Institute of Architecture of Application Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

Risk-Aware Hierarchical Planning for Smart Non-Residential Buildings

Christof Hohmann

Course of Study:	Autonomous Systems
Examiner:	Dr. Ilche Georgievski
Supervisor:	Ebaa Alnazer, M.Sc.
Commenced:	March 6, 2023
Completed:	September 6, 2023

Abstract

Smart buildings are a multidisciplinary topic that has attracted interest in recent years. Research often focuses on smart homes, with little attention paid to other building categories. However, these contribute almost as much to the building sector's emissions as residential buildings. An essential step towards more efficient electricity supply and better grid stability is demand-responsive electricity generation. In the smart grid, however, this requires a forecast of electricity demand and thus a plan of future electricity consumption.

Automated planning is a field of computer science that deals with the generation of plans in modelled environments. Uncertainty about the future demands for the consideration of associated risk and the quality of generated plans. In this context, the risk attitude of decision-makers is another decisive factor defining the steps chosen to operate a building. The solution of a planning problem by means of search algorithms as well as the approaches of optimisation and Satisficing are discussed.

We present a hierarchical planning domain for the purpose of estimating the electricity demand of a smart non-residential building. Considering all parties involved makes it a multi-objective planning problem. This is why there are no unique optimal but multiple non-dominated solutions. A suitable trade-off between comfort and energy consumption can be found by choosing a suggested plan. The perceived satisfaction that is expected from a plan also depends on the willingness to take risk. Considering all these factors offers the possibility to quantify the expected impact on occupant satisfaction which is associated with a certain energy demand. An implementation of this domain and its application in illustrative scenarios is presented in the following. Proving a good abstraction level of the domain, as well as an effective generation of non-dominated plans. For application in the real world, however, the implementation faces limits of space complexity. The search for non-dominant plans could be made more efficient through analysis of the domain and the search algorithm. Different search algorithms and heuristics are discussed, as well as the dominance of plans in the case of multiple objectives. A comparison with sequential decision processes further reveals a gap in the seamless integration of automated planning and acting. The need for hand-crafting domains yields another topic for future research, namely automatic domain generation.

Kurzfassung

In den letzten Jahren haben Smart Buildings als multidisziplinäres Thema an Bedeutung gewonnen. Die Forschung konzentriert sich dabei häufig auf Smart Homes und schenkt anderen Gebäudetypen wenig Beachtung. Diese tragen jedoch fast genauso viel zu den Emissionen des Gebäudesektors bei wie Wohngebäude. Ein entscheidender Schritt zu einer effizienteren Stromversorgung und einer höheren Netzstabilität ist die bedarfsgerechte Stromerzeugung. Eingebunden in das Smart Grid erfordert dies eine Prognose des Strombedarfs und damit eine Planung des zukünftigen Stromverbrauchs.

Als Teilgebiet der Informatik, beschäftigt sich die automatisierte Planung, mit der Generierung von Plänen in modellierten Systemen. Dabei erfordert die Ungewissheit bezüglich der Zukunft die Berücksichtigung des damit einhergehenden Risikos, ebenso wie der Qualität der erzeugten Pläne. In diesem Zusammenhang ist die Risikobereitschaft des Entscheidungsträgers ein entscheidender Faktor, welcher die gewählten Aktionen für den Gebäudebetrieb bestimmt. Es werden die Lösung von Planungsproblemen mittels Suchalgorithmen sowie die Ansätze der Optimierung und des Satisficing diskutiert.

Im Rahmen dieser Arbeit wird eine hierarchische Planungs-Domain zur Abschätzung des Strombedarfs von smarten Nicht-Wohngebäuden vorgestellt. Die Berücksichtigung aller beteiligten Akteure macht das Ganze zu einem mehr-zieligen Planungsproblem. Aus diesem Grund gibt es keine eindeutige optimale Lösung mehr, sondern zahlreiche nicht-dominierte. Durch die Wahl eines vorgeschlagenen Plans kann ein geeigneter Kompromiss zwischen Komfort und Energieverbrauch gefunden werden. Die empfundene Zufriedenheit, die von einem Plan ausgeht, hängt auch von der Bereitschaft ab, Risiken einzugehen. Die Berücksichtigung all dieser Aspekte bietet die Möglichkeit, die erwarteten Auswirkungen auf die Zufriedenheit der Bewohner zu quantifizieren, die mit einem bestimmten Energieverbrauch einhergehen. Eine Implementierung dieser Domäne und ihre Anwendung in Beispielsituationen werden anschließend vorgestellt. Hierbei wird ein angemessener Abstraktionsgrad der Domain sowie eine effektive Generierung von nicht-dominierten Plänen erreicht. Für die Anwendung in der realen Welt stößt die Implementierung jedoch an Grenzen des Rechenspeichers. Durch Analyse der Domain und des verwendeten Suchalgorithmus konnte die Suche effizienter gestaltet werden. Es werden unterschiedliche Suchalgorithmen und Heuristiken diskutiert, sowie die Dominanz von Plänen im Falle mehrerer Zielgrößen. Der Vergleich mit Markov'schen Entscheidungsproblemen zeigt zudem eine Lücke in der nahtlosen Integration von automatisiertem Planen und Handeln auf. Die Notwendigkeit, Planungs-Domains von Hand erstellen zu müssen, deutet auf ein weiteres, zukünftiges Forschungsthema, nämlich die automatische Generierung von Planungs-Domains.

Acknowledgments

I would like to thank my supervisor Ebaa Alnazer for her constructive feedback and uncomplicated communication during mentoring.

My thanks also go to my colleagues at ENDEKO for the exciting distractions from this work and granted space to pursue it.

Finally, I would like to express my gratitude to my family and friends. To my dear Lena for her patience. To Birgit, Michael, and Sebastian, who accompanied me on my way, always stood by my side and made me who I am today.

Beharrlichkeit führet zum Ziel

The idea that the future is unpredictable is undermined every day by the ease with which the past is explained.

Daniel Kahneman – Thinking, Fast and Slow

Contents

List of Figures

List of Tables

List of Listings

List of Algorithms

Acronyms

AC air conditioning. 25

AI Artificial Intelligence. 21

AQI air quality index. 44

BESS Battery Energy Storage System. 43

BFS Breadth-First Search. 36

CSP Constraint Satisfaction Problem. 48

DAP day-ahead hourly pricing. 27

DFS Depth-First Search. 36

ECS Environmental Control System. 25

EMS Energy Management System. 25

GTN Goal-Task Network. 35

HDDL Hierarchical Domain Definition Language. 45

HGN Hierarchical Goal Network. 35

HTN Hierarchical Task Network. 21

HVAC heating, ventilation, air conditioning. 43

iff if and only if. 40

IPC International Planning Competition. 45

JSHOP2 Java Simple Hierarchical Ordered Planner 2. 49

LCS Lighting Control System. 25

LP Linear Programming. 39

MDP Markov Decision Process. 38

MEU maximum expected utility. 30

MO multi-objective. 39

PDDL Planning Domain Definition Language. 45

PV Photovoltaic. 59

SWDR shortwave downward radiation. 110

1 Introduction

Global warming is one, if not the major challenge of this century. Despite efforts to reduce global energy consumption and the associated greenhouse gas emissions, they continue to rise. The inability to achieve such a reduction appears to have two reasons. On the one hand, richer countries are unwilling to reduce their living standards, while rising countries strive to achieve such standards. Therefore, an acceptable solution seems to arise only from achieving these standards in a more efficient way or by generating awareness in quantifying involved resources. The construction and maintenance of buildings is one of the key elements for this high standard of living.

According to the International Energy Agency: "the operation of buildings accounted for 30% of global final energy consumption and 27% of total energy sector emissions in 2021 (8% being direct emissions in buildings and 19% indirect emissions from the production of electricity and heat used in buildings). [...] The building sector is therefore directly and indirectly responsible for around one-third of global energy- and process-related CO_2 emissions."[II22] Modern buildings offer their occupants many amenities. Yet, these amenities come at a high energy cost. To reduce their energy demand, it is necessary to use them as intelligently and resource-efficiently as possible. Smart buildings can detect their external and internal state through sensors and partially manipulate it using actuators. There usually is a central logic unit responsible for action management. One approach to making buildings more intelligent and resource-efficient is thus to improve this control system.

Artificial Intelligence (AI) planning is a field of computer science that deals with the generation of strategies and action sequences. Hierarchical Task Network (HTN) planning is one concept of this field. It essentially solves problems by decomposing tasks until their solution is trivially feasible. It is therefore highly suitable for developing plans for complex real-world problems. The quality of solutions is supported by decision theory, which provides concepts for rational decision making. Plans for the future, in particular, must consider the uncertainty of their occurrence. The most common model in this context is the maximisation of expected utility. However, different entities may have different perceptions or subjective utilities. This fact is better known as attitude towards risk. It is particularly important to individual decisions or when unable to assess the likelihood of events occurring.

Based on above considerations, we believe that it would be useful to apply AI planning to the task of efficient energy use in buildings. We intend to develop a concept for doing so within the framework of this thesis. The key research question is to what extent AI and especially risk-aware HTN planning can reduce the energy consumption of buildings while keeping the noticeable impact on inhabitants' comfort as low as possible.

To keep the developed solution comparable with others and to get an overview of different approaches, we will first review and categorise them. Starting from this theoretical foundation, we will then elaborate a new solution to the problem at hand, by tackling shortcomings and combining ideas

from the fields of decision theory, AI planning and building automation. We will then transfer the solution into an HTN implementation and evaluate the benefits of its application in constructed scenarios.

The focus will especially be on non-residential buildings, as the possible leverage is large, and this application is currently considered less compared to residential buildings. In addition, the incentives of building operators and occupants pose a multimodal goal. To fulfil this goal in an environment with an uncertain future, their incentives need to be traded off against each other. This issue becomes more complex as we examine uncertainty, preferences and making automated decisions under the resulting risk, in more detail. Reviewing economic strategies, studies of human behaviour, and the consideration of multiple objectives, make this thesis meaningful for a wide audience.

The remainder of this work is organised as follows: Chapter 2 summarises different types of buildings and what we expect from them. It then explains how to represent human needs and how to quantify their fulfilment. The basics of automated planning and some more advanced concepts are presented. Chapter 3 gives an overview of earlier work on the topic of building automation as well as evolved concepts when planning under uncertainty. A new approach to the problem of smart building management is then elaborated in Chapter 4 and transferred into an executable implementation in Chapter 5. Chapter 6 evaluates the performance of the implementation on some constructed scenarios and pinpoint possible improvements. Major findings are then summarised and concluded in Chapter 7.

2 Preliminaries

To investigate how the energy consumption of buildings can be influenced by using a planning tool, we need to review the following fields: What types of buildings are there, what purpose do these buildings serve, i.e., what do humans expect from buildings, and how is energy used to fulfil these expectations. To automate the fulfilment of expectations, we also must deal with the representation of that expectations and their fulfilment in an uncertain future. Then we arrive at the point where the energy demand to fulfil these expectations can be planned. In this chapter, we therefore review the theoretical basis for describing buildings, preferences, and planning problems. We will also recap the risk induced by planning into an uncertain future, how to make best possible decisions in planning and how planning is automated.

2.1 Building Automation

Ever since man has settled and lived in dwellings, he has tried to improve them. In recent decades, this improvement has been expressed primarily in new concepts for the automated management of buildings. We want to elaborate on the term *residential*, as this work is specifically concerned with the automation of non-residential buildings. Studies concerned with greenhouse gas emissions and the environmental impact of buildings usually distinguish buildings by their function of inhabiting people as residential and non-residential buildings [II14; II22]. Therefore, this work will concern itself with the buildings, defined as the opposite of residential buildings, which are in turn defined by their purpose of residential occupancy. Non-residential buildings can serve various commercial, industrial, institutional, and public functions. Seidl [Sei06] gives a list of building types and their functions, from which the following excerpt is of particular importance.

- Offices: These buildings provide spaces for business operations, professional services, administration, and other office-related activities.
- Retail Buildings: These structures are designed for commercial activities such as stores, shops, malls, or shopping centres where goods or services are offered to customers.
- Industrial Buildings: These buildings house manufacturing, processing, or production activities, including factories, warehouses, plants, or distribution centres.
- Institutional Buildings: These structures serve public or private institutions, such as schools, colleges, universities, hospitals, government offices, libraries, museums, or religious establishments.
- Hospitality Buildings: Such as hotels, resorts, motels, lodges, or other accommodation facilities for travellers and visitors.

- Entertainment Buildings: Such as theatres, cinemas, concert halls, stadiums, arenas, or convention centres, where various forms of entertainment, performances, or events take place.
- Healthcare Buildings: These structures are specifically designed to provide medical services and may include hospitals, clinics, medical offices, or research facilities.
- Recreational Buildings: These buildings are dedicated to recreational activities and can include gyms, sports facilities, fitness centres, or leisure centres.
- Transportation Buildings: Such as airports, train stations, bus terminals, or parking structures designed to facilitate transportation services.

All these buildings have in common, that they should provide protection against external influences such as temperature, wind, and solar radiation. The protection from solar radiation can usually be regulated using blinds. However, it is never possible to completely seal a building off from external heat flow. Without active measures, the temperature inside a building would always follow the outside temperature. The speed of this adaptation depends on the so-called U-value, which is a measure for the thermal transmittance of walls. The change in indoor temperature from one point in time to another can be described as a function of the previous temperature difference as given by

$$T_{t+1} = T_t + \Delta\theta_t \cdot \frac{U \cdot A}{c \cdot m}. \tag{2.1}$$

Where T_t is the indoor temperature at time t in $°C$ and $\Delta\theta_t$ is the current temperature difference between outdoor and indoor in K. U is the thermal transmittance in $\frac{W}{m^2 \cdot K}$ and A is the cross section of transmittance in m^2. c is the specific heat capacity in $\frac{J}{kg \cdot K}$ and m is the mass in kg [MN07].

The above-mentioned process specifically describes heat transfer by *conduction*, i.e., the heat flow that penetrates directly through the external surfaces of the building to the inside or outside. Furthermore, there are heat flows through *convection*, in the case of buildings specifically the transport of heat through air exchange, as well as *radiation*, i.e., a heat flow by electromagnetic radiation. In a closed building, the first of the three effects mentioned is dominant, while the second becomes more important when the building is ventilated. The third is only of greater relevance in the case of direct solar radiation [MN07].

By reducing the step size of t, the curve of Equation (2.1) converges to an exponential-function. Both functions are plotted in Figure 2.1. There are many processes that can be represented as an asymptotic approximation of final values by an e-function. This comparison has the advantage that one can abstract from the actual effects and summarise all variables in a time constant τ. In Equation (2.1), the values $\frac{U \cdot A}{c \cdot m}$ could also be interpreted as $\frac{1}{\tau}$ and thus $\tau = \frac{c \cdot m}{U \cdot A}$. The time constant τ gives an idea of how fast the process is progressing. After that time ($t = \tau$), the internal temperature has changed by 63.2% of the initial temperature difference θ_0, i.e. $T_{t=\tau} = T_0 + \theta_0 \cdot (1 - e^{\frac{\tau}{\tau}}) = T_0 + \theta_0 \cdot 0.632$.

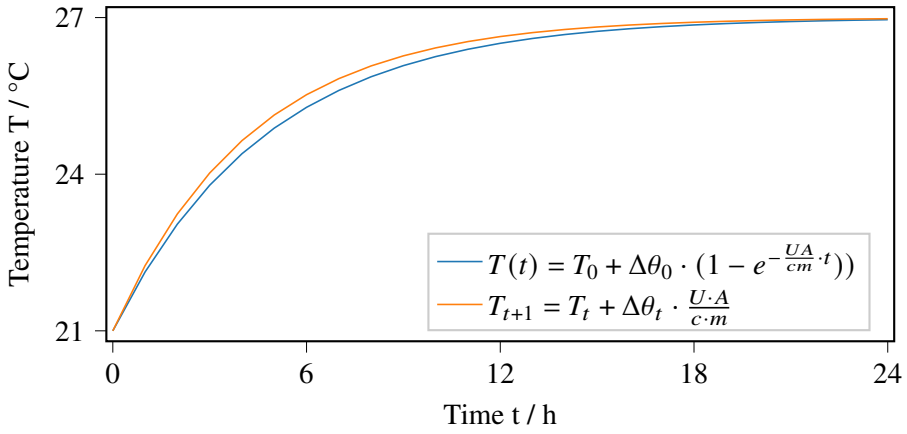


Figure 2.1: Temperature evolution due to heat conduction

Where the considered room has an initial temperature of 21 $°C$ and the outside temperature is a constant 27 $°C$. The step size t of Equation (2.1) is 1 h.

The relative humidity RH of air is defined as the ratio of the partial pressure of water vapour p to the saturation vapour pressure $p_s(T)$ of water at temperature T, expressed as

$$RH(p) = \frac{p}{p_s(T)}. \tag{2.2}$$

The saturation vapour pressure can be approximated by the Antoine equation

$$log_{10} p_s(T) = A - \frac{B}{C - T}. \tag{2.3}$$

Where A, B and C are coefficients for a certain substance. For water between 1 and 100 $°C$ these are given by the values 8.07131, 1730.63 and 233.426, respectively [CW13]. Accordingly, the amount of water vapour which is dissolved in the air increases exponentially with temperature. Thus, relative humidity decreases as the temperature rises if the amount of dissolved water vapour does not change.

In a normal building, countermeasures against these effects would of course be taken, such as the use of air conditionings (ACs), which are usually controlled automatically. Many of the buildings mentioned above also support other automatic functions and are supplemented with intelligent control. There have been a lot of inventions to automate buildings, starting from steam automated doors in ancient Roman Egypt [AW15] to modern systems that manage whole groups of buildings [Cle20b]. Among others, common intelligent buildings feature the following systems:

- Energy Management System (EMS): Optimises energy consumption, demand response management and integration of renewable energy sources.
- Environmental Control System (ECS): Is concerned with occupant comfort, productivity, and well-being, therefore controls various environmental factors within the building, such as temperature, humidity, air quality, and ventilation.
- Lighting Control System (LCS): Control of natural and artificial lighting within buildings.

Building Category	Non Automated	Automated	Intelligent	Smart	Thinking
Control Flow	Manual or no control	Automated	Reactive	Adaptive	Predictive
Comfort and Efficiency, Information Input	Primitive / Simple	Uniform interface	Systems and Data Integration	Enterprise integration and building as a system approach	Undefined / Ambiguous Data
Occupant Interaction and Efficiency	Some control, but low efficiency	Centralized Control, medium efficiency	More control, higher efficiency	Inherent control, higher efficiency	Predictive control, higher efficiency
Interaction of Operation with Occupants	None to manual	Pre-programmed	Ability to react to occupancy data in real time	Building operation defined by and adapted to building occupants	Effective operation based upon predicted use by occupants for a specified function

Table 2.1: Building autonomy levels adopted from [BMB14]

Despite having common subsystems, there is some discussion what exactly an *intelligent* and what a *smart* building is. Buckman et al. [BMB14] separate different building classes according to their level of self-sufficiency. The authors list categories, some of which are shown in Table 2.1.

Buckman et al. conclude that smart buildings take a holistic view of intelligence, control and construction. Their design is about adaptability, not reactivity. They are designed with energy efficiency, comfort and satisfaction in mind. On the other hand, intelligent buildings focus on individual intelligent systems that use information reactively. In other words, the development of the control system, building materials and construction are separate processes [BMB14]. In contrast Clements-Croome argues, that intelligent buildings include a broader view. Automation aspects and high technology, especially in information and communications technology, is what makes buildings smart. On the other hand, buildings need to respond to social and environmental factors as well, which is articulated by the language of low-tech passive environmental design. An intelligent building thus increases the environmental socio-economic value [Cle20b].

Following that line of argumentation, we only want to deal with the control of buildings operation here and use the term *smart*. The controller of a smart building should operate not only reactively to current state changes but anticipate actions using predictions of the future. Weather forecasts or fluctuating electricity prices, for example, can provide such insights. AI planning is known as a technique for deriving courses of action, even when faced with an uncertain future. However, as predictions always pose the threat of being flawed, the willingness to accept errors has to be determined.

In the face of new electricity consumer and provider arrangements, there is often talk of the *smart grid*. This concept enables participants to communicate with each other and thus yield more efficient power management. Such information distribution enables techniques like dynamic day-ahead hourly pricing (DAP) and demand response or load scheduling [Mom12]. In the context of this work, we want to limit ourselves to the concepts mentioned and to electricity as the primary source of energy. Even though there are differences in energy supply, depending on the location, this saves further conversions and is geared towards future infrastructures. Note that concepts such as DAP require planning a day in advance. This means, that a plan into the uncertain future must be generated, whereby the execution of actions, and thus the confirmation about their effect, is delayed.

2.2 Acting in Uncertain Environments

In modelling the domain of smart buildings using an AI planning tool, we have to account for uncertainty and other eventualities. These might have different characters and origins. In addition, the distinction between risk and uncertainty is sometimes not as clear as it should be. Before diving into details, however, we want to give a short notion of terms used in AI planning, as they are needed to understand following remarks. A detailed explanation is given in Section 2.3.

Let us first distinguish *planning* as the operation of finding a course of action, whereas *acting* is its application. A decision-making entity in the context of planning is generally referred to as an *agent*, regardless of whether it is a human being or a program. The environment in which the agent operates is then called the planning *domain*. It provides the set of variables and scope of functionality. As a model, it limits the possibilities and makes various abstractions through the description. Therefore, planning domains are always a simplification of reality and can never represent all details. A common simplification is to consider the domain as a *discrete* sequence of states. Where a *state* is a snapshot of all the variables in the system, which can be represented by statements or *predicates*. The system can only transit from one defined state to another. The fact that earlier decisions may influence later ones is also called *sequential*. The *goal* for a planning *problem* can be either to reach a desired state, or to perform a given set of *tasks*. Furthermore, the agent's objective is to derive a *plan* - an ordered list from the set of possible *actions* - that, when applied to the *initial* state, yields the *goal* state.

2.2.1 Uncertainty and Risk

Uncertainty and risk are still often confused in everyday speech, even though Knight highlighted this more than a hundred years ago [Kni21]. According to Knight's work, *risk* refers to situations where the probabilities of various outcomes are known or can be estimated objectively. *Uncertainty*, on the other hand, pertains to situations where the probabilities of outcomes are unknown or cannot be calculated. Alnazer et al. [AGA22] further elaborate on this topic and categorise the five stages of uncertainty reported in Table 2.2.

Level	Description
1	Complete certainty about the outcome.
2	Risk without uncertainty, where outcomes and their probability distribution are completely known.
3	Fully reducible uncertainty, where the outcomes are fully known, but their probability distribution is unknown. The uncertainty in this level is fully reducible to risk by statistical inference of the probability distribution of outcomes.
4	Partially reducible uncertainty, where there is a limit to what can be deduced about the outcomes and their probability even by significant statistical inference. The probabilities in this level reflect beliefs rather than frequencies of repeated trials as defined in Levels 2 and 3.
5	Irreducible uncertainty, which is the state of total ignorance that cannot be solved by collecting more data nor using sophisticated methods of statistical inference.

Table 2.2: Five stages of uncertainty based on [AGA22]

Following Knight's argumentation there would be no profit without uncertainty, as competitive forces tend to eliminate potential profit opportunities - or, as we may see, maximise expected utility. True profit can therefore only arise from successful navigation through not fully derivable uncertainties [Kni21].

2.2.2 Expectations, Preferences and Utility

The term of utility in the context of economy was first elaborated by von Neumann and Morgenstern [NM44]. They propose that individuals assign subjective values - or *utilities* - to different outcomes based on their preferences. These utilities guide people's decision-making process and determine their choices. The authors assume that individuals are rational decision makers and analyse the effect of this assumption in various game scenarios. They equate the notion of *rational* behaviour with the aim of utility maximisation. Their line of reasoning leads to the field of *game theory*, which is the study of strategic decision-making in situations where the outcome of one person's choice depends on the choices made by others. However, thereby always assuming rational behaviour and complete information, including preferences.

In the context of *utility theory*, the purpose of a utility value is to represent a preference - sometimes not precisely expressed - as a mathematical value that can be compared. A utility function U maps from lotteries X to real numbers [RN21]

$$U : X \to \mathbb{R}. \tag{2.4}$$

Consider an agent, who *prefers* the outcome A of a lottery over B (written as $A \succ B$), has a utility function that yields $U(A) > U(B)$. Or, if he is *indifferent* between two outcomes B and C (written $B \sim C$), his utility function yields $U(B) = U(C)$. This fact is described by Russell and Norvig as the *existence of a utility function*.

The characteristics of utility functions are defined in [NM44] and are commonly referred to as *six axioms of utility*. However, Russell and Norvig point out that the established axioms are rather axioms about preferences. Therefore, they add the *existence of a utility function* and the *expected utility of a lottery* to the axioms. All eight axioms are listed in Table 2.3 as they are presented in [RN21].

Name	Meaning	Equation
Orderability	Given any two lotteries, a rational agent must either prefer one or else rate them as equally preferable.	Exactly one of $(A \succ B)$, $(B \succ A)$, or $(A \sim B)$ holds
Transitivity	Given any three lotteries, if an agent prefers A to B and prefers B to C, then the agent must prefer A to C.	$(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$
Continuity	If some lottery B is between A and C in preference, then there is some probability for which the rational agent will be indifferent between getting B for sure and the lottery that yields A with probability p and with probability $1 - p$.	$A \succ B \succ C \Rightarrow \exists p [p, A; 1 - p, C] \sim B$
Substitutability	If an agent is indifferent between two lotteries A and B, then the agent is indifferent between two more complex lotteries that are the same except that B is substituted for A in one of them.	$A \sim B \Rightarrow [p, A; 1 - p, C] \sim [p, B; 1 - p, C]$
Monotonicity	Suppose two lotteries have the same two possible outcomes, A and B. If an agent prefers A to B, then the agent must prefer the lottery that has a higher probability for A (and vice versa).	$A \succ B \Rightarrow (p > q \iff [p, A; 1 - p, B] \succ [q, A; 1 - q, B])$
Decomposability	Compound lotteries can be reduced to simpler ones using the laws of probability.	$[p, A; 1 - p, [q, B; 1 - q, C]] \sim [p, A; (1 - p)q, B; (1 - p)(1 - q), C]$
Existence of a utility function	If an agent's preferences obey the axioms of utility, then there exists a function U such that $U(A) > U(B)$ if and only if A is preferred to B, and $U(A) = U(B)$ if and only if the agent is indifferent between A and B.	$U(A) > U(B) \iff A \succ B$; $U(A) = U(B) \iff A \sim B$
Expected utility of a lottery	The utility of a lottery is the sum of the probability of each outcome times the utility of that outcome.	$U([p_1, S_1; \ldots; p_n, S_n]) = \sum_i p_i U(S_i)$

Table 2.3: Axioms of utility from [NM44] and their consequences according to [RN21]

Combining utility theory with probability theory leads to *decision theory*, according to which maximising the expected utility is one way to make rational decisions. The *expected utility EU* of an action a is defined as the sum over the utility of outcomes that it might yield, times the probability of their occurrence [RN21]. This means that an action a can lead from state s to different states s' with corresponding probabilities. The expected utility of an action can therefore be written in terms of the utility of expected states as

$$EU(a) = \sum_{s'} P(Result(a) = s') \cdot U(s'). \tag{2.5}$$

Where $U(s')$ is the agent's utility of state s' and $P(Result(a) = s')$ is the probability of ending up in a specific state s'.

An agent striving to find the action that yields the maximum expected utility (MEU) is then called a *rational* agent. While the corresponding plan is referred to as *optimal* plan. A criticism of expected utility maximisation is given by Smith and Winkler [SW06]. The authors show that the selection procedure of the option that yields the MEU generally overestimates the expected return and call it the *Optimizer's Curse*. They further point out that the expected disappointment can be predicted using Bayes' rule. Russell and Norvig point out that decision theory provides several ways of making rational decisions other than maximising expected utility. Furthermore, utilities and thus underlying preferences are often implicitly defined as linear functions, not considering an agent's attitude towards risk and uncertainty. As they put it: "[...] an agent might prefer to have a prime number of dollars in its bank account; in which case, if it had $16 it would give away $3. This might be unusual, but we can't call it irrational."[RN21]

Savage [Sav54] is one of the first to formalise a set of other rational decision-making strategies. He presents the concepts of sufficiency and minimaxity. A statistical estimator is said to be *sufficient* when it captures all relevant information about an unknown parameter of interest. Thereby ensuring, that no additional information could improve the estimation. Furthermore, he introduces the *minimax* decision rule which minimises the maximum possible loss associated with a decision. This decision rule provides robust decisions against worst-case scenario outcomes and marks a conservative - or as we may see risk-averse - attitude. Savage emphasises the importance of individual beliefs and preferences especially in the face of uncertain outcomes. He thereby extends the notion of expected utility beyond level 3 of uncertainty from Table 2.2. This was limited to levels 2 and 3, as a probability distribution is required. He argues that an agent's decision-making can be represented by assigning subjective probabilities and utilities to different outcomes, generally referred to as *Savage's theorem*. As we have stated before, probabilities cannot always be observed completely and objectively, so personal expectations and preferences are more important. However, human preferences do need to be understood, but the tools of utility theory help to model them in a consistent manner.

Against the newly unfolded background of maintaining control over AI systems, Russell and Wefald [RW19] argue that such systems should never assume to be completely certain about user preferences. This is due to multiple reasons: For one, humans are sometimes not even aware of, or sure about their own preferences. Second, even if they are, there is no guarantee, that humans always choose their actions completely compliant with their preferences. Thus, when considering human actions as indicator of their preferences, there might be some distortion along the way. Furthermore, chosen actions can rarely be traced back to a particular preference, as multiple incentives may be involved, even when only considering a single human being.

A more realistic model of human preferences and resulting actions was studied by Kahneman [Kah11], who tried to model the often irrational seeming behaviour of humans by two internal entities or systems. *System 1* represents the fast, automatic, and intuitive thinking, which relies on heuristics, patterns, and associations stored in memory, but is, however, prone to cognitive biases, leading to errors in judgement and decision-making. *System 2*, on the other hand, represents slow, deliberate, and analytical thinking, thereby involving abstract thinking, considering multiple factors, and engage in logical and systematic thought processes. It can override automatic responses from System 1 if necessary [Kah11]. Especially in cases where agents assign different utilities to events or where the probabilities of outcomes are not accessible, an agent's *risk attitude* is another important factor for his decisions. Kahneman gives examples for human decision-making, that contradict an objective outcome maximisation. The examples he gives are lotteries where the chances of winning are close to an extreme probability (either 0 or 100 % success). This makes agents unwilling to gamble, or an agent's possession distorts its assessment of possible outcomes.

2.2.3 Risk Attitudes

The observation of different risk attitudes described by Kahneman does not contradict the utility maximisation model but can be described by it easily. Such behaviour simply shows that an agent has a non-linear utility function (compare Equation (2.4)) for a given quantity (e.g. money). For simplicity of presentation, we now move from abstract preferences to preferences over measurable quantities. We assume that an agent has a strictly monotonic rising preference towards this quantity. That is, he always prefers to have more of that quantity over less. In the context of planning theory there are *gain*-domains, which are about *rewards* r and *loss*-domains, which are about *costs* c. However, costs can just be negative rewards and vice versa (i.e., $r = -c$) in a domain where one can win and lose a resource. The MEU principle then leads to the fact that costs are minimised, and/or rewards are maximised. *Resources* are certain quantities that are needed for a specific purpose and are usually scarce goods, e.g., money.

The simplest way to represent a strictly monotonic preference for some resource is by a linear utility function, that assigns utility values proportional to the quantity of interest. Note that this is only possible if the amount of resource is measurable. Given such a utility function means, that an agent values having double the amount of a quantity twice as much. It also means that an agent, obeying the axioms from Table 2.3, is indifferent to a lottery yielding double the amount with half the probability. This agent is called *risk-neutral*, as it is indifferent to a lottery with arbitrary probabilities, if only the promised outcome compensates the probability. In contrast to that, an agent is said to be *risk-averse*, if it prefers an amount less than half but with double the probability. That is, its utility function has a concave down curvature - it assigns disproportional higher utilities to lower rewards. The functions shown in Figure 2.2 can be used to represent such behaviour.

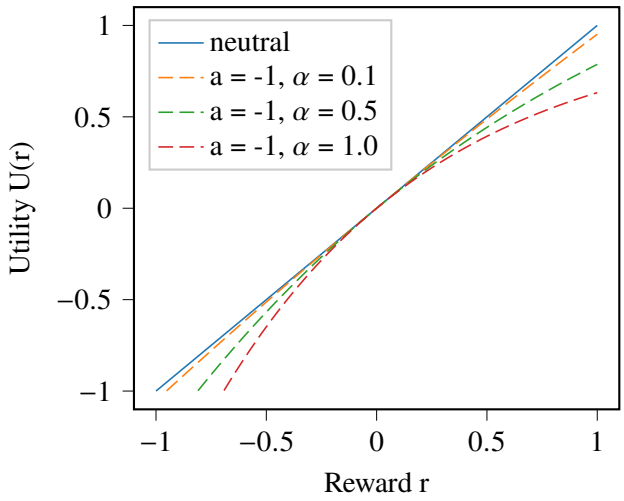


Figure 2.2: Utility functions describing a risk-averse attitude

Where the x-axis represents the reward r and the y-axis the associated utility $U(r)$. Note that a plot of cost would be mirrored along the y-axis. The linear blue curve represents the reference of a neutral attitude. While the dashed curves describe stronger risk-averse attitudes as they bend away from it.

On the other hand, an agent is said to be *risk-seeking*, if it prefers double the amount but with less than half the probability. Meaning, its utility function has a concave up curvature - it assigns disproportional higher utilities to higher rewards. This type of behaviour can be represented by functions such as the ones shown in Figure 2.3.

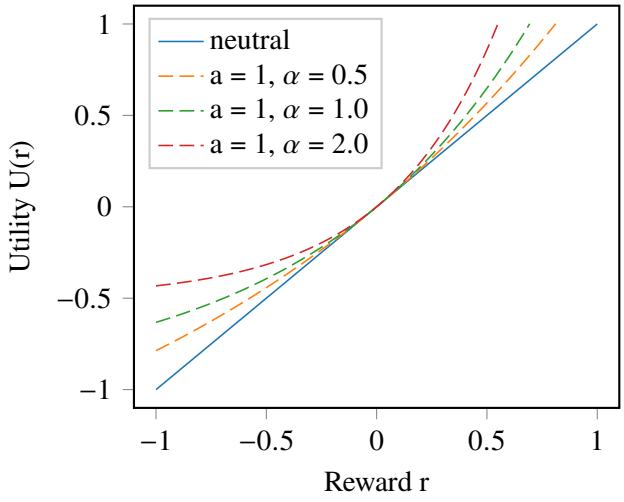


Figure 2.3: Utility functions describing a risk-seeking attitude

Where the x-axis represents the reward r and the y-axis the associated utility $U(r)$. The linear blue curve again represents the reference of a neutral attitude. While the dashed curves describe stronger risk-seeking attitudes as they bend away from it.

In real-world situations the aforementioned risk attitudes reflect the agent's beliefs about outcomes and underlying probabilities, rather than observed ones. This fact was already introduced before as Savage's theorem. However, as decision-making proceeds, an agent might change its preference

and thus its utility function. For example, the possessions held prior to a lottery and the general history of the decision maker influence how they behave [AGA22]. Such a shift in preferences can be depicted by means of a *dynamic* risk attitude. In order to model changing risk attitudes depending on the state we refer to the class of so called *m-switch* utility functions, introduced by Bell [Bel88]. In this thesis, we do not want to deal with dynamic risk attitudes, but we want to give an idea that there are ways to deal with such scenarios. Here we focus on the class of *static* utility functions discussed in [AGA22]. Accordingly, a static utility function is described by

$$U_s(r) = \begin{cases} r & \text{if neutral} \\ \frac{a(e^{a \cdot \alpha \cdot r} - 1)}{\alpha} & \text{otherwise} \end{cases}. \tag{2.6}$$

Where r refers to the original reward, a is an attitude-determinant coefficient, and α is a curving coefficient driving the shape of the utility function.

The functions depicted in Figure 2.2 and 2.3 follow Equation (2.6), with $a = -1$ (risk-averse) and $a = 1$ (risk-seeking) respectively. There exist other functions that can represent such risk attitudes. Functions only need to obey the Axioms given in Table 2.3. However, the proposed type, with only two parameters a and α, is a versatile and elegant one.

So far, we have only looked at maximising the utilities of single agents and about single resources. However, it gets more complicated when there are several preferences that should be fulfilled. This can be addressed by using a combined utility function that aggregates multiple preferences and thus weights different preferences against each other or by viewing them independently. This subject is further elaborated in Section 2.3.4.

Let us conclude this part by summarising risk attitudes as the willingness of an agent to accept nonlinear relations between outcome and probability. The agent however still maximises its *subjective* utility by calculating optimal solutions. Another interesting decision-making strategy is *Satisficing* introduced by Simon [Sim56]. In contrast to von Neumann and Morgenstern, Simon argues that a rational choice in reality involves settling 'for an alternative that is judged to be good enough in the light of available information and prevailing goals.' Winter [Win18], reflects on these topics and emphasises that an *optimum* is only defined among other less optimal alternatives. She goes on to say that Satisficing is actually a different approach to the problem: The agent does not seek to find maximisation from a set of options but seeks plausible rules to generate the next alternative [Win18].

2.2.4 Preference Fulfilment

As mentioned earlier, assigning utilities, or defining decision-making rules in general is a way to express what an agent wants. Utility functions are a means to transform preferences into mathematical constructs. Let us now take a closer look at how preferences and their fulfilment are perceived from the outside.

One of the simplest, but most expressive models of human preference fulfilment is the Kano model for customer satisfaction [KSTT84]. A manufacturer - like an AI system - does not know all the preferences of its customers or users. The inclusion of hidden underlying preferences and their influence on satisfaction is of particular importance in the design of such a system.

In their original form KANO et al. mention three categories of user requirements:

1. *Must-be* or *basic* requirements.
2. *Performance* or *one-dimensional* requirements.
3. *Attractive* or *exciting* requirements.

Each type of requirement or preference leads to a different level of satisfaction, depending on their fulfilment. The model from [KSTT84] maps this relation as reported in Figure 2.4.

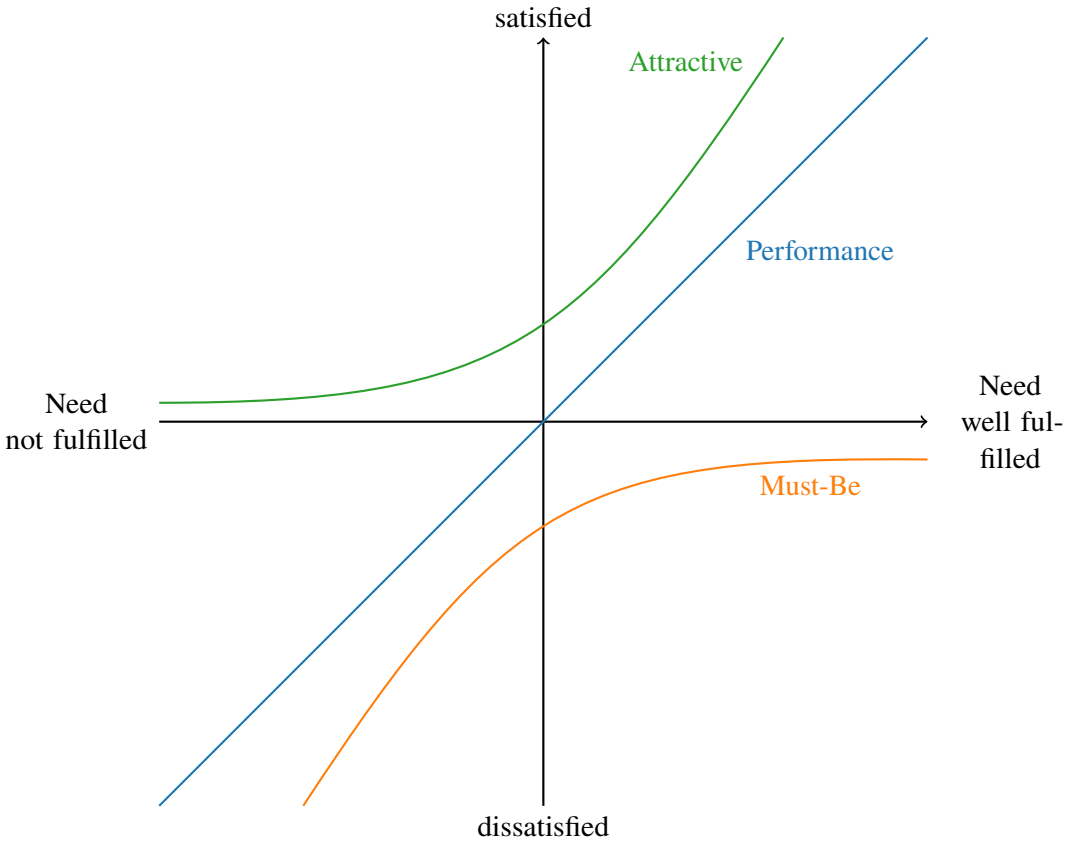


Figure 2.4: Kano model for customer satisfaction according to [KSTT84]

Where the x-axis represents the level of implementation of a particular functionality and the y-axis represents the resulting customer satisfaction.

2.3 Automated Planning

Now that we have established the basics of decision theory, we want to extend the terminology given in Section 2.2 and formally define how to use decision theory in planning domains. For this purpose, we will look at various concepts that make it possible to represent real-world environments and to reason about them in an automated way. For this, it is important to separate acting in this environment from planning, as mentioned before.

2.3.1 Classical and Hierarchical Planning

For a fundamental insight let us start by reviewing classical planning as well as the concept of hierarchical planning.

Definition 2.3.1
Formally a planning problem is a 3-tuple $\langle s_0, g, D \rangle$. *With* s_0 *being the* initial state*,* g *the* goal state *and* D *the domain respectively. The domain* D *contains* A*, the set of all actions, and* P*, a set of predicates, which describe variables and their relationships to each other and thus refine the set of states* S*. An action can specify a* precondition *that must before it can be applied. Furthermore, it specifies a* DEL *(delete) and an* ADD*-List, stating how it manipulates the state - also referred to as its* effect *[RN21].*

When an action is applied, the systems transits from one state s to the next state s'. The next resulting state s' can be derived by removing the predicates given in the DEL-list from s and appending the ones from the ADD-list to s. Formally given by

$$s' = RESULT(s, a) = (s - DEL(a)) \cup ADD(a). \tag{2.7}$$

With the help of Equation (2.7) and the set A, all possible subsequent states can now be calculated from the initial state s_0. These can then be compared with the goal state g. It is also possible to rearrange the equation and try to infer the initial state from the goal state. Basically, a search tree is created in which a connection between the initial state and the goal state must be found. This path then represents the list of actions - i.e., the plan - whose application to the initial state leads to the goal state and thus solves the planning problem. A simple way of representing this is a *network* or *graph* in which states are nodes and actions are links. We will come back on how to search for paths in such graphs later, but first want to introduce another concept that can help to solve problems.

Complex problems are usually solved by breaking them down into smaller problems until these problems are simple enough to solve them immediately. Hierarchical planning mimics this strategy by extending the framework of classical planning by *methods*. Methods describe ways that high-level, or *compound* tasks can be decomposed into smaller tasks. These, in turn, can be either compound tasks or *primitive* tasks that can be solved immediately. This is modelled by an *operator*, yielding an action that accomplishes the primitive task. The reason for this separation is that there can be several actions that fulfil a certain task. Thus, the cost of an action is transferred to be the *operator cost* instead. This circumstance once again illustrates the difference between planning and execution: An operator is applied to a primitive task during planning, while an action is what is later executed in a particular state.

There exist various concepts of this hierarchical kind in AI planning, which are well summarised by Bercher et al. [BAH19]. In the context of this work, we want to focus on the class of HTNs. Where the approach is to start with an initial task (network) and decompose it using the given methods, until every task is primitive. In contrast, Hierarchical Goal Networks (HGNs) define goals that are broken down into subgoals. In general, however, this is only a difference in how to think of a specific problem, and the two paradigms can be transformed into each other. There are also frameworks that combine both approaches, so-called Goal-Task Networks (GTNs).

Definition 2.3.2
An HTN planning problem is a 3-tuple $P = \langle s_0, tn_0, D \rangle$. With tn_0 being the initial task network and s_0 the initial state as before. Further the domain $D = \langle O, M \rangle$ featuring a set of operators O and a set of methods M to decompose [AGA22].

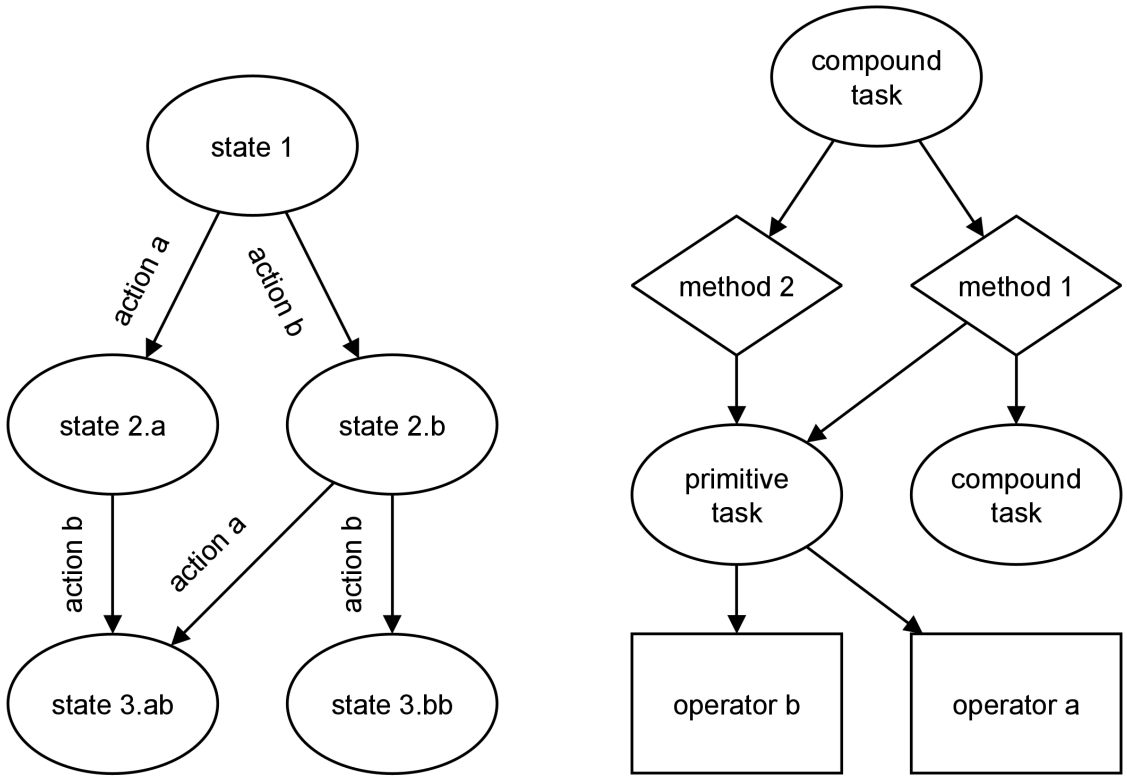
There are two popular ways to solve HTN planning problems: plan-based and state-based. These approaches differ in their search space representation: in *plan-based* (or *decomposition-based*) the search space exits only of task networks, with the advantage that these networks might be explored in various ways. In contrast, in a *state-based* (or *progression-based*) approach, the search space is a subset of the state space, which in turn is constrained by task decomposition [AGA22]. This means that task networks must be explored in a chronological fashion. From Equation (2.7) it can be seen that a consecutive state can only be computed when an action was chosen. It is only possible to continue searching from the next state after it has been calculated. In contrast to classical planning, the completion of the initial task marks reaching the goal state. Consequently, in state-based search, variables must be *bound* early, with the selection of operators. That is, if multiple instances of the same object exist, it has to be decided which one to use or which variable to assign, as the operator is chosen. This setup of the search space is a limitation of state-based over plan-based, as advanced search algorithms cannot be applied directly. However, state-based approaches have an advantage over plan-based approaches in their clarity and expressiveness [GA15].

Note that hierarchical planning can be seen as a way to incorporate expert knowledge of how problems are usually solved in a specific environment. That is to say, hierarchical problems can in general be transferred into classical search problems with similar actions, when method preconditions are treated accordingly. In other words, actions in classical search are meant to model physics, while the hierarchical structure of methods can be used to introduce advice and thereby narrow the search space. [BAH19]

2.3.2 Search for Optimal Solutions

We have seen what planning problems are and how they are modelled, let us now look at how they can be solved and how to find optimal solutions. As already mentioned, a good graphical representation of planning problems is possible via directed graphs. Where *nodes* (or *vertices*) represent states and *edges* (or *links*) represent actions, which would be the search space of state-based planning (Figure 2.5a). In contrast, the plan-based approach can be represented in such a way that tasks and methods form nodes which are connected by edges. These edges then show that a task can be decomposed by a certain method, or that the decomposition leads to the following subtasks.

Such a comparison has the advantage that many optimisation concepts which are known from *graph theory* can be applied. Graph theory is about the relation between nodes, which are represented through links. Applying the concept of graph theory to planning, using the representations of Figure 2.5 is to investigate the relation between the initial state and the goal state, or between the initial task (network) and the resulting actions. Weights can be assigned to links to represent, for example, a distance or a cost. Thus, it is also possible to find the shortest or cost minimal path. Two straightforward search algorithms for directed graphs are Breadth-First Search (BFS) and Depth-First Search (DFS). Where BFS visits all vertices at one depth before moving on to the next



(a) Example of a state-based representation

Where nodes depict states and edges actions.

(b) Example of a plan-based representation

Where ovals represent tasks, rhombuses decomposition methods and rectangles operators. Edges starting from an oval indicate which methods / operators can be applied to a task, and edges from a rhombus show into which subtasks a method decomposes a compound task.

Figure 2.5: Representations of HTN constructs

level. DFS always follows the first edge until it reaches an end (a node with no outgoing edges), before tracing back and branching differently at the last node. Both methods visit all nodes and can determine the shortest path between two nodes using the weights of the edges.

However, the algorithms mentioned must know the entire graph in order to determine the shortest path with certainty. The size of the graph depends on two factors: The branching factor b and the maximum depth d. In the state-based representation, b is determined by the number of applicable actions and d by the number of actions necessary to reach the goal. Since not all actions are applicable in every state, this narrows the search space, but a complete search is usually too exhaustive. In the plan-based representation, b is determined by the number of methods that can decompose a task and by the number of subtasks that are decomposed into. The depth d describes the distance between the initial task and an actual operator. Again, the size of the search space can grow rapidly, so an exhaustive search is not desirable.

More sophisticated search strategies such as *A** use a *heuristic* to guide the search and only visit nodes that promise a short connection to the goal. Thus, not all nodes have to be visited, which speeds up the process. This means that the algorithm discards certain paths because they no longer lead to the shortest path, which is called *pruning*. However, the involved heuristics must be defined beforehand, which is why A* search belongs to the class of *informed* search paradigms. The nature of the heuristic determines the optimality guarantee of the algorithm. A heuristic is called *admissible* if it always (optimistically) underestimates the cost to reach the goal. Using an *inadmissible* heuristic, might yield a suboptimal solution, but at lower computational complexity. That is, deciding between searching for an optimal or suboptimal solution by using an admissible or a weighted inadmissible heuristic respectively [RN21].

Like System 1 described by Kahneman, informed search makes use of simplifications to arrive quickly at the optimal solution. But with every simplification comes a loss of accuracy. That is what heuristics do, they give a good estimation of reality with less computational effort. As already mentioned, the aim of rational decision systems is to make the best possible decision, considering the uncertain future. The term *optimal* is used to describe the best of all possible solutions. In the view of simplifications and the associated reduction of the search space, optimality is also limited. As Kahneman and Simon have shown, heuristics (System 1) are responsible for most human decisions. This means that the possibility of a simple (good enough) solution is often preferred to a complex calculation of an optimal one.

In its original form, classical planning is about finding feasible plans, it is therefore Satisficing in nature. The transfer of the problem to a shortest path search in graphs makes it possible to search for optimal plans. Uninformed search algorithms, however, must visit all nodes in order to determine the shortest path. The effort of this search increases with the complexity of the domain. A reduction of the effort is possible by using informed search algorithms even for large problems, but suitable heuristics are necessary for this. These heuristics must first be derived to use them effectively for the search. Besides automated planning, there are other forms of representation that describe decision-making in real-world environments.

2.3.3 Automated Decision-Making

The presented planning concepts of Classical planning and HTN planning have an important premise: The next state can be determined according to Equation (2.7). This means that actions have a deterministic effect, and the state is only changed as consequence to an action of the agent. However, this assumption is rarely true in reality. Other concept for the representation of real-world environments that do not require this assumption are *sequential decision processes*. A well-known representative are Markov Decision Processes (MDPs) which, however, rely on the assumption, that state transitions are *Markovian*. Again, the system is always in a certain state s. In contrast, the transition to the next state s' does not only depend on the action chosen by the agent, but there is a probability function $P(s'|s,a)$, which indicates the probability of ending up in a next state. Instead of a goal state, there is a reward function $R(s,a,s')$, which measures the value of being in a certain state.

Definition 2.3.3
Formally an MDP is a 4-tuple $\langle S, A, P, R \rangle$. Where S and A are finite sets of states s and actions a respectively. $P(s'|s, a)$ is a transition model describing the probability of ending up in state s', given that the agent applies a certain action in a certain state. $R(s, a, s')$ is a reward function, that specifies the agent's incentive of being in a specific state and selecting a specific action [RN21].

There are several methods to solve this type of problem, the goal always being to find a *policy* π that yields an action depending on the state ($\pi : S \rightarrow A$). That is, because of the randomness of the state transition, it is not enough to specify a plan, but rather a strategy for every state, that the agent might be in. So, unlike planning, following a fixed plan does not guarantee ending up at the goal. Known methods for solving such problems are value iteration, policy iteration, Linear Programming (LP), Reinforcement Learning and the Monte Carlo method. The reward function can be used to depict utilities. Thus, using the probabilities given by the state transition model, it is possible to compute an *optimal policy* $\pi*$ that yields the highest expected utility.

As a naive approach, any combination of state and action can be simulated. Keeping track of the reward of this combination, the action that yields the highest expected reward can be determined. With enough simulation, this function converges to the optimal policy $\pi*$. However, it should be noted that the size of the problem increases with the size of the sets S and A. There are only some special cases for which efficient solutions exist since these sequential decision processes may or may not have a fixed horizon. Some solutions are known from LP, as well as control theory.

One way to find optimal solutions to such problems, is to formulate them as a convex optimisation problem. Thereby constraints form a convex region that has one global maximum by definition. In this way, an optimal solution can be efficiently calculated using the equations that define the constraints. Known solutions are LP and (Mixed-)Integer LP, where constraints are linear inequalities. With (Mixed-)Integer LP, some variables are restricted to be integers and thus discrete in order to be able to solve them efficiently.

An extension of MDPs are Partially Observable MDPs, where the current state of the world is not directly observable. The agent therefore only has a belief of what the current state could be, which is supported by some observations or sensor readings. After an action the state estimate is updated using the last state estimate, the action applied and new observations.

HTNs and MDPs are two paradigms that help to find actions to problems, when it is possible to phrase the environment in their language. In fact, sequential decision problems include search and planning problems as special cases [RN21]. HTN planning is about finding one or more executable plans, i.e., a list of actions that lead from the initial to the target state. Sequential decision processes, on the other hand, are about executing the optimal action at each point in time so that the accumulated reward is maximised. A solution is always possible through a complete simulation of the subsequent states, but this is not desirable. Suitable heuristics or closed formulas are needed to quickly find optimal solutions.

2.3.4 Multi-Objective Optimisation

There exist various concepts to deal with a multi-objective (MO) problem to satisfy. In general, the different intentions can only be fulfilled to a certain degree at the same time. Additional criteria can either be taken into account *a priori* - during the search - or solutions found are examined *a*

posteriori for their fulfilment of intentions. If all objectives can be depicted using utility functions and these are known, they can be depicted in the form of a *utility vector*. This vector summarises the individual utilities without mixing objectives inseparably. This is especially important when the utilities express preferences about different things.

An a posteriori solution can again be found via LP. If the utility functions are already known, however, a priori optimisation is possible by comparing utility vectors during computation. Since there is no longer a clear optimum, the set of solutions is further narrowed down by other measures. In general, only the *Pareto optimal* solutions are considered. Therefore the utility over a state s is expressed by a k-dimensional vector $\vec{u}(s) \in \mathbb{R}^k$. Where the expected utility of an action a is then given by $\vec{eu}(a) = \sum_{s'} P(Result(a) = s') \cdot \vec{u}(s) \in \mathbb{R}^k$. The expected utility $\vec{eu}(\Pi)$ of a plan Π is the sum of the expected utilities of all its actions $\vec{eu}(\Pi) = \sum_a \vec{eu}(a) | a \in \Pi$.

Definition 2.3.4
$\vec{eu}_1$ Pareto dominates $\vec{eu}_2$, denoted $\vec{eu}_1 \succ \vec{eu}_2$, if and only if (iff) for all its components $i = 1...k$: $\vec{eu}_1^i \geq \vec{eu}_2^i$ and additionally $\vec{eu}_1 \neq \vec{eu}_2$. Note that $\vec{eu}_1 \succ \vec{eu}_2$ implies $\vec{eu}_1^i > \vec{eu}_2^i$ for at least one i. Dominance is a strict partial order, i.e., it is transitive and asymmetric [GHTT22].

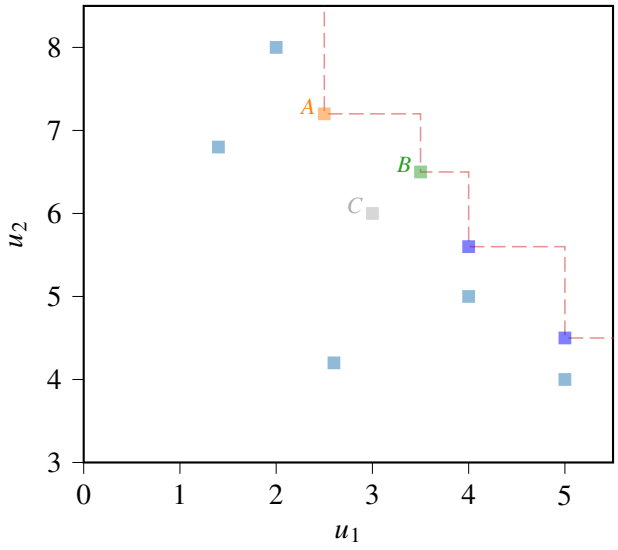


Figure 2.6: Pareto front: Subset of non-dominated solutions

Where the horizontal axis represents utility function u_1 and the vertical utility function u_2. Squares mark the utilities of possible solutions corresponding to u_1 and u_2. The Pareto front is represented by connecting the non-dominated plans by means of the dashed red line. Examine the points A, B and C, which are highlighted in different colours. Point B Pareto dominates point C, while A and B are both non-dominated. Note that C is only dominated by B, but not by A, i.e., if the point B were unknown, C would also be non-dominated.

The subset of utility vectors that marks the outline is then referred to as the *non-dominated* set. In other words, those solutions for which no other solution exists, that fulfils one objective better without sacrificing another. However, this subset usually still features a variety of solutions. To

choose one solution, some other criteria to optimise for is needed, e.g., a weighting between utility functions [MA04]. Considering the simple case of two utility functions u_1, u_2, all solutions can be plotted on a utility plane as depicted in Figure 2.6.

2.3.5 Risk Aware Planning

A novel framework for HTN planning that includes agents' risk attitudes is presented in [AGA22]. This model extends HTN planning to include operators that can have a deterministic effect with variable costs, or those that have a probabilistic effect with a given probability. Operators are therefore considered to be *cost-variable*. The expected utility of an operator can be calculated according to Equation (2.5). Depending on the accessibility of probabilities, these actions are either referred to as *risk* or *uncertainty inducing* and their domains are likewise called *risk* or *uncertainty involving* domains.

Definition 2.3.5
A risk-aware HTN planning problem is a 4-tuple $P_r = \langle s_0, tn_0, D, U \rangle$. With s_0, tn_0 the initial state and task network respectively. The domain $D = \langle O, M \rangle$ as before, but now involving cost-variable operators and a utility function U that expresses a certain risk attitude [AGA22].

Like with MDPs, there is a probability distribution for the state transition in this description, which is, however, encoded in operators and only depends on the actions of the agent itself. In contrast to the reward function $R(s, a, s')$ of MDPs, there is the initial task network tn_0. The computation of cost-optimal plans, which fulfils the initial task, opposes the determination of an optimal policy. In the case of deterministic actions with a single effect, the solution can again be found by transferring it to a search in a corresponding graph. The selection of actions can be done based on the risk-aware utility function U. The methods of the domain can again narrow the search space in the process of decomposing the initial task. Furthermore, it is possible to select methods based on their expected costs, and thus accelerate the search.

[AGA22] offers two approaches, one for each representation: Generating an admissible heuristic by relaxing the problem to classical planning for the state-based case. For the plan-based case, the costs of methods are determined to be the sum of all cost estimations of tasks in their task network. The cost of tasks, on the other hand, is determined to be the minimum cost among all estimated costs of methods that can decompose it. This propagation of costs allows to estimate them beforehand and rank the order in which methods are decomposed.

3 State of the Art

To design a planning domain for the problem at hand, we first need to look at how three problems are approached in state of the art solutions. First, the components that are used for building automation and how they are used in smart buildings. Then, how planning problems of uncertain environments and multiple objectives are modelled and solved. Finally, how to combine these two fields: Which approaches of planning algorithms are used for the operation of smart buildings. From these approaches and their shortcomings, we can then derive where to locate a new solution.

3.1 Building Automation and Occupant Awareness

Modern buildings incorporate advanced technologies that seamlessly control, monitor, and optimise various building functions. These include heating, ventilation, air conditioning (HVAC), lighting, security, access control, energy management, etc. Key components of modern building automation systems include sensors, actuators, controllers, communication networks, and software platforms. Their purpose is to improve energy efficiency, occupant comfort, safety, and operational effectiveness while reducing costs and environmental impact. Clements-Croome [Cle20a] names the following key criteria to realise good-quality intelligent buildings:

- Satisfying stakeholder objectives,
- Meeting social and environmental needs, and
- Recognising available resources.

Similar aspects are also given by Bauer et al. [BMS10]. However, mentioned criteria are very broad and must be mapped to individual aspects. Especially as non-residential buildings are often occupied and operated by different entities, we need to look more closely at their, sometimes conflicting, preferences. We therefore need to review how considered subsystems transfer above criteria.

An in-depth explanation of energy management strategies is presented in [SP16]. State of the art concepts include load scheduling, demand response programs and energy-efficient setpoint adjustments. Further concepts by Clements-Croome include adaptive control algorithms, demand-controlled ventilation, and zone-based temperature control. Lighting control strategies are for example daylight harvesting, occupancy sensing, and dimming techniques. To enable such techniques, sensors for environmental monitoring, occupancy detection and indoor air quality assessment are needed. Furthermore, an interface to receive user instructions and feedback as well as facilities with corresponding actuators are necessary.

Various optimisation strategies in the area of Battery Energy Storage System (BESS) management are discussed in [JKU+13]. Here we pick up the following:

- Time-of-Use Optimisation or Peak Shaving: This strategy involves charging the BESS during off-peak hours when electricity prices are lower and discharging during peak hours when prices are higher. It allows for cost savings by utilising cheaper electricity and potentially earning revenue through demand response programs.
- Self-consumption Optimisation: This strategy focuses on maximising self-consumption of solar or renewable energy generated on-site. The BESS is charged during periods of excess renewable generation and discharged when demand exceeds supply. It allows for greater utilisation of clean energy, energy independence, and potentially reduces reliance on the grid.
- Demand Charge Management: For commercial or industrial applications with demand charges, the BESS can be used to manage and reduce peak demand. By discharging during peak load periods. It helps to flatten the demand curve, mitigating high demand charges and optimising overall energy costs.

In practice, these optimisation strategies are usually made possible by means of LP or heuristic search. The integration and interaction of renewable energy sources, storage and the smart grid is explored by [Mom12]. The author presents the smart grid as a demand response regulator. In this context, he also discusses the Time-of-Use concept and DAP, through which a reduction in peak loads can be achieved from a global perspective.

Other relevant topics from this area are the scheduling of appliances and the prediction of harvestable renewable energies. Probabilistic forecasting models for solar irradiance are compared in [DSS19; PMZ21], showing that a normal distribution yields the best predictions in most comparisons. First Come First Serve scheduling and Appliances First scheduling polices on preemptive and non-preemptive appliances of residential buildings are compared in [AW14]. Reducing customer electricity bills and smoothing peak demand on the grid. The authors show that an Appliances First policy is the more balanced of the two. They also emphasise that grid stability could be threatened by uncoordinated battery charging and discharging. In contrast a tunable Deep Q-Network algorithm is presented in [LL20]. It trades multiple objectives, namely cost, peak power and punctuality. The authors use a Pareto Front to visualise different results due to a change in reference weights. Furthermore, they use a binary measure to determine punctuality in terms of user desired operation / completion time as well as a binary comparison to the average electricity price of the day.

In the field of room management and building automation, both [BMS10] and [LK20] mention common comfort zones for offices and such buildings. These are partly derived from national and international standards, but also reflect analyses of well-being under different conditions. According to [DD07], indoor temperature should be around $24.5°C$ degrees in summer and $22\,°C$ in winter, or within a range of ±3 degrees around these values, depending on the class of the building. For the illumination of workplaces, conference rooms and meeting rooms, [DD21] prescribes a brightness level of 500 lux [LK20].

Regarding fresh air supply [DD12; DD19] defines two aspects: Firstly the number of persons, according to which the supply of fresh air should be $4 - 10\frac{l}{s \cdot person}$ depending on the building class. Secondly, the air pollution generated in the room plays a role. Accordingly, the ventilation rate should be $0.3 - 0.5\frac{l}{s \cdot m^2}$, depending on the pollution load. A corresponding parameter for determining air quality is the air quality index (AQI). The index summarises the load by different pollutants and presents them on a scale. Applied scales vary, depending on the country, but most use values from 0 (good) to 500 (very unhealthy) [VLN08].

In this context we need to look at the measurement of preference satisfaction. Recall the Kano-Model from Figure 2.4: A similar model of occupant satisfaction is presented by Clements-Croome [Cle20a] and called *Flourishing*. Surveys like [FSG+12] show how certain climate characteristics affect the well-being of occupants. We are only going to look at one specific part of this study: Indoor climate quantities like temperature and humidity, are only noticed when they do not fulfill occupant preferences and otherwise have no impact on satisfaction. That is, they are perceived as must-be requirements according to the Kano-model.

3.2 Hierarchical and Multi-Objective Planning

As emphasised in Section 2.3: The objective of classical planning is to determine whether there is a feasible plan for a posed problem. Extensions of this concept consider the unpredictability of outcomes, but planning is usually about finding a sequence of actions with the highest possible success rate. Hierarchical approaches help to increase the structure and thus limit the search space. The concept of expected utilities helps to find optimal plans even when actions have variable costs. On the other hand, MDPs account for possible external influences. However, this reduces the expressiveness and increases the complexity of finding optimal solutions. Here we look at how recent approaches can help to find qualitative plans in the case of actions with uncertain effects and multiple objectives.

3.2.1 Implementations and Extensions of Planning

The quasi standard to describe classical search problems is the Planning Domain Definition Language (PDDL) presented in [GKW+98]. Ever since there have been various updates and refinements, like Numeric Fluents, Durative Actions, Continuous Effects, Preferences, Processes and Events to name a few (compare [Gre]). All these concepts attempt to grasp the nature of real-world environments. However, classical planning does not allow for the introduction of guiding knowledge in the way that hierarchical structures do. Unlike classical planning, various languages and concepts were suggested for hierarchical planning problems [BAH19]. New concepts are often presented at the International Planning Competition (IPC), several planning constructs are listed on their website [Ber]. Most description languages are oriented towards PDDL, where the most used might be Hierarchical Domain Definition Language (HDDL) introduced by Höller et al. [HBB+20].

Besides the description language there are various concepts on how to incorporate the hierarchical nature. Georgievski and Aiello [GA15] analyse and classify different styles of HTN planning. They categorise them by, for example, variable commitment strategy and constraint management. Bercher et al. [BAH19] update this classification and extend it by HGNs and GTNs, which are capable of different goal representations. Programs that incorporate these concepts further differ in the way they search for a feasible plan.

3.2.2 Guiding Search Strategies

We already looked at the two most common ways to solve HTN planning problems, namely plan-based and state-based search. We also saw that state-based planning has to work in a "left-to-right"fashion, as arising states have to be computed in a logical order. On the other hand, plan-based search can make use of sophisticated search strategies. Similar strategies are applied in classical planning, for example by computing a bound on the number of actions needed, assuming that they do not have negative effects (no DEL-list).

Höller et al. [HBBB20] explore how to transfer such heuristics to state-based HTN planning. The authors introduce a generic method to use arbitrary classical heuristics to guide search. This is done by transforming the HTN planning problem into a classical problem used to calculate the heuristics. Revealing that such heuristics can speed up plan generation. However, in the end, hierarchical methods pose a way to incorporate expert knowledge on how to find a good solution quickly. That is an expert's subjective opinion on how a qualitative solution looks like and how to find it. At the same time, we want to take user preferences into account when searching for that very solution.

Geißer et al. [GHTT22] investigate how admissible heuristics can be created in the case of MO planning. They clarify that the often used ideal-point heuristic offers no improvement over an uninformed search if objective functions are mutually exclusive. The authors review NAMOA* search, a transfer of A* search to the MO case. As already mentioned, there is no longer a unique optimal solution, but only the set of non-dominated solutions that form the Pareto front. Based on this, suitable trade-offs between objective functions can be selected later.

Geißer et al. also investigate how to transfer well-known classes of admissible planning heuristics to the MO case. Namely abstraction-based, critical path-based, and LP-based heuristics. The authors also present two methods for calculating the maximum of two sets of cost vectors. However, there does not exist a unique definition of those since cost vectors are only partially ordered by dominance. Namely they present the *comax* function, which forms the component-wise maxima of cost vectors. As well as the anti-dominance maxima (*admax*), which is the union of the two subsets of vectors from each set, not dominated by the other set. For these, they find that comax is consistent, but its calculation is often more complex, while admax is inconsistent. Overall, they find that the MO heuristics presented are often more informed than their respective ideal point combination. Furthermore, some of the MO heuristics are far too expensive to evaluate, in particular the iterative method of operator counting. In conclusion, there is no universal method to create an admissible heuristic for MO planning problems [GHTT22].

3.2.3 Incorporating Preferences

There have been some attempts to incorporate preferences into HTN planning. Sohrabi and McIlraith [SM09] give an overview and suggest to consider preferences directly by soft constraints that might only be met to a certain degree. Georgievski and Aiello [GA14] are the first to introduce the notion of utility and risk attitudes into HTN planning. They, however, define risk aversity as minimising the MEU and risk seeking as maximising the minimum expected utility. Furthermore, they distinguish the attitude of consumption awareness as maximising the sum of utility values, while a risk neutral attitude minimises the average expected utility.

Alnazer [Aln19] implements this notion of attitudes by splitting planning into two steps. In a first pre-processing step utilities are computed, which then define the order in which decomposition methods are picked during the second step of computing the actual plan. Thereby achieving higher quality plans when acting in domains that involve limited resources. However, as utilities are computed before the actual planning step, they can only be constant throughout planning.

The framework proposed in [AGA22] generalises this approach by defining classes of utility functions that can represent various risk attitudes. Furthermore, incorporating expected utility maximisation into HTN planning can then be applied to compute optimal plans. However, doing an exhaustive search is in general infeasible and the remaining amount of a resource might be uncertain during planning [AGA22].

Another approach to incorporate uncertainty into HTN planning and compute optimal plans is presented by Richter [Ric18]. The author combines Partially Observable MDPs and HTNs by introducing abstract observations. Thereby constructing Partially Observable HTNs, which are shown to be optimisation problems. These can be interpreted as a generalisation of the deterministic totally ordered HTN approach [Ric18].

In conclusion, planning usually considers deterministic effects. Hence, incorporating *expected* utilities holds little advantage over using measurable costs. However, there are several concepts for designing heuristics and efficient search. Such heuristics can be transferred to MO problems, when the quality of plans is not determined by minimising costs. Several concepts have been presented to overcome other shortcomings, including a combination with MDPs. However, it is harder to compute optimal policies for the same size of problems. In addition, application to real-world problems requires not only concepts but also implementations. To the best of our knowledge, there is currently no out-of-the-box implementation of these concepts that can be used to solve new problems.

3.3 Automated Planning in Building Operation

Georgievski and Aiello [GA17] define a framework for modelling planning domains in the field of ubiquitous computing. They point out that solutions based on pre-defined responses are too limited to deal with the dynamics and uncertainty of this environment. Therefore they suggest to apply AI planning to tackle the problem. A standard procedure to set up such a system includes the following steps:

1. Knowledge acquisition,
2. Addressing problem requirements,
3. Domain model,
4. Implementation,
5. Evaluation.

Georgievski and Aiello [GA17] review 53 former studies and categorise them for better evaluation. A first clustering is done by means of behavioural inputs, behavioural outputs, physical properties (temporal and spatial) and the role of uncertainty.

The authors further distinguish by purpose of planning (control, assistive or organisational), as well as planning technique (such as hierarchical, probabilistic, heuristic-based, Constraint Satisfaction Problem (CSP)-based and partial-order). Other criteria are kind of plans, modelling constructs as well as monitoring and recovery. Stating, that HTN planning is the most frequently used, because of its causality, flexibility, and effectiveness. Probabilistic planning constructs such as MDPs capture uncertainty and quantify the quality of generated plans. Others try to memorise earlier decisions, thereby enhancing learning ability, adaptive capacity, and plan improvement.

Georgievski and Aiello find that most studies represent problems by classical planning. Most classified studies use PDDL as modelling language, several make use of hierarchical planning constructs [GA17]. Furthermore only 22 of 53 studies support monitoring and recovery. Planning is often done during runtime and systems generally consist of five components, namely: problem generator, planner, knowledge base, executor, and monitor. Four dimensions of evaluation are presented: demonstrations as well as quantitative, qualitative and usability evaluation. Only few studies exhibit more than two of these evaluations. Georgievski and Aiello conclude by giving directions for future work: Extended forms of procedural goals, Preferences, Human-aware plans, automatic knowledge engineering and Planning in real-life settings to name a few. Moreover, they emphasise the following limitation of planning constructs: HTN planning needs domain-specific knowledge. Solving large problems with Partially Observable MDP planning is difficult because probabilities create continuous and infinite belief states [GA17].

An early study in the field of smart non-residential buildings is by Georgievski et al. [GDP+12]. Their aim is to save energy and overall bill cost in offices connected to the smart grid. Starting point is the hypothesis that office managers will be incentivised to reduce energy consumption by attractive real-time pricing and so that the grid can be balanced on a global stage. In their use case, devices are modelled as a state machine with variable energy consumption levels and a centralised control. One of six policies can be specified for each device, namely: Repeat (duty cycle), Total (without exact timing), Multiple (number of jobs to be scheduled), Strict (interval), Pattern (only information to schedule other devices) or Sleep (during night). The authors further consider different energy providers, which offer a fixed amount of energy at a certain price. The optimal schedule is then claimed to be the one with the lowest price. They implement a priority queue with BFS to solve the optimisation problem.

To reflect realistic prices, they model a wind turbine as sunk cost and solar cells to have a cheaper price per unit of energy. A speciality of this report is the evaluation in a real use case. Data is gathered for two weeks as a baseline and the system is then applied to determine actual saving potential. The authors report up to 50 % savings in cost and up to 15 % in terms of energy. A search for the optimal schedule can take an impractically large amount of time. However, the problem is tackled by dynamically relaxing the requirement for optimality and thus searching for a Satisficing schedule. Furthermore, the system runs in the background to impact users as little as possible. Therefore, measures for HVAC are not addressed in the proposed solution, although they are recognised as very important contributors to peak energy demand [GDP+12].

Another related work is [Hal22], which deals with HTN planning in residential buildings. Thereby using the framework given in [GA17] to establish a domain model, analyse uncertainties and research procedural goals and preferences. Trying to depict the real world as accurately as possible, but always keeping an eye on computational efficiency. The following procedural goals are considered:

- Electricity pricing strategy in the smart grid: Aim to make cheap purchases and automatically support load-shifting as provider arranges prices accordingly.
- Uncertainty about the amount of generated solar energy: Keeping track of prediction error to calculate cost of operator to store solar power.
- Calculation of optimal temperature set-points for pre-tempering: Implementation of an algorithm suggested earlier by Avci et al. [AEA12].

The algorithm by Avci et al. identifies a price range (minimum to maximum) for each of the 24 hours of a day, based on the (aggregated) day ahead price. Each set-point is then assigned a value according to the price range of its hour. The values are represented by

- ρ_i averaged DAP for electricity at hour i,
- T_{min}, T_{max}: lower and upper bounds for temperature set-points,
- $\alpha \in \{-1, 0, 1\}$ occupants' tolerance level ($-1 \mathrel{\widehat{=}}$ prefer comfort, $0 \mathrel{\widehat{=}}$ neutral, $1 \mathrel{\widehat{=}}$ discomfort acceptable).

Temperatures are ordered from small to big (indicated by j) and for a neutral tolerance level each set-point is assigned a value $r_j = r_{j-1} + \frac{\rho_{max} - \rho_{min}}{T_{max} - T_{min} + 1}$. Whereas for $\alpha \neq 0$ the value is $r_j = r_{j-1} + (\rho_{max} - \rho_{min}) \cdot \frac{2^{\alpha(j-1)}(1-2^{\alpha})}{(1-2^{\alpha n})}$, i.e. price bounds do not grow linearly (compare [Hal22]). We want to point out, that this algorithm is in itself a heuristic to select a trade-off between occupant's comfort level and energy price. The value α specifies a static risk attitude of residents to accept lower comfort (satisfaction) for a lower price (less money paid). However, setting the temperature range here describes a requirement and thus a hard constraint.

Domain and logic are modelled using algorithms, and best practices from coding such as the single-responsibility principle. Algorithms include loops, conditionals, and recursion, that are transferred to HTN planning constructs. Uncertainty is modelled only for the storage of solar power, by keeping track of the difference between estimated and real value. However, for an efficient estimator, this value should converge towards zero. Precisely, the cost is calculated as $cost(store\_solar\_o) = p_h \cdot min(E(X), F)$ where F represents the current forecast value and $E(X)$ is the expected value over the probability distribution of earlier forecasts [Hal22].

Later, Java Simple Hierarchical Ordered Planner 2 (JSHOP2) is used to implement the described domain. Reasons for that are deterministic method decomposition (allowing for if - else statements) and external function calls. An evaluation of the provided domain is done quantitatively, qualitatively, and through demonstrations. Stating, that planning times are kept low, showing an efficient implementation. Electricity cost could be reduced by almost two thirds proving a quality increase for the user. Nevertheless, it is argued that load shifting serves grid stability on a large scale, more than reducing prices locally. The work concludes by highlighting the relevance of occupants' behaviour (i.e., their willingness to decrease their comfort level) as well as physical building properties (e.g., thermal mass and insulation). Leaving the connection to electric vehicles, an implementation with another planner and commercial buildings as future topics.

3.4 Towards a Multi-Objective Planning Domain

Reviewed literature reveals that, on the one hand, there are great efforts to adapt the automation of buildings to the needs of occupants. On the other hand, there are various approaches to automatically construct optimal and explainable plans. Automated planning and optimisation techniques are already being used in the management of buildings. Therefore, the potential for improvement in energy demand through further optimisation of the planning process, while maintaining the same level of comfort, is small.

Most work, however, is concerned with the management and optimisation of smart homes. These usually have the characteristic that operator and occupant are represented by the same entity. This union makes it relatively easy to find a possible trade-off between comfort and cost. For non-residential buildings, however, such a simplification is rather difficult. Therefore, in this thesis we will address the illustrated limitations by designing and implementing a planning domain for the operation of smart non-residential buildings. Techniques of MO planning are employed to realise the separation of entities mentioned above. The uncertainty induced by planning the future, coupled with the unobservability of satisfaction, leads to a further concern with risk attitudes.

4 A Hierarchical Task Network Planning Domain for Smart Buildings

Now that we have seen different approaches in the field of building automation and automated planning, we take a look on how to approach the problem at hand. In order to design a domain for smart non-residential buildings, we must first identify the requirements of the system to be designed. We then try to fulfil these requirements with the means presented. To do this, we first look at typical scenarios we expect for the problem. We will then approach modelling the domain from three sides: First, we will try to reflect the physics of the environment by actions. In a second step, we then formulate occupant and operator requirements as high-level tasks. Finally, we look at how we can bring the tasks together with the actions and search for high quality plans efficiently. As augured, energy saving can only come from achieving occupant satisfaction more efficiently (i.e., more efficient actuators and controllers) or by finding and quantifying other trade-offs between energy usage and satisfaction.

4.1 Preliminary Considerations

What should be designed is a planning domain for non-residential buildings that generates a plan to manage a building efficiently the next day. This plan should consider the incentives of occupants and operators of the building, which makes it difficult to determine an optimal plan in the first place. The main purpose of the plan is to determine the hourly energy demand of the building upfront to obtain energy from a provider according to DAP. The actual execution of the actions is a bonus and could also be performed by another system. This corresponds to *organisational* planning according to [GA17]. In the process of modelling, we need to find a good trade-off between detail and abstraction for efficient planning. We set the smallest unit of time to be one hour, which determines the step size and, in combination with the horizon of the next day, creates the temporal setup of the system. This means, that the system provides a plan that can be divided into sections for each hour and provides the expected electricity demand for each of these sections. More precisely, the system can perform any number of actions at one hour. After that, time advances by one hour and the conditions in the building evolve according to the previously achieved state.

For the domain to represent the real environment as accurately as possible, actions should reflect pure physics. i.e., they should reflect the functionality range of actuators. Furthermore, discrete actions must be used, so that they can effectively be processed by a planning tool. For example, an action might involve turning a heater on or off, but it cannot tell to heat by a certain temperature difference. This is because the choice of the appropriate value would mean a further optimisation within the search space. Most discrete actions can, however, be mapped to a quasi-continuous operation by defining a duty cycle of operation. This would require a sufficiently fine temporal resolution. We assume the effect of actions to be deterministic, to simplify the simulation of future

states for further planning. We do not consider any of the usual sensors in the smart building environment, as they are intended for *inter*-action. However, since we are concerned with planning and not with execution, we consider only input values that are determined before planning takes place. We pursue state-based planning due to the time-dependent nature of the environment.

For goals or tasks, we have to consider the involved stakeholders. As examined in Section 3.1, a smart building should be aware of its occupants and try to satisfy their needs. However, non-residual buildings are often owned or operated by an entity different from its occupants. That being said, there generally are two opposing desires that the system has to consider. On the one hand, building operators aim to minimise their cost of operation, while occupants, want to maximise their comfort and satisfaction. As argued, this comfort usually comes at a high energy demand. The *ideal point heuristic* being maximum comfort without electrical demand, which is obviously not possible in a natural environment.

Since actions are deterministic, they lead to a certain degree of fulfilment of these desires. In the framework of planning algorithms, costs are only associated with actions, but not with being in certain states. Hence, the quality of plans must be computed differently. To represent the involved objectives faithfully without weighing them directly against each other, utilities must be processed separately. We choose this representation because it is difficult to link a monetary value to the well-being of individuals, which is what a comparison of the two desires would ultimately amount to.

The actual quality of a plan can be recognised only after its generation by the concept of Pareto dominance. To generate high quality plans, we define methods based on expert knowledge and common sense to narrow the search space. Furthermore, we only consider the planning of tasks that affect the building. Specifically, this means that we do not include the task of selecting an ideal energy provider. This task could also be solved in a separate optimisation process. We also do not consider different levels of automation. The system should rather fulfil desires according to expressed preferences and the agent's risk attitude, as a specific plan is generated before execution.

4.2 Summary of Acquired Knowledge

High level task can be derived from the systems mentioned by Clements-Croome [Cle20b]. We will look specifically at the following:

1. Room management - including HVAC and lightning,
2. Operation scheduling of appliances that do not have a fixed time slot,
3. Energy management - including various sources and storage of energy.

Where each task involves different kinds of resources and desires: In order to distinguish them from each other, we use the term *occupant* for the persons affected by room management, *user* for those affected by appliance scheduling and *operator* for those affected by energy management. Because of this terminology, we will speak of actions rather than operators in the context of HTN planning to avoid confusion. Therefore, planning operators represent only one action at a time.

For room management, the system generally transforms some electrical power into occupant comfort. Scheduling of operation is connected to user satisfaction and an electricity requirement at a certain point in time. Whereas in energy management, the system has to equalise energy demand and supply. This can be achieved by interaction with the electricity grid and other devices. Thus, we can summarise the involved resources to be:

- Occupant satisfaction,
- User satisfaction,
- Electrical power and
- Money.

From a higher perspective, the smart building is able to convert resources into one another by applying certain actions. This means that when the building operates, for example, a heating system, it converts electrical energy into occupant comfort. Drawing the necessary energy from the electricity grid in turn converts money into electricity.

Since actions are deterministic, the resulting state can be computed. So, when the heater is operated, it consumes a deterministic amount of energy. The purchase of this amount of energy in advance is therefore also deterministic and has a defined cost for the operator. Where the operator's incentive is to minimise his monetary costs incurred for the operation of the building. On the other hand, the system can neither know nor observe the satisfaction of occupants directly, so they cannot be deterministic. However, we have a model of their preferences and can thus define an expected utility function. Recall that Savage's theorem extends the idea of expected utilities to cases involving uncertainty of level 4 from Table 2.2.

In the domain, actions for occupant comfort and their effects are based on actuators mentioned by Lauckner and Krimmling [LK20]. However, we restrict ourselves to those actuators that allow discrete actions. We therefore consider the following actuators:

- Heaters,
- ACs to cool and dehumidify,
- Artificial light sources,
- Windows for ventilation,
- Blinds to shade from natural light.

While user preferences are determined by punctuality of service. Regarding occupant preferences and the resulting requirements, we refer to the claims of [BMS10]. Furthermore, we refer to the Kano model and the concept of flourishing presented by [Cle20a] to measure their fulfilment. We consider *temperature*, *humidity*, *air quality* and *light level* to be the main indicators of room climate and therefore quantify:

- Temperature in $°C$,
- Relative humidity in %,
- AQI,
- Illuminance in lux.

4.3 Expected Scenarios

For the problem at hand we expect the building of interest to be a school or an office (or any of the buildings mentioned in Section 2.1). The building features several rooms, all equipped with necessary actuators for occupant comfort. Each room can be occupied between certain hours that contribute to occupant satisfaction. We further assume to have a sufficiently accurate weather forecast for outdoor climate and solar irradiance. Users can specify a number of appliances and respective energy demand to be scheduled in a given time frame. The building can possibly be equipped with a local energy storage with a certain capacity and renewable sources with a certain power. The initial task is then to take care of room management for the hours of occupation and the scheduling of appliances. Further to allocate necessary or distribute remaining energy, in order to settle a deal with the energy provider upfront. Consideration of different problem instances in the same domain is supported by two separate files. The initial state s_0 and the task tn_0 are described in a *problem-file*, while the domain D from Definition 2.3.2 is described in a separate *domain-file*. The input data describing external conditions, i.e., weather data and electricity prices, could look as shown in Figure 4.1. An example of comfort ranges specified by occupants regarding the corresponding indoor quantities are shown in Figure 4.2. User specifications regarding the appliances to be scheduled are provided as an ordered list and are therefore not presented separately.

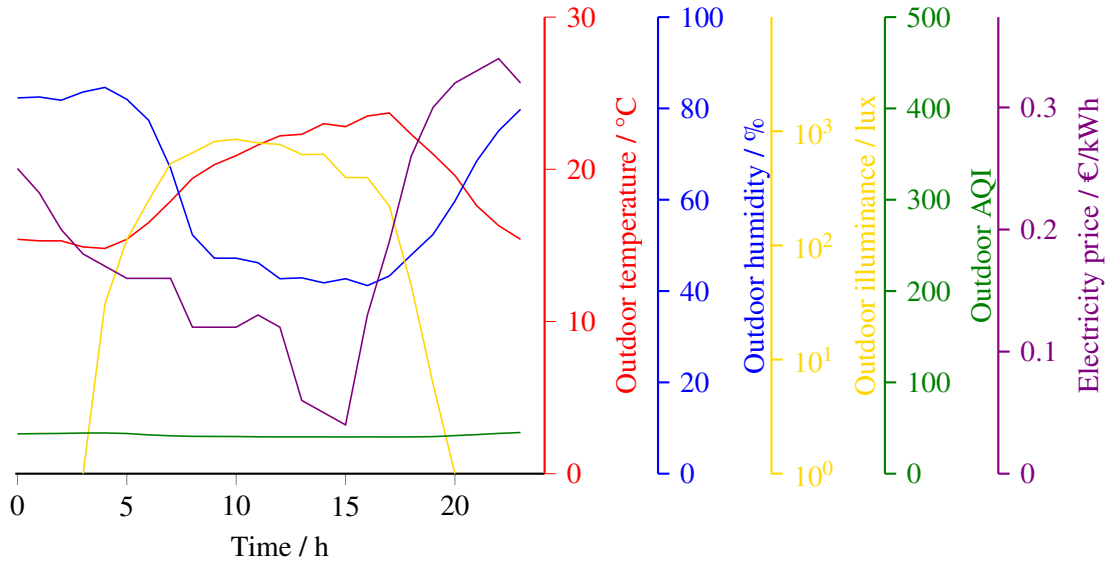


Figure 4.1: Exemplary input data

Where the x-axis indicates the course of time in hours and the y-axes various climate quantities. More precisely, the red curve shows the temperature in $°C$, the blue one the relative humidity in % and the yellow one the illuminance in *lux* (on a logarithmic scale). The green curve indicates air quality given by the AQI and the purple curve the energy price in €/kWh.

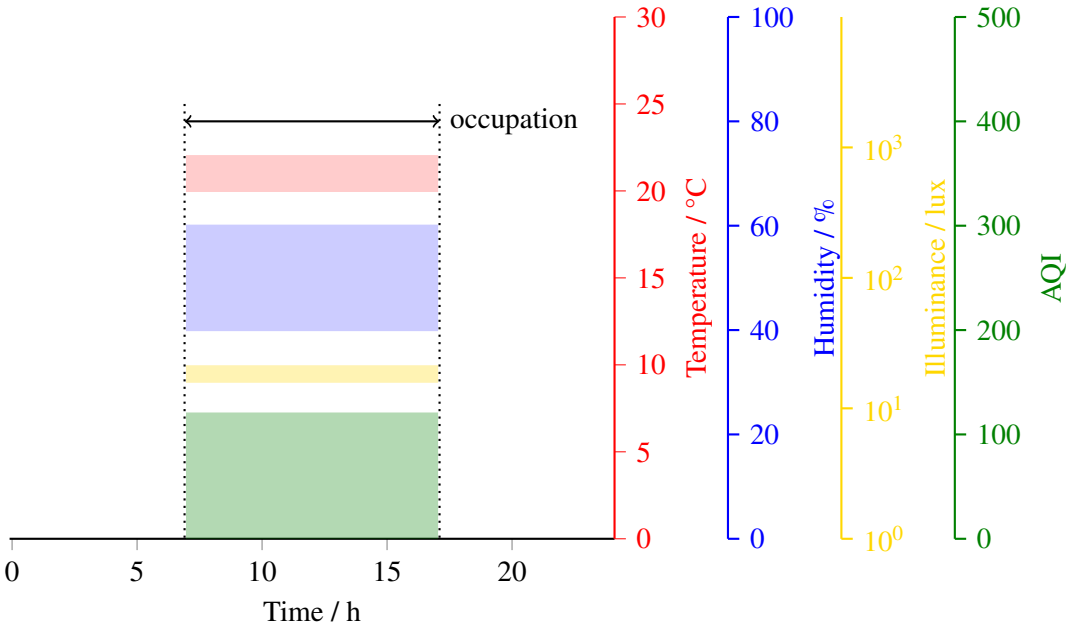


Figure 4.2: Indoor comfort preferences

Where the x-axis indicates the course of time in hours and the y-axes the preference range for indoor climate quantities. Again, the red curve shows the temperature in $°C$, the blue one the relative humidity in %, the yellow one the illuminance in lux and the green curve indicates air quality given by the AQI. A range is only specified during hours of occupation.

4.4 Domain Model

Now that we have defined the resources and basic properties of the environment, we need to decide how we can model it sufficiently accurately. We want to describe the basic characteristics of the presented model using the framework from [GA17].

4.4.1 Physical Properties

The physical properties of a domain define in what detail to depict spatial and temporal relations of the environment. Spatial relations are categorised by object and human locations [GA17]. We do not want to look at individual objects or humans in hour domain but restrict that considered rooms must be inside the same building, and humans reside in rooms during specified hours of occupation.

The temporal aspect is categorised by intervals and time points. Where both can *precede* and be the *same*, but two intervals can also *meet*, i.e., part of their range overlaps [GA17]. For the domain we want to incorporate both categories, by a single notation. That is, we only represent discrete points in time, but device operations for example can span from one point in time to another, depicting intervals. Our temporal resolution is defined to be one hour. With these properties, we can now go on to model the physic of actuators in the building domain. We divide the transfer to actions according to the three subsystems mentioned above.

Room Management

The room management model of a smart building should include HVAC and lightning. For this purpose, we utilise *behavioural inputs* in the form of *preferences* from [GA17]. That is, occupants may set a range for the four quantities of room climate as presented in Figure 4.2. The system then tries to arrange those quantities inside the preferred range on a soft constraint basis. In contrast to *requests* their satisfaction is not mandatory but missing them reduces occupant satisfaction. The degradation in occupant satisfaction should depend on how far a quantity actually differs from the preferred range. In that way, defining a narrow range, depicts a stronger preference by increased dissatisfaction.

The behavioural outputs of the domain are instructions to operate devices i.e., common actuators in smart buildings. To this end, some actions might change multiple quantities at once. For example, opening a window will in general increase air quality, but at the same time change temperature and humidity. In other words, while incentives for physical quantities come from human preferences about single properties, the functional range of an actuator determines the possible manipulation of multiple quantities. Table 4.1 gives a complete list of actions that we consider in the domain.

Action	Precondition	Effect
Open window	Window closed	Temperature, humidity, and air quality correspond to their quantity outdoors while open.
Close window	Window opened	Above-mentioned quantities are again decoupled from their respective outdoor value.
Operate heater	-	Temperature increases and humidity decreases by a certain value if window is closed.
Operate ac	-	Temperature and humidity decrease by a certain value if window is closed.
Close blinds	Blinds opened	Light level is a fraction of the light level outdoors.
Open blinds	Blinds closed	Light level equals the light level outdoors.
Operate light	-	Light level increases by a certain value.

Table 4.1: Sketch of actions contributing to room management

In addition to the effects mentioned above, all actions here also require a certain energy at the time of their execution. Besides these actions, there is always the possibility to leave everything as it is. Thus, at a certain hour the system can decide to choose any combination of those possibilities. The system can perform any of these actions in any order. However, some of the actions differ in their effects. While *operate heater* determines to operate the heater for the next hour, *open window* will change the state until it is changed again. Consequently, some actions have other - *conditional* - effects.

Here we encounter a limitation of the chosen approach to model the environment as planning domain: In the planning concepts presented in Section 2.3, the state always changes directly and only as a consequence of an action executed by the agent. The progression of time is basically irrelevant. However, we want to determine a set of actions and their energy requirements at one

point in time and then progress to the next hour. We can remedy this difference by introducing an additional action. This action must be carried out by the agent at the end of each hour and simulates the progression of time by one hour.

To be able to model the effect of time progression accordingly, we require conditional effects in the domain. Depending on the state of the windows, the temperature in rooms develops according to Equation (2.1). We model this using two time constants: One when the window is open and one when it is closed. Furthermore, the operation of a heater or the AC has no noticeable effect when windows are open. To complete the simulation, we need a weather forecast for the following day. Also, the air quality and humidity evolve in relation to their outdoor quantities when the windows are open. When the windows are closed, the relative humidity develops in relation to the temperature according to Equation (2.3). Air quality decreases in relation to the number of people in the room. The state of blinds determines whether the default indoor illuminance is equal to the outdoor illuminance or reduced by a certain factor. We summarise all these described conditional effects in the action *evolve time*. It has no preconditions and is executed as described when the agent has chosen all actions for this hour. The agent can thus also simulate what would happen before taking action and choose its actions accordingly.

We define a default state for all devices at the start of the day. This means, that windows are closed, blinds are open, lights, heating and ACs are off. Further note, that most actions have no precondition except that the actuator is in the opposite state beforehand. In other words, nothing physically prevents a device from operating, except if a room is not equipped with such a device, the device is broken or there is not enough energy to supply it. For managing rooms, we assume unlimited energy supply and fully functional rooms.

Appliance Scheduling

Apart from the devices needed to manage indoor climate, there may be appliances that do not have a fixed time of use, but a frame within which they should be operated. There are different kinds of appliances, however. The only thing that matters for their scheduling is, whether their operation has to be continuous or not. In other words, whether they are *preemptive* or *non-preemptive*. For example, a chipping machine that works through an ordered process should in general not be preempted during operation, while a fridge for example could be switched off for one hour without massive damage. We leave it up to the user to decide whether or not to allow preemption.

Equivalent to room management, we encode user preferences on the time slots in which appliances should be scheduled, as range. That is, the range of operation for an appliance to schedule is again a *behavioural input* in the form of *preferences*. The user defines an earliest and a latest hour between which the appliance should be scheduled, as well as its duration. The earliest hour is a hard limit, as there may be earlier steps in the operation of the appliance which could cause conflicts. Assume a bakery, where the start of a baking process should be scheduled but can only start after the oven was loaded. On the other hand, the appliance can run later than the latest specified hour, with the consequence that other processes may be delayed, and the user is dissatisfied. The only physical action of that field is thus depicted in Table 4.2.

Action	Precondition	Effect
Operate appliance at hour x	x ≥ earliest hour of operation	Appliance operates at hour x.

Table 4.2: Sketch of actions contributing to appliance scheduling

Here the precondition is not pure physics, i.e., the oven could of course also be operated without being loaded. However, the possibility represents such significant limitation for the process that we do not want to allow it. Furthermore, there are several ways in which the operation hours of an appliance can be determined. Conversely, it is also possible that a device will never be scheduled, which has to be prevented somehow.

Energy Management

The final task for the planning system is the management of energy sources and storage. We assume that a building has various sources and sinks of energy. Possible actions are intended to represent all physically reasonable paths of energy flow, which are shown in Figure 4.3.

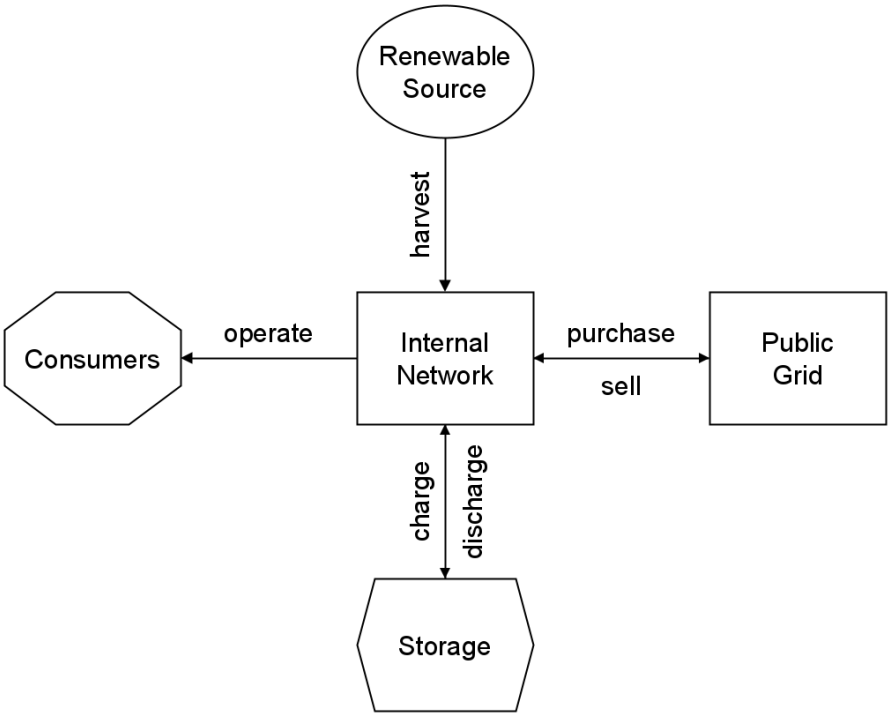


Figure 4.3: Interconnection of the electrical network model

Where shapes describe different energy sources and sinks and the directions of the arrows with respective arguments determine the direction of possible energy flows.

Unlike in other works, the internal network acts as a switch, connecting all the sources and sinks. Being connected to the public electricity grid, allows to purchase energy from the grid or feed it back. A local energy storage (i.e., a battery) has a capacity range, within which it can be charged and discharged. Whereas renewable energy sources like Photovoltaic (PV) or wind turbines can only deliver energy in a stochastic manner. We want to model this by the actions described in Table 4.3.

Action	Precondition	Effect
Purchase energy	-	Receive amount x of energy from the grid and pay the provider.
Sell energy	-	Deliver amount x of energy to the grid and get money from the provider.
Discharge storage	Amount x $\leq$ energy left	Get amount x of energy from the storage.
Charge storage	Amount x $\leq$ empty capacity left	Save amount x of energy to the storage.
Harvest renewable	-	Receive amount x of energy from the renewable source.

Table 4.3: Sketch of actions contributing to energy management

Notice that here charging and discharging is physically restricted by the available amount of energy. Whereas we assume electrical capacity of the grid and monetary resources of the operator to be unlimited. The amount of energy that can be harvested is also physically restricted, but uncertain at planning time and therefore has no precondition. To map this, the amount of energy that the agent would like to harvest is calculated. For further planning, however, we assume this amount to be deterministic. The fact that an estimate turns out to be wrong only affects the operator's costs, not the available amount of energy. In other words, planning electricity supply and demand in advance in order to make a deal with the energy provider. On the day of the operation, however, the amount may be different, and the operator pays for this with a possibly higher price.

4.4.2 Stakeholder Incentives

To evaluate actions as well as measuring their effectiveness, we have to look at intentions of the involved parties. Together with actions defined before, these then lead to a general structure of the domain.

Assigning Cost

In classical planning, costs are used to depict a loss in resource by performing an action. In the domain at hand, however, resources (e.g., satisfaction) are rather lost over time. The elapse of time is simulated by introducing an artificial action into the domain. In this way, the costs associated with the non-fulfilment of preferences can also be simulated. More precisely, previously identified, measurable quantities of the indoor climate used to measure costs. Results from [FSG+12] show that indoor climate can be interpreted as a basic requirement according to the Kano model or concept of Flourishing. However, the fact that they are measurable makes them also a performance

requirement. Derivation from occupants' preference range can be measured and thus depict expected (dis)satisfaction. Therefore, we restrict the expected occupant satisfaction to the performance requirements of the negative half-plane. To model this, we define that staying in the preferred comfort range has no costs. Whereas a climate quantity outside of its range incurs some cost depending on how far it lies outside. Doing so, penalises not satisfying occupant preferences, but keeps the notion of soft constraints. The described behaviour is given for temperature as

$$c_{occupant}(T) = \begin{cases} \frac{(T-T_{min})}{(T_{max}-T_{min})} & T < T_{min} \\ 0 & T_{min} \leq T \leq T_{max} \\ \frac{(Tmax-T)}{(T_{max}-T_{min})} & T_{max} < T \end{cases} . \tag{4.1}$$

Where T is the expected temperature that results from an action and T_{min}, T_{max} is the preferred temperature interval. To keep illuminance values in a comparable range, we use an abstract value L for comparison instead of illuminance E_v (given in lux). We define $L = e \cdot ln(E_v)$ with e being Euler's number and ln the natural logarithm. Equation (4.1) can be applied accordingly to the other three quantities of indoor climate. We want to describe the actions that influence the quantities of indoor comfort as uncertainty-inducing. Even if their execution does not generate costs per se, but only the evaluation of the resulting state does. This is where the uncertainty of the weather forecast, as well as the uncertainty about the actual satisfaction of occupants comes into play. More precisely, the comfort actions in the domain have a single deterministic effect on the indoor climate and energy consumption. However, since the system plans ahead and buys electricity for the next day based on the plan, they have fixed costs for the operator and variable costs for occupants. This is because the system plans and does not actually apply actions, therefore the actual indoor climate and its effect on occupant satisfaction cannot be observed.

In the same manner we can model cost in terms of appliance scheduling. We say that scheduling an appliance outside of its preferred operation hours is expected to dissatisfy the user. Whereas he is indifferent to operation times inside the provided range. That is, scheduling is a basic requirement, where the deviation from the preferred range defines the level of dissatisfaction, given as

$$c_{user}(h) = \begin{cases} 0 & h_{start} \leq h \leq h_{end} \\ \frac{(h_{end}-h)}{(h_{end}-h_{start})} & h_{end} < h \end{cases} . \tag{4.2}$$

Where h is the actual hour that the system schedules to operate and h_{start}, h_{end} define the preferred time interval in which the appliance should be scheduled. The *operate appliance* action is modelled in such a way that the system has full control over the appliances and does not take consider any failures or errors. It is therefore deterministic and results in deterministic costs for operators.

In terms of energy management, costs are directly linked to actions. However, it is still tricky, as we consider energy to be a means of exchange only. Thus, the possession of energy does not bring any value, but only purchasing or selling it contributes to the operator's utility. We assume the operator to have a monotonic utility of money, i.e., he values more money over less. However, we do not want the system to trade energy between days or gain money from intraday trading. We therefore calculate the costs through electrical energy as

$$c_{operator}(h) = p(h) \cdot (E_{purchased} - E_{sold}). \tag{4.3}$$

Where $p(h)$ is the price of energy at hour h and $E_{purchased}, E_{sold}$ is the amount of electricity purchased and sold to the grid, respectively.

We assume to have an accurate hourly electricity offer from one electricity provider. Hence, the system has a deterministic price at which it can purchase or sell energy for every hour of the following day. Without, however, saying anything about the price the operator pays in case of short-term changes. For simplicity, we assume that the price of buying and selling electricity is the same.

As the costs described represent different things, they cannot be compared with each other, but must be treated independently. We therefore express the utility over a state s by

$$\vec{u}(s) = \begin{pmatrix} U(c_{occupant}(s)) \\ U(c_{user}(h)) \\ U(c_{operator}(h)) \end{pmatrix} \in \mathbb{R}^3 \tag{4.4}$$

and thus the expected utility of an action by $\vec{eu}(a) \in \mathbb{R}^3$. This allows to compare the quality of two plans according to Definition 2.3.4.

We further assume operation with the grid and storage to be deterministic, while harvesting renewable energies is a stochastic process of time. For the sake of simplicity, we will limit the presentation to PV only. More precisely we model the amount of renewable energy to be normally distributed around a predicted value as suggested by [DSS19; PMZ21]. However, misjudging the amount of energy that can be harvested does not have a direct negative impact, but only through the fact that the system must compensate with another source of energy. This source is assumed to be the grid because the actual state of charge of a storage might be different during execution. As a result, misjudging the amount of renewable energy comes at the cost of eventually having to buy the remainder or selling the surplus. We therefore describe the *harvest renewable* action to be risk-inducing in terms of action cost for the operator.

For further planning, however, a fixed value of energy is required to continue the calculation. The action is therefore deterministic and has a single effect. Since the plan to be found should maximise the agent's utility, the amount of energy must be determined accordingly. Hence, the optimal amount to be harvested is determined by

$$E_{to\_harvest} = argmax(p_{renewable}(E, h) \cdot U(E)). \tag{4.5}$$

Where $p_{renewable}(E, h)$ is a probability distribution that is continuous in the energy estimate E and discrete in time. More precisely it is a collection of normal distributions for every hour h which have their mean at the predicted value $E_{predicted}$ and a standard deviation of $\frac{E_{predicted}}{3}$. Furthermore $U(E)$ is a utility function of that energy estimation.

The amount of energy whose harvesting maximises the expected utility of the agent can then be determined with the help of Equation (4.5). The scenario described is illustrated in Figure 4.4.

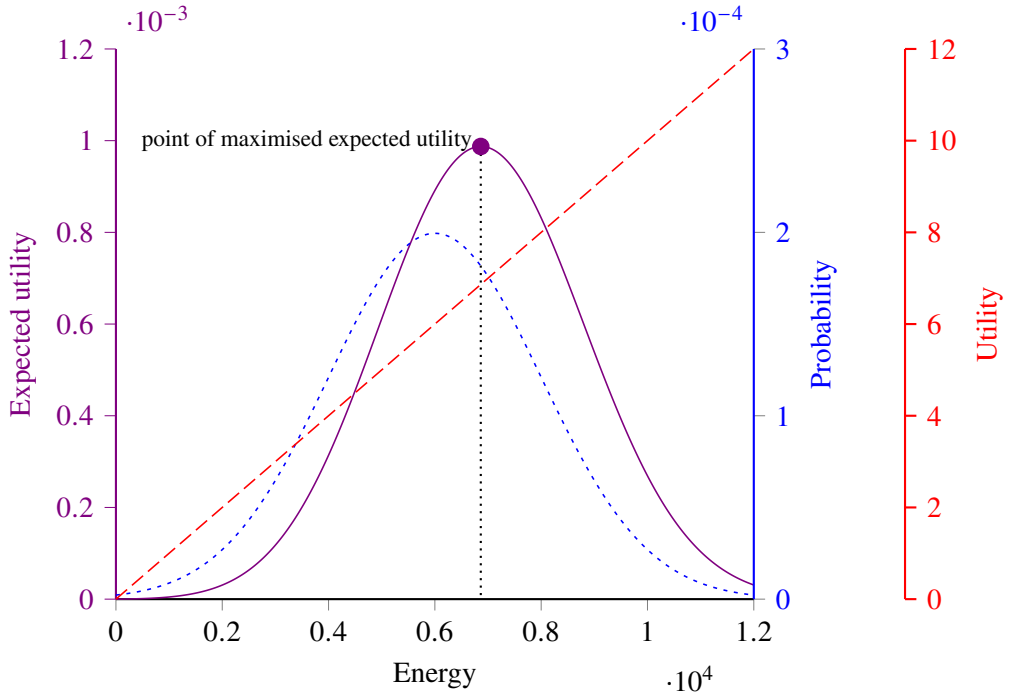


Figure 4.4: Expected utility of the amount of renewable energy to be harvested

Where the horizontal axis represents the amount of electrical energy harvested and the left vertical axis represents the associated expected utility. On the right vertical axes, the utility perceived from a certain energy amount is plotted on the one hand and the probability of generating this value on the other hand. The point marks the maximum of the product of these two.

The utility for a neutral agent increases linearly with the value of the energy harvested. Note that the risk-neutral agent would harvest a little more than the mean value of the prediction. This is because utility increases with the amount of energy and thus the maximum of the expected utility lies to the right of the mean value of the probability distribution. This influence was already highlighted in the study of the Optimizer's Curse [SW06]. Finally, charging or discharging the storage exhibits no cost at all, as the energy has already been paid for and can be used or sold later.

With the set of actions, defined above, it is possible to construct a search space and search for non-dominated plans. A domain of these actions would correspond to a classical planning domain and thus a blind search without guidance. This is however not desirable, as the resulting search space is quite large. Considering that the agent might choose any combination of the seven actions for room management alone makes 128 different states after one hour. Admittedly, not every combination of actions makes sense, but even if the agent could only choose one of seven actions per hour, there would still be 7^{24} possible states at the end of the day. However, we can use the methods of the HTN construct to systematically reduce this number.

Structuring the Problem

Let us start by assigning the actions to their respective resorts, i.e., breaking down the initial task of planning the smart building operation for the next day into its subtasks. In Section 4.1 we already defined the subtasks of building management to be:

- Room management,
- Operation scheduling and
- Energy management.

This means that the main task can be decomposed into the hierarchy from Figure 4.5.

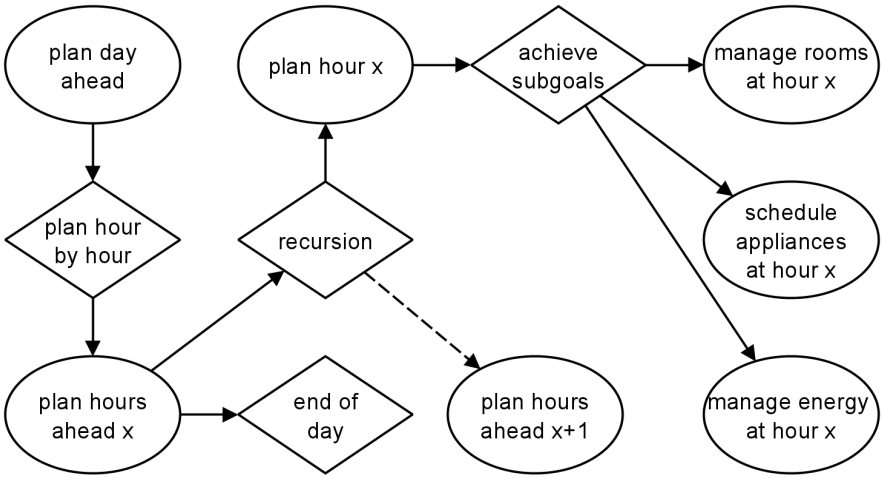


Figure 4.5: General hierarchy of tasks

Nodes are chosen according to Figure 2.5b and a dashed line indicates a recursion. The hierarchy includes the iteration over time i.e., 24 hours of a day.

The task of room management can further be divided into compliance with individual physical quantities such as temperature etc. This means that each room should be managed at all hours, each appliance should be given a time slot and the energy should be allocated and distributed accordingly. Furthermore, the order of tasks is important in that room management and appliance scheduling precede energy management. This is because the first two indicate demand of energy, while the last is responsible for its supply.

4.4.3 Guiding Through Methods

Let us see, how the three subtasks of building management can be decomposed given prior knowledge and how doing this can lead to high quality plans. The three subtasks shown represent a human solution to the problem, that is, a division into subsystems. In the following, however, we will look at how to get from these tasks to the actions needed to fulfil them. In other words, methods describe a way we already know how to accomplish the task.

Expert Knowledge

We generally have to structure the domain, by bringing together goals or preferences about quantities and actions that can affect these quantities. For example, keeping the temperature within a certain range can only be achieved by applying actions that effect temperature. Further attention must be paid to the direction of an effect, i.e., does the action reduce or increase temperature. In addition, we exclude the combination of actions, if their effects interfere each other. For example, operating heaters or ACs while the window is open, might in general not be a good idea, even though it is physically possible. In other words, there is an additional cost in terms of energy consumption, but no benefit in terms of occupant satisfaction. Similarly, it is generally cheaper to tap the outdoor temperature reservoir, as the energy demand for windows is lower than that for heating or cooling. Furthermore, it makes no difference to user satisfaction at what time an appliance is scheduled within the preferred range, but it does make a difference to energy costs. Since energy prices and appliance operations are deterministic, this knowledge can be leveraged to schedule an appliance within its preferred time frame. In terms of energy costs, it is always cheapest to draw the electricity from the storage, or to buy it at favourable hours and store it there. However, interaction with the storage unit is limited by its capacity. Harvesting renewable energies should always bring an advantage, as long as the generated values is not overestimated by far. So, for most actions, there is a fixed order in which to accomplish them. This knowledge of how to order actions is put into the decomposition of tasks through appropriate methods. Our model of energy management is shown in Figure 4.6.

The task of appliance scheduling can be broken down into two subtasks for each hour: Try to start the operation of appliances whose earliest hour is now and continue the operation of appliances that are non-preemptive and have already started. This task decomposition is shown in Figure 4.7. Note that the general order of subtasks already implements an Appliance First Schedule as described in [AW14].

Regarding the management of rooms we transfer the actions and the expert knowledge into the hierarchy depicted in Figure 4.8. Even though the management of different indoor quantities is equally important, we choose to implement an order. The reason for this is that actions are not equally powerful. Some actions affect several physical quantities, while others change only one. Some quantities can only be affected by a single action, while others can be manipulated in several ways. So, the system can arrive at the same action through different requirements, which is why actions must be self-exclusive. On the other hand, it does not make sense to trigger actions such as operating the heater or the AC if the window has already been opened. Therefore, we choose to decompose the *manage ventilation* task first, as AQI can never be too good, whereas if it is bad, the only way to improve it in this domain is opening a window.

The next step is to *manage humidity*. Here, again, if humidity is too low, the only option is to open a window, (provided that outdoor humidity is in the desired range) as the domain does not feature a humidifier. Whereas if humidity is too high, the system can either use the outdoor reservoir or operate heater or AC. For *manage temperature* the system has multiple options for every case, so only the order in which it tries to apply methods matters. What remains is *manage light*, where there is a possible action for each case, which does not interfere with other quantities.

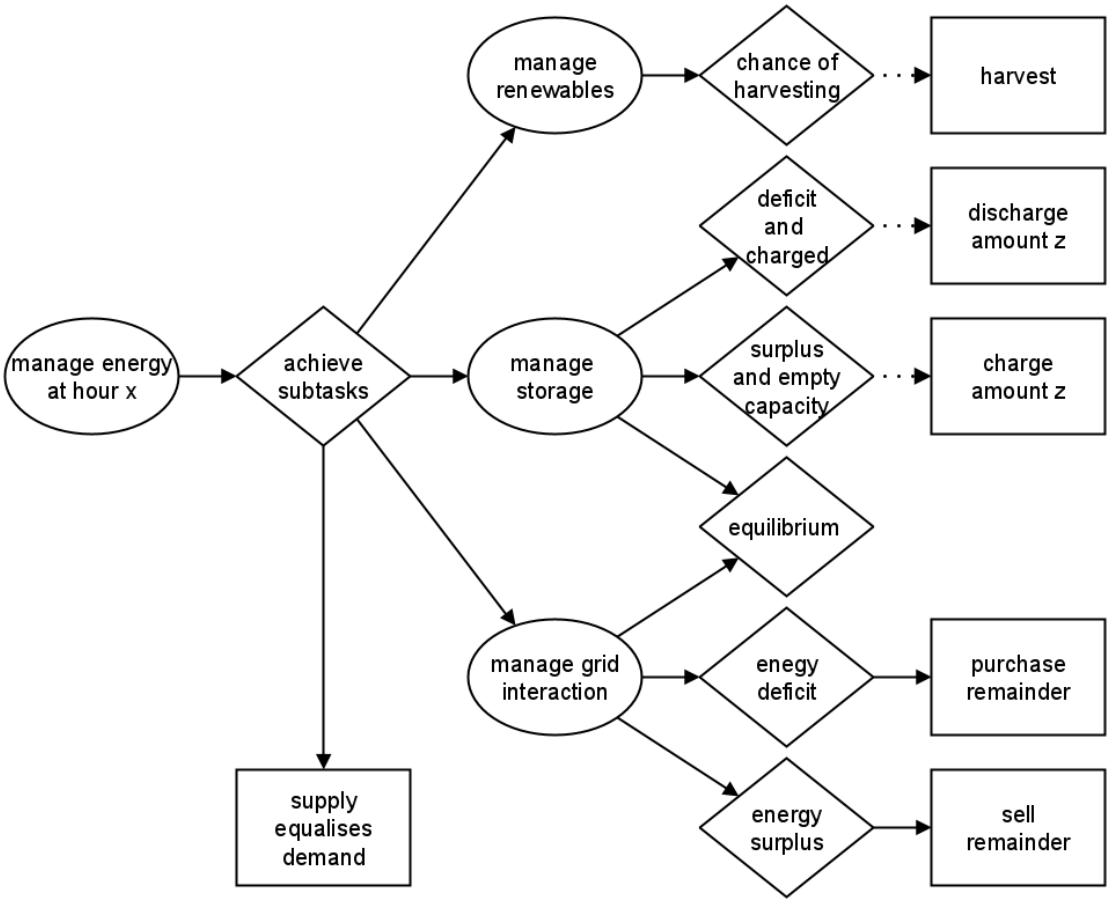


Figure 4.6: Hierarchy of energy management

Nodes are chosen according to Figure 2.5b. The three subtasks have to be decomposed in the given order from top to bottom to perform a self-consumption optimisation.

We have already argued that an additional action is needed to evaluate the resulting climate values based on actuator states. This action can also be used to sum up the cost arising from the resulting climate. However, it is imperative that this action is executed cyclically. That is why we need to anchor it at method level and make sure that it is evaluated every hour with the decomposition of room management.

The methods presented so far depict a reactive way to handle deviations from a desired comfort range. This narrows the search space to the point that actions are only taken, if a respective quantity exceeds its range. Hence, the size of the search space is conditionally dependent on outdoor climate. Furthermore, heater and AC are never operated at the same time, or when the window is open. For illumination, either the blinds are opened, the light is switched on or nothing is done. The various actions for room management are therefore reduced to 21 combinations per hour. Furthermore, one quantity should probably not exceed its range every hour, which further reduces the effective combinations. However, the system should act not only reactive, but proactively. Thus, there have to be more methods to achieve the described tasks.

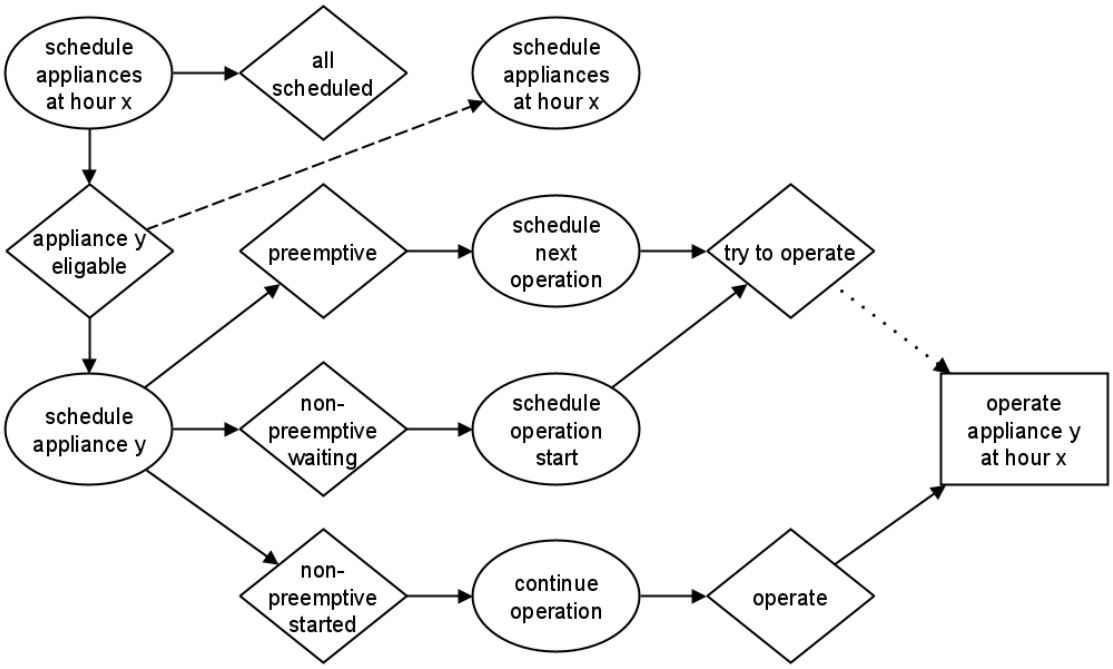


Figure 4.7: Hierarchy of appliance scheduling

Nodes are chosen according to Figure 2.5b. The dashed arrow indicates the recursion over all instances of appliances. A dotted arrow indicates that the primitive task can also be decomposed into taking no action, which is also indicated by the prefix *try*.

Cost Efficiency

As we have seen, the diversity of actions leads to a large search space. However, not all feasible plans will be non-dominated. We can further narrow this consideration by examining the dominance between subplans. We can get an estimate for the maximum size of the non-dominated set by looking at two tasks of the domain and assigning costs to their actions. In the task of appliance scheduling, there is only one action, namely *operate appliance*. This can only cause costs in the dimension of users, while costs for the operator only arise through the allocation of energy. However, we want to relax this separation for this analysis. We attribute both costs to the action *operate appliance*. The value for the operator costs is calculated using Equation (4.3). It would be possible to consider the surplus amount of renewable energy for the current hour as well, but not for later hours, where the demand is not yet determined. However, since the surplus energy is stored or sold and we assume the same costs for buying and selling, only considering the electricity price of the provider seems a reasonable assumption.

The attribution of costs allows to create a utility vector to *operate appliance* for each hour and appliance instance. Since the costs for users are zero for scheduling in the defined range, but the operator costs depend on the electricity price, one subset of non-dominated subplans can be determined directly from the operator costs. However, there may be times outside the range that cause lower operator costs. If one continues to assume that users are indifferent to times within the range, but earlier times are better in order not to jeopardise the process, one non-dominated plan can directly be determined. For times outside the schedule, however, there are several alternatives if

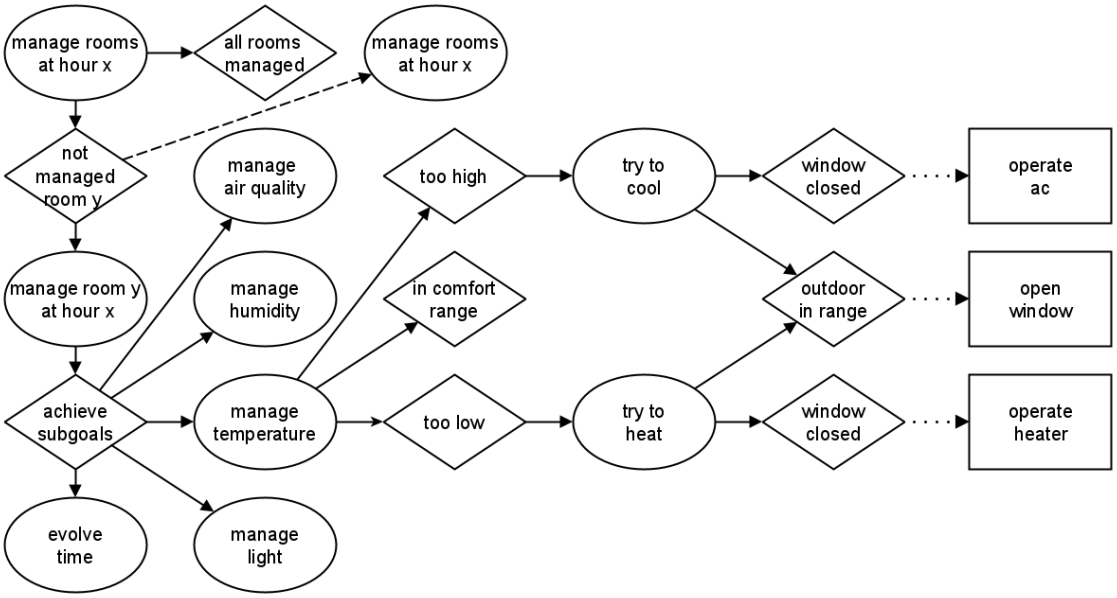


Figure 4.8: Hierarchy of room management

Nodes and vertices are chosen according to Figure 2.5b. For the sake of simplicity, only the methods for the *manage temperature* are depicted, but other subtasks can be decomposed in a similar fashion.

the electricity price is decreasing. This only applies to non-preemptive appliances. For preemptive ones, the operation can be reduced to one-hour blocks in the number of the total duration. For every block, there is one non-dominated subplan inside the range and n subplans outside that range. Sets of non-dominated subplans can be combined by *comax*, to determine an upper bound on the number of non-dominated plans.

For room management, we can take a similar approach by attributing occupant dissatisfaction and operator costs to the actions of indoor comfort. However, this is more difficult because these actions do not have a direct effect on $c_{occupant}$, but only through the *evolve time* action. If the environment did not evolve over time, the action could be arbitrarily moved forward to the point where it caused the least operator costs and thus dominated the other subplans. However, delaying the action causes unpredictable user costs and thus leads to a multitude of possible subplans. Hence, again, it is not possible to further narrow down the set of non-dominated subplans without comparing the utility vectors of generated plans directly.

In the two subsystems mentioned above, we attributed the energy costs to the comfort and scheduling actions and could hardly determine an upper bound on the number of non-dominated plans. In the model, however, the operator costs are actually incurred with the energy management task. Here, we consider the investments in renewable energy sources and storage unit as sunk costs. which is why energy drawn from these sources does not cause any further operator costs. Accordingly, from the operator's point of view, the storage would always be discharged, or energy drawn from renewable sources, as these do not lower his utility. However, the storage would also never be charged, as the possession of energy does not drive his utility. This illustrates very well the problem of an ideal point heuristic. Ideally, one would have no operator costs, but thus also no energy to operate any devices.

Of course, one could start to buy electricity at favourable hours and sell it at expensive hours, i.e., a Time-of-Use optimisation as described by [JKU+13]. However, this directly leads to two further optimisation problems: Firstly, how much and when energy is charged/discharged, and secondly, how large must the storage capacity ideally be? Since we do not want to deal with these issues in this work, we implement a self-consumption optimisation. In this case, the electricity demand is always fed from the renewable sources or the storage first. Any energy surplus is first stored in the storage unit until it is full and only then sold.

This seems to contradict the previously made assumptions about operator costs for scheduling and indoor comfort actions. Here, the energy costs should serve as a possible way to reduce the search space and thus simplify decisions during search. If both the other two cost dimensions and $c_{operator}$ are zero, there is always a manifold of plans that are indistinguishable. The energy costs should therefore rather be considered as a representative value of energy at the respective point in time. If one did not have a storage unit or reaches its capacity limits, this value is what would be incurred by the operator.

Making Proactive Decisions

As we have seen in the cost analysis of comfort actions: If time would not progress in the domain, it would make sense to advance actions until hours when electricity prices are cheaper. Even with the progression of time, this can still have a certain benefit. However, the effect tends to weaken over time. This is specifically related to the heat capacity and thermal conductivity of the building and is what we have described as its time constant τ.

The calculation of temperature set-points described in [AEA12] leads to a similar effect. A shift of comfort actions towards more favourable hours and thus a fluctuation of room temperature within a tolerance band. Here, however, occupant satisfaction is directly compared with energy prices. To reduce the operator costs in the model through proactive actions without causing additional occupant costs, we could proceed as follows: First, determine a schedule with any actions so that no occupant costs are incurred. Secondly, try to perform these actions earlier and reduce the operator costs without increasing the occupant costs. This is possible because most building types considered, such as offices, schools, etc., are occupied only part of the day. Hence, no costs are incurred outside the time of use. However, shifting actions retrospectively as described is difficult to implement in the designed HTN domain because of its sequential nature. Instead, the decision to take such action would have to be made at an earlier point in time, taking the future into account. In this way, however, we potentially increase the search space again by allowing actions to be taken in a time period in which they would not actually have been necessary. Whereas, due to the development over time, we cannot make any statement about whether an earlier action will lead to the same action being necessary later or not. The introduction of proactive actions should therefore be treated with caution.

For the implementation of pre-heating or pre-cooling strategies, we add another method in the field of room management, which has several preconditions. For example, a decision on possible actions can only be made in the period before the first use of a room. Furthermore, the energy costs should be below a certain threshold so that possibly arising operator costs are low. More precisely, the effect of an action must be considered in relation to the time of first room occupancy and its energy

costs. To heat a room ahead of time, only *operate heater* can be performed, while for cooling both *operate ac* and, on cold nights, *open window* are possible. We want to implement the possibility of proactive tempering using the method shown in Figure 4.9

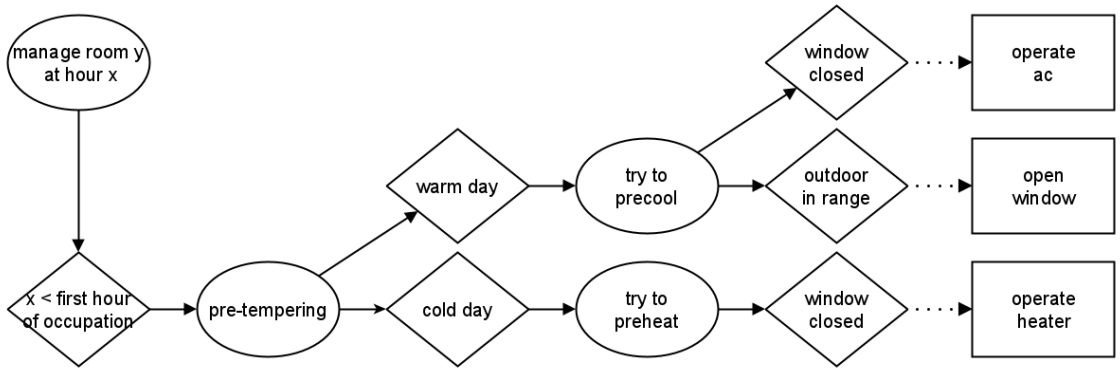


Figure 4.9: Additional task of pre-tempering

Nodes are chosen according to Figure 2.5b. This task replaces room management tasks in the hours before a room is occupied.

The way an appliance is scheduled also originates from the sequential structure of the HTN domain. That is, the connection drawn between the fulfilment of preferences about states and physically restricted actions. At an hour t when a decision must be made, a preview of the future is needed to limit the search space. For the operation of an appliance within its range, this is done by comparing the energy costs between this hour and the next. However, this comparison can only arrive at a local minimum of energy costs within the preferred time frame. By recursion over all subsequent points in time, one can determine the non-dominated set beforehand. However, this is unnecessarily complex to implement in HTN constructs. In contrast, a blind search without restriction to the best operation hours of appliances is also not desirable.

We therefore want to move towards determining the best times to operate an appliance in advance. This can be done outside the HTN domain in a separate optimisation procedure. In this way, the non-dominated subplans in appliance scheduling can be calculated before planning begins. The calculation of possible operation hours is determined by corresponding algorithms, one for preemptive tasks (Algorithm 4.2) and one for non-preemptive (Algorithm 4.1). Each algorithm determines a list of subplans that are non-dominated, which can then be consulted during planning.

By pre-calculating all non-dominated subplans of the appliance scheduling task, the effective size of the search room can be reduced. However, the size is mainly determined by the number of actions associated with the room management task. It will hardly reach its maximum in real problems, because of a buildings thermal mass. However, only the size of the worst case and not the nominal case can be estimated theoretically. Let us consider only actions that influence temperature directly and indirectly for a moment. Namely operating heater, AC or windows (which can be opened or closed for an hour). Add to this the alternative of doing nothing, results in four possibilities. Considering a day when the outside temperature is partly above and partly below the desired comfort range will not be an extreme case. The worst-case scenario will be an extremely hot or cold day. On such a day, all temperature range conditions will either be only *too hot* or *too cold* (cancelling either

Algorithm 4.1 Algorithm to determine starting hours for non-preemptive appliances

procedure GETHOURSTOSTART($this\_hour, duration, cost\_vector$)
  $h \leftarrow this\_hour$
  while $h \leq 24 - duration$ **do**
    $i \leftarrow h$
    while $i \leq duration$ **do**
      $EU(user, h) \leftarrow EU(user, h) + EU(c_{user}(h+i))$
      $EU(operator, h) \leftarrow EU(operator, h) + EU(c_{operator}(h+i))$
      $i \leftarrow i + 1$
      $h \leftarrow h + 1$
    end while
  end while
  $EU.filter\_nondominated$
  return EU
end procedure

Algorithm 4.2 Algorithm to determine hours of operation for preemptive appliances

procedure GETHOURSTOOPERATE($this\_hour, hours\_to\_operate\_left, cost\_vector$)
  $h \leftarrow this\_hour$
  while $h \leq 24 - hours\_to\_operate\_left$ **do**
    $EU(user, h) \leftarrow EU(c_{user}(h))$
    $EU(operator, h) \leftarrow EU(c_{operator}(h))$
    $h \leftarrow h + 1$
  end while
  $EU.filter\_nondominated$
  return EU
end procedure

operate AC or heater). Further assume no early temperature control action and that a building is only occupied between 8 am and 16 pm (common school or office). Then the number of feasible plans can be estimated to be smaller than $3^8 = 6.561$.

4.4.4 Incorporating Risk

We started off with the incentive to incorporate risk attitudes into the system design. So far, little attention has been paid to this fact in the domain. Recall, that a risk attitude models an agent's willingness to prefer disproportionate relations between outcome and probability. We have already described how uncertainty affects the expected utility of actions in case of one utility function, but also in the case of a utility vector in Section 2.3.4. We have identified the following sources of uncertainty in the domain: The deviation of the weather from its forecast and thus the amount of renewable energy that can actually be harvested, as well as the actual development of room climate depending on the outdoors. Furthermore, the actual satisfaction of occupants as a reaction to the applied plan. The first case provides a probability distribution, i.e., a risk over the amount

of renewable energy harvested. It also leads to uncertainty about the actual development of room climate depending on selected actions. The second case creates a risk in the resulting occupant costs, as the actual satisfaction is not observable.

However, since the costs incurred by the operator due to chosen actions are mostly deterministic, their expected utility is barely influenced by the agent's attitude. Only the amount of renewable energy harvested and any associated spontaneous changes in electricity consumption create uncertainty in the operator's costs. This leads to a shift in the Pareto front depending on the risk attitude, when considering the expected utility for determining the set of non-dominated plans.

The risk attitude of the agent influences the amount of energy harvested. Since the amount of energy is determined by Equation (4.5) based on the MEU for the agent, we need to modify it accordingly to be

$$\tilde{E}_{to\_harvest} = \text{argmax}(p_{renewable}(E, h) \cdot U(E, a, \alpha)). \tag{4.6}$$

Where the probability distributions $p_{renewable}(E, h)$ stay the same, but the utility $U(E, a, \alpha)$ now depends on the agent's attitude.

The result of this adjustment can be seen in Figure 4.10. The figure highlights the points of MEU for different risk attitudes.

4.5 Guiding Heuristics

In the course of this chapter, we have used the acquired knowledge to design an HTN planning domain that can lead to a very large set of feasible plans. However, since the plans incur costs in different dimensions that cannot be compared a priori, there is no unique plan of MEU. Still, some plans are superior to others, which is described by Pareto dominance. To minimise the computational effort of searching, the search algorithm looks for non-dominated plans specifically. For this purpose, we have already reduced the size of the search space by defining suitable methods. Alnazer et al. [AGA22] suggest to use a heuristic for state-based planning. However, the design of heuristics often requires that a variation of the problem has already been solved.

Several ways to design search heuristics for MO problems are presented by Geißer et al. [GHTT22]: The authors note that counting necessary actions as a distance to the goal is rarely profitable. An LP-based heuristics does not seem constructive, since some optimisations could then directly solved using the corresponding LP problem. An abstraction does not seem to be very helpful here, since described actions are already minimalistic. We therefore want design a heuristic based on a critical path. Here, a subset of propositions Γ of size m is defined, which is considered to be a critical path and whose compliance is therefore mandatory. A lower bound on the total cost is then using an action a that does not remove the critical subset from Γ. The problem is solved recursively, for various sizes of m, yielding an admissible heuristic for a set of states.

Let us apply this procedure to the domain as follows: For room management, the critical path represents keeping one indoor quantity at a certain level. For the task of energy management, the critical path is energy demand. Starting from a certain hour h onward, it is possible to generate all action state pairs, that solve a critical path. The heuristic can be built recursively by starting from the last hour and computing the maximum utility decrease in each dimension. The recursion

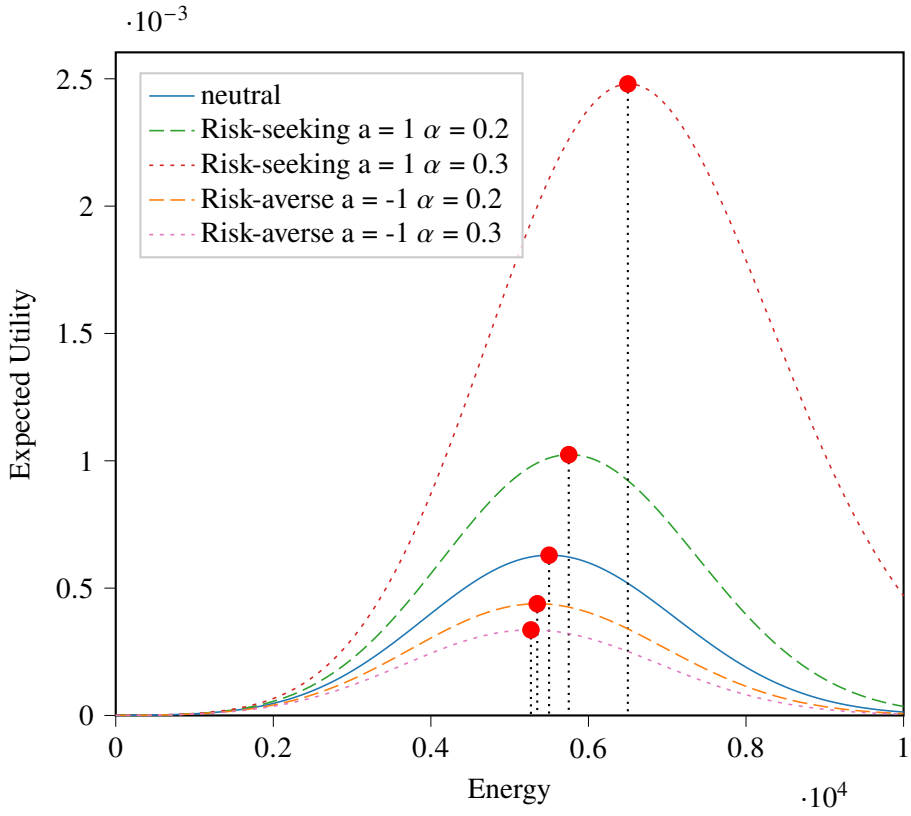


Figure 4.10: Dependence of energy amount harvested on the risk attitude

Where the x-axis again represents the amount of harvested energy and the y-axis the expected utility associated. The curves mark the expected utility of agents with different risk attitudes. The point on each curve marks the amount of energy that maximizes the expected utility of the respective agent and increases with the willingness to take risk.

moves on to the next earlier hour and uses minimum of these estimates as a starting point. This formulation poses a simpler version of the problem by assuming climate and energy were within range until a certain hour, thereby decreasing the number of necessary actions. However, it requires solving part of the original problem for every input scenario.

Finally let us review the described domain. From the stakeholder objectives, we derived three major tasks, which could be broken down into smaller tasks. On the other side we derived actions from the physical properties of actuators. Cost of states were derived and mapped to actions. Some actions are risk and some uncertainty inducing and thus exhibit attitude dependent utilities. These desires over states and possible manipulations of these are brought together by methods in the structure of the HTN model.

5 Realisation of the Domain

Let us now transfer the HTN model and considerations from Chapter 4 into a computable implementation. In order to do so, we have to decide on a planning software, before starting to explain methods and operators. Finally, we want to discuss the implementation and locate its limitations.

5.1 Choice of Planner

Table 5.1 presents recent and established HTN planning software and gives their main characteristics.

Planning Software	Characteristics
GTPyHop	A fusion of PyHop and Goal Decomposition Planner written in python. It supports GTNs and external functions can directly be implemented in the python domain. However, it does not support unification, so variables must be grounded or propagated manually. Problems are defined in the python domain itself and therefore do not follow any popular planning language nor a separation of domain and problem.
PANDA	Started as plan-based planning system, but now offers state-based solving as well. Uses HDDL as input format and supports various search heuristics. Features plan explanation and plan repair, which, however, are not available at the moment. No statement about the possibility of external functional calls.
JSHOP2	Java implementation of the SHOP2 planner. Therefore, support external function calls to Java classes. Own input language, that is very similar to HDDL. Uses DFS in the space of task networks with state update.
HyperTensionN (U)	Inspired by PyHop and JSHOP, written in Ruby. Supports various input formats such as HDDL and the one of JSHOP thanks to a parser. Solves problems by domain rewriting plus lifted DFS and supports external function calls. However, suffers from lazy variable evaluation and unordered execution of tasks.

Table 5.1: Comparison of HTN planning software

Let us recall the characteristics of the designed planning domain to compare planners with respect to them. Objectives are partly goal and partly task bound, these however, can be transformed into each other. External functions are needed to calculate the actual value of cost variable actions at planning

time. Utility-bound search heuristics can help to simplify the search for Pareto optimal solutions. None of the planners mentioned fulfils all these requirements. Nevertheless, we want to exclude GTPyHop, as it does not use a common planning language. In addition, PANDA does not seem to provide external function calls. HyperTensionN might be a powerful tool, but it differs significantly from other planning concepts by using strings, symbols, arrays, and hashes. We therefore decide to implement the described domain in JSHOP2. In addition to a language close to HDDL, this offers the advantage of being able to implement modifications through Java calls. Furthermore, the construct is well established and used frequently.

5.2 Problem Instances

In Section 4.3 we saw, what kind of scenarios we expect from the environment. Let us first look at how we can represent these input parameters for planning in JSHOP2. We only present the constructs that are relevant here and refer to [Ilg06] for a complete documentation.

5.2.1 Knowledge Representation in JSHOP2

The separation into problem and domain-file has already been discussed in Chapter 4. A problem-file contains the knowledge about the initial state s_0 encoded as list of *predicates* and *terms*, as well as the initial task network tn_0. Individual statements are separated by round brackets. A particular piece of information could thus be represented as in Listing 5.1.

Listing 5.1 JSHOP2 predicate syntax

```
(predicate ?term1 ?term2 ...)
```

Each statement is enclosed in parentheses, starting with the name of the predicate, followed by a list of terms. Where the value of a term can be a string or a number. Blank spaces are used to separate different elements within a statement.

There can be many predicates with the same name and different terms. These can then be used to reason about the knowledge and find a suitable substitution for term variables. To better illustrate this, some examples are given in Listing 5.2.

Listing 5.2 Example predicates for the designed domain

```
(room office)
(room classroom)
(window_closed office)
(room_temperature office 20)
```

Where the first two predicates show that there exist two rooms: *office* and *classroom*, that might have different properties. The latter two, for example, state that the *window* in room office is *closed* and the *temperature* is $20\,°C$.

With the predicates represented in Listing 5.2, the state of the system can be further explored. For example, one could search for a substitution that fulfils several criteria: On the one hand it is a room and on the other hand the windows are closed. This is possible as follows:

A term in the problem-file is considered true or *ground*. A call term is considered ground as a result of *binding* its variables to a value. With the statements shown so far, only *office* could correctly substitute *?room* in Listing 5.3. If, on the other hand, the statement *(window_closed office)* is removed from the knowledge base, there is no longer a valid substitution for *?room* satisfying both statements. A non-existent predicate is regarded as false. In the case of direct opposites, this can be made easier to read by an axiom as shown in Listing 5.6.

Listing 5.3 Search for a variable binding

```
(and
    (room ?room)
    (window_closed ?room)
)
```

A substitution for *?room* is searched, where both, the first *and* the second statement apply. Unless otherwise indicated, statements are linked with a logical and, here it is inserted for clarification only.

5.2.2 Representing a Building Environment

We now want to describe the relevant information about the environment of buildings discussed in Section 4.3 using the syntax of JSHOP2. To do this, we need to provide the following information about the input problem:

- Considered appliances,
- Characteristics of that appliances.
- Considered rooms of the building,
- Characteristics of that rooms and their devices.
- Forecast of the outdoor climate,
- Preferences about the indoor climate, and
- Initial state of the indoor climate.
- Hourly electricity prices of the energy provider,
- Risk attitude of the agent.

We want to do that using the predicates in Listing 5.4, which depict an exemplary subset of the information listed above.

Listing 5.4 Excerpt of a problem-file for the smart building domain

```
(appliance cleaning)
(appliance_preemptive cleaning)
(appliance_to_schedule cleaning 9 17 4)
(hours_to_operate_left cleaning 4)

(room office)
(window_closed office)
(blinds_opened office)
(heater_effect office 2)
(heater_energy_demand office 1000)

(outdoor_temperature_at_hour 12 20.9)
(outdoor_humidity_at_hour 12 45.6)
(outdoor_aqi_at_hour 12 41)
(outdoor_light_at_hour 12 35.2)

(room_temperature_limit_at_hour office 12 20 22)
(room_humidity_limit_at_hour office 12 40 60)
(room_light_limit_at_hour office 12 18 21)
(room_aqi_limit_at_hour office 12 0 150)

(room_temperature office 20)
(room_humidity office 50)
(room_light office 0)
(room_aqi office 50)

(grid_price_at_hour 12 0.56)
(renewable_predicted_at_hour 12 3647)

(risk_attitude 1 0.5)
```

The information is presented separately according to its subject and in the order of the listing above. The first four predicates describe an appliance named *cleaning* that should be scheduled between *9* and *17* o'clock and has a total duration of *4* hours, from which *4* are still left. The next five predicates describe a room named *office*, where the windows are closed, and the blinds are opened and the installed heater in the office has a power demand of *1000 W* per hour and heats up the room by *2 K* per hour when operated. The next four predicates specify the outdoor climate at *12* o'clock to be *20.9 °C*, *45.6 %*, *41 AQI* and *35.2 lux* respectively. The following four predicates define the current climate in *office* to be *20.9 °C*, *45.6 %*, *41 AQI* and light $L_{indoor} = 35.2$. The next two specify the electricity price at *12* o'clock to be *0.56 €/kWh* and the predicted mean of renewable energy to be harvested in that hour to be *3647 W*. The last predicate then determines the risk attitude of the agent to be risk seeking with $a = 1$ and $\alpha = 0.5$.

5.3 Transfer of the Domain

After seeing how we can represent knowledge about the system of interest, we look at how the system can reason and make decisions about it. To do this, we need to translate the methods and operators described in Section 4.4 into JSHOP2 constructs. We will first look at their syntax and then proceed in reverse order as in Chapter 4: First we will transfer the methods and then the operators. Note that we use *operator* here in its sense as HTN construct.

The initial task network tn_0 forms the link between problem and domain-file. It is placed at the end of the problem-file, as shown in Listing 5.5. The domain then defines methods that show how this task can be decomposed.

Listing 5.5 Initial task network

```
(:unordered
    (plan_day_ahead)
)
```

Where there is only one task *plan_day_ahead* to be decomposed. The keyword *unordered* specifies no order if there are multiple initial tasks.

5.3.1 JSHOP2 Domain Syntax

As mentioned before, JSHOP2 is implemented in the programming language Java. Therefore, solving a planning problem and thus the planner itself is implemented using Java functions. That leaves space to inspect and modify functions, as well as implementing own functions that can later be used inside the domain. However, the planning domain and planning problems itself are described using a modelling language close to HDDL. A Parser takes care of translating the domain into Java code, so that it can be executed. The JSHOP2 syntax for methods, operators and axioms is described in Listing 5.7, 5.8 and 5.6 respectively.

Listing 5.6 JSHOP2 axiom syntax

```
(:- (window_opened ?room)
   (not (window_closed ?room)))
```

Where a prefix with the axiom marker *:-* is used instead of an infix notation. So, the statement says that the predicate *window_opened* for a substitution *?room* is equivalent to the statement *window_closed* with the same substitution not being present.

Listing 5.7 JSHOP2 method syntax

```
(:method (compound_task ?term1 ?term2)
    method_brach_a  (
    (   (precondition ?term1)
        (precondition ?term2)
        ...
    )
    (   (subtask ?term1)
        (subtask ?term2)
        ...
    )

    method_brach_b
    ...
)
```

Where the depicted method can decompose the task *compound_task* for *?term1* and *?term2* into the two *subtasks* according to *method_brach_a* iff the variables satisfy the *preconditions* above. Otherwise, they are possibly decomposed according to *method_brach_b* or another method that specifies how to decompose *compound_task*.

Listing 5.8 JSHOP2 operator syntax

```
(:operator (!primitive_task ?term1 ?term2)
    ((precondition   ?term1) ...)
    ((delete ?term1) ...)
    ((add ?term2) ...)
    operator_cost
)
```

Where the depicted operator yields an action to *!primitive_task* with values for *?term1* and *?term2* iff the *precondition* is fulfilled. The action then removes the predicate *(delete ?term1)* from the knowledge base and adds *term2*, while adding *operator_cost* to the current plan cost. In case the preconditions are not fulfilled, the *!primitive_task* is possibly accomplished by another operator or the search algorithm has to backtrack.

Simple arithmetic operations like equality, inequality, summation and multiplication are depicted using calls as given by Listing 5.9.

Listing 5.9 JSHOP2 syntax for external function calls

```
    (call + ?term1 ?term2 ...)
```

Where a prefix notation is used instead on a common infix notation, so that the regarding function + can be applied to all elements of the following term list.

5.3.2 Methods

Let us now transform the methods designed in Chapter 4 into JSHOP2 constructs as described above. To keep this explanation as comprehensible as possible, we want to restrict ourselves to some exemplary parts of the problem. Let us start by transferring the general recursion to account for every hour of the day. When the respective task of a problem is *(plan_day_ahead)*, the system starts by decomposing this task and managing subsystems at every hour as depicted in Listing 5.10

Listing 5.10 Recursion to plan each hour of the next day

```
(:method (plan_day_ahead)
    one_hour_at_a_time
    ()
    ((plan_hours_ahead 1))
)
(:method (plan_hours_ahead ?hour)
    recursion
    (   (call < ?hour 24)) (
        (plan_hour_ahead ?hour)
        (plan_hours_ahead (call + ?hour 1))
    )
    end
    (   (call = ?hour 24)
    )
    ((plan_hour_ahead ?hour))
)
```

Where the initial task (plan_day_ahead) is decomposed into planning each hour of the next day recursively like using a for loop.

For room management let us look at the problem of keeping temperatures in a comfortable range reactively. To do so the system first has to localize what temperature would result from taking no action, by simulating via *(!!evolve_time)*. From here it can decide whether and what kind of action should be taken. Then for the task of heating, for example, there are multiple ways to achieve it. Consequently the methods describing how to heat are implemented as given by Listing 5.11.

Listing 5.11 Methods connected to heating

```
(:method (manage_temperature ?room ?hour)
    too_cold
    (   (room_temperature_prediction ?room ?temperature_without_action)
        (room_temperature_limit_at_hour ?room ?hour ?lower_limit ?upper_limit)
        (call < ?temperature_without_action ?lower_limit)
    )
    (   (try_to_heat ?room ?hour)
    )

    too_warm
    ...
)
(:method (try_to_heat ?room ?hour)
    use_window
    (   (window_closed ?room)
        (outdoor_temperature_in_range ?room ?hour)
    )
    (   (!open_window ?room ?hour)
    )
)
(:method (try_to_heat ?room ?hour)
    use_heater
    (    (window_closed ?room)
    )
    (   (!operate_heater ?room ?hour)
    )
)
```

Where the task (manage_temperature) can only be decomposed by one method, but *(try_to_heat)* can be accomplished in various ways. Good modelling style suggests to add layers to the hierarchy when encountering common preconditions. Here we, however, kept them together for compactness of presentation.

Listing 5.11 includes two application forms of methods: In the first (*manage_temperature*) there is only one method with several branches, while the second (*try_to_heat*) uses several methods, each with only one branch. The first variant represents a case distinction: One of the cases described must be true. Either the temperature is too high, too low or in range. On the other hand, heating can be performed in several ways or not at all if no possibility seems appropriate. Some ways can be ruled out directly because of their preconditions, while the choice between the others poses the essential task of optimisation.

For appliance scheduling we discussed two designs: The first one can be implemented according to temperature management. The second one can be implemented according to the harvesting of renewable energy. That is, we use an external function call, that determines the optimal hours to schedule and returns them. We will skip the details of these for reasons of similarity and refer to the implementations that have already been presented as well as the ones following. Instead, we want to discuss another peculiarity about the implementation in JSHOP2 that was not considered in Chapter 4, namely the order in which methods are selected for decomposition.

As can be seen in the example of the *try_to_heat* task, some tasks can be decomposed by several methods. The order in which this is done, could have a significant influence on the complexity of the search, if some subplans are already excluded at the time of task decomposition, because they are dominated by others. JSHOP2 translates the methods described in the domain-file into Java classes, according to their order of occurrence. It is the same order that they are selected for task decomposition. Finding a reasonable order can be used for an informed search.

Note that JSHOP2 itself does, however, not consider plan cost during search. Meaning, that generally a complete DFS in the space of task networks is performed. An intended early termination is only possible by giving a maximum number of feasible plans to be found. The absence of an early plan selection, however, allows to control the search by modifying the algorithm or inserting additional actions. Furthermore, external function calls are used to calculate the cost that result from a state after taking or not taking a certain action. This function implements the equations explained in Section 4.4.2. External function calls are also used to calculate the expected utility of costs according to Equation (2.6) considering the agent's risk attitude.

5.3.3 Operators

Actions from Chapter 4 can be transferred to JSHOP2 operators in a similar fashion. Good modelling style suggests using two leading exclamation marks to indicate primitive tasks without real-world representation, whereas real-world tasks have only one. Consequently the operators described in Section 4.4 are implemented in the domain as depicted in Listing 5.12, 5.13, 5.14 and 5.15. We only show some of the domain operators here, but most of them are implemented according to Listing 5.12. Contrary to ideas in Chapter 4, we decided to separate the calculation of the utility vector from simulating the elapse of time. A joint representation would have been possible, but it would require an unnecessarily high number of conditional statements. Instead, the elapse of time is simulated by the operator in Listing 5.13 and the resulting state is then evaluated by the operator in Listing 5.14.

Listing 5.12 Some operators of the domain

```
(:operator (!open_window ?room ?hour)
    (   (window_closed ?room)
        (window_energy_demand ?room ?window_energy_demand)
        (total_energy_demand ?demand_without)
    )
    (   (window_closed ?room)
        (total_energy_demand ?demand_without)
    )
    (   (total_energy_demand (call + ?demand_without ?window_energy_demand))
    )
    0
)
(:operator (!operate_heater ?room ?hour)
    (   (heater_energy_demand ?room ?heater_energy_demand)
        (total_energy_demand ?demand_without)
    )
    (   (total_energy_demand ?demand_without)
    )
    (   (heater_operated ?room)
        (total_energy_demand (call + ?demand_without ?heater_energy_demand))
    )
    0
)
```

Where the first operator describes preconditions and effects of opening a window, while the second does the same for operating the heater in a room. Both actions have action cost 0, as the occupant satisfaction is evaluated with the resulting state and operator cost are derived depending on the actual energy sourcing.

Listing 5.13 Operator to determine the resulting state due to time progression

```
(:operator (!!evolve_time ?room ?hour)
    (   ; get relevant variables
        (room_aqi ?room ?aqi_now)
        (outdoor_aqi_at_hour ?hour ?aqi_outdoors)
        ...
    )
    (   ; remove old values
        (forall (?r) ((room ?room)) (
            (room_aqi ?room ?aqi_now)
            ...
            )
        )
    )
    (   ; set new values conditionally to current actuator positions
        (forall (?r) ((window_opened ?room)) (
            (room_aqi ?room ?aqi_outdoors)
            ...
        )
        ...
    )
    0
)
```

The above operator describes the evolution of time in a room, after all comfort actions for that hour have been chosen. The new values of variables are calculated depending on the resulting state of actuators. To make the presentation easier to read, only an excerpt is shown.

Listing 5.14 Operator to calculate a utility component

```
(:operator (!!calculate_utility ?room ?hour)
    (   ; get variable values
        (room_aqi ?room ?aqi)
        (room_aqi_limit_at_hour ?room ?hour ?aqi_min ?aqi_max)
        (risk_attitude ?a ?alpha)
    )
    (   ; remove old utility value
        (occupant_utility ?utility_before)
    )
    (   ; calculate expected occupant utility and add it accordingly
        (occupant_utility
            (call + ?utility_before
                (call CalcRiskAwareUtility
                    (call +
                        (call CalcCostOccupant ?aqi ?aqi_min ?aqi_max)
                        ...
                    )
                ?a ?alpha)
            )
        )
    )
    0
)
```

Where above operator calculates the expected utility for occupants depending on the resulting state and adds it to what the value was before. To make the presentation easier to read, only an excerpt is shown. Two function calls are used: *CalcRiskAwareUtility* and *CalcCostOccupant*, which implement the corresponding equations from Equation (2.6) and Section 4.4.2.

Listing 5.15 Operator to harvest renewable energy

```
(:operator (!harvest_renewable ?hour)
    (   (energy_supply ?supply_before)
        (renewable_available_at_hour ?hour ?predicted)
        (grid_price_at_hour ?hour ?price)
        (grid_price_range ?min ?max)
        (operator_utility ?utility_before)
        (risk_attitude ?a ?alpha)
    )
    (   (energy_supply ?supply_before)
        (operator_utility ?utility_before)
    )
    (   (energy_supply
            (call +
                ?supply_before
                (call CalcAmountToHarvest ?predicted ?price ?min ?max ?a ?alpha)
            )
        )
        (operator_utility
            (call +
                ?utility_before
                (call CalcUtilityToHarvest ?predicted ?price ?min ?max ?a ?alpha)
            )
        )
    )
    0
)
```

Where the operator essentially calls two external functions: The first is *CalcAmountToHarvest*, which calculates the amount of renewable energy that maximizes the agent's utility function. The second is *CalcUtilityToHarvest*, which calculates the expected cost resulting from that amount to be harvested.

Multiple function calls are required, as there is no way to assign a value directly to a term or to return several values with one function call in JSHOP2. The two functions essentially perform the same calculation: They create the expected utility as product of probability distribution (using $E_{predicted}$) and utility (using a and α, compare Equation (2.6)). Then the point of MEU is determined, by iterating over this distribution. Finally, returning the associated energy amount (Equation (4.6)) on the one hand and the accumulated expected utility on the other. The implementation of the function call as a Java class is shown in Listing 5.16 for the function *CalcAmountToHarvest*.

Listing 5.16 Function to determine the amount of renewable energy

```
import JSHOP2.*;
import java.io.*;
import org.apache.commons.math3.distribution.NormalDistribution;

public class CalcAmountToHarvest implements Calculate{

    public CalcAmountToHarvest(){}

    @Override
    public Term call(List l){
        amount_to_harvest = 0;
        double e_predicted =((TermNumber)l.getHead()).getNumber();
        l = l.getRest();
        double price_at_hour =((TermNumber)l.getHead()).getNumber();
        l = l.getRest();
        double a =((TermNumber)l.getHead()).getNumber();
        l = l.getRest();
        double alpha =((TermNumber)l.getHead()).getNumber();

        if (e_predicted == 0) return new TermNumber(amount_to_harvest);

        NormalDistribution norm = new NormalDistribution(e_predicted, e_predicted/3);
        double maximum_expected_utility = 0, utility = 0, expected_utility = 0;
        int start = 0;
        int stop = (int) e_predicted * 2;
        int i = start;
        while (i < stop){
            utility = price_at_hour * i * 0.001;
            if (a != 0) {
                utility = (Math.exp(a * alpha * utility) - 1) * a / alpha;
            }
            expected_utility = utility * norm.density(i);
            if (expected_utility > maximum_expected_utility){
                maximum_expected_utility = expected_utility;
                amount_to_harvest = i;
            }
            i++;
        }
        return new TermNumber(amount_to_harvest);
    }
}
```

A Java class that implements the *Calculate* interface and overrides the *call* function is required, according to the JSHOP2 documentation. In the depicted function, the parameters are first extracted from the term list, before checking whether there is any chance for renewable energy. Then the expected utility is maximised by iterating over the amount of energy. The value $a = 0$ is used to encode a neutral attitude. Finally, the value is returned as a JSHOP2 term.

5.3.4 Generation of Plans

During planning, the JSHOP2 algorithm takes the current list of tasks and expands it in the given order. Thereby applying methods in the order, they are given in the domain-file. JSHOP2 implements a DFS for each task until it finds operators that can be applied. Depending on an input parameter provided either generating only the first feasible plan or a number of plans. Listing 5.17 lists an example of some actions chosen - an excerpt of a resulting plan.

Listing 5.17 Excerpt of a generated plan

```
(!open_window office 12)
(!operate_appliance cleaning 12)
(!harvest_renewable 12)
(!charge_battery 12 723)
(!sell_energy_to_grid 12 992)
```

Where the list above specifies actions to be performed at *12* o'clock plus specific term substitutions to form a plan.

As emphasised in Section 2.3.2, DFS is an uninformed search algorithm that explores the entire search space. JSHOP2 offers the possibility to terminate search early by defining an upper limit of feasible plans to be generated. In this case, however, not all plans are known, which makes it impossible to guarantee that all non-dominated plans are found. We do not want to change the search algorithm of JSHOP2 in the scope of this work, although this an interesting research topic. The implementation of a heuristic as presented in Section 4.5 is not possible, as it would require either an informed search algorithm or a dynamic selection of methods as presented by [Aln19]. However, considerations and observations made can still benefit the optimisation of the search algorithm. This can be achieved by modifying the domain itself in such way, that further exploration of a plan is terminated, when it can no longer lead to a non-dominated plan. This approach was introduced in Section 2.3.2 as pruning.

From considerations in Section 4.4.2, it can be said that occupant utility can only decrease as a result of an action (or rather the omission of an action). On the other hand, building operator utility can increase and decrease as well. In particular, it increases when actions are not taken and there is a surplus of energy, while it decreases when energy has to be purchased. However, let us assume for a moment that both utilities can only decrease. Specifically, this means that selling energy does not increase the operator's utility, and a utility vector consisting of the two can only remain in the third quadrant. Since planning is performed in a sequential order and the expected utility changes with the progression of time or with the choice of an energy source at each hour, it is possible to check whether a plan can still belong to the non-dominated set after each of these changes. This is especially possible before a plan has evolved completely since that plan can only decrease in both expected utility values as planning proceeds. All plans start with $\vec{eu} = \vec{0}$. Due to the nature of the Pareto front, a plan can never become non-dominated again by taking any action, once it has crossed the Pareto front. Thus, if some non-dominated plans have already been found, subplans can be filtered during planning by comparing them with the non-dominated set.

To conduct an early sorting of plans during the search, it is necessary to:

1. Keep and update a list of non-dominated plans already found.

2. Always check the location the current subplan with respect to the Pareto front during generation and discontinue a path if necessary.

Both functions can be realised by an external function call. To do so, an additional task *!!filter_non_dominated* is introduced to the domain, which must be decomposed at the end of each hour. The precondition of the corresponding operator contains a function call that evaluates to true if the plan is still non-dominated and false if not. In this way, further pursuit of subplans that are no longer non-dominated can be avoided at an early stage. This is because a false precondition of an action will cause JSHOP2's DFS to backtrack. If a path taken during search leads to the plan being dominated by a plan that has already been found, another path must be selected until the plan is no longer dominated or no other plan can be completed. This requires the external function to be called with two parameters: For one, the utility vector of current subplans must be compared with those of the non-dominated plans already found, returning a corresponding Boolean value. Secondly, for a complete plan, that is still non-dominated at the end of the day, it must also be added to the list of known non-dominated plans. In the process of exploring all plans, it could happen that a plan from the list of non-dominated plans is dominated by one found later. The list could therefore be constantly updated and checked for such instances. However, this is not necessary because dominance is a transitive order. The probability that already found plans are dominated by their successors can, however, be influenced by the order of methods in the domain.

To order methods accordingly, we come back to the separation of costs or utilities caused, discussed above. Recall that the expected occupant utility decreases mainly by the omission of comfort actions, while the expected operator utility is decreased according to the energy demand of those actions. In general, the Pareto front can be explored from the point of maximum occupant utility as well as from the point of maximum operator utility. We can influence the direction by ranking methods according to their energy consumption for the task of room management or according to their energy costs in the task of energy management. However, the chosen strategy of self-consumption optimisation already defines an order here.

In the assessment made so far, we have neglected the fact that expected operator utility can also increase as a result of selling energy. This is critical in that it makes it difficult to identify a subplan beyond doubt as non-dominated within the course of planning. To be able to judge at an early stage, it is necessary to estimate the maximum amount by which the operator's utility can increase from this hour onwards. More precisely, an overestimation of the amount is necessary in order not to discard a non-dominated plan early. However, overestimating too much leads to too many plans being pursued until the end, where their location with respect to the Pareto front can be determined without doubt. For the best possible search, therefore, the cumulative value of the maximum possible increase in operator utility at a certain hour is necessary. However, this depends on the amount of energy to be harvested and thus on the agent's risk attitude. The estimation therefore requires a preliminary calculation in another external function or a plan generation in advance. Therefore, we order the comfort actions so that the first plan found is the one that maximises the expected operator utility. This plan can be distinguished by the external function that filters out non-dominated plans and stored in class variables for later comparison.

5.4 Discussion of the Implementation

During implementation and design of the model we had to make assumptions about the environment and modifications to the planning software. Here we want to discuss the character of the resulting implementation and take a look at possible alternatives.

5.4.1 Level of Abstraction

We started off by mapping the environment of non-residential buildings to a corresponding planning domain. Therefore, we first abstracted from the actual building by considering only certain rooms in the building. Further, we divided the time into discrete steps of one hour each and specified that the agent can take decisions between each of these hours. We restricted occupant satisfaction to be quantified by four measurable, physical properties. Most of the assumptions are related to the time dependence of these quantities from the outdoor environment, of which we only consider certain effects. Namely, the dependence of indoor temperature from outdoor temperature as a result of the conductivity of walls, the dependence of relative humidity on the temperature as well as the effect of solar radiation on the renewable energy generation and the indoor lighting. In doing so, we have neglected and simplified some effects, such as heat transfer by radiation, the segmentation of a rooms' heat capacities, and thus the exact temperature change by air convection. Furthermore, for the illuminance of rooms we have neglected any geometric arrangement and areas of windows and have not considered reflections and scattering or absorption. Instead, we have equated the indoor illumination with the outdoor illumination and limited the effect of blinds to a dull division. For the ventilation of rooms, we have moved away from a prescribed volumetric flow per hour and towards ventilation via window operation, depending on the AQI. In the process, the consumption of oxygen and the degradation of air quality were assumed to linearly increase due to people occupying a room. The measures taken result on the one hand from the restriction to discrete actions in planning domains, and on the other hand from the computational complexity that results from a co-dependency of actions and their effects. For the scheduling of appliances, we consider neither faults nor failures of devices, to keep a deterministic representation. For the EMS we allow any reasonable path of energy flow, even if this is not given in every building. Furthermore, we only consider PV plants to be able to model their energy output by a normal distribution.

All these abstractions result in an already complex HTN model, which takes an essential co-dependency into account. A further simplification of the domain would only be possible by neglecting any external effects or by not considering time itself. However, the relevance of generated plans would then also decline considerably. Enhancing model complexity is possible in any way: The interaction of building and environment could be described more precisely. The actual occupation of rooms could be considered in more detail. It could, for example, be specified with a certain probability via human presence sensors. This probability could then be used as an input distribution to occupants expected utility. For the scheduling of appliances, it would be possible to add dependencies between different appliances, so that one appliance may only begin after the completion of another. For energy management, it would be possible to consider losses of devices and the storage unit. The model's level of detail is limited primarily by the unnecessary complexity of encoding information. We therefore believe that, with the present model, we have found a good compromise between abstraction and attention to detail, which makes it possible to map the continuously changing environment of buildings onto a discrete planning domain.

5.4.2 Modifications and Tweaks

Some of the abstractions described above originate from two essential properties of planning domains: Firstly, their discrete character and secondly, that they do not consider the influence of external effects and thus the effect of time itself. These circumstances have made it necessary to apply some tweaks in order to transfer the described problem to a planning domain. This includes the introduction of supplementary actions that do not correspond to an actuator in the real environment in order to be able to simulate the progression of time and the associated influences on the environment. Related to this is the fact that some costs are caused by states rather than the execution of actions, which is specifically related to the implementation as an HTN domain. Leading to the fact that actions of the agent represent discrete actuator articulation. As shown in the consideration about windows and ventilation systems: A different implementation would of course be possible, but this rather leads to a numerical optimisation about the actually supplied air flow. Such optimisation is better described by a convex optimisation problem or similar. We partially avoid this problem by outsourcing it to external function calls, as it is done with the determination of the amount of energy to be harvested or the calculation of appliance operation hours. The possibility of implementing such calls and the labelling of tasks without real representation highlight the versatility of the planning software used. At the same time, it highlights, that approaches to model real world problems still have their limitations and do not have all the means to describe them completely. The absence of sophisticated data types also emphasises that JSHOP2 is not intended for problems like the one at hand.

The consideration of discrete actions, however, reveals another critical point, namely the co-dependency of actions on each other. A window operation not only influences ventilation, but also temperature and humidity. A heater or AC influences not only temperature but also humidity. In particular, all actions influence both occupants and operators. Of course, it would have been possible to neglect these effects and consider all tasks and actions isolated. But this misses the point of the optimisation. By the same reasoning, it would have made sense to optimise each of the three major subtasks in a separate iteration over the hours of a day. This would simplify further approaches to optimise energy sourcing. However, the separation of the optimisations fails to reflect the mutuality of the problem and to find a good trade-off between objectives. Such formulation yields only one solution: Namely that of maximum occupant comfort and consequently an allocation of necessary electricity. This, in turn, leads to a cost optimisation regarding the size of the PV system and storage, or mixes up cost of different dimensions. Therefore, a fair solution can only lie in a combined consideration of the problem as presented here. Of course, it would also be possible to include corresponding optimisations regarding the allocation of electricity in the described domain. However, this would have gone beyond the scope of this study. In the course of planning within the domain, it becomes clear that when there are co-dependencies, a simulation of the resulting states is crucial. Whether it is about the necessity of comfort actions or the determination of non-dominated plans.

Another essential character of a deterministic planning problem is the state-based nature of the presented domain, which also originates from the fact that some costs depend on states rather than actions. This, however, complicates the application of advanced search algorithms. It makes it difficult to accelerate the search in the presented problem, because the ideal point lies in a separate consideration of problems. By analysing possible alternative heuristics, further limits about the abstractness of the model are revealed. This circumstance could not be solved by choosing a different planning software, since to our knowledge no planning software exists that allows the

description of the problem at hand out-of-the-box. Modelling the presented problem as MDP would have simplified the representation of time progression and resulting effects for the system. However, MDPs lack the expressiveness of planning problems and, most importantly, solving them, would not reveal a direct advice for buying electricity form a provider. But that was the problem that had to be solved in the first place.

6 Evaluation of the Solution

In the last two chapters we presented an HTN model for the operation of smart buildings. We discussed the characteristic of the presented solution, as well as its limitations. Now we want to investigate how the implementation actually solves problem instances in order to evaluate its performance. This evaluation should include the computational complexity, as well as the quality of generated plans.

6.1 Approach

To examine the performance of the presented planner, we want to use *demonstrations*, *quantitative evaluation* and *qualitative evaluation* as given by [GA17]. For better clarity and readability, we will only discuss two aspects of the main task here, namely room and energy management. This reduces the dimensionality of the utility vector to two components and makes it better to visualise. In addition, the scheduling of appliances in the presented domain is purely deterministic and therefore better optimised by other means, as shown. A qualitative evaluation of the planning results and the domain is performed by several *scenarios* on room management and associated energy allocation. The same scenarios are used to investigate the *computational factors* and *scalability* of the planner, but in a quantitative way.

6.1.1 Metrics

Computational complexity - and thus scalability - is usually defined by terms of time and space complexity. On the one hand, this can be determined theoretically for a particular algorithm as a function of input size. On the other hand, computation time and memory requirements can be determined by practical experiments. Since we have not designed a planning algorithm here, the theoretical analysis can be traced back to the DFS used by JSHOP2. We already looked at branching factor b and maximum depth d (compare Section 2.3.2) during model design. An even more detailed analysis can only be performed through further case distinctions and is therefore not considered here. However, we want to show that planning takes place fast enough. Of course, the actual planning time also depends on the machine on which the algorithm is executed. In our context, *fast enough* means that the algorithm completes planning for the next day within a fraction of that time, on a machine that is not particularly powerful. Hence, we will record the time needed for planning across input problems and compare them against each other. An evaluation will also be made based on the magnitude of planning time compared to one day.

We will then assess the effectiveness of risk attitudes and generated plans in a qualitative manner. However, since none of the approaches examined in Chapter 3 considers multiple objectives (except for [LL20], but using different measures), plans can only be compared among each other. A useful

approach to this end is a comparison of the plans expected utility vectors. Different risk attitudes can be compared based on the position of their Pareto front as well as the choice of individual actions contributing to a plan. Furthermore, the number of non-dominated plans, and the position of extreme cases can provide information about the quality of the planning domain.

6.1.2 Setup

The generation of plans is carried out with a publicly available JSHOP2 distribution [Aut] executed using the Java runtime environment (version 8.0.3710.11), on a computer system running Microsoft Windows 11 Pro (22H2). The software is executed on a machine featuring an Intel Core i7-8550U CPU and 16GB of RAM (2400 MHz). Corresponding commands for compiling and executing the domain are executed in a terminal (version 10.0.22621.2134).

To keep the evaluation comparable and comprehensible, we use publicly available data. However, no dataset is so extensive that it provides all the data considered in the presented domain. We therefore have to interleave data from different sources: We rely on the guidelines given in [BMS10], for setting room comfort requirements. For outdoor conditions, we use the Jena Weather dataset [Max], a very large dataset for weather forecasting, where we use the actual measurement as predicted value. Accordingly, we assume the building's location to be Germany. Data on the local electricity price is obtained from the German Bundesnetzagentur [BB].

In order to present as diverse scenarios as possible with a manageable number of tests, we select four days from the year 2020. More precisely, the days with the fewest and most hours of sunshine, as well as the two days in between. That is, 21^{st} December, 21^{st} June, 21^{st} March and 21^{st} September respectively. For each of these days, all possible plans are generated according to the design of the domain. These are filtered during search using the pruning strategy described in Section 5.3.4. To compare the utility vectors, the actual non-dominated set is then filtered out again.

The setup of problem-files also requires the properties of the rooms to be managed. Section 2.1 lists a wide variety of building types. It can be noted that many of the buildings mentioned have rooms in a medium size range. For example, classrooms, offices, or patient rooms in hospitals fall into that category. For further consideration, we assume a building with one such room of 60 m^2 must be managed. The characteristics of this room relevant to the planning domain are determined by rough calculation. For more details, we refer to Appendix A. Resulting values for the considered parameters are:

- Two outside walls, thermal time constant of 72 h.
- Proportionate to the building an effective PV area of 6 m^2.
- Proportionate to the building a battery with 5 kWh storage capacity.
- One heater with 4 kW of power, heating 2 $\frac{K}{h}$ per operation.
- One AC with 1.5 kW of power, cooling 2 $\frac{K}{h}$ per operation.
- Ten windows with an area of $1m^2$ each. The actuators consume a total of 300 Wh per operation, reducing the time constant to 4 h when opened.
- One blind per window. The drivers consume a total of 150 Wh per operation, dividing illuminance by 20 when closed.

- Twelve neon tubes consuming 700 W per hour, adding an illuminance of 600 lux.

To perform the tests as described, one problem-file for each instance is generated automatically from the Jena Weather dataset and electricity prices. An example of what input data looks like was already presented in Figure 4.1 and 4.2. The exact course of data is of secondary importance to understand following argumentation. However, all input scenarios are presented in Appendix B for completeness. To examine the impact of the agent's risk attitude, each input day is evaluated with different attitudes. We select five static risk attitudes according to Equation (2.7) for investigation:

1. Highly risk-averse: $a = -1$, $\alpha = 0.5$,
2. Highly risk-averse: $a = -1$, $\alpha = 0.125$,
3. Risk-neutral,
4. Weakly risk-seeking: $a = 1$, $\alpha = 0.5$,
5. Highly risk-seeking: $a = 1$, $\alpha = 2$.

6.2 Execution

The presented planning domain is compiled and executed on generated problem-files using the hardware setup given above. Four days and five risk attitudes each, make a total number of 20 scenarios. These are presented by day for the qualitative comparison and by risk attitude for the quantitative comparison.

6.2.1 Data Representation

Data is converted into a JSHOP2 problem-file according to the transformation described in Chapter 5. All feasible plans are returned as a list of actions with the associated expected utility vector and the total planning time, when executing the planner. To analyse the data, they are prepared for examination as follows:

For a quantitative evaluation of complexity and scalability we present the time needed for planning in each scenario. To compare the results, the number of plans generated, and the number of search steps required are depicted as well. Furthermore, the number of actual non-dominated plans resulting. Finally, the order of magnitude of planning time is compared with the 24 hours of one day without further graphical representation.

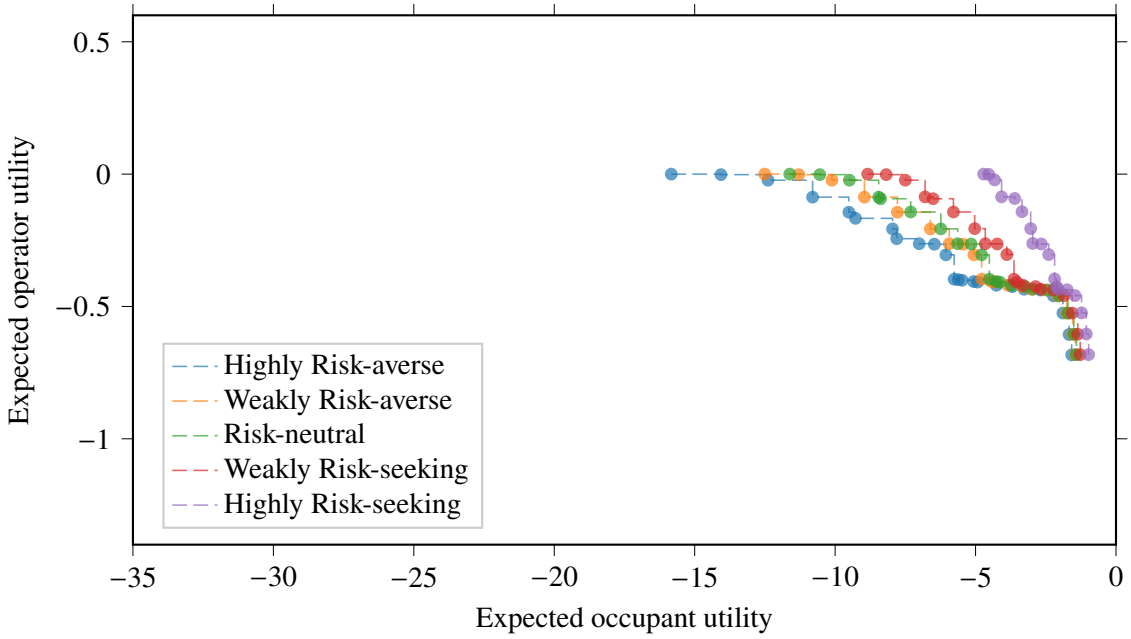
For a qualitative evaluation of plans we present two components of the expected utility vector. These are shown according to Figure 2.6, where one utility function is given by the occupant component and the other by the operator component. For clarity, only the set of non-dominated plans is depicted. Respective plans are coloured equally for one risk attitude and connected by a line representing the Pareto front. An evaluation can then be performed on basis of relative and absolute position of plans to each other. Similarly, the Pareto fronts of different risk attitudes can be compared to each other. This provides information on the general functionality of the domain and search algorithm. Individual plans of different risk attitudes but similar expected utilities can be compared based on their actions. Such a comparison can highlight the effectiveness of different risk attitudes.

6.2.2 Data Analysis

Results of the two evaluations are presented and analysed in the following. We will first perform a qualitative analysis of generated plans and then move on to the analysis of problem complexity.

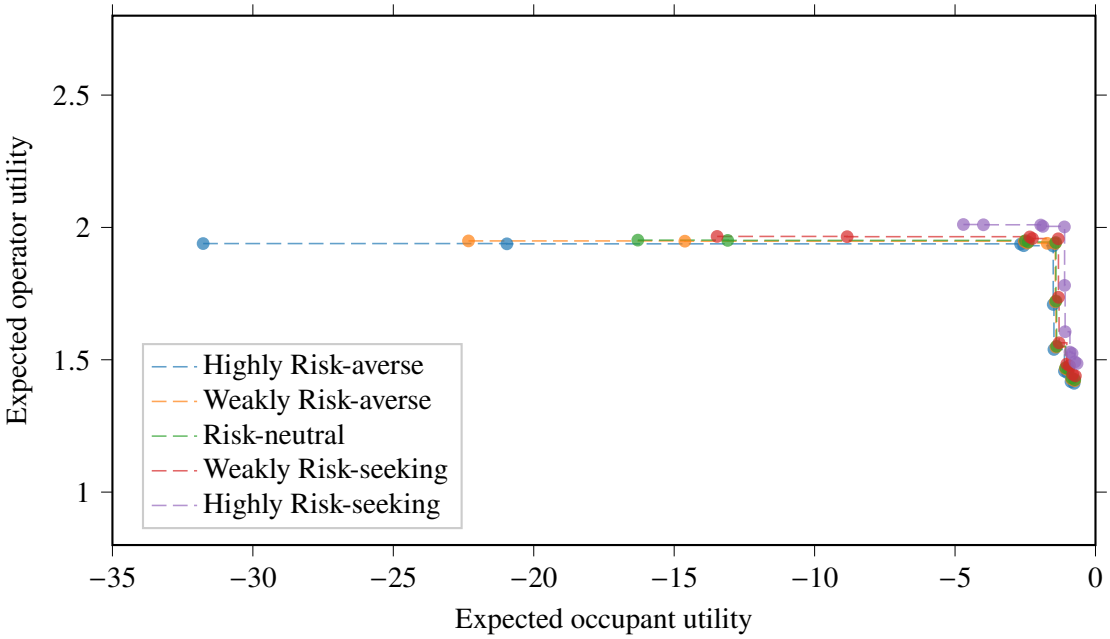
Comfort in Various Scenarios

Here we want to analyse the planning domain's functionality and the quality of generated plans. For this purpose, we neglect the task of appliance scheduling and only perform room and energy management. As described, we will perform this evaluation for 20 different scenarios. Note that there will be two extreme cases in each of these scenarios: Firstly, a plan in which no comfort actions are taken, yielding the maximum expected operator utility. Secondly, a plan that spares no cost or effort to satisfy occupants and thus maximises expected occupant utility. The position of these two extremes and the course of the Pareto front between them is of particular interest.

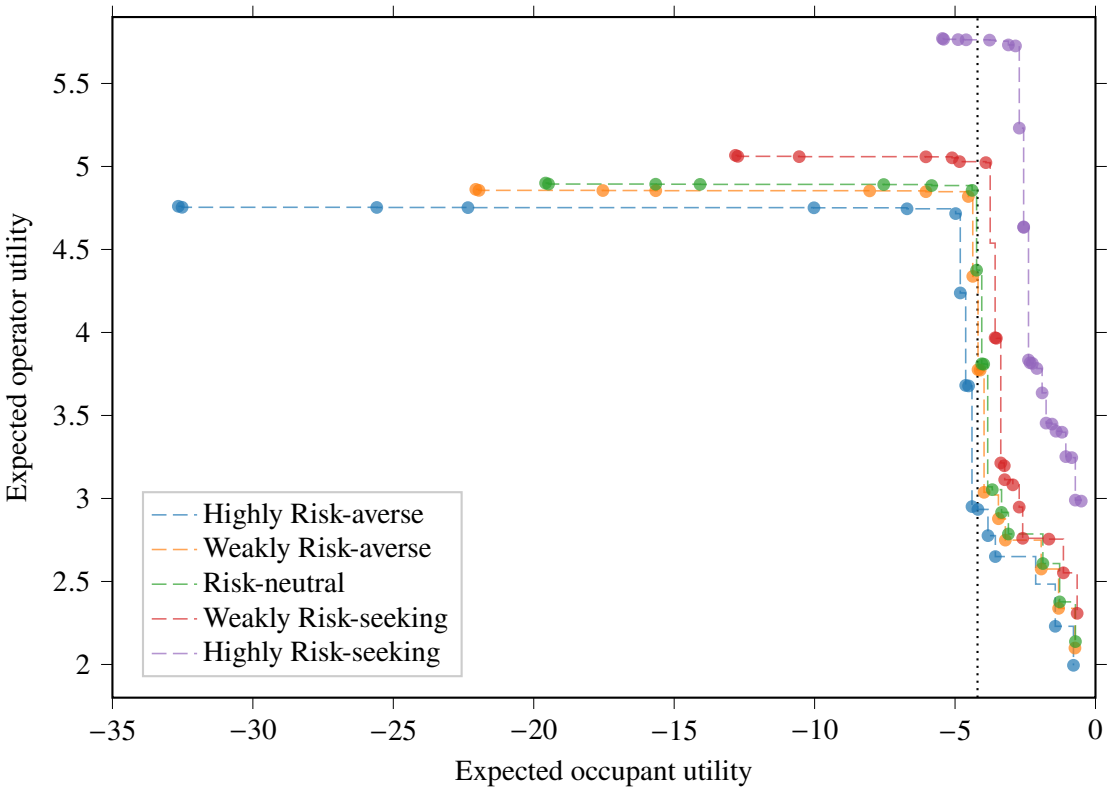


(a) Generated plans for 21^{st} March 2020

Figure 6.1: Expected utility vectors of non-dominated plans

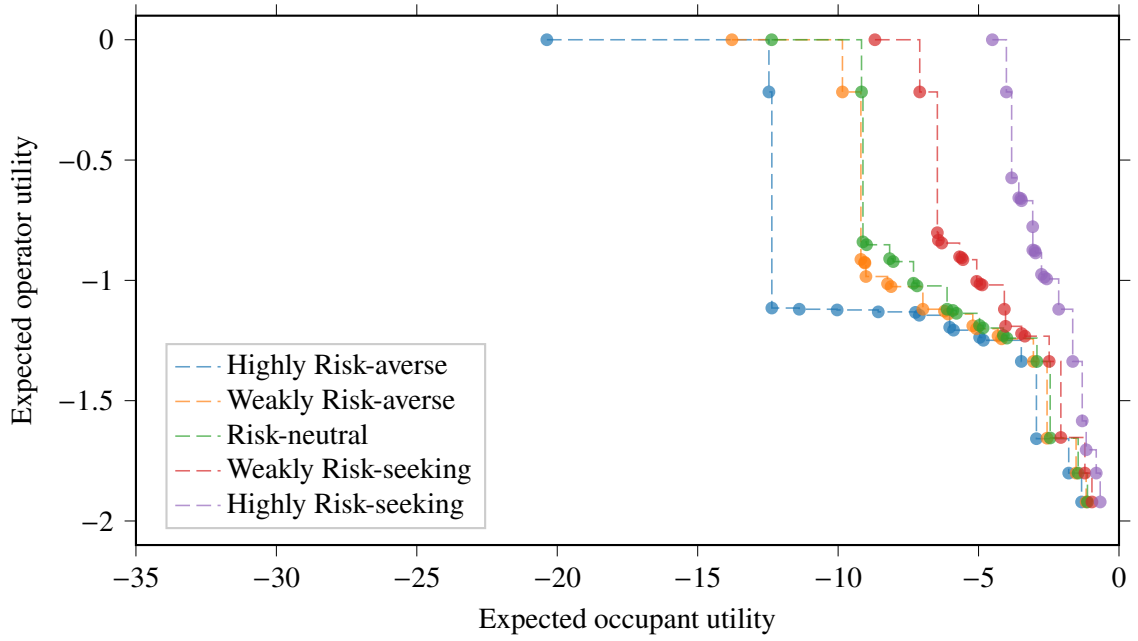


(b) Generated plans for 21^{st} June, 2020



(c) Generated plans for 21^{st} September 2020

Figure 6.1: Expected utility vectors of non-dominated plans



(d) Generated plans for 21^{st} December 2020

Figure 6.1: Expected utility vectors of non-dominated plans

The horizontal axis depicts expected occupant utility and the vertical axis the expected operator utility. Each point corresponds to a plan, with plans of a certain risk attitude coloured equally. A line connects all points of one risk attitude - the Pareto front generated using further support points from the x value of the last and the y value of the next plan.

The vectors of expected utilities are presented and grouped by days in Figure 6.1. Overall, it can be seen that on colder days, i.e., 21^{st} December and 21^{st} March, the operator utility is only in negative range. In contrast, it is between 1.4 and 2 on 21^{st} June and even between 2 and 6 on 21^{st} September. The large discrepancy on 21^{st} September is probably due to the disproportionately high energy price of that day. In addition, more energy is available for harvesting according to the hours of sunshine. Expected occupant utility reaches about -20 in March and December, while it drops to -35 in June and September. This seems to be due to blinds and windows staying in their default position in the extreme case without comfort actions. While a closed window is a good protection against cold temperatures ($\tau = 72h$), open blinds are not a good protection against light on bright days. This observation therefore suggests that the selected default state and the choice of $c_{occupant}$ should be reviewed. The system is not able to keep the expected occupant utility at zero in any of the presented scenarios. This is especially surprising for the extreme case in which all necessary comfort actions are taken. It can, however, be attributed to the dependence of several quantities on single actions. E.g., if air quality is too poor, the window will be opened, regardless of whether the resulting drop in temperature leads to greater dissatisfaction.

Furthermore, the Pareto front of Figure 6.1b is rather edgy compared to those of Figure 6.1a, 6.1c and 6.1d. The main reason for this seems to be the general size of the non-dominated set, which we will examine in more detail in Section 6.2.2. However, also in Figure 6.1c the transition between occupant and operator utility seems to be rather binary. Meaning that there are plans of high

expected occupant utility and plans of high expected operator utility. There is a clear transition point where the one switches to the other. Whereas in Figure 6.1a and 6.1d there are a number of plans in the intermediate segment. On the one hand, this can be attributed to the fundamental demand for more comfort actions to stay in occupant comfort range. This is because the outside temperature on these days is significantly below the comfort range, while it is only slightly above the comfort range on warm days. On the other hand, it can be attributed to the delay in taking actions, because each time actions are not taken, the temperature drops and so does expected occupant utility. This can also be seen in the spread of different risk attitudes on these days. Note that the Pareto front is compressed in the direction of expected occupant utility with increasing willingness to take risk, while it is shifted slightly upwards in direction of operator utility. The reason for this is that, due to its unobservability, occupant dissatisfaction is largely based on estimates. In contrast, operator utility is largely deterministic. Only the amount of renewable energy to be harvested and the associated expected utility increase significantly with the willingness to take risk.

Within one risk attitude, there seem to be greater leaps in operator utility than in occupant utility on most days. This is mainly due to the different dimensions of the axes, i.e., a certain disparity between the two utilities. As already discussed, these represent aspects that cannot be compared with each other directly. Therefore, an argumentation regarding something like a *slope* is rather difficult. What can be compared, however, are points of equal utility values between different risk attitudes. We can, for example, compare the plans of 21^{st} September, which achieve an expected occupant utility of about -4. For each risk attitude, we select the plan that is closest to that line. The exact coordinates of the points and the actions of the corresponding plans are listed in Table 6.1.

Overall, it can be said that the risk-averse and the risk-neutral plan include the same comfort actions, even though a risk-neutral agent delays their execution. However, both perceive about the same expected occupant utility. In contrast, the expected operator utility is far lower for the risk-averse agent. This is due to harvesting less energy and higher associated cost if the estimation is still too high. Compared to these two plans, the risk-seeking agent takes only one convenient action, namely closing the blinds at 7 a.m. After that, first the storage is filled up and then all generated energy is sold to the grid. This results in a significantly higher operator utility because more energy is available for sale and the probability of overestimating the generated solar power is disregarded. From the plans and differences shown, it can be seen that the functionality of both the domain and the risk-awareness of agents is fulfilled. However, it is questionable whether the application of these plans makes sense in reality. In the case of a risk-seeking agent, for example, the room is never ventilated during the day, which is likely to result not in dissatisfaction but a health risk for occupants. This is possibly related to the choice of a linear dependency in Equation (4.1). Re-qualifying the quantities according to their importance, for example by using Pavlov's Pyramid, could help resolve this inaccuracy.

Computing Complexity

Computing time is a critical factor of applications, especially when they are meant to create plans for subsequent execution. In this evaluation we want to analyse several facets of this aspect: Firstly, whether the implementation presented is fast enough for its purpose. Secondly, which considered input factors have an influence on the problem complexity. Finally, how problem complexity affects its scalability.

Highly Risk-averse		Risk-neutral		Highly Risk-seeking	
expected occupant utility	expected operator utility	expected occupant utility	expected operator utility	expected occupant utility	expected operator utility
−3.828	2.776	−4.049	3.811	−3.771	5.761
(!charge_battery 6 95)		(!charge_battery 6 95)		(!charge_battery 6 95)	
(!close_blinds office 7)		(!close_blinds office 7)		(!close_blinds office 7)	
(!charge_battery 7 485)		(!charge_battery 7 488)		(!charge_battery 7 500)	
(!charge_battery 8 982)		(!charge_battery 8 996)		(!charge_battery 8 1075)	
(!charge_battery 9 1412)		(!charge_battery 9 1443)		(!charge_battery 9 1634)	
(!charge_battery 10 1733)		(!charge_battery 10 1775)		(!charge_battery 10 1696)	
				(!sell_energy_to_grid 10 345)	
(!charge_battery 11 293)		(!charge_battery 11 203)			
(!sell_energy_to_grid 11 1636)		(!sell_energy_to_grid 11 1770)		(!sell_energy_to_grid 11 2245)	
(!open_window office 12)					
(!sell_energy_to_grid 12 1934)		(!sell_energy_to_grid 12 2007)		(!sell_energy_to_grid 12 2272)	
		(!open_window office 13)			
(!close_window office 13)					
(!operate_heater office 13)					
(!discharge_battery 13 2193)					
		(!sell_energy_to_grid 13 1840)		(!sell_energy_to_grid 13 2061)	
		(!close_window office 14)			
(!charge_battery 14 1551)					
		(!sell_energy_to_grid 14 1544)		(!sell_energy_to_grid 14 1701)	
		(!operate_heater office 15)			
(!charge_battery 15 642)		(!discharge_battery 15 2825)			
(!sell_energy_to_grid 15 519)				(!sell_energy_to_grid 15 1241)	
		(!charge_battery 16 709)			
(!sell_energy_to_grid 16 704)				(!sell_energy_to_grid 16 733)	
		(!charge_battery 17 226)			
(!sell_energy_to_grid 17 225)				(!sell_energy_to_grid 17 228)	
(!open_blinds office 18)		(!open_blinds office 18)			
(!discharge_battery 18 6)		(!discharge_battery 18 6)			
				(!sell_energy_to_grid 18 9)	

Table 6.1: Excerpt of resulting plans for 21st June 2020
All actions with real world representation of regarded plans are listed above. All other actions have been removed for the purpose of clarity and have no significance here. The first term after an action name indicates the hour of intended execution, a second term the amount of energy in kWh.

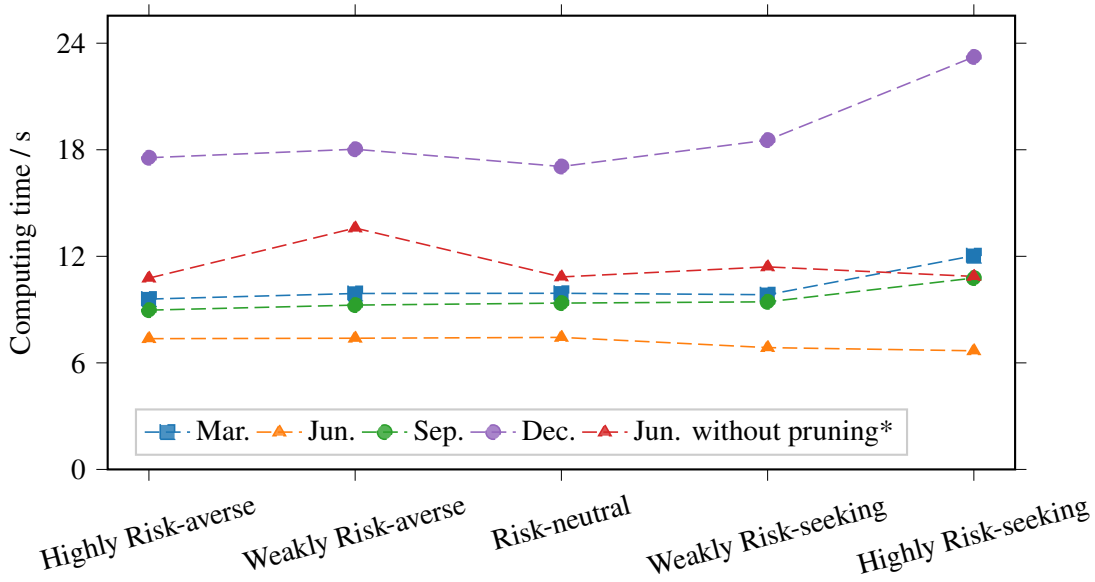


Figure 6.2: Computing time for generation of plans

Required computing time is plotted on the y axis, while the x axis separates risk attitudes. Each point corresponds to the computing time for one of the 20 scenarios considered. In other words, the time needed to generate one curve from Figure 6.1. Points of the same day are connected by a dashed line. * Calculation without pruning was only possible for 21^{st} June.

Results of the computing time evaluation are depicted in Figure 6.2. It can be seen that planning time generally depends on the considered input data, i.e.,the date. Required planning time is lowest for 21^{st} June with approx. 7 *s*, while it is highest for 21^{st} December with about 18 to 24 *s*. The 21^{st} of March and September are in between with about 10 *s* each. This can be attributed to the conditional methods discussed in the context of room management and the aforementioned outdoor temperature of these days. If quantities of indoor climate remain within their preferred range, there is no need to take action resulting in a small search space. A dependency on hours of sunshine is therefore given rather indirectly and in reverse direction. A clear dependency of planning time on the risk attitude is not evident. There is no general trend, although planning times for the *Highly Risk-seeking* attitude are highest in March, September and December compared to the other risk attitudes on each day. An increase in computation time is probably partly due to the allocation of already sparse RAM (a computation without pruning was only possible for 21^{st} June, all other attempts failed due to shortage of heap memory). It is therefore worth looking at the number of search steps and generated plans. Even the longest planning time of about 24 *s* is far below the duration of a day and can therefore be neglected. The limiting factor is more likely to be space rather than time complexity. This is perhaps due to the complicated presentation of information in the implementation.

Figure 6.3 and 6.4 present the number of generated and non-dominated plans for all 20 problem instances, respectively. The number of generated plans ranges from 150 to around 400, while the number of actual non-dominated plans is only between 10 and 25. A dependency between the number of plans and the input problem can be observed in both here. The fewest plans have been generated for June, which also results in the fewest non-dominated plans. In contrast, for December only the most plans are generated, while March has the most plans that are actually non-dominated.

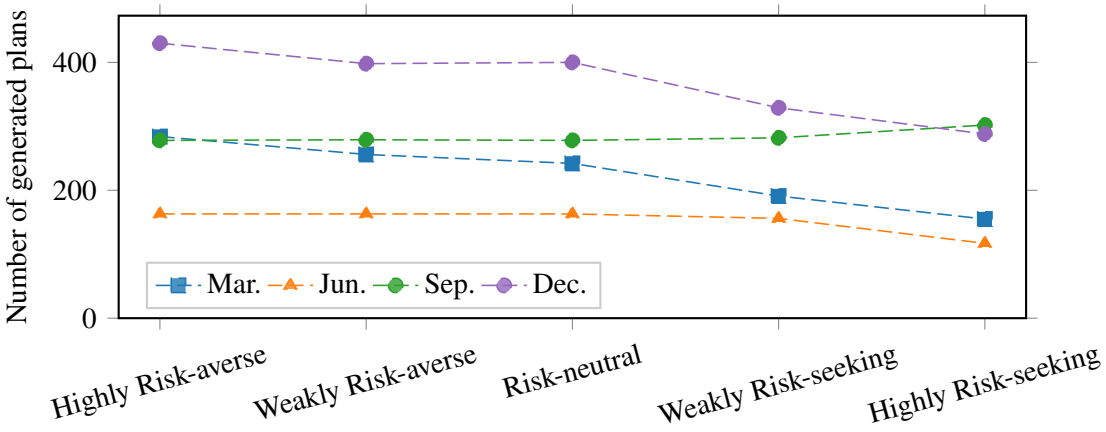


Figure 6.3: Number of generated plans

The number of computed plans is plotted on the y axis, while the x axis separates risk attitudes. Each point corresponds to the number of generated plans.

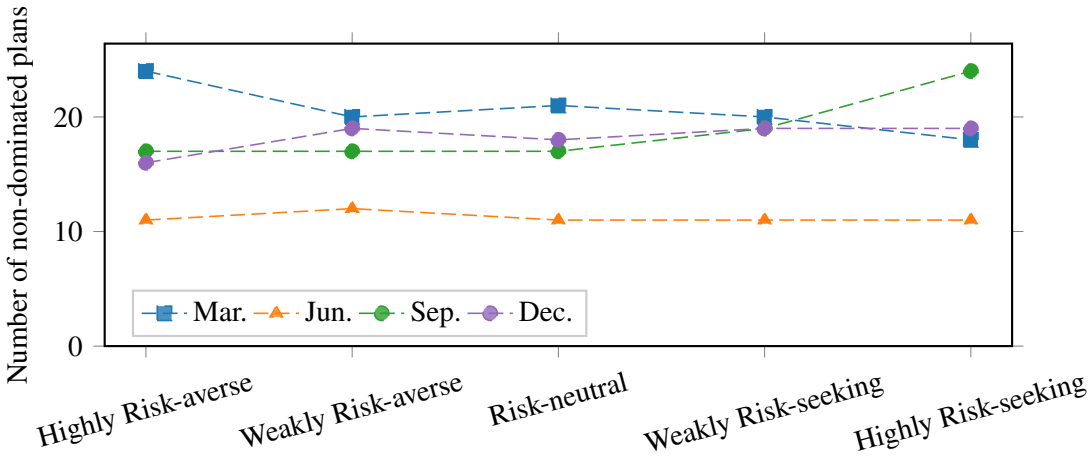


Figure 6.4: Number of non-dominated plans

The number of computed plans is plotted on the y axis, while the x axis separates risk attitudes. Each point corresponds to the number of actually non-dominated plans.

However, this again suggests an increased complexity due to a low outdoor temperature and conditional methods. The number of generated plans seems to decrease with increasing willingness to take risk. Only September does not follow this trend. In contrast, there is no clear trend in the number of actual non-dominated plans on the agent's risk attitude. This indicates that an earlier discarding of plans occurs due to Equation (2.6) and the increased dependence on temperature. However, a dependence on risk awareness per se cannot be observed.

The number of steps required to find all non-dominated plans for one scenario are depicted in Figure 6.5. Search expansion steps range from 70.000 and 350.000. There is a high degree of correlation with Figure 6.2 (Considering *Weakly Risk-aware* for 21^{st} June without pruning an outlier). Thus, planning time is primarily related to the number of search steps. A profound

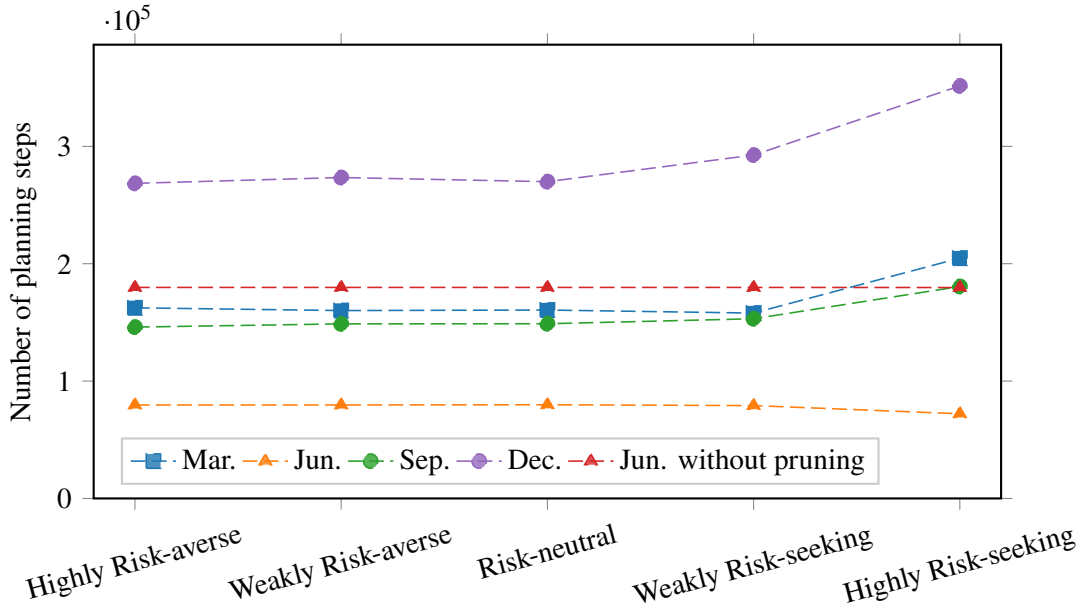


Figure 6.5: Number of expanded search steps

The number of search steps is plotted on the left y axis, while the x axis separates risk attitudes. Each point corresponds to the number of search steps during planning.

dependency on risk attitude cannot be observed. However, there is an influence if the risk attitude leads to a plan being considered non-dominated throughout search. In contrast, there is a large dependency on the input problem. This generally results in poor scalability, as none of the scenarios considered represent a true extreme case for room management. This is also highlighted by the fact that search without pruning was only possible for the least complex problem of 21^{st} June. However, it can be stated that pruning cuts the number of expanded nodes in half. Reducing planning time by an average of 30 - 40 %. The means used therefore prove to be effective improvements for the search algorithm.

6.2.3 Data Interpretation

In the last section we presented and analysed multiple results of the described implementation. Now we want to discuss critically and interpret what they mean for the presented planning domain and its implementation in JSHOP2.

Feasibility and Abstraction of the Solution

The dependency of planning time on input data reveals that the presented implementation does not show good scalability. Nevertheless, planning time is at a negligible magnitude for the considered use case. The main problem, however, seems to be the excessive memory demand due to a large search space. Although this could be reduced by pruning, it is still on the edge of computability. Note that the chosen degree of abstraction is already quite high. So high, in fact, that the generated plans are only suitable for DAP energy trading, but not for the execution of generated plans in a real building. On the one hand, the temporal resolution is too coarse. On the other hand, the

discretisation of actions as well as the simplification and restriction of considered physical effects are obstacles for such application. The strategy for finding good solutions must therefore be improved, since further abstraction of the domain from its environment is not desirable. The concept of Satisficing therefore appears to be the better approach than an exhaustive search for optimal solutions.

The issue of computability seems to be related to complicated information encoding. As Algorithm 4.2 and 4.1 suggest, some optimisations are better performed outside a planning domain. An implementation in a different programming environment also facilitates the application of more advanced data structures. In general, the set of non-dominated plans could be generated by iteration over all combinations of actions using regular programming techniques. Nevertheless, such an iteration can be represented by HTNs in a more expressive language. As already pointed out by Georgievski [Geo15] and Schattenberg: State-based HTN planners are rather problem-solving programming languages.

JSHOP2 is also not designed to describe MO planning problems, as the encoding of one-dimensional costs illustrates. Although many features can be added by implementing them in Java, manual extensions for a multitude of specialities are not convenient. The consideration of risk attitudes is not yet implemented in any HTN planning software as well. Revealing that there is still a lot of potential in the field of automated planning and the implementation of planning software. Especially since the generation of action sequences without feedback is an essential capability for automated systems.

Quality of Generated Plans

Generating the Pareto front is a good way to reduce the set of plans to a manageable size. In some cases, however, there are still a few plans whose expected utility does not differ significantly. A comparison of individual actions or a grouping of points could help to further reduce their number. A clustering in the space of expected utility values could help to limit the range of considered solutions. The generation of a selected solution set is also facilitated by considering a hypervolume indicator. Such indicators can be used to guide the search to solutions in defined regions, while ensuring Pareto optimality. However, the hypervolume requires the specification of a reference point, which is subject of ongoing research [IISN18; PIH+23].

The specification of a risk attitude seems to be a useful tool to support narrowing the set of plans in the MO case. This might seem contradictory at first, as the consideration of several risk attitudes in our analysis initially generates more alternative plans. However, specifying a risk attitude changes the relationship of plan utilities to each other, so that some move closer together or drift apart. Combined with a grouping of plans, this could help to present a smaller set of more diverse solutions. Furthermore, the specification of a risk attitude in the presented domain already constitutes a certain weighting between occupant and operator utility, because the second is largely deterministic. Thus, the number of solutions effectively decreases as the willingness to take risk increases. Overall, however, it must be noted that risk attitudes challenge the law of large numbers in the long run. With an increasing number of applications using a static risk attitude, the risk-neutral attitude becomes more relevant again. The consideration of dynamic risk attitudes therefore becomes meaningful for an application in real world problems.

7 Conclusion and Outlook

The starting point for this thesis was to what extent automated planning can reduce the energy consumption of buildings while keeping the noticeable impact on inhabitants' comfort as low as possible. It soon became apparent that the impact on inhabitants is difficult to quantify. However, a review of utility, risk and uncertainty pointed to Savage's theorem. This states that a subjective belief about the probability of outcomes has the same value for decision-making as the real probability distribution when a full assessment is not possible. In the literature researched, comfort and monetary value are often put on the same level with each other. This decision is, however, hard to justify. It is possible to infer the monetary value that a person is willing to spend for a certain level of comfort by *preference elicitation*. But such an elicitation is not performed, nor would it be a good compromise to assume the same value for everyone. Therefore, a utility vector of separate utilities was used in this thesis to consider preferences independently. Research in MO decision-making could be used to describe this problem of non-residential buildings more precisely.

We present a new AI planning domain for the environment of smart non-residential buildings. Particularly the HTN concept was chosen for this purpose, as a risk-aware approach was presented recently. The three tasks room management, appliance scheduling and energy management were identified as those mainly contributing to the expected electricity demand of a building. A quantification of this value is needed to buy electricity from a provider in advance when the building is connected to the smart grid. The aim is to reduce local electricity bills and improve global grid stability. However, an actual reduction in electricity demand can only be achieved through more efficient actuators, more efficient building designs or a reduction in comfort actions. Due to the diverse nature of the problem, there is no longer a unique optimum, but rather a set of non-dominated plans. However, the number of generated solutions needs to be further reduced, since *one* acceptable compromise between comfort and energy consumption is to be chosen later. Methods such as the use of a hypervolume indicator could resolve this issue. The designed planning domain can be used to generate plans for a variety of static risk attitudes. The consideration of these attitudes shows only an indirect impact on planning complexity. The use of dynamic risk attitudes should be further investigated for permanent real-world application. Therefore, the initial question can only be answered in a limited way: Automated planning can reduce the energy consumption of buildings by reducing comfort to a certain level. However, the impact of a reduction and the associated energy savings can be quantified and adjusted subsequently. Investigating the willingness of occupants to accept such reductions remains a point for further research.

The presented domain was realised using the planning software JSHOP2, which is an established implementation. Insights for improving search algorithms could be obtained by reviewing graph theory and creating heuristics for MO cases. These could, however, not be implemented without modifying JSHOP2's search algorithm due to the state-based nature of the domain. Instead, a pruning mechanism was integrated into the domain itself. The scalability of the solution is insufficient due to high memory demand. A short planning time, however, indicates that the chosen approach is reasonable for the problem posed. Hence, the weak spots are the implementation as

well as the cumbersome representation of information in planning domains. The number of existing planning softwares and languages suggests that none of them is capable to capture all ideas of their users. The number of constructs presented to model real world environments also indicates that there is not yet a universally applicable concept to automate planning. While specialisation and different levels of abstraction are useful for such constructs, they should be supplementary and inclusive. However, none of the analysed concepts offers an expressive description of real-world problems, with an executable implementation, which makes it possible to find efficient solutions for the problem posed.

The solution presented therefore simplifies the environment to generate efficient solutions, as do other approaches. Efficient search is achieved through a high degree of abstraction as well as a focused search. The limitations of the presented solution lie primarily in the abstraction from the described problem: The restriction to a few measurable quantities, the omission of physical principles as well as the modelling of actions and effects. The designed domain can therefore hardly be used for more than an estimation of the hourly electricity demand. The use of generated plans as instructions at the day of application is limited by the low temporal resolution and the discretisation of actions. However, a higher temporal resolution is limited by the increasing computational complexity, while the planning of continuous actions requires the use of a different planning construct. The deterministic nature of the domain also opposes the use of plans as instructions, which would hardly stand up to an application in the real world.

The presented implementation can be applied to similar problems easily, due to separation into problem- and domain-file. Thus, the actual building type is of minor importance due to the level of abstraction if the properties can be quantified in the form of the specified parameters. The automatic generation of problems allows a uniform format and problems could be received through an input mask. Reducing the number of considered actuators does not pose a problem. However, a room with more or different actuators requires a modification of the domain. An extension of the planning problem to larger building complexes is hindered by memory requirements of the presented implementation. It must be noted that, if the domain is used for multiple rooms at the same time, the expected occupant utility accumulates the satisfaction of all occupants. This limitation can be overcome by a dynamic extension of the utility vector. The solution presented is, however, distinct from many other approaches in this field by considering independent utilities. The self-evident way in which different objectives are combined in other approaches shows how this complication is omitted. Thus, maximising comfort is presented to be the only solution. However, in a world of increasing greenhouse gas emissions and thus rising temperatures, humanity cannot afford to maximise without alternatives. The continued pursuit of greater comfort leads to an increase in the total energy demand, even with more efficient implementation of that comfort. The ambition to such goals illustrates the blindness of our society towards the limitation of resources.

Examined shortcomings provide several directions for future research: On the one hand, the described domain can be extended and detailed in different aspects. Here we would like to emphasise the detection of occupant presence and the implementation of alternative energy management strategies. Furthermore, the implementation of a resulting domain in a different planning software could be useful to reduce the number of modifications required and increase search efficiency. For this purpose, HyperTensionN U seems to be a suitable choice, even though, or perhaps because, it uses different attributes to describe problems. An improvement in planning could also be achieved by modifying the search algorithm of JSHOP2. Not least, modelling the environment of smart buildings in a different planning construct seems to be appropriate. Such a construct should grasp

continuous actions and the probabilistic nature of their effects, but allow the generation of explicit plans at the same time. The concept of risk attitudes has proven to be effective, and its application should be explored in other fields of application.

The discrepancy of different planning concepts, simultaneous planning and acting approaches and resulting plan repair strategies raises the question of a need for new planning concepts. In particular, the role of probabilistic actions, the possibility of external influences on states, and the coupling of plans, actions and state feedback need to be reconsidered. In MDPs the current state is always known before taking one action, while the most probable state is estimated in a Partially Observable MDP. In comparison, the history of states is always known in a deterministic planning construct, which enables the generation of an action sequence. Probabilistic planning models, as well as triggering replanning when plan execution does not achieve the intended goal, are attempts to close the gap between these two paradigms. Humans often plan and act in several iterations and layers. Using goals, tasks, state-based and plan-based approaches as well as reactive closed and open loop controls. Detailing of sub-plans takes place with respect to the time horizon of their application. Further research on GTNs and concepts like lazy lookahead or finite horizon planning is therefore interesting. Traversing a planning search space to a certain depth depending on the time until execution seems a reasonable approach. The importance of Satisficing as a lazy generator of alternatives compared to the pre-generation of all optimal alternatives should be reconsidered.

The need to modify the presented domain when considering different types of actuators reveals another interesting field of research: The automatic generation or adaptation of planning domains, as presented in [AFP+18; DPR21]. A complete generation, however, also requires the identification of - possibly manifold - objectives involved. The field of AI is receiving more attention than ever before in its almost seventy-year history. Recent breakthroughs in deep neural networks and large language models are interpreted as evidence for a soon accomplishment of Artificial General Intelligence. However, AI involves much more than machine learning. When looking at human problem-solving capabilities, these include not only condensing information to solve specific problems, but also identifying and generalising them. That means finding a suitable level of abstraction for a task at hand to recognise relevant objects and actions of an environment. Anticipating known solutions by abstracting and categorising helps to solve supposedly new problems more efficiently. This requires constant recognition of details, classification, generative expansion of opportunities and selection of a context to then specify an appropriate subset of these information. Therefore, the field of AI planning and especially the automatic adaptation and generation of planning domains remains an important subject of research.

A Scenario Modelling

For the purposes of this study, we will assume a medium sized room, like a classroom or a single office of 60 m^2. Let us further assume that the room is 6 m wide and 10 m long, with a ceiling height of 2.5 m. Of the four walls, two are external walls, while the other two, as well as the ceiling and floor, are adjacent to other rooms and are therefore neglected. If we further assume that the room has 10 windows, each with an area of 1 m^2, the thermal conductivity value of the room ($U \cdot A$) can be estimated to be

$$0.4\frac{W}{m^2K} \cdot (6+10)m \cdot 2.5m - 10 \cdot 1m^2 + 1.2\frac{W}{m^2K} \cdot 10 \cdot 1m^2 = 24\frac{W}{K}.$$

The necessary heat output due to transmission heat at a temperature difference of 30 K (20 $°C$ indoors and -10 $°C$ outdoors) is then

$$24\frac{W}{K} \cdot 30K = 720W.$$

Additional heat output due to air exchange at $2\frac{m^3}{h}$ per m^2

$$60m^2 \cdot 2.5m \cdot 2m^3 \cdot 0.34\frac{m^3}{m^3K} \cdot 30K = 3060W.$$

A heater is then configured with the corresponding power rating to compensate for the sum of the losses ($720W + 3060W = 3780W$). Let us further assume that external walls are 0.24 m thick and are built as a sandwich of insulation and cement. Thus, a 0.08 m layer of cement per wall in the room contributes to the heat capacity of the room (outside, air and insulation are neglected). The heat capacity ($c \cdot m$) of the room is thus given by

$$0.278\frac{Wh}{kgK} \cdot 2000\frac{kg}{m^3} \cdot (6+10)m \cdot 2.5m \cdot 0.08m \approx 1,780\frac{Wh}{K}.$$

With the windows closed and the heating permanently operated, this results in a heating of

$$3780Wh/1,780\frac{Wh}{K} = 2.12K.$$

For AC we assume the same cooling effect with a performance coefficient of 3. An installed unit therefore consumes

$$3780Wh/3 = 1260Wh$$

per hour of operation. From the thermal conductance ($U \cdot A$) and the heat capacity ($c \cdot m$) we can also calculate the time constant of the room to be

$$1,780\frac{Wh}{K}/24\frac{W}{K} \approx 74h.$$

Assumptions and estimations according to [VV15]. For open windows we assume a far lower time constant of about

$$4h.$$

The outdoor AQI is derived by dividing the concentration of CO_2 in ppm by 10, when neglecting other volatile organic compounds [Bre20]. While the windows are closed, the air in the room remains almost constant. A human exhales approx. 0.5 kg of CO_2 per day, or 20 g per hour [cg12]. This increases the CO_2 concentration in the air per hour and person by

$$20g/(60m^2 \cdot 2,5m \cdot 1,25\frac{kg}{m^3}) \approx 100ppm.$$

We assume a two-storey building with a fully PV equipped roof, so the proportionate active area of the considered room is

$$60m^2/2 = 30m^2.$$

The PV panels in use have an efficiency of approx. 20 % and thus the effective area is

$$30m^2 \cdot 0.2 = 6m^2.$$

We calculate $E_{predicted}$ to be the shortwave downward radiation (SWDR) in $\frac{W}{m^2}$ multiplied by the effective PV area. According to the area, we assume a proportionate storage size of

$$5kWh.$$

Assumptions and estimations according to [OL17; SCJ17].

To illuminate the room, 12 neon tubes of 60 W each are available, which emit 100 $\frac{lm}{W}$ at a beam angle of 100°. Illuminance on a table 1 m from the ground is then

$$60W \cdot 100\frac{lm}{W}/(1.5m)^2 \cdot 1\frac{m^2lux}{lm} \cdot cos(100°) = 598lux.$$

For their operation, an energy of

$$12 \cdot 60Wh = 720Wh$$

per hour is required. For the actuators of the windows, we assume 200 W and that a complete cycle of motion takes one minute, so that one actuation takes

$$10 \cdot 200W \cdot 1/60h = 33Wh.$$

For the actuators of the blinds, we assume half the power, so that one operation takes

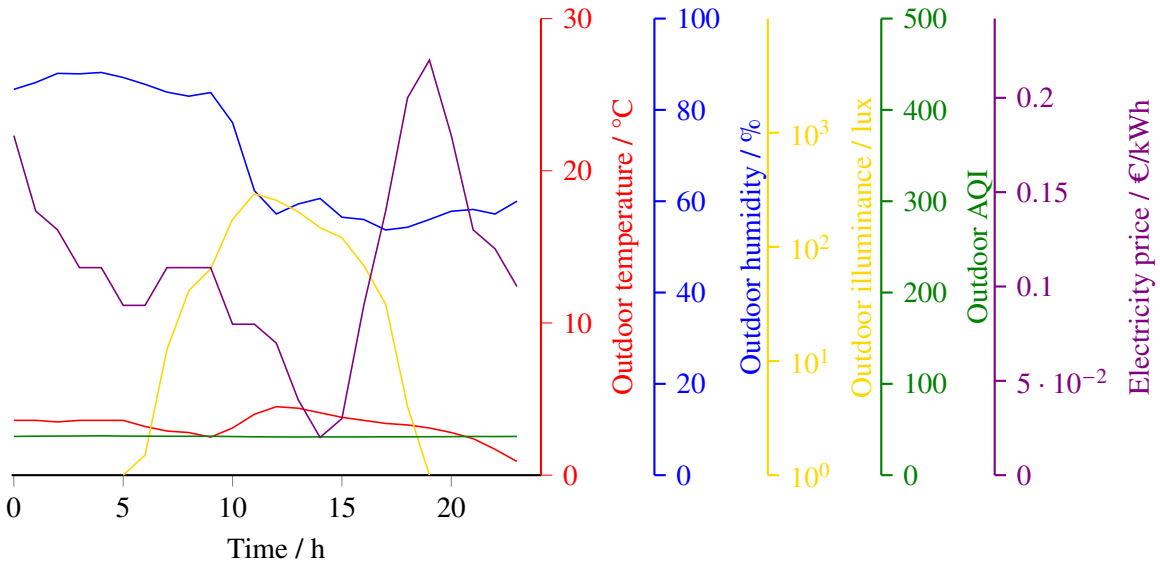
$$10 \cdot 100W \cdot 1/60h = 17Wh.$$

The outdoor illuminance on the ground can be derived from SWDR by multiplying it with 685. We assume that the horizontal component is 10% of this value due to reflection and scattering. Maximum illuminance due to the sun given horizontal irradiation is approximately 20,000 lux, we assume that blinds reduce this effect by 20, e.g.,

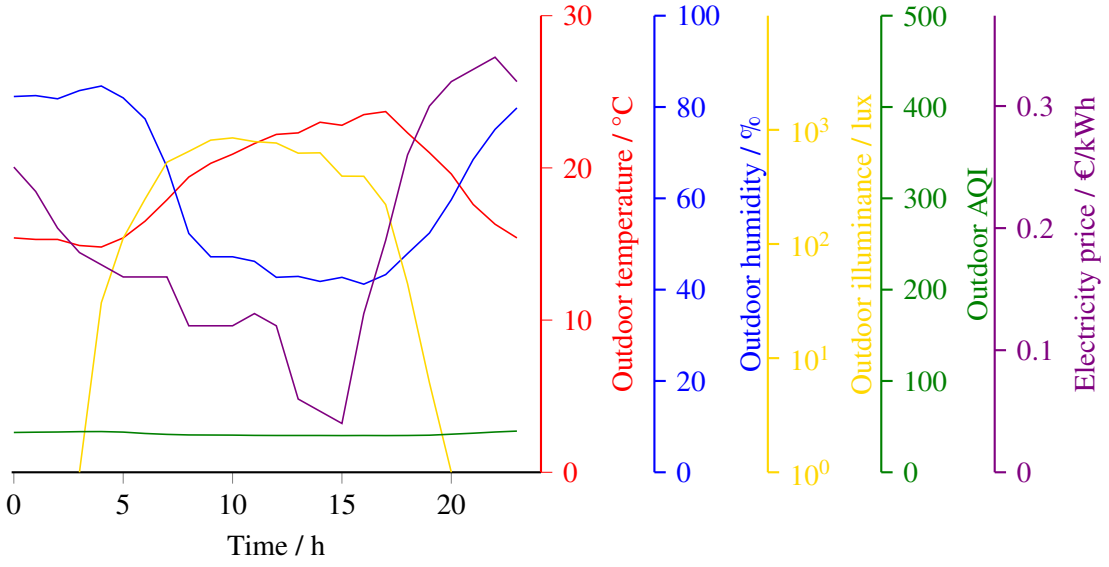
$$20,000lux/20 = 1000lux.$$

Assumptions and estimations according to [DD21].

B Input Data

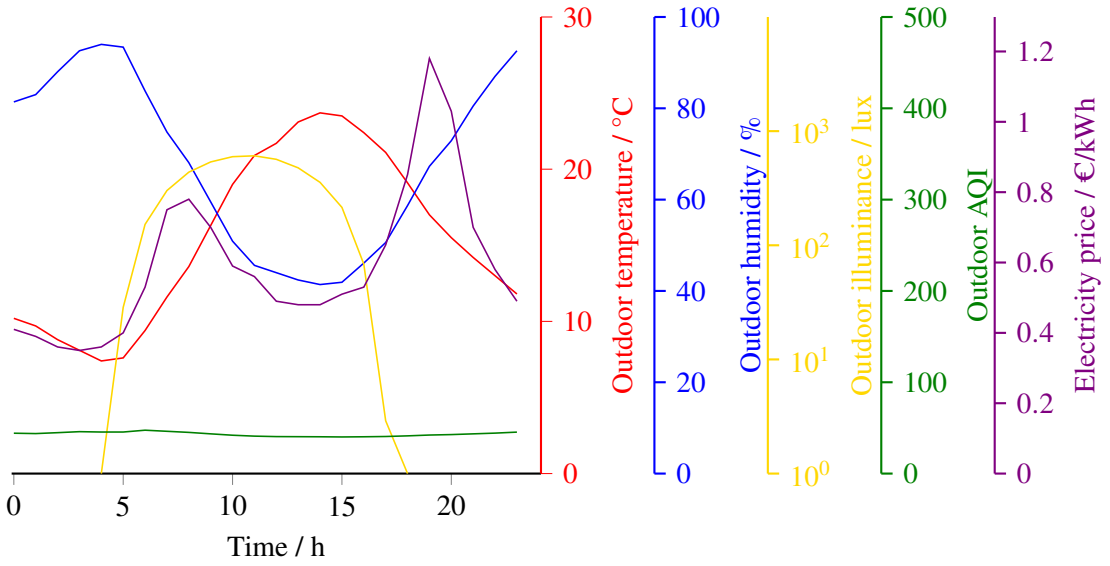


(a) Input data of 21^{st} March 2020

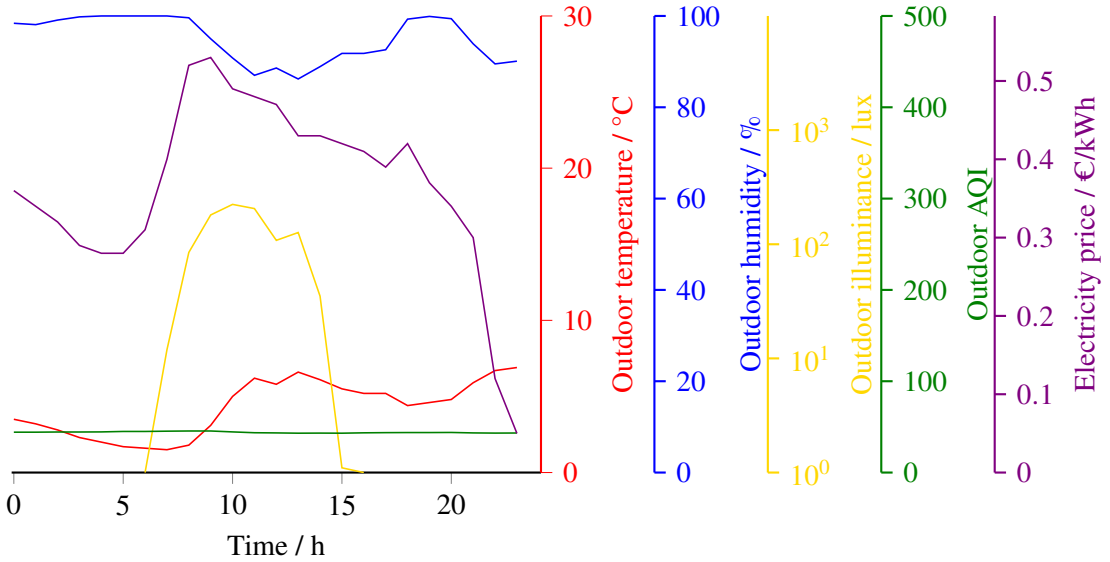


(b) Input data of 21^{st} June 2020

Figure B.1: Evaluation input data



(c) Input data of 21^{st} September 2020



(d) Input data of 21^{st} December 2020

Figure B.1: Evaluation input data

Where the x-axis indicates the course of time in hours and the y-axes the various input quantities. More precisely, the red curve shows the temperature in $°C$, the blue one the relative humidity in % and the yellow one the illuminance in lux (on a logarithmic scale). The green curve indicates the air quality according as AQI and the purple curve the energy price in €/kWh.

Bibliography

[AEA12] M. Avci, M. Erkoc, S. S. Asfour. "Residential HVAC load control strategy in real-time electricity pricing environment". In: *2012 IEEE Energytech*. Cleveland, OH, USA: IEEE, May 2012, pp. 1–6. ISBN: 978-1-4673-1834-1. DOI: 10.1109/EnergyTech.2012.6304636. URL: http://ieeexplore.ieee.org/document/6304636/ (cit. on pp. 49, 68).

[AFP+18] A. Arora, H. Fiorino, D. Pellier, M. Métivier, S. Pesty. "A review of learning planning action models". en. In: *The Knowledge Engineering Review* 33 (2018), e20. ISSN: 0269-8889, 1469-8005. DOI: 10.1017/S0269888918000188. URL: https://www.cambridge.org/core/product/identifier/S0269888918000188/type/journal_article (cit. on p. 107).

[AGA22] E. Alnazer, I. Georgievski, M. Aiello. *Risk Awareness in HTN Planning*. arXiv:2204.10669 [cs]. Apr. 2022. URL: http://arxiv.org/abs/2204.10669 (cit. on pp. 27, 28, 33, 36, 41, 47, 71).

[Aln19] E. Alnazer. "HTN Planning with Utilities". en. In: (2019). Publisher: Universität Stuttgart. DOI: 10.18419/OPUS-10661. URL: http://elib.uni-stuttgart.de/handle/11682/10678 (cit. on pp. 47, 87).

[Aut] Autonomous Systems Group at Hochschule Bonn-Rhein-Sieg. *JSHOP2 planner*. URL: https://github.com/mas-group/jshop2 (cit. on p. 94).

[AW14] C. O. Adika, L. Wang. "Smart charging and appliance scheduling approaches to demand side management". en. In: *International Journal of Electrical Power & Energy Systems* 57 (May 2014), pp. 232–240. ISSN: 01420615. DOI: 10.1016/j.ijepes.2013.12.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0142061513005206 (cit. on pp. 44, 64).

[AW15] H. Alexandria, B. Woodcroft. *Pneumatica: The Pneumatics of Hero of Alexandria*. CreateSpace Independent Publishing Platform, 2015. ISBN: 978-1-5197-2900-2. URL: https://books.google.de/books?id=c8oNjwEACAAJ (cit. on p. 25).

[BAH19] P. Bercher, R. Alford, D. Höller. "A Survey on Hierarchical Planning – One Abstract Idea, Many Concrete Realizations". en. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. Macao, China: International Joint Conferences on Artificial Intelligence Organization, Aug. 2019, pp. 6267–6275. ISBN: 978-0-9992411-4-1. DOI: 10.24963/ijcai.2019/875. URL: https://www.ijcai.org/proceedings/2019/875 (cit. on pp. 35, 36, 45).

[BB] BNA, Bundesnetzagentur | SMARD.de. *Marktdaten - Großhandelspreise Deutschland 2020*. URL: https://www.smard.de/home/downloadcenter/download-marktdaten/ (cit. on p. 94).

[Bel88] D. E. Bell. “One-Switch Utility Functions and a Measure of Risk”. en. In: *Management Science* 34.12 (Dec. 1988), pp. 1416–1424. ISSN: 0025-1909, 1526-5501. DOI: 10.1287/mnsc.34.12.1416. URL: https://pubsonline.informs.org/doi/10.1287/mnsc.34.12.1416 (cit. on p. 33).

[Ber] P. Bercher. *Hierarchical Planning Systems*. URL: https://hierarchical-task.net/software/planning-systems (cit. on p. 45).

[BMB14] A. Buckman, M. Mayfield, S. B.M. Beck. “What is a Smart Building?” en. In: *Smart and Sustainable Built Environment* 3.2 (Sept. 2014), pp. 92–109. ISSN: 2046-6099. DOI: 10.1108/SASBE-01-2014-0003. URL: https://www.emerald.com/insight/content/doi/10.1108/SASBE-01-2014-0003/full/html (cit. on p. 26).

[BMS10] M. Bauer, P. Mösle, M. Schwarz. *Green Building: Guidebook for Sustainable Architecture*. en. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. ISBN: 978-3-642-00635-7. DOI: 10.1007/978-3-642-00635-7. URL: https://link.springer.com/10.1007/978-3-642-00635-7 (cit. on pp. 43, 44, 53, 94).

[Bre20] Breeze Technologies UG. *Calculating an actionable indoor air quality index*. 2020. URL: https://www.breeze-technologies.de/blog/calculating-an-actionable-indoor-air-quality-index/ (cit. on p. 110).

[cg12] co2online, gemeinnützige Beratungsgesellschaft mbH. *How much CO2 does a human exhale?* 2012. URL: https://www.co2online.de/service/klima-orakel/beitrag/wie-viel-co2-atmet-der-mensch-aus-8518/ (cit. on p. 110).

[Cle20a] D. Clements-Croome. *Designing buildings for people: sustainable liveable architecture*. Ramsbury: The Crowood Press, 2020. ISBN: 978-1-78500-709-5 (cit. on pp. 43, 45, 53).

[Cle20b] D. Clements-Croome. *Intelligent Buildings: An Introduction*. eng. OCLC: 1157055828. S.l.: TAYLOR & FRANCIS, 2020. ISBN: 978-0-367-57632-5 (cit. on pp. 25, 26, 43, 52).

[CW13] G. Cerbe, G. Wilhelms. *Technische Thermodynamik: theoretische Grundlagen und praktische Anwendungen ; mit 40 Tafeln, 135 Beispielen, 138 Aufgaben und 182 Kontrollfragen*. ger. 17., überarb. Aufl. München: Hanser, 2013. ISBN: 978-3-446-43638-1 (cit. on p. 25).

[DD07] DIN, Deutsches Institut für Normung. *DIN EN ISO 7730 Berichtigung 1:2007-06, Ergonomie der thermischen Umgebung - Analytische Bestimmung und Interpretation der thermischen Behaglichkeit durch Berechnung des PMV- und des PPD-Indexes und Kriterien der lokalen thermischen Behaglichkeit (ISO_7730:2005); Deutsche Fassung EN ISO 7730:2005, Berichtigungen zu DIN EN ISO 7730:2006-05*. Norm. 2007. DOI: 10.31030/9852496. URL: https://www.beuth.de/de/-/-/98624598 (cit. on p. 44).

[DD12] DIN, Deutsches Institut für Normung. *DIN EN 15251:2012-12, Eingangsparameter für das Raumklima zur Auslegung und Bewertung der Energieeffizienz von Gebäuden - Raumluftqualität, Temperatur, Licht und Akustik; Deutsche Fassung EN 15251:2007*. Norm. 2012. DOI: 10.31030/1912934. URL: https://www.beuth.de/de/-/-/155677389 (cit. on p. 44).

[DD19] DIN, Deutsches Institut für Normung. *DIN 1946-6:2019-12, Raumlufttechnik - Teil 6: Lüftung von Wohnungen - Allgemeine Anforderungen, Anforderungen an die Auslegung, Ausführung, Inbetriebnahme und Übergabe sowie Instandhaltung*. Norm. 2019. DOI: 10.31030/3113944. URL: https://www.beuth.de/de/-/-/314483915 (cit. on p. 44).

[DD21] DIN, Deutsches Institut für Normung. *DIN EN 12464-1:2021-11, Licht und Beleuchtung - Beleuchtung von Arbeitsstätten - Teil 1: Arbeitsstätten in Innenräumen; Deutsche Fassung EN 12464-1:2021*. Norm. 2021. DOI: 10.31030/3233193. URL: https://www.beuth.de/de/-/-/334047306 (cit. on pp. 44, 110).

[DPR21] M. Diehl, C. Paxton, K. Ramirez-Amaro. "Automated Generation of Robotic Planning Domains from Observations". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Prague, Czech Republic: IEEE, Sept. 2021, pp. 6732–6738. ISBN: 978-1-66541-714-3. DOI: 10.1109/IROS51168.2021.9636781. URL: https://ieeexplore.ieee.org/document/9636781/ (cit. on p. 107).

[DSS19] S. Dukkipati, V. Sankar, P. Srinivasa Varma. "Forecasting of Solar Irradiance using Probability Distributions for a PV System: A Case Study". In: *International Journal of Renewable Energy Research* v9i2 (2019). ISSN: 13090127. DOI: 10.20508/ijrer.v9i2.9216.g7643. URL: https://www.ijrer.org/ijrer/index.php/ijrer/article/view/9216 (cit. on pp. 44, 61).

[FSG+12] M. Frontczak, S. Schiavon, J. Goins, E. Arens, H. Zhang, P. Wargocki. "Quantitative relationships between occupant satisfaction and satisfaction aspects of indoor environmental quality and building design: Indoor environmental quality". en. In: *Indoor Air* 22.2 (Apr. 2012), pp. 119–131. ISSN: 09056947. DOI: 10.1111/j.1600-0668.2011.00745.x. URL: https://onlinelibrary.wiley.com/doi/10.1111/j.1600-0668.2011.00745.x (cit. on pp. 45, 59).

[GA14] I. Georgievski, M. Aiello. "An Overview of Hierarchical Task Network Planning". In: (2014). Publisher: arXiv Version Number: 1. DOI: 10.48550/ARXIV.1403.7426. URL: https://arxiv.org/abs/1403.7426 (cit. on p. 46).

[GA15] I. Georgievski, M. Aiello. "HTN planning: Overview, comparison, and beyond". en. In: *Artificial Intelligence* 222 (May 2015), pp. 124–156. ISSN: 00043702. DOI: 10.1016/j.artint.2015.02.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0004370215000247 (cit. on pp. 36, 45).

[GA17] I. Georgievski, M. Aiello. "Automated Planning for Ubiquitous Computing". en. In: *ACM Computing Surveys* 49.4 (Dec. 2017), pp. 1–46. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3004294. URL: https://dl.acm.org/doi/10.1145/3004294 (cit. on pp. 47, 48, 51, 55, 56, 93).

[GDP+12] I. Georgievski, V. Degeler, G. A. Pagani, T. A. Nguyen, A. Lazovik, M. Aiello. "Optimizing Energy Costs for Offices Connected to the Smart Grid". In: *IEEE Transactions on Smart Grid* 3.4 (Dec. 2012), pp. 2273–2285. ISSN: 1949-3053, 1949-3061. DOI: 10.1109/TSG.2012.2218666. URL: http://ieeexplore.ieee.org/document/6377248/ (cit. on p. 48).

[Geo15] I. Georgievski. “Coordinating services embedded everywhere via hierarchical planning”. Thesis. University of Groningen, 2015. URL: https://www.semanticscholar.org/paper/Coordinating-services-embedded-everywhere-via-Georgievski/5c7405bcf5ddc554a18b973c6c51ae5655521a80 (cit. on p. 104).

[GHTT22] F. Geißer, P. Haslum, S. Thiébaux, F. Trevizan. “Admissible Heuristics for Multi-Objective Planning”. In: *Proceedings of the International Conference on Automated Planning and Scheduling* 32 (June 2022), pp. 100–109. ISSN: 2334-0843, 2334-0835. DOI: 10.1609/icaps.v32i1.19790. URL: https://ojs.aaai.org/index.php/ICAPS/article/view/19790 (cit. on pp. 40, 46, 71).

[GKW+98] M. Ghallab, C. Knoblock, D. Wilkins, A. Barrett, D. Christianson, M. Friedman, C. Kwok, K. Golden, S. Penberthy, D. Smith, Y. Sun, D. Weld. “PDDL - The Planning Domain Definition Language”. In: (Aug. 1998) (cit. on p. 45).

[Gre] A. Green. *The AI Planning and PDDL Wiki - Reference*. URL: https://planning.wiki/ (cit. on p. 45).

[Hal22] A. Halacheva. “HTN planning under uncertainty for sustainable buildings”. Englisch. http://dx.doi.org/10.18419/opus-12488. Bachelor’s Thesis. Stuttgart: Stuttgart, 2022. URL: http://elib.uni-stuttgart.de/handle/11682/12507 (cit. on pp. 48, 49).

[HBB+20] D. Höller, G. Behnke, P. Bercher, S. Biundo, H. Fiorino, D. Pellier, R. Alford. “HDDL: An Extension to PDDL for Expressing Hierarchical Planning Problems”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.06 (Apr. 2020), pp. 9883–9891. ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v34i06.6542. URL: https://ojs.aaai.org/index.php/AAAI/article/view/6542 (cit. on p. 45).

[HBBB20] D. Höller, P. Bercher, G. Behnke, S. Biundo. “HTN Planning as Heuristic Progression Search”. In: *Journal of Artificial Intelligence Research* 67 (Apr. 2020), pp. 835–880. ISSN: 1076-9757. DOI: 10.1613/jair.1.11282. URL: http://jair.org/index.php/jair/article/view/11282 (cit. on p. 46).

[II14] IPCC, Intergovernmental Panel on Climate Change, eds. *Climate change 2014: mitigation of climate change: Working Group III contribution to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. OCLC: ocn892580682. New York, NY: Cambridge University Press, 2014. ISBN: 978-1-107-05821-7 (cit. on p. 23).

[II22] IEA, International Energy Agency. *Energy Systems - Buildings*. 2022. URL: https://www.iea.org/reports/buildings (cit. on pp. 21, 23).

[IISN18] H. Ishibuchi, R. Imada, Y. Setoguchi, Y. Nojima. “How to Specify a Reference Point in Hypervolume Calculation for Fair Performance Comparison”. en. In: *Evolutionary Computation* 26.3 (Sept. 2018), pp. 411–440. ISSN: 1063-6560, 1530-9304. DOI: 10.1162/evco_a_00226. URL: https://direct.mit.edu/evco/article/26/3/411-440/1072 (cit. on p. 104).

[Ilg06] O. Ilghami. “Documentation for JSHOP2”. In: (Apr. 2006) (cit. on p. 74).

[JKU+13] N. Javaid, I. Khan, M. Ullah, A. Mahmood, M. Farooq. "A Survey of Home Energy Management Systems in Future Smart Grid Communications". In: *2013 Eighth International Conference on Broadband and Wireless Computing, Communication and Applications*. Compiegne, France: IEEE, Oct. 2013, pp. 459–464. ISBN: 978-0-7695-5093-0. DOI: 10.1109/BWCCA.2013.80. URL: http://ieeexplore.ieee.org/document/6690929/ (cit. on pp. 43, 68).

[Kah11] D. Kahneman. *Thinking, fast and slow.* Thinking, fast and slow. Pages: 499. New York, NY, US: Farrar, Straus and Giroux, 2011. ISBN: 978-1-4299-6935-2 (cit. on pp. 31, 38).

[Kni21] F. H. Knight. "Risk, Uncertainty and Profit". PhD thesis. University of Illinois, 1921. URL: https://ssrn.com/abstract=1496192 (cit. on pp. 27, 28).

[KSTT84] N. KANO, N. SERAKU, F. TAKAHASHI, S.-i. TSUJI. "Attractive Quality and Must-Be Quality". In: *Journal of The Japanese Society for Quality Control* 14.2 (1984), pp. 147–156. DOI: 10.20684/quality.14.2_147 (cit. on pp. 33, 34).

[LK20] G. Lauckner, J. Krimmling. "Grundlagen der Gebäudeautomation". de. In: *Raum- und Gebäudeautomation für Architekten und Ingenieure*. Wiesbaden: Springer Fachmedien Wiesbaden, 2020, pp. 5–93. ISBN: 978-3-658-30142-2. DOI: 10.1007/978-3-658-30143-9_2. URL: http://link.springer.com/10.1007/978-3-658-30143-9_2 (cit. on pp. 44, 53).

[LL20] M. Lu, J. Lai. "Review on carbon emissions of commercial buildings". en. In: *Renewable and Sustainable Energy Reviews* 119 (Mar. 2020), p. 109545. ISSN: 13640321. DOI: 10.1016/j.rser.2019.109545. URL: https://linkinghub.elsevier.com/retrieve/pii/S1364032119307531 (cit. on pp. 44, 93).

[MA04] R. Marler, J. Arora. "Survey of multi-objective optimization methods for engineering". In: *Structural and Multidisciplinary Optimization* 26.6 (Apr. 2004), pp. 369–395. ISSN: 1615-147X, 1615-1488. DOI: 10.1007/s00158-003-0368-6. URL: http://link.springer.com/10.1007/s00158-003-0368-6 (cit. on p. 41).

[Max] Max-Planck-Institut für Biogeochemie, Jena. *Jena Weather Dataset*. URL: https://www.kaggle.com/datasets/harishedison/jena-weather-dataset (cit. on p. 94).

[MN07] R. Marek, K. Nitsche. *Praxis der Wärmeübertragung: Grundlagen, Anwendungen, Übungsaufgaben ; mit 48 Tabellen, 50 vollständig durchgerechneten Beispielen sowie 93 Übungsaufgaben mit Lösungen*. ger. München: Fachbuchverl. Leipzig im C. Hanser Verl, 2007. ISBN: 978-3-446-40999-6 (cit. on p. 24).

[Mom12] J. Momoh. *Smart Grid: Fundamentals of Design and Analysis*. Hoboken, NJ, USA: John Wiley & Sons, Inc., Mar. 2012. ISBN: 978-1-118-15611-7. DOI: 10.1002/9781118156117. URL: http://doi.wiley.com/10.1002/9781118156117 (cit. on pp. 27, 44).

[NM44] J. von Neumann, O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944. ISBN: 978-0-691-13061-3. URL: http://www.jstor.org/stable/j.ctt1r2gkx (cit. on pp. 28, 29, 33).

[OL17] B. D. Olaszi, J. Ladanyi. “Comparison of different discharge strategies of grid-connected residential PV systems with energy storage in perspective of optimal battery energy storage system sizing”. en. In: *Renewable and Sustainable Energy Reviews* 75 (Aug. 2017), pp. 710–718. ISSN: 13640321. DOI: 10.1016/j.rser.2016.11.046. URL: https://linkinghub.elsevier.com/retrieve/pii/S1364032116308498 (cit. on p. 110).

[PIH+23] L. M. Pang, H. Ishibuchi, L. He, K. Shang, L. Chen. “Hypervolume-Based Cooperative Coevolution with Two Reference Points for Multi-Objective Optimization”. In: *IEEE Transactions on Evolutionary Computation* (2023), pp. 1–1. ISSN: 1089-778X, 1089-778X, 1941-0026. DOI: 10.1109/TEVC.2023.3287399. URL: https://ieeexplore.ieee.org/document/10155313/ (cit. on p. 104).

[PMZ21] H. Panamtash, S. Mahdavi, Q. Zhou. “Probabilistic Solar Power Forecasting: A Review and Comparison”. In: *2020 52nd North American Power Symposium (NAPS)*. Tempe, AZ, USA: IEEE, Apr. 2021, pp. 1–6. ISBN: 978-1-72818-192-9. DOI: 10.1109/NAPS50074.2021.9449746. URL: https://ieeexplore.ieee.org/document/9449746/ (cit. on pp. 44, 61).

[Ric18] F. Richter. “Hierarchical planning under uncertainty”. en. PhD thesis. Universität Ulm, 2018. DOI: 10.18725/OPARU-5243. URL: https://oparu.uni-ulm.de/xmlui/handle/123456789/5300 (cit. on p. 47).

[RN21] S. J. Russell, P. Norvig. *Artificial intelligence: a modern approach*. Fourth edition. Pearson series in artificial intelligence. Hoboken: Pearson, 2021. ISBN: 978-0-13-461099-3 (cit. on pp. 28–30, 35, 38, 39).

[RW19] S. Russell, E. H. Wefald. *Do the Right Thing - Studies in Limited Rationality*. eng. OCLC: 1191040719. Cambridge: The MIT Press, 2019. ISBN: 978-0-262-28277-2 (cit. on p. 30).

[Sav54] L. J. Savage. *The foundations of statistics*. The foundations of statistics. Pages: xv, 294. Oxford, England: John Wiley & Sons, 1954 (cit. on p. 30).

[SCJ17] A. Sani Hassan, L. Cipcigan, N. Jenkins. “Optimal battery storage operation for PV systems with tariff incentives”. en. In: *Applied Energy* 203 (Oct. 2017), pp. 422–441. ISSN: 03062619. DOI: 10.1016/j.apenergy.2017.06.043. URL: https://linkinghub.elsevier.com/retrieve/pii/S030626191730778X (cit. on p. 110).

[Sei06] E. Seidl, ed. *Lexikon der Bautypen: Funktionen und Formen der Architektur*. Stuttgart: P. Reclam, 2006. ISBN: 978-3-15-010572-6 (cit. on p. 23).

[Sim56] H. A. Simon. “Rational choice and the structure of the environment.” en. In: *Psychological Review* 63.2 (1956), pp. 129–138. ISSN: 1939-1471, 0033-295X. DOI: 10.1037/h0042769. URL: http://doi.apa.org/getdoi.cfm?doi=10.1037/h0042769 (cit. on pp. 33, 38).

[SM09] S. Sohrabi, S. A. McIlraith. “On Planning with Preferences in HTN”. In: (2009). Publisher: arXiv Version Number: 1. DOI: 10.48550/ARXIV.0909.0682. URL: https://arxiv.org/abs/0909.0682 (cit. on p. 46).

[SP16] C. B. Smith, K. E. Parmenter. *Energy management principles: applications, benefits, savings*. eng. Second edition. OCLC: 932016669. Amsterdam: Elsevier, 2016. ISBN: 978-0-12-802644-1 (cit. on p. 43).

[SW06] J. E. Smith, R. L. Winkler. "The Optimizer's Curse: Skepticism and Postdecision Surprise in Decision Analysis". en. In: *Management Science* 52.3 (Mar. 2006), pp. 311–322. ISSN: 0025-1909, 1526-5501. DOI: 10.1287/mnsc.1050.0451. URL: http://pubsonline.informs.org/doi/10.1287/mnsc.1050.0451 (cit. on pp. 30, 62).

[VLN08] S. Van Den Elshout, K. Léger, F. Nussio. "Comparing urban air quality in Europe in real time". en. In: *Environment International* 34.5 (July 2008), pp. 720–726. ISSN: 01604120. DOI: 10.1016/j.envint.2007.12.011. URL: https://linkinghub.elsevier.com/retrieve/pii/S0160412007002462 (cit. on p. 44).

[VV15] VDI, Verein Deutscher Ingenieure e.V. *VDI 2078:2015-06 Berechnung der thermischen Lasten und Raumtemperaturen (Auslegung Kühllast und Jahressimulation)*. Guideline. 2015. URL: https://www.vdi.de/richtlinien/details/vdi-2078-berechnung-der-thermischen-lasten-und-raumtemperaturen-auslegung-kuehllast-und-jahressimulation-1 (cit. on p. 110).

[Win18] S. G. Winter. "Satisficing". en. In: *The Palgrave Encyclopedia of Strategic Management*. Ed. by M. Augier, D. J. Teece. London: Palgrave Macmillan UK, 2018, pp. 1509–1513. ISBN: 978-0-230-53721-7. DOI: 10.1057/978-1-137-00772-8_594. URL: http://link.springer.com/10.1057/978-1-137-00772-8_594 (cit. on p. 33).

All links were last followed on 5^{th} September 2023.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

---

place, date, signature