DEVELOPMENT OF A HEURISTIC SEARCH BASED PLANNER FOR AN AMBIENT INTELLIGENT SYSTEM

A THESIS SUBMITTED TO

THE FACULTY OF ARCHITECTURE AND ENGINEERING

OF

EPOKA UNIVERSITY

BY

DAVID VELIU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR

THE DEGREE OF MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

June, 2020

Approval sheet of the Thesis

This is to certify that we have read this thesis entitled **"Development of a heuristic search based planner for an Ambient Intelligent System"** and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

____

Dr. Ali Osman Topal
Head of Department
Date: 06, 26, 2020

Examining Committee Members:

Dr. Ali Osman Topal	(Computer Engineering)	__________________
Assoc. Prof. Dr. Dimitrios Karras	(Computer Engineering)	__________________
Dr. M. Maaruf Ali	(Computer Engineering)	__________________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Surname: David Veliu

Signature: ____________

ABSTRACT

DEVELOPMENT OF A HEURISTIC SEARCH BASED PLANNER FOR AN AMBIENT INTELLIGENT SYSTEM

David Veliu
M. Sc., Department of Computer Engineering,
Supervisor: Assoc. Prof. Dr. Dimitrios Karras

With the improvement of living standards, the aging of society is increased. As the population age, the number of people with cognitive disabilities is growing in various regions of the world. In the context, the traditional architectural forms have been unable to meet the basic needs of people with cognitive dis-abilities. Therefore, in recent years, Artificial Intelligence (AI) has been rapidly developed. However, the integration of AI in this range of society is lacking.

The main focus of my Master Thesis is to migrate these challenges by developing an AI system with the help of a planning platform called PDDL (Planning Domain Definition Language).

Keywords: Alzheimer Disease, Artificial Intelligence, Ambient Assisted Living, Assisted Daily Living, Planning Domain Definition Language, Planning Domain Definition Language, For Java, VALidator

ABSTRAKT

ZHVILLIMI I NJE KERKIMI HEURISTIK NE NJE AMBIENT INTELIGJENT

David Veliu
Master Shkencor, Departamenti i Inxhinierisë Kompjuterike
Udhëheqësi: Assoc. Prof. Dr. Dimitrios Karras

Me zvhillimin e standarteve te jeteses, mosha mesatare ne popullsi rritet. Me plakjen e popullsise, numri i personave me aftesi te kufizuara rritet ne zona te ndryshme te tokes. Ne kete kontekst format tradicionale arkitekturore nuk kane mundur te ndihmojne ne nevojat bazike te njerezve me aftesi te kufizuara. Prandaj, ne vitet e fundit, Inteligjenca Artificiale (AI) eshte zhvilluar. Megjithate, integrimi i AI per kete game te shoqerise ka veshtiresi.

Fokusi kryesor i kesaj teze eshte te migrohen keto veshtiresi duke zhvilluar nje system AI me ndihmen e nje platforme qe quhet PDDL (Planning Domain Definition Language).

Keywords: Alzheimer Disease, Artificial Intelligence, Ambient Assisted Living, Assisted Daily Living, Planning Domain Definition Language, Planning Domain Definition Language, For Java, VALidator

ACKNOWLEDGEMENTS

I would first like to thank my thesis supervisor Assoc. Prof. Dr. Dimitrios Karras of the department of Computer Engineering at EPOKA UNIVERSITY. He consistently allowed this paper to be my own work but steered me in the right the direction whenever he thought I needed it. I would also like to ac-knowledge Prof. Dr. Ali Osman TOPAL as the head of department, I am gratefully indebted for his very valuable help and comments throughout these years of master on the paper of previous project thesis.

Big thanks go to Danien Pellier for the great job done developing the PDDL4J library. Every result described in this thesis was accomplished with the help and support of the library created and published as open source by Mr. Pellier. I greatly benefited from his work on implementing new ideas based on the requirements tailored in the needs of the project.

Finally, I must express my very profound gratitude to my parents and to my girlfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

TABLE OF CONTENTS

ABSTRACT III

ABSTRAKT IV

ACKNOWLEDGEMENTS V

LIST OF FIGURES VIII

LISTINGS IX

LIST OF TABLES & GRAPHs X

LIST OF ABBREVIATIONS XI

CHAPTER 1 1

INTRODUCTION 1

1.1 RELATE WORK 3

1.2 GOAL 3

1.3 STRUCTURE DEFINITION 4

1.4 DEMENTIA & ALZHEIMER DISEASE 5

1.4.1 DEMENTIA STAGES 6

1.5 SOCIAL IMPACT OF ASSISTIVE TECHNOLOGIES 8

CHAPTER 2 12

TOOLS & METHODS 12

2.1 APPROACHES 12

2.2 SATE OF THE ART ANALYSIS 13

2.3 PDDL ARCHITECTURE 15

2.3.1 PDDL PLANNERS 16

2.3.2 PDDL VALIDATOR 18

CHAPTER 3 20

ANALYSIS 20

3.1 USER MODELLING & SCENARIOS 20

3.2 DOMAIN 23

3.3 PROBLEM 28

CHAPTER 4 31
SOLUTIONS...31
4.1 PDDL TO OBJECT ORIENTED PROGRAMMING (OOP)..................31
4.2 HEURISTIC SEARCH...34
4.3 PARTITIONING..41
4.4 KSH PLANNER..42
CHAPTER 5 43
EVALUATION & VALIDATION...43
5.1 PARSING..43
5.2 ENCODING 45
5.3 SEARCHING 46
5.4 VALIDATION 52
CHAPTER 6 53
DISCUSSIONS & CONCLUSIONS...53
6.1 FUTURE ENHANCEMENTS...53
6.1.1 IMPLEMENTING MORE PROBLEMS FOR THE WORLD 53
6.2 AUTOMATION OF TABULAR DATABASE.......................................53
6.3 DATA VISUALISATION 54
6.4 Graph visualization..54
6.5 Partitioning & Heuristic visualisation 55
Bibliography 56

LIST OF FIGURES

Figure 2.1: Waterfall Model 12

Figure 4.1: Visualization of a problem. 35

Figure 4.2: Transition function 37

Figure 4.3: Regression function 38

Figure 4.5: actions : change-room(lucy room-a room-b) 38

Figure 4.6: Critical Path heuristic h^1 representation through graph 39

Figure 4.7: Critical Path Heuristic h^m 40

Figure 5.1: Output of SAPA planner 49

Figure 5.2: SAPA computation time 50

Figure 5.3: Results of WebPlanner 50

Figure 5.4: PDDLeditor output report 51

Figure 5.5: PDDLeditor error report 51

Figure 5.6: Results of KSH planner 52

Figure 5.7: Validation generated output 53

Figure 6.1: Sample Graph representation 56

LISTINGS

Listing 2.1: Sample Domain 15

Listing 2.2: Sample Problem 16

Listing 3.1: Domain declaration. 23

Listing 3.10: Lucy gives medicine. 28

Listing 3.11: Problem Declaration. 29

Listing 3.12: Problem Declaration. 30

Listing 3.2: Predicates. 24

Listing 3.3: Lucy changes room. 24

Listing 3.4: Lucy switch lights. 25

Listing 3.5: Lucy picks glass. 25

Listing 3.6: Lucy picks water. 26

Listing 3.7: Lucy pour water. 26

Listing 3.8: Lucy gives water. 27

Listing 3.9: Lucy picks medicine. 27

Listing 4.1: Fluency declaration. 33

LIST OF TABLES & GRAPHS

Table 3.1: Personae Jens 20

Graph 5.1: Computational Time Of Parsing Module 44

Graph 5.3: Encoding Memory Test 47

Graph 5.4: Search Test 48

Graph 5.5: Searching Memory Test 49

LIST OF ABBREVIATIONS

AD	Alzheimer Disease
AI	Artificial Intelligence
AAI	Ambient Assisted Living
ADL	Assisted Daily Living
PDDL	Planning Domain Definition Language
PDDL4J	Planning Domain Definition Language for Java
VAL	VALidator

CHAPTER 1

INTRODUCTION

The industrial western societies are ageing and therefore faced with an increasing number of cases of dementia and Alzheimer's disease. As for today, 47 million people are expected to be living with dementia worldwide (Prince, Sept. 2016).

Dementia is defined as a decline in multiple cognitive capacities such as memory loss, language, orientation, reasoning, comprehensive thinking, attention deficits, and others. This is most caused by Alzheimer's disease (Kurz, Dec. 2016). The disease is typically progressive and cannot be cured, al-though symptoms can alleviate additional symptoms arrive in combination with loss in cognitive capacities, disorientation, and loss of impulse control, especially with the awareness of the individual of not being healthy and decaying.

In the recent years, much work has been done in the area of ambient assisted Living (AAL) which addresses the specific needs of elderly people. Technology has been integrated intelligently and assistive in living environments to support critical activities. It also tries to ensure as much independence for the residents for as long as possible (Prince, Sept. 2016).

Assistive technology studies have been focused, on making systems more operational and practical, aiming at offering users with experiences that are suitable for their background knowledge and goals (Starner, T., J. Weaver, and A. Pentland, 1998)The main group of people with disabilities that is focused by the researchers are those of the elderly people (Gregor, Peter, Alan F. Newell, and Mary Zajicek, 2002). It has been the main concern for researchers on human-computer interaction. Widespread access has become a conventional topic and issue for reachable computing research (Shneiderman, 2003). The main ambition is by providing an achievable state of interactions with applications, service products anywhere at any time (Stephanidis, Constantine and Anthony Savidis, 2001). assistive technologies are those applications implemented according to each user's needs. So the most suitable system for a user is an individual implementation to their particular needs or preferences. User modeling (Fischer, 2001) and adaptation techniques are critical providing suitable solutions for that group of users, such as disabled people (Stephanidis, 2001).

More recently, assistive applications can help users with limitations in different activities of their daily living. Each day, all of us are involved in many activities related to study, work, housework and so on. Within the day, there are a lot of decisions to be taken, either in regular situations or in unexpected ones. However, while some people can make these decisions easily, this may be difficult for people with cognitive limitations. Computer-based "training" applications can help people to improve their abilities (Keates, 2002). Projects like Dem@Home (Stavropoulos, 2016) focus specifically on the needs and demands of elderly people who live with dementia. It identifies that the prolonging of independent living supports cognitive state and improvement of mood. According to (Aumayr, 2016) the development and maintenance of a daily routine is also identified to be important for people to live with dementia.

Nevertheless, a large community of researchers dedicated their efforts to integrate Artificial Intelligence (AI) planning into daily routine activities of elderly that suffer from dementia (Mart´ın, Estefan´ıa, Pablo A. Haya, and Rosa M. Carro, 2013). This could benefit those affected, by guiding them during the day-to-day activities, independently of their initial state.

However, with all the features that AAL aims to provide, the process pursued by this community often ignores specific problems when it comes to developing and integrating a complete AI planning system (Strobel, Volker and Alexandra Kirsch, 2014). In conjunction between standard AI planning and actual AAL systems, the Planning Domain Definition Language (PDDL) based AI planners introduces some drawbacks for systems that aid elderly people. a lack of integration of PDDL based planners with AAL is one of the key weakness in this topic.

Even with the lack of integration in this field, the use of heuristic search planners has increased significantly due to the immediate solution finder (Bonet, Blai and Hector Geffner, 2001a). Heuristic search planners transform planning problems into problems of heuristic, by automatically extracting heuristics from Strips encoding. The uniqueness of heuristic concepts is that they differ from a specialized problem.

Therefore, it is needed to create an automated system with PDDL with the help of a heuristic function tailored to the needs of the user. That represents a promising approach for the extension of an independent and self-conducted life of elderly people thereby, improving their quality of life and minimizing the need for care. The construction of trustworthy living assistance system is an extremely challenging task and requires novel approaches for dependable self-adapting software architectures.

1.1 RELATE WORK

Assistive systems show great potential for users with special needs. The first related project with AAL that I faced was a puppet system. The aim of a puppet system is to establish a human-machine interaction that can improve the health and mood of an informal caregiver, such as their family member. This is achieved during times of absence of the caregiver where the Puppet supports the patient on daily activities. This can reduce the stress level of these informal caregivers. In order to maintain a certain level of the family bond between caregiver and patient should be supported. To achieve this objective, the puppet provides the functionality to establish communication between the client and their relatives. In addition, the puppet can detect stressed situations and can apply certain activities to aid the client to overcome these situations. For example, changing the mood through jokes. In the case of a not solvable or critical situation, the puppet will try to calm the patient and at the same time will try to notify the caregivers. The final goal was to develop an all in one system that incorporates assistive techniques to ensure user-friendliness and "smart" interactions for all the potential users.

1.2 GOAL

The aim of this works is to develop a PDDL based planner for an ambient Intelligent System and overcome the introduced drawbacks with the improvement by a heuristic search. The system is tailored to the needs of people with cognitive disabilities. The planning time is one of the main keys to determine good enhancements for the new system. To accomplish this goal, two tasks have been identified:

1. Design of a living environment of people with cognitive disabilities in such a way that the AI can aid humanity.

2. Develop a search algorithm using heuristic and the A* algorithm.

In other words, the goal is to create a shorter planning time system than other solutions that are used for people with cognitive disabilities in the market.

1.3 STRUCTURE DEFINITION

The following chapter will give some basic knowledge of Dementia & Alzheimer disease along with the impact technology has on these people. It will cover the related work in this area, that leads to the current idea of developing this solution.

The second chapter will describe the waterfall model used as a development process for the problem and will explain why "problem-driven" approach is used in this case. This chapter will also include the research process of the technologies that are applied for people with dementia and the solution each technology is providing in this area. Next will be an outline of the architecture and its features in relation to PDDL.

The third chapter introduces and defines the user scenario and the corresponding personae. This use case then transposed to PDDL in the form of a domain and problem description. This third chapter also gives a logical explanation of the domain and problem code parts.

In chapter four, it is described how the proposed solution is performed and the steps followed to achieve the goal. This chapter starts by explaining the integration of the PDDL4J library. Next is explained how the translation of the PDDL to Java is made and what changes are needed. It also reasons why this step is so important for heuristic improvement. This chapter will introduce the use of a heuristic and will describe the most used ones where critical path heuristic is chosen and applied. A description of the process with the help of examples is done and explained in mathematical formulas. The last section of this chapter will be partitioning used that aids in speeding up the process of the heuristic search.

Coming to chapter five, an illustration of the system evaluation is given with the included parsing and encoding process. After this a comparison of the final product with other platforms in the market is done and represented as charts. At the end of the chapter, the use of the validation process will be described, and the output will be explained in relation to the algorithm.

To conclude with the last chapter, some new features that may be implemented in the future are presented.

1.4 DEMENTIA & ALZHEIMER DISEASE

The German psychiatrist Dr. Alois Alzheimer is credited with defining for the first time a dementing condition which later named after his name as AD. In his innovatory thoughts, he made a conference lecture and a subsequent article where Alzheimer described the case of a 51-year-old woman with a peculiar dis-ease of the cerebral cortex (Selkoe, 2002). She was diagnostic with language impairment, disorientation, progressive memory, behavioral symptoms, delusions paranoia, hallucinations, and psychosocial impairment (Strassnig, Martin and Mary Ganguli, 2005). Surprisingly, many of the clinical observations and pathological findings that Alzheimer labeled more than a century ago continue to remain vital to our knowing of AD today (Mesulam, 2019).

AD is a deterioration of the brain disorder that causes a considerable amount of disruption of the normal brain structure and function. At the cellular level, AD is identified by a progressive loss of cortical neurons, especially pyramidal cells, that mediate higher cognitive functions (Mann, 1996) (Norfray, Joseph F. and James M. Provenzale, 2004). Significant evidence also suggests that AD causes synaptic dysfunction early in the disease process, disrupting communication within neural circuits important for memory and other cognitive functions (Selkoe, 2002). AD-related degeneration begins in the medial temporal lobe, specifically in the entorhinal cortex and hippo-campus (Groot, 2018). Damage to these brain structures results in memory and learning deficits that are classically observed with early clinical manifestations of AD. The brain decay then spreads around the temporal association cortex and to parietal areas. As the disease progresses, the degeneration can be seen in the frontal cortex and eventually throughout most of the remaining neocortex (Brickman, 2018). In addition to cognitive impairment across multiple domains (memory, language, reasoning, executive, and visuospatial function), patients with AD show an impaired ability to per-form activities of daily living and often experience psychiatric, emotional, and personality disturbances (Risacher, 2017). It has been theorized that the neuronal damage seen in AD is related to the deposition of abnormal proteins both within and outside of neurons. (Ossenkoppele, 2019)

Dementia is a clinical syndrome that involves progressive deterioration of intellectual function. Various cognitive abilities can be weakened with dementia, including memory, reasoning, decision making, language, visuospatial function, orientation, and attention (Gilman, 2010). Individuals suffering from dementia, cognitive impairments are often followed up by changes in personality, emotional regulation, and social behaviors. Importantly, the cognitive and behavioral changes that occur with dementia interfere with social activities, work, relation-ships and affect a persons ability to perform routine daily activities (e.g., driving, managing finances, shopping, cooking, housekeeping, and personal care). There are several reversible and irreversible causes of dementia (Gilman, 2010). Reversible dementias (also referred to as ‘pseudo-dementias’) are relatively rare but potentially treatable and occur secondary to another medical condition, including depression, nutritional deficiencies (e.g., vitamin B12), metabolic and endocrine disorders(e.g., hypothyroidism), space-occupying lesions (e.g., brain tumor), normal pressure hydrocephalus, or substance abuse (Gilman, 2010).

Certain classes of medications also have the potential to cause cognitive impairment in older adults (e.g., anticholinergic, psychotropics, analgesics, sedative-hypnotics). Irreversible (primary) dementias involve neurodegenerative and/or vascular processes in the brain AD is the most common cause of irreversible dementia, accounting for up to 70% of all dementia cases in Germany (Gilman, 2010). Other types of primary dementia include vascular dementia (10-20%), dementia associated with Parkinson’s disease, dementia with Lewy bodies, and frontotemporal dementia (Cahill, 2007).

1.4.1 DEMENTIA STAGES

as becoming mostly unclear for researchers to cover dementia as a unique single disease, therefore, it is categorized in stages which implies to what level of reach the disease has progressed. This method assists researchers to determine the best treatment approach and helps the communication between caretakers and the actual person with dementia (Topo, Technology Studies to Meet the Needs of People With Dementia and Their Caregivers: A Literature Review, 2009a). Dementia is subdivided into three stages that determine the progression of it: early stages, middle stages, and late stages. In the following list, a representation of all stages of dementia is described. However, in most cases, an exact stage of dementia is evaluated based on the person symptoms (Cohen-Mansfield, Jiska, Perla Werner, and Barry Reisberg, 1995).

Stage 1: No Cognitive Decline.

- **Signs and symptoms**: In this stage, the person functions normally, has no memory loss and is mentally healthy.
- **Stage Duration**: N/a
- **Diagnosis**: No Dementia

Stage 2: Very Mild Cognitive Decline.

- **Signs and symptoms**: This stage is used to describe normal forgetfulness associated with aging.
- **Stage Duration**: Unknown
- **Diagnosis**: No Dementia

Stage 3: Mild Cognitive Decline.

- **Signs and symptoms**: This stage includes increased forgetfulness, slight difficulty concentrating, and a decrease in work performance. People may get lost more frequently or have difficulty finding the right words. At this stage, a person's loved ones will begin to notice a cognitive decline.
- **Stage Duration**: average duration is between 2-7 years
- **Diagnosis**: Early Stages

Stage 4: Moderate Cognitive Decline

- **Signs and symptoms**: This stage includes difficulty concentrating, decreased memory of recent events, and difficulties managing finances or traveling alone to new locations. People have trouble completing complex tasks efficiently or accurately and may be in denial about their symptoms. They may also start withdrawing from family or friends because socialization becomes difficult.
- **Stage Duration**: average duration is between 2 years
- **Diagnosis**: Early Stages

Stage 5: Moderately Severe Cognitive Decline

- **Signs and symptoms**: People in this stage have major memory deficiencies and need some assistance to complete their daily living activities (dressing, bathing, preparing meals, etc.). Memory loss is more prominent and may include major relevant aspects of current lives.

- **Stage Duration**: average duration is between 1 and a half years
- **Diagnosis**: Mild Stage

Stage 6: Severe Cognitive Decline (Middle Dementia)

- **Signs and symptoms**: People in Stage 6 require extensive assistance to carry out their activities of Daily Living (ADLs). They start to forget names of close family members and have little memory of recent events. Many people can remember only some details of earlier life. Individuals also have difficulty counting down from 10 and finishing tasks. Incontinence (loss of bladder or bowel control) is a problem in this stage. Ability to speak declines. Personality & emotional changes, such as delusions and anxiety agitation may occur.
- **Stage Duration**: average duration is between 2 and a half years
- **Diagnosis**: Mid to Late Stage

Stage 7: Very Severe Cognitive Decline (Late Dementia)

- **Signs and symptoms**: People in this stage have essentially no ability to speak or communicate. They require assistance with most activities
- **Stage Duration**: average duration is between 1 and a half and 2 years
- **Diagnosis**: Late Stage

1.5 SOCIAL IMPACT OF ASSISTIVE TECHNOLOGIES

People with dementia need a great deal of assistance and support, and the necessity arises as the disease continues to progress. In cases of moderate and severe dementia, help is often needed 24 hours a day (Cohen-Mansfield, Jiska, Perla Werner, and Barry Reisberg, 1995). When family caretakers with dementia were interviewed, their main concerns were lack of meaningful activities, safety in the home, lack of time for themselves, and difficulties experienced in time orientation (Topo, Technology Studies to Meet the Needs of Peo-ple With Dementia and Their

Caregivers: A Literature Review, 2009a). Studies based on observations and interviews of people with dementia in residential care center have shown that meaningful activities are often lacking and evidence exists that by providing more stimuli and activities, such peoples quality of life can be improved (Cahill, 2007). Technology has been identified as one tool that can be used to improve independent living, improve the safety and autonomy of people with dementia, and support the quality of life of such people and their family caretakers (Brooker, 2005).

The number of people living with dementia is expected to rise by 131 billion in the next 30 years and it is natural to expect an increased economic impact of dementia in the near future (Risacher, 2017). Thus, technology has to assist people with special needs to increase their inclusion in the society for as long as possible and narrow down the gap between them and people who do not face such difficulties like memory loss in their everyday lives. according to Moore's law, the number of transistors doubles approximately every 2 years (Moore, 1998). The same rule applies to technology in general as it is related to the hardware progress in computing power. Nowadays digital technologies are more affordable than before and will continue to become more cost-effective. This opportunity can be used to increase the presence of technology in dementia care and cover people in rural or low-income areas who do not have access to medical facilities or they are very limited. In a high-income society like Germany, technology can lower down the cost of dementia care significantly by reducing the need for highly qualified staff and medical equipment. World population is aging and it is estimated that by 2050 there will be elder people (>60 years old) than younger (<15 years old) and this global trend is irreversible (Cohen, 2000).

a woman named Mary Marshall has proposed that assistive technology is mostly perceived as an increase of help and the provision of adaptations (Marshall M. N., 1996). She has listed some technology ideas that could assist and support people with dementia. In her list are included: reminders, relaxation, technology for stimulation, behavior management, safety, compensation, surveillance, service coordination technology and control assistance for relatives (Marshall M. , 2004). In this list, the addition of technology for communication should be added. However, using domestic technologies in dementia people may lead to difficulties. If people suffering from dementia somehow notice a way to reduce the struggle to handle these problems, they are still not productive in the long term (Kuchinomachi, 1999). Therefore, researchers are insisting that assistive technology is required to make up for their loss of cognitive functioning. Researcher Stephen Wey has mentioned some roles of assistive technology in the recovery of people with dementia, summarizing five key points for technology (Topo, Technology Studies to Meet the Needs of People With Dementia and Their Caregivers: A Literature Review, 2009a). In his list description, a lot of attention is focused on the daily routine of the person with dementia. First on the list is to assist and simplify the person's memory, direction, and other cognitive abilities essential to the person everyday life. Secondly, to enable the person to handle activities and tasks that are moving beyond their skills or that are in far beyond their reach. Finally, such active involvement is also vital to enable essential activity during the day, including leisure activities, and the preservation of valued roles in the family and other social networks. Two further roles include guaran-teeing the person's security, supporting and restoring confidence to caretakers. Caregivers working with dementia people report frequent physical disorder that is caused by emotional factors and exhaustion (Pekkarinen, Resident care needs and work stressors in special care units versus non-specialized long-term care units, 2006). The reason for these Symptoms are related to a lack of knowledge of dementia, problems in care center management and insufficient level of staffing in care units (Pekkarinen, Work Stressors and the Quality of Life in Long-Term Care Units, 2004) (Pekkarinen, Resident care needs and work stressors in special care units versus non-specialized long-term care units, 2006) Assistive technology is playing an important role in improving the condition of people working in dementia care (Hughes, Julian and Gill Campbell, 2003). Narrowing down the requirements depending on the aim of the technology used may be a challenging task. The user of the technology may be a person who suffers from dementia, a caretaker, an authorized caregiver, or even a police or emergency services where in some cases, the users may be some of these same individuals. These different user groups have very different user requirements and require a different approach to designing products in this topic is demanding because it requires simplistic solutions (Orpwood R. , 2004), whereas solutions for family caretakers and formal caregivers may require a new learning process for certain tasks (Orpwood R. , 2007). Nevertheless, the same technology can be used for different reasons by one user for other purposes and by other users like caretakers Additionally, many researchers have proclaimed difficulties in associating technologies in this topic (Topo, Technology Studies to Meet the Needs of People With Dementia and Their Caregivers: A Literature Review, 2009b). For example, a person with dementia may disagree with the fact of using surveillance cameras since he or she does not see any reason for installing them. On the other hand, a family member which is taking care of the person with dementia may find this crucial to provide better safety of the person and to relieve his or her own stress on certain concerns (Topo, Technology Studies to Meet the Needs of People With Dementia and Their Caregivers: A Literature Review, 2009b). Researcher Inger Hagen (Hagen, 2004) proposed that when assessing the use and usefulness of assistive technology for people with dementia, at least five aspects should be considered:

(a) the impact of the technology,

(b) the impact of the personal characteristics on the person with dementia,

(c) the impact of the family caretaker,

(d) the impact of the environment, and finally,

(e) the impact of the research process and the researchers.

For example, providing qualified smart puppets when dealing with daily activities and acting on behalf of a caretaker. Ambient Assisted Living is one of the possible solutions to help cognitive impaired people to preserve their lives at home for as long as possible and to reduce the need for caregivers

CHAPTER 2

TOOLS & METHODS

2.1 APPROACHES

Applied technologies for dementia people is a widespread topic. To develop software to assist people with dementia, many different approaches exist to guide this process. In this work, we consider the waterfall model as optimal for the case. The main aspect to select this

model is its well fit for smaller projects where requirements are well-defined and understood. The waterfall model described in the picture 2.1 is a plan-driven-process, where a plan must be scheduled for all the activities in order to maintain the deadline.

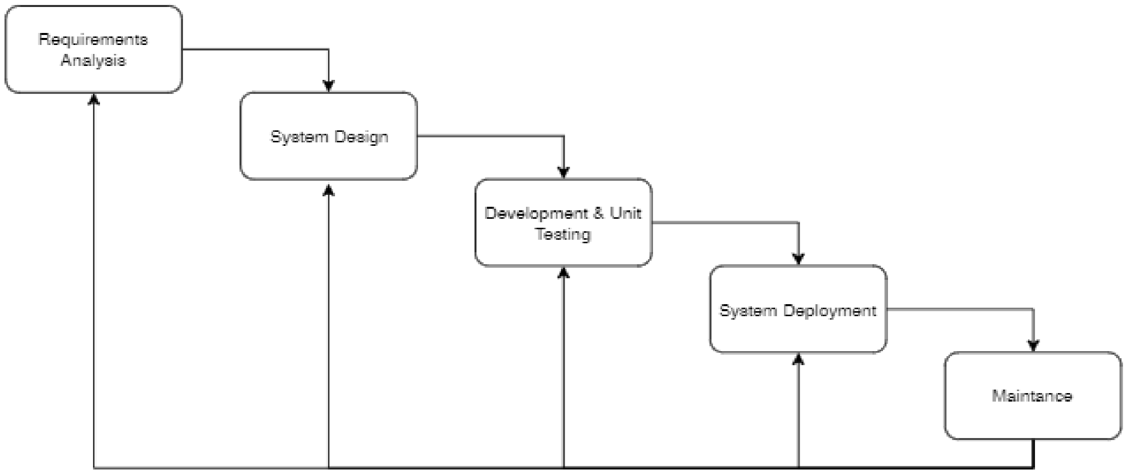


Figure 2.1: Waterfall Model

The requirements for the system are descriptions of the tasks that the system should be able to execute, the services that it should provide and the restrictions on its functionality. The process of requirement analysis is used on, finding out, checking and documenting services and restrictions. These requirements are derived from the needs of a persona in a scenario which needs aiding in their daily activities. In this work, each module designed is implemented. Unit testing is used to analyze whether each module is running properly or not. The first testing occurs by the developers, this is called Alpha testing. After the development of the software and a successful alpha testing, integration testing and user tests, the deployment of the system to a small group of users is done. During the use of the system new problems may arise known as bugs and new releases of the system may be launched with fixes to counter bugs and improvements needed. This process is named as Beta testing.

In this thesis, the deployment of the system is not feasible due to a lack of a real-world testing environment. Therefore, a visualization of it is achievable and easier to understand.

2.2 SATE OF THE ART ANALYSIS

Technology has evolved quite rapidly in the past few decades. It became an essential commodity and modern people cannot imagine their lives without it. It is crucial that society creates equal opportunities for all parts of the world population and technology can help in achieving that goal. Assistive technologies have been implemented for a range of applications such as dementia and Alzheimer, to name only a few.

Improving the quality of life for people suffering from dementia by a “smart” wristband was done by researchers at the Fraunhofer Institute in Berlin. The device itself is able to measures the patient’s vital signs, environmental data, and physical activity. The data records from the patient’s family and healthcare personnel are merged with the collected information from the device to provide ongoing advice. It helps the patient with normal day-to-day activities and notifies on disease progression.

The Hasbro Robo Pets which are produced by Joyforall company as companion RoboPets has a market Value of US$ 119 (Joy For All Companion, n.d.). At first sight, they look like a children toy, but these robopets are serving well in therapy scenarios for dementia people. As trained therapy animal pets the responsive pet can mimic a normal cat or dog. The built-in sensors can react to touch and use movement to mimic the behavior of a real-life friendly cat.

A Japanese company comes up with the first idea of a Therapeutic Robot in 1996 called PARO that has a focus in animal therapy to aid such patients in environments like hospitals and care facilities where live animals present treatment or logistical difficulties (PARO Therapeutic Robot, n.d.). It is constructed by five different sensors: temperature, tactile, light, noise, and posture sensors, that allow PARO to identify people and their environment. PARO is one of the first puppets that presented the implementation of learning algorithms in human interaction. It can self-learn to behave in user preferences and to respond intelligently. However, the entry level in the market of the puppet is US$ 4000 and with support option included, it can reach up to US$ 6000. Another limitation is the actuation capabilities.

SoftBank Robotics presented a robot named Pepper in 2015 designed with the ability to read emotions (SoftBank Robotics, n.d.). This main feature of Pepper derives from the ability to analyze voice tones and expressions. It is currently being used as a receptionist at several offices in the UK and can identify visitors with the use of facial recognition. It sends alerts for meeting organizers and serves drinks to guests. Pepper is said to be able to chat to prospective clients. The robot has also been used at banks and medical facilities in Japan and the starting price is US$ 1650.

The idea of smart home systems is applied by the company Silver Mother (Silver Mother, n.d.). It is a US$ 300 life-monitoring solution which offers remote monitoring for a different range of people like kids' babies and elderly people. Silver Mother can identify if a pill is taken or is missed. In addition to medication reminders, it can be customized to add more features like daily activity monitoring, hydrations measurements, sleep tracking, temperature monitoring, and mobility alerts. If a condition is not met, it automatically sends smartphone notifications to caregivers and to family members as well as sends reminder calls to the user.

All of the above-mentioned technologies have in common that they monitor and track human interactions but a few of them are applying both tasks simultaneously. Even though these products may be sufficient to provide aid in this society, the executable tasks are basic in functionalities, excluding PARO technology. None of the systems mentioned was focusing on the time efficiency instead they were more task-related jobs. With PDDL, a different and complex approach is being used where planning efficiency is the main goal. Due to the fact that PDDL is a search-based system of the encoded PDDL information, a graph representation of the information is needed. With such a graph an A* algorithm to find optimal plans.

2.3 PDDL ARCHITECTURE

To come up with a better understanding of Planning Domain Definition Language (PDDL), some knowledge regarding planning is needed. Planning is a process of judgment about activities required to accomplish a goal. PDDL is a way of expressing planning-based problems in a computer parsable representation. It was firstly developed by Dre McDermott in 1998 and motivated by STRIPS unique problem construction (Younes, 2005). The first official version launched was PDDL 1.2, where it separated the model of the planning problem in two parts, first the domain description and second the problem de-scription. The domain description was initially uniquely named and it consists of requirements where a declaration of all model elements is given 2.1 followed by predicates. Predicates defined in a domain have no basic meaning. The predicate section denotes only what are the names of predicates used in the domain and their composition. The idea of a predicate is to encode in relation to a composition of entities that it is valid(true) or not valid(false). Its relationship to other predicates is determined by the effects that possible actions in the domain apply to a given state. Which actions are considered, is based in the initial state of the problem definition (McCluskey, 2003). Actions have parameters that denote entity place holders to be used with objects, preconditions, and effects. Triggering the actions depend on the used parameters and preconditions.

They are used to handle entities and can introduce a changed predicate in the resulting state when an action is selected for the application. The preconditions of actions are expressed as logical propositions constructed from predicates and argument terms with logical connectives.

```
   (define (domain DOMAIN_UNIQUE_NAME)
 (:requirements [:strips] [:equality] [:typing] [:adl])
 (:predicates (PREDICATE_1_NAME ?A1 ?A2 ... ?AN)
              (PREDICATE_2_NAME ?A1 ?A2 ... ?AN)
        ...)

 (:action ACTION_1_NAME
   :parameters (?P1 ?P2 ... ?PN)
   :precondition (and/or (PRECOND_FORMULA1)
      (PRECOND_FORMULA2)
   :effect EFFECT_FORMULA
  )

 (:action ACTION_2_NAME
   ...)
    )
```

Listing 2.1: Sample Domain

The problem description is again uniquely named, even though the name is not used. It helps with the description of tasks that are required to be solved for the domain. The problem consists in declaring all entities used, initial conditions init and the definition of the goal state 2.2.

```
    (define (problem PROBLEM_UNIQUE_NAME)
(:domain DOMAIN_UNIQUE_NAME)
(:objects X X1 X2 OBJECT1 X_N)
(:init V1 V2 V3 V_N)
(:goal FORMULA_CONDITION)
)
```

Listing 2.2: Sample Problem

As mentioned, a planning problem is constructed by the compilation of a domain description with a problem definition. However, the same domain can be used with many different problem definitions to generate other plans in the same domain.

Although PDDL is based on STRIPS formalism, the language itself extends beyond that. The extended syntax and logic add the capability to express a type structure for the objects in a domain allowing to apply typing the parameters that appear in actions and constraining the types of arguments of predicates. The syntax also allows actions with negative preconditions and conditional effects. It gives the capability to express quantification in both pre-and post-conditions (Helmert M. , Concise finite-domain representations for PDDL planning tasks, 2009). The above mention extension is known as ADL.

ADL was introduced in PDDL version 3.0 where it assisted as a big development in the planning technology (Tim, BRengle and Cunniff Ross, 2016). This evolution was followed up by the latest version of PDDL 3.1 where it introduced object-fluent that allows the objects to be encoded as integers or it could be any type object also.

2.3.1 PDDL PLANNERS

Automated planning deals with the problem of finding a sequence of actions to achieve a given goal. This sequence is called a plan. A plan changes the given initial state of the world to a state satisfying a certain goal condition. With the help of PDDL, many planners were developed as for example: FF, LAMA, Graphplan, SAPA, HSP. These are some of the most known planners where each of them is equipped with a different algorithm implementation to find a plan. The planners developed are grouped into categories state-space planners and classical graph path finding.

The state-space planners are those planners that rely on search algorithms that operate on a search space. Where a search space is a subset of the state space. So each node created corresponds to a state of the world and each edge corresponds to a state transition, for example in each case that an action is triggered. The well-known planners in this category are HSP (Heuristic Search Planner), FF (Fast Forward), FD (Fast Downward) LAMA.

HSP was firstly introduced by B. Bonnet (Bonnet, Blai and Hector Geffner, 1998) and the final version was launched in 2001. This planner included basic search algorithms like breadth-first search, depth-first search, and best first search, where A* was used.

FF planner was based on the relaxed planning graph heuristic (Hoffmann, J. and B. Nebel, 2001). The main search technique is based on the idea of a hill-climbing search. The idea of this search algorithm is to be able to perform an exhaustive search only for the best states (Bonet, Blai and H´ector Geffner, 2001). This algorithm uses a heuristic to compute the successor states. Even, with heuristic and enforced hill-climbing FF is still incomplete and to guarantee the completeness it launches the A* when the hill climbing search fails. FF is known to be the most efficient planner in practice but it does not find always optimum solutions.

Fast Downward (FD) is another state-space planner presented by Helmert (Helmert M. , 2006). The algorithm itself was based in a greedy best-first search procedure, which is a modified version of classical best-first search with deferred heuristic evaluations. This approach was later used to create a new planner called LAMA where it is one of the most reliable planners in general use cases (Richter, S. and M. Westphal, 2010).

The classical planning resembles the problem of searching a path in a huge graph, where nodes describe states of the world and edges correspond to state transitions via actions. Then the planning task is to match a sequence of actions such that, it has initial states for each condition where is applicable to reach the goal state. To solve cost-optimization problems, a non-negative cost is given to each action and the task is to achieve a plan with the lowest cost.

The major distinction from classical pathfinding is that the state space plan-ner in practical applications is memory heavy. Hence a logical representation of actions, states, and state transitions is required. In the domain model, a state is represented as a vector of attribute values. Actions which are affecting values of certain variables through effects while demanding values of some attribute variables as preconditions. In a structured representation, a state model de-scribes objects possibly with attributes as well as relations between the objects. Action models based on first-order logic are similar to this representation, but the sources gained (Strobel, Volker and Alexandra Kirsch, 2014) show a lack of used planning domain modeling language based on a structured representation that leads to efficient and capable planners.

The KSH planner has been proposed to (module) supports both factored and structured representations of states, where states are represented by any term. It is based on the use case and derived requirements from conceptual design to a software implementation of a functional prototype.

2.3.2 PDDL VALIDATOR

PDDL is a tool to encode domain description and problem definition. Automated planners were developed to solve the encoding problem and to provide a solution for the description. With all the automated planners proposed, since the PDDL launched (more than 5000), it was seen as essential to consider a validator tool to assist on confirming that the plan was following the correct logical path, in other words, was checking if the algorithm was working properly. If the plan is finite the validation process is feasible. It could be easily validated by simulating the plan execution. When PDDL 2.1 was introduced, the validation becomes complex in this version numeric fluents and durative actions (which could have non-discrete lengths, conditions, and effects) were introduced (Howey, Richard, Derek Long, and Maria Fox, 2008). During the use of durative actions, a question was raised over whether a plan can be considered valid if it does not contain all of the effects of the actions included in the plan. When these cases occur, it was shown that plan validity could be compromised by ignoring the end effects of actions. After all the validator was a tool provided for PDDL contest competitors for checking their planner’s development. This was later improved by the community to handle these situations and being able to work on different plan representation (Howey, Richard, Derek Long, and Maria Fox, 2008). The official tool used in PDDL competitions was a library named VAL.

The use of VAL is critical for understanding the structure of large plans, with its visualization and reporting facilities. VAL can report a flaw of the plan and then the human can try to fix their logical errors. It also supports this process with suggestions to fix the plan. Then with the assist of VAL, a human can reprogram the planner and try again, completing a mixed initiative planning cycle.

CHAPTER 3

ANALYSIS

3.1 USER MODELLING & SCENARIOS

The persona method is often seen as a usability method, but, personas are more of a design method covering all aspects and all of the phases of a development project (Nielsen, Lene and Kira Storgaard Hansen, 2014). Working with personas requires a broad understanding of the user's lifeworld. Thus, when gathering data, detailed and tailored information must be extracted to serve with a purpose. The persona method tries to break with the automated perception and instead create empathic descriptions of the user.

With the version of personas presented here, I attempt to represent the needs of the individual to live with dementia as well as the needs of the caretaker to develop requirements around which the specific solution was implemented. Defining a persona aids the system design process since the allocation of requirements plays a huge role in establishing an overall software architecture. Hence, the system's functionality requirements were primarily based on the needs of the individual with dementia with respect to the needs of the caretaker. The caretaker is primarily responsible for the individual, but he is not playing a role in the system. Therefore the person with dementia is identified to be the actual user of the developed solution (Mart´ın, Estefan´ıa, Pablo A. Haya, and Rosa M. Carro, 2013). In accordance with the specific needs found in the literature of people who live with dementia, their approaches in AAL, a Personae is created and is represented as following by the name Jens.

Table 3.1: Personae Jens

Jens	Male
Social Information	Widowed & No Children
Age	68
Job (current/past)	Retired construction worker
Clinical Condition	Early stage dementia (Alzheimer's Disease)

Current Situation	Planning for long term care and possible future move to care home as the disease progresses
Physical abilities	Independently mobile, small risk of falls, back pain
Cognitive abilities	Short term memory loss, struggles to learn new skills, sometimes forgets what he was planning
Perceptual abilities	May struggle to distinguish between different objects where there is little contrast
Communication	Sometimes has difficulties figuring out the correct word
Built environment issues	Jens may find it difficult to locate the bathroom, kitchen and living room Jens faces difficulties in finding the corresponding switch for the lights Jens sometimes forget to take the medicine Puppet could assist Jens in finding the room. To help Jens to remember where is he and why he is there. The room should resemble a typical homelike and dining room as far as possible. To prevent falls and injuries, furniture such as tables and chairs should contrast well with the floor and walls.

Based on personae Jens, a living space of the user is designed as an extension of an intelligent assistance system (Ramos, Carlos, Juan Carlos Augusto, and Daniel Shapiro, 2008). It offers the system to pick up and interpret the signals of the human user's behavior useful for inferring their goals, intentions, affective states and needs. To model the user cognitive and affective state of the system, all the knowledge implied from persona is required about the user attitudes before planning the behavior to adopt in a given situation and to observe the user feedback to the system behavior. Since all the data regarding Jens is gathered, a simple living space is designed 3.1 for the purpose of a proof concept.

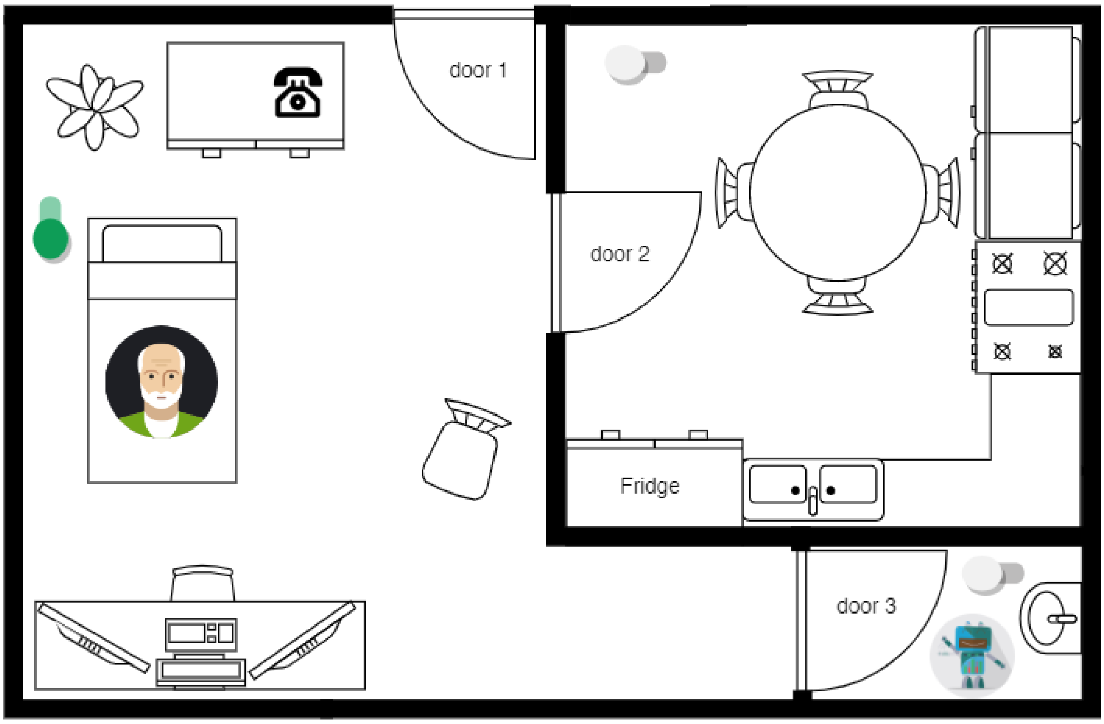


Figure 3.1: Visualizing Jens living space.

Following up the designed living space a use case is built, being someone who has taken up technology as a tool to combat their dementia progression and wants to have a reminder system which effects on their health. From the analysis of the collected data, I depicted the following scenario. However, in this scenario, we want to outline the importance of integrating the social aspects of taking care of a person with task-oriented assistance.

Jens is an old pensioner living alone in his apartment with equipped devices typical of an AAL system. He suffers from early dementia and has no children. Lucy is a social robot equipped with two grippers that has the role of taking care of Jens. Lucy, to perform its tasks, can take advantage of the planning system capabilities. For instance, Lucy may help Jens in his daily life. One of the most important actions of Jens during his daily activity is to eat, drink and to take the medicine. Nevertheless, to fulfill these actions of Lucy, some certain conditions must be accomplished (like switch on/off the light) in order to achieve the end goal result. Lucy's selected plan is made up of the following actions correspondent to the daily activities of Jens.

3.2 DOMAIN

To demonstrate capabilities of the KSH planner an implementation of a traditional planning benchmark domain (Assisting human) is given. The main criteria of tailoring this domain were based on the scenarios where the existence of natural knowledge of how humans would use to solve problems in this domain. The planning domain description is based on a collection of types: a collection of global objects (domain constants), a collection of predicates, a collection of functions, and a collection of action schemata. The use of predicated and functions is to encode state variables. On creation of a PDDL domain file, it is given an exclusive name when referring to the domain in problem definitions. During the implementation of the domain, research for other domains was implied due to creating a factored & structured representation of the domain. This is done for the main reason that PDDL 3.1 (latest version) uses a factored & structured representation of states, actions, preconditions, and effects. A factored representation is to re-formulate the problem as a set of smaller problems over the space, basically using the divide & conquer method (Howe, 2000). A structured representation helps in creating a proper workflow of the domain for each problem served.

Based on the use case, consider the following domain, where the robot can move around in a set of rooms connected by doors. Each room has a door which can be opened and closed. The doors can only be opened when the light is switched on. The robot is equipped with two grippers that allow him to make two actions. For each action made by the robot, a free griper is required. The robot has the possibility to move into rooms to assist the human. The actions that can be handled by the robot are providing the human the medicine required, however, in order to meet this requirement an accomplishment of a basic task must be done before reaching the end goal. By basic tasks, I mean switching the light on/off, and moving through rooms.

```
(define (domain lucy-assist-human)
(:requirements :strips :conditional-effects
   :negative-preconditions :equality :adl)
```

Listing 3.1: Domain declaration.

In the listing 3.1 is presented the header of the domain file where a unique name is given. In order to use all the functionalities for the domain the commonly used are :strips which are the basic subset of PDDL, :equality which allows the use of equal sign "=", :requirements which is the basic starting point with strips ,

:negative-preconditions from the declaration itself allows the use of not to negate and: ADL which offers conditional effects with an upgrade of strips.

```
1     (:predicates
2     (is-lucy ?lucy)(is-human ?human)
3     (is-room ?room)(is-medicine ?medicine)
4     (is-water ?water) (is-glass ?glass)
5     (is-shellf ?shelf)(is-fridge ?fridge)
6     (is-light-in-room ?room)
7     (is-gripper ?grip)(is-free-gripper ?grip)
8     (is-human-free ?human)
9     (is-in ?anything ?any)
10    (is-inside-fridge ?medicine ?fridge)
11    (are-linked-by ?room-1 ?room-2 ?door)
12    )
```

Listing 3.2: Predicates.

In the above PDDL code 3.2 are listed all predicates used in the domain file. As mentioned earlier the PDDL is an open world assumption so all these predicates are presumed true. The is-in predicate in line 9 is used to describe generally if any object is linked with another object. Later the use of the predicate will be described, and it will be easier to realize the use of it.

```
(:action lucy_changes_room
    :parameters (?lucy ?room-1 ?room-2 ?door ?grip)
    :precondition (and
      (is-lucy ?lucy)(is-in ?lucy ?room-1)
      (is-gripper ?grip) (is-free-gripper ?grip)
      (is-door ?door)
      (is-room ?room-1)(is-room ?room-2)
      (are-linked-by ?room-1 ?room-2 ?door)
)
:effect (and ;a conjuction of lucy and room effects
      (is-in ?lucy ?room-2)
      (not (is-in ?lucy ?room-1) ) )
)
```

Listing 3.3: Lucy changes room.

In the action of lucy changing room 3.3 are required four parameters to be declared that will take actions. Every parameter is declared, and preconditions are required that rooms are linked and lucy is inside a room. The effect will take place only and only if preconditions are matched and lucy will change the room with the use of negated equality.

```
(:action lucy_switch_lights_on
    :parameters (?lucy ?room ?grip)
    :precondition (and
      (is-gripper ?grip) (is-free-gripper ?grip)
      (is-lucy ?lucy)(is-in ?lucy ?room)
      (is-room ?room)
      (not (is-light-in-room ?room) ) )
    :effect (is-light-in-room ?room) )

(:action lucy_switch_lights_off
    :parameters (?lucy ?room ?grip)
    :precondition (and
      (is-gripper ?grip) (is-free-gripper ?grip)
      (is-lucy ?lucy)(is-in ?lucy ?room)
      (is-room ?room)
      (is-light-in-room ?room)  )
    :effect (not(is-light-in-room ?room)) )
```

Listing 3.4: Lucy switch lights.

For lucy to make certain actions lights should be switched on 3.4 . This action requires two parameters room and lucy. As precondition is required that lucy should be inside a room. To achieve the subgoal, the room should not be lightened otherwise the subgoal is reached and the action will not take place. The same logic is applied to the action of switching off the lights. The above-mentioned actions are the main core to make the goal feasible.

```
    (:action lucy_picks_glass_in_shelf_with_gripper
    :parameters (?lucy ?grip ?glass ?room ?shelf)
    :precondition (and
      (is-lucy ?lucy) (is-in ?lucy ?room)
      (is-gripper ?grip) (is-free-gripper ?grip)
      (is-shelf ?shelf)(is-glass ?glass)
      (is-room ?room) (is-in ?shelf ?room)
      (is-in ?glass ?shelf) (is-light-in-room ?room)
    )
    :effect (and
      (not (is-free-gripper ?grip) )
      (not (is-in ?glass ?shelf) )
      (is-in ?glass ?grip)
    )
)
```

Listing 3.5: Lucy picks glass.

The action above 3.5 is describing a feature of lucy of being able of getting the glass from the shelf. To reach this subgoal lights must be switched on and a free gripper to handle the glass. Glasses are placed inside the shelf and this is initialized in the problem description.

```
(:action lucy_picks_water_in_fridge_with_gripper
:parameters (?lucy ?grip ?water ?room ?fridge)
:precondition (and
  (is-lucy ?lucy)(is-in ?lucy ?room)
  (is-room ?room) (is-gripper ?grip)
  (is-fridge ?fridge) (is-water ?water)
   (is-light-in-room ?room)
  (is-in ?fridge ?room)(is-free-gripper ?grip)
  (is-inside-fridge ?water ?fridge))
:effect (and
  (not (is-free-gripper ?grip) )
  (not (is-in ?water ?fridge) )
  (is-in ?water ?grip)
))
```

Listing 3.6: Lucy picks water.

For the action above 3.6 the same logic is used as similar as previous of lucy picking glass 3.5 where water is placed in the fridge. Must be realized that Lucy already has one gripper busy by the glass that it was already picked up from the shelf.

```
  (:action lucy_pour_water
 :parameters (?lucy ?grip-1 ?grip-2 ?glass ?water ?room
    ?fridge)
 :precondition (and
  (is-lucy ?lucy)(is-water ?water)
  (is-gripper ?grip-1) (is-gripper ?grip-2)
  (not(is-free-gripper ?grip-1))
  (not(is-free-gripper ?grip-2))
  (is-room ?room)(is-light-in-room ?room)
  (is-in ?water ?grip-1)(is-in ?glass ?grip-2)
 )
 :effect (and
  (is-free-gripper ?grip-1)
  (is-in ?water ?glass))
)
```

Listing 3.7: Lucy pour water.

To jump in the process of pouring water lucy must acquire two main preconditions 3.7 having water in one grip and glass in the other one. In these circumstances, lucy has both gripers not free where is obliged to pour water in the glass. This effect is achieved with is-in precondition and offers a free grip for the next upcoming actions.

The action listed 3.8 provide human with a glass of water. It requires human to be free since some actions are planned to keep the human busy. The main preconditions are at line 8: when it is required that the glass is in the grip and the glass already contains water so this action takes place only after the pouring process 3.7.

The process of picking the medicine is 3.9 requesting a free griper and to have lights switched on in the room. The medicine is placed in the fridge and lucy can pick it up with the free gripper.

The end goal is to assist human 3.10 with the medicine and in order to achieve this goal state, the preconditions to be matched are: that lucy should have picked the medicine from fridge 3.9 and the human should have been busy since he already received the glass of water.

```
1   (:action lucy_gives_water_with_glass_to_human
2   :parameters (?lucy ?grip ?water ?human ?room ?glass)
3   :precondition (and
4     (is-lucy ?lucy)(is-in ?lucy ?room)
5     (is-gripper ?grip)(not (is-free-gripper ?grip) )
6     (is-human ?human) (is-in ?human ?room)(is-human-free
        ?human)
7     (is-room ?room)(is-light-in-room ?room)(is-in ?glass
        ?grip)(is-in ?water ?glass)
8     )
9   :effect (and
10    (not (is-human-free ?human) )
11    (is-free-gripper ?grip)(is-in ?water ?human)
12  ))
```

Listing 3.8: Lucy gives water.

```
(:action lucy_picks_medicine_in_fridge_with_gripper
:parameters (?lucy ?grip ?medicine ?room ?fridge
    ?human ?water)
:precondition (and
  (is-lucy ?lucy)(is-room ?room)
  (is-gripper ?grip)(is-free-gripper ?grip)
  (is-in ?lucy ?room)  (is-light-in-room ?room)
  (is-in ?fridge ?room)(is-medicine ?medicine)
  (is-inside-fridge ?medicine ?fridge)
)
:effect (and
  (not (is-free-gripper ?grip) )
  (not (is-inside-fridge ?medicine ?fridge) )
  (is-in ?medicine ?grip)
) )
```

Listing 3.9: Lucy picks medicine.

```
(:action lucy_gives_medicine
:parameters (?lucy ?grip ?medicine ?human ?room)
:precondition (and
  (is-lucy ?lucy)(is-in ?lucy ?room)
  (is-medicine ?medicine)(is-in ?medicine ?grip)
  (is-gripper ?grip)(not(is-free-gripper ?grip))
  (is-human ?human) (is-in ?human ?room)
     (not(is-human-free ?human))
  (is-room ?room)(is-light-in-room ?room) )
:effect (and
  (not (is-in ?medicine ?grip))
  (is-human-free ?human)(is-free-gripper ?grip)
  (is-in ?medicine ?human)
  ) ) )
```

Listing 3.10: Lucy gives medicine.

3.3 PROBLEM

as described a planning problem is constructed by the compilation of a domain with a problem description. The problem description is uniquely named even-though that it is not used, it helps with the description of tasks that are required to be solved for the domain. Since the relation of the domain with the problem can be one to one or one to many, constructing a problem is inferred by the user requirements. Based on the sketched living space of the personae Jens 3.1 the problem description is more feasible.

The problem descriptions consist of defining the problem name which in the case is named lucy-assisting-problem which is the purpose of the robot. This problem is linked with the domain defined as lucy-assist-human 3.1. In the scope of the problem, static and dynamic objects are generated. Static objects are rooms and doors for the main reason that they will never change the state but are used for the only purpose of describing the situation and later used as connections to jump from rooms. As for dynamic objects are objects that change the states. PDDL is an open world assumption(list of state variables are initially true) sometimes the objects mentioned 3.12 should change states in order to achieve the task for the final goal.

```
(define (problem lucy-assisting-problem)
        (:domain lucy-assist-human)
        (:objects
          ; static
          room-1 room-2 room-3
          door-1 door-3 door-2
          ; dynamic
          human
          lucy
          grip-1 grip-2
          medicine water fridge
          glass shelf
          )
```

Listing 3.11: Problem Declaration.

For this reason, in the init(initialization) section, the declaration of the static and dynamic object is required to describe the situation. In this section variables are divided in two groups: static variables are and dynamic variables. Static variables are used only to describe the situation and cannot change the state where dynamic variables can change the state depending on the effects of the actions. Inferred from the use case static objects initialized are three rooms and two doors where doors are used for connecting rooms with each other. This link is done two-sided which offers the robot to enter and exit by the same door. In the dynamic section, each room is equipped with a switch where it can give the possibility to lucy to control lights. However, a condition is made that the room has lights where the human is present. The robot or Lucy is equipped with two grippers where both of them are free and the starting point of it is in room03. The fridge is placed in room02 (kitchen) with shelf as well. Medicine is located inside the fridge and glass inside shelf where both are located in room02. The final desired goal is described in the goal section where the robot must assist the human with providing the medicine and come back in the starting position with both grippers free (Hoffmann, 2003).

```
(:init ;this is the initial state of the scenario
 ;static variables
 (is-light-in-room room01)
 (not (is-light-in-room room02))
 (not (is-light-in-room  room03))
 (is-room room01)(is-room room02)(is-room room03)
 (is-door door02) (are-linked-by room02 room01 door02)(are-linked-by room01 room02 door02)
 (is-door door03) (are-linked-by room03 room01 door03)(are-linked-by room01 room03 door03)
 ;dynamic
 (is-lucy lucy)(is-in lucy room03)
 (is-gripper grip1) (is-free-gripper grip1)
 (is-gripper grip2) (is-free-gripper grip2)
 (is-human human)  (is-human-free human) (is-in human room01)
 (is-fridge fridge) (is-in fridge room02)
 (is-shelf shelf) (is-in shelf room02)
 (is-glass glass) (is-in glass shelf)
 (is-water water)    (is-inside-fridge water fridge)
 (is-medicine medicine)    (is-inside-fridge medicine fridge)
)
(:goal
 (and
 (is-in medicine human)
 (is-free-gripper grip1)
 (is-free-gripper grip2)
 (is-in lucy room03)
   )
 )
```

Listing 3.12: Problem Declaration.

CHAPTER 4

SOLUTIONS

4.1 PDDL TO OBJECT ORIENTED PROGRAMMING (OOP)

Ambient Intelligence is characterized by a heterogeneous and highly dynamic infrastructure. Developing this type of infrastructure requires an object-oriented language representation (Dedecker, 2008). Many technologies developed in the field of AAL, like the wrist band developed in Fraunhofer and Silver Mother are using an object-oriented approach. The developed solution in this work is an integration of PDDL with Java OOP language. To achieve this, we use the Pddl4j library.

Pddl4j is an open source library providing parsing, of a PDDL domain and problem. The main goal of the library is to aid the development phase of new planners and techniques in the planning community. Pddl4j offers all the mandatory tools to work with PDDL language and a Java API to design new algorithms. The Pddl4j unique construction is based on several independent modules. It provides a PDDL parser, that already has been validated on all the available benchmarks of the International Planning Competitions. The implementation of the parser is achieved through the JavaCC library, which is a tool that reads a grammar specification and converts it to a Java program. This then can recognize matches to the grammar and parse it. It also has good error reporting, which is very useful for debugging.

When the parsing process occurs, pre-processing modules are called to instantiate module that then is executed to improve the parsing. The parser has also an instantiation module that transforms the operators of the planning domain into base actions. That leads to the main reason why Java language is chosen. Nonetheless, Java support is extraordinary, and the language is highly performable compared to other OOP languages. The second main key is that the Java platform is independent of the computer and operating system architecture which ease the integration of the developed solution with other platforms like Puppet.

A parser is acting as a decoder of data representation to a computer understandable form. The parser of Pddl4j provides a semantic and syntax analysis of the problem and domain files. Which are transformed into an internal representation by the instantiation module. an extension is introduced based on JavaCC, which is the same tool used by Pddl4J.

The parser has two important layers that verify that the semantic can be converted, or meaningful errors and warning can be given. The main use of warnings is to inform the user, in order to provide a more resembled structure. Some warning may be variables declared but not used, domain name declared is not matching with the problem file and a predicate or a function never used in the domain or problem file. a Warning still allows domain and problem description to be converted. Warning issues will not affect the overall problem. as for errors, these are critical issues on parsing the lexical structure. Errors indicate that the domain and problem do not fit the requirements of the semantic. To mention some of them, a conflicting type hierarchy, any syntax error of mis-placement of tokenization characters (brackets, etc.), a function or predicated used but not declared, wrong typing of the lexical semantic used and etc.

As described earlier in this chapter Pddl4j is the base of this thesis. During the development of the domain description, object fluency was a feature that the translator was lacking. This feature was allowing functions or preconditions to act as objects or integers. This new feature was introduced on the latest version of PDDL what is not in Pddl4j. Based on this problem an update of a part of the lexical structure was done with the needed was done in this work. To achieve this extension the following changes were applied:

First, the process of parsing substitutes all fluents and modules by replacing their parameters with all applicable objects of the related types. Each substitution has a unique mapping to a different variable. For instance, in the domain description of this work is declared a boolean fluent is-in that has two parameters of the type objects. In the end, the substitution covers fluency with the introduction of new predicates. Furthermore, the problem file specifies two objects: obj1 and obj2. In this case the parsing function produces the following constraints and assigns each of them to a separate variable:

```
variable one : \textit{is-in obj1 obj1}

variable two : \textit{is-in obj1 obj2}

variable three : \textit{is-in obj2 obj1}

variable four : \textit{is-in obj2 obj2}
```

Listing 4.1: Fluency declaration.

Using the given example, a representation of the parameters is determined as an object and integer concurrently. This information can be distinguished in the first variable where obj1 is acting as an object and integer simultaneously. For each variable, a translation table that encodes its content as real value is generated. Boolean fluents with the value 0 represents true and all other values false. as for numeric fluents, it is not needed to provide a table since their numeric value is directly used. The enumeration of all objects of a fluent type is achieved with 0 for the first object, 1 for the second and n-1 for the nth object. Negative indexes are represented as an invalid object. The same schema is used for actions.

after the parsing process, it constructs a compact internal representation of the planning problem. That then is used by the instantiation module. The main goal of the instantiation module is to convert the operators of the planning domain into base actions 4.1. In the instantiation phase, a slight modification of the module was done to on reducing the search space. The instantiation modularization into different formalisms improves the solver quality (Sanchez Nigenda, 2018). So, the conditions and effects of the actions are applied to the tabular system which is referred to as mapping. This process is followed up by an iteration of effects through all actions. all these modifications are used to reduce the search space of the planning and therefore improving the search performance.

after the final process of parsing, the resulted mapping is written into a file. The output file contains the variables mappings for all used terms, the initial states, and the goal condition. as for proof of concept, the names of the actions are written into that file to provide a human-readable format. Resulting in a reduction of the search space leads in a faster outcome.

4.2 HEURISTIC SEARCH

With the launch of PDDL, a lot of planners were based on heuristic search methods and it was the proper approach to chase. Planners based on heuristic search have always won in the International Planning Competition (Lian, 2012) (Coles, 2012). Nowadays heuristic search planners have become a "must" use approach to find solvable plans in large deterministic search space. With the use of heuristics, the planning model was improved in terms of scalability and reliability. In theory, heuristic functionality is to guide the search by using a function heuristic h that tries to estimate the cost h* of an optimal path for a starting state s to an end goal g. In other words, a procedure of heuristic makes use of h to select which pending of states' s to expand first. Usually, the expand happen on the smallest value calculated by h(s, g). For the algorithm to be functional the heuristic must be admissible. The heuristic is admissible if it never overestimates the optimal cost h* from a starting point to an end goal. A typical idea to assume heuristic is done by ignoring some features of the original problem to obtain a simplified problem. This idea is known as heuristic relaxation. With the improvement of PDDL, many challenges were emphasized regarding heuristics and the scope of this topic was expanding fast. In order to narrow down, this problem heuristics were grouped in four main categories based on methods and automatic generation (Katz, Michael and Carmel Domshlak , 2008).

Landmarks heuristics (Richter, S. and M. Westphal, 2010) are focused on reducing the search space by observing some propositions that are true at some point. Abstraction heuristics (Korf, Richard E., Michael Reid, and Stefan Edelkamp, 2001) (Helmert M. , Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces, 2014) (Katz, Michael and Carmel Domshlak , 2008) tries to reduce the size of the search space by collapsing several states into one. It is more feasible to find an optimal solution when the abstraction is small enough.

The idea behind delete relaxation heuristics (Hoffmann, J. and B. Nebel, 2001) (Domshlak, Carmel, J¨org Hoffmann, and Michael Katz, 2015) is to ignore the negative effects of actions to accomplish the estimation of the goal state. The additive heuristic ignores the negative effects of the actions and it approximates the distance form a state s to a state g.

To conclude with critical path heuristics, (Bonet, Blai and Hector Geffner, 2001a) (Bonet, Blai and H´ector Geffner, 2001) are estimating the goal distance by computing a lower bound estimate on the cost of achieving sets of facts of a predefined size. all the above-mentioned heuristics are admissible, however, based on inter-national planning competition 2013, the critical path heuristics are known to be the fastest in generating a sufficient output response. The critical path heuristic (Haslum, Patrik and Hector Geffner, 2000) his determined by identifying the longest max stretch from the starting point of a particular subset to the end goal node and estimating the time required to visit the nodes. An example is designed to have a better understanding of the use of heuristic.

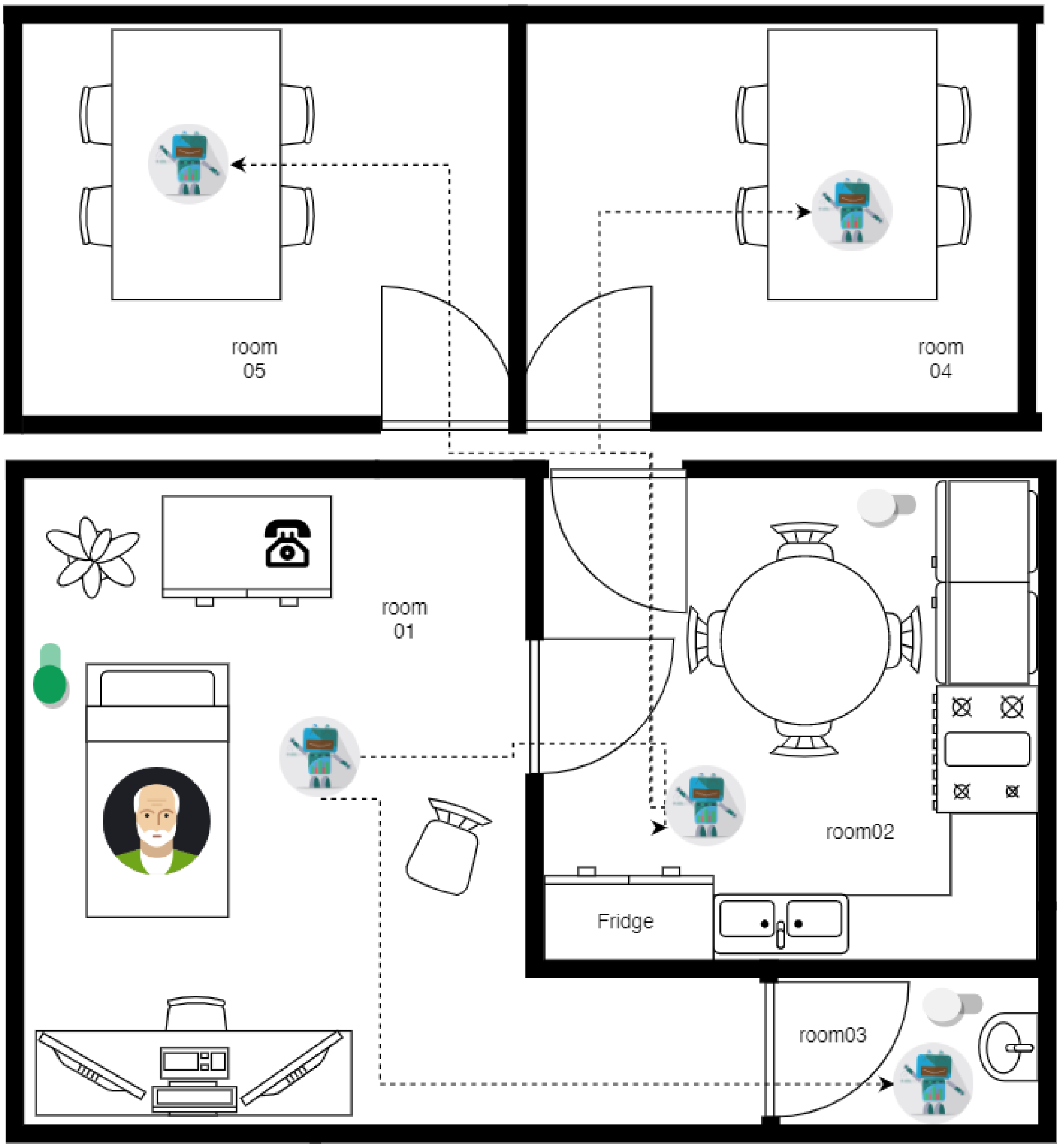


Figure 4.1: Visualization of a problem.

In this example the goal of the robot is to visit room five where the starting point is room one. Through common sense, the calculated heuristic from initial state to goal state h (I, G) is 2. So, the assumption unit cost is 2 but in " the most costly subgoal" we may use size > 1! To understand this process of approximation, it is easier if a reversing approach of the steps done is used to accomplish the goal. This process is called regression-based characterization of h* and is denoted by r*. For this example, a problem and a domain is provided where STRIPS are being used as planning task. This is examples is named as "heuristic Examples" where it contains:

Definition 1 (Piece of knowledge) is a specific statement about the environment, the user, or devices in the environment. a statement can express the process in a task, the assignment of values or any other context information including raw or interpreted sensor data. A piece of knowledge is denoted by k.

Definition 2 (a State) is described by a set of pieces of knowledge of specific context that is relevant for the user's situation. It can be written as: pn = k1, k2, ..., kn. The space of all possible states is denoted as P with pn ∈ P.

Definition 3 (The Initial State) P initials the current valid set of pieces of knowledge k describing the user situation where P initial ∈ P.

Definition 4 (a Goal) G is defined as the desired state that represents the users wish and side conditions that can be defined by an external person. Therefore, G is a set of pieces of knowledge that should be fulfilled when the user's wish is reached and can be denoted as G=k1g, k2g, kNg.

Definition 5 (an Edge cost) is the heuristic estimated cost value of an action that connects two related states. This relation is denoted as c = cost of a| {a ∈ a}where a is member of a and a is representing a set of executable actions for the domain.

Definition 6 (a Subgoal) is a goal that aids on reaching the final end goal where gn is a subset of G and it can be written as gn ⊆ G

Definition 7 (a starting point) is the starting state for a subgoal where it is denoted by s where s ⊆ P and s, g ∈ Pn The perfect regression heuristic∗ for this examples is the function r∗(start):=r∗(start, goal).

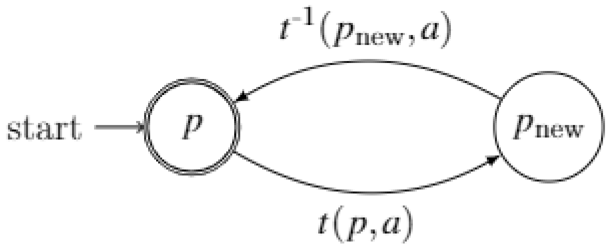


Figure 4.2: Transition function

Definition 7.1 (: =) it means that the function on the left-hand side is being defined to be what is on the right-hand side. The perfect regression formula stands for the cost of achieving a subgoal g with the best-case scenario and this process is known as a transition system.

Definition 8 (Transition system) are some systems whose behavior can be represented as states. The system may move from one state to another in response to a transition function t(p,a).

Definition 9 (The transition function) t(p, a)is generating the new state p new when applying the action a in the state p. p_{new} =t(p, a)|p new, p ∈ P, a ∈ a. The reverse function is denoted ast-1(p_{new}, a) = p and it returns the state that has been visited before applying the action a. See figure below.

Definition 10 (a Partition) is described as a set of states, which describe a path from one statep1to a state pg and can be constructed as followed. Pn ⊆ P | Pn =p1, p2, ..., pg

Definition 11 (The minimum estimated value of heuristic) is the sum of all minimum costs from starting goal s to the final subgoal g. This is achieved with a recursion loop where regression is called until the minimum cost estimations are determined for the subgoal. The critical path heuristic h^1 for the function $h^1(s) := h^1(s; G)$.

$$r^*(s,g) = \begin{cases} 0, & \text{if } g \subseteq s \\ \min_{a\in a, t^{-1}(p_{\text{new}},a)|} := \; c(a) + r^*(s, t^{-1}(p_{\text{new}}, a) & \text{otherwise} \end{cases}$$

Figure 4.4: Critical Path Herustic h^1

$$h^1(s,g) = \begin{cases} 0, & \text{if } g \subseteq s \\ \min_{a\in a, t^{-1}(p_{\text{new}},a)} := \; c(a) + h^1(s, t^{-1}(p_{\text{new}}, a)), & |g| = 1 \\ \max_{g^1 \in g} := \; h^1(s, \{g^1\}) & |g| > 1 \end{cases}$$

Figure 4.3: Regression function

As denoted for a subgoal g, the use of regression r was the same for G. For subgoal set g use the cost of the most costly singleton subgoal which is part of the main goal. Where a stands for actions contributing to reach the goal and g^1 is a subgoal of g. $h^1(s;t^{-1}(p_{new}; a)$ is becoming the new goal to be achieved from action a to state s through the transition function. The following function describes the minimum and maximum cost of a critical path found for one subgoal. The critical path is the cheapest path cost related to the most costly subgoals g.

Going back to the problem described, a solution is given through heuristic search 4.1. a reduction of actions is done, to understand the usage of the heuristic. The robot can only move through rooms to reach a subgoal. The final subgoal in the given example is to visit room05. a visualization is given in the following graph.

$$cost(change - room) = \begin{cases} 1.5 & \{\text{lucy,room-a, room-b}\} = \{\text{lucy,01,03}\} \\ 1 & \{\text{lucy,room-a, room-b}\} = \{\text{lucy,01,02}\} \\ 2.5 & \{\text{lucy,room-a, room-b}\} = \{\text{lucy,02,04}\} \\ 3 & \{\text{lucy,room-a, room-b}\} = \{\text{lucy,02,05}\} \end{cases}$$

Figure 4.5: actions : change-room(lucy room-a room-b)

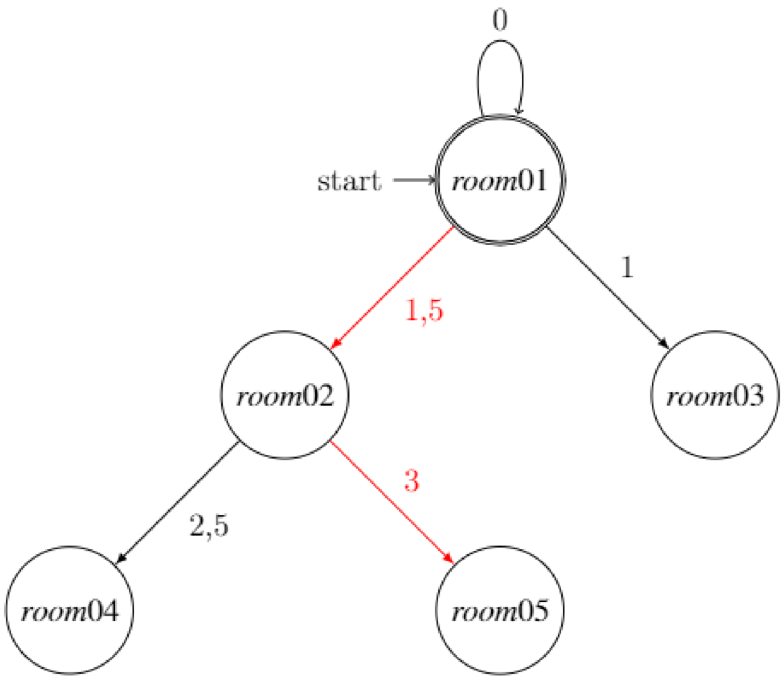


Figure 4.6: Critical Path heuristic h^1 representation through graph

The pseudocode written in the following section will describe a step by step solution with the usage of heuristic h^1. In order to determine the critical path value for a subgoal, the starting point will be lucy in room01 and the end subgoal will be for lucy to visit room05. This is a simplified version of the domain. In this lucy is only capable to use action: change room. Here subgoals (g) are used instead of Goals (G) due to the simplified domain and it is aiding in the final goal.

- Step 1 (Initial State) h^1 (Pinitial) = {at-room01}
- Step 2 (Goal State) g: = {at−room05and∀ri∈R visited}(ri):=true|i={1,2,3,4,5}
- Step 3 (staring position) s: ={at−room01} where h^1=0
- after declaring all the necessary information for the example given, the use of heuristic search with transition function is processed.
- Step 4 (Recursion 1) h^1(s; at room05) = 3 + change-room (room05, room02) = 3 + h^1(at room02).
- Step 5 (Recursion 2) h^1 (s, at-room02) = 1.5
- Step 6 (End Goal) h^1 (s, at-room05) = 3 + 1.5 = 4.5

$$min\ h^1(s, at - room02) = \begin{cases} 2.5 + h^1(s, at - room04) \\ 3 + h^1(s, at - room05) \\ 1.5 + h^1(s, at - room01) \end{cases}$$

As it is noticed the transition function is applied from room05 to room02 and the next starting point s = fat room02g. Furthermore h^1(I; at room04) = 2:5 + 1:5 = 4 and h^1 = (I; at room03) = 1. From the application of h^1 the critical path is= room01, room02, room05.

From the computation of h^1 the general case scenario is derived where: h^m is the function $h^m(s)$: = h^m (s; G). a stands for actions contributing to reach the goal g and h^m (s; $t^{-1}(p_{new}$; a) is becoming the new goal to be achieved from a to state s.

$$h^m(s,g) = \begin{cases} 0, & \text{if } g \subseteq s \\ \min_{a \in a, t^{-1}(p_{\text{new}},a)} := \ c(a) + h^m(s, t^{-1}(p_{\text{new}}, a)), & |g| < m \\ \max_{g^1 \subseteq g, |g| = m} := \ h^m(s, \{g^1\}), & |g| > m \end{cases}$$

Figure 4.7: Critical Path Heuristic h^m

h^m (s; g) is always admissible for a fine fixed m and it can be computed in polynomial time. The higher the value of m the more informative is the heuristic. With transition function, the heuristic is relatively slow to compute at each state. However, an admissible variant of h^m was presented by (Haslum, 2016). This computation is more efficient in time compared to the first launched version. an additive version of the heuristic h^m is an efficient way of cost relaxation and can be applied to any admissible heuristic for sequential planning, the main enhancement is the partitioning of a set of actions, where h^m is computed into other disjoint subsets. Then, for each subset, the h^m is separately computed for each subset and the sum of these independent computations is returned as the final heuristic value. Basically, is a use of the divide and conquer method. This method is not automated, and it is achieved only for the current domain. In this method, a problem is recursively split into smaller tasks to accomplish the final goal. More information will be provided in the following sections.

4.3 PARTITIONING

As mentioned in earlier to combine the tabular database with additive h^m actions must be partitioned.

Tabular database with pattern matching heuristics are based on problem abstraction: functions arrange states of the search space to partition it into a smaller space, with the property that the goal path is a finite abstract space (Culberson, Joseph C. and Jonathan Schaeffer, 1996). The process of abstraction is to achieve a re-duction of the level-of-detail of description and thereby to change the representation into a simplified version of the original. Then the heuristic outcomes are stored in a table which we will refer to as a tabular database for the pattern of the subset. By projecting a state and looking up the cost of the corresponding abstract state, stored in the table, we obtain a good approximation of a perfect heuristic.

The memory required to store the tubular database, and the time needed to compute it, grows exponentially with the size of the subset goal.

However, as shown by Edelkamp (Edelkamp, 2001), memory can be used more effectively by establishing abstractions level on multivalued variables. Such variables correspond to a particular type of invariant, the property that exactly one from a set of the subgoal is true in every reachable state. Methods for automatically extracting such invariants from STRIPS encodings have been propose (Fickert, Maximilian, Joerg Hoffmann, and Marcel Steinmetz, 2016) (Helmert M. , Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces, 2014).

Because h^m and tubular database rely on each other, a method is used for automatic selection of an action partitioning for the h^m heuristic and patterns for the tabular system heuristic. This method of action partitioning is to create one partition of each disjoint subsets for each subgoal and assign actions to the partition where they appear to contribute the most to the sum. To determine if an action is applicable for the subgoal, a comparison of the heuristic resulting from relaxing of the cost of action to the one that does not. Initially, each of the partition of each disjoint subset contains all actions. When an action is "selected " to one of these partitions, it is removed from all other partitions so at the end of the process when all actions are assigned the partitions are disjoint.

Since the partition is done only for the current domain and no experiment of it is done with other domains, I am assuming that the tabular database created is not automated for all the domains in the world. The resulting quality of heuristic often depends on the choices of the tabular database so they must be tailor to domain and problem description well for the heuristic to make efficient approximations.

4.4 KSH PLANNER

Sensing and acting provide links between an intelligent algorithm and the real world on which it operates. In order to make such algorithms responsive, adaptive, and beneficial to a user, a number of types of reasoning must take place. These include recognition and decision making or in other words a plan. To build such a plan by a computer system, a planner is needed.

In the planner developed, we have introduced a refinement to the algorithm. The additive h^m is defined by computing several instances of h^m, each counting the cost of only a subset of the actions in the problem while approximating the cost of renaming actions. Both tabular database and additive h^m have free parameters which need to set properly for producing good heuristic estimates. The method of selecting these parameters is done in the STRIPS encoding process and is tailored specifically for the current domain. The main reason for using domain depended approach is because this avoids the problem of selecting the wrong parameters.

For the desired application in abient-Intelligent systems, a refined planner is developed. This planner, the KSH planner, is developed for this work. In the following chapter, an evaluation of its efficiency and valid plan findings is done.

CHAPTER 5

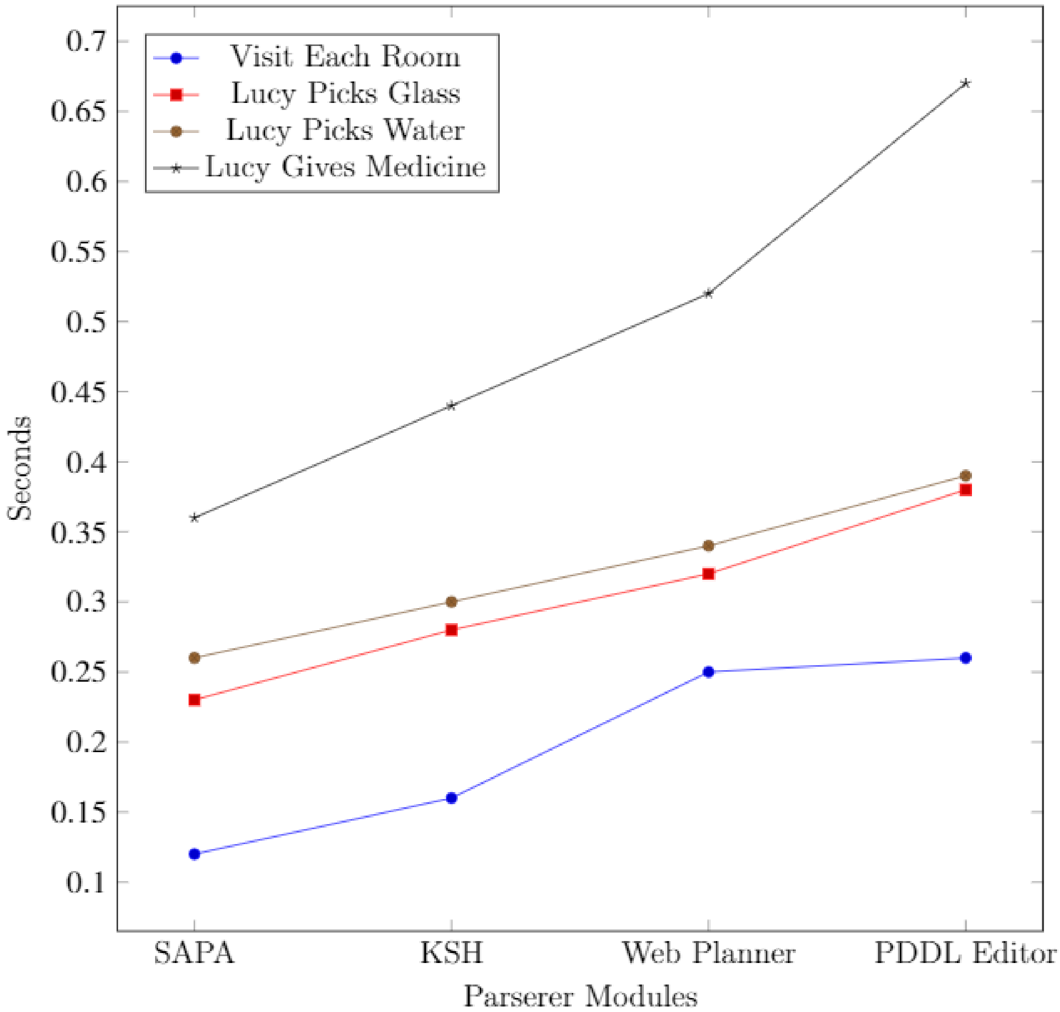
EVALUATION & VALIDATION

In this section, is described the comparison of KSH planner with three other planning platforms: Web Planner (Mauricio C. Magnaguagno, Ramon Fraga Pereira and Felipe Meneguzzi Martin D. More, 2017) which is based in Fast Downward(FD), PDDL editor & Online parser and SAPA (Do, M. and S. Kambhampati, 2003) which is based on Fast For-ward(FF). For each platform, an evaluation is done with the three main com-parable processes: the parsing process, the encoding module and the most important one searching process. The experiments were conducted on a MacBook Pro where 16 GBytes of memory were allocated to support the needs of the planners.

5.1 PARSING

The performances of the parsing modules are shown in the figure below 5.1. For every problem, the figure illustrates the average time in seconds needed by KSH, Pddl4j, SAPA, and Web planner to parse all the domains and problems files. SAPA has the most performant parsing module. Then others are Pddl4j, KSH and finally Web Planner with PDDL editor. The implementation of the parsing module of Web Planner written in Javascript and PDDL editor in python is definitely less performant than the implementation of Pddl4l / KSH based on Java CC. However, as it is noticed from the graph the parsing implementation average time is relatively small and not much comparable.

The approach of the parser module is the same as the Pddl4j library, for this reason, no comparison of it is done with Pddl4j.



Graph 5.1: Computational Time Of Parsing Module

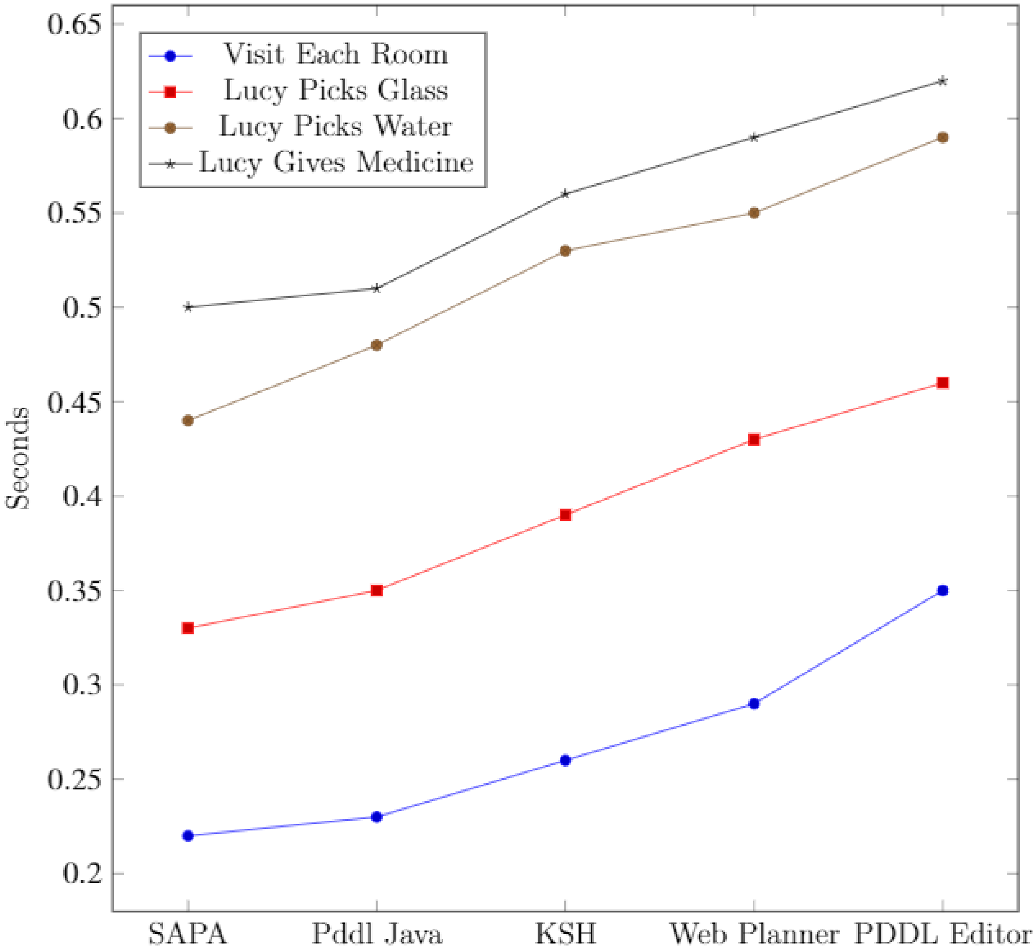
A comparison of Pddl4j is done with KSH since the planner translator is derived from Pddl4j parser. A small distinguish is given where still Pddl4j is achieving better results on parsing the files even though an introduction of new schematics was done for KSH. However, SAPA outperforms all the above-mentioned parsing modules.

5.2 ENCODING

After a successful parsing process comes to the encoding module that converts the translated files domain and problem description into a specialized format. The evaluation of the performance for the encoding process is compared by:

1. the average time needed to encode all the planning problems of a domain and
2. memory needed to store all the executed planning problems of a domain. The time needed to encode the planning problems remains relatively small (between 1 to 10 seconds for most of the domains).

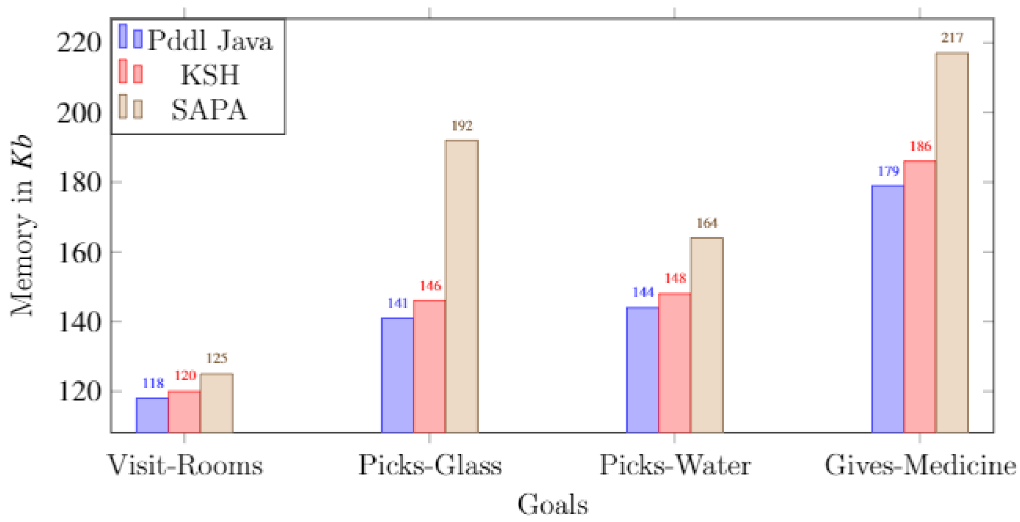
A comparison of Pddl4j is done with KSH since the planner encoder is derived from Pddl4j. A small distinguish is noticed from the graph 5.2 where Pddl4j is achieving better results on encoding the files. This feature is acceptable since the encoding process introduced earlier is using the tabular database and this is affecting negatively in time and memory. However, SAPA outperforms all the above-mentioned encoding modules.



Graph 5.2: average Time To Encode

The encoding module of Web Planner is much slower than KSH and SAPA in this order. Web Planner encoding module takes more time for the large problems (those identified in the evaluation of the parsing modules). The results above are based on the current problem description. Form the graph 5.3 it is noticed that the processing time of the encoding module of SAPA is outperforming Pddl4j and KSH.

Concerning the average memory used to store the encoded planning problems a test is done for the KSH and SAPA. During the memory test, KSH was able to perform better than SAPA especially in the final goal where many preconditions were requested to be matched. However, the encoding module of Pddl4j was less memory hungry compared to KSH & SAPA. as for two other platforms, it was difficult to fetch a result from other planners since they were web-based.



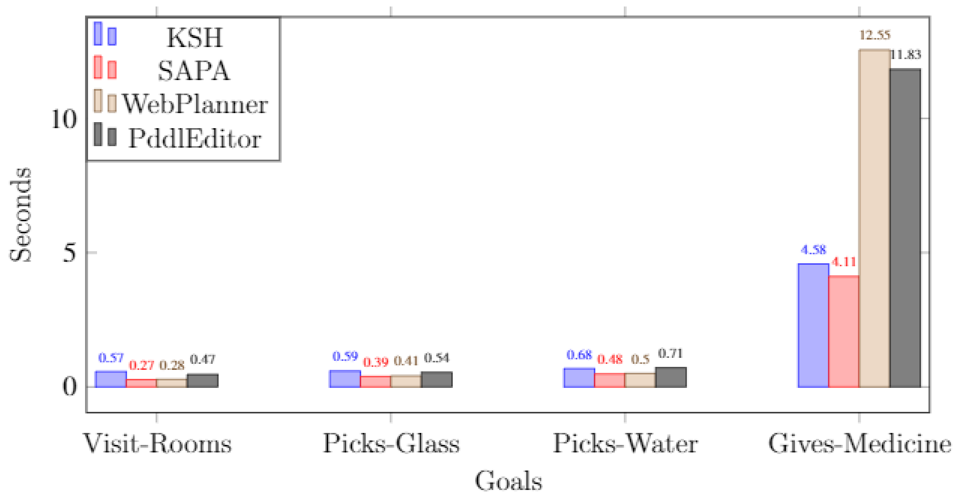
Graph 5.3: Encoding Memory Test

5.3 SEARCHING

The performance of heuristic algorithms for time optimization is often sensitive to the problem description. In some cases, a specialized heuristic algorithm may perform exceptionally well on a particular set of problems while fail to produce acceptable solutions on other problems. Such interesting cases are evident in algorithms which are developed for PDDL where planners are generated for a huge range of domain like vehicle routing, job scheduling, human interaction, etc. The testing and comparison of heuristic algorithms have been a subject of much discussion in recent years. A reasonable approach of algorithmic testing is to show that the proposed algorithm is performing better at least in one aspect than the current algorithm or platforms. In this section, we formulate an algorithm comparison an algorithm comparison is formulated for a variety of problems description of the domain. The results are described in the following graph 5.4.

Definition 10 (Plan costs) are the steps that the program should perform in order to reach the end goal desired.

During the evaluation process, we demonstrate that the performance of algorithms could be highly sensitive to problem instances. From the graph 5.5 is derived that for simple task with a lower plan cost(lower than 10) KSH is performing poorly compared with other platforms. The reason why KSH is performing badly in these tasks comes out from the tabular system and partitioning function. The tabular database needs more time to store the data for the nodes and to serve to the partitioning function. On the other hand, for complex tasks with a higher plan cost (higher than 10), KSH is resulting in a faster outcome and more efficiently. Here comes in handy the tabular system that already has some information about the visited nodes and when providing this information to partitioning the graph is constructed will fewer nodes. This results in a faster outcome.



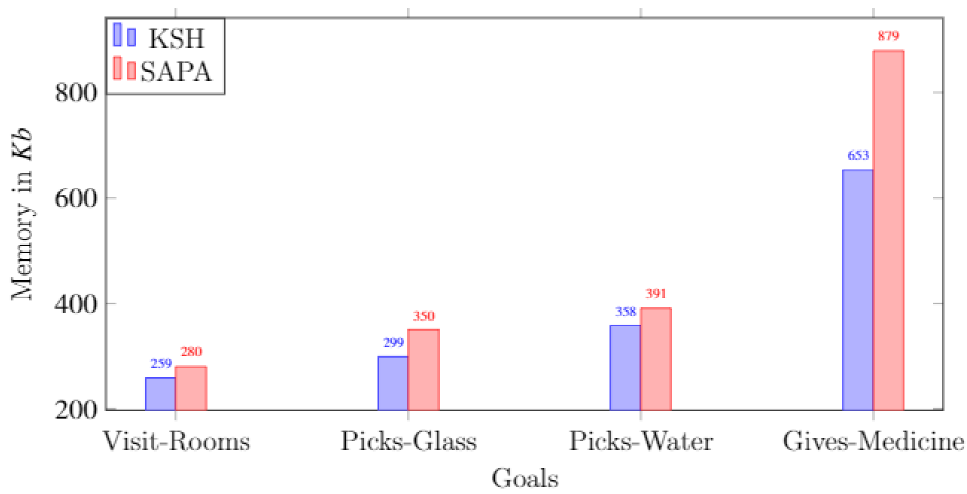
Graph 5.4: Search Test

However, this comparison is not decent since two platform Web Planner & PddlEditor are web-based services and no information about the processing power is gathered. A distinguish can be done for SAPA since it is running in the machine and a comparison of it can be more feasible.

The above-mentioned comparisons are done for one specific goal where only one task is required to be solved. To determine the efficiency of the algorithms, a combination of the goals to the final problem is done. The final goal is a combination of three subgoals:

1) Lucy is giving the human the medicine.
2) Lucy is returning to starting position after the medicine is given.
3) In the end, both grippers are free.

This combination of the above subgoals leads to a complete interesting out-put.



Graph 5.5: Searching Memory Test

SAPA which is based on FF is a well-known that the algorithm is able to generate the fastest output but providing an output in such timing, leads in wrong results. During the test done, it occurred that the planner was generating a wrong section of logical actions for the domain description 5.1. This is verified in the 5th and the 7th step of the planned output. Lucy is not able to pick the medicine the glass because this leads to a deadlock since both grippers are not free. To continue, the action of pouring water cannot be achieved if lucy will not have both grippers busy with glass and water.

Figure 5.1: Output of SAPA planner

Surprisingly, the computation time for the SAPA planner was 8 seconds 5.2

```
time spent:       0.12 seconds parsing
                  0.10 seconds encoding
                  0.04 seconds searching
                  0.26 seconds total time

memory used:      0.13 MBytes for problem representation
                  0.02 MBytes for searching
                  0.14 MBytes total


Planner found 1 plan(s) in 8.237secs.
```

Figure 5.2: SAPA computation time

The Web Planner which is based on FD was taking 19 seconds to solve the problem and the generated output was not able to determine the correct actions to fetch the final goal 5.3. The Web Planner was facing the same logical problem with a distinguish in step 11 where lucy picks water in the fridge. Still, the outputted result is:

```
Result: SUCCESS
Domain: lucy-assist-human
Problem: lucy-problem1
Plan:
  (lucy_switch_lights_on lucy room03 grip1)
  (lucy_changes_room lucy room03 room01 door03 grip1)
  (lucy_changes_room lucy room01 room02 door02 grip1)
  (lucy_switch_lights_on lucy room02 grip1)
  (lucy_picks_medicine_in_fridge_with_gripper lucy grip1
  (lucy_picks_water_in_fridge_with_gripper lucy grip2 wa
  (lucy_pour_water lucy grip2 grip1 medicine water room0
  (lucy_changes_room lucy room02 room01 door02 grip2)
  (lucy_gives_water_with_glass_to_human lucy grip1 water
  (lucy_changes_room lucy room01 room02 door02 grip1)
  (lucy_picks_water_in_fridge_with_gripper lucy grip1 wa
  (lucy_changes_room lucy room02 room01 door02 grip2)
  (lucy_gives_medicine lucy grip1 medicine human room01)
  (lucy_changes_room lucy room01 room03 door03 grip1)
Execution time: 19.2911s
```

Figure 5.3: Results of Web Planner

The PDDL editor was complaining about an error, however, it was able to find a plan of 14 steps to follow but no output was generated through the process because of the error complain 5.4. The reason is yet not known.

```
Total time: 1.552
Nodes generated during search: 11469
Nodes expanded during search: 169
Plan found with cost: 14
BFS search completed
```

Figure 5.4: PDDLeditor output report

As it is seen from the figure, a syntax error in line 414 was denying the output. On the other hand, in the domain description are written 202 lines of code and in the problem description 47 lines 5.5. Not to mention that we are using the exact same domain and problem description in other platforms.

```
Failed to parse the problem -- invalid syntax (, line 414)

 --- OK.
 Match tree built with 1022 nodes.

PDDL problem description loaded:
        Domain: LUCY-ASSIST-HUMAN
        Problem: LUCY-PROBLEM1
        #Actions: 1022
        #Fluents: 28
Landmarks found: 4
Starting search with IW (time budget is 60 secs)...
rel_plan size: 12
#RP_fluents 18
```

Figure 5.5: PDDLeditor error report

The KSH was able to generate the correct order of steps for lucy to follow. The algorithm needed only 5.2 seconds to solve the problem 5.6. The following output is the correct logical path of actions for Lucy to reach the final goal.

```
found plan as follows:

00: (                                        lucy_switch_lights_on lucy room03 grip2) [1]
01: (                               lucy_changes_room lucy room03 room01 door03 grip2) [1]
02: (                               lucy_changes_room lucy room01 room02 door02 grip2) [1]
03: (                                        lucy_switch_lights_on lucy room02 grip1) [1]
04: (                  lucy_picks_glass_in_shelf_with_gripper lucy grip2 glass room02 shelf) [1]
05: (                lucy_picks_water_in_fridge_with_gripper lucy grip1 water room02 fridge) [1]
06: (                           lucy_pour_water lucy grip1 grip2 glass water room01 room02) [1]
07: (                               lucy_changes_room lucy room02 room01 door02 grip1) [1]
08: (              lucy_gives_water_with_glass_to_human lucy grip2 water human room01 glass) [1]
09: (                               lucy_changes_room lucy room01 room02 door02 grip1) [1]
10: (lucy_picks_medicine_in_fridge_with_gripper lucy grip1 medicine room02 fridge room01 room03) [1]
11: (                               lucy_changes_room lucy room02 room01 door02 grip2) [1]
12: (                            lucy_gives_medicine lucy grip1 medicine human room01) [1]
13: (                               lucy_changes_room lucy room01 room03 door03 grip1) [1]

plan total cost: 14.00


time spent:       0.11 seconds parsing
                  0.21 seconds encoding
                  5.13 seconds searching
                  5.45 seconds total time
```

Figure 5.6: Results of KSH planner

To summarize, the experiments show that KSH is competitive with the three main research planning platforms SAPA, Web Planner, and PDDL editor. However, KSH is designed for this domain but it is not domain dependent. This means that this planner will perform extremely well for the current domain be-cause of the tabular database provided. For other domains, the tabular database will not work, and this will slightly affect the speed of the algorithm. KSH is still able to generate a resulting plan without the need of the tabular database.

5.4 VALIDATION

As mentioned in chapter 2 2.3.2 , the final step is the application of VAL tool where it analyses the planner output. Since this tool is still used to validate planners for the Pddl Impoc competitions, I decided to validate the developed solution with the standards required by the competition. For the Validator to analyze the plan, it needs three inputs: domain & problem description with the generated plan from your planner. This important step confirms that the developed planner can handle the domain and & problem description properly. However, it confirms only for that domain developed and for other domains, it is not yet tested due to lack of time. The generated output was resulting in a valid plan 5.7.

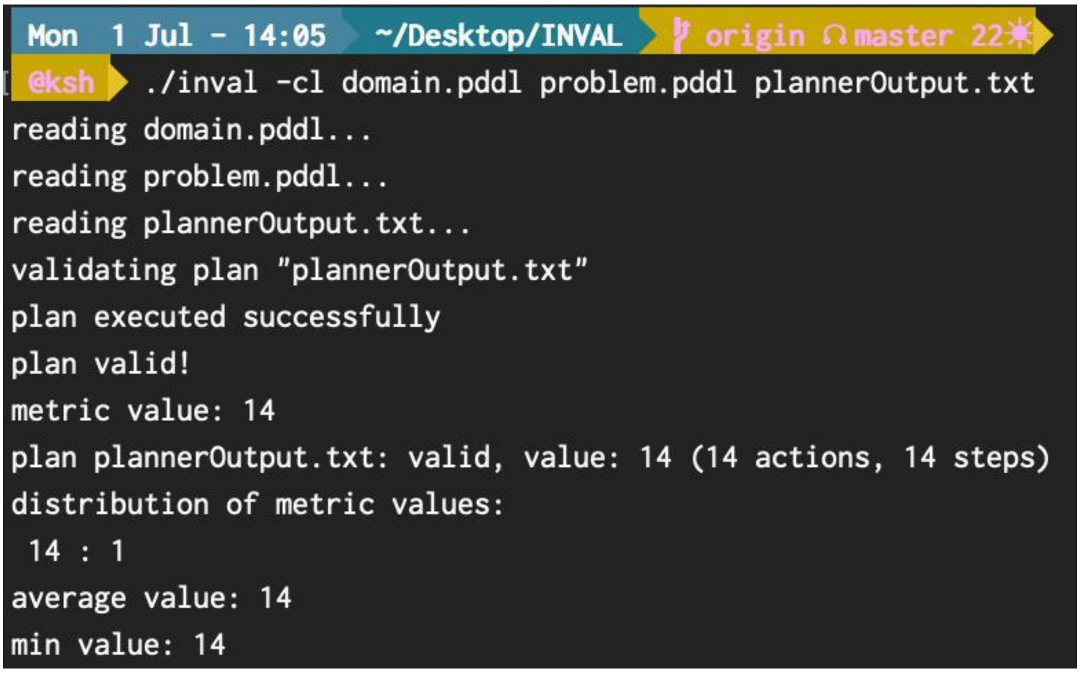


Figure 5.7: Validation generated output

CHAPTER 6

DISCUSSIONS & CONCLUSIONS

6.1 FUTURE ENHANCEMENTS

Many different integration, tests, and experiments have been left for the future due to lack of time (i.e. the experiments with real data are usually very time consuming). Future work concerns deeper analysis of mechanisms, new proposals to try different methods, or simply curiosity.

6.1.1 IMPLEMENTING MORE PROBLEMS FOR THE WORLD

The developed problem is derived from persona and use cases tailored for a simple living space for one human. an extension of it can be done for more humans or for a bigger living space. Constructing more personas for this use case is extremely tough since a real testing living environment is demanded to come up with more personas. However, the domain description provided is developed to become capable of handling many problems and an extension of the problem can impose new outcomes for the planner. More advanced problems can be constructed with the new use cases derived for the new personas. another challenging task would be to tailor up the domain description for care facilities. This will expand the range of the domain This can highly benefit on the comparison of the planner on higher execution times which may lead to different results.

6.2 AUTOMATION OF TABULAR DATABASE

As mention in chapter 4 no experiments were conducted on the action partitioning for other domains. another extension of the planner may become an automated way of generating a tabular database for at least a specific range of domains. The idea is to use the same approach of action partitioning as described earlier but with an improvement of it with a preliminary division of actions into sets of related actions and perform test on these sets. We simple add a feature of finding a maximal set of additive variables and let the actions that affect each of the variable form a set. By selecting additive variables, we ensure that the corresponding action sets are disjoint.

The heuristic itself could forms nodes in different partitions and the costs of the actions could form a better subset. This may also have a bad effect but for sure it will increase the range of experiments for the algorithm and can lead in a better debugging experience.

However, the partitioning would be a different topic to discuss because the approaches used may be bounded to the domain and no partition can be sufficient for a huge range of domains. To establish this approach the range of experiments must be expanded to conclude with a general case scenario.

6.3 DATA VISUALISATION

Visualization techniques aim to transmit information using graphical representation where it aids on a better understanding of the scenario.

6.4 GRAPH VISUALIZATION

In the problem described the feature of visualization can be displayed effectively using graphs. Relation information between parent and child can form edges wherein a hierarchical tree represents the relation between nodes. a hierarchical tree can be integrated with a heuristic visualization. Since nodes are represented as datasets, data visualization with notations can explain the state/s of the node/s. Nodes like initial state and end goal can be colored where the initial state is the root node.

6.5 PARTITIONING & HEURISTIC VISUALISATION

The nodes that are portioned in the subset would be distinguished with different colors and edges(action) that corresponds to each subset can have a higher contrast value for each critical path. When the partition function is finished a gradient color function can be used to represent the edges with the lower critical path value. This can denote the path that the algorithm chose in order to achieve the final goal.

In the end, a graph representation of the problem can be more self-explanatory for the uninformed end user.

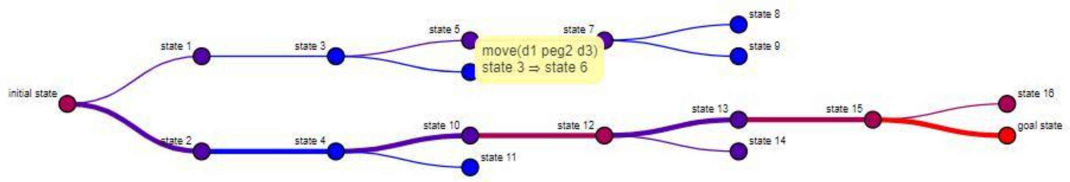


Figure 6.1: Sample Graph representation

BIBLIOGRAPHY

Aumayr, G. (2016). From Ambient Assisted Living to Active and Assisted Living: A Practical Perspective on Experiences and Approaches. In G. Aumayr, *Information Technologies in Medicine* (pp. 3–13). Springer Inter-national Publishing.

Bonet, Blai and H´ector Geffner. (2001). Planning as heuristic search. In B. a. Bonet, *Artificial Intelligence* (pp. 5–33). doi:10.1016/S0004-3702(01)00108-4

Bonet, Blai and Hector Geffner. (2001a). Heuristic Search Planner 2.0. In B. a. Bonet, *AI Magazine* (pp. 77–77). doi:10.1609/aimag.v22i3.1576

Bonnet, Blai and Hector Geffner. (1998). HSP: Heuristic Search Planner. In B. a. Bonnet, *HSP: Heuristic Search Planner*.

Brickman, A. M. (2018). An MRI measure of degenerative and cerebrovascular pathology in Alzheimer disease. In A. M. Brickman, *Neurology*. doi:10.1212/WNL.0000000000006310

Brooker, D. (2005). Dementia Care Mapping: A Review of the Re-search Literature. In D. Brooker, *Gerontologist 45 (suppl_1)* (pp. 11–18). doi:10.1093/geront/45.suppl_1.11

Cahill, S. (2007). Technology in dementia care. In S. Cahill, *Technology and Disability* (pp. 55–60). Retrieved from https://content.iospress.com/articles/technology-and-disability/tad00227

Cohen, J. E. (2000). World Population in 2050: Assessing the Projections. In J. E. Cohen.

Cohen-Mansfield, Jiska, Perla Werner, and Barry Reisberg. (1995). Temporal Order of Cognitive and Functional Loss in a Nursing Home Population. In J. P. Cohen-Mansfield, *Journal of the American Geriatrics Society* (pp. 974–978). doi:10.1111/j.1532-5415.1995.tb05560.x

Coles, A. (2012). A Survey of the Seventh International Planning Competition. In A. Coles, *A Survey of the Seventh International Planning Competition* (pp. 83–88). doi:10.1609/aimag.v33i1.2392

Culberson, Joseph C. and Jonathan Schaeffer. (1996). Searching with pattern databases. In J. C. Culberson, *Advances in Artifical Intelligence* (pp. 402–416).

Dedecker, J. (2008). Object Technology for Ambient Intelligence and Pervasive Computing. In J. Dedecker, *Object-Oriented Technology* (pp. 171–181).

Do, M. and S. Kambhampati. (2003). SAPA: A Multi-objective Metric Temporal Planner. In M. a. Do, *SAPA: A Multi-objective Metric Temporal Planner* (pp. 155–194). doi:10.1613/jair.1156

Domshlak, Carmel, J¨org Hoffmann, and Michael Katz. (2015). Red–black planning: A new systematic approach to partial delete relaxation. In C. J. Domshlak, *Artificial Intelligence* (pp. 73–114). doi:10.1016/j.artint.2014.12.008

Edelkamp, S. (2001). First Solutions to PDDL+ Planning Problems. In S. Edelkamp, *In PlanSIG Workshop* (pp. 75–88).

Fickert, Maximilian, Joerg Hoffmann, and Marcel Steinmetz. (2016). Combining the Delete Relaxation with Critical-Path Heuristics: A Direct Characterization. In M. J. Fickert, *Combining the Delete Relaxation with Critical-Path Heuristics: A Direct Characterization* (pp. 269–327). doi:10.1613/jair.5057

Fischer, G. (2001). User Modeling in Human–Computer Interaction. In G. Fischer, *User Modeling and User-Adapted Interaction* (pp. 65–86). doi:10.1023/A:1011145532042

Gilman, S. (2010). Oxford American Handbook of Neurology. In S. Gilman, *Oxford American Handbook of Neurology.* Oxford University Press.

Gregor, Peter, Alan F. Newell, and Mary Zajicek. (2002). Designing for Dy-namic Diversity: Interfaces for Older People. In P. A. Gregor, *Proceedings of the Fifth International ACM Conference on Assistive Technologies. Assets ’02* (pp. 151–156). doi:10.1145/638249.638277

Groot, C. (2018). Clinical phenotype, atrophy, and small ves-sel disease in APOEe2 carriers with Alzheimer disease. In C. Groot, *Neurology.* doi:10.1212/WNL.0000000000006503

Hagen, I. (2004). Development of a Protocol for the Assessment of Assistive Aids for People with Dementia. In I. Hagen, *Dementia* (pp. 281–296). doi:10.1177/1471301204045161

Haslum, P. (2016). New Admissible Heuristics for Domain-Independent Planning. In P. Haslum, *New Admissible Heuristics for Domain-Independent Planning* (p. 6).

Haslum, Patrik and Hector Geffner. (2000). Admissible Heuristics for Optimal Planning. In P. a. Haslum, *Admissible Heuristics for Optimal Planning* (p. 10).

Helmert, M. (2006). The Fast Downward Planning System. In M. Helmert, *The Fast Downward Planning System* (pp. 191–246). doi:10.1613/jair.1705

Helmert, M. (2009). Concise finite-domain representations for PDDL planning tasks. In M. Helmert, *Artificial Intelligence. Advances in Automated Plan Generation* (pp. 503–535). doi:10.1016/j.artint.2008.10.013

Helmert, M. (2014). Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces. In M. Helmert, *Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces.* doi:10.1145/2559951

Hoffmann, J. (2003). Utilizing Problem Structure in Planning: A Local Search Approach. In J. Hoffmann, *Utilizing Problem Structure in Planning: A Local Search Approach.*

Hoffmann, J. and B. Nebel. (2001). The FF Planning System: Fast Plan Generation Through Heuristic Search. In J. a. Hoffmann, *The FF Planning System: Fast Plan Generation Through Heuristic Search.* doi:10.1613/jair.855

Howe, A. E. (2000). Exploiting Competitive Planner Performance. In A. E. Howe, *Recent Advances in AI Planning* (pp. 62– 72).

Howey, Richard, Derek Long, and Maria Fox. (2008). VAL: Automatic Plan Validation, Continuous Effects and Mixed Initiative Planning using PDDL. In R. D. Howey, *VAL: Automatic Plan Validation, Continuous Effects and Mixed Initiative Planning using PDDL.*

Hughes, Julian and Gill Campbell. (2003). The electronic tagging and tracking debate. In J. a. Hughes, *Nursing and Residential Care* (pp. 174–177). doi:10.12968/nrec.2003.5.4.11592

Joy For All Companion. (n.d.). Retrieved from Ageless Innovation LLC: https://joyforall.com/pages/businesssolutions

Katz, Michael and Carmel Domshlak . (2008). Optimal Additive Composition of Abstraction-based Admissible Heuristics. In M. a. Katz, *Optimal Additive Composition of Abstraction-based Admissible Heuristics* (p. 8).

Keates, S. (2002). User Models and User Physical Capability. In S. Keates, *User Modeling and User-Adapted Interaction* (pp. 139–169). doi:10.1023/A:1015047002796

Korf, Richard E., Michael Reid, and Stefan Edelkamp. (2001). Time complexity of iterative-deepening-A*. In R. E. Korf, *Artificial Intelligence* (p. 199). doi:10.1016/S0004-3702(01)00094-7

Kuchinomachi, Y. a. (1999). The relationship between the cognitive function decrease of elderly people and the usability of domestic appliances and participation in outside activities. In Y. a. Kuchinomachi, *Technology and Disability* (pp. 169–175). Retrieved from https://content.iospress.com/articles/technology-and-disability/tad00044

Kurz, A. (Dec. 2016). “RHAPSODY – Internet-based support for caregivers of people with young onset dementia: program design and methods of a pilot study”. In A. Kurz, *International Psychogeriatrics* (pp. 2091–2099). Retrieved from https://www.cambridge.org/core/journals/international-psychogeriatrics/article/rhapsody-internetbased-support-for-caregivers-of-people-with-young-onset-dementia-program-design-and-methods-of-a-pilot-study/16260AF4141569F609579E95D3273423

Lian, K. (2012). Optimization of process planning with var-ious flexibilities using an imperialist competitive algorithm. In K. Lian, *Int J Adv Manuf Technol* (pp. 815–828). doi:10.1007/s00170-011-3527-8

Mann, D. M. (1996). Pyramidal Nerve Cell Loss in Alzheimer’s Disease. In D. M. Mann, *Neurodegeneration* (pp. 423–427). doi:10.1006/neur.1996.0057

Marshall, M. (2004). Perspectives on Rehabilitation and Dementia. In M. Marshall. Jessica Kingsley Publishers.

Marshall, M. N. (1996). Sampling for qualitative research. In M. N. Marshall, *Fam Pract* (pp. 522–526). doi:10.1093/fampra/13.6.522

Mart´ın, Estefan´ıa, Pablo A. Haya, and Rosa M. Carro. (2013). User Modeling and Adaptation for Daily Routines. In E. P. Mart´ın, *Providing Assistance to People with Special Needs.* Retrieved from https://www.springer.com/de/book/9781447147770

Mauricio C. Magnaguagno, Ramon Fraga Pereira and Felipe Meneguzzi Martin D. More. (2017). WEB PLANNER: A Tool to Develop Classical Planning

Domains and Visualize Heuristic State-Space Search. In R. F. Mauricio C. Magnaguagno. Retrieved from https://web-planner.herokuapp.com/

McCluskey, T. L. (2003). PDDL: A Language with a Purpose? In T. L. McCluskey, *PDDL: A Language with a Purpose?* (p. 5).

Mesulam, M.-M. (2019). Cortical cholinergic denervation in primary progressive aphasia with Alzheimer pathology. In M.-M. Mesulam, *Neurology*. doi:10.1212/WNL.0000000000007247

Moore, G. (1998). Cramming More Components Onto Integrated Cir-cuits. In G. Moore, *Proceedings of the IEEE* (pp. 82–85). doi:10.1109/JPROC.1998.658762

Nielsen, Lene and Kira Storgaard Hansen. (2014). Personas is Applicable: A Study on the Use of Personas in Denmark. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, (pp. 1665–1674). Toronto, Ontario, Canada. doi:10.1145/2556288.2557080

Norfray, Joseph F. and James M. Provenzale. (2004). Alzheimer's Disease: Neuropathologic Findings and Recent Advances in Imaging. In J. F. Norfray, *American Journal of Roentgenology* (pp. 3–13). doi:10.2214/ajr.182.1.1820003

Orpwood, R. (2004). User Involvement in Dementia Product Development. In R. Orpwood, *Dementia* (pp. 263–279). doi:10.1177/1471301204045160

Orpwood, R. (2007). Designing technology to support quality of life of people with dementia. In R. Orpwood, *Technology and Disability* (p. 103). Retrieved from https://content.iospress.com/articles/technology-and-disability/tad00232

Ossenkoppele, R. (2019). Associations between tau, Ab, and cortical thickness with cognition in Alzheimer disease. In R. Ossenkoppele, *Neurology*. doi:10.1212/WNL.0000000000006875

PARO Therapeutic Robot. (n.d.). Retrieved from http://www.parorobots.com/

Pekkarinen, L. (2004). Work Stressors and the Quality of Life in Long-Term Care Units. In L. Pekkarinen, *Gerontologist* (pp. 633–643). doi:10.1093/geront/44.5.633

Pekkarinen, L. (2006). Resident care needs and work stressors in special care units versus non-specialized long-term care units. In L. Pekkarinen, *Research in Nursing & Health* (pp. 465–476). doi:10.1002/nur.20157

Prince, M. (Sept. 2016). World Alzheimer report 2016: improving healthcare for people living with dementia: coverage, quality and costs now and in the future. In M. Prince. Retrieved from http://www.alz.co.uk/

Ramos, Carlos, Juan Carlos Augusto, and Daniel Shapiro. (2008). Ambient Intelligence the Next Step for Artificial Intelligence. In C. J. Ramos, *IEEE Intelligent Systems* (pp. 15–18). doi:10.1109/MIS.2008.19

Richter, S. and M. Westphal. (2010). The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks. In S. a. Richter, *The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks* (pp. 127–177). doi:10.1613/jair.2972

Risacher, S. L. (2017). Alzheimer disease brain atrophy subtypes are associated with cognition and rate of decline. In S. L. Risacher, *Neurology* (pp. 2176–2186). doi:10.1212/WNL.0000000000004670

Sanchez Nigenda, R. (2018). Design and evaluation of planning and mathematical models for generating learning paths. In R. Sanchez Nigenda, *Computational Intelligence* (pp. 821–838). doi:10.1111/coin.12134

Selkoe, D. J. (2002). Alzheimer's Disease Is a Synaptic Failure. In D. J. Selkoe, *Science* (pp. 789–791). doi:10.1126/science.1074069

Shneiderman, B. (2003). Promoting Universal Usability with Multi-layer In-terface Design. In B. Shneiderman, *Proceedings of the 2003 Conference on Universal Usability* (pp. 1–8). doi:10.1145/957205. 957206

Silver Mother. (n.d.). Retrieved from https://www.technologyintegrator.net/article/best-of-ces-mother-the-creepy-iot-device/

SoftBank Robotics. (n.d.). Retrieved from https://www.softbankrobotics.com/emea/en/index

Starner, T., J. Weaver, and A. Pentland. (1998). Real-time American sign language recognition using desk and wearable computer based video. In T. J. Starner, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (pp. 1371–1375). doi:10.1109/34.735811

Stavropoulos, T. G. (2016). Dem@Home: Ambient Monitoring and Clinical Support for People Living with Dementia. In T. G. Stavropoulos, *The Semantic Web* (pp. 26–29). Springer International Publishing.

Stephanidis, C. (2001). Adaptive Techniques for Universal Access. In C. Stephanidis, *User Modeling and User-Adapted Interaction* (pp. 159–179). doi:10.1023/A:1011144232235

Stephanidis, Constantine and Anthony Savidis. (2001). Universal Access in the Information Society: Methods, Tools, and Interaction Technologies. In C. a. Stephanidis, *Universal Access in the Information Society* (pp. 40–55). doi:10.1007/s102090100008

Strassnig, Martin and Mary Ganguli. (2005). About a Peculiar Disease of the Cerebral Cortex. In M. a. Strassnig, *Psychiatry (Edgmont)* (pp. 30–33). Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2993534/

Strobel, Volker and Alexandra Kirsch. (2014). Planning in the Wild: Modeling Tools for PDDL. In V. a. Strobel, *KI 2014: Advances in Artificial Intelligence* (pp. 273–284). Springer International Publishing.

Tim, BRengle and Cunniff Ross. (2016). The ADL Programmer's Reference Manual. In B. a. Tim, *The ADL Programmer's Reference Manual.* Retrieved from http://adl.sourceforge.net/

Topo, P. (2009a). Technology Studies to Meet the Needs of Peo-ple With Dementia and Their Caregivers: A Literature Review. In P. Topo, *J Appl Gerontol* (pp. 5–37). doi:10.1177/0733464808324019

Topo, P. (2009b). Technology Studies to Meet the Needs of People With Dementia and Their Caregivers: A Literature Review. In P. Topo, *J Appl Gerontol* (pp. 5–37). doi:10.1177/0733464808324019

Younes, H. L. (2005). The First Probabilistic Track of the International Planning Competition. In H. L. Younes, *The First Probabilistic Track of the International Planning Competition* (pp. 851–887). doi:10.1613/jair.1880