IMPROVING PRIORITY-AWARENESS OF NON-FUNCTIONAL REQUIREMENTS DURING DECISION-MAKING IN SELF-ADAPTIVE SYSTEMS

HUMA SAMIN

Doctor of Philosophy

ASTON UNIVERSITY

July, 2022

Abstract

Self-adaptive systems (SASs) exhibit autonomous decision-making to deal with uncertainty in their operating environments. A fundamental problem with SASs is to ensure that their requirements remain satisfied as they adapt. Trade-off analysis of the non-functional requirements (NFRs), based on their satisfaction priorities, is a key to establishing a balance among them. Such trade-off analysis is often based on optimization techniques comprising decision analysis and utility theory. A problem with these techniques is that they use a single-scalar utility value to specify a combined priority for all the NFRs. Nevertheless, this combined priority does not give any information about the impacts of the environmental contexts on the individual priorities of NFRs. Moreover, these separate NFR priorities may change according to the runtime environmental contexts. Therefore, there is a need to have an approach that supports the runtime, autonomous reasoning with the distinct priorities of NFRs during the decision-making process. This PhD thesis addresses this problem by presenting Pri-AwaRE, a self-adaptive architecture for decision-making in SASs. The approach uses Multi-Reward Partially Observable Markov Decision Process (MR-POMDP) as a runtime specification model to support the modelling of the individual NFRs' priorities. The MR-POMDP model also provides runtime reasoning and autonomous tuning of these separate priorities. Therefore, it underpins priority-aware decisions. The approach has been evaluated using two substantial case studies from the different networking domains. A comparison with other state-of-the-art approaches has also been carried out. The results have shown that the priority-aware decisions offered by Pri-AwaRE provide compliance with the requirements for both the case studies even under the changing environmental contexts at runtime.

Keywords: Runtime models, Non-Functional Requirements, Priorities, Autonomous Tuning, Multi-Reward POMDPs.

Declaration

I declare that this thesis has been written by myself and presents my own work developed for my PhD in Computer Science at Aston University. The work reported in this thesis has not been previously submitted in any form for another degree.

The Pri-AwaRE approach based on the MR-POMDP model has been designed and developed by myself. The implementation of Pri-AwaRE makes use of an MR-POMDP solver known as Optimistic Linear Support with Alpha Reuse (OLSAR) developed by the researchers at the Department of Computer Science, University of Oxford. For evaluation of the approach, two case studies based on Remote Data Mirroring (RDM) and Internet of Things (IoT) networks have been used. For the RDM case, the simulation tool of RDMSim has been designed and implemented by myself to support the idea and evaluation of the approach presented in this thesis. For the IoT case, an existing well-known exemplar tool of DELTA-IoT has been used. The DELTA-IoT is an open-source exemplar which has been developed by the researchers at Katholieke Universiteit (KU) Leuven, Linnaeus University, Belgium.

Nevertheless, this work does build on the ideas proposed by the POMDP-based model called RE-STORM and has benefited considerably from the Software Engineering at Aston (SEA) Research Group at Aston University. Hopefully, this thesis has contributed to the progress and expansion of the vision of the SEA Research Group and the department.

To my family for their endless love, support and encouragement.

Acknowledgements

I would like to acknowledge the contributions of many people, including friends, family and colleagues, who have helped me to go through this journey, and to whom I am sincerely grateful.

First, I would like to express my gratitude to my supervisors; Prof. Peter Sawyer, Dr Nelly Bencomo and Dr Paul Grace, for their guidance and the precious time they invested in me throughout this journey. Second, I would like to thank my lab fellows and friends for all the fun and fruitful discussions. A special thanks to Luis H. G. Paucar, Juan M. P. Ullauri, Owen Reynolds, Sanobar Dar, Sima Iranmanesh and Jodie Ashford.

Also, I would like to thank my qualifying report examiners, Dr Antonio Garcia-Dominguez and Dr Felipe Campelo for their time and helpful advice. I would also like to thank my PhD examiners Prof. Aniko Ekart and Prof. Arosha K. Bandara for their insightful feedback and advice.

Finally, I would like to thank my family, especially my sister Samia for constantly being my support throughout this journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context and Motivation

Self-adaptive systems (SASs) represent systems that are required to perform adaptation decisions under uncertain environmental contexts [86, 93, 139, 165]. The goal is to ensure that their requirements remain satisfied as they adapt. The unforeseen changes in the environment affect the satisfaction of non-functional requirements (NFRs). Therefore, to meet the required NFR satisfaction levels[1], these adaptation decisions typically involve trade-offs between the NFRs under the different environmental contexts at runtime. Let us consider an example of a self-adaptive Internet of Things (IoT) network [7, 173] to provide motivation for the research:

1.1.1 Motivating Example

IoT refers to the networked interconnection of small computing systems (also known as nodes) comprising of different hardware and software components such as Radio-Frequency Identification (RFID) tags, sensors, actuators, mobile phones etc. These nodes interact with each other to achieve the target functional goal of transmission of information within the network. Due to the limitations of size and operational costs, the nodes in the IoT network have to deal with limitations of computational storage and energy resources. Hence, the IoT system is required to increase the lifetime of the network by minimizing the energy consumption of the nodes. Moreover, it is also required to improve the packet delivery

[1]Satisfaction level of an NFR refers to the degree to which it is satisfied.

performance under the uncertain environmental contexts of dynamic interference levels[2] and packet traffic loads on the links. These environmental contexts have different effects on the satisfaction of individual NFRs such as Minimization of Energy Consumption (MinEC) and Maximization of Packet Delivery Performance (MaxP). The higher the traffic load, the lower MinEC will be, and higher link interference would lead to lower satisfaction of MaxP [90, 106, 143]. As the environmental factors affect the satisfaction of NFRs, it requires the SAS to make a trade-off between NFRs based on their individual priorities for satisfaction under different contexts. Hence, the SAS is required to perform decision-making that takes into account the individual NFRs' priorities.

1.1.2 Problem Statement

A number of specification models have been developed that specify the decision-making process based on the NFR trade-offs, and involve alternate adaptation actions and NFRs' priorities [64, 93]. Suppose these specification models, developed at design time, can also be used and updated at runtime. In such a case, a SAS can be made requirements-aware [139, 164] by monitoring its compliance with the requirements at runtime. Analysts derive the specification model ($\boldsymbol{S}$) from the requirements and the knowledge domain ($\boldsymbol{K}$). Monitoring compliance with the requirements ($\boldsymbol{R}$), according to Zave and Jackson [176], can be done by compliance with the specification model ($\boldsymbol{S}$) as follows:

$$S, K \vdash R \tag{1.1}$$

Hence, based on equation 1.1, we can argue that $\boldsymbol{S}$ will remain a valid implementation of $\boldsymbol{R}$ if $\boldsymbol{K}$ does not change from the moment $\boldsymbol{S}$ was built until the compliance with the requirements is assessed. However, as $\boldsymbol{K}$ is subject to change, there is uncertainty about $\boldsymbol{K}$ in a SAS [33]. Hence, the SAS may be required to learn about its environment and thereby, reduce $\boldsymbol{K}$'s uncertainty.

Several runtime optimisation approaches have been developed to support the decision-making process specified in $\boldsymbol{S}$ of SASs [3, 18, 28, 51, 57, 103, 167]. These approaches are typically based on optimisation methods, including decision analysis and utility theory. Such methods involve selecting an adaptation action yielding the highest utility value from

[2]Link interference refers to constraints that affect the signal quality typically due to noise or unwanted disturbances in the electric signals.

a set of alternatives.

A problem with these approaches is that they usually use single-objective optimisation techniques. These single-objective techniques use a single scalar cumulative utility value to specify a combined cardinal priority for all the NFRs [129, 149]. However, the adaptation decisions performed by a SAS can have varied impacts, either positive or negative, on the satisfaction levels of individual NFRs [94, 149, 167]. For example, in an IoT system, the decision to increase transmission power on the links under the context of high interference will positively impact the packet delivery performance. However, it will negatively impact energy consumption [173]. The single-objective optimization approaches, using a single combined priority for NFRs, do not give any information about these distinct impacts of adaptations on the satisfaction levels of individual NFRs. Furthermore, these adaptation impacts may change based on the evolution of the SAS over time, leading to the evolution of the adaptation strategies by the SAS. Therefore, the priorities assigned at design time may not be valid at runtime based on the newly found contexts which in turn may result in violation of an NFR.

Based on the above, the limitations of the existing optimisation techniques are as follows:

(i) the treatment of the NFRs' priorities is done as a single combined value which does not provide information about the distinct impacts of adaptations on the satisfaction levels of individual NFRs, and

(ii) the assigned NFRs' priorities are considered to be fixed (i.e. they do not change).

The point to be argued in this thesis is that adaptation decisions need to be aware of the individual NFRs' priorities to perform better trade-offs. This is something that cannot be accomplished if the priorities are aggregated into a single combined value. Nevertheless, it may prove that the priorities assigned at design time may not be appropriate under certain situations. Hence, the priorities may need to be reassessed, updated and informed by the knowledge attained by the SAS encountering such situations. Therefore, *priority-awareness* can be defined as:

Definition 1. *Priority-awareness is the capability of providing autonomous changes of NFRs' priorities to address the required satisfaction levels of NFRs.*

Considering the above, the key challenges for the research are described in the next

subsection as follows:

1.1.3 Key Challenges

According to Zave and Jackson [176], compliance with the requirements (***R***) can be accomplished using a runtime specification model (***S***) equipped with the newly found knowledge (***K'***) that has an impact on changing individual priorities of NFRs.

$$S, K' \vdash R \tag{1.2}$$

The key challenges here are to have a runtime specification model that can provide compliance with the requirements by:

Challenge 1: modelling of the individual priorities of NFRs and considering those individual NFRs' priorities during the decision-making process to offer better-informed, *priority-aware* decisions.

Challenge 2: updating the NFRs' respective priorities according to the changing environmental contexts at runtime, and this update should be done in an autonomous way.

1.2 Aim and Research Questions

The overall aim of this thesis is to present an architecture supported by the specification of a runtime model that has the capability of:

1) modelling and reasoning with the individual cardinal priorities of NFRs.

2) supporting tuning of the NFRs' priorities according to the newly encountered environmental situations and acquired knowledge, while respecting their relative importance of priorities.

In this thesis, we move towards this direction by defining the following research questions:

RQ1: Can modelling and reasoning of the priorities of individual NFRs under uncertain environmental contexts be supported?

RQ2: Can decision-making in SASs include tuning of the NFRs' priorities to match the dynamic runtime situations?

The **RQ1** is related to the first research challenge and the **RQ2** is related to the second challenge.

1.3 Research Contributions

In this section, the main contributions towards addressing the research challenges are described. To address the research challenges, Pri-AwaRE, a self-adaptive decision-making architecture has been developed. Pri-AwaRE embodies a runtime specification model known as *multi-reward partially observable markov decision process plus plus* (MR-POMDP ++) that works as part of a Monitor, Analyze, Plan and Execute over the Knowledge base (MAPE-K) feedback loop [79]. MR-POMDP++ is based on the multi-objective reinforcement learning technique and is used to support:

Contribution 1: Modelling and reasoning with priorities of individual NFRs which is part of the first research challenge.

Contribution 2: Tuning of NFRs' priorities to better match the newly discovered environmental situations and acquired knowledge which is part of the second research challenge.

Moreover, for evaluation purposes, the approach has been applied to two substantial case studies from the Remote Data Mirroring (RDM) [78] and Internet of Things (IoT) domains [72]. For the RDM case study, **RDMSim** [137], a new simulation paradigm to support decision-making in SASs, has been developed as part of the research project presented in this thesis.

List of Publications

Following is the list of publications that arose from this research:

1. H. Samin, "Priority-awareness of non-functional requirements under uncertainty," in

2020 IEEE 28th International Requirements Engineering Conference (RE), Doctoral Symposium Track, IEEE, 2020.

2. H. Samin, L. Garcia Paucar, B. Nelly, and P. Sawyer, "Towards priority-awareness in Autonomous Intelligent Systems," in 36th ACM/SIGAPP Symposium On Applied Computing (SAC). ACM, 2021.

3. H. Samin, Luis H. G. Paucar, Nelly Bencomo, Cesar M. Carranza Hurtado, Erik M. Fredericks, "RDMSim: An Exemplar for Evaluation and Comparison of Decision-Making Techniques for Self-Adaptation", in 16th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Artefacts Track, 2021

4. H. Samin, N. Bencomo, and P. Sawyer, "Pri-AwaRE: Tool support for priority-aware decision-making under uncertainty," in 2021 IEEE 29th International Requirements Engineering Conference (RE), Posters and Tools Demonstration Track, IEEE, 2021

5. H. Samin, N. Bencomo, and P. Sawyer, "Decision-Making under Uncertainty: Be Aware of your Priorities," in International Journal on Software and Systems Modeling (SoSyM), 2021.

1.4 Research Methodology

A number of research methods exist in Computing Science [70, 75] that are categorised as *qualitative* or *quantitative* [65]. The qualitative research method is based on the study of subjective data. In contrast, the quantitative method is based on collecting and analysing numerical data that could be structured into statistics. In Software Engineering (SE), the traditional quantitative approach is considered suitable for some research topics, for instance, where performance is a research challenge. However, there exist research topics where the empirical value of quantitative approaches is not evident. In such cases, the SE practitioners follow a qualitative approach [141].

Moreover, research in SE is mainly considered as a *synthetic* approach which focuses on making and inventing [110]. Hence, SE research involves the design and development

of abstract mechanisms to support software developers by not only improving development process but also to understand and model complex systems [151]. These proposed abstract mechanisms are useful in terms of presenting new concepts, or new relationships between existing unrelated concepts.

Considering the above, the work presented in this thesis followed a Design Science methodology [170]. Design Science is a popular problem-solving paradigm in SE and involves the design and investigation of artifacts to solve real-world problems [171]. In SE research, an artifact refers to an algorithm, a method, a technique or a conceptual framework created for some practical purpose. The Design Science framework typically involves the activity of designing the artifacts within a problem context to improve something in that context. Based on the Design Science approach, the research presented in this thesis focuses on presenting the technique of Pri-AwaRE as an artifact to improve the decision-making process in SASs with better-informed, and priority-aware decisions.

Considering the Design Cycle presented in [171], the research study followed the following three steps:

A. Problem Investigation

The first step of the design cycle involved the investigation of the problem under study. For this purpose, a systematic scientific approach of conducting a literature study of the state-of-the-art Artificial Intelligence (AI) based approaches for decision-making in SASs was carried out. In addition, the techniques that focus on prioritization of requirements during the decision-making process at runtime were also studied. After critically analysing the existing approaches, the research focused on the application of the selected approaches to support priority-aware requirements-driven decision-making in SASs.

B. Treatment Design

The treatment design involved the design of an artifact as a treatment for the problem under investigation. Therefore, the research project focused on designing and developing a self-adaptive architecture to support the runtime awareness of NFRs' priorities during decision-making.

C. Treatment Validation

The treatment validation involved demonstrating and evaluating the proposed Pri-AwaRE architecture by its application to two case studies from domains of IoT and RDM systems. Experimental evaluations also involved comparing the proposed approach with the existing state-of-the-art techniques to fulfil the proposed solution's evaluation.

1.5 Research Plan

In order to reach the aim of the research study, the following plan was followed according to the Research Methodology explained in Section 1.4:

1) To undertake a literature review of the state-of-the-art decision-making techniques in SASs for identifying the research gaps that eventually have driven the research presented in this thesis.

2) To undertake a literature review of the state-of-the-art techniques for requirements prioritization and their role in the decision-making process in SASs.

3) To evaluate and examine the state-of-the-art techniques for requirements prioritization in SASs, specifically the runtime techniques and identify the research gaps.

4) To carry out an investigation on the runtime models for self-adaption and usage of runtime models as a means to support priority-aware decision-making and reasoning under uncertainty.

5) To develop a structured technique, which is called Pri-AwaRE, to support runtime modelling and reasoning with priorities of NFRs. The technique also provides *autonomous tuning* of these priorities according to the changing environmental contexts to support decision-making in SASs.

6) To evaluate the viability and benefits of Pri-AwaRE by applying it two different real case studies. For both the case studies, the decision-making mechanism has been designed.

7) To evaluate the quality of decision-making offered by the Pri-AwaRE approach using techniques like Logistic Regression [21] and DeSiRE [44].

The research has also involved the design and development of **RDMSim** [137], a simulation tool to represent the operating environment for the case study based on RDM. The RDMSim has been designed to offer researchers an RDM environment to test and compare their decision-making techniques against other techniques. For the IoT case study, the existing simulation tool of DELTA-IoT [72] has been used.

1.6 Thesis Outline

The Thesis is organized as follows:

Chapter 2: State of the Art of Decision-Making in Self-Adaptive Systems and Requirements Prioritization describes the state-of-the-art techniques with respect to the scope of this thesis. The focus is on discussing the techniques for decision-making in SASs. In addition, the techniques dealing with the prioritization of requirements during the decision-making process are also described. This chapter motivates the approach presented in this thesis.

Chapter 3: Research Baseline presents the baseline concepts for the research presented in this thesis. First, the basic concepts for decision-making process in a SAS are described. Second, details about the runtime models are provided. In the end, the concept of Multi-objective Reinforcement Learning is described which is a baseline technique for the proposed research approach.

Chapter 4: Pri-AwaRE: A priority-aware self-adaptive architecture for decision making in Self-Adaptive Systems presents Pri-AwaRE which is a self-adaptive architecture for supporting priority-aware decision-making in SASs. The architecture makes use of MR-POMDP++ as part of MAPE-K loop to support better-informed decisions. The details about how the Pri-AwaRE supports modelling and reasoning of individual priorities of NFRs during the decision-making process are provided. The chapter also explains the procedure for autonomous tuning of NFRs' priorities during the decision-making carried out by the Pri-AwaRE.

Chapter 5: Case Studies presents the two case studies (i.e. IoT and RDM system) used for the evaluation of the proposed Pri-AwaRE approach. The chapter provides details about simulation tools used for experiments. The operational model, architecture and uncertainty scenarios representing the different dynamic environmental situations for both the simulated environments are also described.

Chapter 6: Experimental Evaluation provides the experiments for evaluation of the Pri-AwaRE approach. The experiments using both the case studies are provided. Comparisons with the state-of-the-art techniques are also presented.

Chapter 7: Validation of Results provides experimental evaluations for assessing the quality of the decision-making offered by the Pri-AwaRE approach. Comparisons with the Related Work and threats to the validity of the approach are also provided.

Chapter 8: Conclusion and Future Work presents the conclusion and future directions for the research presented in this thesis. Concluding remarks highlighting the main contributions of the research project and answers to the research questions are provided. Furthermore, the future research agenda is also presented.

Chapter 2

State of the Art of Decision-Making in Self-Adaptive Systems and Requirements Prioritization

This chapter presents the literature study on the state-of-the-art decision-making techniques in SASs. The focus is mainly on exploring the techniques that support requirements-driven adaptations. It also explores the techniques focusing on modelling and reasoning with the priorities of NFRs during the decision-making process. A number of studies exist that explore requirements prioritization [77, 118, 157] and decision-making under uncertainty [30, 48, 93] for SASs. These existing studies focus on the techniques that are applied at design time. However, to support the research challenges addressed in this thesis, there is a need to explore the state-of-the-art for requirements-driven decision-making and uncertainty resolution at runtime. The focus is specifically to explore the approaches that support the prioritization of NFRs during the decision-making process. The chapter is organized as follows: Section 2.1 discusses the techniques for decision-making and uncertainty resolution during decision-making in SASs. Section 2.2 presents the techniques dealing with requirements prioritization in SASs which is followed by Summary in Section 2.3.

2.1 Techniques for Decision-Making in Self-Adaptive Systems

This section provides a literature study of the decision-making techniques for SASs and the techniques for uncertainty resolution for SASs. In order to support the critical evaluation of the approaches, a comparison of the techniques is presented in Tables 2.1 and 2.2. The techniques and their comparisons are discussed as follows:

A number of frameworks have been presented in the literature to support the decision-making in SASs. The most notable ones of these are RELAX [169], FLAGS [11] and ZANSHIN [146], which support the notion of *requirements-driven* decision-making in SASs. The focus here is to establish the relationship between satisfaction and priorities of requirements under uncertain environmental contexts at runtime [117].

RELAX is a requirements specification language based on fuzzy logic. RELAX offers a modification in the requirements specification statements to address environmental uncertainty. For this purpose, it offers to include additional information concerning the environmental context, monitored data from the environment and their relationship. Hence, RELAX offers to facilitate the flexibility of requirements satisfaction criteria by providing relaxation of their selected priority thresholds. Based on the RELAX framework, the techniques of AutoRELAX [122] and Providentia [25] have been developed. The focus of these techniques is on the optimization of requirements' priorities by taking a KAOS[1] goal model [38] as a reference. AutoRELAX is based on the approach of genetic algorithms, an Evolutionary Computation [81], which is used to optimise the goal constraints represented in the form of a genome structure. The values of the genome represent the satisfaction values of the requirements. These satisfaction values of the requirements are optimized to get their respective utility values. These utility values represent the preferred thresholds for priorities of requirements. Providentia supports the automated optimization of functional requirements which consequently has an effect to support the satisfaction of individual NFRs in SASs. Similar to AutoRELAX, it uses genetic algorithms, to be used at design time, for quantifying the NFRs' priorities at runtime by optimizing the functional requirements (FRs). Both approaches make use of the genome structure to represent the goal constraints and their satisfaction levels as runtime model entities. However, the approaches

[1]KAOS stands for Keep All Objectives Satisfied. It is a goal-oriented requirements modelling method.

of AutoRELAX and Providentia make use of KAOS goal model as a reference to optimize the requirements (i.e. FRs and NFRs), they lack the causal connection [97] of a goal model to be used as a Model@Runtime (runtime model) to support awareness of requirements. The introduction of causal connection can help support the runtime prioritization of the requirements. Moreover, it can also support handling the dependencies in requirements during the decision-making process.

In order to treat requirements as live runtime entities, the Fuzzy Live Adaptation Goals for SASs (FLAGS) framework [11] proposes an extension of the KAOS goal model. The extended version of KAOS introduces the concept of adaptive goals. The framework differentiates between fuzzy and crisp goals. In FLAGS, crisp goal satisfaction is viewed as binary in nature. In contrast, the satisfaction of fuzzy goals (i.e. the NFRs) is presented in the form of fuzzy constraints. The adaptation strategies based on the concept of adaptive goals are defined to offer trade-offs and re-prioritization between the goals to achieve the satisfaction of goals,

Analogous to the technique of FLAGS, the ZANSHIN framework [146] presents an extension of goal models by including the monitored information, adaptive strategies and the relationships between them. The goal models are incrementally refined by the identification of indicators known as Awareness requirements [147, 152]. Awareness requirements refer to the elements that are required to be monitored along with the parameters and adaptation strategies. They represent what things need to be changed according to the newly identified environmental context. Consequently, the Awareness requirements are used to specify the satisfaction state of other requirements, leading to the re-prioritization of the parameters by applying the adaptation strategies at runtime. Neither the technique of ZANSHIN or FLAGS provide information about the distinct modelling of the NFRs' priorities.

As stated above, AutoRELAX and Providentia use genetic algorithms to model the satisfaction values of requirements but there are also techniques that employ evolutionary computing to support the decision-making in SASs. In [87, 89], Enki, a novelty search-based approach [92], an Evolutionary Computation technique, has been used along with Deep Neural Networks (DNN) to improve the decision-making process to address environmental uncertainty. The approach makes use of evolution based technique to generate synthetic data for representing different environmental situations. The situations that cause extreme and unique behaviors are then selected to train the DNN for improving decision-making.

Furthermore, ENLIL3 [88] uses evolutionary computation and machine learning to predict runtime behaviour of the SAS in advance by a priori detection of uncertain conditions. The automated approach helps the system to uncover the failure situations that are not covered by the existing training datasets, and thereby improving the decision-making process. The approach presented in [83] makes reuse of prior plans to improve the selection of adaptation tactics under unforeseen situations at runtime and as a result satisfies the quality objectives (i.e. NFRs). The decision-making planner reuses the prior knowledge by initializing the population with existing plans as a seed. It then selects the plan of action based on the candidate fitness value computed to measure the expected quality of the entire system.

In [20], a three-step adaptation approach has been presented that models the decision-making as a Multi-Objective Constrained Optimisation Problem (MOCOP). The approach performs adaptations for the satisfaction of requirements based on available resources. The first step of the approach involves the identification of resources to satisfy the requirements. The second step is to search for possible substitutions for satisfaction of requirements if the required resources are not available. And finally, the approach makes the required adaptations based on availability of resources.

The concept of Digital Twins (DT) has also been proposed to model, control and predict behaviour of the SASs. In [156], Case-Based Reasoning (CBR), an AI planning technique, has been presented to leverage the DT. It supports the DT with the domain knowledge and plan for the operations at runtime [23]. The cases in CBR represent the undesired conditions that the system responds to, and the possible reactions that the system can perform in such situations. Taking the cases as a base, the DTs are able to take decisions by finding similar cases under the detected uncertain conditions.

A number of approaches based on control theory to support decision-making have also been developed [27, 114, 115]. The focus is on combining AI-based techniques such as Evolutionary Computation, Regression Analysis and Classification techniques with control theory to offer better adaptation strategies for satisfying requirements [27, 126]. Furthermore, in [145], the presented approach makes use of a search-based approach as part of the feedback control loop to perform the runtime decision-making. The technique uses the Non-dominated Sorting Genetic Algorithm version 2 (NSGAII), an Evolutionary Computation algorithm, to select the optimal configuration for the purpose of satisfaction of the NFRs simultaneously. The approach proposed by Peng et al. [114, 115] presents a PID

controller which is used to make a trade-off among NFRs to support decision-making at runtime. However, the approach does not consider the evolution of the satisfaction level of NFRs during the course of decision-making.

Techniques based on probabilistic models such as Dynamic Decision networks (DDNs) have also been used to perform decision-making in SASs. The DDNs are used to represent goal models as Models@Runtime [18], a mapping of a goal model to a DDN [17] is presented in literature. Based on Bayesian methods and Decision Theory, DDNs provide a procedure to perform runtime decision-making [15]. The approach models the satisfaction levels of NFRs using conditional probabilities. Moreover, utility functions are used to prioritize different decisions. The approach of DDN supports requirements-awareness by providing requirements reflections at runtime but it lacks the reflection of dependencies between them. Moreover, the focus is on the satisfaction of NFRs only.

Techniques based on reinforcement learning such as Markov Decision Processes (MDPs) and Partially Observable Markov Decision Processes (POMDPs) have also been used to perform requirements driven decision-making. These techniques facilitate uncertainty quantification and support awareness of the requirements by maintaining probabilities over the states of the environment. MDPs consider the decision-making agent working in fully observable environment. In [102], the MDP model is constructed using formal methods at design time, and is later used to perform decision-making at runtime. In addition, a technique based on stochastic model checking is presented to propose a language for Stochastic Multiplayer Games and MDPs known as PRISM [31]. The language is used to create a PRISM SMG model where reasoning about the adaptation strategies is carried out. In contrast to MDPs, POMDPs consider the decision-making agent working in a partially observable environment. The POMDP model has been used to model NFRs at runtime and represents the satisfaction levels of NFRs as conditional probabilities [112]. Due to partial observability, the state of satisfaction of NFRs is not directly observable. Instead, a belief about the state of satisfaction of NFRs is maintained based on the monitored observations. Considering the current observation, and state of satisfaction of NFRs, an adaptation action is selected based on expected cumulative utility value [58].

Table 2.1: Approaches for Decision-Making in SASs

Approaches	Design Time/Runtime	Technique Used	FRs/NFRs	Uncertainty Resolution
AutoRELAX	Design Time	Genetic Algorithms	NFRs	No
Providentia	Design Time	Genetic Algorithms	FRs+NFRs	No
FLAGS	Runtime	Fuzzy Logic	FRs+NFRs	Yes
ZANSHIN	Design Time	Control Theory	NFRs	Yes
Enki	Design Time	Evolutionary Computation + DNN	None	Yes
ENLIL3	Design Time	Evolutionary Computation + DNN	None	Yes
Stochastic Search	Runtime	Genetic Programming	NFRs	Yes
Composition based Adaptation	Runtime	MOCOP	FRs+NFRs	No
Digital Twins	Runtime	Digital Twin + Case-Based Reasoning	FRs	Yes
Control Theoretic Self-Tuning	Runtime	Control Theory	NFRs	No
Probabilistic Model	Runtime	Dynamic Decision Networks	NFRs	Yes
PRISM	Runtime	Formal Analysis + MDP	NFRs	Yes
RE-STORM	Runtime	POMDP	NFRs	Yes

Table 2.2: Approaches for Uncertainty Resolution in SASs

Approaches	Technique	Uncertainty Detection	Level of Uncertainty	Uncertainty Resolution
Bayesian Surprise	Kullback Leibler (KL) Divergence	Yes	No	No
RELAX	Fuzzy Logic	No	No	No
RELAXing Claims	Fuzzy Logic	No	No	Yes
REAssure	Fuzzy Logic	No	No	Yes
POISED	Possibility Theory + Fuzzy Mathematics	Yes	No	Yes
Rainbow	Control Systems + Probablity theory	Yes	No	Yes
FUSION	Machine Learning	Yes	No	Yes
Stochastic Search	Genetic Programming	Yes	No	Yes
Enki	Evolutionary Computation	Yes	No	Yes
PRISM	Formal Analysis +MDP	Yes	No	Yes

Table 2.3: Approaches for Prioritization of Requirements

Approaches	Design Time/ Runtime	Technique Used	FRs/NFRs	Individual Priority Modelling	Autonomous Tuning
Multi-Criteria Decision-Making	Design Time	Analytic Hierarchy Process	NFRs	Yes	No
ARROW	Runtime	POMDP + P-CNP	NFRs	Yes	No
AutoRELAX	Design Time	Genetic Algorithms	NFRs	Yes	No
Providentia	Design Time	Genetic Algorithms	FRs+NFRs	Yes	No
Probabilistic Model	Runtime	Dynamic Decision Networks	NFRs	No	No
Markov based Approaches	Runtime	MDPs + DTMC	NFRs	No	No
FLAGS	Runtime	Fuzzy Logic	FRs+NFRs	**No Information**	No
ZANSHIN	Design Time	Control Theory	NFRs	**No Information**	No
Control Theoretic Self-Tuning	Runtime	Control Theory	NFRs	Yes	No

2.1.1 Uncertainty Resolution during Decision-Making

An important aspect of the decision-making in SASs is to deal with uncertain environmental contexts at runtime. Several techniques have been provided in the literature to support the resolution of uncertain environmental situations [61, 69]. These techniques vary from detecting uncertainties to their quantification for assessing and measuring uncertainty at runtime.

To measure the degree of uncertainty and deviations of the SASs from normal behaviour, the Bayesian Theory of Surprise has been presented [17, 22]. A surprise measures the impacts of the model on observed data and assumptions of the world at runtime, and variation between the two helps detect uncertainty. To provide reasoning over uncertainty in SASs, Bayesian surprise has been used along with the approaches of DDNs [13, 15] and POMDPs [57, 112]. Moreover, to support multi-criteria decision making (MCDM) processes, Bayesian surprises have been used along with multi-attribute analysis and Pareto Analysis [67]. Therefore, the usage of Bayesian surprises helps in the evaluation of the importance and impacts of uncertain events. Moreover, it can be used to calculate the extent of the impact that uncertain contexts can have on the decisions taken at runtime.

To deal with uncertainty at runtime, techniques based on fuzzy logic [82] have also been presented in the literature. RELAX, a formal requirements language, uses fuzzy logic in the requirements specification to specify uncertainty [168, 169]. Taking the semantics of RELAX as a base, a technique known as RELAXing Claims has been proposed [123]. The approach studies the impacts of uncertainty, considering problems in the monitoring infrastructure, on the validity of Claims (i.e. assumptions of the environment). Furthermore, to support the idea of RELAXing Claims, a technique called REAssuRE [164] has been presented. The approach of REAssuRE uses goal models and Claims to facilitate runtime decision-making in SASs. In order to study positive and negative effects of uncertainty on the runtime configurations of SASs, an approach called POssIbilistic SElfaDaptation (POISED) has also been proposed [49]. The POISED approach uses possibility theory and fuzzy mathematics to tackle uncertainty for improving the quality of service of SASs at runtime. Possibility theory is an extension of fuzzy logic and is used to deal with different types of uncertainties [175].

As described before, the frameworks of RELAX, FLAGS and ZANSHIN focus on ad-

dressing the environmental uncertainty in SASs. For this purpose, RELAX tries to specify uncertainty in the requirements specification; FLAGS present the concept of adaptive goals by providing an extension of goal models and ZANSHIN provides the concept of *Awareness requirements*. These approaches could be used to improve the results of techniques like AutoRELAX, Providentia and ARROW. The techniques of AutoRELAX, Providentia and ARROW specify fixed priority thresholds. However, situations might arise at runtime that may require the relaxation of the priority threshold of one requirement to support the satisfaction of another requirement. The above frameworks can help in providing such flexibility by making a SAS capable of adjusting the requirements' priorities for the purpose of resolution of uncertainty.

A technique underpinning an architecture based self-adaptive framework known as Rainbow [59] has been proposed. Rainbow augments the architecture of SASs using probabilistic models. The model tries to resolve three types of uncertainties at runtime. The first type of uncertainty is the problem-state identification which is related to the Monitoring and Analysis phases of the MAPE-K loop (described in Section 3.1.1). The running average approach is applied to deal with the variable and stochastic nature of the environment for mitigating this type of uncertainty. The second type of uncertainty is the uncertainty in the strategy selection process which is related to the planning step of the MAPE-K loop. It is mitigated with the help of Stitch language [59] that allows uncertainty modelling in strategies. The third type of uncertainty corresponds to the strategy outcome related to the Execute step of the MAPE-K loop. This type of uncertainty is resolved by specifying the time period for Rainbow to monitor strategy implementation before committing change in the adaptation loop. This feature is also modelled using Stitch language.

A technique called FUSION [46] has been presented to deal with uncertain environmental contexts. FUSION stands for FeatUre oriented Self-adaptatION. The technique uses a feature-oriented self-adaptive procedure to carry out self-adaptation for supporting functional goals and quality attributes. It provides the tuning of the adaptation logic by using the Model Tree Learning approach. The FUSION process consists of the learning cycle and the adaptation cycle. During the learning cycle, the FUSION framework gathers measurements of the observable metrics and checks if any emergent pattern is discovered. This is done as part of the observe phase of the learning cycle. Once an emergent pattern is observed, the new behaviour is learned during the induce phase of the learning cycle. This

newly learned behaviour is stored in the knowledge base for decision-making mechanisms in future cycles. The adaptation cycle consists of three phases: detect, plan, and effect. In the detect phase, the achieved utility (a measure of user-satisfaction) is computed based on the measured metrics to discover the violation of the goal. If a goal violation is recognized, the plan phase searches for the configuration of the features to achieve its satisfaction and maximization of the overall utility. Once a plan is selected, the effect phase executes the plan by applying the selected feature configuration. However, the whole process is carried out at runtime; an initial learning cycle is executed in offline mode before the system's deployment. This is done by using a simulator to train FUSION for inducing a preliminary model of the system's behaviour. Therefore, the accuracy of the results depends on the accuracy of the induced model.

A technique based on stochastic search has been presented to deal with unforeseen environmental situations at runtime [83]. The technique augments the approach of genetic programming [85], an Evolutionary Computation Technique [43], with the reuse of prior plans to deal with the environmental uncertainty. The approach detects newly found situations and plans for the resolution of uncertainty by exploitation of prior knowledge and exploration of new solution space.

For the purpose of resolving uncertainty, tool such as Enki [87] has been presented. The tool makes use of evolutionary computation to generate different environmental situations to train the decision-making model. The approach helps in detection of environmental uncertainty at design time to facilitate the designers in discovering a priori the uncertain situations and training the model in advance.

Although, the techniques of Rainbow, FUSION and Enki provide the capability of resolving uncertainty at runtime but they lack quantification of uncertainty. In contrast, to support quantification of uncertainty, the approach of Bayesian Surprise is presented. Bayesian Surprise has been used along with DDNs to detect uncertain environmental situations at runtime. However, it has not been used to measure the level of uncertainty, to indicate its extent [9, 13] and its impact on the satisfaction priorities of NFRs.

Furthermore, the techniques presented in [31, 102] makes use of formal analysis for explicit specification and reasoning about uncertainty of the environment to support the reduction of uncertainty in the decision-making offered by MDPs.

2.2 Techniques for Requirements Prioritization in Self Adaptive Systems

This section discusses different techniques dealing with prioritization of NFRs in SASs. The techniques have been selected on the basis of the criteria: they should augment the SASs with the capability of self-awareness [79], i.e. having the capability to reason about how well the NFRs are satisfied concerning their priorities for satisfaction. Comparison of the techniques is presented in Table 2.3. The techniques have been classified into two categories as follows:

2.2.1 Design Time Techniques

In order to support ranking and explicit modelling of the NFRs, the approaches based on Multi-Criteria Decision Making (MCDM) techniques, such as Analytic Hierarchy Process [94] and Primitive Cognitive Network Process (P-CNP) [174], have been proposed in the literature. Although they support explicit modelling of the NFRs and their priorities, they are more of design time techniques. Furthermore, the technique of ARROW [113], based on P-CNP, provides support to RE-STORM (using the POMDP model [57]) to deal with the prioritization of NFRs. Using P-CNP, ARROW supports automatic updates for initially defined priorities of NFRs at runtime. However, this priority update is not autonomous and does not work from within the POMDP model.

Furthermore, to support the optimization of NFRs, there are also techniques that make use of search-based approaches such as [25, 122]. These techniques optimize priorities at design time which are further used at runtime in an offline fashion. However, these techniques offer the optimization of the individual priorities of requirements but they are more design time techniques. There is a need to optimize the configuration of the requirements based on their respective priorities under the changing environmental contexts at runtime.

2.2.2 Runtime Techniques

For the purpose of supporting decision-making in SASs, several runtime modelling techniques have been in use to provide priority-awareness. Such techniques involve the usage of probabilistic models such as Dynamic Decision Networks (DDNs) [15]. The DDNs have been used to represent the goal model as a runtime model to support decision-making under

uncertainty. The DDNs, with the help of the Bayesian Surprise [18], support the quantification of uncertainty. Although, usage of DDNs provides the capability of quantification of uncertainty to support requirements-driven decision-making, it lacks modelling of the individual priorities of NFRs. The technique represents the priorities of NFRs as a single scalar utility value to specify a combined priority value for all NFRs. This scalar utility value is then used to deduce the impact (positive or negative) of the adaptation decision on the satisfaction of all NFRs at runtime. Although, usage of DDNs provides the capability of quantification of uncertainty to support requirements-driven decision-making, it lacks modelling of the individual priorities of NFRs. Furthermore, the technique based on DDNs deals with the problem of scalability over time (i.e. the curse of history, the graph to represent the history of observations and actions for the DDN planning grows exponentially with the planning horizon). In contrast, the approaches presented in [3, 28, 51, 103, 112, 167] make use of Markov based approaches. These approaches include Markov Decision Process (MDPs), Partially Observable Markov Decision Processes (POMDPs) and Discrete Time Markov Chains (DTMCs) with probabilistic model checking, and are used to support runtime assurance of NFRs during the decision-making process. As these approaches are Markov based, they provide the quantification of uncertainty by maintaining probabilities over the environment's state. A significant limitation of these approaches is that they lack explicit modelling of the individual priorities of NFRs at runtime. Moreover, the approaches based on MDPs and POMDPs model the ranking of the NFRs as a scalar-reward value to specify a cumulative priority of all the NFRs. As this single cumulative priority value does not give any information about the separate priority-value of each NFR, therefore it deters *priority-awareness*. Furthermore, all the above techniques, make use of the requirements' priorities that are defined by the requirements engineers at design time. However, the priorities assigned at design time may no longer be valid due to the emergence of the unforeseen environmental contexts at runtime. Therefore, the SAS should be capable of tuning the priorities *autonomously* based on the changing runtime contexts.

Control theory based approaches such as [98, 114], have also been used to support explicit runtime configuration and tuning of NFRs. The technique by Maggio et al. [98] lacks the autonomous prioritization of NFRs, and the approach by Peng et al. [114] can't deal with the NFRs having the same priority rank. Moreover, the technique by Peng et al. [114] does not support quantification of uncertainty to support the runtime NFRs' trade-offs.

2.3 Summary

Based on the literature study, it can be deduced that most of the techniques that support decision-making in SASs rely on the design time assignment of the priorities by the requirements engineers. The techniques also lack the autonomous tuning of the priorities of NFRs according to the runtime situations which were not foreseen by the requirements engineers at design time. However, there are approaches that facilitate the runtime modelling and awareness of the requirements, but they lack the modelling of the individual NFRs' priorities. Hence, the research presented in this thesis focuses on trying to fill these research gaps by presenting a self-adaptive architecture *Pri-AwaRE*. The proposed Pri-AwaRE architecture provides awareness of requirements by supporting modelling and reasoning with the individual priorities of NFRs. It also considers the tuning of priorities in an autonomous way based on the changing runtime contexts.

Chapter 3

Research Baseline

This chapter describes baseline concepts for the work presented in this thesis. The research baseline is presented in two parts. The first part describes the decision-making process in autonomous and self-adaptive systems and the concepts of requirements-awareness and runtime models in Section 3.1. The second part describes the concept of multi-objective reinforcement learning (MORL) and the MORL techniques: Partially Observable Markov Decision Process (POMDP) and Multi-Reward Partially Observable Markov Decision Process (MR-POMDP) in Section 3.2.

3.1 Autonomous and Self-Adaptive Systems

Autonomous and self-adaptive systems are defined as systems that are able to adjust their behavior according to the changes in the environment and within the systems. The prefix *self* means that the system autonomously decides how to adapt to changes in its environmental context (internal or external) [166]. These decisions can be taken without or with minimal human intervention. The challenge here is to satisfy certain objectives at runtime along with dealing with *uncertainty* in the environment. Uncertainty is defined as the state of the system having incomplete or inconsistent knowledge, where it is unable to realize which of the alternative system configurations to choose at a particular time [124]. In order to deal with uncertainty, a SAS is required to be *self-aware*. *Self-Awareness* means that the system should be able to predict its next possible state based on analysis of the environment [12, 29]. Moreover, it should be able to *self-reflect*. *Self-reflection* means that the system should have an awareness of its execution environment, software architecture, its functional operational

goals and non-functional quality requirements (NFRs) [37, 80]. Based on this awareness, the system should perform adaptive decisions to continuously meet it operational goals. Considering the definitions of *self-awareness* and *self-reflection*, in this thesis the concept of the *priority-awareness* is introduced. Priority-awareness means that the system should be aware of the NFRs' satisfaction priorities under the changing environmental contexts and should be capable of tuning these priorities to perform better-informed adaptive decisions to satisfy its requirements. Moreover, the focus of the work presented in this thesis is to deal with external environmental uncertainty.

3.1.1 Decision-Making in Autonomous and Self-Adaptive Systems

The main challenge for the decision-making process in SASs is to deal with different sources of environmental uncertainty along with balancing of the objectives of the system. Different environmental contexts have different effects on the satisfaction of the NFRs. Based on the observations of its environment, the SAS makes a decision for the adaptation action to be performed and acts upon it. As a result, these adaptation actions can have different impacts on the satisfaction levels of NFRs.

Considering the above, the decision-making process of a SAS involves the following main concepts [15]:

1) NFRs: The primary focus of decision-making in a SAS is to satisfy its NFRs while achieving its functional operational goals [18]. The NFRs exhibit two main properties at runtime [64] as follows:

- **Satisfaction Level:** The satisfaction level of an NFR refers to the degree to which it is satisfied. During the decision-making process in SASs, an NFR's satisfaction level at a particular point in time corresponds to the extent to which it is satisfied as a result of an adaptation performed by the system, and its level of satisfaction at the previous execution time step. The satisfaction level can be represented by a conditional probability distribution i.e. P(NFRi+1 is satisfied | action a, NFRi)[15, 112].

- **Priority:** The priorities of NFRs refer to the mapping of the NFRs on a level of importance for the stakeholders [116]. They are used to ensure that finite resources are used optimally. During the decision-making process in SASs, the priority is defined as a scalar cardinal value representing the importance or significance of an NFR in terms of its satis-

faction at runtime. The NFRs' priorities, initially at least, are assigned by the developers at design time. Moreover, the priority value of an NFR may change due to the uncertain environmental changes at runtime.

2) Monitorables: At the time of execution, the SAS is required to continuously monitor the environment for changes. Monitorables refer to the observed information regarding the state of the environment. For example, in a computer network, traffic load and interference on the network links can be monitored at runtime.

3) Actions: Actions refer to the discrete set of software configurations, solutions or service components that are chosen by the SAS to fulfill its functional goals and satisfy NFRs [17, 136]. The adaptation actions are selected based on the monitorable values, and the priorities and satisfaction levels of NFRs at a particular point of time [112]. Once the action is performed by the SAS, it has an impact (positive or negative) on the NFRs' satisfaction. For example, in a computer network, based on the monitored interference on the network links, adaptation action of increasing or decreasing the transmission power are selected to improve the satisfaction of packet delivery performance.

4) MAPE-K Architecture

The MAPE-K is an architecture pattern for autonomous systems. It was introduced for the first time by IBM as a vision of Autonomic Computing [79]. The architecture pattern comprises of the two main components which are i) the managed system and ii) the managing system as presented in Fig. 3.1. The managed system refers to the application logic. On the other hand, the managing system refers to the decision-making agent which executes a feedback control. The feedback control loop has four functional phases: *Monitor*, *Analyze*, *Plan* and *Execute*. All of these phases share common *Knowledge*. Hence, this architectural control loop is known as MAPE-K loop in short. The phases of the MAPE-K loop are described as follows:

a) Monitor: During the *Monitor* phase, the managing system observes and collects data (i.e. monitorables) from the managed system and its environment. This collection of data is done using sensors connected to the managed system.

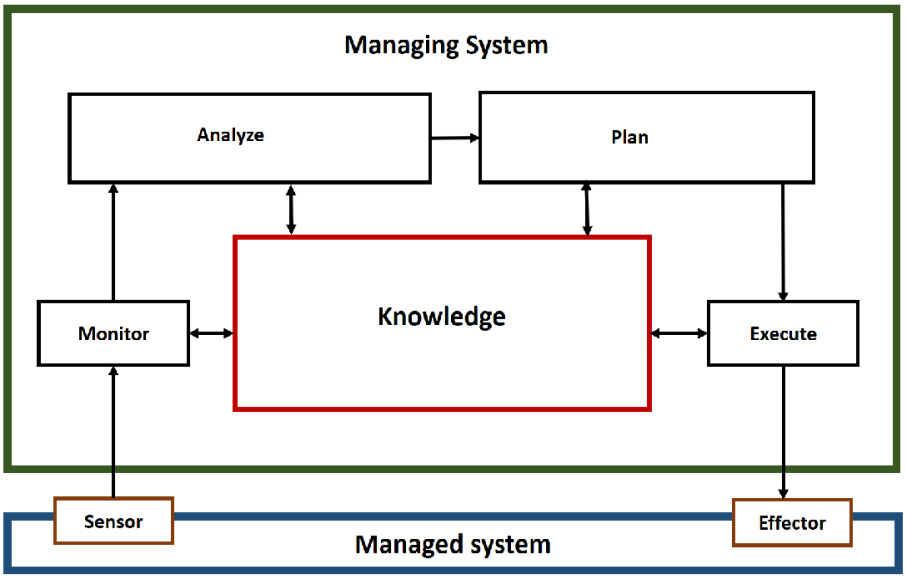


Figure 3.1: MAPE-K Architecture Pattern

b) Analyze: During the *Analyze* phase, the managing system performs analysis of the monitored data to examine if the adaptation is required.

c) Plan: Based on the result of *Analyze*, the decision-making agent plans for the adaptation actions to be performed to fulfill the functional goals and satisfy the NFRs.

d) Execute: During the *Execute* phase, the decision-making agent performs the planned actions over the managed system.

e) Knowledge: The data required by the managing system to execute all the phases of the loop.

3.1.2 Requirements-Awareness

As SASs are required to be able to possess the property of *self-reflection*, therefore the system needs to be aware of its requirements. Requirements-Awareness [16, 139] refers to the idea that a SAS needs to be aware of both its functional goals and NFRs, and how it can contribute towards their fulfillment. The requirements of SASs may change

due to unforeseen changes in the environmental contexts [119, 164]. These changes in the environmental contexts may result in the changes in the satisfaction priorities of NFRs. Therefore, the SAS is required to be aware of its NFRs with respect to their satisfaction when selecting a specific adaptation action at runtime. For this purpose, the system needs to make a trade-off between NFRs in terms of their *re-prioritization* at runtime. Hence, the system is required to be *priority-aware* with respect to the NFRs.

To achieve requirements-awareness with respect to the trade-off between requirements, the SAS needs to possess the capability of modelling and reasoning about its requirements at runtime. For the purpose of treating requirements as live runtime entities, the concept of Models@Runtime (also known as Runtime Models) is used [8, 14, 16].

Next, a description of the Runtime Models and their types is presented.

3.1.3 Runtime Models

A runtime model [14, 19] is defined as a runtime abstraction of the system or any aspect of the system used to realize self-adaptation. The aspects of the system include behavioral, structural and functional or quality goals. As the systems' aspects are related to self-adaptation, they are subject to a continuous change. Hence, the runtime models are also required to be up-to-date and should be able to reflect these dynamic changes at any point of time. Using runtime models, the system maintains a self-representation of itself by having a causal connection between the runtime model and the system itself. Causal connection means if the system changes, the runtime model representation of the system also changes, and vice versa [97, 166].

Types of Runtime Models

According to [166], the runtime models can be classified based on the different dimensions. The dimensions specify particular characteristics of the representation of a SAS or aspects associated to the system. The dimensions are considered as orthogonal to some extent. This means that during the specification of a runtime model, an option of each of the dimensions could be selected. However, not all dimensions could be applied to all runtime models. In that case, the dimensions that do not apply could be ignored. The classification of the runtime models, based on the dimensions, is provided as follows:

Structural or Behavioral:

Structural model refers to a runtime model that represents the actual structure of the system and its constituents or related aspects. It focuses on the composition of the constituent elements of the system and their relationships and states.

Conversely, a behavioral model refers to a runtime model that represents the behavior of a system or its parts with respect to the changes over time. It mainly focuses on how the system will behave based on the uncertain situations and its current state. The behavioral model captures the aspects related to activities. These aspects may include control flow of the computation and data and uncertain events of the environment, aspects related to transitions between the states of the system under uncertain conditions or the aspects regarding the interactions among the components of the system.

For example, for an Internet of Things (IoT) network, the structural model will represent the way the nodes in the network are connected with one another and other external elements such as the node manager. Whereas a behavioral model will specify the flow of activities and interactions between the different nodes of the network.

Declarative or Procedural: The declarative and procedural models complement each other and are usually required in the realization of self-adaptations. A declarative model denotes a runtime model that specifies the knowledge about what is the case and the knowledge about what is required to be done by the system in such a case. For instance, the declarative model is used to specify the knowledge of the adaptation goal indicating the purpose of self-adaptation. Moreover, it also specifies what adaptations the managing system should perform in order to achieve it. On the other hand, a procedural model is a runtime model that specifies the knowledge about how something is done or is required to be done. In other words, it specifies the knowledge of how the adaptation is going to be done. For instance, an adaptation plan specifies how a managed system needs to be adapted.

For example, for an IoT network, the declarative model could be used to specify the adaptation goals, such as the maximum packet loss in the network should not exceed 15 percent. Whereas the procedural model would specify the plan for adaptations by tuning the network link parameters, such as the increase or decrease of the transmission power.

Functional or Quality: The functional or quality models are also known as *requirements models*. They denote runtime models that specify the set of requirements (functional or non-functional) that a system is required to meet. The functional model focuses on specifying the functional and operational requirements of the system and its constituent elements that the system is required to achieve. For example, a functional model can specify internal functionality of a system or an element of it, input or output of elements, or flows between elements. In contrast, the quality model denotes a runtime model specifying the current state of the non-functional quality requirements (i.e. NFRs) of the system under uncertain environmental contexts. For instance, a quality model describes the characteristics that are significant for a system (e.g. its performance, reliability or security). Moreover, it also specifies how these characteristics are to be determined. The main focus of the work presented in this thesis is on *requirements models*, specifically the quality models.

3.2 Multi-Objective Reinforcement Learning

The decision-making in SASs requires the system to reason about multiple and often conflicting objectives under uncertain environmental contexts over time [68] (as described in Section 3.1.1). For example, an IoT network [7, 173] is continuously exposed to different environmental contexts such as dynamic traffic load and link interference. These environmental contexts have an effect on the satisfaction of the quality objectives (NFRs) such as higher levels of energy consumption and lower levels of packet delivery performance [90, 106, 143]. Therefore, a trade-off between the objectives is carried out based on their competing priorities for satisfaction. Several techniques have been developed to perform decision-making in SASs (as described in Chapter 2). However, they do not support the explicit modelling and reasoning with individual priorities of the objectives (NFRs). Instead, Reinforcement Learning approaches such as Partially Observable Markov Decision Process (POMDPs) offer a framework for SASs to naturally model such sequential decision-making problems [129, 130, 153]. Therefore, they allow reasoning about the multiple objectives and uncertainty corresponding to the environment's state. However, the setting of POMDPs typically assume only a single objective. In a setting with multiple objectives, their priorities are represented as a single combined utility value handled through a simple linear combination [112]. However, multi-objective decision problems require the explicit modelling of

the objectives along with their priorities. This could be achieved using multi-objective reinforcement learning techniques such as multi-reward partially observable markov decision processes (MR-POMDPs) [95, 128, 129]. The Pri-AwaRE architecture presented in this thesis makes use of the MR-POMDP approach to offer runtime multi-objective decision-making. Next, the concepts of POMDP and MR-POMDP are presented.

3.2.1 Partially Observable Markov Decision Process

POMDPs [153] present a decision-making framework that considers an agent working in a partially observable environment. An agent is defined as an entity that can carry out different activities or actions under changing environmental conditions [84]. In a partially observable environment, the agent can't directly observe the actual underlying state. Instead, it maintains a belief (as a probability) over the set of possible states based on observations to select the optimal adaptation action. The basic elements of a POMDP model are shown in Fig. 3.2.

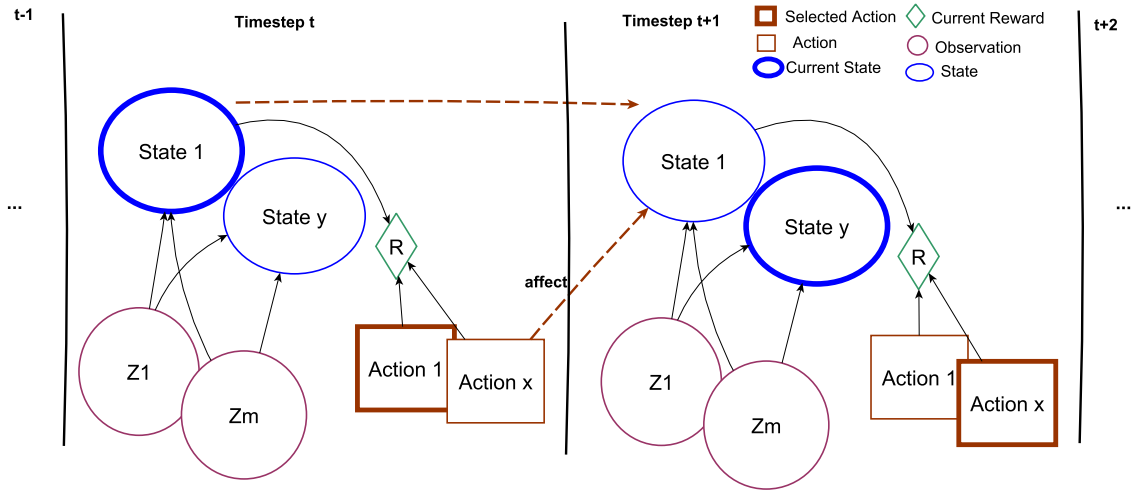


Figure 3.2: POMDP

A POMDP is specified as a tuple <S,A,Z,T,O,R,γ > where:

- S specifies the set of states of the environment;
- A represents the set of Actions that can be selected and performed by the decision-making agent at a particular point in time;

- Z specifies the set of Observations that represents the information related to the state s of the environment received by the agent using sensors or actuators;
- T specifies the transition function $T(s, a, s') = P(s'|s, a)$. It represents the probability of moving to the next state s' based on the current state s and an action a;
- O specifies the observation function $O(s, a, z) = P(z|s, a)$. It represents the probability of observing an observation z based on an action a and the resulting state s;
- R represents the reward function that is denoted by R(s,a) or R. It is a scalar real value generated as a feedback by the environment as a result of an action a performed by the agent based on the state s of the environment;
- γ is the discount factor. It is used to support the preference for the immediate reward values over the future ones. The discount factor value is between 0 and 1.

The agent tries to find the **policy** π, a mapping from state to action, that maximizes the value function i.e. *expected utility value* of the sum of discounted future rewards. The value function V_π is computed as follows:

$$V_\pi = E_\pi[R(s_t, a_t) + \gamma R(s_{t+1}, a_{t+1}) + \gamma^2 R(s_{t+2}, a_{t+2})...|s_t] \tag{3.1}$$

Hence, the value function V_π estimates how much cumulative reward, discounted by γ, we can expect to get in future when a particular action is performed based on the current state s_t at time t. Therefore, the reward value R(s,a) is used to compute the impact of performing an action a given state s using the value function. Hence, a *cardinal scale*, using reward values, is assigned to each adaptation decision during a particular state of the system to specify its priority. The reward values are typically assigned by the domain experts at design time based on their previous experiences or the information provided to them at design time [76, 104, 107].

In order to support quantification of uncertainty, POMDPs maintain a belief b over the partially observed states of the system. In point-based planning techniques for solving POMDPs, the computation of a policy is performed based on the sampling of points from the belief space [144, 154]. In such approaches, the value function over the belief V_b is specified in the form of a set of α-vectors denoted by A. Each α-vector has a length equal

to $|S|$ as it provides a value for each state s, and is associated with an action a. Hence, the α-vector presents a value for each state s after an action a is performed. The α-vector is presented as follows:

$$\alpha_a = [V(s_i), V(s_{i+1}), ..., V(s(_n))] \tag{3.2}$$

The $V(s_i)$ represents the value function for state s_i provided a total number of n states. Therefore, given the set A, the value over the belief is computed as:

$$V_b = \max_{\alpha_a \in A} b.\alpha_a \tag{3.3}$$

Hence, for each belief b, a set of α vectors A gives a policy π_A for the action that maximizes the value. The selected action is then performed by the decision-making agent which results in a change in the state of the system.

3.2.2 Multi-Reward Partially Observable Markov Decision Process

MR-POMDP [129, 148, 149] is a POMDP with multiple rewards presented in the form of a vector-valued reward function $\mathbf{R}$ as shown in Fig. 3.3. In case of MR-POMDP, a separate reward value is associated with each objective. Therefore, the reward vector size is equal to the number of objectives. As the rewards are represented in the form of a vector, the value function, given an initial belief $\mathbf{V}_{b_0}$ is also a vector. Hence, each single element in the $\mathbf{V}_{b_0}$ represents the *expected utility value* associated with each individual objective. As a consequence, it estimates the impact of performing an action on the satisfaction of each objective separately given a particular state. The value of elements in the value vector, create a relative ranking for the objectives. Hence, these *expected utility values* associated with separate objectives specify the priority of the objectives in terms of their satisfaction during the decision-making. The Pri-AwaRE architecture presented in this thesis uses this built-in capability of MR-POMDP to evaluate the *expected utility value* for each objective as the basis of *autonomous tuning* of the priorities. Therefore, it is used to answer the **RQ2**.

As the reward is presented in the form of a vector $\mathbf{R}$, each element in the α-vector is also a vector, thereby creating an α-matrix, $\boldsymbol{A}$. Each row in the α-matrix specifies the values for all the objectives in a specific state. Therefore, the multi-objective value for selecting

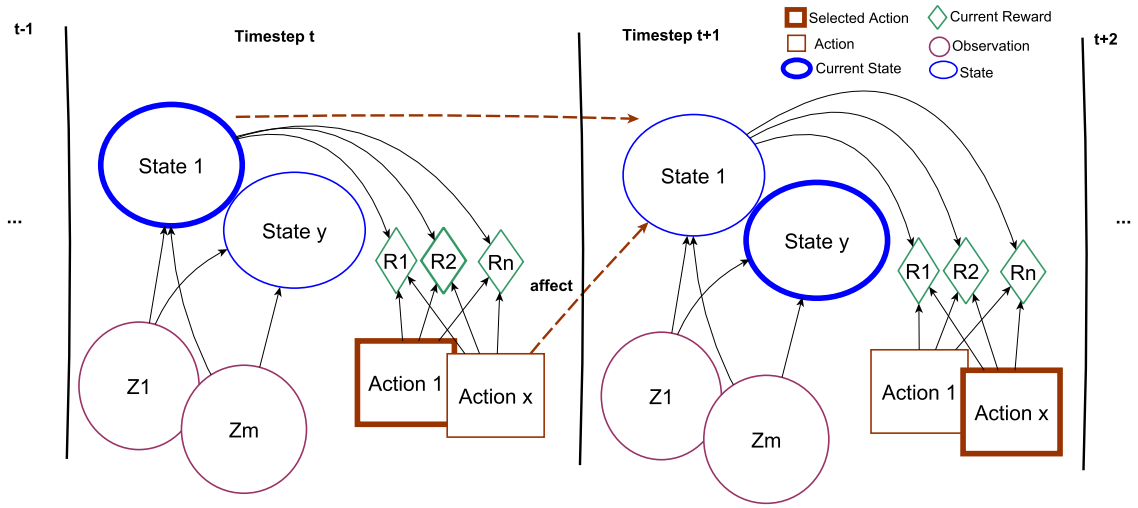


Figure 3.3: MR-POMDP

an action a corresponding to the alpha matrix $\boldsymbol{A}$ under a belief b is calculated as follows:

$$\mathbf{V}_b = b\mathbf{A} \tag{3.4}$$

In MR-POMDP, as the value function is a vector, multiple policies may exist. The value functions associated with these multiple policies can be considered as optimal based on the individual priorities of the objectives. For the purpose of selecting the best optimal policy out of these multiple policies, a scalarization function f($\mathbf{V}_b$,W) is used. The function f($\mathbf{V}_b$,W) scalarizes the value vectors $\mathbf{V}_b$ with the help of weights W, calculated by the agent, corresponding to the objectives [128] as follows:

$$f(\mathbf{V}_b, W) = W.V_b = w_i V_{b_i} + w_{i+1} V_{b_{i+1}} ... + w_n V_{b_n} = V_b(w) \tag{3.5}$$

The w_i and V_{b_i} represents the weight and value for the ith objective having n as the total number of objectives. The size of the weights vector is also equal to the total number of the objectives. The weights vector values can be estimated using different optimization techniques such as Optimistic Linear Support (OLS) [129] and evolutionary computation [148]. For the work presented in this thesis, the approach of OLS is used. The OLS, based on linear programming, computes the optimal weights that maximize the Value $V_b(w)$, as presented by Roijers et al. [129]. To the best of my knowledge, its the only implementation of the MR-POMDP solver available. Description of the OLS is provided in Appendix D.

Hence, for a given belief b, α matrix for each action and weight w, we can calculate the policy π_A that takes the maximal value using equations 3.4 and 3.5 as:

$$V_b^*(w) = \max_{\mathbf{A} \in A} b\mathbf{A}W \tag{3.6}$$

Therefore, in MR-POMDP, the values in the reward vector represents the priorities of objectives by specifying their desirability for satisfaction given a particular state of the environment. These reward vector values are initialized with the estimated values that are assigned by the domain experts. The domain experts assign these values at design time based on their knowledge in the domain and the information provided to them.

3.3 Summary

In summary, the decision-making in SASs requires the system to reason about multiple and often conflicting objectives (i.e. NFRs) under uncertain environmental contexts over time. Different environmental conditions can have different effects on the satisfaction of the NFRs. Hence, a trade-off between the NFRs is carried out based on their competing priorities for satisfaction. Multi-objective Reinforcement Learning techniques, such as MR-POMDPs, offer a framework for sequential decision-making that naturally supports modelling and reasoning with the multiple objectives and their respective priorities. The details of how Pri-AwaRE makes use of MR-POMDPs as part of MAPE-K loop to carry out runtime priority-aware decisions for SASs are provided in Chapter 4.

Chapter 4

Pri-AwaRE: A priority-aware self-adaptive architecture for decision-making in Self-Adaptive Systems

---

The work presented in this chapter has been adapted from the following publications:

[136] H. Samin, L. Garcia Paucar, B. Nelly, and P. Sawyer. Towards priority-awareness in autonomous intelligent systems. In *36th ACM/SIGAPP Symposium On Applied Computing (SAC)*. ACM, 2021.

[138] H. Samin, N. Bencomo, and P. Sawyer. Decision-making under uncertainty: be aware of your priorities. *International Journal on Software and Systems Modeling (SoSyM)*, 2022.

4.1 Introduction

This chapter presents the Pri-AwaRE architecture which is a self-adaptive architecture for decision-making in SASs. The architecture makes use of the Multi-Reward Partially Observable Markov Decision Process plus plus (MR-POMDP++) as a runtime model that works as part of MAPE-K loop. MR-POMDP++ is an extension of MR-POMDP which is a multi-objective reinforcement learning technique. It supports modelling and reasoning with individual priorities of NFRs. It also offers autonomous tuning of NFRs' priorities during the

decision-making process. The "++" refers to the capability of offering *priority-awareness* for NFRs. In the following sections, a description of the Priority-Aware MR-POMDP++ and its use as a part of the Pri-AwaRE architecture is presented using an illustrative case of the Remote Data Mirroring (RDM) network.

4.2 Illustrative Example

Remote Data Mirroring [74, 78] is a disaster recovery technique that is used to tolerate failures by maintaining multiple copies of data at remotely located servers (also known as *Mirrors*). It helps in making the data available by preventing data loss. Each link in the RDM network has an associated operational cost, measurable throughput, latency and loss rate. These parameters can be used to determine the cost, performance and reliability of the RDM system. The goal is to achieve the satisfaction of the NFRs such as *Minimization of Operational Costs*[1] *(MinC). Maximization of Performance*[2] *(MaxP)* and *Maximization of Reliability*[3] *(MaxR)* under uncertain environmental conditions of network link failures and varied bandwidth consumption [74]. The uncertainty in the environment has an effect on the satisfaction of NFRs. Therefore, to maintain the required NFRs' satisfaction levels, the network is required to take adaptive decisions by switching between the topologies such as *Minimum Sapnning Tree (MST)* and *Redundant Topology (RT)* [53]. Both the topologies have different impact on the satisfaction of NFRs. *MST* topology supports the satisfaction of *MinC* and *MaxP* by maintaining a minimum spanning tree for the network. However, it has a negative impact on the satisfaction of *MaxR*. Conversely, *RT* topology supports the *MaxR*. Whereas it has a negative impact on the satisfaction of *MinC* and *MaxP* as the non-stop application of *RT* results in higher operational costs and lower performance due to data redundancy.

4.3 MR-POMDP++

In this section, the MR-POMDP++ runtime model to perform priority-aware decision-

[1]Operational Cost is measured in terms of inter-site network traffic.
[2]Performance is measured as the sum of writing time of all copies of data on each remote site.
[3]Reliability is measured in terms of the number of active network links.

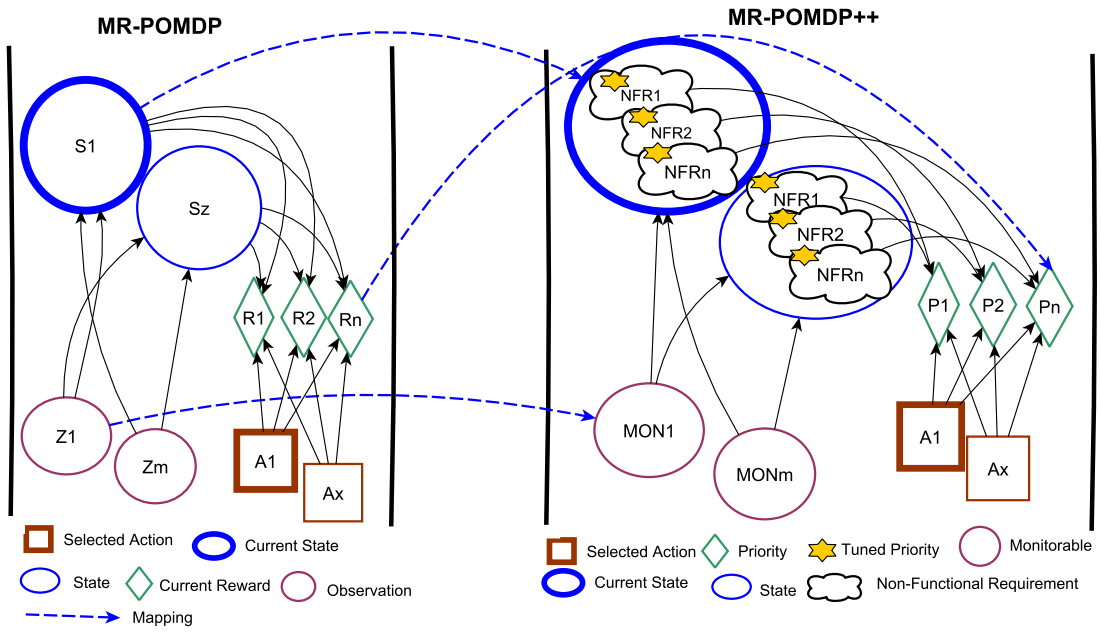


Figure 4.1: Mapping of MR-POMDP to Priority-Aware MR-POMDP++

making in SASs is presented. The MR-POMDP++ is an extension of MR-POMDP. It offers the runtime modelling and reasoning with the individual priorities and satisfaction levels of NFRs during decision-making. The "++", therefore, refers to the ability to offer *priority-awareness* for NFRs. The mapping of MR-POMDP to MR-POMDP++ is shown in Fig. 4.1. As presented in Fig. 4.1, the states are used to represent the satisfaction state of NFRs, the rewards are used to represent the priorities of the NFRs, the actions are used to represent the adaptation actions for the SAS and the observations are used to represent the monitorables. Next, a detailed description of the mapping using the mapping rules for NFRs' representation in the form of MR-POMDPs is presented.

1) NFR satisfaction and MR-POMDP states

Let us recall that the satisfaction level of an NFR refers to the degree to which it is satisfied (see Section 3.1.1). During the decision-making process in SASs, an NFR's satisfaction level at a particular point in time corresponds to the extent to which it is

satisfied as a result of an adaptation performed by the system, and its level of satisfaction at the previous execution time step. Therefore, to achieve satisfaction of NFRs, a SAS is required to take adaptive actions. These adaptive actions can have different effects (good or bad) on the NFRs' satisfaction. Due to such lack of crispness in the nature of NFRs, they can't be labelled as fully satisfied nor fully violated. Therefore, the NFRs' satisfaction levels can't be represented in the form of an absolute value of True or False [63]. Hence, the satisfaction levels can be modelled as the probability distributions such as P(NFR=True). For an NFR to be considered as satisfied, it should meet the required acceptability threshold. The acceptability threshold refers to an acceptance condition or criterion for the NFR's satisfaction. If the satisfaction level of an NFR meets the acceptance condition, it is considered as satisfied and vice versa. The acceptability thresholds for NFRs are defined by the domain experts at design time based on the stakeholders' requirements [155]. For example, in the case of the RDM, the satisfaction level of *MaxR* could be represented as P(MaxR=True)=0.9 or P(MaxR=True)=0.5. If the acceptability threshold for satisfaction of *MaxR* is 0.8, then *MaxR* having P(MaxR=True)>=0.8 will be considered as highly satisfied.

Such specification of NFRs' satisfaction levels can be modelled using the states of MR-POMDP. Therefore, in MR-POMDP++, each state is considered to present the set of combinations of satisfaction levels of NFRs, as shown in Fig. 4.1. As the states in MR-POMDP are partially observable, a belief (i.e. a probability) over each state is maintained. Hence, the satisfaction levels of NFRs can be represented as the marginalized probability distributions $P(NFR_i = True)$ where NFR_i is a member of the set of NFRs [112].

Based on the above description, a mapping rule is derived as follows:

Rule: 1 *The state $s \in S$ in MR-POMDP++ represents the set of combinations of satisfaction levels of the non-functional requirements ($NFR_1...NFR_n$). As the states in the MR-POMDP++ are partially observable, the satisfaction levels of the NFRs can be represented in the form of probability distributions $P(NFR_i = True)$.*

These probabilities can be used to conclude if the satisfaction levels meet the acceptability thresholds.

Using Rule 1, the total number of states in MR-POMDP++ in terms of the satisfaction

Table 4.1: States of the RDM Network in terms of NFRs

S	NFR_1=MinC	NFR_2=MaxR	NFR_3=MaxP
s_1	True	True	True
s_2	True	True	False
s_3	True	False	True
s_4	True	False	False
s_5	False	True	True
s_6	False	True	False
s_7	False	False	True
s_8	False	False	False

levels of NFRs can be calculated as: $|S| = |2|^{|NFR|}$. The $|S|$ represents the size of set S, $|NFR|$ represents the number of NFRs, and *2* indicates True and False. For example, in the RDM network, if we consider 3 NFRs of Minimization of Operational Cost (*MinC*), Maximization of Reliability *(MaxR)* and Maximization of Performance (*MaxP*), so the number of states for MR-POMDP++ will become $|S| = 2^3 = 8$ as shown in Table 4.1.

2) NFR priorities and Reward Vectors

A priority of an NFR refers to the importance of an NFR in terms of satisfaction. The higher the priority, the more important it is to satisfy that NFR. Therefore, the satisfaction priorities of individual NFRs should be considered during decision-making in a SAS.

MR-POMDPs supports the modelling of these individual NFR priorities with the help of a vector-valued reward function, as shown in Fig. . Each value in the reward vector is associated with each separate objective (i.e. NFR in case of Pri-AwaRE). The reward values are generated in the form of a feedback signal as a result of the decisions (adaptation actions) taken by MR-POMDPs. The reward value for a specific objective refers to the effect (positive or negative) of performing an adaptation action on the satisfaction of that objective. Therefore, the values in the reward vector represent a relative ranking of the objectives (NFRs). This relative ranking is with respect to the *cardinal* effect that an action will have on the satisfaction of the objectives under an uncertain environmental situation. As a consequence, an objective having a higher reward value indicates that the objective has a higher priority (importance level) for satisfaction. These priorities, in the form of rewards, are considered by MR-POMDPs while selecting adaptation actions.

For example, in the RDM network, if link failures occur at a particular point of time, they will result in the loss of data packets. Consequently, there will be a decrease in the reliability of the network. To support the satisfaction of *MaxR*, the decision-making agent might take the adaptation decision of switching to the topology of *RT*. The application

of *RT* topology will have a positive impact on the satisfaction of *MaxR* whereas it will negatively impact *MinC* and *MaxP*. As the application of *RT* activates more network links, it increases the operational cost and reduces the performance[4] of the system. Therefore, based on the selected adaptation action of *RT*, under the current environmental situation of link failures, the system will generate a higher immediate reward for *MaxR* (such as 80) than for *MinC* and *MaxP* (that could be -40 and -20 respectively). The reward for *MaxR* is higher than the rewards for *MinC* and *MaxP* because it is more important to satisfy *MaxR* under the current environmental situation.

The reward values, in the form of a vector, specify a relative ranking of the NFRs *MinC*, *MaxR* and *MaxP* concerning the impact that an adaptation action a (i.e. *RT*) will have on their satisfaction state s as follows:

$$\mathbf{R(s,a)} = [R_{MinC}, R_{MaxR}, R_{MaxP}] = [-40, 80, -20]$$

The initial assignment of the reward values is done by the requirements engineers or domain experts at design time [155]. In MR-POMDP++, these reward values are considered as the initial expert defined priorities for NFRs.

Based on these concepts, in MR-POMDP++, the priorities representation for NFRs is described by the following rule:

Rule: 2 *The values in the reward vector* ***R(s,a)*** *over the execution of an action* $a \in A$ *given state* $s \in S$ *in MR-POMDP++ represent the priorities of non-functional requirements (NFRs).*

$$\mathbf{R(s,a)}=[\mathrm{R}_{NFR1}, \mathrm{R}_{NFR2},..., \mathrm{R}_{NFRm}]$$

The R_{NFR1} specifies the reward for NFR1 indicating the priority value of NFR_1 and so on. The mapping of the rewards to the priorities of the NFRs is presented in Fig. 4.1.

Assignment of Reward values

As noted above, NFR priorities are typically initialized using estimated values at design time. These values are defined by the domain experts based on their expertise in the

[4]Performance for the RDM is measured in terms of total time to write data.

domain and the information available to them. The assignment of priorities at design time is a normal requirements practice. However, in case of SASs, the priorities assigned at design time may not be appropriate under all the dynamic contexts encountered by the system at runtime. Moreover, deploying expert knowledge in such cases is not always easy, as highlighted in [155], where the experts' consensus proved hard to achieve. In the work presented in this thesis, simulation environments of [72, 137] are used to come up with good initial values for the rewards.

3) Expected Utility Values and Autonomous Tuning of Priorities

In SASs, the priorities of NFRs may change due to the variation in the environmental conditions. The capability to adapt autonomously according to the changing environmental contexts is what MR-POMDP++ offers. For instance, in an Internet of Things (IoT) network, the conservation of energy may be the NFR with highest priority. However, if the sensors' batteries become fully charged at a particular time, generating a higher reward to support this NFR may not be the optimal behaviour in such a case. To handle such kind of situations, MR-POMDP++ using reward vector provides an opportunity to *autonomously* tune the individual priorities of NFRs. This is performed by computing an individual *expected utility value* for each NFR EU_{NFR} during the decision-making carried out by MR-POMDPs (using equation 3.1) as follows:

$$EU_{NFRi} = V_{NFRi} = E_{\pi}[Ri_t + \gamma Ri_{t+1} + \gamma^2 Ri_{t+2}...|s_t] \tag{4.1}$$

where EU_{NFRi} and Ri specify the *expected utility value* and reward values for NFRi respectively. As presented in equation 4.1, the rewards, specifying the initial expert assigned priorities, are used to calculate the individual *expected utility value* for each NFR at runtime. Therefore, these *expected utility values* represent the newly *tuned priority* of the individual NFRs. The *expected utility values* take into account the impact of executing an adaptation action on the satisfaction of each NFR given an uncertain environmental condition. Therefore, they are considered by MR-POMDP++ while taking decisions for adaptations at runtime.

*Hence, MR-POMDP++ as a runtime specification model (**S**) considers the new knowledge (**K'**) about the individual priorities of the NFRs to perform runtime decision-making for SASs in order to conform to the Requirements (**R**).*

Next, the Pri-AwaRE self-adaptive architecture that uses Priority-Aware MR-POMDP++ to perform decision-making for SASs is presented.

4.4 Pri-AwaRE Architecture for Decision-Making in SASs

The Pri-AwaRE architecture for decision-making in SASs takes inspiration from the feedback architecture of the Reinforcement Learning (RL) process. The RL process comprises of the interactions between the decision-making agent and its operating environment [95]. As a result of the monitored observations, the decision-making agent performs an action for the purpose of achieving the desired goals. The process is repeated continuously. On the basis of this description, the Pri-AwaRE architecture is defined. Pri-AwaRE comprises of two components: 1) the managed system (i.e. the environment in RL process), and 2) the managing system (i.e. the decision-making agent in RL process), as shown in Fig. 4.2.

The Pri-AwaRE architecture structures the *managing system* interacting with the *managed system* using the probe and effector interfaces [137], as presented in Fig. 4.2. The main focus of Pri-AwaRE lies in the aspects of the managing system, which is based on the MAPE-K loop.

Next, the components of the Pri-AwaRE architecture such as the managed and the managing system are presented. The usage of *priority-aware* MR-POMDP++ to support modelling of the satisfaction levels and priorities of individual NFRs is also described.

1) Managed System

The *managed system* specifies the environment for which the self-adaptive capabilities are implemented. It is equipped with the components of *probe* and *effector* that act as an interface for interacting with the managing system. For example, an RDM network executing according to its pre-defined settings can be viewed as a *managed system*. For the purpose of adding self-adaptive capabilities, an external *managing system* has to be

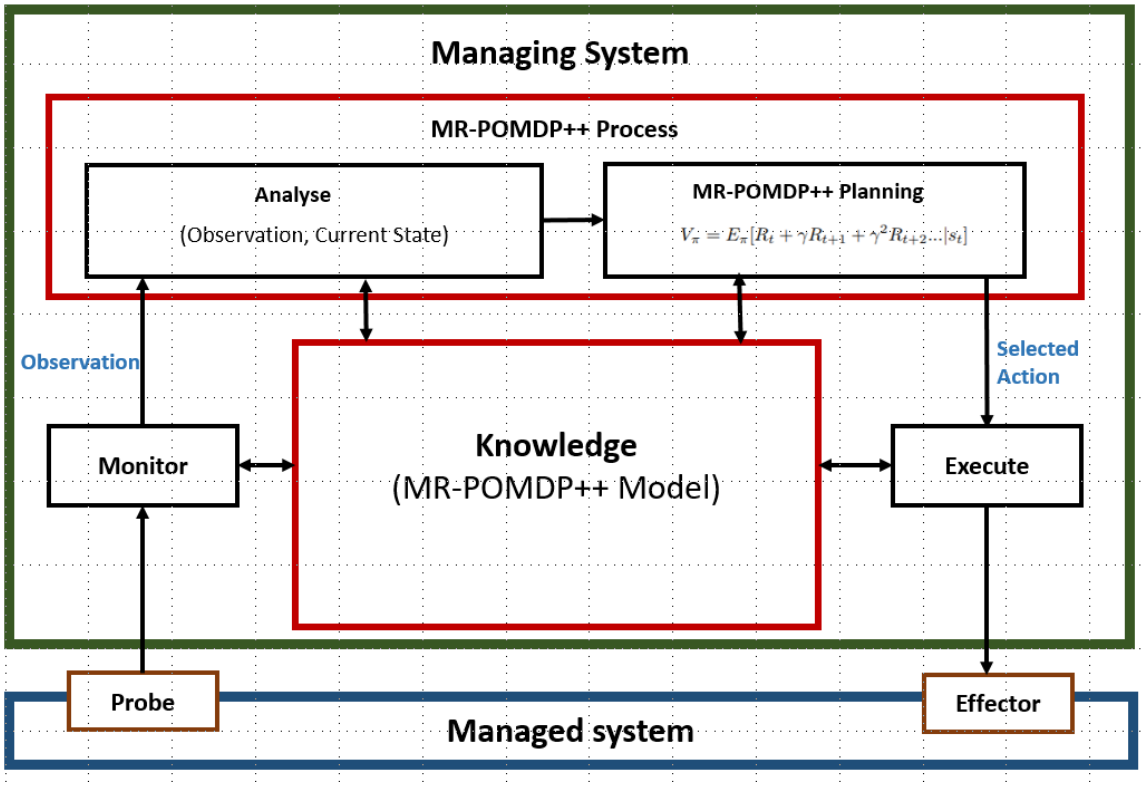


Figure 4.2: Pri-AwaRE Architecture

connected to the network. This connection between the *managing* and *managed system* is established using the *probes* and *effectors* [72, 137].

2) Managing System

The structure of the *managing system* is based on the phases of the MAPE-K feedback loop. Hence, the managing system comprises of the following components:

a) Monitoring Component: The *monitoring* component of the managing system makes use of sensors to acquire data related to the monitorable values (e.g. traffic load on the network links in the RDM network). These monitored values are then sent as an input to the MR-POMDP++ to indicate observations for the model.

b) Analysis and Planning Components: The *analysis* and *planning* components of the managing system depends on the steps of the MR-POMDP++ process. MR-POMDP++ analyses the observations received from the monitoring component and the runtime ***Knowl-***

edge in the form of current belief over the states (maintained by MR-POMDP++). Based on this information, it then performs planning for selecting adaptation actions. The goal here is to choose adaptation actions that help achieve the network's target operational goals. Moreover, the satisfaction of the NFRs with respect to complying with the *requirements specification* (***R***) should also be achieved.

c) Execution Component: The *execution* component takes the selected action by MR-POMDP++ as an input and executes it on the managed system using the effectors.

d) Knowledge Component: The *knowledge* component comprises of the runtime knowledge maintained by MR-POMDP++ which is considered during the decision-making process.

This process is repeated continuously during the execution of the SAS. Next, the application of Pri-AwaRE architecture to the example case of the RDM network is described.

4.5 Pri-AwaRE Architecture for decision-making in the RDM Network

The Pri-AwaRE Architecture for decision-making of the RDM network comprises of the following two components:

1)The Managed System: represents an RDM network. As a concrete example, let's consider an RDM network presented by the RDMSim simulator [137]. More details about the simulator are provided in Chapter 5.

2)The Managing System: is based on the MAPE-K loop and MR-POMDP++. It comprises the following components:

a) Monitoring Component

At a particular time, data related to the different network parameters (such as total

bandwidth consumption, total time to write data and number of active links etc.) is accumulated by the *monitoring component* using the *probe* of the RDM network. The monitored data is then sent to the *Analysis* and *Planning* components.

b)Analysis and Planning Components

To support priority-awareness during decision-making, the *analysis* and *planning* components are based on the steps of MR-POMDP++. The monitored network parameter values are sent as the input observations to the MR-POMDP++ model. The model analyses the observed *network parameter* values and the *current belief* (i.e. the *Knowledge K*) maintained by the MR-POMDP++ model itself at runtime. It then plans for the selection of the next adaptation action to be performed by the RDM system. The *knowledge* component of the MAPE-K loop consists of the knowledge maintained by MR-POMDP++ at runtime.

c)Execution Component

The *execution component* executes the chosen adaptation action, in the form of the topology configuration, on the managed RDM system by using the *effector*.

4.6 Summary

In summary, the proposed Pri-AwaRE architecture offers priority-awareness during the decision-making process of the SASs. The architecture makes use of the priority-aware runtime model, known as MR-POMDP++, as part of the MAPE-K feedback loop. MR-POMDP++ is based on the multi-reward POMDP model. It defines a separate reward value for each NFR using a vector-valued reward function. These reward values refer to the initial defined priority of the NFRs indicating their importance for satisfaction. During the decision-making process, the MR-POMDP++ offers *autonomous tuning* of the NFRs' priorities according to the changing runtime contexts. This autonomous tuning is carried out by computation of the individual *expected utility values* for the NFRs.

Chapter 5

Case Studies

The work presented in this chapter has been part adapted from the following publication:
[137] H. Samin, L. H. G. Paucar, N. Bencomo, C. M. C. Hurtado, and E. M. Fredericks. RDMSim: An Exemplar for Evaluation and Comparison of Decision-Making Techniques for Self-Adaptation. In *16th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Artefact Track*, 2021.

5.1 Introduction

This chapter describes the case studies used for the evaluation of the research presented in this thesis. As a proof of generalization of the Pri-AwaRE approach, two case studies from the different networking domains have been selected for evaluation purposes. The first case study is based on a self-adaptive IoT network. As a concrete example for the IoT case, an existing artefact of DELTA-IoT [72] has been selected. The DELTA-IoT provides a simulated environment for an IoT network for a smart university campus. On the other hand, the second case study is based on the self-adaptive remote data mirroring (RDM) network. For the RDM system, a simulation tool of RDMSim [137] has been developed as part of the research presented in this thesis. The RDMSim is designed to provide simulations of the different dynamic contexts of the RDM network. The runtime decision-making offered by the Pri-AwaRE focuses on the SASs that are not hard real time. The selected case studies provide a good representation of such systems. Moreover, both the IoT and RDM applications are well accepted in the research community, and are already in use by the other teams [26, 54, 72, 73]. The details about both the case studies and the respective

simulation tools are provided in the next sections.

5.2 Case Study 1: Internet of Things

Internet of Things (IoT) [7, 173] refers to the networked interconnection of different computing systems (also known as nodes). The nodes interact with each other to achieve the target goal of transmission of information within the network. These computing systems may include hardware and software components such as Radio-Frequency Identification (RFID) tags, sensors, actuators and mobile phones etc. To connect the digital processes to our physical world, the nodes in the network monitor and control the physical environment. The IoT systems serve as a base to the application areas of smart homes, assisted living for elderly care etc. Hence, such systems play an important role in improving the activities of daily living. Due to the limitations of size and operational costs, the nodes in the IoT network have to deal with limitations of computational storage and energy resources. The main goal for IoT systems is to increase the network's lifetime by minimizing the energy consumption of the nodes. Moreover, the system is also required to maintain the packet delivery performance while dealing with the uncertainties such as communication interference and dynamic traffic load. To deal with the environmental uncertainties, network settings are required to be tuned autonomously at runtime. For this purpose, the IoT network is required to maintain an optimal configuration to satisfy the NFRs, by performing local self-adaptive decisions for each node. To evaluate the Pri-AwaRE approach, from the perspective of performing local decisions, the simulation tool of DELTA-IoT [72] has been used as a concrete example. Next, the operational model, architecture and the uncertainty scenario of the DELTA-IoT network are described.

5.2.1 Operational Model

The DELTA-IoT simulator is an exemplar representing an IoT network for a smart campus. The simulator represents a multi-hop IoT network comprising 15 nodes which are installed across the different buildings of the KU Leuven campus. The nodes are based on LoRa (Long-Range) radio communication. In each building, the nodes are deployed to offer access to labs, monitor the occupancy status, and sense the temperature using RFID, passive infrared, and temperature sensors respectively. The nodes communicate with each other

to achieve the functional requirement of relaying information to the central gateway. The gateway is installed at the central monitoring facility of the campus and is responsible for linking the network nodes and monitoring the traffic on the network. The main goal for the IoT systems is that they are required to survive longer on a single battery and provide reliable communication. Therefore, the DELTA-IoT network is required to satisfy the NFRs: Minimization of Energy Consumption (MinEC) and Minimization of Packet Loss (MinPL). To satisfy these NFRs under uncertain environmental contexts such as link interference or dynamic traffic load at runtime, the network is required to configure the network link settings (e.g. the transmission power, communication range[1] and distribution factor[2] for links) for each individual node using different local adaptation decisions.

5.2.2 Architecture

The DELTA-IoT simulator has been designed by the experts of Internet of Things domain [73, 166] to support the self-adaptive decision-making process for an IoT network. The architecture for the DELTA-IoT simulator, as presented in [72], is shown in Fig. 5.1. The architecture comprises of the Managing System Tier placed on top of the Managed System that simulates an IoT network (known as DELTA-IoT network). The Managed System comprises of the three tiers known as IoT Network Tier, Gateway Tier and Management Tier. The Managing System tier is responsible for selecting the adaptation decisions for the DELTA-IoT network. The IoT Network Tier includes the implementation of nodes that constitute the DELTA-IoT network. The Gateway Tier is responsible for providing an implementation of the network gateway. All the nodes in the network transfer the data to the central gateway which can then be passed to the Managing system. Finally, the Management Tier is responsible for monitoring and managing the gateway and all the network nodes using the probe and effector application programming interfaces (APIs). The probing APIs can be used by the managing system to collect the sensor data about the simulated DELTA-IoT network. On the other hand, the effector APIs can be used to perform tuning of the network parameters as a result of the decision made by the managing system. Using the architecture of the DELTA-IoT, the simulator can, therefore, be used to facilitate the two-layered architecture of the Pri-AwaRE approach, as presented in Chapter

[1] The communication range refers to the distance over which signals can be transmitted in the IoT network.

[2] Distribution factor refers to the percentage of the traffic that is sent by a node to its parent node. In an IoT network, the sum of the distribution factors for one node is 100 [72].

4. According to the Pri-AwaRE architecture, a *managing system* using the MAPE-K loop [79] is structured on top of the *managed system*. Hence, using the probe and effector APIs, the DELTA-IoT network can be used as the *managed system* for the Pri-AwaRE.

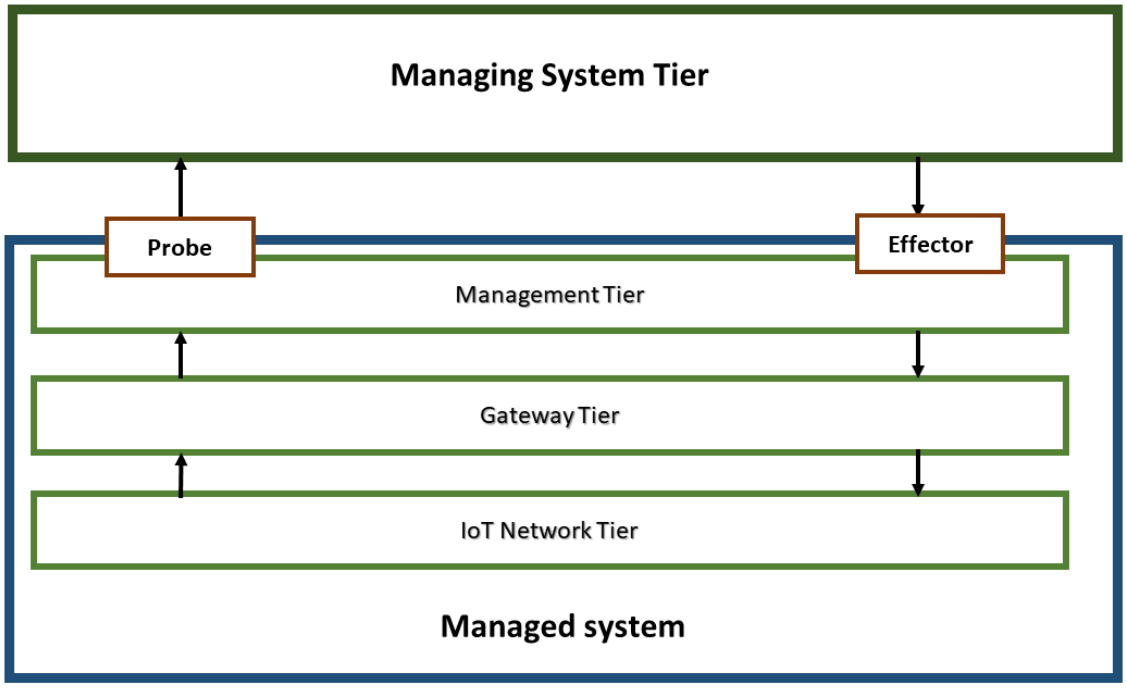


Figure 5.1: DELTA-IoT Architecture

5.2.3 Uncertainty Scenario

The uncertainty scenario representing the dynamic environmental situation for the DELTA-IoT network is described as follows:

Scenario - Wireless Interference. Under this scenario, the uncertain situations caused by wireless network interference on the network links are considered. The communication interference observed has an effect on the energy consumption and also increases packets loss. The goal here is to minimize the packet loss (i.e. MinPL) and reduce energy consumption (i.e. MinEC) by tuning the network link settings such as transmission power and communication range.

5.3 Case Study 2: Remote Data Mirroring

RDM is a technique used to protect data by storing multiple copies (i.e. replicas) on the remotely located servers (i.e. mirrors) [74, 78]. As a consequence, a RDM system prevents loss of data and ensures data availability. Moreover, to ensure data protection, the RDM system must perform the mirroring efficiently and reliably while incurring low operational costs. To evaluate the Pri-AwaRE approach, the simulation tool of RDMSim is developed to serve as a concrete example for the RDM case study. Next, a description of the RDM network presented by the RDMSim tool is provided.

5.3.1 RDMSim: Remote Data Mirroring Simulator

The RDMSim is an exemplar that is developed considering the operational model of an RDM system presented in [74, 78]. It simulates the RDM system as a fully connected network of mirrors. The RDMSim allows the creation of a customized RDM network based on the experiments' requirements by offering flexibility to change the number of mirrors. The focus is to make self-adaptive decisions in the form of topologies of Minimum Spanning Tree (MST) and Redundant Topology (RT). An MST topology uses the least possible number of network links to transmit data among different remote servers. Contrarily, an RT topology uses simultaneously, several network links paths to transmit information among remote servers. The application of these topologies has an impact on the different network parameters such as bandwidth consumption and writing time of the data. As a consequence, it affects the satisfaction of NFRs such as the minimization of operational costs and maximization of performance of the network. A trade-off of such impacts needs to be considered as part of the decision making process [45, 64, 121, 131, 139]. These topological impacts in the RDMSim have been defined based on the expert knowledge presented in [56]. Moreover, different scenarios that specify the different possible uncertain environmental conditions for the RDM have also been implemented in the RDMSim. The simulator has been implemented in Java, and is publicly available at [132]. The simulator has been developed in such a way that it can also be used by researchers working on the different decision-making techniques such as Evolutionary Computation [120], Multi-Criteria Decision-Making (MCDM) [158] among others. Furthermore, considerable research efforts have targeted the domain of Remote Data Mirroring [2, 25, 52, 56, 120] for SASs. However, the RDM applications are

very costly to implement as the equipment used to install such applications is expensive. To the best of my knowledge, there is no simulator available to facilitate experiments for decision-making techniques for a self-adaptive RDM system.

In this thesis, the RDMSim has been used to perform experimental evaluations for the proposed Pri-AwaRE approach. Next, the operational model, architecture and the different uncertainty scenarios of the RDMSim are described.

5.3.2 Operational Model

The RDM application constitutes data servers and network links [74, 78]. The primary focus is on the replication and distribution of data efficiently. Moreover, the system is also required to minimise consumed bandwidth and ensure that the distributed data is neither lost nor corrupted [74]. The RDM system must attain the functional requirements of constructing a connected network and distributing data. These functional requirements can be accomplished through alternative realization strategies specified by two different topologies: MST and RT. An MST Topology uses the minimum possible number of network links to transfer data among different remote servers (i.e. mirrors). In contrast, an RT topology simultaneously uses numerous redundant network links paths for the transmission of information across the servers. The operational model of RDMSim also considers the satisfaction of the NFRs concerning the quality and performance of the RDM system [63]. The NFRs that have been considered are Minimization of Operational Cost (MinC)[3], Maximization of Reliability (MaxR)[4] and Maximization of Performance (MaxP)[5]. The satisfaction levels related to the performance, reliability and operational costs of the RDM are determined according to trade-offs which are based on the application of the topologies as follows:

- An RT Topology offers higher levels of reliability as compared to MST topology. However, sustaining an RT topology may be expensive in some contexts, given the additional cost of required bandwidth consumption.
- On the other hand, MST Topology offers lower operational costs and higher levels of performance in comparison to the RT topology. However, the system's reliability is affected negatively by the application of MST Topology.

[3]Operational Cost is measured in terms of inter-site network traffic.
[4]Reliability is measured in terms of the number of active network links.
[5]Performance is measured in terms of total time to write data i.e. the sum of writing time of all copies of data on each remote site.

5.3.3 Architecture

Similar to the case of DELTA-IoT, the RDMSim supports the two-layered architecture of Pri-AwaRE. The architecture for RDMSim to facilitate the implementation of a self-adaptive RDM network is shown in Fig. 5.2. Next, a description of each layer is provided.

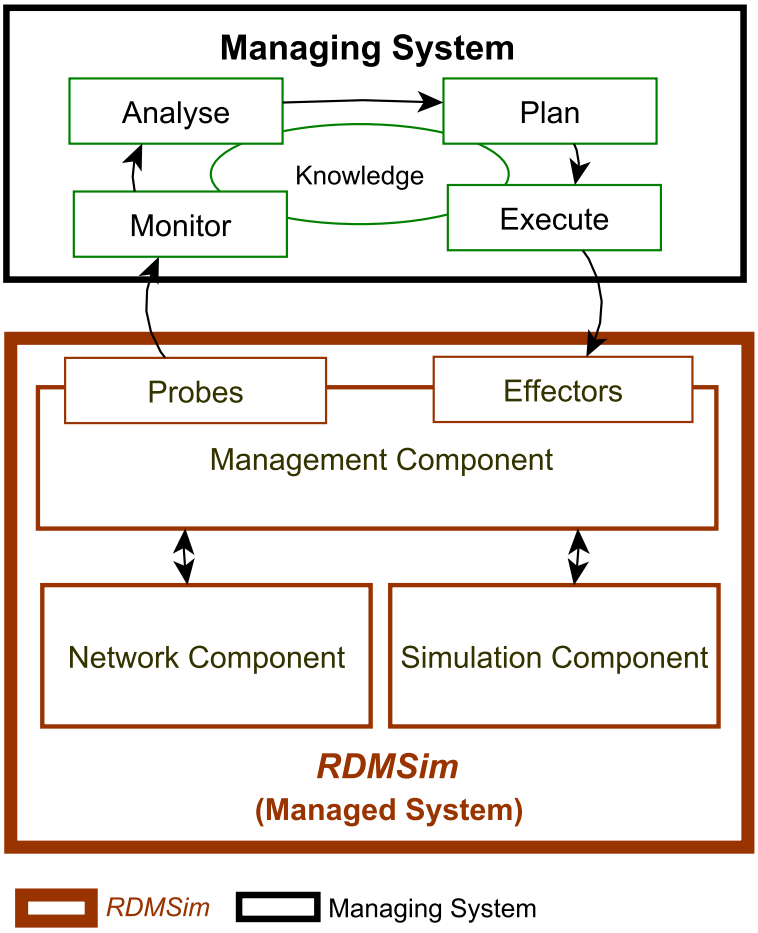


Figure 5.2: RDMSim Architecture

1) Managing System

At the upper layer, the **Managing system** is responsible for providing the logic for self-adaptive decision-making. A MAPE-K loop is implemented to monitor the environment and **Managed system** (e.g. RDMSim) and perform adaptations on the Managed system when necessary. Different decision-making techniques such as MCDM [158], Reinforcement Learning [136], and Evolutionary Computation [25, 120] can be used to serve as a Managing

Table 5.1: Probe Functions

Function	Description
Topology getCurrentTopology()	Returns the current topology for the network.
int getBandwidthConsumption()	Returns the total bandwidth consumption of the network.
int getActiveLinks()	Returns the number of active links.
int getTimeToWrite()	Returns the total time to write data for the network.
Monitorables getMonitorables()	Returns the values for all the monitorable metrics.

system. In case of Pri-AwaRE, the Managing system makes use of MR-POMDP++ as part of MAPE-K loop to perform decisions.

2) Managed System

The RDMSim simulator represents the Managed system. It provides *probes* and *effectors* that the Managing System can use to interact with the simulator. *Probes* are used to monitor information from the environment. Whereas the *effectors* are responsible for execution of the adaptation decisions on the Managed system.

Next, the architecture of the RDMSim implemented in the form of Java Packages is described. The components of the RDMSim architecture, presented in Fig. 5.2, are as follows:

Management Component

The Management Component serves as a bridge between the Managing system and internal components of the RDMSim. It offers an implementation of probes and effectors that the Managing system can use to apply decisions. The functions provided by the probes can be used to monitor the status of the RDM in terms of cost, performance and reliability. On the other hand, the effectors' functions can be used to apply the change of network topology and different network parameters based on the selected decisions. The probe and effector functions as provided in Tables 5.1 and 5.2, respectively.

Network Component

The Network Component implements the main physical elements of the RDM. These elements comprise the number of mirrors and the network links to create a fully connected network of mirrors. For example, for the 25 mirrors, a fully connected network of 300 links will be created. A customized RDM network can be created by modifying the number of

Table 5.2: Effector Functions

Function	Description
void setNetworkTopology(int timestep,Topology selectedtopology)	To set the network topology at a particular simulation timestep.
void setActiveLinks(int active_links)	To set the number of active links for the network.
void setTimeToWrite(double time_to_write)	To set the time to write data for the network.
void setBandwidthConsumption(double bandwidth_consumption)	To set bandwidth consumption for the network.
void setCurrentTopology(Topology current_topology)	To set topology for the network.

mirrors. The Network Component also implements the topologies and monitorables for the network. In the RDMSim, implementation of three monitorables is provided:

Mon1– Active Network Links: specify the currently active links of the network used to measure the reliability of the RDM. The RDM provides higher reliability levels with a more significant number of active links.

Mon2– Bandwidth Consumption: represents the current bandwidth consumed. It is used to measure the operational cost for the RDM based on the inter-site network traffic. Operational costs for the RDM will increase with higher levels of bandwidth consumed. The bandwidth consumption is measured in GigaBytes per second.

Mon3– Time to Write Data to mirrors: specifies the network's performance in the form of writing time to maintain multiple copies of data on each remotely located server. A higher writing time results in a reduction in the performance of the RDM. Time to Write Data is measured in milliseconds.

The design of RDMSim considers *synchronous mirroring* [2, 78] for the communication between the mirrors. Sequential writing is performed during the *synchronous mirroring* for the purpose of preventing data loss [2]. During the sequential writing process, the primary mirror (known as the sender) waits for an acknowledgement (i.e. handshake) about the receipt and writing of data from the secondary mirror (known as the receiver). This process is applied for each active link located on the communication path between the mirrors. Hence, the time to write data is calculated as:

$$\textit{Total Writing Time} = (\alpha \times \textit{number of active links}) \times \textit{Time to Write Data Unit} \quad (5.1)$$

In equation 5.1, α specifies a fraction of active links that constitute the communication

path between mirrors. The α can have a value of greater than zero and less than and equal to one. For the experiments presented in this thesis, α has been set to 1. The reason behind this choice is to utilize all the active links as part of the communication path. The usage of different α values, to set up different communication paths for the network, is out of the scope of the research presented in this thesis. Furthermore, for implementation of realistic impacts, time to write a data unit is randomly generated between 10 to 20 milliseconds at runtime [137]. However, this range might vary depending on the equipment installed and the mirroring protocol being used [2, 78].

In a similar way, the bandwidth consumption also depends on the number of active links. More active links indicate more transmission of data leading to higher levels of bandwidth consumption [2]. Furthermore, for implementation of realistic impacts at runtime, the Bandwidth per link is randomly generated between the range of 20 to 30 GBps [137]. However, this range might vary depending on the equipment installed [55]. The Bandwidth Consumption is calculated as:

$$\textit{Total Bandwidth Consumed} = (\alpha \times \textit{number of active links}) \times \textit{Bandwidth per link} \tag{5.2}$$

Simulation Component

The Simulation Component implements the uncertainty scenarios specifying the different dynamic environmental situations that the RDM system can face. It allows the setting of the different simulation properties, such as selecting the uncertainty scenario for experiments and the number of simulation runs.

A partial class diagram presenting the main elements of the Management Component, Network Component and Simulation Component is shown in Fig. 5.3. The *NetworkManagement* class and the *Probe* and *Effector* interfaces offer an implementation of the Management Component. The *Probe* interface provides the probing API functions that could be used by the Managing System to get information about the current topology, Current values of the network parameters (i.e. bandwidth consumption, active links and wrting time) and information about all the monitorable network parameters as a single object. On the other hand, the *Effector* interface provides API functions to change the values of the network parameters and topology for the network at a particular simulation time step. The

NetworkManagement class is used to provide the management capabilities for the network by setting of the network properties and simulation properties.

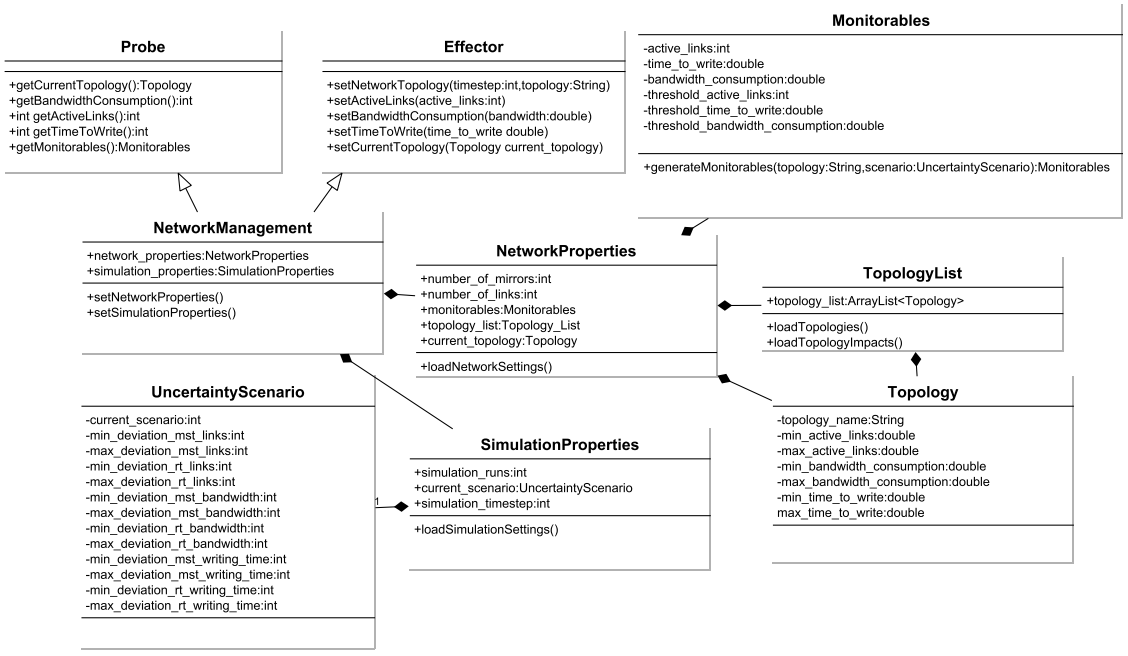


Figure 5.3: RDMSim Class Diagram

Furthermore, the classes *NetworkProperties*, *Monitorables*, *Topology* and *TopologyList* are part of the Network Component and implement the respective features of the RDM. The *NetworkProperties* class provides the functionality of initial setup of the network such as the number of mirrors, number of links and topologies and monitored network parameters. Moreover, the classes *Monitorables*, *Topology* and *TopologyList* are used to define the monitorable network parameters and implementation of the topologies and their characteristics respectively.

Finally, the Simulation Component is implemented using *SimulationProperties* and *UncertaintyScenario* classes. The *SimulationProperties* class provides functionalities with respect to the simulations to be executed including the number of simulation runs, the uncertainty scenario for the simulation and the current simulation time step. The *UncertaintyScenario* scenario class provides the functionality related to the currently selected uncertainty scenario and the deviations (minimum and maximum) in the initial values of network parameters (i.e. active links, bandwidth and writing time) when a particular topology (MST or RT) is applied under that scenario.

5.3.4 Uncertainty Scenarios

In the RDMSim, six different scenarios to specify the different dynamic environmental contexts for the RDM system have been defined. These scenarios have been designed to simulate different archetypal real situations which can affect the satisfaction of NFRs. The reason behind defining different scenarios is to facilitate the evaluation of the decision-making offered by the Pri-AwaRE approach. Moreover, it can help in assessing how the Pri-AwaRE might react under uncertain situations compared to the system working under stable conditions. The description of the scenarios is provided as follows:

Default scenario S_0: A default scenario specifying an environment envisioned by the requirements experts [53, 56] is defined for comparison purposes. Under the default scenario, the RDM system is considered to be working under stable environmental situation. Under stable conditions, the following thresholds for the satisfaction levels related to the operational cost, performance and reliability are required to be achieved: Bandwidth consumed at particular point in time should be less than or equal to 40% of the total bandwidth consumption of the entire network on average. Correspondingly, the time to write data at a particular point in time should, on average, be less than or equal to 45% of the total writing time of the entire network. In contrast, the number of active network links should be greater than or equal to 35% of the total number of network links. Under scenario S_0, the MST topology is used as the initial topology for the network.

Scenario S_1 - Unexpected Packet Loss during MST: Scenario S_1 represents the environmental situation where a phase of consecutive and unexpected packet loss is observed during the execution of MST topology. Such a situation leads to a reduction in the reliability of the system as the data packet loss specifies link failures in the RDM system. These link failures may be caused due to failures of the equipment (e.g. switch or router) or power failures [74]. Under scenario S_1, the MST topology is set as the initial topology.

Scenario S_2 - Unexpected Packet Loss during RT: The initial topology being used by the RDMSim under this scenario is RT Topology. Unexpected data packet loss during the execution of the RT generates an unusual rate of data forwarding due to more

active links. This leads to an increase in the bandwidth consumption (i.e. operational cost) and reduction in the system's performance. As mentioned earlier, in the RDM system, the inter-site links communication cost is a function of the data sent over them. Hence, the RT topology, involving more inter-site network links, is more expensive than the MST topology. The operational cost increases with an increase in the number of active links, and a reduction in the system's performance could also be expected.

Scenario S_3: Scenario S_3 represents the dynamic environmental situation where scenarios S_1 and S_2 are simultaneously observed. The initial topology for scenario S3 is selected randomly.

Scenario S_4 - MST topology execution failures: Scenario S_4 involves the behaviour specified in scenario S_1. Moreover, an increase in the bandwidth consumption (MinC) and a reduction in the system's performance (MaxP) is also observed during the execution of the MST topology. The reason behind this behaviour is due to the implementation of synchronous mirroring. In this scenario, the initial topology being used by the RDMSim is MST topology.

Scenario S_5 - RT Topology execution failures. Scenario S_5 includes the behaviour specified in scenario S_2. Furthermore, during the execution of RT topology, a reduction in the system's reliability (MaxR) is also observed because of the failures of the equipment (i.e. routers and switches). In this scenario, the initial topology being used is RT topology.

Scenario S_6 - Significant site failure. The initial topology in this scenario is selected randomly. Scenario S_6 includes the simultaneous occurrence of scenarios S_4 and S_5. It represents a significant site failure situation, where both repeated and multiple simultaneous failures are expected [74, 78]. The situation may be caused by a power outage affecting all the campus buildings or buildings within several metropolitan areas. Under such a situation, the worst-case data loss may happen in different RDM node sites. For example, a site can be down before the full information backup is shipped offsite. Site failure disasters rarely occur, having approximately a failure rate of once per year [62, 78].

The above scenarios cover almost all of the possible dynamic contexts that an RDM

network would encounter at runtime. Therefore, these scenarios have been used for experimental evaluations of the Pri-AwaRE approach which are presented in Chapter 6.

5.4 Summary

To sum up, two case studies from different domains of IoT and RDM have been selected to evaluate the Pri-AwaRE approach. The first case study is based on an existing exemplar of the DELTA-IoT, which supports simulations of the self-adaptive IoT network. The DELTA-IoT network allows the adaptation decisions at the individual node level to tune their associated network links. The second case study is based on the self-adaptive RDM network. To provide simulations for the RDM network, an exemplar tool of RDMSim has been implemented as part of the research presented in this thesis. In contrast to the DELTA-IoT, the RDMSim network allows adaptation decisions of change in topology for the entire network.

Chapter 6

Experimental Evaluation

The work presented in this chapter has been adapted from the following publications:
[136] H. Samin, L. Garcia Paucar, B. Nelly, and P. Sawyer. Towards priority-awareness in autonomous intelligent systems. In *36th ACM/SIGAPP Symposium On Applied Computing (SAC)*. ACM, 2021.
[138] H. Samin, N. Bencomo, and P. Sawyer. Decision-making under uncertainty: be aware of your priorities. *International Journal on Software and Systems Modeling (SoSyM)*, 2022.
[135] H. Samin, N. Bencomo, and P. Sawyer. Pri-aware: Tool support for priority-aware decision-making under uncertainty. In *2021 IEEE 29th International Requirements Engineering Conference (RE), Poster and Tools Demonstration Track*. IEEE, 2021.

6.1 Introduction

This chapter describes the experiments for the evaluation of the Pri-AwaRE approach. The experiments have been executed for all the scenarios defined for the two case studies of the IoT and RDM systems presented in Chapter 5. Furthermore, comparison with the state-of-the-art single-objective POMDP technique has also been provided. The approach has been further evaluated using T test [142] and Non-Inferiority Trial [140]. Section 6.2 presents the hypotheses for the experiments, followed by the experimental evaluations for both the case studies in Sections 6.3 and 6.4, respectively.

6.2 Experimental Hypotheses

In this section, the hypotheses for the experiments are defined. The Hypotheses have been designed to address the **RQs**:

RQ1: Can modelling and reasoning of the priorities of individual NFRs under uncertain environmental contexts be supported?
RQ2: Can decision-making in SASs include tuning of the NFRs' priorities to match the dynamic runtime situations?

We wish to assess how well the Pri-AwaRE approach offers *priority-aware* decision-making and informed choices related to the individual NFRs under changing environmental contexts.
Let us revisit Definition 1 *priority-awareness* as presented in Chapter 1.

"Priority-awareness is the capability of providing autonomous changes of NFRs' priorities to address the required satisfaction levels of NFRs."

Based on the above concept of *priority-awareness*, the null $\boldsymbol{H}_0$ and alternative $\boldsymbol{H}_a$ hypotheses are described as follows:

$\boldsymbol{H}_0$*: There is no difference in the quality of decision-making under uncertainty with or without Pri-AwaRE.*

$\boldsymbol{H}_a$*: There is difference in the quality of decision-making under uncertainty with or without Pri-AwaRE.*

The quality is measured in terms of the approach's capability of offering *priority-awareness* during decision-making for SASs. It means the approach should be capable of modelling and reasoning with individual priorities of NFRs. Moreover, these priorities should be tuned autonomously considering the runtime environmental contexts to achieve the required NFRs' satisfaction levels. The experimental evaluations for the case studies of

Table 6.1: States of Priority-AwaRE MR-POMDP++ for DELTA-IoT Network

S	NFR_1=MinEC	NFR_2=MinPL
s_1	True	True
s_2	True	False
s_3	False	True
s_4	False	False

IoT and RDM to test the hypotheses are presented in the following sections.

6.3 Experiments for DELTA-IoT

The first case study represents the IoT network for a smart university campus simulated using the DELTA-IoT exemplar tool [72]. The experimental setup using the DELTA-IoT exemplar is described as follows:

6.3.1 Experimental Setup

The experimental setup considers the operational model of the DELTA-IoT network as described in Chapter 5. The experiments have been executed for 100 simulation time steps for the DELTA-IoT network. Each simulation time step corresponds to 15 minutes of network activity. At each time step, the Pri-AwaRE approach is applied for each node (i.e. sensor) individually to make adaptation decisions. The nodes have ids from 2 to 15. The Pri-AwaRE makes these local node-level decisions to perform the adaptation action ITP or DTP based on the monitored link interference values. Hence, the experiments comprise a total of 1400 adaptation decisions during the 100 simulation time steps which is a considerable number to evaluate the effects of adaptations performed on the satisfaction levels of NFRs of the DELTA-IoT network. For experiments with the DELTA-IoT, the focus is on the NFRs of MinEC and MinPL. These NFRs are related to the network's quality and performance [63]. For the purpose of verification of the behavior of the DELTA-IoT under the adaptation decisions offered by Pri-AwaRE, experiments have been repeated for 5 simulation runs. The results logs for the simulation runs are provided in [133]. The experiments have been performed on a Lenovo Thinkpad with intel Core i7, 8th Gen processor and 16 GB RAM.

a) Components of MR-POMDP++ for DELTA-IoT network

The components of the MR-POMDP++ model for the considered DELTA-IoT network are explained as follows:

States: According to Rule 1 (presented in Chapter 4), for MinEC and MinPL, four states are identified as presented in Table 6.1.

Actions: Actions represent the adaptation strategies performed to achieve the satisfaction of MinEC and MinPL. The actions for the DELTA-IoT network are Increase Transmission Power (ITP) and Decrease Transmission Power (DTP). ITP supports MinPL by increasing the nodes' communication range and adjusting the distribution factor on the links. The value of the communication range is directly proportional to the transmission power [90, 91]. Hence increasing the communication range improves packet delivery performance at the expense of high energy consumption [72]. In contrast, the action DTP supports MinEC by decreasing the communication range and adjusting the links' distribution factor. A decrease in the communication range leads to a decrease in transmission power and therefore leads to a lower level of energy consumption.

Rewards: According to Rule 2 of Pri-AwaRE (presented in Chapter 4), the reward vector is used to represent the priorities of the individual NFRs. As the considered NFRs for DELTA-IoT network are 2 in number, the size of the reward vector is 2 which is presented as follows:

$$\mathbf{R(s,a)}=[\mathrm{R}_{MinEC},\mathrm{R}_{MinPL}].$$

R_{MinEC} and R_{MinPL} specify the priority values for MinEC and MinPL respectively. The reward vector values for NFRs in the DELTA-IoT network have been defined by the domain experts [155] and are shown in Table 6.2. The rewards values for the NFRs are positive to indicate the positive impact of application of the adaptation actions ITP and DTP on the satisfaction of both the NFRs MinEC and MinPL. For example, selection of ITP will have more priority for the NFR MinPL but it will still provide support for the satisfaction of MinEC.

Transition Function: On the basis of Rule 1, the states in MR-POMDP++ are specified as a combination of NFRs. Moreover, according to [112], the transition probabilities *T(s,a,s')* can be factored as marginal conditional probabilities of the NFRs (i.e. $P(MinEC' \mid MinEC, a)$ and $P(MinPL' \mid MinPL, a)$) using the property of conditional independence and Bayes rule as follows:

$$T(s,a,s') = P(s'|s,a) = P(MinEC' \mid MinEC, a)P(MinPL' \mid MinPL, a)$$

Table 6.2: Reward Values (i.e. Initial Priorities) for the NFRs in the DELTA-IoT Network

S	Action(A)	Reward Vector Values	
		$R_{NFR1} = R_{MinEC}$	$R_{NFR2} = R_{MinPL}$
s_1	DTP	89	80
s_2	DTP	75.1	20
s_3	DTP	75	30
s_4	DTP	10	5
s_1	ITP	80	89.67
s_2	ITP	40	75
s_3	ITP	30	70
s_4	ITP	5	10

The transition probabilities for the DELTA-IoT case are presented in Appendix A. These transition probabilities are provided by the domain experts [112, 138].

Observations: As the states of the NFRs are not directly observable, the monitorables are used to acquire observations about their satisfaction based on the information accumulated from the environment. In the DELTA-IoT network, under the uncertainty scenario presented in Chapter 5, the monitorable SNR is considered to observe link interference [72]. Based on the SNR values, i.e. being less than, greater than or equal to zero, the observations fall into three categories. If the value of SNR is greater than 0, it refers to a strong signal and low susceptibility to interference. On the other hand, the value of SNR less than zero indicates a weak signal and high susceptibility to interference. Whereas, the SNR value equal to zero refers to no signal.

b) Requirements Specification

The experimental setup also considers the *Requirements* ($\boldsymbol{R}$) for the DELTA-IoT network that indicate the required satisfaction levels of NFRs. These *requirements* are defined by the experts [72]. The requirements actually specify the satisfaction thresholds indicating the suitable zone of satisfaction of NFRs. The requirements for the DELTA-IoT network provided by the experts are as follows:

R1: *Total Energy Consumption in the network should be less than or equal to 20 Coulombs i.e.* $SAT_{MinEC} <= 20$.

R2: *Total Packet Loss in the network should be less than or equal to 20 percent i.e.* $SAT_{MinPL} <= 0.20$.

6.3.2 IoT-based Experiments

The experiments have been designed to study how well Pri-AwaRE supports informed choices of priorities related to the individual NFRs, and therefore satisfy the requirements ($\boldsymbol{R}$). Due to uncertain environmental contexts, NFRs' priorities assigned at design time may no longer be valid at runtime. Hence, it would require the Pri-AwaRE to tune these priorities. For the purpose of evaluation of Hypotheses, two experiments have been designed.
i) The first experimental evaluation focuses on the assessment of the capability of Pri-AwaRE to offer priority-aware decisions and autonomous tuning of NFRs' priorities.
ii) The second experimental evaluation focuses on the assessment of the impacts of priority-aware decisions on satisfaction levels of NFRs to comply with the requirements ($\boldsymbol{R}$). Comparison to a single-objective decision-making technique, known as RE-STORM [112], is also provided. The RE-STORM approach is considered as single-objective as it is based on the single-objective POMDP and uses scalar reward to represent a combined priority for all the NFRs (as described in Chapter 3). The results have been further evaluated using T-Test [142], for computing statistical significance of the results, and the NI Trial approach [140].

The experiments are presented as follows:

IoT Experiment 1: Priority-Aware Decisions and Autonomous Tuning of NFRs' Priorities

As described earlier, for experiments, the Pri-AwaRE has been applied for each node in the network during each simulation time step. Pri-AwaRE monitors the link interference on the outgoing links for a particular node. Based on the monitored values, Pri-AwaRE decides to increase or decrease the transmission power on the links by increasing or decreasing the communication range. Therefore, the approach takes a local decision related to each node at a particular time step to configure its corresponding links, as shown in Table 6.3. For example, at time step t1, all the links for node 2 are configured by the application of the adaptation action of DTP. Consequently, the satisfaction level for MinEC becomes *33.961559* Coulombs, and the satisfaction level for MinPL becomes *0.041667* (i.e. 4.1 percent), respectively. The approach is then executed to perform adaptations for all other nodes i.e. node 3, node 4 and so on.

Table 6.3: Pri-AwaRE: Experiment Results for time step 1

Node Id	Action	$\mathbf{EU}_{MinEC}$	$\mathbf{EU}_{MinPL}$	$\mathbf{Sat}_{MinEC}$	$\mathbf{Sat}_{MinPL}$
2	**DTP**	**764.171898**	**605.188297**	**33.961559**	**0.041667**
3	ITP	650.060976	788.044468	38.485869	0.008333
4	DTP	788.335155	642.632839	36.958176	0.015385
5	ITP	650.420667	788.205516	37.720203	0.030769
6	DTP	788.338781	642.644331	29.817813	0.141667
7	**ITP**	**650.420788**	**788.205554**	**31.967988**	**0.083333**
8	**DTP**	**788.338787**	**642.644351**	**33.204893**	**0.0**
9	ITP	650.420789	788.205554	37.024339	0.030769
10	DTP	788.338787	642.644352	30.900698	0.01
11	ITP	650.420789	788.2055545	36.969265	0.146154
12	DTP	788.338787	642.644351	32.772624	0.058333
13	ITP	650.420789	788.205554	37.853066	0.046154
14	DTP	788.338787	642.644351	43.682137	0.014286
15	ITP	650.420789	788.205554	33.083955	0.009091

*EU_{NFRi} represents the expected utility value of NFRi
*Sat_{NFRi} represents satisfaction level of NFRi

The initial setup of experiments considers the initial NFRs' priorities for the DELTA-IoT network which are taken into account during the selection of adaptation actions. These initial priorities were defined in the form of rewards for the MR-POMDP++ model (presented in Table 6.2). Let us study how the priorities of NFRs represented in the form of rewards impact the decisions of action selection to configure links of a particular node in the DELTA-IoT network, as shown in Table 6.3. Considering the case of time step t1 as an example, the *expected utility* (EU) values for NFRs, using equation 4.1, are considered during the decision-making. For example, for node 2, the EU for MinEC has a higher value of *764.171898* than that for MinPL, which is *605.188297*. Therefore, the action selected for node 2 is DTP to support MinEC. On the other hand, for the links' configuration of node 7, the action of ITP is chosen based on the higher EU of *788.205554* for MinPL compared to the EU of MinEC, which is *650.420788*. The application of the decision of ITP shows an increase in the satisfaction level of MinPL from *0.083333* percent to *0.0* percent, representing an ideal situation of no packet loss, as shown in Table 6.3. Hence, it shows that *expected utility values*, representing the *autonomously tuned NFRs' priorities*, impact action selection decisions leading to priority-aware and informed choices related to the individual NFRs. This in turn provides evidence that Pri-AwaRE does provide *priority-aware* decision-making and therefore answers the **RQs**.

As a result of tuning all the nodes' link configurations, at the end of time step t1, the satisfaction levels of MinEC and MinPL become *33.083955* Coulombs and *0.009091* (i.e. 0.9 percent), respectively. This process is repeated at each time step, leading to compliance

to the required satisfaction levels for both the NFRs, as shown in Fig. 6.2. Pri-AwaRE shows an average satisfaction of *17.860959* and *0.141865* for MinEC and MinPL respectively. Therefore, it satisfies the requirements $\text{SAT}_{MinEC} <= 20$ and $\text{SAT}_{MinPL} <= 0.20$. Hence, Pri-AwaRE shows promising results in terms of satisfying both MinEC and MinPL.

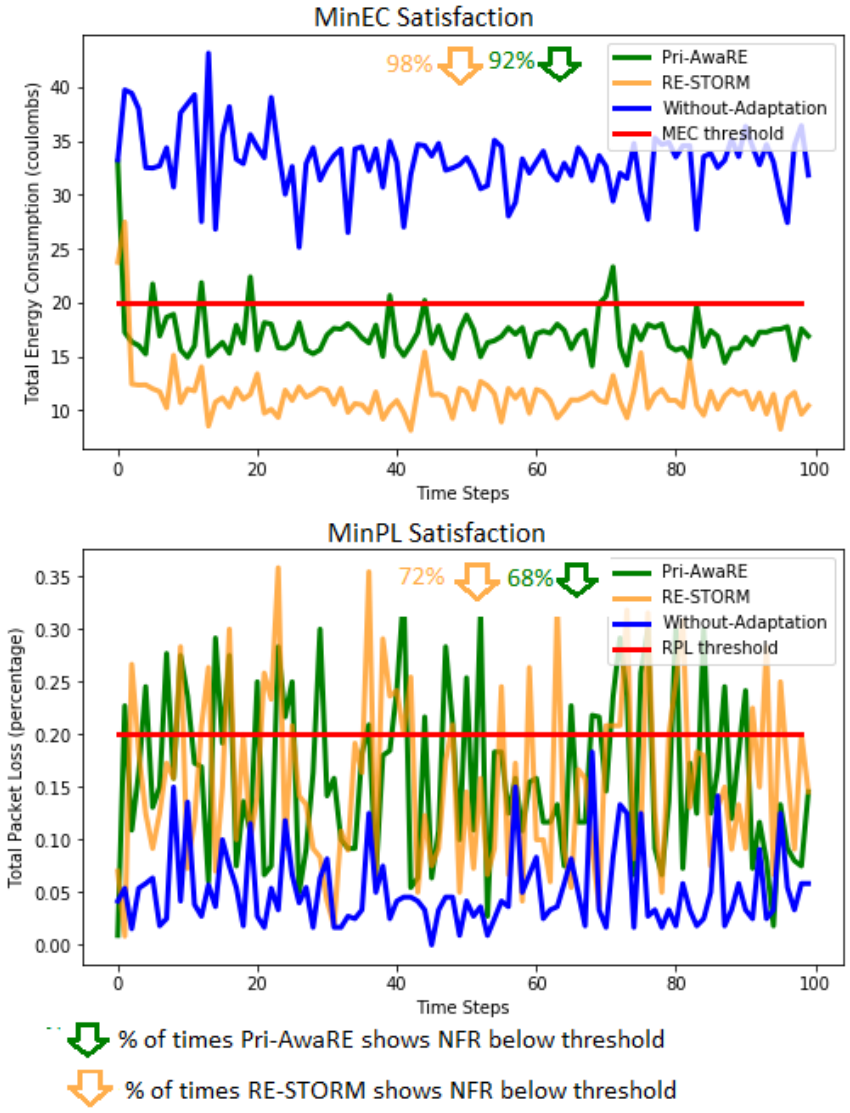


Figure 6.1: Satisfaction of NFRs over Time without adaptation, by applying Pri-AwaRE and RE-STORM.

IoT Experiment 2: Impact of Priority-Aware Decisions on satisfaction levels of NFRs

In this experiment, the impact of the priority-aware decisions, taken by the Pri-AwaRE approach, on the satisfaction of NFRs of the DELTA-IoT network is discussed. The goal

is to assess how well Pri-AwaRE achieves the required satisfaction levels of NFRs. In addition, comparison with the approach of RE-STORM [112] is also provided. RE-STORM is a single-objective POMDP based optimization technique that uses scalar reward value to represent a combined priority for all the NFRs. The details about the single-objective POMDP have been provided in Chapter 3. The approach of RE-STORM is implemented using Perseus [154], a single-objective POMDP solver. Comparison of the results with the network performing without any adaptive approach are also discussed.

Let us consider the case of the DELTA-IoT network operating without adaptation. In such a situation, the network focuses on supporting the MinPL at the expense of a higher energy consumption value. The satisfaction level of MinPL is well below the satisfaction threshold of *20* percent throughout the timeline compared to the satisfaction level of MinEC. MinEC ranges between *25.0* and *44.0* Coulombs, which is significantly higher than the required satisfaction threshold of *20* Coulombs, as shown in Fig. 6.1. In contrast, as a result of the adaptation decisions offered by Pri-AwaRE, the DELTA-IoT network showed improvement in compliance with the requirements of $\text{SAT}_{MinEC} <= 20.0$ and $\text{SAT}_{MinPL} <= 0.20$.

Furthermore, the approach also exhibits better satisfaction levels of the NFRs MinEC and MinPL compared to the approach of RE-STORM. Let us observe Fig. 6.1. Pri-AwaRE gives promising results with respect to the satisfaction of MinPL in comparison to RE-STORM. The RE-STORM results in higher levels of packet loss than Pri-AwaRE by having a packet loss above the satisfaction threshold (i.e. *20* percent of total packet loss) more often during the simulation time steps, as shown in Fig. 6.1. Let us consider the results of all the local decisions performed for the individual nodes, during each simulation time step. RE-STORM shows violations for the *25.4* percent of the times for MinPL whereas Pri-AwaRE exhibits the violations for the *21.28* percent of the times which is lower than RE-STORM. Moreover, if we consider the satisfaction levels achieved at the end of each simulation time step, Pri-AwaRE satisfies MinPL *68* percent of the simulation time steps, whereas for RE-STORM, it is *72* percent of the simulation time steps. Furthermore, after the first simulation time step, where the initial configuration for the system is done, the maximum packet loss shown by Pri-AwaRE is *0.336* (i.e. *33.6* percent), whereas in the case of RE-STORM, the maximum packet loss is *0.358* (i.e. *35.8* percent), which is higher than Pri-AwaRE. Hence, Pri-AwaRE offers comparable satisfaction levels for MinPL with RE-STORM. The difference between the average satisfaction levels for MinPL achieved by Pri-AwaRE and RE-STORM

is statistically significant having a p-value of 2.459E-151.

On the other hand, both the Pri-AwaRE and RE-STORM show comparable results concerning the satisfaction of MinEC. The satisfaction level of MinEC exhibited by Pri-AwaRE is below or closer to the threshold at almost all the time steps. At the first simulation time step, the energy consumption value is *33.083955* Coulombs which is quite high in case of Pri-AwaRE. However, at the second time step, the approach shows a reduction in the energy consumption to *17.226979* Coulombs. Hence, the Pri-AwaRE shows an improvement in the satisfaction of MinEC by complying to the requirement of $\text{SAT}_{MinEC} <= 20$, as shown in Fig. 6.1. In case of Pri-AwaRE, the total percentage violation for MinEC, as a result of all the local decisions performed for the nodes, is *14.64* percent of the times. Whereas for RE-STORM, it is *2.78* percent of the times. Furthermore, considering the results at the end of each simulation time step (presented in Fig 6.1), Pri-AwaRE has shown conformance to the requirements *92* percent of the times by having the satisfaction level below the threshold, whereas RE-STORM has shown the satisfaction of requirements *98* percent of the times. Furthermore, after the first simulation time step, where the initial configuration for the system is done, the maximum energy consumed by Pri-AwaRE is *23.31* Coulombs. In contrast, RE-STORM shows a maximum energy consumption of *27.478* Coulombs which is higher that of Pri-AwaRE. Moreover, the difference between the average satisfaction levels for MinEC achieved by Pri-AwaRE and RE-STORM is statistically significant having a p-value of 2.8398E-151.

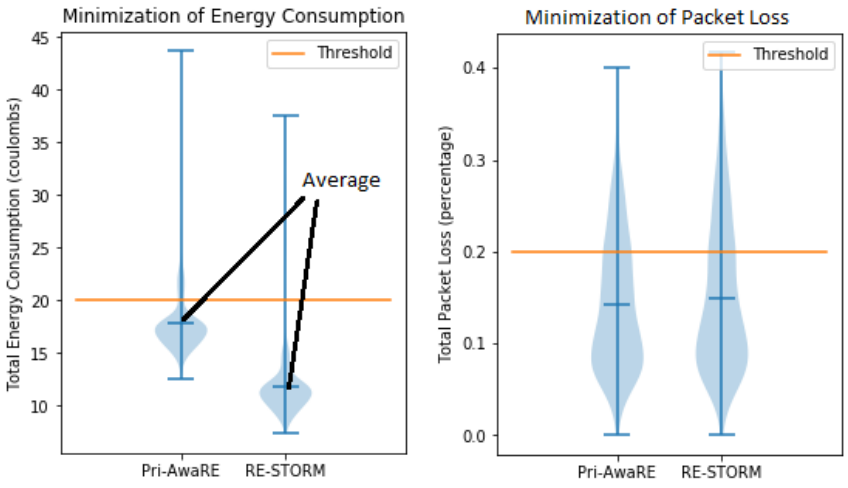


Figure 6.2: Average Satisfaction of NFRs

6.3.3 Discussion

From the results, it can be deduced that the Pri-AwaRE approach maintains the required satisfaction levels, as evident from the average satisfaction levels of MinEC and MinPL as shown in Fig. 6.2. The average satisfaction levels for the NFRs MinEC and MinPL, generated by Pri-AwaRE, are *17.860959* and *0.141865* respectively. RE-STORM also shows similar results by having the average satisfaction level *11.845* for MinEC and *0.14943* for MinPL. Moreover, the difference between the average satisfaction levels for NFRs achieved by Pri-AwaRE and RE-STORM is statistically significant having a p-value < 0.05. In case of RE-STORM, the average satisfaction level of MinEC lies between the confidence interval of *11.67242* and *12.01765*, showing a confidence level of 95 percent, with a standard error of *0.0879*, as shown in Figs. 6.3 and 6.4. Moreover, the average satisfaction level of MinPL lies between the confidence interval of *0.14523* - *0.15363* having a standard error of *0.00214*, with a confidence level of 95 percent. In comparison for Pri-AwaRE, the average satisfaction level of MinEC lies between the confidence interval of *17.6989* and *18.0229*, with a standard error of *0.0826*. Furthermore, the average satisfaction level of MinPL, exhibited by Pri-AwaRE, lies between the confidence interval between *0.1381* and *0.1457* with a standard error of *0.0019*. Hence, Pri-AwaRE offers conformance to the requirements specification of $\text{SAT}_{MinEC} <= 20$ and $\text{SAT}_{MinPL} <= 0.20$. Moreover, in comparison to RE-STORM, Pri-AwaRE offers *priority-awareness* and informed choices for selection of adaptation decisions related to the individual NFRs. Furthermore, comparison of Pri-AwaRE to RE-STORM using the approach of Non-Inferiority (NI) Trial has also been performed and Pri-AwaRE has proven to be non-inferior to RE-STORM. The results are reported in Appendix F. The results have also been provided in a GitHub repository [71].

Furthermore, the Pri-AwaRE with priority-awareness offers more flexibility in terms of balancing the satisfaction of NFRs based on their individual priorities at runtime. Consequently, the NFRs are more RELAXable [169] in the interest of better resource utilization. It means Pri-AwaRE, with its capability of autonomous tuning of priorities, can facilitate the relaxation of the priority of one NFR to satisfy another NFR considering the runtime context. Hence, based on the above, the scope of the decision-making offered by Pri-AwaRE qualifies for the systems where the NFRs can be RELAXed [168]. More specifically, the focus is on the NFRs related to the quality and performance attributes [63]. The approach

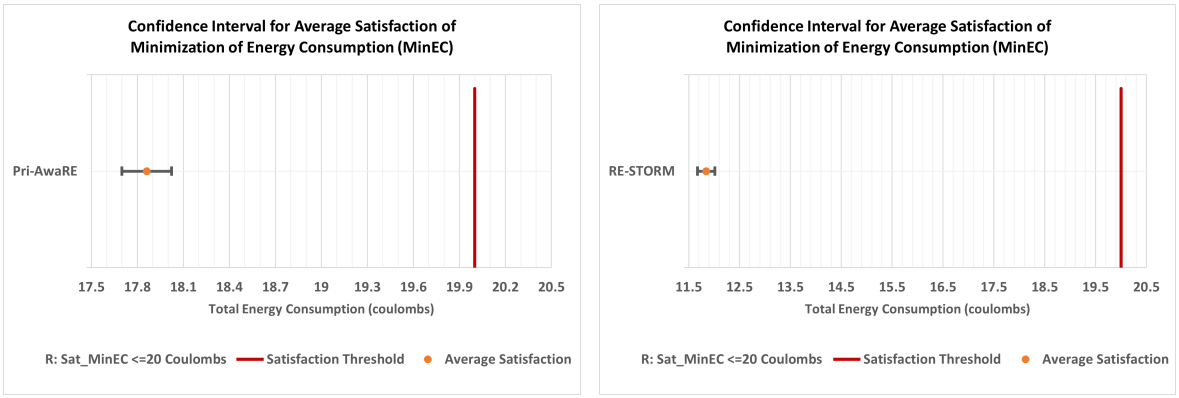


Figure 6.3: Confidence Interval for Average Satisfaction of MinEC

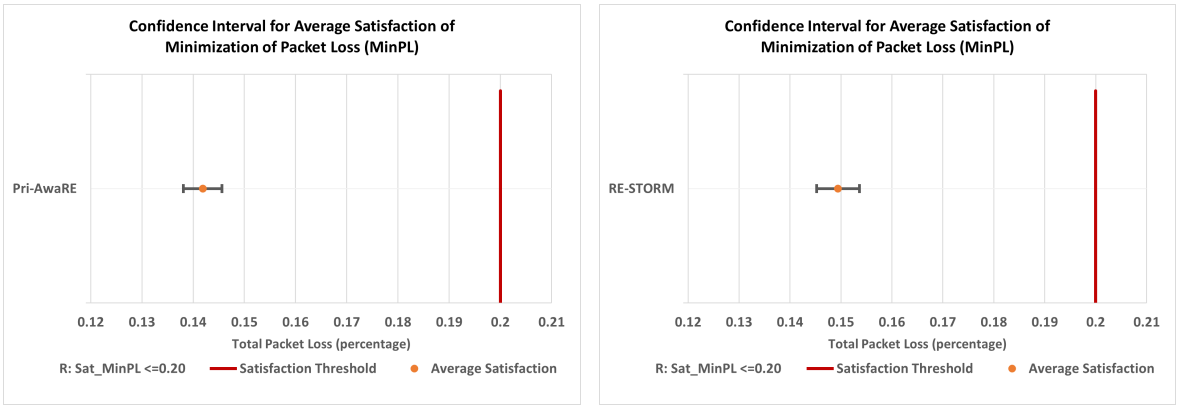


Figure 6.4: Confidence Interval for Average Satisfaction of MinPL

is not suitable for the systems where the NFRs are hard such as in safety critical systems where the experts' initial assignment of priorities must be respected.

To sum up, based on the experimental evaluations, it can be concluded that Pri-AwaRE offers priority-aware decision-making and informed choices related to the individual NFRs by autonomously tuning of the NFRs' priorities at runtime. Moreover, it also shows statistically significant better results, when compared to RE-STORM, in maintaining the required satisfaction levels of the NFRs on average. Hence, we reject Hypothesis $\boldsymbol{H}_0$ and accept $\boldsymbol{H}_a$.

6.4 Experiments for the RDM

The second case study is based on the RDM network simulated using the RDMSim exemplar. The experimental setup using the RDMSim is described as follows:

Table 6.4: States of Priority-Aware MR-POMDP++ for RDMSim Network

S	NFR_1=MinC	NFR_2=MaxR	NFR_3=MaxP
s_1	True	True	True
s_2	True	True	False
s_3	True	False	True
s_4	True	False	False
s_5	False	True	True
s_6	False	True	False
s_7	False	False	True
s_8	False	False	False

6.4.1 Experimental Setup

The RDMSim [137] exemplar considers the RDM network based on the operational model presented in [74, 78]. The experimental setup considers a fully connected RDMSim network of 25 RDM mirrors and 300 network links. The experiments involve the execution of the decisions having a global impact of change in topology (MST or RT) of the network. These decisions have an effect on the satisfaction levels of the NFRs MinC, MaxR and MaxP. The satisfaction thresholds for the NFRs are selected based on the requirements provided by the RDMSim experts as described in [137]. The experiments have been executed for 500 simulation time steps. For the experiments presented in this thesis, 1 simulation time step is considered to represent 1 hour of network activity [138]. Therefore, an adaptation decision taken by Pri-AwaRE represents a decision to change the network topology after an hour of network activity. The experiments have been performed on a Lenovo Thinkpad with intel Core i7, 8th Gen processor and 16 GB RAM. Next, the components of the MR-POMDP++ model and the requirements specifications for the RDMSim network are described.

a) Components of MR-POMDP++ for the RDMSim network

The components of MR-POMDP++ for the considered RDMSim network are as follows:

States: For the RDMSim, three NFRs of MinC, MaxR and MaxP are considered. According to Rule 1, for these three NFRs, eight states are identified as presented in Table 6.4.

Actions: Actions are performed to achieve satisfaction of the NFRs MinC, MaxR and MaxP. These actions specify the adoption of one of the two topologies: Minimum Spanning Tree (MST) and Redundant Topology (RT).

Rewards: According to Rule 2, the reward vector is used to represent the priorities of the individual NFRs. As the considered NFRs for RDMSim network are 3 in number,

Table 6.5: Reward Values (i.e. Initial Priorities) for the NFRs in the RDM Network

S	Action(A)	Reward Vector Values		
		$R_{NFR1} = R_{MinC}$	$R_{NFR2} = R_{MaxR}$	$R_{NFR3} = R_{MaxP}$
s_1	MST	39.17	39.0	40.0
s_2	MST	41.0	40.0	39.0
s_3	MST	39.0	38.0	38.5
s_4	MST	17.0	16.0	15.0
s_5	MST	44.0	43.0	43.5
s_6	MST	29.0	28.0	27.0
s_7	MST	14.0	13.0	13.5
s_8	MST	2.0	1.0	1.0
s_1	RT	41.0	43.0	41.0
s_2	RT	32.0	33.0	31.0
s_3	RT	28.0	29.0	27.0
s_4	RT	26.0	27.0	25.0
s_5	RT	28.0	29.0	27.0
s_6	RT	16.0	17.0	15.0
s_7	RT	23.0	24.0	22.0
s_8	RT	11.0	12.0	10.0

the size of the reward vector is 3 which is presented as follows:

$$\mathbf{R(s,a)}=[\mathrm{R}_{MinC},\mathrm{R}_{MaxR},\mathrm{R}_{MaxP}].$$

Where R_{MinC}, R_{MaxR} and R_{MaxP} specify the priority values for MinC, MaxR and MaxP respectively. The values of the reward vector for NFRs in the RDM network are shown in Table 6.5. These values have been defined by the domain experts considering the simulating environment of RDMSim [137, 138].

Transition Function: According to Rule 1, the states in MR-POMDP++ are specified as a combination of NFRs. Similar to the case of the DELTA-IoT network, the transition probabilities *T(s,a,s')* have been factored as marginal conditional probabilities of NFRs (i.e. $P(MinC' \mid MinC, a)$, $P(MaxR' \mid MaxR, a)$ and $P(MaxP' \mid MaxP, a)$) using the property of conditional independence and Bayes rule [112] as follows:

$$T(s, a, s') = P(s'|s, a) = P(MinC' \mid MinC, a)P(MaxR' \mid MaxR, a)P(MaxP' \mid MaxP, a)$$

The transition probabilities for the RDMSim network are presented in Appendix A. These transition probabilities are provided by the domain experts [56].

Observations: Due to the partial observable states of the NFRs, the monitorables are used to acquire observations of the satisfaction levels of NFRs based on the information accumulated from the environment. In case of the RDMSim, three network parameters of *Total Bandwidth Consumption* (TBC), *Active Network Links* (ANL) and *Total Time to Write Data* (TTW), related to the NFRs of MinC, MaxR and MaxP, respectively, are considered as the

monitorable variables. A high value of the ANL parameter is a proxy for a high satisfaction level of MaxR. On the other hand, low values of TBC and TTW imply higher satisfaction levels for MinC and MaxP. In the RDMSim network, all of these monitorables have range boundaries which are assigned by the domain experts [56, 137] as shown in Table A.5.

Similar to the transition model, the observation model is also factored into the product of conditional probabilities [112] as follows:

$$O(s', a, z) = P(z|s', a) = P(Mon_1, ..Mon_n|s', a)$$

Therefore,

$$P(z|s', a) = P(TBC, ANL, TTW|s', a) = P(TBC|s', a)P(ANL|s', a)P(TTW|s', a)$$

The observation probabilities for the RDMSim network, defined the domain experts [56, 112], are presented in Appendix A.

b) Requirements Specification

Similar to the case of DELTA-IoT, the experimental setup for RDMSim also considers the *Requirements* (***R***) to indicate the required satisfaction levels of the NFRs. The requirements for the RDMSim network, provided by the experts of RDM, are as follows:

R1: *The bandwidth consumption should be less than or equal to 40 percent of total bandwidth consumption to satisfy MinC.*
R2: *The time to write data should be less than or equal to 45 percent of total writing time to satisfy MaxP.*
R3: *The number of active links should be greater than or equal to 35 percent of total links to satisfy MaxR.*

For an RDMSim network comprising 25 mirrors, the required satisfaction thresholds for MinC, MaxR and MaxP are $SatMinC <= 3600GBps$, $SatMaxR >= 105$ active links and $SatMaxP <= 2700$ milliseconds respectively. The NFRs are considered in their poor zone of satisfaction if they do not meet the required thresholds for satisfaction.

Table 6.6: Experiment Results for time steps 158 - 164 under Scenario S_0

Time	Action	$\mathbf{EU}_{MinC}$	$\mathbf{EU}_{MaxR}$	$\mathbf{EU}_{MaxP}$	$\mathbf{Sat}_{MinC}$	$\mathbf{Sat}_{MaxR}$	$\mathbf{Sat}_{MaxP}$
158	MST	391.7338	388.0187	394.9388	1276	58	638
159	MST	391.4357	387.7522	394.8336	3720	124	1984
160	**MST**	**391.6977**	**388.0349**	**395.1344**	2320	80	880
161	MST	391.4272	387.7457	394.8357	2376	99	1881
162	MST	391.4207	387.7397	394.8322	2376	108	1728
163	**RT**	**386.2634**	**391.7545**	**385.8755**	**2160**	**80**	**1440**
164	MST	391.6597	387.9502	394.9176	3500	175	3150

*EU_{NFRi} represents the expected utility value of NFRi
*Sat_{NFRi} represents satisfaction level of NFRi

6.4.2 RDM-based Experiments

The experiments have been designed to study how well Pri-AwaRE supports informed choices of priorities related to the individual NFRs, and therefore satisfy the requirements (R). Similarly to the case of DELTA-IoT, experiments for the RDMSim have been designed to study how well Pri-AwaRE supports informed choices of priorities related to the individual NFRs, and therefore satisfy the requirements (R). The following two experiments have been performed to evaluate the hypotheses considering all the dynamic scenarios (presented in Chapter 5) for the RDMSim. The results have been further evaluated using T-Test [142], for computing statistical significance of the results, and the NI Trial approach [140].

RDM Experiment:1– Priority-Aware Decisions and Autonomous Tuning of NFRs' Priorities

Experiment 1 demonstrates *priority-awareness* during the decision-making offered by the Pri-AwaRE approach. It also shows how the approach supports compliance with the requirements specification by autonomously tuning of the NFRs' priorities. For the purpose of making priority-aware decisions, Pri-AwaRE makes use of MR-POMDP++ to model the distinct priorities of NFRs in the form of rewards. The experiment studies the way the individual NFRs' priorities are considered by the Pri-AwaRE during the selection of the adaptation action as shown in Table 6.6. For example, at *time step 160*, Pri-AwaRE offers the best possible trade-off in the form of a decision to use MST as the preferred topology instead of RT. This is due to the fact that the *expected utility values* (*EUs*) for MinC and MaxP are *391.6977* and *395.1344* respectively, which are higher than the *EU* for MaxR which is *388.0349* as presented in Table 6.6. Conversely, at *time step 160*, the *EUs* in case of RT

topology were *385.9493*, *385.5438* and *391.3854* for MinC, MaxP and MaxR respectively. As MST offered higher impacts than RT, Pri-AwaRE decided to select MST as the preferred topology to support the reduction of the operational cost and improvement in the network's performance. This means that MST is selected considering the priorities for satisfaction (represented by EU) of MinC and MaxP which is higher than the priority of MaxR. As a result of this decision, the MST topology is configured for the network. Therefore, to offer priority-awareness, the decisions made by Pri-AwaRE make the system aware of the explicit impacts that the adaptation decisions have on the NFRs' satisfaction, as presented in Table 6.6.

In contrast, at *time step 163*, the decision of switching from MST to RT topology is taken. The reason behind this decision is that the EU of MaxR (i.e. *391.7545*) is higher than the *EUs* of MinC (i.e. *386.2634*) and MaxP (i.e. *385.8755*) as presented in Table 6.6. As a result of the application of RT topology, a rise in the network's reliability is observed due to increase in the number of active links from *80* to *175*, as shown in Table 6.6. Hence, it shows compliance to the required threshold for satisfaction which is $SatMaxR >= 105$ active links. Therefore, the Pri-AwaRE approach considers these *EUs* representing the *autonomously tuned priorities* of NFRs during the decision-making process. This *autonomous tuning* offered by Pri-AwaRE during the decision-making process helps a SAS in making priority-aware decisions.

Similar to the case of DELTA-IoT, the initial setup of experiments considers the initial NFRs' priorities for the RDM network. These initial priorities were set by the experts [136, 138] by taking into account the different foreseen runtime contexts. As per the rules defined in Chapter 4, this initial set of priorities was defined as rewards for the MR-POMDP++ model (presented in Table 6.5). According to the runtime situations, these pre-defined priorities are tuned autonomously by Pri-AwaRE by computation of the separate EU for each NFR using equation 4.1. This tuning of individual priorities by Pri-AwaRE helps the SASs comply with the *requirements* ($\boldsymbol{R}$) by achieving high levels of satisfaction for NFRs.

RDM Experiment:2–Impacts of Priority-Aware Decisions on satisfaction levels of NFRs

Experiment 2 studies the impacts of the *priority-aware* decisions offered by Pri-AwaRE on satisfaction of NFRs under the dynamic scenarios of the RDMSim. The goal is to assess how well Pri-AwaRE achieves the required satisfaction levels of NFRs. Similar to the case of DELTA-IoT, the results of Pri-AwaRE have been compared with the existing single-objective technique of RE-STORM [112]. The results presented in Table 6.6 show the effects of applying the selected topology for the network on the NFRs' satisfaction levels at a particular simulation time step. This decision of topology selection considers the individual *EUs* for NFRs (i.e. the autonomously tuned NFRs' priorities). The experiment results for all the scenarios are shown in Figs. 6.5 to 6.11.

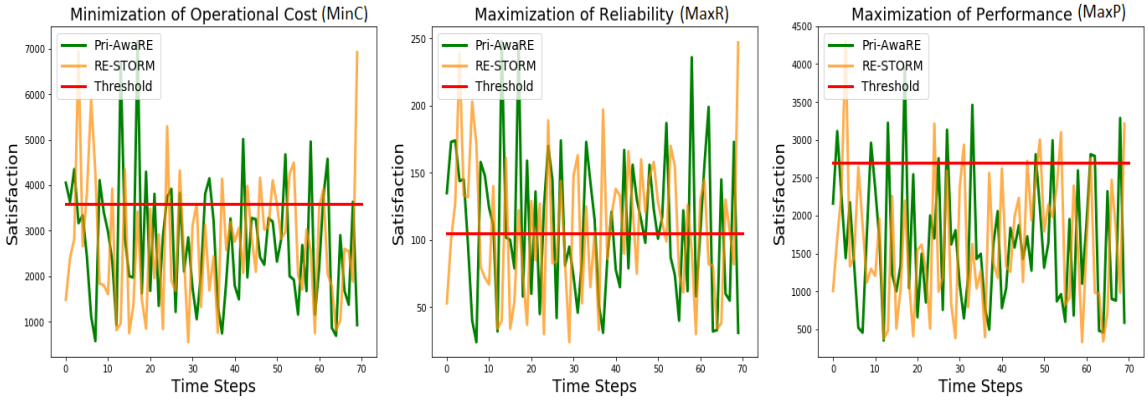


Figure 6.5: Satisfaction of NFRs over Time under Scenario S_0

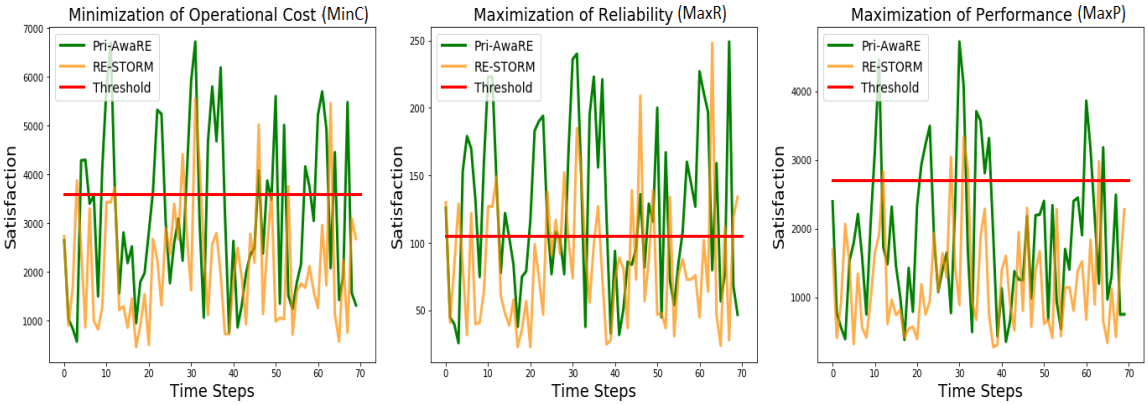


Figure 6.6: Satisfaction of NFRs over Time under Scenario S_1

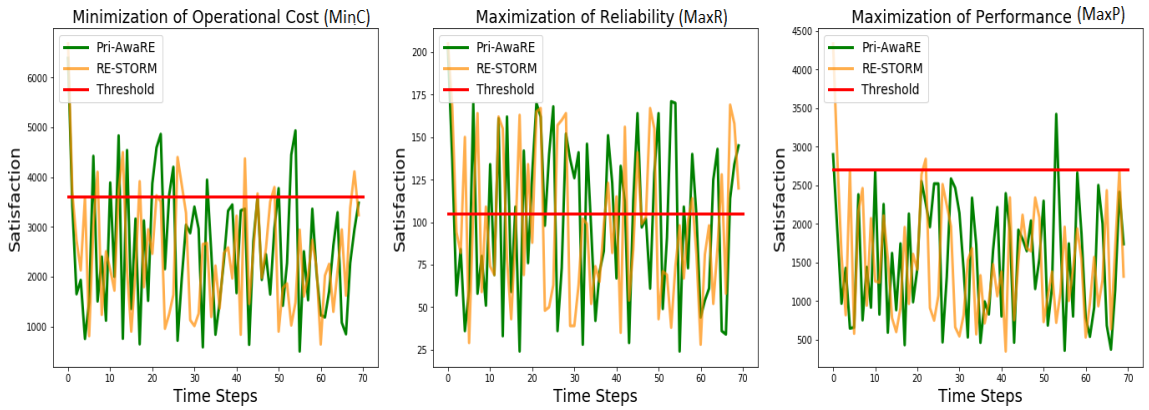


Figure 6.7: Satisfaction of NFRs over Time under Scenario S_2

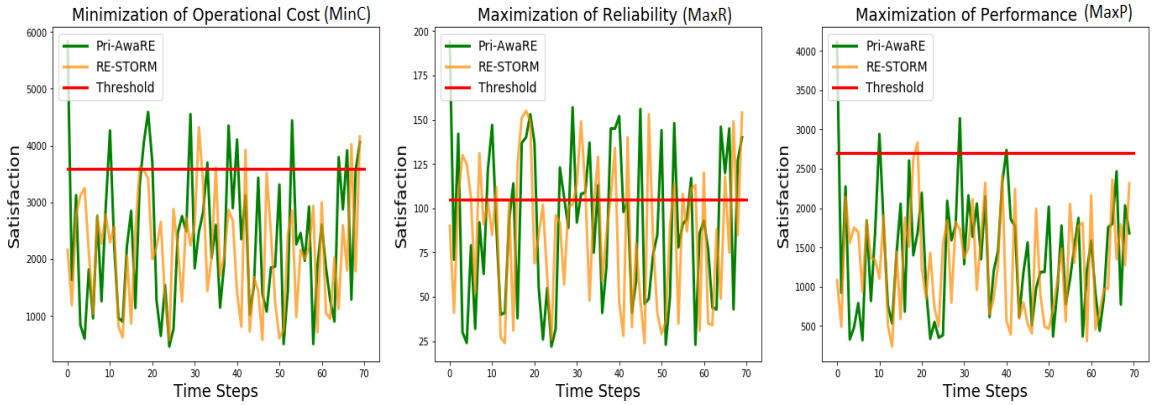


Figure 6.8: Satisfaction of NFRs over Time under Scenario S_3

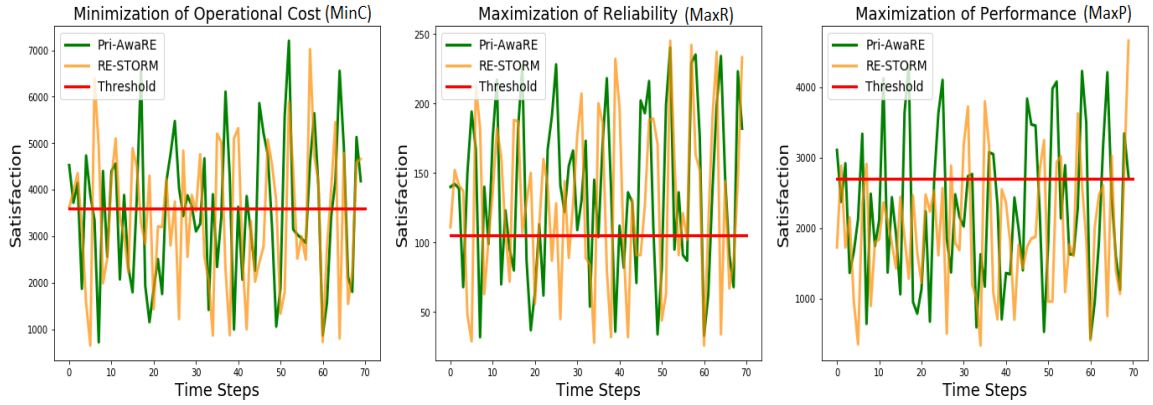


Figure 6.9: Satisfaction of NFRs over Time under Scenario S_4

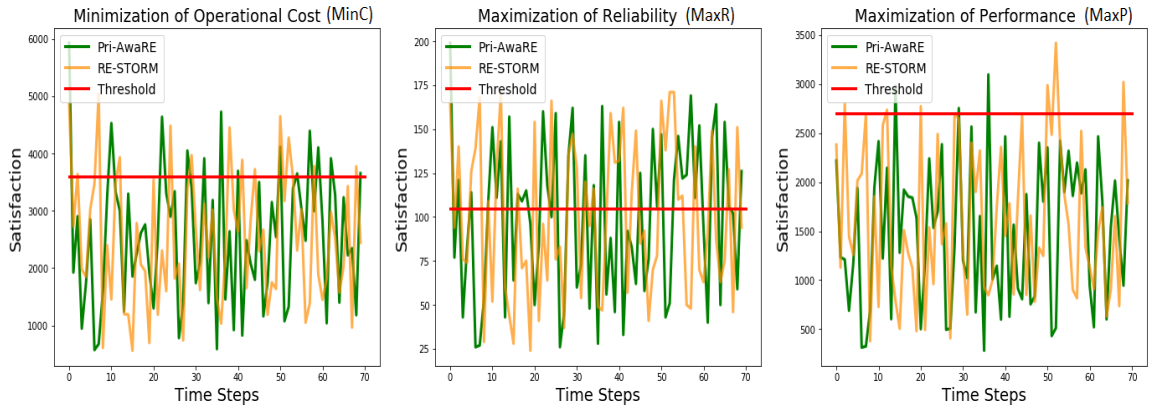


Figure 6.10: Satisfaction of NFRs over Time under Scenario S_5

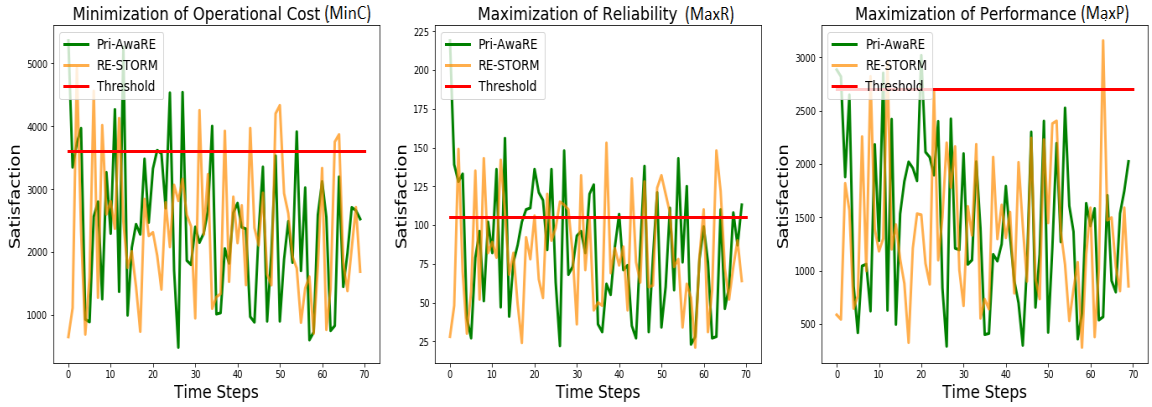


Figure 6.11: Satisfaction of NFRs over Time under Scenario S_6

Let us observe Figs. 6.5 to 6.11 representing the results of Pri-AwaRE and RE-STORM under i) the stable scenario S_0, and ii) the dynamic scenarios S_1 to S_6. Under all the scenarios, both Pri-AwaRE and RE-STORM show comparable results by showing compliance with the requirements i.e. achieving the required satisfaction thresholds of $SatMinC <= 3600$ GBps, $SatMaxR >= 105$ active links and $SatMaxP <= 2700$ milliseconds. Both the approaches show a preference for the MST topology under all the scenarios as shown in Figs. 6.12 and 6.13. Under scenario S_0, for the 500 simulation time steps, the percentage usage of MST topology by Pri-AwaRE is *90.6* percent and RE-STORM uses the MST topology *90.2* percent of times. Furthermore, both Pri-AwaRE and RE-STORM show an increase in the usage of MST topology under dynamic scenarios of S_2, S_3, S_5 and S_6 as shown in Figs. 6.12 and 6.13. Under these situations, selection of MST is the most suitable decision as it provides lower levels of operational costs and better performance [56, 137]. Moreover, it also helps in maintaining the minimal level of reliability for the network. Furthermore, to

support reliability, the approach of Pri-AwaRE shows an increase in the usage of RT topology (which was *9.4* percent under scenario S_0) to *33.2* and *39.4* percent under scenarios S_1 and S_4, respectively. This is the expected behavior by the self-adaptive RDM network under the dynamic contexts of S_1 and S_4 as defined by the experts [53, 137].

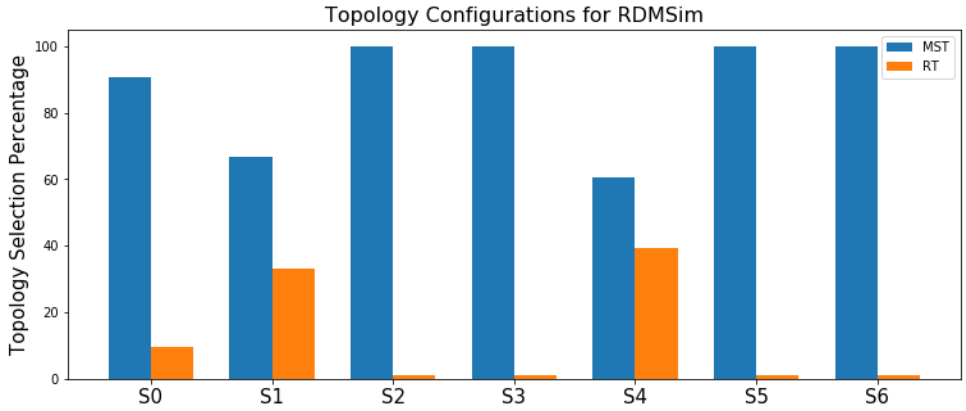


Figure 6.12: Topology Selection by Pri-AwaRE under Scenarios S0 to S6

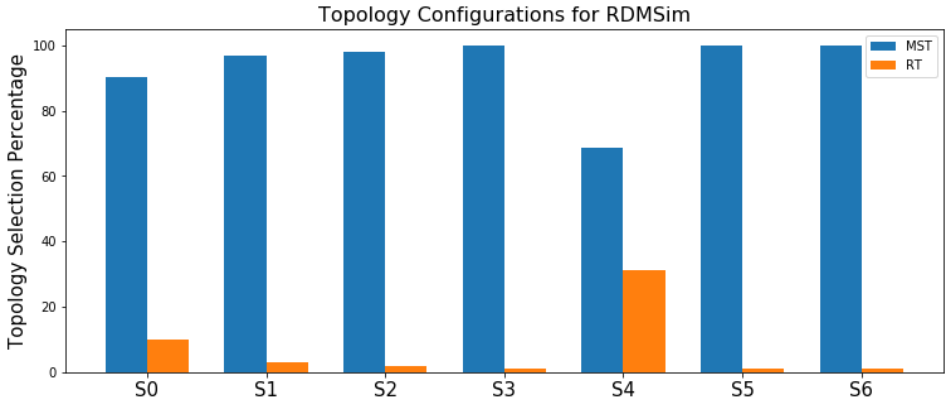


Figure 6.13: Topology Selection by RE-STORM under Scenarios S0 to S6

Similar to the case of Pri-AwaRE, RE-STORM also shows an increase in the usage of RT topology to *31.4* percent under scenario S_4. However, it shows a decrease in the usage of RT topology under scenario S_1. As a consequence of the behavior of RE-STORM under S_1, the reliability of the network is affected as shown in Fig. 6.6. Under S_1 and S_4, the percentage violations for MaxR, exhibited by Pri-AwaRE, during the 500 simulation time steps are *41.8* and *37.4*, respectively. Compared to Pri-AwaRE, RE-STORM shows higher violations for MaxR under S_1 and S_4. The percentage violations exhibited by RE-STORM under S_1 and S_4 are 60.4 and 45.6, respectively. Hence, Pri-AwaRE shows better levels of satisfaction for

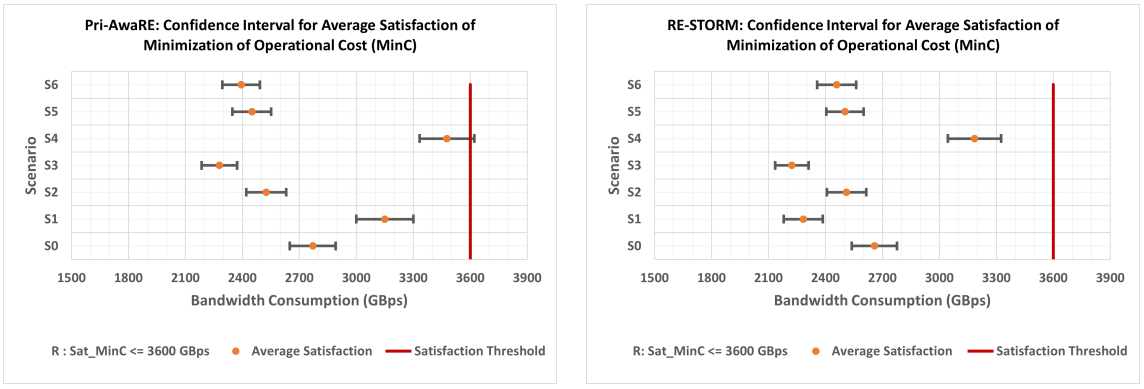


Figure 6.14: Confidence Interval for Average Satisfaction of MinC under Scenarios S0 to S6

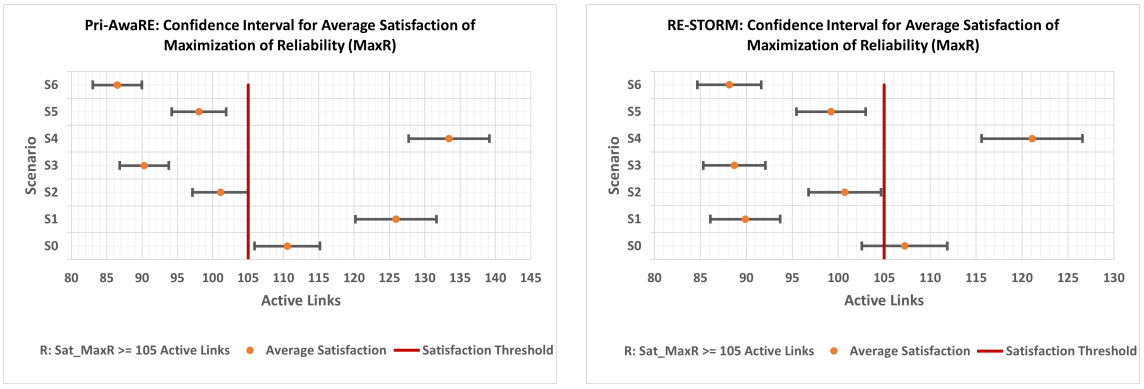


Figure 6.15: Confidence Interval for Average Satisfaction of MaxR under Scenarios S0 to S6

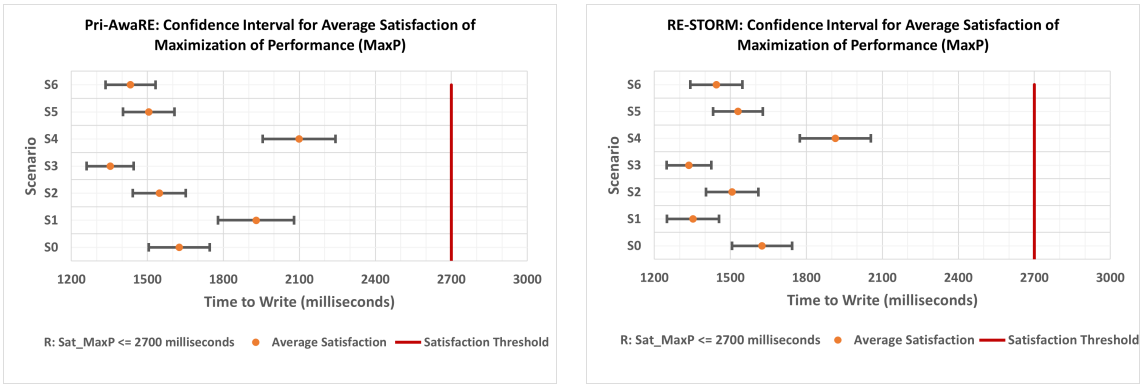


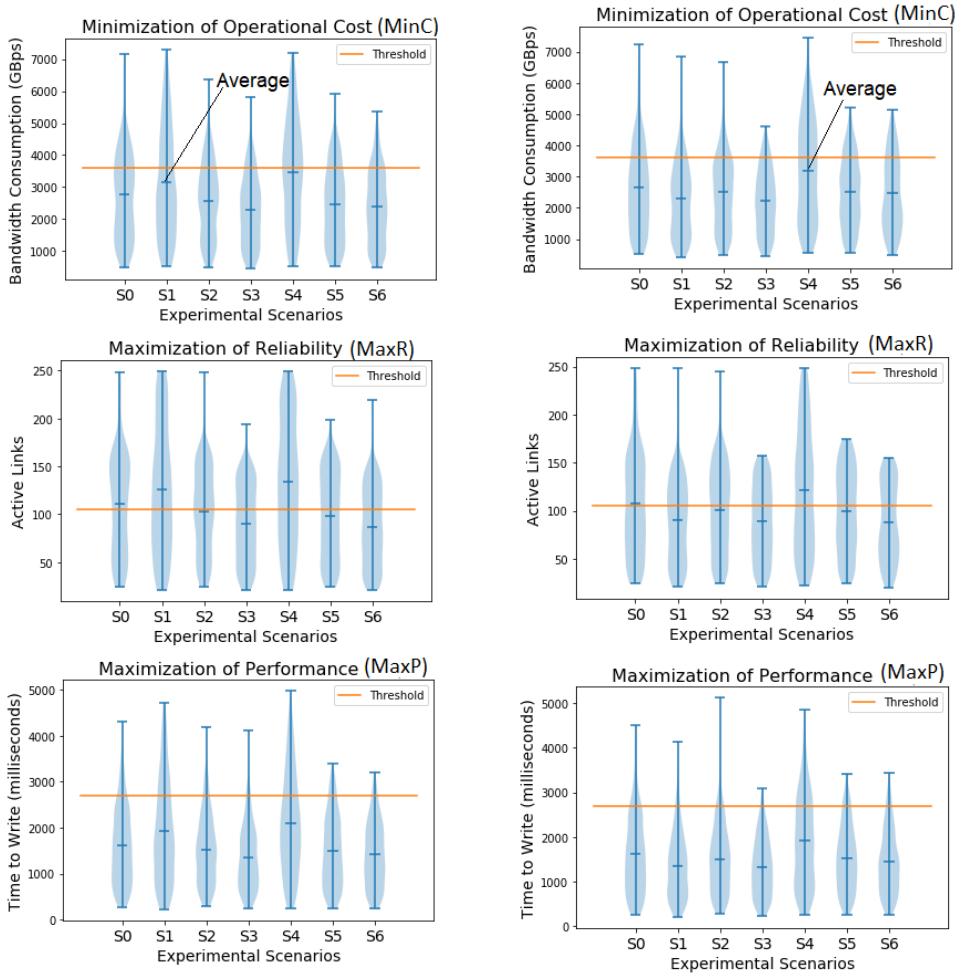
Figure 6.16: Confidence Interval for Average Satisfaction of MaxP under Scenarios S0 to S6

MaxR as compared to RE-STORM under both S_1 and S_4. Moreover, under these scenarios, the percentage violations for MinC and MaxP exhibited by Pri-AwaRE are *47.4* and *28.79* respectively under S_4, and *37.0* for MinC and *25.4* for MaxP under S_1. Whereas RE-STORM shows a percentage violation of *14.0* for MinC and *5.0* for MaxP under S_1, and *41.4* for MinC and *24.2* for MaxP under S_4. This is due to the fact that RE-STORM shows more support to MinC and MaxP under these dynamic contexts which is not the expected

behaviour under these scenarios. Let us recall that S_1 and S_4 represent environmental situations where a phase of consecutive and unexpected packet loss is observed due to link failures during the execution of MST topology and in case of S_4, increase in bandwidth consumption and writing time is also observed. Due to link failures, reliability of the network is affected. Hence, the prioritized requirement under both the scenarios is MaxR. From the topology selection behavior of RE-STORM (presented in Fig. 6.13), it is evident that the RE-STORM approach is not making informed decision by not realizing the priority for satisfaction for MaxR. Therefore it leads to reduction in the satisfaction of MaxR as shown in Fig. 6.15. In comparison, Pri-AwaRE is making informed choice by giving priority to MaxR, and therefore achieves the required satisfaction level of $SatMaxR >= 105$. Furthermore, for the purpose of verification of the behavior of the RDMSim under decisions offered by both the approaches, experiments have been executed for 5 different number of simulation runs for each dynamic scenario. The experiments have shown similar results for all the 5 simulation runs. The results logs for the simulation runs are provided in [133].

6.4.3 Discussion

In summary, the experiment results show that Pri-AwaRE offers compliance with the requirements specification for the NFRs under almost all of the scenarios. This is evident from the average satisfaction levels of NFRs presented in Fig. 6.17. The results show a confidence level of 95 percent. The confidence intervals along for average satisfaction of NFRs under all the dynamic scenarios are presented in Figs. 6.14, 6.15 and 6.16. Let's observe the results for MinC and MaxP. Under all the scenarios, the average satisfaction levels for MinC and MaxP are below the satisfaction thresholds. Therefore, Pri-AwaRE meets the threshold requirements of $SatMinC <= 3600$ GBps and $SatMaxP <= 2700$ milliseconds. For example, under scenario S_1, the average satisfaction level for MinC is *3149.716* GBps with a standard error of *76.4134* as shown in Fig. 6.14. Moreover, for MaxP, Pri-AwaRE shows an average satisfaction of *1929.314* milliseconds with a standard error of *49.4895* as presented in Fig. 6.16. The Pri-AwaRE also exhibits compliance with the threshold requirements for MaxR. For example, under scenarios S_0, S_1 and S_4, the average satisfaction for MaxR is *110.556*, *125.928* and *133.432*, respectively. Hence, it satisfies the requirement of $SatMaxR >= 105$ active links as shown in Fig. 6.15. Furthermore, Pri-AwaRE approach shows similar results under other scenarios as well. Under scenarios S_2 and S_5, the average satisfaction level for



(a) Pri-AwaRE **(b)** RE-STORM

Figure 6.17: Average Satisfaction of NFRs under Scenarios S0 to S6

MaxR is closer to the satisfaction threshold. In case of S_2, the average satisfaction level is *101.096* active links with a standard error of *2.0134*. Whereas for S_5, the satisfaction average is *98.054* active links with a standard error of *1.9662*. The exception lies in case of S_3 and S_6. Under these scenarios, the average satisfaction for MaxR is *90.296* and *86.484* with confidence intervals of *86.8165 - 93.7755* and *83.0079 - 89.9601*, respectively. These satisfaction values for MaxR under the dynamic scenarios of S_3 and S_6, where situations such as significant site failures are observed, are considered to be usual. Moreover, under such circumstances, the preference is given to support MinC and MaxP more than MaxR as presented by the topology selection behavior presented in Fig. 6.12. However, the average satisfaction level for MaxR under the scenarios S_3, S_5 and S_6 are below the satisfaction threshold of *105 active links*, Pri-AwaRE shows significantly better results with respect to

satisfying MaxR compared to RE-STORM. The difference between the average satisfaction levels for MaxR achieved by Pri-AwaRE and RE-STORM is statistically significant having a p-value < 0.05. The results for RE-STORM are described next.

The RE-STORM approach shows similar results as Pri-AwaRE concerning the satisfaction of MinC and MaxP under all the scenarios. The average satisfaction levels for NFRs exhibited by RE-STORM are presented in Fig. 6.17. For example, under scenario S_1, the average satisfaction for MinC is *2282.842* GBPs with a standard error of *52.3704*. Moreover, for MaxP the average satisfaction level is *1352.444* milliseconds with a standard error of *32.1922*. The confidence interval for MinC and MaxP are *2179.9483 - 2385.7357* and 1289.1949 - 1415.6931 respectively, as shown in Figs. 6.14 and 6.16. Hence, the average satisfaction level for both MinC and MaxP, achieved by RE-STORM, comply with the required thresholds of $SatMinC <= 3600$ GBps and $SatMaxP <= 2700$ milliseconds. The confidence intervals along with the standard error for average satisfaction of NFRs under all the scenarios are shown in Figs. 6.14, 6.15 and 6.16. Furthermore, RE-STORM also shows similar results to Pri-AwaRE in terms of satisfying MaxR under all the scenarios. The exception lies in case of scenario S_1. Under S_1, RE-STORM shows a satisfaction average of *89.878* active links for MaxR with a confidence interval of *86.0913 - 93.6647*. Whereas in case of Pri-AwaRE, the satisfaction average for MaxR is *125.928* active links with a confidence interval of *120.1966 - 131.6594*. Hence, Pri-AwaRE shows higher reliability levels than RE-STORM under S_1. Furthermore, under scenarios S_0 and S_4, the average satisfaction levels of MaxR, exhibited by RE-STORM, are *107.216* and *121.096* respectively. These satisfaction values are above the satisfaction threshold and comply with the requirements. For rest of the scenarios, RE-STORM doesn't show compliance with the requirement of $SatMaxR >= 105$ active links as shown in Fig. 6.17. Moreover, under all the scenarios, Pri-AwaRE is making informed choices for adaptation decisions by realizing the priorities of the NFRs based on the environmental situations. Under scenarios S_2, S_3, S_5 and S_6 Pri-AwaRE is showing more support to the NFRs MinC and MaxP than MaxR which is the required behaviour under these scenarios. This is due to the fact that these scenarios represent environmental situations that cause reduction in the satisfaction of MinC and MaxP (see Section 5.3.4). On the other hand, under scenarios S_1 and S_4, Pri-AwaRE shows more support to the NFR MaxR as the environmental context presented by these scenarios reduces the reliability of the network. Hence, Pri-AwaRE realizes the satisfaction priorities

for the NFRs under all these scenarios and makes informed choices by showing required topology selection behaviours as shown in Fig. 6.12. In comparison, RE-STORM, being a single-objective approach, doesn't realize the satisfaction priority for MaxR under S_1 and S_4. This is evident by the topology selection behaviour of the RE-STORM presented in Fig. 6.13. As a result of this, the satisfaction of MaxR is not achieved and RE-STORM doesn't show compliance to the requirement $SatMaxR >= 105$.

To further compare the results of Pri-AwaRE to RE-STORM, evaluation using the approach of NI Trial has also been carried out and the results show that Pri-AwaRE in non-inferior to RE-STORM. The results are reported in Appendix F. The results are also provided in a GitHub repository [71].

Hence, from the experimental evaluations, it can be concluded that Pri-AwaRE offers more *awareness* to the decisions by using the individual NFRs' priorities during decision-making. Furthermore, under all the scenarios, Pri-AwaRE shows no significant difference compared to RE-STORM in terms of satisfying the NFRs (MinC, MaxR and MaxP). Moreover, under some environmental scenarios, Pri-AwaRE shows even better satisfaction levels for NFRs (such as MaxR under Scenarios S_1 and S_4) compared to the single-objective technique of RE-STORM. To further evaluate the approach, comparison with another single-objective POMDP based technique known as RE-STORM-ARROW [113] has also been performed. The comparison results are reported in Appendix B, and show that the Pri-AwaRE performs better than RE-STORM-ARROW.

6.5 Summary

To sum up, the Pri-AwaRE approach provides a framework that considers the individual NFRs' priorities during the decision-making process of SASs. The approach also has the capability of autonomously tuning these individual NFRs' priorities under uncertain environmental contexts. As a proof of concept, Pri-AwaRE has been applied to the two case studies from the IoT and RDM domains. The results have shown that the Pri-AwaRE based decision-making offers compliance with the requirements ($\boldsymbol{R}$), and out-performs single-objective POMDP based techniques.

Chapter 7

Validation of Results

This chapter aims to assess the quality of the decision-making offered by the Pri-AwaRE approach. For this purpose, validation of the experiments' results reported in Chapter 6 is presented. The chapter is organized as follows:

- Section 7.1 presents the evaluation of the Extent of Satisfaction of NFRs using the quantitative approach of DeSiRE [44].
- Section 7.2 provides an evaluation of the fidelity of the belief satisfaction probabilities maintained by the Pri-AwaRE approach.
- Section 7.3 discusses a comparison with the related work.

Further, threats to the validity of the approach are presented in Section 7.4.

7.1 Evaluation of Extent of Satisfaction of NFRs provided by Pri-AwaRE

Pri-AwaRE has been applied to two case studies to perform decision-making. The goal is to achieve the required satisfaction levels of NFRs. An immediate question the reader can ask is: *"To what Extent have the NFRs been Satisfied when using Pri-AwaRE?"*

To answer this question, the quantification approach of *Degrees of Satisfaction in Requirements Engineering* (DeSiRE) [44] has been used. The Pri-AwaRE results for both case studies (presented in Chapter 6) have been evaluated by computing the Extent of Satisfac-

tion (*ExS*) of the NFRs using DeSiRE. Next, a description of the DeSiRE approach and experimental evaluations for both the case studies are presented.

7.1.1 Degrees of Satisfaction in Requirements Engineering (DeSiRE)

Degrees of Satisfaction in Requirements Engineering (DeSiRE) [44] is a statistical approach used to quantify the level of satisfaction of NFRs. It measures the extent to which an NFR is satisfied or violated. Most multi-objective decision-making techniques treat the NFRs' satisfaction as a boolean style measure. Here, the value of zero represents the NFR to be fully violated and the value of one represents the NFR to be fully satisfied. On the other hand, DeSiRE considers zero as the boundary point between the satisfaction and violation of NFR. The positive and negative values represent the extent of satisfaction and violation, respectively. DeSiRE calculates the extent of satisfaction for an NFR using the following equation:

$$ExS(NFR_n) = \Delta p/\sigma p \tag{7.1}$$

Here $ExS(NFR_n)$ represents the extent of satisfaction of nth NFR. Δp represents the difference between the referenced value p_r(threshold value for satisfaction) and the measured value p_m of the monitored property of NFR. The Δp is calculated using equations of DeSiRE operators [44].
For minimization of an NFR, Δp is calculated as:

$$\Delta p = p_r - p_m \tag{7.2}$$

Whereas, for maximization of an NFR, Δp is calculated as:

$$\Delta p = p_m - p_r \tag{7.3}$$

σp is the standard deviation of the measured values of the monitored property of NFR. The extent of an NFR being satisfied or violated is proportional to the magnitude of the measure of $ExS(NFR_n)$. For example, in an IoT network, if the measured value for energy consumption at a particular time step is 30 coulombs and the threshold for satisfaction, considered as the reference value, is 20 coulombs. Then, Δp for Minimization of Energy

Consumption (MinEC) will become $\Delta p = referencevalue - measuredvalue = 20 - 30 = -10$. If the σp for the measured values is 4.67, then $ExS(MinEC)$ will be equal to -2.14. As the *ExS(MinEC)* is less than the satisfaction boundary of zero, it means the NFR MinEC is violated. The *ExS* value specifies the extent (i.e. degree) of its violation. Hence, the DeSiRE approach presents, in a normalized way, the severity of an NFR's satisfaction or violation which is independent of the NFR's metrics (e.g. numbers of packets lost) presented in Chapter 6. Moreover, for an NFR to be considered as satisfied, the requirement is to have an *ExS* value greater than or equal to zero.

7.1.2 Experimental Evaluations

The main objective of the evaluations presented in this section is to perform a validation analysis of the decision-making results offered by Pri-AwaRE. The goal is to assess the extent of satisfaction of NFRs for both case studies by addressing the following question:

Q: To what extent does Pri-AwaRE satisfy the required extent of satisfaction for NFRs (i.e. ExS>= 0)?

Next, the experimental evaluations using the DeSiRE approach for both the case studies are presented.

IoT case: Experimental Evaluations

Experimental results for the Extent of Satisfaction (*ExS*) of the NFRs computed using DeSiRE [44] are shown in Fig 7.1. Based on the specifications of the DeSiRE, the value of zero represents the satisfaction boundary between the positive and negative degree of satisfaction (i.e. *ExS* values). As presented in Fig 7.1, the Pri-AwaRE approach shows positive *ExS* values for MinEC at almost all of the simulation time steps. However, the *ExS* value for MinEC does go below zero at several steps, yet this drop in the value never dips far below the satisfaction boundary.

On the other hand, Pri-AwaRE also demonstrates promising results with respect to the satisfaction of MinPL. The *ExS* value for MinPL is above zero most of the time steps, representing a positive degree of satisfaction for MinPL. When the *ExS* value does dip below zero, the maximum deviation in the *ExS* value ranges between -1.5 to -2.0. which is not

that far from the satisfaction boundary. The reason behind this behaviour is to create a balance between the NFRs in achieving the required extent of satisfaction of NFRs. The Pri-AwaRE approach achieves this balance by making a trade-off between NFRs based on their priorities. Hence, in order to achieve higher energy consumption levels, MinPL is slightly violated. However, this violation is not of a higher magnitude, and is re-balanced at the time step following the dip, as presented in Fig. 7.1.

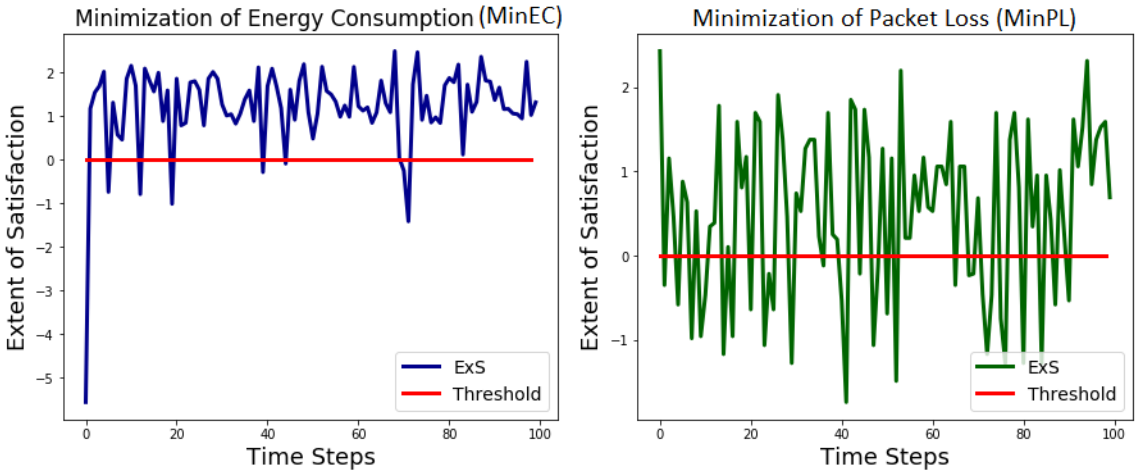


Figure 7.1: IoT Case: Extent of Satisfaction (*ExS*) of NFRs over Time

Summary of Findings

To sum up, from the results it is evident that Pri-AwaRE shows high *ExS* values for both the NFRs indicating high degrees of satisfaction. For the 100 simulation time steps, the *ExS* value of MinPL was 68 percent of the simulation time steps greater than zero, whereas for MinEC, *ExS(MinEC)* was positive 92 percent of the times. Pri-AwaRE shows promising results in terms of minimizing violations of MinEC while also providing a fair level of satisfaction of MinPL.

RDM Case: Experimental Evaluations

For the RDM case, experimental evaluations using DeSiRE have been performed to evaluate Pri-AwaRE's results for all the scenarios (described in Chapter 5) of RDMSim. Let us discuss the experiment results for scenarios S_0 and S_1. Scenario S_0 presents the default scenario where the RDM system is considered to be working under a stable environmental situation. Under S_0, all the three NFRs MinC, MaxR and MaxP, are required to comply

with the requirements most of the time. On the other hand, Scenario S_1 presents the environmental situation where a phase of consecutive and unexpected packet loss is observed during the execution of MST topology. This packet loss is due to equipment failures and affects the system's reliability.

Experimental results presenting the *ExS* values of the NFRs for Scenario S_0 are shown in Fig 7.2. Under Scenario S_0, the Pri-AwaRE approach shows positive *ExS* values for MinC and MaxP at almost all of the simulation time steps. Although the *ExS* value for these NFRs does drop below zero at a few time steps, these drops in the value are tolerable as they are re-balanced following the dip. With respect to the satisfaction of MaxR, the *ExS* value is above zero at most of the time steps, but the results also show a drop in the *ExS* value for MaxR at several time steps. However, the *ExS(MaxR)* value goes below zero, this deviation in the *ExS* value ranges between -1.0 to -1.5 which is not that far from the satisfaction boundary. Moreover, this violation of MaxR is acceptable as the more prioritized NFRs under S_0 are MinC and MaxP. This is due to the fact that MST is considered as the preferred topology under such a situation, as described by the experts of RDM [56].

Under Scenario S_1, the decisions carried out by Pri-AwaRE show similar results concerning the *ExS* values, which are mostly positive for all the three NFRs, as shown in Fig 7.3. Nevertheless, there is also a decrease in the *ExS* value for the NFRs at several time steps. This reduction in the *ExS* is quite close to the satisfaction boundary. For MinC, the drop in the *ExS* value ranges from -0.3 to -1.8. For the MaxR and MaxP, it ranges from -0.4 to -1.3 and -0.3 to -1.8, respectively. The experimental evaluations for all the other scenarios of RDMSim also exhibit comparable results in terms of *ExS* values, and therefore show maintenance of the required extent of satisfaction, i.e. $ExS >= 0$ in more than 50 percent of the simulation time steps. Experimental evaluations for the scenarios S_2 to S_6 are provided in Appendix C.

Summary of Findings

In summary, the Pri-AwaRE approach shows high *ExS* values indicating positive degrees of satisfaction for the NFRs MinC, MaxR and MaxP. The *ExS* value for all the NFRs is greater than or close to zero at almost all of the simulation time steps. Under Scenario S_0, *ExS* values for MinC, MaxR and MaxP are positive 70.2, 54.6 and 88.8 percent of the times, respectively. Under S_1, the results show a positive *ExS* value for MinC 63 percent of times

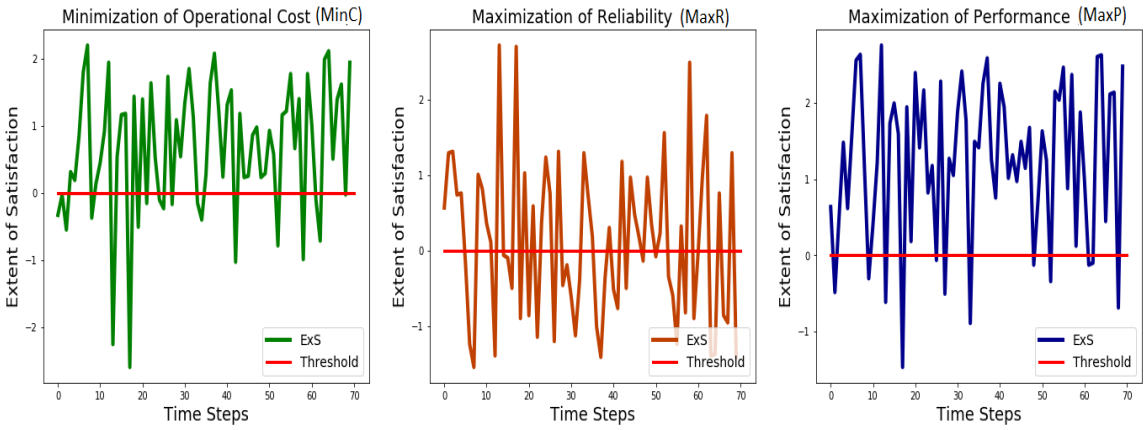


Figure 7.2: RDM Case: Extent of Satisfaction (*ExS*) of NFRs over Time under Scenario S_0

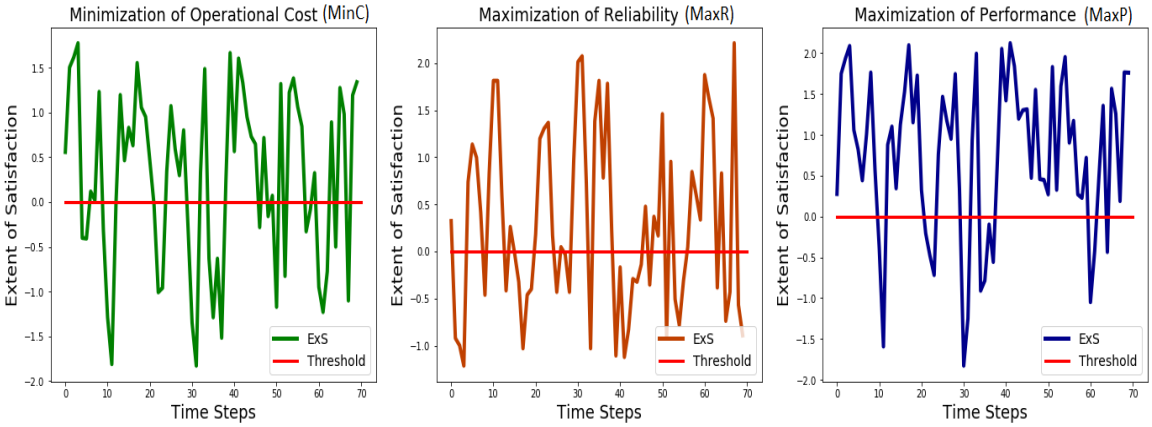


Figure 7.3: RDM Case: Extent of Satisfaction (*ExS*) of NFRs over Time under Scenario S_1

and for MaxR and MaxP 58.2 and 74.6 percent of the times, respectively. This satisfaction rate is based on the priorities of NFRs under both the scenarios and is considered as acceptable under these environmental contexts. Moreover, Pri-AwaRE exhibits similar results under all the scenarios by showing $ExS >= 0$ in more than 50 percent of the simulation time steps for all three NFRs.

Therefore, based on the evaluations performed using DeSiRE for both the case studies, we can say that:

The decision-making offered by Pri-AwaRE approach maintains the required extent of satisfaction for NFRs (i.e. ExS>= 0) in more than 50 percent of the simulation time steps.

7.2 Evaluation of Fidelity of Belief Satisfaction Probabilities

Let us recall that the Pri-AwaRE approach, based on MR-POMDP++, maintains a belief (i.e. probability) over the satisfaction state of the NFRs due to partial observability. A question that the reader can ask is: "*How accurate these beliefs are when reflecting the satisfaction state of the NFRs?*"

Therefore, to further assesses the quality of the decision-making offered by the Pri-AwaRE, this section presents evaluation of the fidelity of these belief satisfaction probabilities. Here, the fidelity of the beliefs refers to the extent to which they are a faithful representation of the real satisfaction values of the NFRs for the SASs. This relationship between real observations and the belief satisfaction probabilities of the NFRs (maintained by Pri-AwaRE) has been evaluated using Logistic Regression [99]. The Logistic Regression approach has been applied to evaluate the decision-making results for both case studies (presented in Chapter 6). It considers both the belief satisfaction probabilities for the NFRs and the network parameters' values coming from the simulated environments for the DELTA-IoT [72] and the RDMSim [137] networks. By using Logistic Regression, we want to check when both the features are considered: to what extent they represent a specific satisfaction state of an NFR; and how good the beliefs are in reflecting the satisfaction state of the NFRs at runtime.

Furthermore, due to the uncertain (random) nature of the belief values, the real environmental observations do not strongly correlate with beliefs for the hidden satisfaction state. This is evident from the dataset[1] provided at [133]. Due to this fact and the binary nature of the satisfaction state (True or False) of the NFRs, Binary Logistic Regression [99] is selected for evaluation purposes. It helps study the fidelity of beliefs to reflect the satisfaction states of the NFRs using simple binary classification. Moreover, the Softmax function [21] is applied during Logistic Regression to determine the extent to which a particular belief satisfaction probability belongs to a particular satisfaction class (True or False).

Next, hypotheses and experimental evaluations for Pri-AwaRE approach are described.

[1]The dataset is based on the observations and belief satisfaction probabilities maintained by Pri-AwaRE at runtime during different simulation time steps.

Table 7.1: IoT Case: DataSet Format

Total Energy Consumption	Satisfaction Probability	Actual Satisfaction State
27.713032	0.882122473	False
27.166639	0.838070816	False
26.961975	0.831177457	False
14.415679	0.882122473	True
15.693936	0.838070816	True

Hypotheses

The Hypotheses for the experimental evaluations are as follows:

$\boldsymbol{H}_0$*: During the decision-making process for SASs, the satisfaction states for the NFRs are not accurately reflected by the belief satisfaction probabilities maintained by Pri-AwaRE based on the real observations.*

$\boldsymbol{H}_a$*: During the decision-making process for SASs, the satisfaction states for the NFRs are accurately reflected by the belief satisfaction probabilities maintained by Pri-AwaRE based on the real observations.*

Next, the experimental evaluations performed for the validation of the results of decision-making carried out by Pri-AwaRE for both the case studies are presented.

7.2.1 IoT case: Setup for Evaluation

In the IoT case, the experiments have been executed to evaluate the Pri-AwaRE based decision-making for the uncertainty scenario presented in Chapter 5. The scenario considers the uncertainty context of network link interference measured using the signal-to-noise-ratio (SNR) and its effects on the NFRs of MinEC and MinPL.

DataSet:

The dataset format for the experiments is presented in Table 7.1. The input and output variables are presented as follows:

Input Features:

1) Value of the Network Parameter such as total energy consumption or total packets lost after each decision at a particular simulation time step.
2) Probability of Satisfaction of an NFR.

Output Variable:

Actual Satisfaction State having a value of True when an NFR is satisfied and False otherwise.

For the experiments, the training data set consists of decision-making results for Pri-AwaRE from 4 simulation runs of DELTA-IoT. Each run was executed for 50 simulation time steps. One simulation time step in case of the DELTA-IoT network is equal to 15 minutes of network activity [72]. During each time step, local decisions were taken for each of the nodes (sensors) individually. As a result, the training set is composed of 2800 training examples. For the purpose of tuning of the parameters of the Logistic Regression model, the Stochastic Gradient Descent (SGD) Algorithm [21] is used. To tune the parameters, cross-validation has been applied by splitting the training set into 80/20 proportions. During cross-validation, the 80 percent of the training set data is used as training set and remaining 20 percent is used as cross-validation set. After tuning of the parameters using cross-validation, the results for the test set comprising of the newly generated decision-making results from 50 simulation time steps were collected. Hence, the test set comprises 700 test examples. The data set is available at [133].

7.2.2 IoT case: Experimental Evaluations

Experimental evaluations have been performed to evaluate the decision-making results for the Pri-AwARE approach. In the experiments, the fidelity of beliefs to reflect the satisfaction state of NFRs at runtime for Pri-AwaRE is assessed. The parameter tuning and the classification results using Logistic Regression are described as follows:

1) Parameter Tuning

For the purpose of tuning of the parameters for the Logistic Regression model, the SGD algorithm is executed for 10,000 iterations with different learning rates for the Pri-AwaRE results of each NFR separately. On the basis of the accuracy scores presented in Table 7.2, the learning rate of 0.009 is selected for the model to evaluate the test set.

Table 7.3: IoT Case: Example Classification Results for MinEC

Test Example	Energy Consumed	SatProb	Actual State	Predicted State	Probability per class
634	11.31384	0.83118	True	True	[6.68881718e-04 9.99331118e-01]
635	10.3967	0.88212	True	True	[1.70297893e-04 9.99829702e-01]
636	13.24697	0.83807	True	True	[0.00285094 0.99714906]
637	16.28405	0.83118	True	True	[0.0337675 0.9662325]
638	11.86322	0.88212	True	True	[5.46885746e-04 9.99453114e-01]

Table 7.4: IoT Case: Example Classification Results for MinPL

Test Example	Packets Lost	SatProb	Actual State	Predicted State	Probability per class
440	0.083333333	0.926129374	True	True	[2.21770858e-05 9.99977823e-01]
441	0.061538462	0.979701363	True	True	[1.68675727e-06 9.99998313e-01]
442	0.142857143	0.981213319	True	True	[0.00313043 0.99686957]
443	0.2	0.926129374	True	False	[0.52700155 0.47299845]
444	0.275	0.979701363	False	False	[0.99851764 0.00148236]

Table 7.2: IoT: Learning Rate Accuracy Score for MR-POMDP++

Learning Rate	Accuracy Score MinEC	Accuracy Score MinPL
0.0001	1.0	0.83542
0.0003	1.0	0.92308
0.0005	1.0	0.94097
0.0007	1.0	0.96422
0.0009	1.0	0.98032
0.001	1.0	0.98032
0.003	1.0	0.98032
0.005	1.0	0.98389
0.007	1.0	0.99821
0.009	**1.0**	**1.0**
0.01	1.0	1.0
0.03	1.0	1.0
0.05	1.0	1.0
0.07	1.0	1.0
0.09	1.0	1.0

2) Classification Results

For the purpose of description, 5 classification results examples for the NFRs are presented in Tables 7.3 and 7.4. The results for all the other test cases have been provided in the repository [133]. The results show an accuracy score of 1.0 for MinEC and 0.9943 for MinPL with a precision of 1.0 for both MinEC and MinPL. The F1 scores for the classification results of MinEC and MinPL are 1.0 and 0.9963, respectively. The classification results for the NFRs are discussed as follows:

a) Classification Results for MinEC

For the 700 test examples, collected as a result of 50 simulation time steps, the model correctly classifies the satisfaction state of MinEC as shown in Fig. 7.4.

Moreover, based on the results presented in Table 7.3, we can deduce the extent to which MinEC can be considered as satisfied, when both the energy consumption and satisfaction probability are considered. For example, for test example 636, the energy consumed is 13.24697 coulombs and the satisfaction probability is 0.83807. Similar to the actual state, the satisfaction state predicted by the model is True with 0.99714906 probability. Hence, given the input values, there is around 99 percent chance of MinEC being satisfied i.e. having satisfaction state as True.

b) Classification Results for MinPL

For the 700 test examples collected as a result of 50 simulation time steps, the model correctly classifies the satisfaction state of MinPL with an exception of 4 test examples, as shown in Fig. 7.4. For these test examples, the actual satisfaction state for MinPL was True but it was predicted as False.

Moreover, based on the results presented in Table 7.4, we can deduce the extent of the satisfaction state of MinPL, when both the percentage packets lost and satisfaction probability are considered. For example, for test example 441, the packet loss is 0.061538462 and the satisfaction probability is 0.979701363. The output satisfaction state predicted by the model is True, similar to the actual state, with 9.99998313e-01 probability of being True. It means that given the input values, there is around 99 percent chance of MinPL being satisfied i.e. having the satisfaction state as True.

Summary of Findings

In summary, the results for the IoT case show an overall accuracy score of around 99 percent in predicting the satisfaction states of the NFRs of MinEC and MinPL. The predictions are based on the beliefs and observations. Therefore, it satisfies the hypotheses for the experiments by proving the fidelity of beliefs to reflect the *hidden satisfaction state* of NFRs, which is estimated to be approximately 99 percent. Hence, it rejects the hypothesis $\boldsymbol{H}_0$ and satisfies $\boldsymbol{H}_a$.

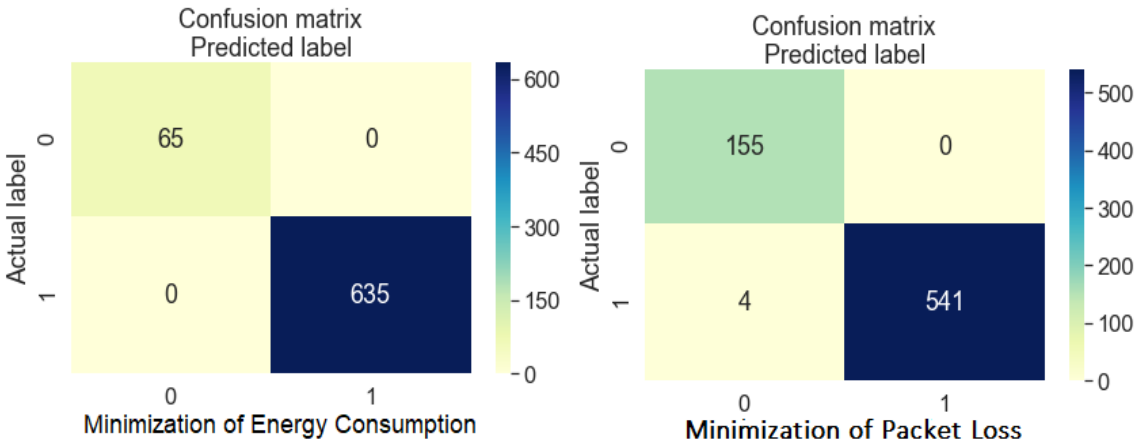


Figure 7.4: IoT: Confusion Matrix for Classification of Satisfaction State of NFRs

7.2.3 RDM Case: Setup for Evaluation

In the RDM case, the experiments have been executed to evaluate the Pri-AwaRE based decision-making for all the dynamic scenarios described in Chapter 5.

DataSet

The dataset for the experiments is of the format presented in Table 7.5. The input and output variables are presented as follows:

Input Features:

1) Value of the Network Parameter such as Bandwidth Consumption, Time to Write Data and Active Links generated by RDMSim.
2) Probability of Satisfaction of an NFR.

Output Variable:

Actual Satisfaction State having a value of True when an NFR is satisfied and False otherwise.

For the experiments, the training data set consists of results for Pri-AwaRE from 4 simulation runs of RDMSim. Each run was executed for 500 simulation time steps. As a result, the training set is composed of 2000 training examples. For the purpose of tuning of the parameters of the Logistic Regression model, the SGD [24] algorithm is used. To

Table 7.5: RDM Case: DataSet Format

Bandwidth Consumption	Satisfaction Probability	Actual Satisfaction State
2376	0.88656982	True
4050	0.904919218	False
2465	0.878517531	True
2320	0.902641707	True
1080	0.906088102	True

Table 7.6: RDM: Learning Rate Accuracy Score for MR-POMDP++

Learning Rate	Accuracy Score MinC	Accuracy Score MaxR	Accuracy Score MaxP
0.0001	0.96241	0.98496	0.93985
0.0003	0.97995	0.99749	0.96742
0.0005	0.98496	0.99749	0.97494
0.0007	0.98496	0.99749	0.98496
0.0009	0.98747	0.99749	0.98747
0.001	0.98747	0.99749	0.98747
0.003	0.99248	0.99749	0.99499
0.005	0.99749	0.99749	0.99499
0.007	0.99749	0.99749	0.99749
0.009	0.99749	0.99749	0.99749
0.01	0.99749	1.0	0.99749
0.03	1.0	1.0	1.0
0.05	0.99749	1.0	1.0
0.07	0.99749	1.0	1.0
0.09	0.99749	1.0	1.0

tune the parameters, cross-validation is applied by splitting the training set into 80/20 proportions. During cross-validation, the 80 percent of the training set data is used as training set and remaining 20 percent is used as cross-validation set. After tuning of the parameters using cross-validation, results for the test set comprising the newly generated results from 500 simulation time steps have been executed. The data set for the experiments is available at [133].

7.2.4 RDM Case: Experimental Evaluations

The experiments have been executed to evaluate results for all the scenarios provided by RDMSim that represent different dynamic environments for an RDM network. For the purpose of discussion, the evaluation results for one of the dynamic scenario for the RDMSim network, known as Scenario S_1, have been presented in this section. The Scenario S_1 represents the dynamic situations of unexpected packet loss during the execution of an MST topology. Evaluation results for the rest of the scenarios are reported in [133]. The parameter tuning and the classification results using the Logistic Regression approach are described as follows:

1) Parameter Tuning

For the purpose of the tuning of the parameters, the SGD algorithm is executed for 20,000 iterations with different learning rates for the results of Pri-AwaRE. On the basis of the accuracy scores presented in Table 7.6, the learning rate of 0.03 is selected for the model to evaluate the test set.

2) Classification Results

For the purpose of description, 7 classification results' examples for the NFRs are presented in Tables 7.7, 7.8 and 7.9. The results for all the other test cases have been provided in the repository [133]. The results show an accuracy score of 0.986 for MinC, 0.998 for MaxR and 1.0 for MaxP with a precision of 1.0 for MinC and MaxP, and 0.9967 for MaxR. The F1 scores for MinC, MaxR and MaxP are 0.9889, 0.9983 and 1.0 respectively.

a) Classification Results for MinC

For the 500 simulation time steps, under Scenario S_1 of RDMSim, the model correctly classifies the satisfaction state of MinC with the exception of 7 simulation time steps, as shown in Fig. 7.5. For these 7 simulation time steps, the actual satisfaction state for MinC was True but it was predicted as False.

Moreover, based on the results presented in Table 7.7, we can deduce the extent to which MinC can be considered as satisfied, when both the bandwidth consumption and satisfaction probability are considered. For example, at time step 482, the bandwidth consumption is 5550 GBps and the satisfaction probability is 0.725491276. The satisfaction state predicted by the model is also False with the 1.00000000e+00 probability of being False. It means that given the input values, there is 100 percent chance of MinC not being satisfied i.e. having satisfaction state of False.

Table 7.7: RDM Case: Example Classification Results for MinC for time steps 482-488 under Scenario S_1

Step	Bandwidth Consumed	SatProb	Actual State	Predicted State	Probability per class
482	5550	0.725491276	False	False	[1.00000000e+00 4.28385292e-10]
483	4833	0.783454523	False	False	[9.99998958e-01 1.04154805e-06]
484	4725	0.791813864	False	False	[9.99996635e-01 3.36520499e-06]
485	2625	0.864214984	True	True	[5.38856619e-05 9.99946114e-01]
486	1944	0.902410103	True	True	[3.51550671e-08 9.99999965e-01]
487	5175	0.867215377	False	False	[9.9999996e-01 4.0101029e-08]
488	3525	0.873227415	True	True	[0.40579247 0.59420753]

Table 7.8: RDM Case: Example Classification Results for MaxR for time steps 482-488 under Scenario S_1

Step	Active Links	SatProb	Actual State	Predicted State	Probability per class
482	222	0.958693166	True	True	[2.3037888e-17 1.0000000e+00]
483	179	0.964496786	True	True	[2.57926305e-11 1.00000000e+00]
484	225	0.952357296	True	True	[8.70731399e-18 1.00000000e+00]
485	125	0.83784989	True	True	[9.89801317e-04 9.99010199e-01]
486	81	0.766044144	False	False	[9.99337825e-01 6.62174864e-04]
487	225	0.902908016	True	True	[8.61891872e-18 1.00000000e+00]
488	141	0.83331521	True	True	[5.55843368e-06 9.99994442e-01]

Table 7.9: RDM Case: Example Classification Results for MaxP for time steps 482-488 under Scenario S_1

Step	Writing Time	SatProb	Actual State	Predicted State	Probability per class
482	2220	0.716539246	True	True	[0.00121563 0.99878437]
483	2148	0.699744551	True	True	[4.48927419e-04 9.99551073e-01]
484	2475	0.699741136	True	True	[0.04798941 0.95201059]
485	1375	0.80810347	True	True	[4.86431874e-09 9.99999995e-01]
486	1215	0.910140268	True	True	[3.73099626e-10 1.00000000e+00]
487	2925	0.849803269	False	False	[0.95807766 0.04192234]
488	2820	0.823122431	False	False	[0.84301066 0.15698934]

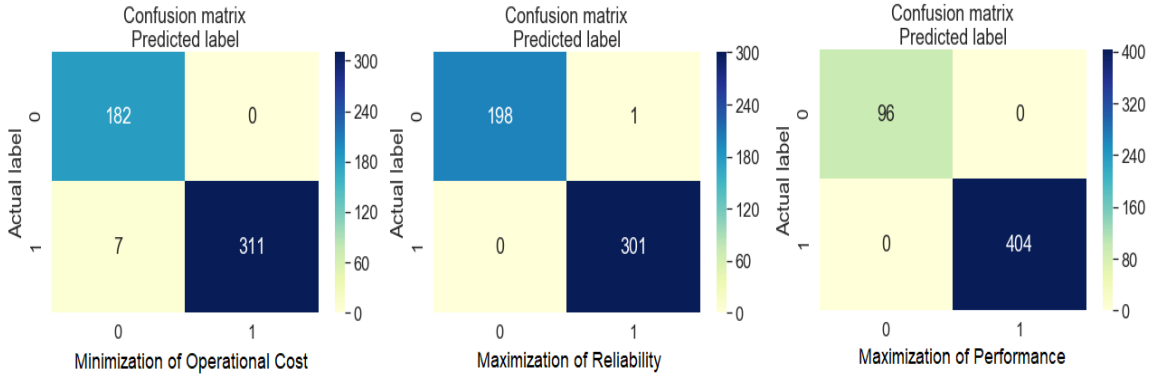


Figure 7.5: RDM: Confusion Matrix for Classification of Satisfaction State of NFRs under Scenario S_1

b) Classification Results for MaxR

For all of the 500 simulation time steps comprising the test set, the model correctly classifies the satisfaction state of MaxR with the exception of only 1 simulation time step, as shown in Fig. 7.5. For this 1 simulation time step, the actual satisfaction state for MaxR was False but it was predicted as True.

Moreover, based on the results presented in Table 7.8, we can deduce the extent of the satisfaction state of MaxR, when both the active links and satisfaction probability are considered. For example, at time step 482, the active links are 222 and the satisfaction probability is 0.958693166. The output satisfaction state predicted by the model is True similar to the actual state with the 1.000e+00 probability of being True. It means that given the input values, there is 100 percent chance of MaxR being satisfied i.e. having satisfaction

state of True.

c) Classification Results for MaxP

For all of the 500 simulation time steps in the test set, the model correctly classifies the satisfaction state of MaxP as shown in Fig. 7.5.

Moreover, based on the results presented in Table 7.9, we can deduce the extent of the satisfaction state of MaxP, when both the writing time and satisfaction probability are considered. For example, at time step 482, the writing time is 2220 ms and the satisfaction probability is 0.716539246. The output satisfaction state predicted by the model is True similar to the actual state with the 0.99878437 probability of being True. It means that given the input values, there is around 99 percent chance of MaxP to be satisfied i.e. having satisfaction state of True.

Summary of Findings

In summary, the results show an accuracy score of around 99 percent for prediction of the satisfaction states of all the NFRs of MinC, MaxR and MaxP based on the beliefs and given observations. Hence, it proves that the chances of beliefs in Pri-AwaRE to reflect the *hidden* satisfaction state of NFRs is estimated to be approximately 99 percent. Moreover, the evaluation of fidelity of beliefs in Pri-AwaRE for two different case studies shows that beliefs can be considered as a strong representation of the state of the NFRs during the decision-making for SASs. Hence, it rejects the hypothesis $\boldsymbol{H}_0$ and satisfies the hypothesis $\boldsymbol{H}_a$ for the experiments.

Therefore, based on the evaluations performed using Logistic Regression for both the case studies, we can say that:

During the decision-making process for SASs, the satisfaction states for the NFRs are accurately reflected by the belief satisfaction probabilities maintained by Pri-AwaRE based on the real observations.

Table 7.10: Comparison with Related Work

Approaches	Technique Used	Scalability w.r.t. NFRs	Scalability w.r.t. Time	Runtime	Individual Priority Modelling	Autonomous Tuning	Quantify Uncertainty
Multi-criteria Decision Making	Analytic Hierarchy Process	✓	✓-	X	✓	X	X
ARROW	POMDP + P-CNP	✓-	✓	✓	✓	X	✓
AutoRELAX	Genetic Algorithms	✓	-	X	✓	X	X
Providentia	Genetic Algorithms	✓	-	X	✓	X	X
Probabilistic Model	Dynamic Decision Networks	✓-	X	✓	X	X	✓
Markov based Approaches	MDPs + DTMC	✓-	✓	✓	X	X	✓
FLAGS	Fuzzy Logic	✓-	-	✓	-	X	X
ZANSHIN	Control Theory	✓-	-	X	-	X	X
Control Theoretic Self-Tuning	Control Theory	✓-	-	✓	✓	X	X
Pri-AwaRE	MR-POMDP	✓-	✓	✓	✓	✓	✓

X: the approach does not fulfill the criteria. ✓-: the approach is in process to fulfill the criteria.
-: the approach does not provide information. ✓ : the approach fulfills the criteria.

7.3 Comparison with Related Work

This section compares the Pri-AwaRE approach with the related approaches (as described in Chapter 2). These approaches deal with the modelling and reasoning of NFRs' priorities during the decision-making process for SASs. Comparative analysis, presented in Table 7.10, is performed by taking into consideration the aims of the research using the following criteria:

1) Scalability: The Mutli-Criteria Decision-Making (MCDM) [94] and search-based techniques [25, 122] exhibit scalability in terms of supporting the ranking of multiple NFRs. In contrast, the technique based on DDNs deal with the problem of scalability over time (i.e. the curse of history). The graph representing the history of observations and actions for DDN planning grows exponentially with the planning horizon. In comparison, Markov-based techniques such as [3, 28, 51, 103, 112, 167] resolve this issue of the curse of history. The techniques of ARROW [113], FLAGS [11], ZANSHIN [146] and Control Theory-based approaches [98, 114] have presented initial implementations mostly related to

a single application domain, but they still require further exploration in terms of scalability.

2) Modelling of Individual NFRs' priorities: The approaches based on MCDM, such as Analytic Hierarchy Process [94] and Primitive Cognitive Network Process (P-CNP) [174], support the explicit modelling of the NFRs' priorities. The technique of ARROW [113], which is based on P-CNP, also allows the explicit specification of the NFRs' priorities. Moreover, similar to MCDM based techniques, search-based techniques such as AutoRELAX [122] and Providentia [25] also provide modelling and reasoning with the distinct priorities of NFRs. However, these techniques optimize priorities at design time and these initially optimized priorities are further used at runtime in an off-line fashion. In contrast, the techniques based on probabilistic models such as DDNs [15] and Markov-based approaches [3, 28, 51, 103, 112, 167] model the NFRs' priorities as a single scalar utility value representing a combined priority for all NFRs. These approaches lack the explicit modelling of the individual priorities of NFRs, and therefore do not support priority-awareness. Moreover, the approaches based on Control Theory [98, 114] provide support for explicitly modelling the NFRs' priorities at runtime. Finally, the techniques of ZANSHIN and FLAGS do not provide information about the distinct modelling of the NFRs' priorities.

3) Design time vs Run time: As discussed, the MCDM techniques presented in [94] and [174] provide support for explicit modelling of the NFRs and their priorities. However, they are more design time techniques, i.e. they make use of the priorities assigned by the requirements engineers at design time. These design time priorities are then used at runtime during the decision-making process. The technique of ARROW, which is based on P-CNP, provides support to RE-STORM (using the POMDP model [57]) to deal with the prioritization of NFRs. The techniques of AutoRELAX, Providentia and ZANSHIN also support the optimization of priorities at design time. In the case of AutoRELAX and Providentia, these initially optimized priorities are further used at runtime in an off-line fashion. All of the other techniques support the runtime optimization of NFRs' priorities.

4) Autonomous Tuning of Priorities: The techniques presented in Table 7.10 do not support the autonomous tuning of the NFRs' priorities under the changing environmental contexts. Although these techniques provide the optimization or tuning of the priorities,

this update of the priorities is not done autonomously. It is mostly done at design time or in offline mode. The only techniques that support the runtime tuning of priorities are the techniques based on Control Theory [114] and ARROW [113]. However, ARROW supports automatic updates for initially defined priorities of NFRs at runtime. This priority update is not autonomous and does not work from within the POMDP model. The Control Theory approach [114] cannot deal with the NFRs having the same priority rank.

5) Quantification of Uncertainty: The DDNs, with the help of the Bayesian Surprise [18], support the quantification of uncertainty. Moreover, Markov-based techniques based on MDPs and POMDPs [3, 28, 51, 103, 112, 167], similar to Pri-AwaRE, also consider uncertainty as a quantifiable measure. In contrast, the other techniques do not support the quantification of uncertainty.

Compared to the above techniques, the Pri-AwaRE approach offers runtime modelling and reasoning with the individual priorities of NFRs by using the MR-POMDP++ model. The MR-POMDP++ supports explicitly modelling NFRs' priorities using a vector-valued reward function. Moreover, the technique also supports the autonomous tuning of the priorities of NFRs during the decision-making process. As Pri-AwaRE is based on MR-POMDP++, it considers the decision-making agent working in a partially observable environment. Hence, it supports the quantification of uncertainty. The approach uses the OLSAR MR-POMDP solver [129] that resolves the curse of history problem, and therefore offers scalability. However, the approach's scalability in dealing with a large number of NFRs still requires more exploration.

7.4 Threats to Validity

This section presents the threats to the validity of the approach to assess the factors to dispute the research findings presented in this thesis. The threats to validity, based on the classification described in [50], are presented as follows:

External Validity: External validity indicates the generalization of the outcomes which are outside the scope of the study. A key threat to validity of the proposed Pri-AwaRE approach lies in the computational cost. The Pri-AwaRE is based on MR-POMDP++.

Solving an MR-POMDP is a computationally intractable problem in its worst case, even with the help of the OLSAR algorithm [129] that can deal with the issues related to scalability (such as the "curse of history" problem). In the case of Pri-AwaRE, the states are represented as the combinations of satisfaction levels of NFRs. If the NFRs are 2 in number, 4 states will be defined for MR-POMDP++, for 3 NFRs, it would be 8 and so on. Hence, practically, the Pri-AwaRE approach will work with only a small number of NFRs. This means that experts using the Pri-AwaRE approach, must limit their reasoning to the NFRS that are critical to driving the self-adaptations. Therefore, the approach belongs to the group of techniques that focus theoretically and practically on a few objectives to support multi-objective sequential decision-making [128].

Furthermore, the experiments presented in this thesis have been executed using the exemplars of DELTA-IoT [72] and RDMSim [137] which focus on a centralized setting. The Pri-AwaRE approach has not been tested in a decentralized setup. More experiments would be required to test the feasibility of Pri-AwaRE for applications in both centralized and decentralized domains.

Internal Validity: Internal validity refers to ensuring that the treatment for the setup of experiments is similar to the real setting. It corresponds to the extent to which the Pri-AwaRE approach performs when working in an actual environmental setup. The experiments presented in this thesis focus on a case study approach based on a simulator. All the experimental results are based on environmental factors, simulated by tools of DELTA-IoT and RDMSim, not an actual physical network. The case studies that have been selected are well-known and provide simulations that are close to real settings. Both RDM [53, 137] and IoT [72] are well-accepted example cases in the research community and are already in use by other teams.

Construct Validity: Construct validity focuses on the relationship between the theory behind the experiments and the actual observations. Construct validity for the Pri-AwaRE approach refers to the mirroring relationship between Pri-AwaRE and the managed system. In this thesis, reflections have been established for the current state of the managed system by MR-POMDP++ and are verified by estimation of the fidelity of the belief over the states (as described in Section 7.2).

7.5 Summary

In summary, based on the validation of the decision-making results offered by Pri-AwaRE, it can be deduced that Pri-AwaRE supports the maintenance of the Extent of Satisfaction of NFRs. It also shows statistically sound results in terms of reflecting the real satisfaction state of NFRs using the belief probabilities maintained by MR-POMDP++. Furthermore, compared to the existing state-of-the-art approaches, Pri-AwaRE offers better-informed, priority-aware decisions by providing runtime modelling and autonomous tuning of the NFRs' priorities during the decision-making process. Therefore, Pri-AwaRE addresses the research challenges described in Chapter 1, and thereby answers the **RQs**.

Chapter 8

Conclusion and Future Work

This chapter sums up the work presented in this thesis. The main motivation for developing the proposed Pri-AwaRE architecture has been the challenges concerning the need for new techniques for decision-making under uncertainty. The challenges specify that the approach should: i) support modelling of the individual priorities of NFRs, and ii) perform autonomous tuning of the priorities according to the changing runtime contexts.

The chapter is organized as follows: Section 8.1 revisits the main contributions of the thesis which is followed by the future research directions in Section 8.2.

8.1 Contributions

This section presents the main contributions of the research presented in this thesis in terms of answering the research questions presented in Chapter 1.

RQ1: *Can modelling and reasoning of the priorities of individual NFRs under uncertain environmental contexts be supported?*

To address **RQ1**, this thesis presents Pri-AwaRE, a self-adaptive architecture for decision-making in SASs. The architecture uses MR-POMDP++, a runtime specification model, as part of the MAPE-K loop to select decisions at runtime. The technique tries to overcome the limitations of the existing approaches by supporting the modelling and reasoning with individual NFRs' priorities during the decision-making process. MR-POMDP++ is an extended version of MR-POMDP, and supports the modelling of individual NFR priorities

using a vector-valued reward function. Each value in the reward vector is associated with a separate NFR. The reward values are generated as a feedback signal due to the decisions (adaptation actions) taken by MR-POMDP++. The reward value for a specific NFR refers to the effect (positive or negative) of performing an adaptation action on the satisfaction of that NFR. Therefore, the reward vector values represent a relative ranking of the NFRs in terms of the cardinal effect an action will have on the NFRs' satisfaction under an environmental situation. Consequently, an NFR with a higher reward value indicates that the NFR has a higher priority for satisfaction and vice versa. These priorities, in the form of rewards, are considered by MR-POMDP++ while selecting adaptation actions during the decision-making process.

The mapping rules for rewards to represent the initial NFR priorities are presented in Chapter 4. Furthermore, this modelling of priorities by MR-POMDP++ serves as a base to answer **RQ2**.

RQ2: *Can decision-making in SASs include tuning of the NFRs' priorities to match the dynamic runtime situations?*

MR-POMDP++ using the reward vector enables the Pri-AwaRE to *autonomously* tune the individual priorities of NFRs according to the changing runtime contexts. Therefore, it addresses **RQ2**. This autonomous tuning of priorities is performed by the computation of an individual *expected utility value* for each NFR during the decision-making performed by MR-POMDP++ using equation 4.1 (presented in Chapter 4). The *expected utility values* represent the newly *tuned priority values* of the individual NFRs. Moreover, the rewards, representing the initial expert assigned priorities, are used to calculate the individual *expected utility value* for each NFR at runtime. Hence, the *expected utility values* take into account the impact of executing an adaptation action on the satisfaction of an NFR separately under an environmental context. These *expected utility values* are considered while taking decisions for adaptations at runtime. Therefore, the Pri-AwaRE approach using MR-POMDP++ supports *priority-awareness* during the decision-making process.

As a proof of concept, the approach is applied to the two well-established case studies of IoT and RDM networks. The experiments show that Pri-AwaRE provides the *autonoumous tuning* of NFRs' priorities according to the dynamic contexts for both the cases. Moreover,

the results also show that the decisions take into account the individual priorities of the NFRs which are adjusted according to the runtime situations to meet compliance with the requirements.

DELTA-IoT: For the IoT case, an existing simulation tool of DELTA-IoT [72] has been selected. The DELTA-IoT provides a simulated environment for an IoT network for a smart university campus. The simulator represents a multi-hop IoT network comprising 15 nodes which are installed across the different buildings of the campus. The nodes communicate with each other and relay information to the central gateway.

RDMSim: For the RDM case study, a simulation tool of RDMSim [137] has been developed as part of the research work presented in this thesis. The simulator has been designed on the basis of the operational model presented in [74, 78]. The simulator also provides a number of uncertainty scenarios to represent the different dynamic situations for the RDM network. The RDMSim simulator is publicly available and can be downloaded from [132]. Moreover, the simulator provides the probe and effector components as APIs for connecting to the RDMSim network. The researchers working with different decision-making approaches, such as evolutionary computation [120] or multi-criteria decision-making techniques [158], can use these APIs for working with RDMSim. The simulator also provides researchers with the flexibility of designing their own uncertainty scenarios.

Comparisons with the existing approaches of RE-STORM [112] and RE-STORM-ARROW [113] have also been provided. The results show that the Pri-AwaRE offers comparable and sometimes even better results under certain dynamic situations compared to both approaches.

To sum up, the contributions of the research presented in this thesis are as follows:

Contribution 1: Pri-AwaRE, a priority-aware self-adaptive architecture for decision-making in SASs that offers:

i) modelling and reasoning with the individual priorities of NFRs which is answer to **RQ1**, and

ii) autonomous tuning of the NFRs' priorities during the decision-making process which is answer to **RQ2**.

Contribution 2: Application of the Pri-AwaRE approach to the two case studies of IoT and RDM networks and design of experiments for the case studies.

Contribution 3: Design and development of the RDMSim, a simulator for the RDM case study. The simulator is publicly available at [132] and can be used by researchers working on different decision-making techniques for SASs to test their techniques.

Contribution 4: Design of the decision-making mechanism for both the case studies. For the IoT case, a local decision-making mechanism has been designed to carry out local decisions for each individual node of the network. For the case study of RDM, the decision-making mechanism has been designed to perform global decisions of changing the entire topology for the network.

Contribution 5: The implementation of Pri-AwaRE approach along with data set of experiments' results logs for both case studies. The implementation can be downloaded to carry out experiments using Pri-AwaRE. The data set is provided to facilitate the researchers working on different decision-making approaches for comparisons. The Pri-AwaRE implementation and data set are available at [133, 134].

8.2 Future Work

The contributions presented in this thesis can be developed further. As a future research agenda, new research areas have been identified that would allow researchers to produce more useful knowledge. The future research directions are as follows:

8.2.1 Elicitation of Expert-Defined NFR Priorities

The initial NFR priorities are typically defined by the domain experts at design time. However, there might be situations which are unforeseen by the experts and they might challenge the assigned priorities. Hence, one of the future directions can be usage of the Pri-AwaRE approach, with its capability of autonomous tuning of priorities, as a tool for *a priori*-elicitation of priorities for NFRs. This will facilitate the experts in defining of the initial priorities. The idea would be to execute simulations in order to learn about the environment, and to therefore uncover the contexts that would not be anticipated otherwise. Based on this newly discovered knowledge ***K'***, a new specification ***S'*** could

be used in the implementation of new releases of the SAS. Furthermore, as the approach supports in enriching the decision-making process in SASs with newly discovered knowledge ***K'***, the Pri-AwaRE could also be used to provide *explanations* for unclear adaptation decisions [111, 139].

The elicitation process can be supported by automation tools:

Firstly, as the priorities are represented as rewards in the MR-POMDP++, techniques like Inverse Reinforcement Learning (IRL) [35, 178] could infer these reward values. Therefore, such values can support the elicitation process by guiding the experts. IRL is an AI based technique that supports imitation of the preferred system behavior by using its behavioral history. It helps in the inference of the reward values by taking the observed history of policies as an input. The usage of IRL could be considered a prospective research direction for inference of the initial NFRs' priorities.

In addition to IRL, to support elicitation of NFR priorities, techniques based on multi-objective evolutionary computation [159, 177] along with reinforcement learning [41, 81] can also be explored as one of prospective research directions.

8.2.2 Further Specification of Uncertainty Quantification to Improve Priority Awareness of NFRs

MR-POMDP++ considers the decision-making agent working in a partially observable environment. It therefore supports the quantitative specification of uncertainty by maintaining a belief over the state. However, more research is needed to study the quantification of uncertainty. For this purpose, usage of uncertainty quantification techniques such as Bayesian Surprise [13, 17] could be explored. More work is needed to evaluate the levels of uncertainty using Bayesian Surprise and their impacts on the priorities of NFRs. Based on these impacts, Bayesian Surprise can provide the temporal RELAXation [169] of priority values of one or more NFRs to support the satisfaction of a critical NFR.

Moreover, the analysis of results with respect to the Extent of Satisfaction *ExS* (described in Chapter 7) also generates an opportunity for studying the RELAXation of NFRs [169]. For instance, if we consider the situation where a particular NFR has a high *ExS* value at a specific simulation time step. Another critical NFR shows a slight violation by having the *ExS* value below but closer to zero. In such a situation, the first NFR can be RELAXed [169] at that simulation time step to benefit the satisfaction of the other. The

exploration of such *autonomous* RELAXation of priorities facilitated by Bayesian Surprises could be one of the future research directions to follow. Moreover, the goal will be to perform uncertainty resolution using Bayesian Surprise and modelling it in terms of the RELAX [169].

8.2.3 Fidelity Marker for Digital Twins

The results reported in this thesis show the potential use of the Pri-AwaRE approach to reflect the satisfaction state of NFRs based on the belief values maintained at runtime. Crucially, the decision-making process based on beliefs can be used to support technologies such as Digital Twins (DTs) [10, 36, 100]. It will help in providing simulations of the real world to study the consequences of executing the adaptations. For this purpose, fidelity (i.e. accuracy) of the beliefs to reflect the real but partially observable state of the NFRs needs to be assessed. Further exploration in this domain could also be one of the future research directions to follow.

8.3 Final Remarks

This thesis has presented Pri-AwaRE, a self-adaptive architecture for decision-making in SASs. The approach offers the capability of modelling and reasoning about the individual priorities of NFRs. Moreover, it also supports autonomous tuning of individual NFRs' priorities based on the changing runtime environmental contexts. Hence, Pri-AwaRE provides *priority-awareness* during the runtime decision-making. The approach has been applied to the case studies belonging to two networking domains. The results have shown that the priority-aware decision-making offered by Pri-AwaRE offers compliance with the requirements during execution. Also, more information concerning the individual satisfaction priorities of NFRs is considered during decision-making. The PhD research project also led to the development of RDMSim, a simulator for the RDM network.

An insight into the usage of MR-POMDP as a runtime model to support modelling and reasoning with the separate NFRs' priorities is also provided. The belief satisfaction probabilities maintained by the MR-POMDP to represent real satisfaction state of NFRs help establishing a causal connection with the real system. Based on this, we can envision that Pri-AwaRE has the potential to support technologies such as Digital Twins to provide

realistic and accurate simulations of the world. In addition, we can also envisage the usage of Pri-AwaRE as a tool for providing further insight to the experts when eliciting NFRs' priorities. Moreover, the approach could also be foreseen as a tool to support the quantitative specification of uncertainty.

List of References

[1] Definition of efficacy - NCI Dictionary of Cancer Terms - NCI, Feb. 2011. URL `https://www.cancer.gov/publications/dictionaries/cancer-terms/def/efficacy`.

[2] Chapter Two - Database Management Systems. In O. Curé and G. Blin, editors, *RDF Database Systems*, pages 9–40. Morgan Kaufmann, Boston, Jan. 2015. ISBN 978-0-12-799957-9.

[3] M. Abundo, V. Cardellini, and F. L. Presti. An mdp-based admission control for a qos-aware service-oriented system. In *2011 IEEE Nineteenth IEEE International Workshop on Quality of Service*, pages 1–3. IEEE, 2011.

[4] Z. Alhassan, A. S. McGough, R. Alshammari, T. Daghstani, D. Budgen, and N. Al Moubayed. Type-2 diabetes mellitus diagnosis from time series clinical data using deep learning models. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, pages 468–478. Springer, 2018.

[5] Z. Alhassan, M. Watson, D. Budgen, R. Alshammari, A. Alessa, N. Al Moubayed, et al. Improving current glycated hemoglobin prediction in adults: Use of machine learning algorithms with electronic health records. *JMIR Medical Informatics*, 9(5): e25237, 2021.

[6] T. A. Althunian, A. de Boer, R. H. Groenwold, and O. H. Klungel. Defining the noninferiority margin and analysing noninferiority: an overview. *British journal of clinical pharmacology*, 83(8):1636–1642, 2017.

[7] L. Atzori, A. Iera, and G. Morabito. The Internet of Things: A survey. *Computer Networks*, 54(15):2787–2805, Oct. 2010. ISSN 13891286.

[8] U. Aßmann, N. Bencomo, B. H. C. Cheng, and R. B. France. Models@run.time (Dagstuhl Seminar 11481). page 33 pages, 2012.

[9] P. Baldi and L. Itti. Of Bits and Wows: A Bayesian Theory of Surprise with Applications to Attention. *Neural networks : the official journal of the International Neural Network Society*, 23(5):649–666, June 2010. ISSN 0893-6080.

[10] S. Barat, R. Parchure, S. Darak, V. Kulkarni, A. Paranjape, M. Gajrani, and A. Yadav. An agent-based digital twin for exploring localized non-pharmaceutical interventions to control covid-19 pandemic. *Transactions of the Indian National Academy of Engineering*, 6(2):323–353, 2021.

[11] L. Baresi, L. Pasquale, and P. Spoletini. Fuzzy Goals for Requirements-Driven Adaptation. In *2010 18th IEEE International Requirements Engineering Conference*, pages 125–134, Sept. 2010.

[12] K. L. Bellman, C. Landauer, P. Nelson, N. Bencomo, S. Götz, P. Lewis, and L. Esterle. Self-modeling and self-awareness. *Self-Aware Computing Systems*, pages 279–304, 2017.

[13] N. Bencomo. QuantUn: Quantification of uncertainty for the reassessment of requirements. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pages 236–240, Ottawa, ON, Canada, Aug. 2015. IEEE. ISBN 978-1-4673-6905-3.

[14] N. Bencomo. The Role of models@run.time in Autonomic Systems: Keynote. In *2017 IEEE International Conference on Autonomic Computing (ICAC)*, pages 293–294, July 2017.

[15] N. Bencomo and A. Belaggoun. Supporting decision-making for self-adaptive systems: from goal models to dynamic decision networks. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 221–236. Springer, 2013.

[16] N. Bencomo, J. Whittle, P. Sawyer, A. Finkelstein, and E. Letier. Requirements reflection: requirements as runtime entities. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - ICSE '10*, volume 2, page 199, Cape Town, South Africa, 2010. ACM Press. ISBN 978-1-60558-719-6.

[17] N. Bencomo, A. Belaggoun, and V. Issarny. Bayesian artificial intelligence for tackling uncertainty in self-adaptive systems: The case of dynamic decision networks. In *2013 2nd International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE)*, pages 7–13, May 2013.

[18] N. Bencomo, A. Belaggoun, and V. Issarny. Dynamic decision networks for decision-making in self-adaptive systems: A case study. In *2013 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 113–122, San Francisco, CA, USA, May 2013. IEEE. ISBN 978-1-4673-4401-2 978-1-4799-0344-3.

[19] N. Bencomo, S. Götz, and H. Song. Models@ run. time: a guided tour of the state of the art and research challenges. *Software & Systems Modeling*, 18(5):3049–3082, 2019.

[20] A. Bennaceur, A. Zisman, C. McCormick, D. Barthaud, and B. Nuseibeh. Won't take no for an answer: resource-driven requirements adaptation. In *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 77–88. IEEE, 2019.

[21] E. Bisong. Logistic regression. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 243–250. Springer, 2019.

[22] J. Bocanegra, L. H. Garcia Paucar, J. Pavlich-Mariscal, and N. Bencomo. Improving the Decision-Making Support in Context-Aware Applications: The Case of an Adaptive Virtual Education Learning Management System, Apr. 2018.

[23] T. Bolender, G. Bürvenich, M. Dalibor, B. Rumpe, and A. Wortmann. Self-adaptive manufacturing with digital twins. *arXiv preprint arXiv:2103.11941*, 2021.

[24] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[25] K. M. Bowers, E. M. Fredericks, and B. H. Cheng. Automated optimization of weighted non-functional objectives in self-adaptive systems. In *International Symposium on Search Based Software Engineering*, pages 182–197. Springer, 2018.

[26] K. M. Bowers, E. M. Fredericks, R. H. Hariri, and B. H. Cheng. Providentia: Using search-based heuristics to optimize satisficement and competing concerns between functional and non-functional objectives in self-adaptive systems. *Journal of Systems and Software*, 162:110497, 2020.

[27] R. D. Caldas, A. Rodrigues, E. B. Gil, G. N. Rodrigues, T. Vogel, and P. Pelliccione. A hybrid approach combining control theory and ai for engineering self-adaptive systems. In *Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 9–19, 2020.

[28] J. Cámara, A. Lopes, D. Garlan, and B. Schmerl. Adaptation impact and environment models for architecture-based self-adaptive systems. *Science of Computer Programming*, 127:50–75, 2016.

[29] J. Cámara, K. L. Bellman, J. O. Kephart, M. Autili, N. Bencomo, A. Diaconescu, H. Giese, S. Götz, P. Inverardi, S. Kounev, et al. Self-aware computing systems: Related concepts and research areas. *Self-Aware Computing Systems*, pages 17–49, 2017.

[30] J. Cámara, D. Garlan, W. G. Kang, W. Peng, and B. Schmerl. Uncertainty in self-adaptive systems. Technical report, Technical Report CMU-ISR-17-110. Institute for Software Research, Carnegie . . . , 2017.

[31] J. Cámara, W. Peng, D. Garlan, and B. Schmerl. Reasoning about sensing uncertainty and its reduction in decision-making for self-adaptation. *Science of Computer Programming*, 167:51–69, 2018.

[32] E. Campello, L. Spiezia, C. Simion, D. Tormene, G. Camporese, F. Dalla Valle, A. Poretto, C. Bulato, S. Gavasso, C. M. Radu, et al. Direct oral anticoagulants in patients with inherited thrombophilia and venous thromboembolism: a prospective cohort study. *Journal of the American Heart Association*, 9(23):e018917, 2020.

[33] B. Chen, X. Peng, Y. Yu, B. Nuseibeh, and W. Zhao. Self-adaptation through incremental generative model transformations at runtime. In *Proceedings of the 36th International Conference on Software Engineering - ICSE 2014*, pages 676–687, Hyderabad, India, 2014. ACM Press. ISBN 978-1-4503-2756-5.

[34] H.-T. Cheng. *Algorithms for partially observable Markov decision processes*. PhD thesis, University of British Columbia, 1988.

[35] J. Choi and K.-E. Kim. Inverse reinforcement learning in partially observable environments. *Journal of Machine Learning Research*, 12(Mar):691–730, 2011.

[36] T. Clark and V. Kulkarni. Adaptive complex systems: Digital twins. *Artificial Intelligence to Solve Pervasive Internet of Things Issues*, pages 211–238, 2021.

[37] A. Computing et al. An architectural blueprint for autonomic computing. *IBM White Paper*, 31(2006):1–6, 2006.

[38] A. Dardenne, A. Van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of computer programming*, 20(1-2):3–50, 1993.

[39] A. J. De Craen, T. J. Kaptchuk, J. G. Tijssen, and J. Kleijnen. Placebos and placebo effects in medicine: historical overview. *Journal of the Royal Society of Medicine*, 92 (10):511–515, 1999.

[40] A. O. de Sousa, C. I. Bezerra, R. M. Andrade, and J. M. Filho. Quality evaluation of self-adaptive systems: Challenges and opportunities. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, pages 213–218, 2019.

[41] M. M. Drugan. Reinforcement learning versus evolutionary computation: A survey on hybrid algorithms. *Swarm and evolutionary computation*, 44:228–246, 2019.

[42] K. Duan, S. S. Keerthi, and A. N. Poo. Evaluation of simple performance measures for tuning svm hyperparameters. *Neurocomputing*, 51:41–59, 2003.

[43] D. Dumitrescu, B. Lazzerini, L. C. Jain, and A. Dumitrescu. *Evolutionary computation*. CRC press, 2000.

[44] R. Edwards and N. Bencomo. Desire: further understanding nuances of degrees of satisfaction of non-functional requirements trade-off. In *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*, pages 12–18, 2018.

[45] G. Elahi and E. Yu. Requirements trade-offs analysis in the absence of quantitative measures: A heuristic method. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, pages 651–658, 2011.

[46] A. Elkhodary, N. Esfahani, and S. Malek. Fusion: a framework for engineering self-tuning self-adaptive software systems. In *Proceedings of the eighteenth ACM SIGSOFT international symposium on Foundations of software engineering*, pages 7–16, 2010.

[47] M. N. Elshafei, M. F. Mohamed, A. El-Bardissy, M. B. Ahmed, I. Abdallah, H. Elewa, and M. Danjuma. Comparative effectiveness and safety of direct oral anticoagulants compared to warfarin in morbidly obese patients with acute venous thromboembolism: systematic review and a meta-analysis. *Journal of Thrombosis and Thrombolysis*, 51: 388–396, 2021.

[48] N. Esfahani and S. Malek. Uncertainty in self-adaptive software systems. In *Software Engineering for Self-Adaptive Systems II*, pages 214–238. Springer, 2013.

[49] N. Esfahani, E. Kouroshfar, and S. Malek. Taming uncertainty in self-adaptive software. In *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering - SIGSOFT/FSE '11*, page 234, Szeged, Hungary, 2011. ACM Press. ISBN 978-1-4503-0443-6.

[50] R. Feldt and A. Magazinius. Validity threats in empirical software engineering research-an initial survey. In *Seke*, pages 374–379, 2010.

[51] A. Filieri and G. Tamburrelli. Probabilistic verification at runtime for self-adaptive systems. In *Assurances for Self-Adaptive Systems*, pages 30–59. Springer, 2013.

[52] E. M. Fredericks. Mitigating uncertainty at design time and run time to address assurance for dynamically adaptive systems. *Michigan S.University. PhD Thesis.*, 2015.

[53] E. M. Fredericks. *Mitigating Uncertainty at Design Time and Run Time to Address Assurance for Dynamically Adaptive Systems.* Michigan State University. Computer Science, 2015.

[54] E. M. Fredericks and B. H. Cheng. Automated generation of adaptive test plans for self-adaptive systems. In *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 157–167. IEEE, 2015.

[55] P. S. Ganney. Chapter 8 - information communications technology. In A. Taktak, P. Ganney, D. Long, and P. White, editors, *Clinical Engineering*, pages 107–131. Academic Press, Oxford, 2014. ISBN 978-0-12-396961-3.

[56] L. Garcia Paucar. Requirements-aware models to support better informed decision-making for self-adaptation using partially observable markov decision processes. *PhD thesis. Aston University. United Kingdom*, 2020. URL `https://publications.aston.ac.uk/id/eprint/41929/`.

[57] L. H. Garcia Paucar and N. Bencomo. Knowledge Base K Models to Support Trade-Offs for Self-Adaptation using Markov Processes. In *2019 IEEE 13th International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*, pages 11–16, June 2019. ISSN: 1949-3681, 1949-3673.

[58] L. H. Garcia Paucar and N. Bencomo. Knowledge Base K Models to Support Trade-Offs for Self-Adaptation using Markov Processes. In *2019 IEEE 13th International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*, pages 11–16, June 2019. ISSN: 1949-3681, 1949-3673.

[59] D. Garlan, S.-W. Cheng, A.-C. Huang, B. Schmerl, and P. Steenkiste. Rainbow: Architecture-based self-adaptation with reusable infrastructure. *Computer*, 37(10): 46–54, 2004.

[60] I. Gerostathopoulos, T. Vogel, D. Weyns, and P. Lago. How do we evaluate self-adaptive software systems?: A ten-year perspective of seams. In *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 59–70. IEEE, 2021.

[61] H. Giese, N. Bencomo, L. Pasquale, A. J. Ramirez, P. Inverardi, S. Wätzoldt, and S. Clarke. Living with Uncertainty in the Age of Runtime Models. In N. Bencomo, R. France, B. H. C. Cheng, and U. Aßmann, editors, *Models@run.time: Founda-*

tions, Applications, and Roadmaps, Lecture Notes in Computer Science, pages 47–100. Springer International Publishing, Cham, 2014. ISBN 978-3-319-08915-7.

[62] P. Gill, N. Jain, and N. Nagappan. Understanding network failures in data centers: measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011 Conference*, pages 350–361, 2011.

[63] M. Glinz. On non-functional requirements. In *15th IEEE International Requirements Engineering Conference (RE 2007)*, pages 21–26. IEEE, 2007.

[64] H. J. Goldsby, P. Sawyer, N. Bencomo, B. H. Cheng, and D. Hughes. Goal-based modeling of dynamically adaptive system requirements. In *15Th annual IEEE international conference and workshop on the engineering of computer based systems (ecbs 2008)*, pages 36–45. IEEE, 2008.

[65] P. S. Gray, J. B. Williamson, D. A. Karp, and J. R. Dalphin. *The research imagination: An introduction to qualitative and quantitative methods*. Cambridge University Press, 2007.

[66] D. GUIDANCE. Guidance for industry non-inferiority clinical trials. *Center for Biologics Evaluation and Research (CBER)*, 220, 2010.

[67] S. Hassan, N. Bencomo, and R. Bahsoon. Minimizing Nasty Surprises with Better Informed Decision-Making in Self-Adaptive Systems. In *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 134–145, Florence, Italy, May 2015. IEEE. ISBN 978-0-7695-5567-6.

[68] C. F. Hayes, R. Rădulescu, E. Bargiacchi, J. Källström, M. Macfarlane, M. Reymond, T. Verstraeten, L. M. Zintgraf, R. Dazeley, F. Heintz, et al. A practical guide to multi-objective reinforcement learning and planning. *arXiv preprint arXiv:2103.09568*, 2021.

[69] S. M. Hezavehi, D. Weyns, P. Avgeriou, R. Calinescu, R. Mirandola, and D. Perez-Palacin. Uncertainty in self-adaptive systems: A research community perspective. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 15(4):1–36, 2021.

[70] H. J. Holz, A. Applin, B. Haberman, D. Joyce, H. Purchase, and C. Reed. Research methods in computing: What are they, and how should we teach them? In *Working*

group reports on ITiCSE on Innovation and technology in computer science education, pages 96–114. 2006.

[71] S. Huma. Ni trial: Dataset and results repository, 2023. URL `https://github.com/humasamin/NI_Trial_Repo.git`.

[72] M. U. Iftikhar, G. S. Ramachandran, P. Bollansee, D. Weyns, and D. Hughes. DeltaIoT: A Self-Adaptive Internet of Things Exemplar. In *2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 76–82, Buenos Aires, Argentina, May 2017. IEEE. ISBN 978-1-5386-1550-8.

[73] M. U. Iftikhar, D. Weyns, F. Oquendo, Linnéuniversitetet, and Fakultetsnämnden för naturvetenskap och teknik. *A Model-Based Approach to Engineer Self-Adaptive Systems with Guarantees*. Eget förlag, 2017. ISBN 978-91-88761-05-7. OCLC: 1026743934.

[74] M. Ji, A. C. Veitch, and J. Wilkes. Seneca: remote mirroring done write. In *USENIX Annual Technical Conference, General Track*, 2003.

[75] C. Johnson. Basic research skills in computing science. *Department of Computer Science, Glasgow University, UK*, 2006.

[76] S. R. Johnson, G. A. Tomlinson, G. A. Hawker, J. T. Granton, H. A. Grosbein, and B. M. Feldman. A valid and reliable belief elicitation method for bayesian priors. *Journal of Clinical Epidemiology*, 2010. ISSN 0895-4356.

[77] G. Kaur and S. Bawa. A survey of requirement prioritization methods. *Int. J. Eng. Res. Technol*, 2(5):958–962, 2013.

[78] K. Keeton, C. Santos, D. Beyer, J. Chase, and J. Wilkes. Designing for Disasters. Mar. 2004.

[79] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer*, 36 (1):41–50, 2003.

[80] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer*, 36 (1):41–50, 2003.

[81] S. Khadka and K. Tumer. Evolutionary reinforcement learning. *arXiv preprint arXiv:1805.07917*, 2018.

[82] M. J. Khan, S. Shamail, and M. M. Awais. Decision Making in Autonomic Managers Using Fuzzy Inference System. In *2009 Fifth International Conference on Autonomic and Autonomous Systems*, pages 214–219, Apr. 2009.

[83] C. Kinneer, Z. Coker, J. Wang, D. Garlan, and C. L. Goues. Managing uncertainty in self-adaptive systems with plan reuse and stochastic search. In *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*, pages 40–50, 2018.

[84] M. J. Kochenderfer. *Decision making under uncertainty: theory and application*. MIT press, 2015.

[85] J. R. Koza et al. *Genetic programming II*, volume 17. MIT press Cambridge, 1994.

[86] C. Krupitzer, F. M. Roth, S. VanSyckel, G. Schiele, and C. Becker. A survey on engineering approaches for self-adaptive systems. *Pervasive and Mobile Computing*, 17:184–206, 2015.

[87] M. A. Langford and B. H. Cheng. Enhancing learning-enabled software systems to address environmental uncertainty. In *2019 IEEE International Conference on Autonomic Computing (ICAC)*, pages 115–124. IEEE, 2019.

[88] M. A. Langford and B. H. Cheng. "know what you know": Predicting behavior for learning-enabled systems when facing uncertainty. In *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 78–89. IEEE, 2021.

[89] M. A. Langford, G. A. Simon, P. K. McKinley, and B. H. Cheng. Applying evolution and novelty search to enhance the resilience of autonomous systems. In *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 63–69. IEEE, 2019.

[90] T. Le and S. Moh. An energy-efficient topology control algorithm based on reinforcement learning for wireless sensor networks. *Int. J. Control Autom*, 10(5):233–244, 2017.

[91] T. T. Le and S. Moh. Reinforcement-learning-based topology control for wireless sensor networks. *Proceedings of the Grid and Distributed Computing*, 2016:22–7, 2016.

[92] J. Lehman and K. O. Stanley. Novelty search and the problem with objectives. *Genetic programming theory and practice IX*, pages 37–56, 2011.

[93] R. d. Lemos, editor. *Software engineering for self-adaptive systems II: international seminar, Dagstuhl Castle, Germany, October 24-29, 2010: revised selected and invited papers*. Number 7475 in Lecture notes in computer science. Springer, Berlin ; New York, 2013. ISBN 978-3-642-35812-8. OCLC: ocn839358754.

[94] S. Liaskos, S. A. McIlraith, S. Sohrabi, and J. Mylopoulos. Representing and reasoning about preferences in requirements engineering. *Requirements Engineering*, 16(3):227–249, Sept. 2011. ISSN 0947-3602, 1432-010X.

[95] C. Liu, X. Xu, and D. Hu. Multiobjective Reinforcement Learning: A Comprehensive Overview. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3): 385–398, Mar. 2015. ISSN 2168-2216.

[96] Y. Liu, A. Jain, C. Eng, D. H. Way, K. Lee, P. Bui, K. Kanada, G. de Oliveira Marinho, J. Gallegos, S. Gabriele, et al. A deep learning system for differential diagnosis of skin diseases. *Nature medicine*, 26(6):900–908, 2020.

[97] P. Maes. Computational reflection. *The Knowledge Engineering Review*, 3(1):1–19, 1988.

[98] M. Maggio, A. V. Papadopoulos, A. Filieri, and H. Hoffmann. Automated control of multiple software goals using multiple actuators. In *Proceedings of the 2017 11th joint meeting on foundations of software engineering*, pages 373–384, 2017.

[99] S. Menard. *Applied logistic regression analysis*, volume 106. Sage, 2002.

[100] S. Mihai, W. Davis, D. V. Hung, R. Trestian, M. Karamanoglu, B. Barn, R. Prasad, H. Venkataraman, and H. X. Nguyen. A digital twin framework for predictive maintenance in industry 4.0. 2021.

[101] A. S. Mikhalev, V. S. Tynchenko, V. A. Nelyub, N. M. Lugovaya, V. A. Baranov,

V. V. Kukartsev, R. B. Sergienko, and S. O. Kurashkin. The orb-weaving spider algorithm for training of recurrent neural networks. *Symmetry*, 14(10):2036, 2022.

[102] G. A. Moreno, J. Cámara, D. Garlan, and B. Schmerl. Efficient decision-making under uncertainty for proactive self-adaptation. In *2016 IEEE International Conference on Autonomic Computing (ICAC)*, pages 147–156. IEEE, 2016.

[103] G. A. Moreno, J. Cámara, D. Garlan, and B. Schmerl. Flexible and efficient decision-making for proactive latency-aware self-adaptation. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 13(1):1–36, 2018.

[104] D. E. Morris, J. E. Oakley, and J. A. Crowe. A web-based tool for eliciting probability distributions from experts. *Environmental Modelling & Software*, 2014. ISSN 1364-8152.

[105] S. Nagel, D. Sinha, D. Day, W. Reith, R. Chapot, P. Papanagiotou, E. A. Warburton, P. Guyler, S. Tysoe, K. Fassbender, et al. e-aspects software is non-inferior to neuroradiologists in applying the aspect score to computed tomography scans of acute ischemic stroke patients. *International Journal of Stroke*, 12(6):615–622, 2017.

[106] T. N. Nguyen, C. V. Ho, and T. T. Le. A topology control algorithm in wireless sensor networks for iot-based applications. In *2019 International Symposium on Electrical and Electronics Engineering (ISEE)*, pages 141–145. IEEE, 2019.

[107] A. O'Hagan. Probabilistic uncertainty specification: Overview, elaboration techniques and their application to a mechanistic model of carbon flux. *Environmental Modelling & Software*, 2012. ISSN 1364-8152.

[108] A. O'Hagan, C. E. Buck, A. Daneshkhah, J. R. Eiser, P. H. Garthwaite, D. J. Jenkinson, J. E. Oakley, and T. Rakow. Uncertain judgements: eliciting experts' probabilities. 2006.

[109] S. Ostmeier, J. J. Heit, B. Axelrod, L.-J. Li, G. Zaharchuk, B. F. Verhaaren, A. Mahammedi, S. Christensen, and M. G. Lansberg. Non-inferiority of deep learning model to segment acute stroke on non-contrast ct compared to neuroradiologists. *arXiv preprint arXiv:2211.15341*, 2022.

[110] C. L. Owen. Design research: building the knowledge base. *Design studies*, 19(1): 9–20, 1998.

[111] J. M. Parra-Ullauri, A. García-Domínguez, L. H. García-Paucar, and N. Bencomo. Temporal models for history-aware explainability. In *Proceedings of the 12th system analysis and modelling conference*, pages 155–164, 2020.

[112] L. H. G. Paucar and N. Bencomo. RE-STORM: mapping the decision-making problem and non-functional requirements trade-off to partially observable markov decision processes. In *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems - SEAMS '18*, pages 19–25, Gothenburg, Sweden, 2018. ACM Press. ISBN 978-1-4503-5715-9.

[113] L. H. G. Paucar, N. Bencomo, and K. K. F. Yuen. ARRoW: automatic runtime reappraisal of weights for self-adaptation. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing - SAC '19*, pages 1584–1591, Limassol, Cyprus, 2019. ACM Press. ISBN 978-1-4503-5933-7.

[114] X. Peng, B. Chen, Y. Yu, and W. Zhao. Self-Tuning of Software Systems Through Goal-based Feedback Loop Control. In *2010 18th IEEE International Requirements Engineering Conference*, pages 104–107, Sydney, Australia, Sept. 2010. IEEE. ISBN 978-1-4244-8022-7.

[115] X. Peng, B. Chen, Y. Yu, and W. Zhao. Self-tuning of software systems through dynamic quality tradeoff and value-based feedback control loop. *Journal of Systems and Software*, 85(12):2707–2719, Dec. 2012. ISSN 01641212.

[116] M. Pergher and B. Rossi. Requirements prioritization in software engineering: a systematic mapping study. In *2013 3rd International Workshop on Empirical Requirements Engineering (EmpiRE)*, pages 40–44. IEEE, 2013.

[117] J. Pimentel, M. Lencastre, and J. Castro. Implicit priorities in adaptation requirements. In *2016 10th International Conference on the Quality of Information and Communications Technology (QUATIC)*, pages 83–86. IEEE, 2016.

[118] A. M. Pitangueira, R. S. P. Maciel, and M. Barros. Software requirements selection

and prioritization using sbse approaches: A systematic review and mapping of the literature. *Journal of Systems and Software*, 103:267–280, 2015.

[119] N. A. Qureshi, S. Liaskos, and A. Perini. Reasoning about adaptive requirements for self-adaptive systems at runtime. In *2011 2nd International Workshop on Requirements@Run.Time*, pages 16–22, Aug. 2011.

[120] A. Ramirez, B. Cheng, N. Bencomo, and P. Sawyer. Relaxing claims: Coping with uncertainty while evaluating assumptions at run time. *MODELS*, 2012.

[121] A. J. Ramirez and B. H. Cheng. Evolving models at run time to address functional and non-functional adaptation requirements. In *Proceedings of the 4th International Workshop on Models at Runtime*, 2009.

[122] A. J. Ramirez and B. H. C. Cheng. Automatic Derivation of Utility Functions for Monitoring Software Requirements. In J. Whittle, T. Clark, and T. Kühne, editors, *Model Driven Engineering Languages and Systems*, Lecture Notes in Computer Science, pages 501–516. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-24485-8.

[123] A. J. Ramirez, B. H. C. Cheng, N. Bencomo, and P. Sawyer. Relaxing Claims: Coping with Uncertainty While Evaluating Assumptions at Run Time. In *Model Driven Engineering Languages and Systems*, volume 7590, pages 53–69. Springer Berlin Heidelberg, 2012.

[124] A. J. Ramirez, A. C. Jensen, and B. H. Cheng. A taxonomy of uncertainty for dynamically adaptive systems. In *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 99–108. IEEE, 2012.

[125] M. Rinard, C. Cadar, and H. H. Nguyen. Exploring the acceptability envelope. In *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 21–30, 2005.

[126] A. Rodrigues, R. D. Caldas, G. N. Rodrigues, T. Vogel, and P. Pelliccione. A learning approach to enhance assurances for real-time self-adaptive systems. In *2018 IEEE/ACM 13th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 206–216. IEEE, 2018.

[127] D. Roijers. *Multi-objective decision-theoretic planning*. 2016. ISBN 978-94-6328-039-6. OCLC: 6893481195.

[128] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A Survey of Multi-Objective Sequential Decision-Making. *Journal of Artificial Intelligence Research*, 48:67–113, Oct. 2013. ISSN 1076-9757.

[129] D. M. Roijers, S. Whiteson, and F. A. Oliehoek. Point-Based Planning for Multi-Objective POMDPs. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, June 2015.

[130] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online Planning Algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, July 2008. ISSN 1076-9757.

[131] M. Saadatmand and S. Tahvili. A Fuzzy Decision Support Approach for Model-Based Tradeoff Analysis of Non-functional Requirements. In *2015 12th International Conference on Information Technology - New Generations*, pages 112–121, Las Vegas, NV, USA, Apr. 2015. IEEE. ISBN 978-1-4799-8828-0.

[132] H. Samin. Rdmsim: Remote data mirroring simulator for decision-making in self-adaptive systems. 2021. URL `https://github.com/humasamin/RDMSimExemplar.git`.

[133] H. Samin. Pri-aware results repository:. Technical report, 2022. URL `https://github.com/humasamin/PriAwaRE_Repo`.

[134] H. Samin. Pri-aware code:, 2023. URL `https://github.com/humasamin/PriAwaRE.git`.

[135] H. Samin, N. Bencomo, and P. Sawyer. Pri-aware: Tool support for priority-aware decision-making under uncertainty. In *2021 IEEE 29th International Requirements Engineering Conference (RE), Poster and Tools Demonstration Track*. IEEE, 2021.

[136] H. Samin, L. Garcia Paucar, B. Nelly, and P. Sawyer. Towards priority-awareness in autonomous intelligent systems. In *36th ACM/SIGAPP Symposium On Applied Computing (SAC)*. ACM, 2021.

[137] H. Samin, L. H. G. Paucar, N. Bencomo, C. M. C. Hurtado, and E. M. Fredericks. RDMSim: An Exemplar for Evaluation and Comparison of Decision-Making Techniques for Self-Adaptation. In *16th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Artefact Track*, 2021.

[138] H. Samin, N. Bencomo, and P. Sawyer. Decision-making under uncertainty: be aware of your priorities. *International Journal on Software and Systems Modeling (SoSyM)*, 2022.

[139] P. Sawyer, N. Bencomo, J. Whittle, E. Letier, and A. Finkelstein. Requirements-Aware Systems: A Research Agenda for RE for Self-adaptive Systems. In *2010 18th IEEE International Requirements Engineering Conference*, pages 95–103, Sept. 2010.

[140] J. Schumi and J. T. Wittes. Through the looking glass: understanding non-inferiority. *Trials*, 12(1):1–12, 2011.

[141] C. B. Seaman. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on software engineering*, 25(4):557–572, 1999.

[142] H. J. Seltman. Experimental design and analysis, 2012.

[143] J. Shanavas and S. Simi. An energy efficient topology control scheme with connectivity learning in wireless networks. In *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1770–1774. IEEE, 2014.

[144] G. Shani, J. Pineau, and R. Kaplow. A survey of point-based pomdp solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51, 2013.

[145] S. Y. Shin, S. Nejati, M. Sabetzadeh, L. C. Briand, C. Arora, and F. Zimmer. Dynamic adaptation of software-defined networks for iot systems: A search-based approach. In *Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 137–148, 2020.

[146] V. E. Silva Souza, A. Lapouchnian, and J. Mylopoulos. System Identification for Adaptive Software Systems: A Requirements Engineering Perspective. In *Conceptual Modeling – ER 2011*, Lecture Notes in Computer Science, pages 346–361. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-24606-7.

[147] V. E. Silva Souza, A. Lapouchnian, W. N. Robinson, and J. Mylopoulos. Awareness requirements for adaptive systems. In *Proceedings of the 6th international symposium on Software engineering for adaptive and self-managing systems*, pages 60–69. ACM, 2011.

[148] H. Soh and Y. Demiris. Evolving policies for multi-reward partially observable Markov decision processes (MR-POMDPs). In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 713–720, Jan. 2011.

[149] H. Soh, Y. Demiris, H. Soh, and Y. Demiris. Multi-Reward Policies for Medical Applications: Anthrax Attacks and Smart Wheelchairs. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, pages 471–478, 2011.

[150] A. Somani, N. Ye, D. Hsu, and W. S. Lee. DESPOT: Online POMDP Planning with Regularization. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1772–1780. Curran Associates, Inc., 2013.

[151] I. Sommerville. Software engineering, 2010.

[152] V. E. S. Souza, A. Lapouchnian, W. N. Robinson, and J. Mylopoulos. Awareness requirements. In *Software Engineering for Self-Adaptive Systems II*, pages 133–161. Springer, 2013.

[153] M. T. J. Spaan. Partially Observable Markov Decision Processes. *Reinforcement Learning*, page 27, 2012.

[154] M. T. J. Spaan and N. Vlassis. Perseus: Randomized Point-based Value Iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, Aug. 2005. ISSN 1076-9757. arXiv: 1109.2145.

[155] A. Sutcliffe, P. Sawyer, G. Stringer, S. Couth, L. J. Brown, A. Gledson, C. Bull, P. Rayson, J. Keane, X.-j. Zeng, et al. Known and unknown requirements in healthcare. *Requirements engineering*, 25(1):1–20, 2020.

[156] F. Tao, H. Zhang, A. Liu, and A. Y. Nee. Digital twin in industry: State-of-the-art. *IEEE Transactions on Industrial Informatics*, 15(4):2405–2415, 2018.

[157] R. Thakurta. Understanding requirement prioritization artifacts: a systematic mapping study. *Requirements engineering*, 22(4):491–526, 2017.

[158] E. Triantaphyllou. Multi-criteria decision making methods. In *Multi-criteria decision making methods: A comparative study*, pages 5–21. Springer, 2000.

[159] V. L. Vachhani, V. K. Dabhi, and H. B. Prajapati. Survey of multi objective evolutionary algorithms. In *2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]*, pages 1–9. IEEE, 2015.

[160] A. J. Vickers and A. J. de Craen. Why use placebos in clinical trials? a narrative review of the methodological literature. *Journal of clinical epidemiology*, 53(2):157–161, 2000.

[161] J. Walker. Non-inferiority statistics and equivalence studies. *BJA education*, 19(8): 267, 2019.

[162] E. Walraven. SolvePOMDP, 2017. URL `https://www.erwinwalraven.nl/solvepomdp/`.

[163] H. J. Weerts, A. C. Mueller, and J. Vanschoren. Importance of tuning hyperparameters of machine learning algorithms. *arXiv preprint arXiv:2007.07588*, 2020.

[164] K. Welsh, P. Sawyer, and N. Bencomo. Towards requirements aware systems: Runtime resolution of design-time assumptions. In *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*, pages 560–563, Nov. 2011.

[165] D. Weyns. Software engineering of self-adaptive systems. In *Handbook of Software Engineering*, pages 399–443. Springer, 2019.

[166] D. Weyns. *An Introduction to Self-adaptive Systems: A Contemporary Software Engineering Perspective*. John Wiley & Sons, 2020.

[167] D. Weyns and U. Iftikhar. Model-based simulation at runtime for self-adaptive systems. *Proceeding Models at Runtime, Würzburg 2016*, pages 1–9, 2016.

[168] J. Whittle, P. Sawyer, N. Bencomo, B. H. Cheng, and J.-M. Bruel. RELAX: Incorporating Uncertainty into the Specification of Self-Adaptive Systems. In *2009 17th IEEE*

International Requirements Engineering Conference, pages 79–88, Atlanta, Georgia, USA, Aug. 2009. IEEE. ISBN 978-0-7695-3761-0.

[169] J. Whittle, P. Sawyer, N. Bencomo, B. H. C. Cheng, and J.-M. Bruel. RELAX: a language to address uncertainty in self-adaptive systems requirement. *Requirements Engineering*, 15(2):177–196, June 2010. ISSN 0947-3602, 1432-010X.

[170] R. Wieringa. Design science methodology: principles and practice. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2*, pages 493–494, 2010.

[171] R. J. Wieringa. *Design science methodology for information systems and software engineering*. Springer, 2014.

[172] R. Wohlrab, R. Meira-Góes, and M. Vierhauser. Run-time adaptation of quality attributes for automated planning. In *Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 98–105, 2022.

[173] F. Xia, L. T. Yang, L. Wang, and A. Vinel. Internet of Things. *International journal of communication systems*, 25, 2012.

[174] K. K. F. Yuen. The Primitive Cognitive Network Process in healthcare and medical decision making: Comparisons with the Analytic Hierarchy Process. *Applied Soft Computing*, 14:109–119, Jan. 2014. ISSN 1568-4946.

[175] L. A. Zadeh. Possibility theory and soft data analysis. In *Mathematical frontiers of the social and policy sciences*, pages 69–129. Routledge, 2019.

[176] P. Zave and M. Jackson. Four dark corners of requirements engineering. *ACM transactions on Software Engineering and Methodology (TOSEM)*, 6(1):1–30, 1997.

[177] J. Zhang and L. Xing. A survey of multiobjective evolutionary algorithms. In *2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, volume 1, pages 93–100. IEEE, 2017.

[178] S. Zhifei and E. Meng Joo. A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 5(3):293–311, 2012.

[179] I. Zhukovyts'kyy and I. Tsykalo. Methods of using the neural network to detect new types of network attacks. In *International scientific and technical conference Information technologies in metallurgy and machine building*, pages 228–231, 2022.

Appendices

Appendix A

Experimental Setup MR-POMDP++

This appendix presents the experimental setup of the MR-POMDP++ model for both the case studies. Section A.1 presents the transition and observation model for DELTA-IoT which is followed by the transition and observation model for RDMSim in Section A.2.

A.1 DELTA-IoT: MR-POMDP++ Experimental Setup

This section describes the transition and observation model probabilities for the MR-POMDP++ model for the case of DELTA-IoT network.

A.1.1 Transition Model

The DELTA-IoT network focuses on two NFRs of MinEC and MinPL which leads to the identification of 4 states for MR-POMDP++. Based on this, the transition function is factored as marginal conditional probabilities of NFRs (i.e. $P(MinEC' \mid MinEC, a)$ and $P(MinPL' \mid MinPL, a)$) using the property of conditional independence and Bayes rule [56] as follows:

$$T(s, a, s') = P(s'|s, a) = P(MinEC' \mid MinEC, a)P(MinPL' \mid MinPL, a)$$

The transition probabilities for the IoT case, defined by the domain experts [138], are presented in the Table A.1.

A.1.2 Observation Model

In case of DELTA-IoT, one monitorable of Signal-to-Noise Ratio (SNR) on the links associated with the nodes is considered. The SNR can a value of greater than, less than or equal to zero.

Table A.1: Transition Probabilities for NFRs

NFR_1=Minimization of Energy Consumption(MinEC)				
Action(A)	**$MinEC_t$**	**$MinPL_t$**	**P($MinEC_{t+1} = True$)**	**P($MinEC_{t+1} = False$)**
DTP	True	True	0.89	0.11
DTP	True	False	0.92	0.08
DTP	False	True	0.84	0.16
DTP	False	False	0.87	0.13
ITP	True	True	0.85	0.15
ITP	True	False	0.88	0.12
ITP	False	True	0.73	0.27
ITP	False	False	0.76	0.24
NFR_2=Minimize Packet Loss(MinPL)				
Action(A)	**$MinEC_t$**	**$MinPL_t$**	**P($MinPL_{t+1} = True$)**	**P($MinPL_{t+1} = False$)**
DTP	True	True	0.92	0.08
DTP	True	False	0.89	0.11
DTP	False	True	0.96	0.04
DTP	False	False	0.91	0.01
ITP	True	True	0.98	0.02
ITP	True	False	0.96	0.04
ITP	False	True	0.99	0.01
ITP	False	False	0.97	0.03

Hence, based on this the observation model comprises of the probabiliies for the three cases. The observation probability of 0.33 is considered for each situation, as all three situations are considered to be observed equally likely at runtime [72].

A.2 RDMSim: Experimental Setup

This section describes the transition and observation model probabilities for the MR-POMDP++ for the case of RDMSim network.

A.2.1 Transition Model

The RDMSim network focuses on three NFRs MinC, MaxR and MaxP which leads to the identification of 8 states for MR-POMDP++. Similar to the case of DELTA-IoT, the transition probabilities *T(s,a,s')* have been factored as marginal conditional probabilities of NFRs (i.e. $P(MinC' \mid MinC, a)$, $P(MaxR' \mid MaxR, a)$ and $P(MaxP' \mid MaxP, a)$) using the property of conditional independence and Bayes rule [112] as follows:

$$T(s, a, s') = P(s'|s, a) = P(MinC' \mid MinC, a)P(MaxR' \mid MaxR, a)P(MaxP' \mid MaxP, a)$$

The transition probabilities for the RDM network are shown in the Tables A.2, A.3 and A.4. These transition probabilities are provided by the domain experts [56].

A.2.2 Observation Model

In the case of the RDM network, three network parameters Total Bandwidth Consumption (BC), Active Network Links (ANL) and Total Time to Write Data (TTW) related to the NFRs

Table A.2: Transition Probabilities for NFR MinC

NFR$_1$=Minimization of Cost (MinC)					
Action(A)	**MinC$_t$**	**MaxR$_t$**	**MaxP$_t$**	**P(MinC$_{t+1}=T$)**	**P(MinC$_{t+1}=F$)**
MST	True	True	True	0.9	0.1
MST	True	True	False	0.88	0.12
MST	True	False	True	0.92	0.08
MST	True	False	False	0.9	0.1
MST	False	True	True	0.85	0.15
MST	False	True	False	0.83	0.17
MST	False	False	True	0.87	0.13
MST	False	False	False	0.85	0.15
RT	True	True	True	0.86	0.14
RT	True	True	False	0.84	0.16
RT	True	False	True	0.88	0.12
RT	True	False	False	0.86	0.14
RT	False	True	True	0.73	0.27
RT	False	True	False	0.71	0.29
RT	False	False	True	0.75	0.25
RT	False	False	False	0.73	0.27

Table A.3: Transition Probabilities for NFR MaxR

NFR$_2$=Maximization of Reliability (MaxR)					
Action(A)	**MinC$_t$**	**MaxR$_t$**	**MaxP$_t$**	**P(MaxR$_{t+1}=T$)**	**P(MaxR$_{t+1}=F$)**
MST	True	True	True	0.91	0.09
MST	True	True	False	0.93	0.07
MST	True	False	True	0.89	0.11
MST	True	False	False	0.91	0.09
MST	False	True	True	0.93	0.07
MST	False	True	False	0.95	0.05
MST	False	False	True	0.91	0.09
MST	False	False	False	0.93	0.07
RT	True	True	True	0.95	0.05
RT	True	True	False	0.97	0.03
RT	True	False	True	0.93	0.07
RT	True	False	False	0.95	0.05
RT	False	True	True	0.97	0.03
RT	False	True	False	0.99	0.01
RT	False	False	True	0.95	0.05
RT	False	False	False	0.97	0.03

Table A.4: Transition Probabilities for NFR MaxP

NFR$_1$=Maximization of Performance (MaxP)					
Action(A)	**MinC$_t$**	**MaxR$_t$**	**MaxP$_t$**	**P(MaxP$_{t+1}=T$)**	**P(MaxP$_{t+1}=F$)**
MST	True	True	True	0.9	0.1
MST	True	True	False	0.85	0.15
MST	True	False	True	0.92	0.08
MST	True	False	False	0.87	0.13
MST	False	True	True	0.88	0.12
MST	False	True	False	0.83	0.17
MST	False	False	True	0.9	0.1
MST	False	False	False	0.85	0.15
RT	True	True	True	0.82	0.18
RT	True	True	False	0.75	0.25
RT	True	False	True	0.84	0.16
RT	True	False	False	0.77	0.23
RT	False	True	True	0.8	0.2
RT	False	True	False	0.73	0.27
RT	False	False	True	0.82	0.18
RT	False	False	False	0.75	0.25

Table A.5: Observation Probabilities for RDM network

Mon$_1$=Total Bandwidth Consumption(TBC)				
Action(A)	**MinC$_t$**	**P(TBC$_{t+1} < x$)**	**P(TBC$_{t+1} in [x, y]$)**	**P(TBC$_{t+1} >= y$)**
MST	True	0.8	0.15	0.05
MST	False	0.72	0.18	0.1
RT	True	0.78	0.16	0.06
RT	False	0.68	0.2	0.12
Mon$_2$=Active Network Links(ANLs)				
Action(A)	**MaxR$_t$**	**P(ANLs$_{t+1} < r$)**	**P(ANLs$_{t+1} in [r, s]$)**	**P(ANLs$_{t+1} >= s$)**
MST	True	0.06	0.16	0.78
MST	False	0.12	0.2	0.68
RT	True	0.05	0.15	0.8
RT	False	0.1	0.18	0.72
Mon$_3$=Time to Write(TTW)				
Action(A)	**MaxP$_t$**	**P(TTW$_{t+1} < f$)**	**P(TTW$_{t+1} in [f, g]$)**	**P(TTW$_{t+1} >= g$)**
MST	True	0.83	0.13	0.04
MST	False	0.67	0.23	0.1
RT	True	0.8	0.15	0.05
RT	False	0.63	0.25	0.12

MinC, MaxR and MaxP, respectively, are considered as the monitorable variables. Similar to the transition model, the observation model is also factored into the product of conditional probabilities [112] as follows:

$$O(s', a, z) = P(z|s', a) = P(Mon_1, ..Mon_n|s', a)$$

Therefore,

$$P(z|s', a) = P(TBC, ANL, TTW|s', a) = P(TBC|s', a)P(ANL|s', a)P(TTW|s', a)$$

The probabilities for the observation model of the RDM network are presented in Table A.5. These probabilities are provided by the domain experts [56, 112].

Appendix B

Comparison of Pri-AwaRE with RE-STORM-ARROW

This appendix provides the details of the experimental evaluations for the Remote Data Mirroring (RDM) network [78, 137]. Experiments have been performed to compare the Pri-AwaRE approach with the approach of RE-STORM-ARROW [113]. The RE-STORM-ARROW is based on the technique of primitive cognitive network process (P-CNP) [174] and RE-STORM. The approach provides the support of updating initially defined reward values for the technique of RE-STORM. The implementation of RE-STORM-ARROW makes use of the DESPOT POMDP solver [112, 150]. All the experiments have been performed using Lenovo Thinkpad with intel Core i7, 8th Gen processor and 16 GB RAM. Next, the experimental setup and requirements specifications for the experiments are discussed.

B.1 Experimental Setup

The experimental setup involves the usage of the same initial setup of the RDM network and MR-POMDP++ model as described in Chapter 6. The initial setup for comparisons with the RE-STORM-ARROW considers a different set of Requirements Specifications ($\boldsymbol{R}$) defined by the experts of RE-STORM [57, 112]. The requirements specifications for the experiments are as follows:

R1: *The probability of satisfaction of Minimization of Cost shall be greater than or equal to 0.70. i.e.* $\boldsymbol{P(MinC{=}True){>}{=}0.70}$.

R2: *The probability of satisfaction of Maximization of Reliability shall be greater than or equal to 0.85 i.e.* $\boldsymbol{P(MaxR{=}True){>}{=}0.85}$.

R3: *The probability of satisfaction of Maximization of Performance shall be greater than or equal to 0.75. i.e.* ***P(MaxP=True)>=0.75****.*

Next, the experiment results for all the dynamic scenarios S_1 to S_6 of the RDM, presented in Chapter 5, are discussed.

B.2 Experiments Results

The experiments results for all the scenarios are presented in Figs. B.1 to B.6. Let's observe, Figs. B.1, B.2 and B.3 presenting the results for scenarios S_1, S_2 and S_3 respectively. Under these scenarios, the approach of Pri-AwaRE shows better satisfaction levels of NFRs MinC and MaxP compared to the approach of RE-STORM-ARROW. On the other hand, RESTORM-ARROW maintains higher reliability levels than Pri-AwaRE under scenarios S_1 and S_3. The reason behind this behavior is that the RE-STORM-ARROW, using P-CNP, supports the update of the initially defined reward values for the RE-STORM approach. This is not the case with Pri-AwaRE. In case of Pri-AwaRE, the autonomous tuning of priorities is provided at runtime without the update of the initially defined reward values. Furthermore, RE-STORM-ARROW, even supported by the update of reward values, does show poor satisfaction levels of NFRs. For example, under scenario S_5, the approach of Pri-AwaRE shows better satisfaction levels for all the NFRS compared to RE-STORM-ARROW, as presented in Fig. B.5. Both the approaches, show comparable results under scenarios S_4 and S_6. Therefore, from the results, it can be deduced that the Pri-AwaRE approach offers higher levels of satisfaction of NFRs in comparison to the single-objective technique of RE-STORM-ARROW. Furthermore, the approach of RE-STORM-ARROW does support the update of the initial reward values, this update is not performed autonomously. Instead, it requires external support from P-CNP that causes efficiency problems.

B.2.1 Summary of Findings

In summary, the experiment results show that Pri-AwaRE shows compliance with the requirements specification for the NFRs under almost all of the scenarios, as presented in Fig. B.10. The results show a confidence level of 95 percent. The confidence intervals along with the standard error for average satisfaction of NFRs for both Pri-AwaRE and RE-

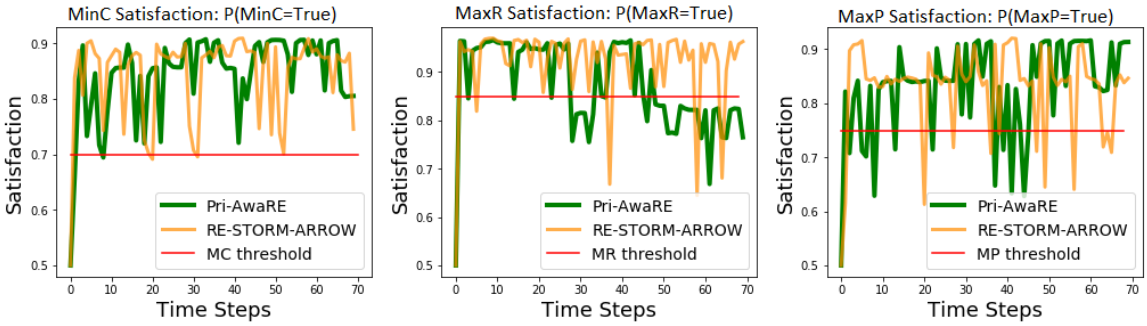


Figure B.1: Satisfaction of NFRs over Time under Scenario S_1

Figure B.2: Satisfaction of NFRs over Time under Scenario S_2

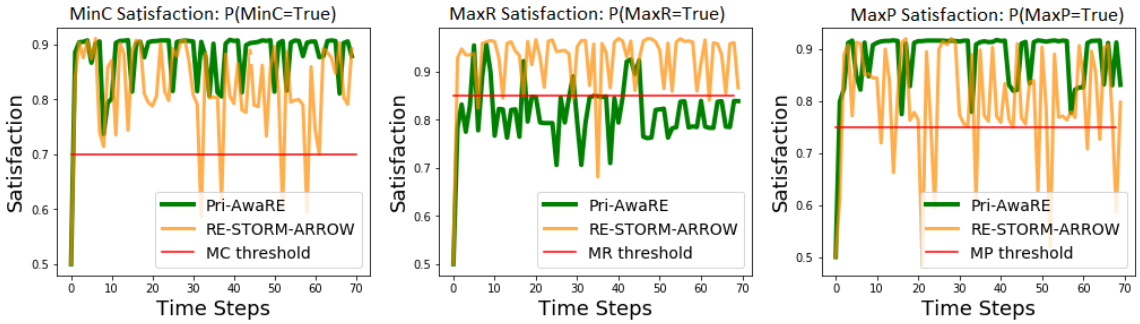


Figure B.3: Satisfaction of NFRs over Time under Scenario S_3

Figure B.4: Satisfaction of NFRs over Time under Scenario S_4

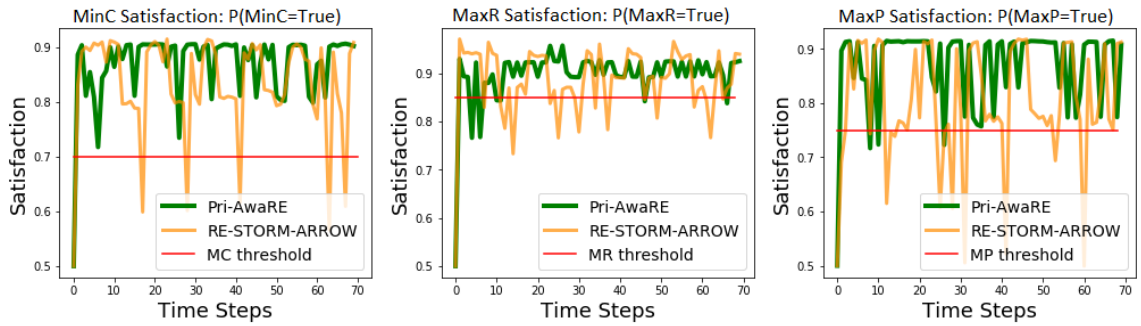


Figure B.5: Satisfaction of NFRs over Time under Scenario S_5

Figure B.6: Satisfaction of NFRs over Time under Scenario S_6

STORM-ARROW are presented in Tables **??** and **??**, respectively. For both Pri-AwaRE and RE-STORM-ARROW, the average satisfaction levels for MinC and MaxP are above the satisfaction thresholds under all the scenarios. Hence, they show compliance with the threshold requirements of $P(MinC = True) >= 0.70$ and $P(MaxP = True) >= 0.75$. For example, under scenario S_1, in case of Pri-AwaRE, the average satisfaction level for MinC is 0.8439 with a standard error of 0.0088. For MaxP, Pri-AwaRE exhibits the average satisfaction of 0.8301 with a standard error of 0.0105. As the average satisfaction is above the satisfaction threshold, it shows compliance with the requirements for both MinC and MaxP under S_1. RE-STORM-ARROW also shows comparable results to Pri-AwaRE for scenario S_1. It exhibits the average satisfaction level of 0.8453 for MinC with a standard error of 0.0091. For MaxP, it shows the average satisfaction of 0.8289 with a standard error of 0.0098.

The experiment results for Pri-AwaRE also show satisfaction of the threshold requirements for MaxR with the average satisfaction being above the threshold of 0.85 under most of the scenarios. For example, under scenarios S_1, S_2, S_4 and S_5, the average satisfaction levels for MaxR are 0.8727, 0.9016, 0.8555 and 0.8965 respectively. As the average

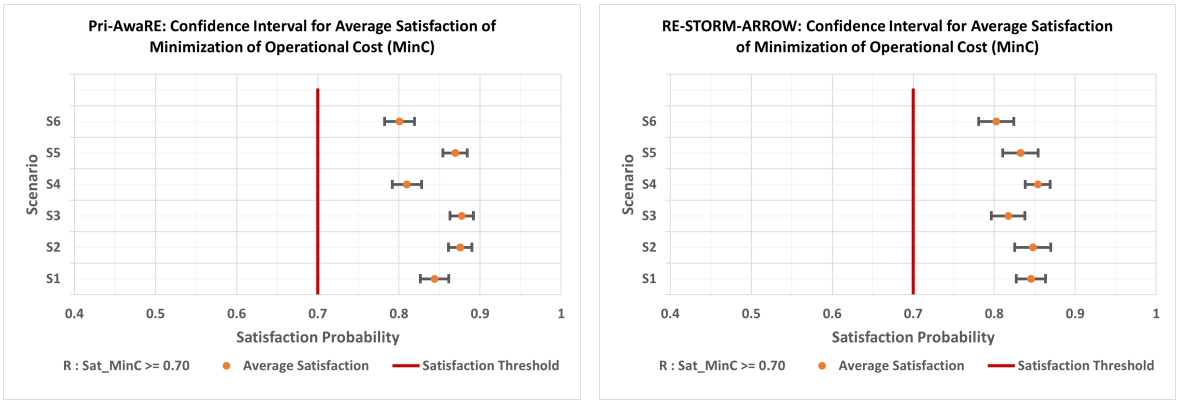


Figure B.7: Confidence Interval for Average Satisfaction of MinC under Scenarios S1 to S6

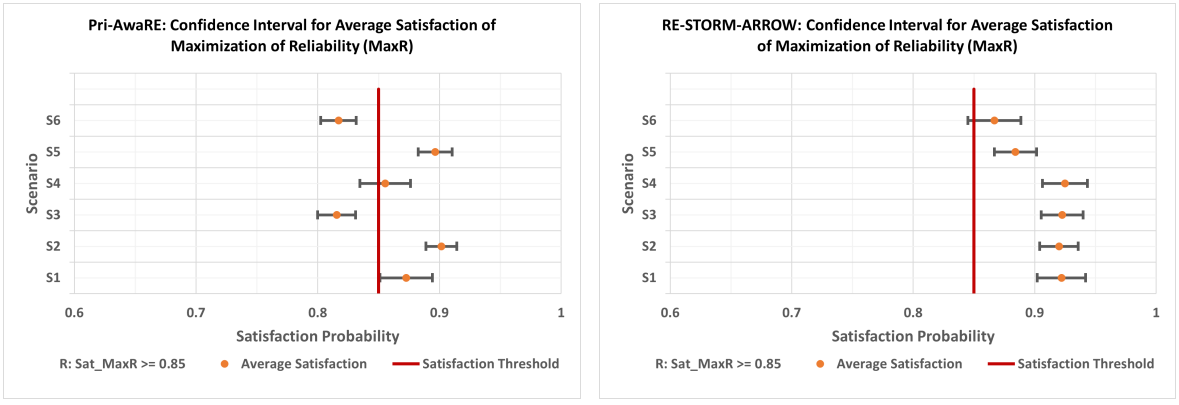


Figure B.8: Confidence Interval for Average Satisfaction of MaxR under Scenarios S1 to S6

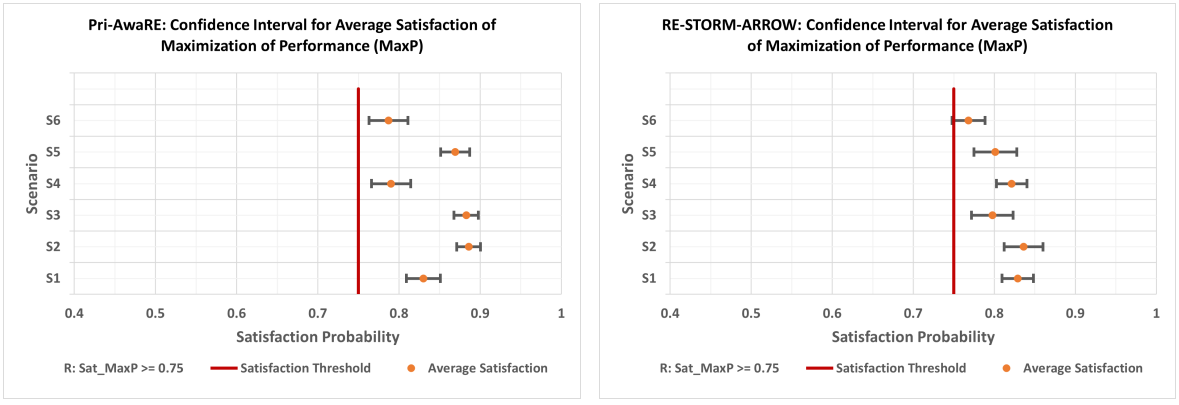


Figure B.9: Confidence Interval for Average Satisfaction of MaxP under Scenarios S1 to S6

satisfaction for MaxR is above the satisfaction threshold, it satisfies the requirement of $P(MaxR = True) >= 0.85$. For RE-STORM-ARROW, the average satisfaction for MaxR under these scenarios complies with the requirements, as shown in Fig. B.11. Furthermore, Pri-AwaRE shows comparable results to RE-STORM-ARROW concerning satisfaction of MaxR under other scenarios as well. In case of S_3 and S_6, the average satisfaction for MaxR achieved by Pri-AwaRE is 0.8155 and 0.8171 respectively. The confidence intervals exhibited by Pri-AwaRE under these scenarios are 0.7999 - 0.8311 for S_3 and 0.8025 - 0.8317 for S_6. However, the average satisfaction level for MaxR is below the threshold, it is closer to the required satisfaction level. Moreover, these satisfaction values for MaxR, achieved by Pri-AwaRE, under the dynamic scenarios of S_3 and S_6 are considered to be usual. As under such circumstances, the preference is given to support the satisfaction of MinC and MaxP more than MaxR. In case of RE-STORM-ARROW, the average satisfaction of MaxR is above the satisfaction threshold under these scenarios, as presented in Fig. B.11. To sum up, it can be concluded from the results that Pri-AwaRE offers statistically sound results in terms of satisfaction of the requirements. Furthermore, under all the scenarios, the Pri-AwaRE approach shows comparable and sometimes even better satisfaction levels for NFRs in comparison to the technique of RE-STORM-ARROW.

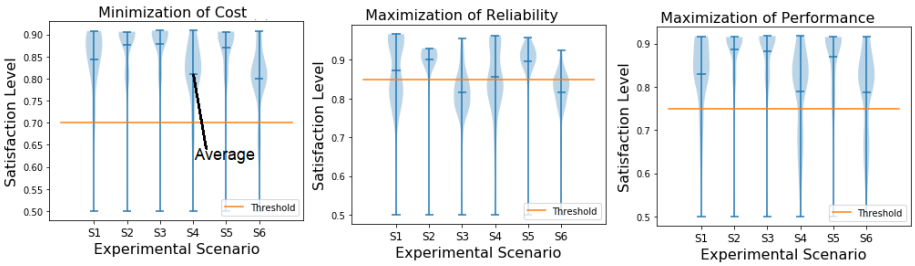


Figure B.10: Average Satisfaction of NFRs using Pri-AwaRE

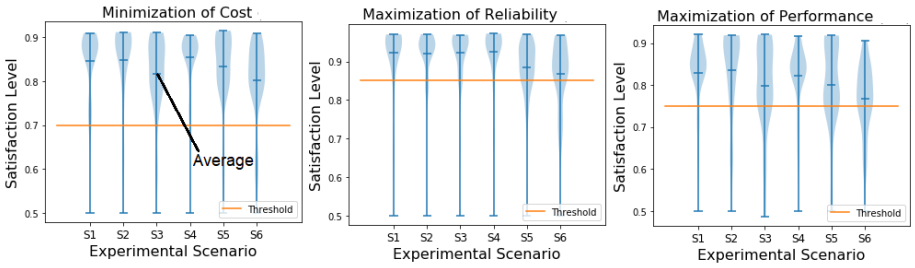


Figure B.11: Average Satisfaction of NFRs using RE-STORM-ARROW

Appendix C

Extent of Satisfaction of NFRs

This appendix presents an evaluation of the Pri-AwaRE's results for the case of RDM network using the quantitative approach of DeSiRE. The DeSiRE approach helps in evaluating the Extension of Satisfaction (*ExS*) of an NFR. The *ExS* indicates the degree of satisfaction or violation of an NFR. Figs. C.1 to C.5 present the *ExS* results for the different scenarios of RDMSim (presented in Chapter 6). Under scenarios S_2, S_3, S_5 and S_6, the *ExS* values for both MinC and MaxP are above the satisfaction boundary at almost all of the time steps.This indicates positive degrees of satisfaction. The exception lies only under scenario S_4 where there is drop in the *ExS* of MinC and MaxP at several time steps. As S_4 represents an environmental situation where a phase of consecutive and unexpected packet loss is observed during the execution of MST topology, this drop is considered as usual. However, there is a reduction in the *ExS*, yet this decline lies closer to the satisfaction boundary of zero, as shown in Fig. C.3.

Furthermore, the Pri-AwaRE also demonstrates promising results with respect to the satisfaction of MaxR under all the scenarios. The *ExS* value for MaxR is above the satisfaction boundary of zero at most of the time steps. It represents positive degrees of satisfaction for MaxR. Furthermore, the results also show a drop in the *ExS* value for MaxR. However, the *ExS* value goes below zero, this deviation in the *ExS* value ranges between -0.2 to -1.8 under scenario S_2, -0.1 to -2.2 under S_3, -0.1 to -1.4 under S_4 and -0.1 to -1.8 under S_5, as shown in Figs. C.1 to C.4. The exception lies in case of scenario S_6. Under S_6, a decline in the *ExS* value of MaxR is observed at most of the time steps. However, there is a decrease in the *ExS* value, but this drop is not large enough and ranges from -0.3 to -2.2, as presented in Fig. C.5. Moreover, all these *ExS* values are closer to the satisfaction boundary of zero. Moreover, Pri-AwaRE shows maintenance of the extent of satisfaction of

NFRs (i.e. *ExS(NFRs)*>= 0) in more than 50 percent of the simulation time steps.

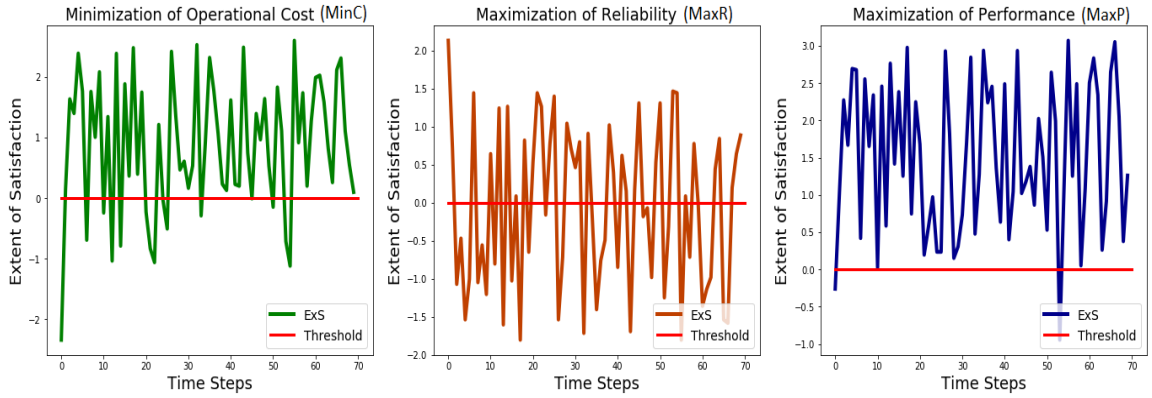


Figure C.1: RDM Case: Extent of Satisfaction (*ExS*) of NFRs over Time under Scenario S_2

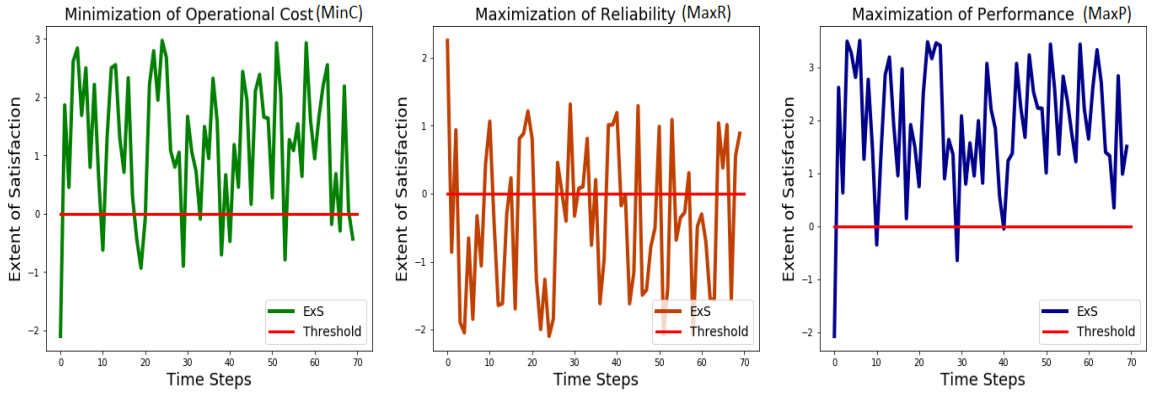


Figure C.2: RDM Case: Extent of Satisfaction (*ExS*) of NFRs over Time under Scenario S_3

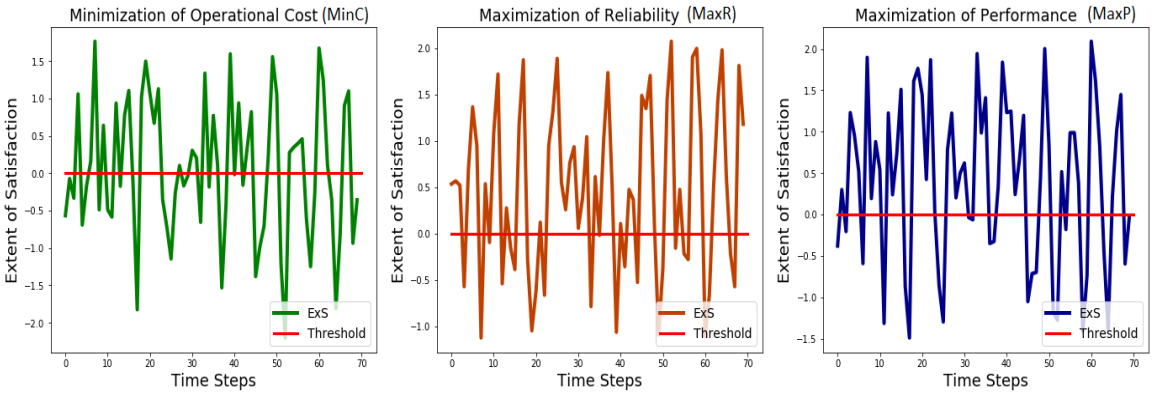


Figure C.3: RDM Case: Extent of Satisfaction (*ExS*) of NFRs over Time under Scenario S_4

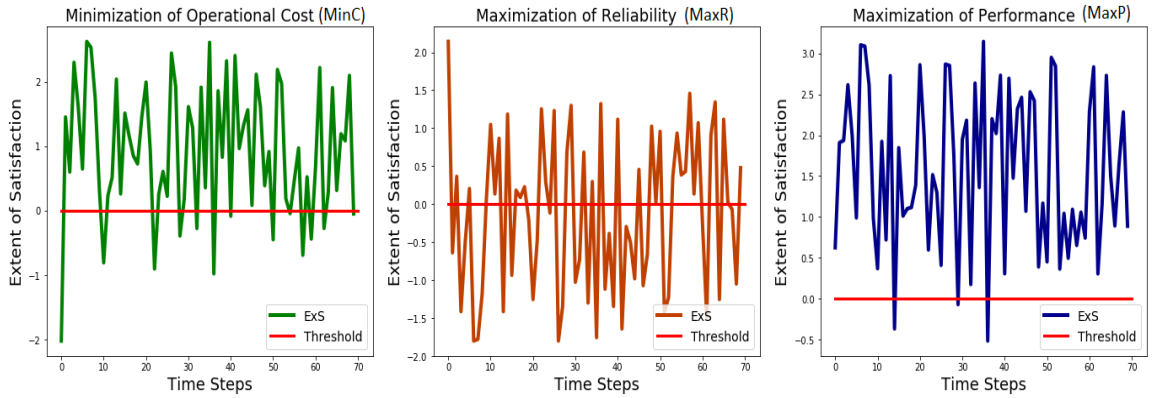


Figure C.4: RDM Case: Extent of Satisfaction (*ExS*) of NFRs over Time under Scenario S_5

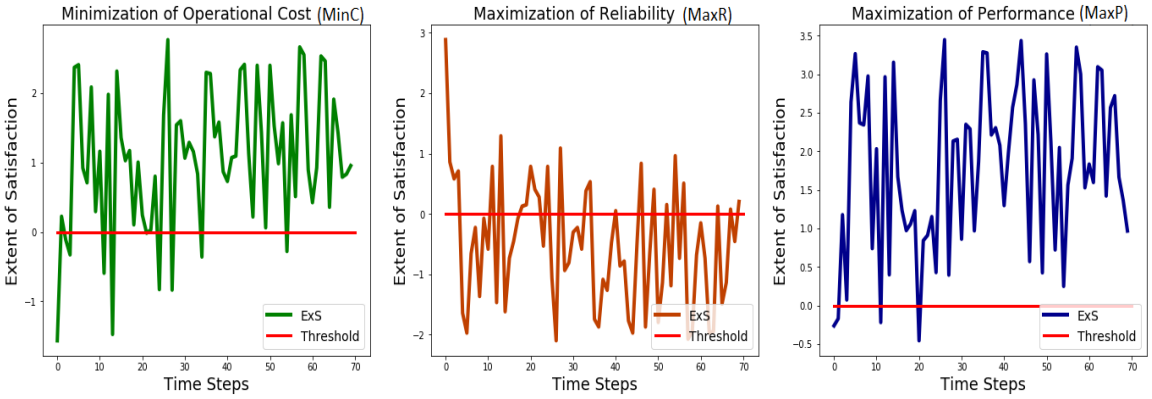


Figure C.5: RDM Case: Extent of Satisfaction (*ExS*) of NFRs over Time under Scenario S_6

Appendix D

Optimistic Linear Support

This appendix discusses the Optimistic Linear Support (OLS) algorithm. The algorithm makes use of the Multi-Reward Persues (MR-Persues) solver, a point-based technique used for solving POMDPs. Details about the MR-Persues algorithm are also described.

D.1 OLS algorithm

The Optimistic Linear Support (OLS) [129] algorithm is an approach used to solve POMDPs, and is based on Cheng's Linear Support [34]. The OLS is presented as Algorithm 1. It follows an outer loop approach by creating an outer shell around Multi-reward Persues (MR-Perseus) solver. Persues [154] is a point-based planning technique for solving POMDPs. By using an outer loop for MR-Persues solver, it generates a solution set referred as the Convex Coverage Set (CCS) X [128]. The CCS X is used to specify the collection of value-vectors $\mathbf{V}^{\pi}$ (representing the multi-objective values) and their associated policies π such that after performing scalarization a maximizing policy is in the set. To select the policy, having a maximizing value $\mathrm{V}^{*}_{X}(w)$, linear scalarization of the value vector is carried out. Linear scalarization uses parameters specified in the form of weights vector[1]. OLS helps in computation of these weights in an intelligent way at runtime. The OLS algorithm for a two objective problem, as presented in [127, 129] is shown in Fig D.1.

The OLS algorithm begins by taking an empty set of X specifying the CCS of value-vectors as presented in line 1 of Algorithm 1. The algorithm repeats steps 2 to 9 up till no improved value-vectors are found. This is estimated by the Maximal Possible Improvement Δ [127, 129]. During the first two iterations of the while loop, the algorithm chooses the

[1]For a two objective problem, the value of one of the weights can be calculated as one minus the other weight by applying linear scalarization as: $w_1 = 1 - w_2$. Where w_1 and w_2 represent the scalarization weights for the values related to objective 1 and objective 2 respectively.

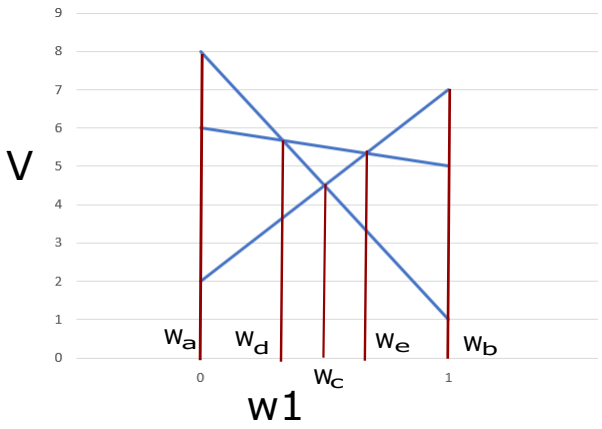


Figure D.1: Optimistic Linear Support

first two corner points known as the extrema of the weights simplex i.e. w_a= 0.0 and w_b=1.0 specified by the red vertical lines shown in Fig. D.1. Then, the value-vectors, for instance V_a= [8,1] and V_b=[2,7], for these corner points (w_a and w_b) are calculated using a MR-Persues Solver. V_a and V_b are specified by the blue lines between these corner points. Based on these value vectors, a new corner point (w_c) is discovered at their intersection. Next, maximal possible improvement Δ [129] is calculated for w_c. If Δ is improved then for this new corner point w_c, a new value vector V_c= [6,5] is computed using the MR-Persues solver as presented in Fig D.1. Based on the V_c, more corner points, such as w_d and w_e, are identified from the intersection points of the existing value vectors. Again Maximal Possible Improvement Δ for each corner point w_d and w_e is computed. The corner point (out of w_d and w_e) having high Δ is then selected, and MR-Persues solver is called again to calculate the value-vector for the selected corner point. The process is repeatedly executed until none of the remaining corner points yield an improvement in the form of Δ.

This whole process returns a CCS set X comprising of the value vectors and their corresponding policies. As more than one policy is returned, the best policy is selected with the help of a scalarization function by taking weight values computed as part of the OLS algorithm. The policy with the maximum scalarized value $V^*_X(w)$ is then chosen. The flowchart showing the step by step execution of OLS algorithm is presented in Fig D.2.

The OLSAR algorithm is an extension of OLS algorithm and follows the same steps as OLS. However, it reuses the alpha matrices from previous iterations to estimate the approximate Convex Coverage Set (CCS) of value vectors.

Algorithm 1: Optimistic Linear Support

$X \leftarrow \emptyset$
while $\neg(\Delta.isMaximum())$ **do**
    Select w ▷ Select Corner Point w
    $(V, \pi) \leftarrow MRPersuesSolver(w)$ ▷ Calculate Value-Vector V and π
    $Calculate\Delta$
    if $\Delta.isMaximum()$ **then**
        $X \leftarrow X \cup V$
    end if
end while
return Convex Coverage Set X

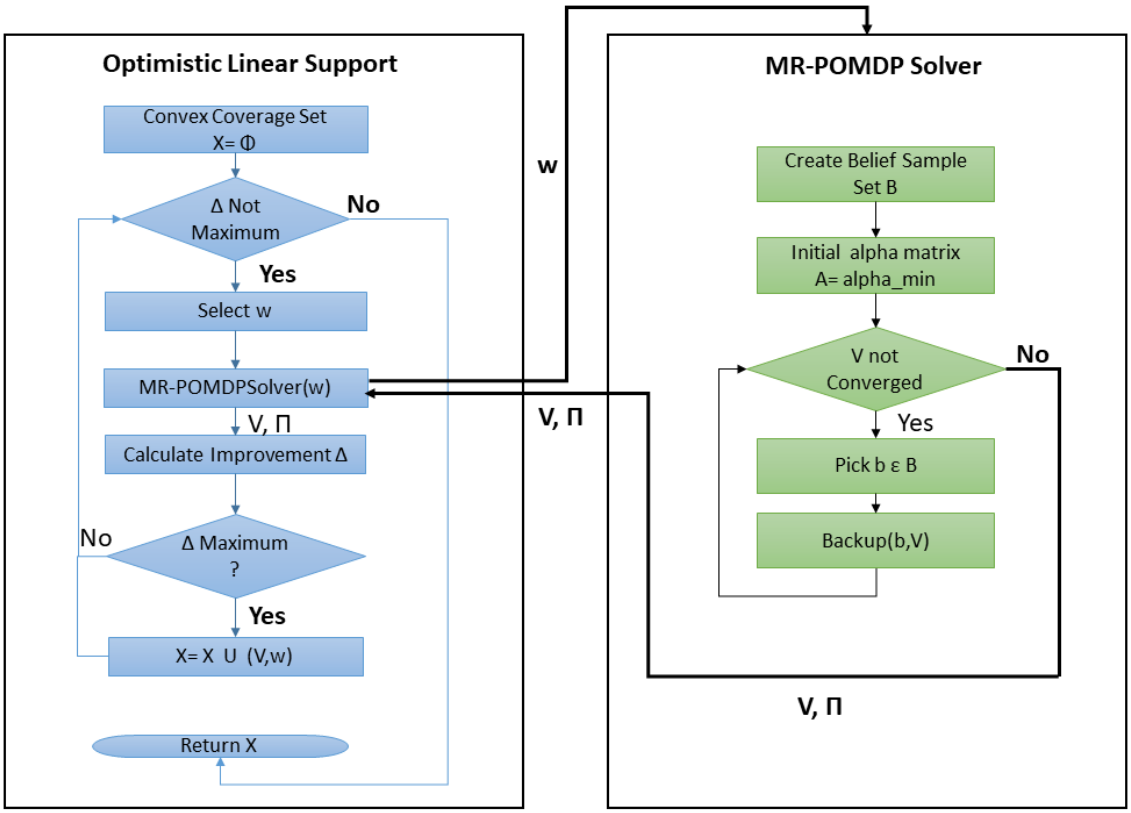


Figure D.2: Step By Step Execution of OLS

D.2 Persues POMDP Solver

Persues is a POMDP solver based on the point-based Value Iteration framework (PBVI) [154]. The algorithm is divided into two main parts, as presented in Algorithms 2 and 3, as follows:

1)Random Exploration of the belief space to gather the belief values for objectives (i.e.

NFRs in case of Pri-AwaRE).

2) Update of the Value function by applying Bellman backup [144].

In point based methods, the Value function V is specified in the form of α-vectors that are presented as follows:

$$V = \alpha = \begin{bmatrix} V(s_1) \\ V(s_2) \\ ... \\ V(s_n) \end{bmatrix} \tag{D.1}$$

and Value over belief is calculated as:

$$V_b = \alpha.b \tag{D.2}$$

Algorithm 2: Perseus: Random Exploration

```
Initialize b ← b_0
repeat
    select a ∈ A randomly
    o = performAction(a)
    b^{a,o} ← beliefUpdate(b, a, o)
    Add b^{a,o} to B
    b ← b^{a,o}
until |B| ≠ n
return B
```

Perseus algorithm begins with creation of a sample set of belief points by performing Random Exploration in the belief space. During the Random Exploration, an action is randomly chosen. The chosen action is then executed which results in an observation. For instance, based on the selected action a and observation o, a new belief sample point is calculated using the belief update procedure presented as follows:

$$b^{a,o}(s') = Pr(s'|b,a,o) = \frac{Pr(s',b,a,o)}{Pr(b,a,o)} \tag{D.3}$$

$$= \frac{Pr(o|s',b,a)Pr(s'|b,a)Pr(b,a)}{Pr(o|b,a)Pr(b,a)} \tag{D.4}$$

$$= \frac{O(a,s',o)\sum_{s\in S} Pr(s'|b,a,s)Pr(s|b,a)}{Pr(o|b,a)} \tag{D.5}$$

Algorithm 3: Perseus: Value Function Update

Initialize $V \leftarrow V_0 \leftarrow \alpha_{min}$
repeat
  $B' \leftarrow B$
  $V' \leftarrow \emptyset$

  repeat
    Randomly pick $b \in B'$
    $\alpha_{new} \leftarrow PointBasedBackup(b, V)$

    if $\alpha.b \geq V_b$ **then**
      $B' \leftarrow b \in B' : \alpha.b < V_b$
      $\alpha_b \leftarrow \alpha_{new}$

    else
      $B' \leftarrow B' - b$
      $\alpha_b \leftarrow argmax_{\alpha \in V} \alpha.b$

    end if
    $V' \leftarrow V' \cup \alpha_b$

  until $B' = \emptyset$
until V converges
return V, $B \leftarrow \emptyset$

where

$$Pr(o|b,a) = \sum_{s \in S} b(s) \sum_{s' \in S} T(s,a,s') O(a,s',o) \tag{D.6}$$

Once the belief sample set is created, the initial value function specified in the form of α-vectors is calculated as follows:

$$R_{min} = minR(s,a) \tag{D.7}$$

$$\alpha_{min} = R_{min}/1 - \gamma \tag{D.8}$$

$$V_0 = \{\alpha_{min}\} \tag{D.9}$$

After the computation of the belief sample set and the initial set of α-vectors, a point-based backup is carried out iteratively to estimate an optimal α-vector. During each iteration, a set B'=B and a new Value function V'=$\emptyset$ are generated. One belief sample point is chosen at random from B', and is used to for the computation of a new α-vector by

executing the point-based Bellman backup. For example, for k+1 iteration, the backup can be computed as follows:

$$backup(b, V) = \arg\max_{\alpha_{k+1}^{b,a}} b.\alpha_{k+1}^{b,a} \tag{D.10}$$

where

$$\alpha_{k+1} = r^a + \gamma \sum_{o \in \Omega} \arg\max_{g^{a,o}} b.g^{a,o} \tag{D.11}$$

$$g_i^{a,o} = \sum_{s' \in S} O(a, s', o) T(s, a, s') \alpha_i(s') \tag{D.12}$$

where $g^{a,o}$ represent the back-projections for all actions a and observations o of each next stage α-vector i.e $\alpha_i \in \alpha_k$.

If $\alpha_{new}.b > V(b)$, then α_{new} is added to the set V' and all other belief sample points $b' \in B'$ the value of which is improved by α_{new} i.e. $\alpha_{new}.b' > V(b')$ are removed from B'. Hence, the values of these belief points are not backed up at the current iteration. On the other hand, if $\alpha_{new}.b < V(b)$ then $\alpha_{old} = argmax_{\alpha \in V} \alpha.b$ is added to V'. The iteration ends when B'= $\emptyset$ and V is set to V'.

The procedure is repeated until V converges i.e. the difference between the V(b) from the current iteration and previous iteration is greater than the precision parameter η. More details about the implementation of the Perseus solver can be found at [162].

D.3 Multi-Reward Perseus

For the purpose of solving multi-objective decision-making problems, multi-objective reinforcement learning techniques that have a vector-valued reward function $\mathbf{R}$, instead of a scalar reward value, are used. Therefore, in such a case, point-based methods represent the value function in the form of α-matrix $\mathbf{A}$, instead of α-vector, to support multiple objectives as follows:

$$\mathbf{A} = \begin{matrix} \\ V(s1) \\ V(s2) \end{matrix} \begin{matrix} obj1 & obj2 & ... & objn \\ V(s1) & V(s1) & ... & V(s1) \\ V(s2) & V(s2) & ... & V(s2) \end{matrix} \tag{D.13}$$

Hence, in Multi-Reward Perseus (MR-Perseus) algorithm, the point-based backup re-

turns an α-matrix instead of an α-vector. It takes the set of sampled belief points and initial set of α-matrix and carries out the computation of backup for the Value vector by calculating the *vectorized* back-projections $\mathbf{G}^{a,o}$ for each next stage α-matrix $\mathbf{A}_i \in A_k$. The point-based backup for MR-Perseus is calculated as follows:

$$backup(b, \mathbf{V}) = \arg\max_{\mathbf{A}_{k+1}^{b,a}} b.\mathbf{A}_{k+1}^{b,a} \tag{D.14}$$

where

$$\mathbf{A}_{k+1} = r^a + \gamma \sum_{o \in \Omega} \arg\max_{\mathbf{G}^{a,o}} b.\mathbf{G}^{a,o} \tag{D.15}$$

$$\mathbf{G}_i^{a,o} = \sum_{s' \in S} O(a, s', o) T(s, a, s') \mathbf{A}_i(s') \tag{D.16}$$

The MR-Persues solver follows the same steps as that of the single-objective Perseus solver except that it works with α-matrices and value vectors.

Appendix E

Statement of Collaboration

Statement from Collaborating Researcher

I am aware that the work presented in this thesis builds on the ideas proposed by the POMDP based model called RE-STORM, which is a technique developed as part of my PhD research. I have collaborated with Huma Samin in the design of the experiments for the RDM case study for comparison with the Pri-AwaRE approach, for which we have published papers together. The contributions based on RE-STORM in the collaboration are explicitly referenced in the text.

Huma and I have worked together as part of the Software Engineering at Aston (SEA) Research Group and we have also worked under the supervision of Dr. Nelly Bencomo.

Dr. Luis Hernan Garcia Paucar

Teaching Associate
Aston University, UK

Appendix F

Technical Report: Evaluation of Pri-AwaRE using Non-Inferiority Trials

Abstract

The main objective of decision-making techniques for self-adaptive systems (SAS) is to support decisions for maintaining the satisfaction of non-functional requirements (NFRs) under environmental fluctuations. For this purpose, these techniques pursue the best possible trade-offs between the NFRs in order to keep the system inside the acceptability envelop of good behaviour. Understanding the nuances of these trade-offs is crucial to exploit the acceptability envelop. Therefore, the following questions are presented: given a decision-making technique, what is the margin of acceptable sacrifice inflicted on one NFR to satisfy another? How one technique compares to other techniques with respect to those margins of acceptability? In this report, an approach based on the Non-Inferiority (NI) Trial, used in clinical trials, has been applied to compare different decision-making techniques in SAS to tackle the previous questions. The evaluation has been performed for the decision-making techniques: Pri-AwaRE and RE-STORM. The results have shown that NI Trials can offer additional valuable insights about the achieved NFRs' satisfaction by decision-making techniques and their nuances, to therefore exploit the envelop of acceptability.

F.1 Introduction

Self-adaptive systems (SAS) are required to continuously adapt under changing environmental conditions [93, 139]. The main goal is to maintain the satisfaction levels of the

multiple quality properties, also known as non-functional requirements (NFRs), such as maximization of performance, maximization of reliability levels and minimization of operational costs or energy consumption for a network [86, 93, 139, 165]. As the environmental conditions change, a SAS performs self-adaptations that imply trade-offs between the different NFRs. These trade-offs involve diminishing one NFR in return for gains in other NFRs. Decision-making techniques for SAS should pursue the best possible NFR trade-offs to keep the system inside the *acceptability envelope* of good behavior. The acceptability envelope is defined as a region where the system delivers acceptable (even if imperfect) service to its users [125]. Different trade-offs can maintain the system inside the acceptability envelop. However, some trade-offs are better than others. For the purpose of optimizing the use of the envelope of acceptability according to the requirements specification, it is crucial to understand the nuances of the NFR trade-offs implied by decision-making techniques. A point to be argued is how much can the satisfaction level of one NFR be sacrificed to further satisfy another while still keeping the system inside the acceptability envelope? In other words, what is the acceptable margin that we can let go for one NFR to compensate the satisfaction of another NFR.

As other authors [4, 5], inspiration has been found from the area of medical research to go in quest of possible answers. In the area of clinical research, *new treatments* for illnesses can be proposed while a working treatment exists (called *active control*). When this happens, the *new treatment* is evaluated to demonstrate that it ***is not*** inferior to an unacceptable extent [66]. The *new treatment* is evaluated for the purpose of assessing its effectiveness (with respect to ease of use, adverse effects or cost etc.) when compared to the existing working treatment. In medicine, effectiveness (also known as efficacy) is defined as the ability of a drug to produce the desired beneficial effect [1]. Specifically, clinical experts are interested in studying possible trade-offs between the quality properties offered by the *new treatment* and those offered by the existing currently used treatment. To evaluate the similarities and differences between the two treatments the statistical test known as the Non-Inferiority (NI) Trial is used [140, 161]. While the existing treatment has already proven to be effective [6], the NI Trial is used to demonstrate that the *new treatment* is not inferior to the existing treatment (i.e. *active control*) to an unacceptable extent.

This is important because using a *new treatment* that has been shown to be not unacceptably worse than the *active control*, might offer benefits. For example, when developing

a *new treatment* to prevent tuberculosis, medical researchers might be willing to sacrifice a small amount of the benefits offered by the existing treatment in order to obtain (e.g.) simpler dosing schedules, or fewer side effects [161] if one or more of these benefits were offered by the *new treatment*.

A number of approaches exist that evaluate the trade-offs offered by the decision-making techniques, typically based on Pareto Front [67] or on Utility Functions [122, 172]. In contrast to these approaches the use of the NI Trial can offer insights about the nuances of the trade-offs of NFRs offered by self-adaptation techniques that are not offered by these approaches. These insights offers better exploitation of the acceptability envelop and decide on the best techniques, given the requirements for the NFRs' satisfaction specified [136, 138].

The report focuses on presenting two main things:
1) How the NI Trial technique can be used to evaluate the level of effectiveness of the decision-making techniques for SAS.
2) A proof of concept based on the application of the NI Trial-based approach to evaluate two decision-making techniques based on reinforcement learning: Pri-AwaRE and RE-STORM.

The report is organized as follows: Section F.2 provides a description of the NI Trial. Section F.3 presents the approach by describing the mapping for the NI Trial to the decision-making techniques for SASs. Section F.4 provides the Experimental Evaluation which is followed by Related Work and Conclusion in Sections F.5 and F.6 respectively.

F.2 Non-Inferiority Trial (NI Trial)

In the field of medicine, *new treatment*s are developed to progressively provide more effective treatment for diseases [66]. Once a new treatment has been developed the differences between it and the existing standards of care or working treatments (also known as *active control*), are explored. To be successful, the new treatment must be shown to perform better, for at least some given parameters that may be preferable under some circumstances; (e.g.) that it is a better treatment for a given sector of the population. The main idea is to demonstrate the superiority of a *new treatment* over an existing one.

However, even if the *new treatment* does not prove to be better than the existing one,

it may still offer advantages with (e.g.) respect to ease of use, adverse effects or cost that may as well contribute towards the effectiveness of the treatment. For example, blood thinning medication Warfarin was widely used in patients with irregular heart rhythms (like Atrial Fibrillation) to prevent complications such as stroke. Patients taking Warfarin require regular monitoring through blood tests for dose adjustment because there is risk of bleeding which could sometimes be life threatening. New blood thinners known as DOACs (Direct Oral Anticoagulants) are now recommended in these patients instead of Warfarin due to a number of reasons. There is a reduced risk of bleeding with DOACs. Therefore, regular monitoring through blood tests is not required. This adds to the effectiveness of DOACs in terms of its cost, safety and ease of use for the patients. However, DOACs has a limitation that it cannot be given to patients with severe kidney failure whereas Warfarin is considered safe for these patients [32, 47].

Hence, clinical experts do make a trade-off based on the quality properties offered by the *new treatment* in comparison to those offered by the *existing treatment*. In order to evaluate how similar the two treatments are in terms of their effectiveness, the statistical test known as the Non-Inferiority Trial is typically used [140, 161]. The goal is not to show that the *new treatment* is better, but to show that the *new treatment* is 'not unacceptably worse' than the existing treatment (i.e. *active control*). Hence, a non-inferiority trial, in contrast to a superiority study [66, 161], demonstrates that the *new treatment* is not inferior to the existing treatment, as presented in Fig. F.1. Using a *new treatment* (such as DOAC), that is not unacceptably worse in effectiveness than the *active control* (such as Warfarin), might be beneficial as it might cause fewer side effects and improved quality of life and dosing regime. Hence, to achieve the satisfaction of such quality properties, medical researchers might be willing to sacrifice some small amount of benefit specified as the margin d_{NI}. The main goal of the NI trial is to test whether the effect of the *new treatment* is not unacceptably worse than the effect of the *active control* by more than a pre-defined NI margin d_{NI}. The effect is represented by the difference in mean outcomes of the two groups for a particular quality property and the margin is defined as the largest acceptable difference with respect to achieving the quality property between the *new treatment* and the *active control* [6].

The steps for carrying out the NI-Trial are as follows:

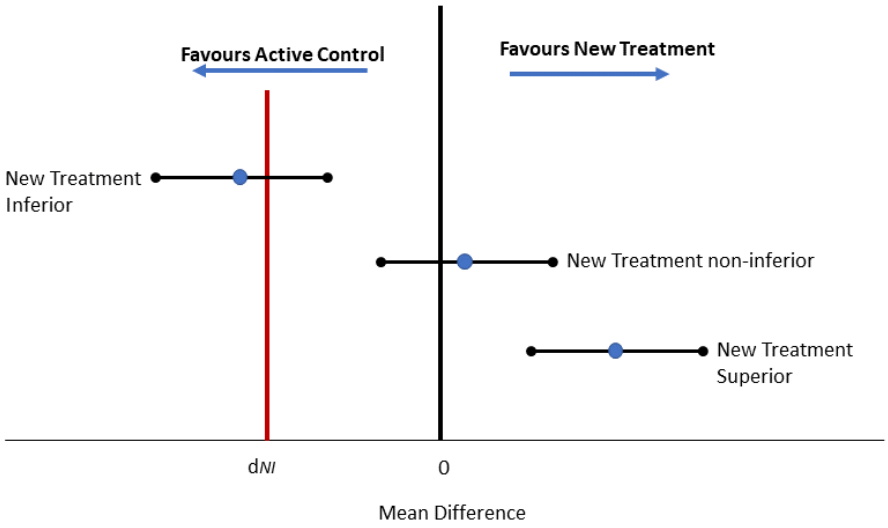


Figure F.1: Non-Inferiority Trial for Medical Treatments[140]

Step 1: Define the Null H_0 and Alternative Hypothesis H_a for the experimental evaluation: The hypothesis for the NI Trial is defined in terms of the difference of means as follows:

$\boldsymbol{H_0}$: *The Confidence Interval of the difference of means is less than or equal to the* d_{NI} *i.e.* $CI\mu_{new} - \mu_{control} <= d_{NI}$

$\boldsymbol{H_a}$: *The Confidence Interval of the difference of means is greater than the* d_{NI} *i.e.* $CI\mu_{new} - \mu_{control} > d_{NI}$

Step 2: Choose an *active control* that has proven to be performing better than *placebo*: *Placebo* is defined as a fake or dummy drug that has no therapeutic effect [39]. It is used in clinical trials to evaluate the effectiveness of new drugs or treatments.

Step 3: Choose the NI margin d_{NI}: The d_{NI} is defined in advance by the clinical experts before performing the NI Trial. The margin can be selected conceptually [140] by asking the experts what level of effectiveness they would be willing to sacrifice in return for the benefits offered by the *new treatment*. Alternatively, a number of formal approaches for setting the margin exist such as synthesis approach (also known as putative placebo) and fixed margin approach etc [161]. Out of these, the fixed margin approach is the one

recommended by the US Food and Drug Administration (FDA) [66]. Therefore, we use the fixed margin method for selection of the margin d_{NI} for our experimental evaluations. The fixed margin approach is described as follows:

Fixed Margin Approach: The fixed margin approach starts by calculating **M1**, where **M1** represents the entire effectiveness of the *active control* compared to *placebo*. This calculation makes use of meta-analytic methods with data from previous experimental studies carried out to evaluate the *active control*. For this purpose, a 95% Confidence Interval (CI) around the difference of means between the *active control* and *placebo* is calculated. The CI of the difference of means is calculated as follows:

$$CI = (\mu_{control} - \mu_{placebo}) \pm 1.96\sqrt{\frac{\sigma^2_{control}}{n_{control}} + \frac{\sigma^2_{placebo}}{n_{placebo}}} \tag{F.1}$$

Here, μ, σ and n represent the mean, standard deviation and sample size for the *placebo* and *active control* respectively. After the CI is computed, **M1** is the lower limit of this CI. Once **M1** is specified then a smaller margin, **M2**, is calculated. **M2** is specified by preserving some pre-determined fraction (50% or 75%) of the estimated effect of the *active control*. This fraction is selected based on clinical practice and expert judgement [140]. The **M2** is interpreted as the largest loss of effect (inferiority) that would be considered as clinically acceptable when comparing the *new treatment* to the *active control*. This **M2** represents the d_{NI} for the NI Trial.

Step 4: Evaluate Non-inferiority of the *new treatment*: This is done by calculation of the 95% CI for the difference of means of the *new treatment* and the *active control* as follows:

$$CI = (\mu_{new} - \mu_{control}) \pm 1.96\sqrt{\frac{\sigma^2_{new}}{n_{new}} + \frac{\sigma^2_{control}}{n_{control}}} \tag{F.2}$$

Here, μ, σ and n represent the mean, standard deviation and sample size for the *new treatment* and *active control*. The NI Trail is considered as successful if the lower limit of the 95% CI for the difference of means lies above the d_{NI}. As a result, the *new treatment* will be considered non-inferior to the *active control*, as shown in Fig. F.1. Therefore, it will reject the $\boldsymbol{H}_0$ and accept $\boldsymbol{H}_a$.

F.3 An NI Trial-based Approach to assess SAS decision-making techniques

This section presents the approach for evaluating decision-making techniques for SASs using the NI Trial. A mapping from the NI Trial in medical science to a NI Trial-based technique to evaluate decision-making in SAS is described as follows:

F.3.1 From New Treatment for Disease in Medical Science to New Decision-Making Technique for SAS

In medical science, the NI Trial is used to assess the effectiveness of the *new treatment* under investigation based on the quality attributes that it offers such as increasing survival time, minimization of costs etc. In a similar way, we can assess the effectiveness of the decision-making techniques for SAS. Using the NI trial, we can evaluate the effectiveness of the *new technique* in terms of the satisfaction of the quality properties (i.e. NFRs) for the SAS. Based on this, Definition 1 is presented as follows:

Definition 1: *In the NI Trial for decision-making techniques in SASs, the new decision-making technique is considered as the new treatment in the NI Trial for medical science.*

In the experimental evaluation provided in this report, Pri-AwaRE based on MR-POMDP is selected as the new technique.

F.3.2 From Active Control in Medical Science to State-of-the-Art Decision-Making Techniques for SAS

In the NI Trials for medical science, *active control* represents the existing treatments for the diseases that are already in use and have proven to be effective. Similarly, in the domain of decision-making for SAS, we have existing state-of-the-art techniques. These techniques have already been applied to different example applications and have proven to be effective in terms of satisfaction of NFRs for SAS. Based on this, Definition 2 is provided as follows:

Definition 2: *In the NI Trial for decision-making techniques in SASs, the state-of-the-art decision-making techniques are considered as the active control in the NI Trial for medical science.*

In the experiments presented in this report, the single-objective approach of RE-STORM is selected as the *active control*.

F.3.3 From Placebo in Medical Science to a Placebo Technique for SASs

The clinical trials in medical science are typically termed as placebo controlled trials. In this context, *placebo* is a dummy or fake drug that doesn't have any physiological benefits. However, in some cases, it is prescribed to the patients for psychological effects. In NI Trials, the *placebo* is used to test the effectiveness of the *active control*. In other words, the *active control*, that is selected, should have proven to be effective when compared to a *placebo*. Similarly, in the NI Trial for decision-making in SAS, a *placebo* technique should be selected that has some kind of effect on the adaptive system. The main challenge lies in the selection of the *placebo* for SAS. Considering the properties of a *placebo* in the controlled trials for medical research, we cannot consider the system working without an adaptive mechanism as the *placebo*. Hence, a random adaptation mechanism has been selected as a *placebo* technique because it might have some random effect on the NFRs of the SAS. Based on this, the third definition is presented as follows:

Definition 3: *In the NI Trial for decision-making techniques in SAS, a placebo technique similar to the placebo in the NI Trial for medical science will be chosen.*

In the experiments of this report, an adaptation technique that comes up with random options to decide the adaptations to be applied is selected as a *placebo*.

F.3.4 From d_{NI} to the margin of acceptability specified in SAS

In NI Trials for medical research, the d_{NI} is selected in advance by clinical experts using their expert judgement [108] As discussed before, the NI margin d_{NI} refers to the largest acceptable difference in effectiveness between the *new treatment* and the *active control*. Setting the d_{NI} in medical research can be seen as defining the requirements specifications for a SAS, which is carried out by requirements engineers [139]. In the case of a SAS, the d_{NI} can be described as the largest acceptable difference in terms of satisfaction of a particular NFR between the *new technique* and the existing state-of-the-art technique (*active control*). As the requirements engineers set the specifications for the NFR satisfaction levels, they

will also specify the d_{NI}. The latter triggers interesting future research threads discussed further in Section F.4.5. Based on the above, Definition 4 is presented as follows:

Definition 4: *In the NI Trial for decision-making techniques in SAS, d_{NI} is defined as the largest acceptable difference for the satisfaction of a particular NFR between the new technique and the existing state-of-the-art technique.*

In the experiments presented in this report, the fixed margin approach has been used for setting the d_{NI}. Let's recall, in the fixed margin approach, the **M1** represents the entire effectiveness of the *active control* compared to the *placebo*. Similarly, in the fixed margin approach for SAS, **M1** would refer to the total required NFR satisfaction achieved by the *active control* technique in comparison to the *placebo* technique. Furthermore, the expert judgement of the requirements engineers would be applied to choose the fraction of **M1** that must be preserved by the *new technique*. The d_{NI} (**M2**) will represent the remaining fraction of **M1**. The *new technique* would be considered non-inferior to the existing decision making technique, if the CI of the difference of the average (mean) satisfaction level (for a NFR) between the *new technique* and *existing technique* is greater than d_{NI}. Here, non-inferior means that the *new technique* is not unacceptably worse than the *existing technique* with respect to satisfying a particular NFR.

F.4 Experimental Evaluations

As part of the work presented in the thesis, Pri-AwaRE based on the multi-objective (multi-reward) reinforcement learning i.e. MO-RL has been applied to offer decision-making to the case studies of Remote Data Mirroring (RDM) network and Internet of Things (IoT) network. The RDM is simulated using the application RDMSim [137] and the IoT network is based on the simulating environment of DELTA-IoT [72]. The experiments for the different scenarios for both the case studies are performed. The approach of RE-STORM, based on single-objective (single-reward) reinforcement learning (SO-RL), has also been applied to offer decision-making for both the case studies. The approaches based on SO-RL present state-of-the-art techniques as they have been applied successfully to a number of example applications [22, 112, 154] in comparison to MO-RL, which is a newer technique for decision-making in SAS.

The study presented in Chapter 6 required the evaluation of the decision-making offered by both techniques (Pri-AwaRE and RE-STORM) in terms of their ability to maintain the required satisfaction levels of the NFRs. However, in this report, further details are provided to identify acceptable trade-offs in terms of the satisfaction of the NFRs with respect to assessing the acceptable loss of a given NFR against the benefits of another NFR. As Pri-AwaRE (based on MR-POMDP) offers additional benefits, in this report, the focus is on assessing the acceptability envelope of good behaviour for the SAS in terms of NFRs' satisfaction.Using the NI Trial-based technique, experimental evaluations of the decision-making offered by Pri-AwaRE (a MO-RL technique) against the approach of RE-STORM (a SO-RL technique) are provided. RE-STORM is used as the *active control*.

Next, the experimental setup and dataset for the NI Trial are described.

F.4.1 Experimental Setup

This section presents the experimental setup for the evaluation of the Pri-AwaRE using the NI Trial. The definitions, presented in Section F.3, have been used to identify the *new technique*, *active control technique* and *placebo technique*. Based on the definitions, Pri-AwaRE is selected as the *new technique*, RE-STORM is the *active control* and Random Adaptation Mechanism is selected as the *placebo*.

F.4.2 Dataset

The dataset for the NI Trial comprises results for the decision-making (presented in Chapter 6) by both the Pri-AwaRE and RE-STORM for the case studies: RDM and IoT. The NI Trial for evaluation of the Pri-AwaRE technique in terms of the satisfaction of NFRs against the RE-STORM is performed. The objective is to prove that Pri-AwaRE is not inferior to RE-STORM in terms of satisfying the NFR. The dataset for the experiments with NI Trial is composed of the satisfaction values of a particular NFR that is achieved by both techniques and *placebo* under a specific environmental scenario. The dataset format is presented in Table F.1. The dataset containing results for all the scenarios for both the case studies are available at [71]. Next, the experimental hypotheses are presented.

F.4.3 Experimental Hypothesis

This susbsection presents the hypotheses for the NI Trial:

Table F.1: Experimental DataSet Format

MinC (Pri-AwaRE)	MinC (RE-STORM)	MinC (Placebo)
2376.0	3360.0	3060.0
4050.0	3048.0	2260.0
2465.0	4160.0	4258.0
2320.0	3225.0	3460.0
1080.0	1122.0	1130.0

*MinC refers to the NFR Minimization of Operational Cost and it is specified as the total bandwidth consumed.

H_0: The Confidence Interval of the difference of mean for an NFR satisfaction is less than or equal to the d_{NI} i.e. $CI\mu_{Pri-AwaRE} - \mu_{RE-STORM} <= d_{NI}$

H_a: The Confidence Interval of the difference of mean for an NFR satisfaction is greater than the d_{NI} i.e. $CI\mu_{Pri-AwaRE} - \mu_{RE-STORM} > d_{NI}$

F.4.4 Experiments

The experimental results for the NI Trial applied to decision-making techniques for SAS are presented. The experiments have been executed for all the RDMSim [137] and DELTA-IoT [72] scenarios. The results can be found at [71]. For the purpose of description, the experiment results for two RDM scenarios Scenario 1 and Scenario 2 are provided. RDM Scenario 1 represents an environmental situation where link failures occur during the execution of MST topology that results affecting the reliability of the RDM network, and RDM Scenario 2 specifies an environmental situation where unexpected packet loss might generate an unsual rate of data forwarding during the execution of Redundant Topology. This would lead to higher bandwidth consumption and also reduce the system's performance.

Next, a description of the results for the NI Trial for the RDM scenarios as shown in Figs. F.2 and F.3 is provided.

Lets take the example of NI Trial for the case of MinC under RDM environmental Scenario 1. The main goal is to test that Pri-AwaRE is non-inferior to RE-STORM in terms of satisfying the NFR MinC. In order to carry out the NI Trial, the NI Margin d_{NI} is selected using the fixed margin approach.

The selection of the margin is described as follows: In order to select the d_{NI}, the CI for the difference of mean of RE-STORM against *placebo* for MinC satisfaction is evaluated. The mean and standard deviation for the satisfaction of MinC using RE-STORM is 2282.842

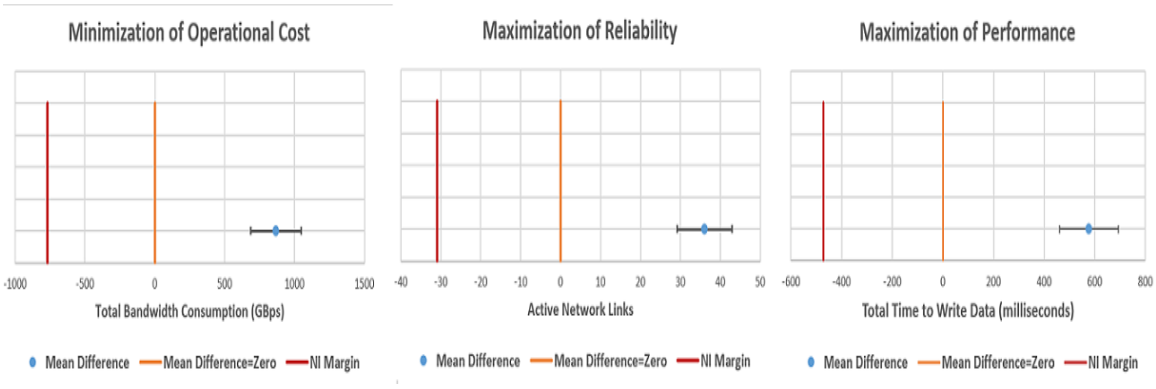


Figure F.2: RDM Case: Non-Inferiority Trial to assess the non-inferiority of Pri-AwaRE in comparison to RE-STORM under Scenario 1

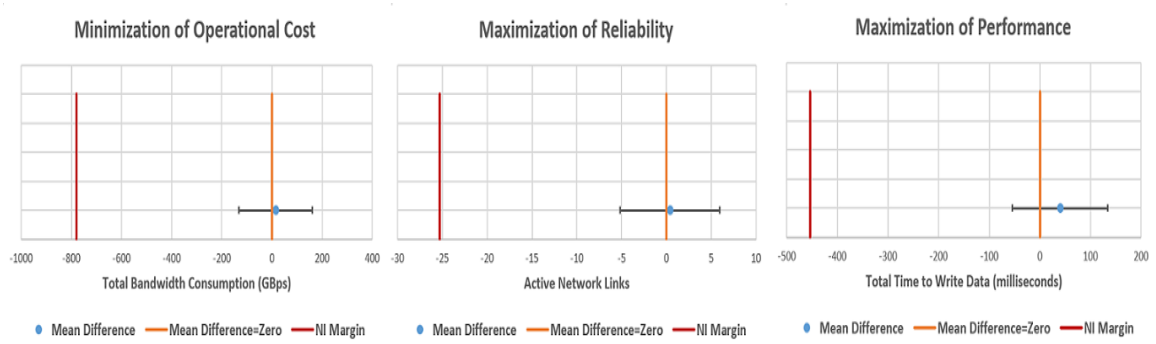


Figure F.3: RDM Case: Non-Inferiority Trial to assess the non-inferiority of Pri-AwaRE in comparison to RE-STORM under Scenario 2

GBps and 1169.866 GBps respectively. In case of *placebo*, the mean and standard deviation for MinC is 3632.87 and 1751.196 GBps respectively. The important point to note here is that the RE-STORM (i.e. *active control*) is showing better satisfaction of MinC on average than the *placebo* (Random Adaptation) under Scenario 1 as presented in Fig. F.4. After the CI for the difference of means is evaluated on the basis of the mean and standard deviation, we can select **M1** as the lower level of the CI which is equal to -1534.628. Next, the **M2**, i.e. d_{NI} is selected. Following the best practice in medical science that is to preserve 50 percent of the effectiveness of the *active control* [140], **M2** is selected as (1-0.50) x M1. As a result, we get the d_{NI} equal to -767.314 under Scenario 1. Based on this, the Pri-AwaRE is considered as non-inferior if the CI of the difference of mean for MinC satisfaction between Pri-AwaRE and RE-STORM is greater than or equal to -767.314 GBps. The d_{NI} here specifies the margin or threshold for the acceptability envelope for the satisfaction of MinC. It means if the CI of the difference of mean (for MinC satisfaction) between Pri-AwaRE

and RE-STORM is greater than the acceptable margin d_{NI} then the Pri-AwaRE would be showing the satisfaction of MinC within the acceptability envelope as it was the case and is explained next.

After computing the d_{NI}, the CI of the difference of means is computed as shown in Fig. F.2. The mean difference for Pri-AwaRE and RE-STORM is 866.874 GBps and the CI for the difference of means is 685.486 - 1048.261. As the lower bound of the CI is greater than the d_{NI}, the Pri-AwaRE is considered non-inferior to RE-STORM in terms of satisfaction of MinC. Moreover, as the CI is greater than zero, it is also considered *superior* to RE-STORM. The NI Trial for the case of MaxP under Scenario 1 shows similar results. Therefore, it proves that the Pri-AwaRE is non-inferior to RE-STORM in terms of satisfying both MaxP and MinC. The CI for the difference of means for the NFRs under RDM Scenarios 1 and 2 are shown in Table F.2. Based on the results provided, the $\mathbf{H}_0$ is rejected to accept $\mathbf{H}_a$ for the case of MinC and MaxP. Furthermore, the NI Trial for MinC and MaxP under RDM Scenario 2 also shows that Pri-AwaRE is non-inferior to RE-STORM as presented in Fig. F.3. Hence, it proves that that the Pri-AwaRE is not unacceptably worse than RE-STORM with respect to satisfying the NFRs MinC and MaxP.

Moreover, the experiment results for the NI Trial also show that Pri-AwaRE is non-inferior to RE-STORM in terms of satisfying MaxR as presented in Fig. F.2. The difference of means between Pri-AwaRE and RE-STORM for satisfying MaxR is 36.05 active links with the CI for difference of 29.204 - 42.896, as shown in Table F.2. The results show that the Pri-AwaRE is non-inferior to RE-STORM but it represents a case of *Assay Sensitivity* [140, 161]. In medical trials, having the case of ***Assay Sensitivity*** means that for some disease experiments the existing treatment might not always show benefit in comparison to a *placebo* [140]. As a result, the NI Trial would demonstrate non-inferiority of the *new treatment*. This is similar to the case of MaxR where the case of Assay Sensitivity is shown. The reason is that the RE-STORM (*i.e. active control*) shows an average satisfaction of 89.87 active links for MaxR which is lower than that of Random Adaptation Mechanism (*placebo*) which exhibits 144.65 active links on average as shown in Fig. F.4. Hence, the RE-STORM shows lower reliability level than the Random Adaptation. As the *active control* is not performing better than the *placebo* in terms of satisfaction of MaxR, the results are not considered as accurate and therefore possess Assay Sensitivity case. The NI Trial for the case of MaxR also shows similar results under Scenario 2 as shown in Fig. F.3. In the

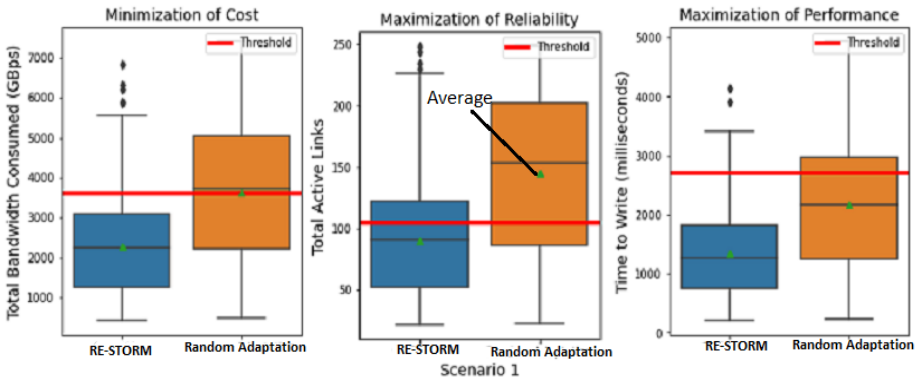


Figure F.4: Comparison of RE-STORM and Random Adaptation under RDM Scenario 1

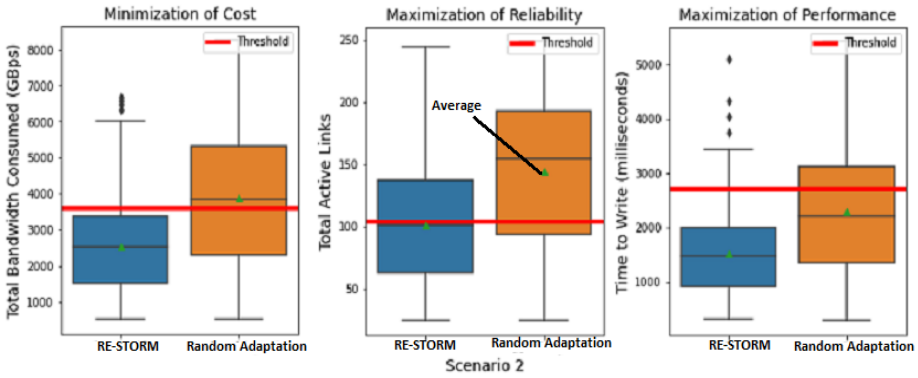


Figure F.5: Comparison of RE-STORM and Random Adaptation under RDM Scenario 2

case of the Assay Sensitivity, the results are inconclusive because the *active control* does not perform better than *placebo*, but would likely hint non-inferiority of the Pri-AwaRE [140] in some cases. Experimental evaluations for all of the RDM Scenarios show similar results and are reported in [71].

Moreover, experimental evaluations for the case of DELTA-IoT have also been performed using the NI Trial. The results for IoT case are also provided in [71]. In the DELTA-IoT case, the experiment results for the NI Trial also show that Pri-AwaRE is non-inferior to RE-STORM in terms of satisfying Minimization of Energy Consumption (MinEC) as presented in Fig. F.6. The difference of means between Pri-AwaRE and RE-STORM for satisfying MinEC is 6.016 coulombs with the CI for difference as 5.779 - 6.252, as shown in Table F.3. The results show that the Pri-AwaRE is non-inferior to RE-STORM as shown in Fig. F.6. Moreover, the experiment results for the case Minimization of Packet Loss (MinPL) represents a case of ***Assay Sensitivity*** [140, 161]. This is similar to the case of MaxR (in case of RDM) where the case of Assay Sensitivity is shown. The reason is that the RE-STORM (*i.e. active control*) shows an average satisfaction of 0.1494 (i.e. 14.94 percent)

Table F.2: RDM Case: CI for the difference of Means for NFR Satisfaction between Pri-AwaRE and RE-STORM under Scenarios 1 and 2

Scenario	NFR	Mean Difference	Confidence Interval	NI Margin
1	MinC	866.874	685.486 – 1048.261	-767.314
	MaxR	36.05	29.204 – 42.896	-30.853
	MaxP	576.87	461.27 – 692.469	-472.027
2	MinC	14.258	-132.615 – 161.132	-778.565
	MaxR	0.398	-5.168 – 5.964	-25.3014
	MaxP	39.9507	-54.227 – 134.128	-454.005

Table F.3: IoT Case: CI for the difference of Means for NFRs Satisfaction between Pri-AwaRE and RE-STORM

NFR	Mean Difference	Confidence Interval	NI Margin
MinEC	6.0159	5.7794 – 6.2523	0.23643
MinPL	-0.00756	-0.0132 – -0.0019	-0.00226

of packet loss which is higher than that of Random Adaptation Mechanism (*placebo*) which exhibits 0.1454 (14.54 percent) of packet loss on average as shown in Fig. F.7. Hence, the RE-STORM shows lower satisfaction of MinPL than the Random Adaptation. In the case of the Assay Sensitivity, the results are inconclusive because the *active control* does not perform better than *placebo*. Moreover, Pri-AwaRE shows an better satisfaction level for MinPL on average in comparison to RE-STORM (as presented in Fig. 6.2). In case of Pri-AwaRE, the average satisfaction for MinPL is 0.1418 (i.e. 14.18 percent) of total packet loss which is lower than that of RE-STORM where the average satisfaction for MinPL is 0.1494 (14.94 percent). Moreover, the results are statistically significant by having a p-value $<= 0.05$.

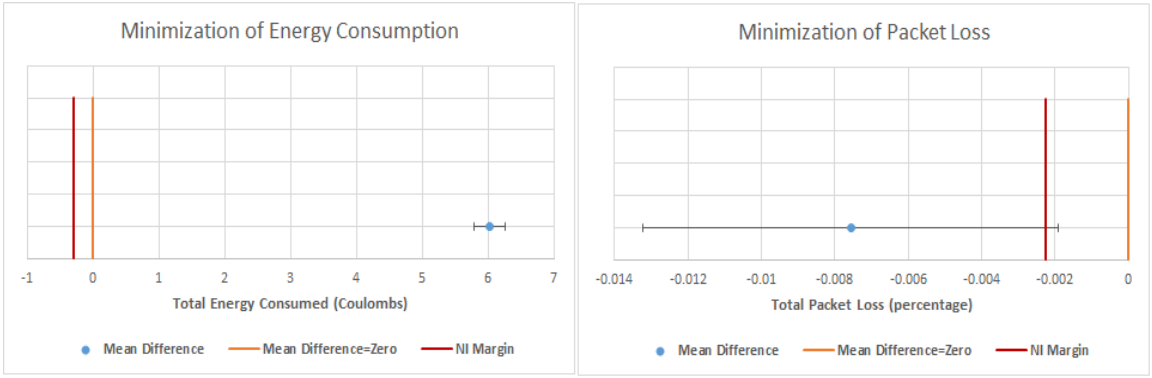


Figure F.6: IoT Case: Non-Inferiority Trial to assess the non-inferiority of Pri-AwaRE in comparison to RE-STORM

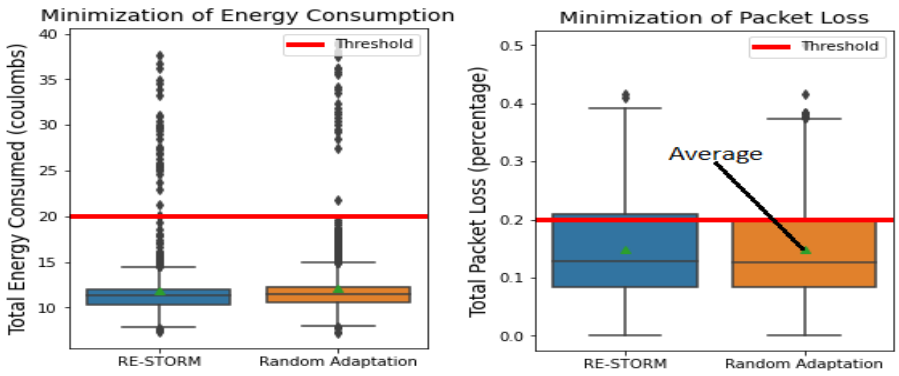


Figure F.7: IoT Case: Comparison of RE-STORM and Random Adaptation

F.4.5 Discussion and Research Outlook on Future Research

Results from the evaluations have shown that the NI Trial provides more insights about the effectiveness of the *new technique* with respect to satisfying the NFRs. Moreover, it helps us in assessing nuances of the NFRs' trade-offs offered by decision-making techniques, and to what extent they are complying to the acceptability envelope of good behavior indicated by the NI margin d_{NI}. In the experiments, the NI Trial has shown that Pri-AwaRE is non-inferior to RE-STORM. In some cases, as probably expected, it has proven to be even superior to the RE-STORM with respect to satisfying the NFRs.

In this report, an initial prototype for application of NI Trial in the domain of decision-making in SASs for better exploitation of the decision-making techniques is presented. With this initial prototype example, I have demonstrated that the NI Trial based approach can be used for evaluation of decision-making techniques to provide further insights for decision-making and satisfaction of the NFRs. To further evaluate the application of the NI approach in the domain of decision-making for SASs, I intend to perform further experiments using other decision-making techniques [26, 28, 172] in future.

Moreover, based on the results presented in the report, the work related to to the usage of NI Trials for evaluation of decision-making techniques could be extended by targeting the following main areas:

Selection of Placebo: In this report, random adaptation mechanism has been selected as the *placebo* for the experiments. However, further investigation to study the effects of the selection of the *placebo* technique is required. The plan is to further explore the different approaches and mechanisms that are followed by the medical scientists for selection of the

placebo [160] and apply those procedures in the domain of decision-making for SAS.

Different types of NI Margin selection methods: So far, the fixed margin approach has been used for the selection of the NI margin (d_{NI}). Further exploration with respect to the application of different available techniques such as synthesis method for the margin selection [66, 140] is required . Furthermore, I consider that the NI margin is relevant during the specification of the stakeholders' requirements while assessing the acceptability of the satisfaction levels of the NFRs. As such, I would like to extend the application of the NI Trial with respect to these aspects. Moreover, at the moment, the fixed margin approach has been used as a way to set a local NI margin for each dynamic scenario for the RDM system. However, it could also be the case that one global NI margin for an NFR could be applied for the SAS working under all the dynamic situations, which would depend on the requirements and context of the application and its domain.

F.5 Related Work

Efforts have been made to devise approaches to evaluate SAS and their NFRs [40, 60], with whom I share similar goals around the evaluation of SAS to pursue new findings when developing new techniques. The related work is divided into two parts as follows:

F.5.1 Evaluation of Decision-Making Techniques for SASs dealing with trade-offs of NFR

Techniques have been developed to deal with the trade-off analysis of the NFRs. The techniques are typically based on the multi-criteria-decision-making [94, 113], evolutionary computation [25, 122], probabilistic models such as Dynamic Decision Networks [15, 67] and different markov based approaches [28, 51, 112]. These techniques have been compared with the existing state-of-the-art techniques using the Pareto Front [67] or using the Utility Functions [122, 172] to evaluate the best trade-offs.

The approach presented in [44], provides a statistical approach to assess the degrees of satisfaction of the NFRs during the decision-making in SAS. Specifically, the authors presented an automated objective statistical approach, as opposed to relying on human expertise, to quantifying the extent that an NFR is violated (or satisfied). As in my case, the authors were motivated to further explore the trade-offs between NFRs in decision

making of SAS.

Different from the work presented in this report, none of the approaches cited above tackle the assessment of techniques and analysis of the trade-offs of the NFRs with respect to the acceptability envelope of good behavior.

F.5.2 Approaches using NI Trial for evaluation of AI-based techniques

A number of studies have used the NI Trial for the evaluation of AI-based techniques or intelligent software systems dealing with domains such as healthcare and diagnosis of diseases [96, 105, 109]. In these studies, the purpose of the NI Trial has been to compare the feasibility of the AI-based techniques against the diagnosis and practices of experts in health care. Moreover, in [163], the NI approach has been used to evaluate the effectiveness of setting the hyper-parameters to default values when compared to tuning of the hyper-parameters for machine learning algorithms. The study shows that setting default value for the hyper-parameters is considered non-inferior to the algorithm working with tuned hyper-parameter values, and is also considered superior in some cases. The studies presented in [42, 179] also evaluate the effectiveness of the hyper-parameters tuning using the NI Trial. Furthermore, the study described in [101] also shows comparison of a meta-heuristic optimization algorithm with the different optimization algorithms such as particle swarm optimization, differential evolution and neural networks etc. However, the authors of [42, 163, 179] and [101] show that their proposed technique is non-inferior to the existing techniques, they don't evaluate the effectiveness of their *active control* with respect to the *placebo* for the selection of the NI margin d_{NI}. Also, the studies don't use a formal technique for the selection of the d_{NI}. Moreover, the NI Trial so far has been used as a technique to offer validation of results for the approaches specified above. In contrast, in this report the NI Trial is used to evaluate the effectiveness of one decision-making technique compared to another with respect to satisfaction of the NFRs for SASs, and specifically about getting further insights from quantifying the acceptable loss of an NFR against the benefits for another NFR.

F.6 Conclusion

In this report, the NI Trial has been presented as a way to assess the decision-making techniques for SAS. A mapping of the NI Trial from medical science to the domain of

decision-making techniques for SAS is presented. As a proof of concept, experimental evaluations for comparison of two SAS decision-making techniques: the Pri-AwaRE and the RE-STORM are provided. With the initial prototype for the NI Trial, insights about the effectiveness of the *new technique* with respect to satisfying the NFRs are provided. Moreover, it has proven to be useful assessing the NFRs trade-offs offered by decision-making techniques. It also helps in studying to what extent the NFRs are complying to the acceptability envelope of good behavior indicated by the requirements and underpinned by the NI margin d_{NI}. From the results, it can be concluded that Pri-AwaRE is non-inferior to RE-STORM in terms of complying to the requirements. Moreover, the results are statistically significant (p<=0.05) in terms of complying to the requirements compared to RE-STORM.

A research outlook on future research for the application of the NI Trial to the assessment of decision-making techniques for SAS is also provided, which can be summarised as:

- Further exploration of the effects of the placebo.
- Evaluation of other decision-making techniques for SAS including [26].
- Compliance to margins/thresholds for the acceptability envelope of good behavior.