled into group B for extension 1. This means that groups (A, B, and so on.) are to indicate one of the optimal extensions that we need to consider during planning. Crates from the same group have to be loaded one after the other. There is also change in distance of the crates. This problem is dealing with 4 crates.

Problem 3: This problem is also same as Problem 1 and 2 and the number of crates are 4 but the only changes in crates weight and groups for extension 1. Fragile crate f (80 Kg) and and heavy crates (70 Kg and 60 Kg) are class into group A for extension 1 whereas, light crate l1 (10 Kg) is kept out of the group.

Problem 4: This problem is consists of 6 crates of different weight and types. There are 4 fragile crates and 2 light crates. Fragile crate f2, f3, and f4 are grouped into Group B for extension 1 and f1 and l1 is categorize into Group A for extension 1 whereas, l2 is class into no group.

(iv) Goal state: this state will define the goal state of the planning model. In this loader robot will placed the crates on the conveyor belt to accomplish the task. One thing in our case we gave priority to the fragile crates. As fragile crates needs more care and they are to be picked and dropped in the priority. If the crates are categorize into groups such as Group A and Group B for extension 1 there is always the priority for fragile crates to be picked up by the mover robots and drooped at the loading bay. Also, loader robot will gives the priority to the fragile crates which means loader robot will placed the fragile crates first on the conveyor belt.

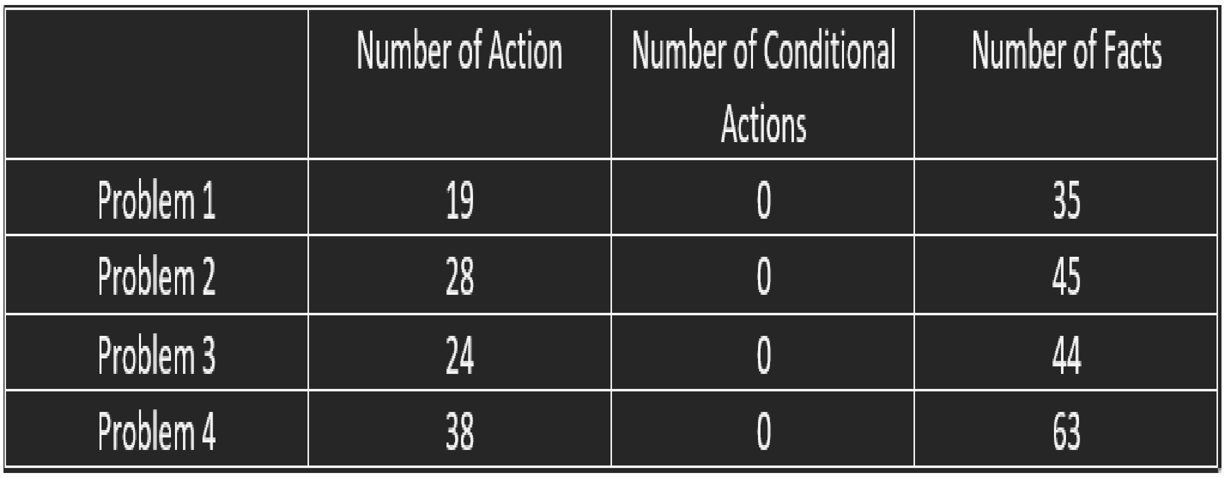
(v) Metric: it is defines as the numeric operations performed on the fluents, which includes single fluents. It is also states that the problem is a temporal problem with the cost function of total-time.

4 Results

This section will describes the experimental results which shows the LPG performance using Numeric and temporal Planning problems. The production of LPG was carryout in terms not only CPU-time required to find a results but also the quality of plan computed. This will also defines that how the mover robots actions to carry the crates according to the groups for extension 1 and accomplishing the task. The mover robots starts from the loading bay and move towards the crates initial position i.e. crates at the warehouse. The mover robots will pick up the crates and drop at the loading bay. Now, at the loading bay loader robot will placed the crates on the conveyor belt.

Here are the some results parameters obtain from the planners such as:

Modality: Incremental Planner



	Number of Action	Number of Conditional Actions	Number of Facts
Problem 1	19	0	35
Problem 2	28	0	45
Problem 3	24	0	44
Problem 4	38	0	63

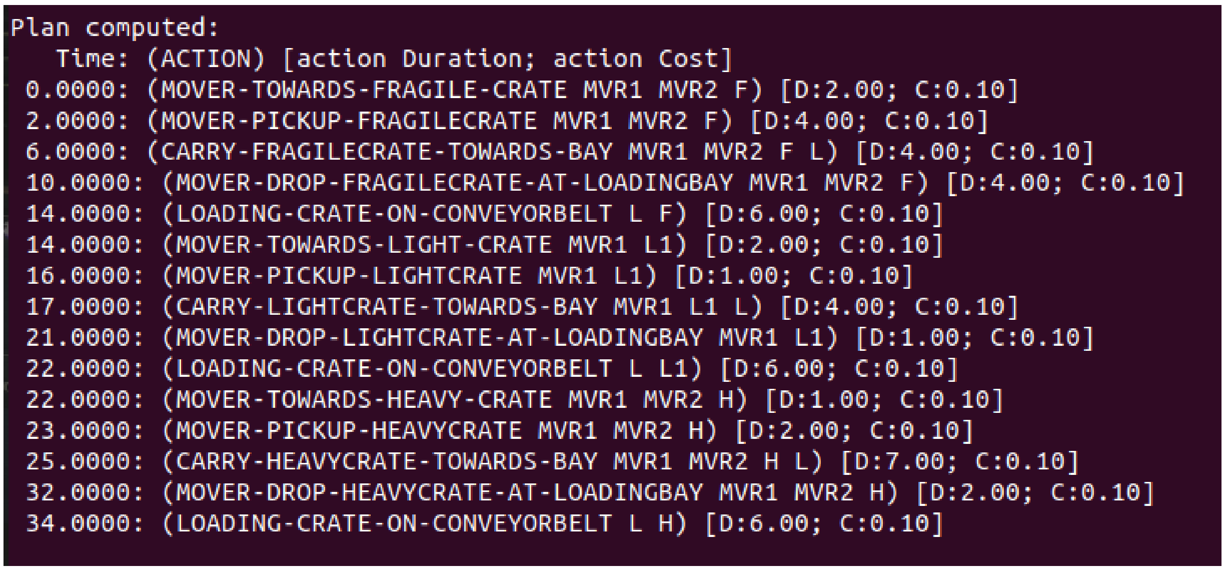
Figure 4.1: Actions to solve a problems

The above figure highlights the table that describes the actions taken by the planer to solve the problems.

Analyzing Planning Problem:
(i) Temporal Planning Problem: YES
(ii) Numeric Planning Problem: YES
(iii) Problem with Timed Initial Literals: NO
(iv) Problem with Derived Predicates: NO

Evaluation function weights:
(i) Action duration 1.00
(ii) Action cost 0.00

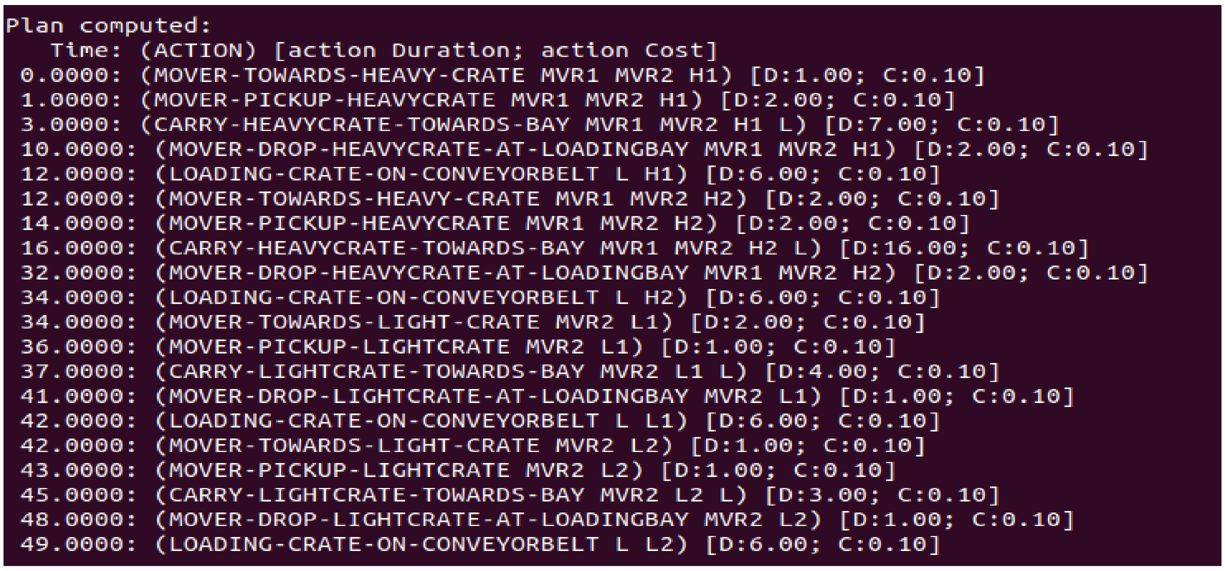
Below are the figures which shows the results after the plan computed.



```
Plan computed:
   Time: (ACTION) [action Duration; action Cost]
 0.0000: (MOVER-TOWARDS-FRAGILE-CRATE MVR1 MVR2 F) [D:2.00; C:0.10]
 2.0000: (MOVER-PICKUP-FRAGILECRATE MVR1 MVR2 F) [D:4.00; C:0.10]
 6.0000: (CARRY-FRAGILECRATE-TOWARDS-BAY MVR1 MVR2 F L) [D:4.00; C:0.10]
 10.0000: (MOVER-DROP-FRAGILECRATE-AT-LOADINGBAY MVR1 MVR2 F) [D:4.00; C:0.10]
 14.0000: (LOADING-CRATE-ON-CONVEYORBELT L F) [D:6.00; C:0.10]
 14.0000: (MOVER-TOWARDS-LIGHT-CRATE MVR1 L1) [D:2.00; C:0.10]
 16.0000: (MOVER-PICKUP-LIGHTCRATE MVR1 L1) [D:1.00; C:0.10]
 17.0000: (CARRY-LIGHTCRATE-TOWARDS-BAY MVR1 L1 L) [D:4.00; C:0.10]
 21.0000: (MOVER-DROP-LIGHTCRATE-AT-LOADINGBAY MVR1 L1) [D:1.00; C:0.10]
 22.0000: (LOADING-CRATE-ON-CONVEYORBELT L L1) [D:6.00; C:0.10]
 22.0000: (MOVER-TOWARDS-HEAVY-CRATE MVR1 MVR2 H) [D:1.00; C:0.10]
 23.0000: (MOVER-PICKUP-HEAVYCRATE MVR1 MVR2 H) [D:2.00; C:0.10]
 25.0000: (CARRY-HEAVYCRATE-TOWARDS-BAY MVR1 MVR2 H L) [D:7.00; C:0.10]
 32.0000: (MOVER-DROP-HEAVYCRATE-AT-LOADINGBAY MVR1 MVR2 H) [D:2.00; C:0.10]
 34.0000: (LOADING-CRATE-ON-CONVEYORBELT L H) [D:6.00; C:0.10]
```

Figure 4.2: Problem 1 plan computed result

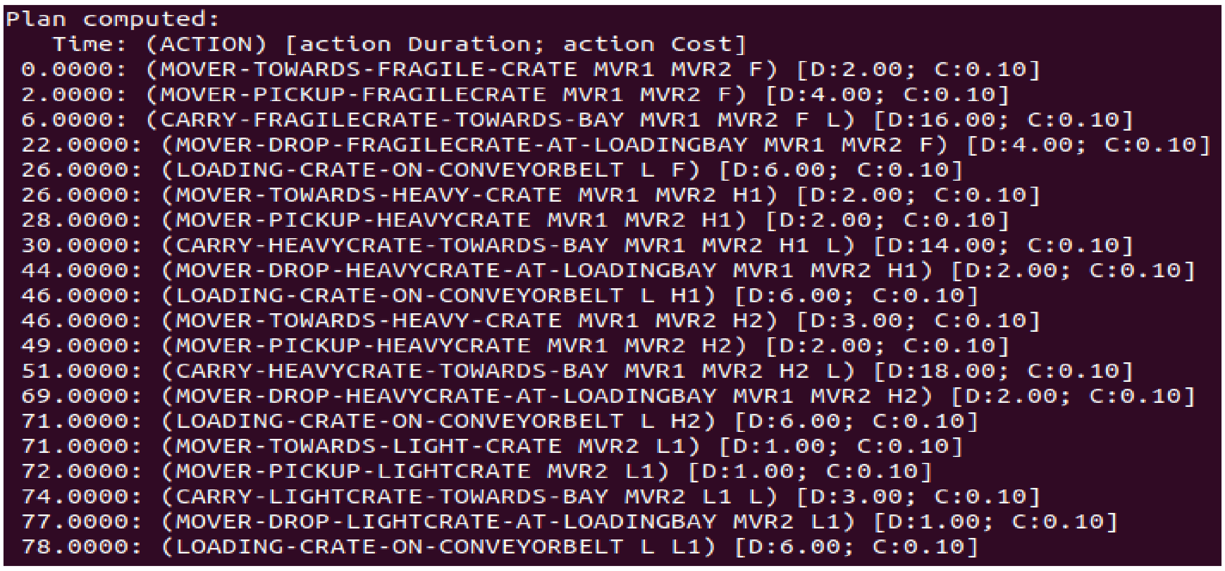
Figure 4.2 shows the results after the plan computation of Problem 1. It highlights the action duration i.e. action cost of the plan. Also, describes the plan execution such as mover robots performance to accomplish the goal.



```
Plan computed:
   Time: (ACTION) [action Duration; action Cost]
 0.0000: (MOVER-TOWARDS-HEAVY-CRATE MVR1 MVR2 H1) [D:1.00; C:0.10]
 1.0000: (MOVER-PICKUP-HEAVYCRATE MVR1 MVR2 H1) [D:2.00; C:0.10]
 3.0000: (CARRY-HEAVYCRATE-TOWARDS-BAY MVR1 MVR2 H1 L) [D:7.00; C:0.10]
 10.0000: (MOVER-DROP-HEAVYCRATE-AT-LOADINGBAY MVR1 MVR2 H1) [D:2.00; C:0.10]
 12.0000: (LOADING-CRATE-ON-CONVEYORBELT L H1) [D:6.00; C:0.10]
 12.0000: (MOVER-TOWARDS-HEAVY-CRATE MVR1 MVR2 H2) [D:2.00; C:0.10]
 14.0000: (MOVER-PICKUP-HEAVYCRATE MVR1 MVR2 H2) [D:2.00; C:0.10]
 16.0000: (CARRY-HEAVYCRATE-TOWARDS-BAY MVR1 MVR2 H2 L) [D:16.00; C:0.10]
 32.0000: (MOVER-DROP-HEAVYCRATE-AT-LOADINGBAY MVR1 MVR2 H2) [D:2.00; C:0.10]
 34.0000: (LOADING-CRATE-ON-CONVEYORBELT L H2) [D:6.00; C:0.10]
 34.0000: (MOVER-TOWARDS-LIGHT-CRATE MVR2 L1) [D:2.00; C:0.10]
 36.0000: (MOVER-PICKUP-LIGHTCRATE MVR2 L1) [D:1.00; C:0.10]
 37.0000: (CARRY-LIGHTCRATE-TOWARDS-BAY MVR2 L1 L) [D:4.00; C:0.10]
 41.0000: (MOVER-DROP-LIGHTCRATE-AT-LOADINGBAY MVR2 L1) [D:1.00; C:0.10]
 42.0000: (LOADING-CRATE-ON-CONVEYORBELT L L1) [D:6.00; C:0.10]
 42.0000: (MOVER-TOWARDS-LIGHT-CRATE MVR2 L2) [D:1.00; C:0.10]
 43.0000: (MOVER-PICKUP-LIGHTCRATE MVR2 L2) [D:1.00; C:0.10]
 45.0000: (CARRY-LIGHTCRATE-TOWARDS-BAY MVR2 L2 L) [D:3.00; C:0.10]
 48.0000: (MOVER-DROP-LIGHTCRATE-AT-LOADINGBAY MVR2 L2) [D:1.00; C:0.10]
 49.0000: (LOADING-CRATE-ON-CONVEYORBELT L L2) [D:6.00; C:0.10]
```

Figure 4.3: Problem 2 plan computed result

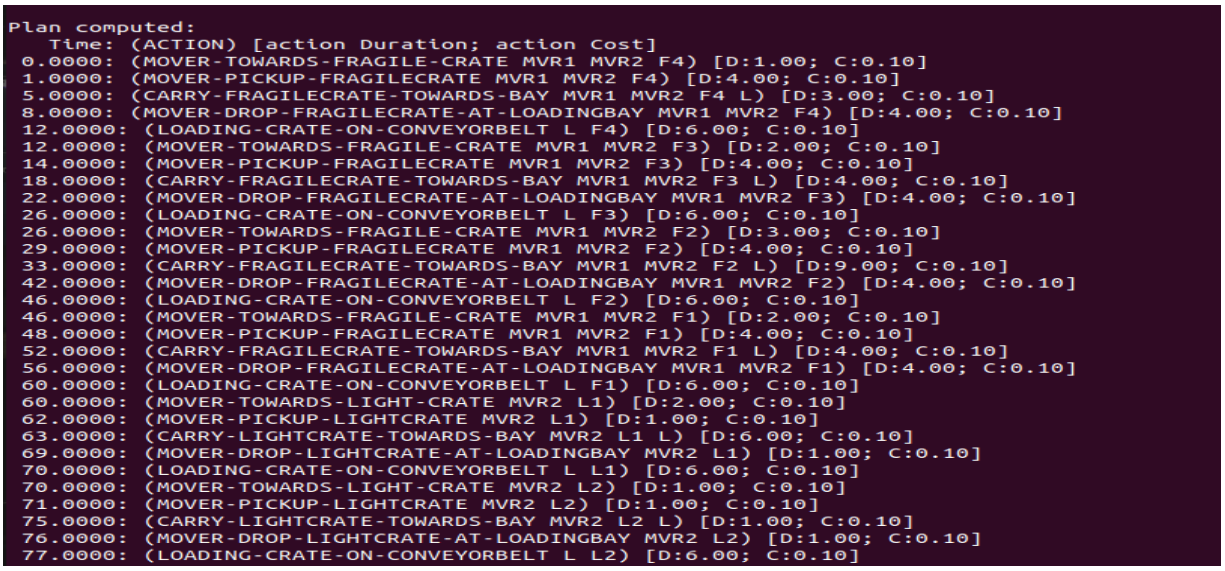
Figure 4.3 shows the results after the plan computation of Problem 2. It highlights the action duration i.e. action cost of the plan. Also, describes the plan execution such as mover robots performance to accomplish the goal.



```
Plan computed:
   Time: (ACTION) [action Duration; action Cost]
 0.0000: (MOVER-TOWARDS-FRAGILE-CRATE MVR1 MVR2 F) [D:2.00; C:0.10]
 2.0000: (MOVER-PICKUP-FRAGILECRATE MVR1 MVR2 F) [D:4.00; C:0.10]
 6.0000: (CARRY-FRAGILECRATE-TOWARDS-BAY MVR1 MVR2 F L) [D:16.00; C:0.10]
 22.0000: (MOVER-DROP-FRAGILECRATE-AT-LOADINGBAY MVR1 MVR2 F) [D:4.00; C:0.10]
 26.0000: (LOADING-CRATE-ON-CONVEYORBELT L F) [D:6.00; C:0.10]
 26.0000: (MOVER-TOWARDS-HEAVY-CRATE MVR1 MVR2 H1) [D:2.00; C:0.10]
 28.0000: (MOVER-PICKUP-HEAVYCRATE MVR1 MVR2 H1) [D:2.00; C:0.10]
 30.0000: (CARRY-HEAVYCRATE-TOWARDS-BAY MVR1 MVR2 H1 L) [D:14.00; C:0.10]
 44.0000: (MOVER-DROP-HEAVYCRATE-AT-LOADINGBAY MVR1 MVR2 H1) [D:2.00; C:0.10]
 46.0000: (LOADING-CRATE-ON-CONVEYORBELT L H1) [D:6.00; C:0.10]
 46.0000: (MOVER-TOWARDS-HEAVY-CRATE MVR1 MVR2 H2) [D:3.00; C:0.10]
 49.0000: (MOVER-PICKUP-HEAVYCRATE MVR1 MVR2 H2) [D:2.00; C:0.10]
 51.0000: (CARRY-HEAVYCRATE-TOWARDS-BAY MVR1 MVR2 H2 L) [D:18.00; C:0.10]
 69.0000: (MOVER-DROP-HEAVYCRATE-AT-LOADINGBAY MVR1 MVR2 H2) [D:2.00; C:0.10]
 71.0000: (LOADING-CRATE-ON-CONVEYORBELT L H2) [D:6.00; C:0.10]
 71.0000: (MOVER-TOWARDS-LIGHT-CRATE MVR2 L1) [D:1.00; C:0.10]
 72.0000: (MOVER-PICKUP-LIGHTCRATE MVR2 L1) [D:1.00; C:0.10]
 74.0000: (CARRY-LIGHTCRATE-TOWARDS-BAY MVR2 L1 L) [D:3.00; C:0.10]
 77.0000: (MOVER-DROP-LIGHTCRATE-AT-LOADINGBAY MVR2 L1) [D:1.00; C:0.10]
 78.0000: (LOADING-CRATE-ON-CONVEYORBELT L L1) [D:6.00; C:0.10]
```

Figure 4.4: Problem 3 plan computed result

Figure 4.4 shows the results after the plan computation of Problem 3. It highlights the action duration i.e. action cost of the plan. Also, describes the plan execution such as mover robots performance to accomplish the goal.



```
Plan computed:
   Time: (ACTION) [action Duration; action Cost]
 0.0000: (MOVER-TOWARDS-FRAGILE-CRATE MVR1 MVR2 F4) [D:1.00; C:0.10]
 1.0000: (MOVER-PICKUP-FRAGILECRATE MVR1 MVR2 F4) [D:4.00; C:0.10]
 5.0000: (CARRY-FRAGILECRATE-TOWARDS-BAY MVR1 MVR2 F4 L) [D:3.00; C:0.10]
 8.0000: (MOVER-DROP-FRAGILECRATE-AT-LOADINGBAY MVR1 MVR2 F4) [D:4.00; C:0.10]
 12.0000: (LOADING-CRATE-ON-CONVEYORBELT L F4) [D:6.00; C:0.10]
 12.0000: (MOVER-TOWARDS-FRAGILE-CRATE MVR1 MVR2 F3) [D:2.00; C:0.10]
 14.0000: (MOVER-PICKUP-FRAGILECRATE MVR1 MVR2 F3) [D:4.00; C:0.10]
 18.0000: (CARRY-FRAGILECRATE-TOWARDS-BAY MVR1 MVR2 F3 L) [D:4.00; C:0.10]
 22.0000: (MOVER-DROP-FRAGILECRATE-AT-LOADINGBAY MVR1 MVR2 F3) [D:4.00; C:0.10]
 26.0000: (LOADING-CRATE-ON-CONVEYORBELT L F3) [D:6.00; C:0.10]
 26.0000: (MOVER-TOWARDS-FRAGILE-CRATE MVR1 MVR2 F2) [D:3.00; C:0.10]
 29.0000: (MOVER-PICKUP-FRAGILECRATE MVR1 MVR2 F2) [D:4.00; C:0.10]
 33.0000: (CARRY-FRAGILECRATE-TOWARDS-BAY MVR1 MVR2 F2 L) [D:9.00; C:0.10]
 42.0000: (MOVER-DROP-FRAGILECRATE-AT-LOADINGBAY MVR1 MVR2 F2) [D:4.00; C:0.10]
 46.0000: (LOADING-CRATE-ON-CONVEYORBELT L F2) [D:6.00; C:0.10]
 46.0000: (MOVER-TOWARDS-FRAGILE-CRATE MVR1 MVR2 F1) [D:2.00; C:0.10]
 48.0000: (MOVER-PICKUP-FRAGILECRATE MVR1 MVR2 F1) [D:4.00; C:0.10]
 52.0000: (CARRY-FRAGILECRATE-TOWARDS-BAY MVR1 MVR2 F1 L) [D:4.00; C:0.10]
 56.0000: (MOVER-DROP-FRAGILECRATE-AT-LOADINGBAY MVR1 MVR2 F1) [D:4.00; C:0.10]
 60.0000: (LOADING-CRATE-ON-CONVEYORBELT L F1) [D:6.00; C:0.10]
 60.0000: (MOVER-TOWARDS-LIGHT-CRATE MVR2 L1) [D:2.00; C:0.10]
 62.0000: (MOVER-PICKUP-LIGHTCRATE MVR2 L1) [D:1.00; C:0.10]
 63.0000: (CARRY-LIGHTCRATE-TOWARDS-BAY MVR2 L1 L) [D:6.00; C:0.10]
 69.0000: (MOVER-DROP-LIGHTCRATE-AT-LOADINGBAY MVR2 L1) [D:1.00; C:0.10]
 70.0000: (LOADING-CRATE-ON-CONVEYORBELT L L1) [D:6.00; C:0.10]
 70.0000: (MOVER-TOWARDS-LIGHT-CRATE MVR2 L2) [D:1.00; C:0.10]
 71.0000: (MOVER-PICKUP-LIGHTCRATE MVR2 L2) [D:1.00; C:0.10]
 75.0000: (CARRY-LIGHTCRATE-TOWARDS-BAY MVR2 L2 L) [D:1.00; C:0.10]
 76.0000: (MOVER-DROP-LIGHTCRATE-AT-LOADINGBAY MVR2 L2) [D:1.00; C:0.10]
 77.0000: (LOADING-CRATE-ON-CONVEYORBELT L L2) [D:6.00; C:0.10]
```

Figure 4.5: Problem 4 plan computed result

Figure 4.5 shows the results after the plan computation of Problem 4. It highlights the action duration i.e. action cost of the plan. Also, describes the plan execution such as mover robots performance to accomplish the goal.

In the above figures it shows that mover robots is giving first priority to the fragile crates to carry from the initial position to the final position i.e. loading bay.

5 Conclusion

Efficient and proper management of the commodities is the need of the hour in the logistics sector. Each and every company, be it a small-scale, medium-scale, or large-scale needs an efficient inventory management system. An automated warehouse providing the maximum efficiency would help these companies deal with the problem of capacity and storage. In the assigned task, we tried to address the problem by proposing and even simulating it which would provide a fruitful outcome. We organized the tasks according to the priority levels and the time assigned to each task was also taken into account thus enabling us to perform the given tasks within the specifications, safety levels, and time consumption. As we are already in the age of Artificial Intelligence, we can already see its impact on the tiniest of areas.

6 References

[1] Fox M.; Long D.(2006). Modelling Mixed Discrete-Continuous Domains for Planning. Journal of Artificial Intelligence Research 27 235–297.

[2] Cushing, W. (2012). When is temporal planning really temporal? PhD Thesis, Arizona State University.

[3] Li, H., and Williams, B. (2011). Hybrid planning with temporally extended goals for sustainable ocean observing. In Proc. AAAI. v

[4] Penberthy, S., and Weld, D. (1994). Temporal Planning with Continuous Change. In Proc. AAAI. v

[5] Vallati, M.; Fawcett, C.; Gerevini, A.; Hoos, H.; Saetti, A.; (2011).ParLPG: Generating Domain-Specific Planners through Automatic Parameter Configuration in LPG.7th International Planning Competition 2011.

[6] Long, D., and Fox, M. (2003). The 3rd international planning competition: Results and analysis. JAIR 20:1–59.

[7] McDermott, D. (1982). A temporal logic for reasoning about processes and plans. Cognitive Science 6:101–155.

[8] Smith, D., and Weld, D. (1999). Temporal planning with mutual exclusion reasoning. In Proceedings of IJCAI-99,Stockholm, 326–337.

[9] Vila, L. (1994). A survey on temporal reasoning in artificial intelligence. AI Communications 7:4–28.

[10] Bacchus, F., and Kabanza, F. (2000). Using temporal logic to express search control knowledge for planning. Artificial Intelligence 116(1-2):123–191.