CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Evaluating sets of multi-attribute alternatives with uncertain preferences
Authors	Toffano, Federico
Publication date	2020-09
Original Citation	Toffano, F. 2020. Evaluating sets of multi-attribute alternatives with uncertain preferences. PhD Thesis, University College Cork.
Type of publication	Doctoral thesis
Rights	© 2020 Federico Toffano. - https://creativecommons.org/licenses/by-nc-nd/4.0/
Download date	2026-11-03 09:23:47
Item downloaded from	https://hdl.handle.net/10468/11277



UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Evaluating Sets of Multi-Attribute Alternatives with Uncertain Preferences

Federico Toffano
MSC COMPUTER ENGINEERING

Thesis submitted for the degree of Doctor of Philosophy



NATIONAL UNIVERSITY OF IRELAND, CORK

COLLEGE OF SCIENCE, ENGINEERING AND FOOD SCIENCE

SCHOOL OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY
INSIGHT CENTRE FOR DATA ANALYTICS

September 2020

Head of Department: Prof. Cormac J. Sreenan

Supervisors: Dr. Nic Wilson

Research supported by Research supported by Insight Centre for Data Analytics and Science Foundation Ireland under Grant No. 12/RC/2289 which is co-funded under the European Regional Development Fund.

Contents

List of Figures iv
List of Tables vi
Abstract ix
Acknowledgements xii

1 Introduction **1**
1.1 Outline and Contributions 4
1.2 Publications 6

2 Background and Related Work **8**
2.1 Multi-Attribute Utility Theory 8
2.1.1 Preferences under certainty 9
2.1.2 Pareto Dominance 10
2.1.3 Preferential independence 11
2.1.4 Additive utility function 12
2.1.5 Generalised additive independence utility model 14
2.2 Classical Methods for Preference Elicitation 14
2.2.1 Complete elicitation of the utility function 15
2.2.2 Interactive optimisation 16
2.2.3 Bayesian preference elicitation 18
2.2.4 Minimisation of an error function 18
2.3 Parameterised Preferences 19
2.3.1 ISMAUT 21
2.3.2 Minimax Regret decision criterion 23
2.3.2.1 Setwise Minimax Regret decision criterion . . . 26
2.3.2.2 Incremental elicitation based on Minimax Regret 29
2.3.2.3 Minimax Regret with linear utility function . . 32
2.4 Relations and Optimality Classes 33
2.4.1 Relations 34
2.4.2 Optimality classes 35
2.5 Conclusions 38

3 Minimality and Comparison of Sets of Multi-Attribute Vectors **40**
3.1 Introduction 41
3.2 Preference Relations for Set of Alternatives 42
3.3 Filtering A and Minimal Equivalent Subsets 48
3.3.1 Operators for set of alternatives 49
3.3.2 Filtering 56
3.3.3 $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ as unique minimal equivalent set 59
3.4 Setwise Max Regret 63
3.5 Implication for Incremental Preference Elicitation 66
3.6 EEU Method for Testing $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ and Computing $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$. 69
3.7 The Case of Multi-Attribute Utility Vectors 76
3.7.1 Linear programming for $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$, and $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ . . . 78

3.7.2 Using extreme points of epigraph to compute minimal equivalent subset . . . 78
3.8 The Structure of the Algorithms . . . 81
3.8.1 Computing minimal equivalent set . . . 82
3.8.2 Testing A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B . . . 84
3.9 Experimental Testing . . . 87
3.10 Conclusions . . . 89

4 A Multi-objective Framework based on User-Preferences 90
4.1 Introduction . . . 91
4.2 Literature review . . . 92
4.3 Problem Requirements . . . 94
4.4 Terminology and Definitions . . . 95
4.5 The Structure of the Framework . . . 96
4.5.1 The Mixed Integer Linear Programming model . . . 98
4.5.2 User-preference elicitation approach . . . 102
4.5.2.1 Max regret . . . 102
4.5.2.2 Discrepancy measure . . . 103
4.5.2.3 Query generation . . . 104
4.5.2.4 Stopping criterion . . . 106
4.6 Computational Experiments . . . 108
4.6.1 Instances structure . . . 108
4.6.2 Experimental results . . . 110
4.7 Conclusions . . . 116

5 An exact algorithm to compute the Setwise Minimax Regret 118
5.1 Introduction . . . 118
5.2 Setwise Max Regret . . . 120
5.3 An Efficient Algorithm to Compute Setwise Minimax Regret . . . 121
5.3.1 Pruning the search space using SAT . . . 122
5.3.2 Computation of setwise max regret . . . 124
5.3.3 Generating subsets of A using depth-first search . . . 125
5.3.4 Further implementation details . . . 126
5.4 Pseudocode . . . 127
5.5 Experimental Results . . . 130
5.6 Conclusions . . . 135

6 Conclusions & Future Work 137
6.1 Summary . . . 137
6.2 Possible Future Works . . . 139

A A1
A.1 Random Problem Generator . . . A1

B B3
B.1 Random Catalogue Generation . . . B3
B.2 Random Database Generator . . . B4

B.3 Lead-time and Lateness Predictor B5

List of Figures

2.1 Plot of the linear utility functions $u_w(\cdot)$ of the alternatives $\alpha = (8,5)$ (blue dotted) and $\beta = (4,7)$ (green solid) with respect to the scenarios $\mathcal{U} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$. 25

2.2 Plot of the linear utility functions $u_w(\cdot)$ for the alternatives $\alpha = (2,8)$ (blue dotted), $\beta = (8,2)$ (green dashed) and $\gamma = (6,6)$ (black solid), and $\text{Val}_{\{\alpha,\beta\}}(w) = \max(u_w(\alpha), u_w(\beta))$ (red dash-dotted) with respect to the set of scenarios $\mathcal{U} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$ 29

2.3 Utility function $u_w(\cdot)$ for each alternative in $\mathsf{A} = \mathsf{A}' \cup \mathsf{A}'' = \{(11,1), (7,5), (6,6), (10,4), (4,7)\}$, where $w \in \mathcal{U} = \{(w_1, w_2) : w_1 + w_2 = 1, w_1 \geq 0\}$. 38

3.1 utility function $u_w(\alpha) = w \cdot \alpha$ for each alternative of the sets, $\mathsf{A} = \{(10,4), (4,7)\}$, $\mathsf{B} = \{(11,2), (8,5)\}$ and $\mathsf{C} = \{(11,1), (7,5))\}$ where $w \in \mathcal{U} = \{(w_1, w_2) : w_1, w_2 \geq 0 \;\&\; w_1 + w_2 = 1\}$. 44

3.2 utility function $u_w(\alpha) = w \cdot \alpha$ for each alternative of the sets $\mathsf{A}' = \{(11,1), (7,5), (6,6)\}$ and $\mathsf{A}'' = \{(10,4), (4,7)\}$ where $w \in \mathcal{W} = \{(w_1, w_2) : w_1 + w_2 = 1 \;\&\; w_1 \in [0, \frac{2}{3}]\}$. 48

3.3 utility function $u_w(\cdot)$ for each alternative in $\mathsf{A} = \{(10,4), (4,7), (6,6), (5,5)\}$, where $w \in \mathcal{U} = \{(w_1, w_2) : w_1 + w_2 = 1\}$. The alternative $(10,4)$ and $(4,7)$ are strongly feasible answers, and $(6,6)$ is a feasible answer, and $(5,5)$ is not a feasible answer . . 68

3.4 Utility function $u_w(\cdot)$ for each alternative in $\mathsf{A} = \{(2,8), (8,2)\}$ (green solid) and $\mathsf{B} = \{(6,6)\}$ (black dashed), where $w \in \mathcal{U} = \{(w_1, w_2) : w_1 + w_2 = 1\}$. The blue area represents the epigraph $\gamma(\mathcal{W}, \mathsf{A}) = \{(w, r) : w \in \mathcal{W}, r \geq \text{Val}_{\mathsf{A}}(w)\}$ of A and the red dotted line represents $\text{Val}_{\mathsf{A}}(w)$. 75

3.5 Utility function $u_w(\cdot)$ for each alternative in $\mathsf{A}' = \{(2,8), (8,2)\}$ (green solid) and $\mathsf{A}' = \{(5,5), (3,3)\}$ (black dashed), where $w \in \mathcal{U} = \{(w_1, w_2) : w_1 + w_2 = 1\}$. The blue area is the epigraph $\gamma(\mathcal{W}, \mathsf{A}) = \{(w, r) : w \in \mathcal{W}, r \geq \text{Val}_{\mathsf{A}}(w)\}$, where $\mathsf{A} = \mathsf{A}' \cup \mathsf{A}''$, and the red dotted line represents $\text{Val}_{\mathsf{A}}(w)$. 82

4.1 Structure of the proposed framework 97

4.2 Number of experiments in which the three methods for query selection achieved the best performances with respect to number of queries and CPLEX time. 114

4.3 Average CPLEX and query computation time in seconds per iteration for the three methods for query selection. The graph shows an average of 20 instances where $|\mathrm{I}| = 10$, $|\mathrm{C}| = 20$ and $\rho = 0.5$. 115

4.4 Average CPLEX and query computation time in seconds per iteration for the three methods for query selection. The graph shows an average of 20 instances where $|\mathrm{I}| = 30$, $|\mathrm{C}| = 50$ and $\rho = 0.5$. 115

5.1 Plot of the linear utility functions $u_w(\cdot)$ for the alternatives $\alpha_1 = (4,4)$ (blue solid), $\alpha_2 = (2,10)$ (black dotted) and $\alpha_3 = (10,2)$ (green dashed) with $w \in \mathcal{W} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$. . 124
5.2 Plot of the linear utility functions $u_w(\cdot)$ for the alternatives $\alpha_1 = (4,4)$ (blue dotted), $\alpha_2 = (2,8)$ (green dashed), $\alpha_3 = (6,6)$ (red solid) and $\alpha_3 = (8,2)$ (yellow dashed-dotted) with $w \in \mathcal{W} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$. 129
5.3 Average computation time in seconds to compute $\mathit{SMMR}^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT (y-axis) over 20 repetitions varying k and p with an input set of 100 undominated alternatives and $\mathcal{W} = \mathcal{U}$. . . . 133

List of Tables

3.1 Execution times (in seconds) of methods to compute $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ (Section 3.8.1), $\mathrm{UD}_{\mathcal{W}}$ filtering, EEU (I) and LP (II) (and number of extreme points of the epigraph), with respect to *dim*($\mathcal{W}$) with $|\mathrm{A}| = 500$ and 4 user preferences. 88

3.2 Number of elements of $\mathrm{A}' = \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, $\mathrm{B}' = \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$ and $\mathrm{B}'' = \{\beta \in \mathrm{B}' : \forall \alpha \in \mathrm{A},\ \alpha \not\succcurlyeq_{\mathcal{W}} \beta\}$ with respect to *dim*($\mathcal{W}$) with $|\mathrm{A}| = |\mathrm{B}| = 500$ and 4 user preferences. 88

3.3 Execution time of methods for testing the dominance $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ (Section 3.8.2), i.e., testing the necessary and the sufficient condition (NSc) (1), $\mathrm{UD}_{\mathcal{W}}$ filtering (2) and algorithms T_{LP} 3(a), $\mathrm{T}_{\mathrm{EPU}}$ 3(b) and $\mathrm{T}_{\mathrm{EEU}}$ 3(c) for testing $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$, with respect to *dim*($\mathcal{W}$) with $|\mathrm{A}| = |\mathrm{B}| = 500$ and 4 user preferences. 89

4.1 Values computed by the three query selection criteria with respect to each possible query selected from the set $\mathcal{S} = \{\alpha = (4,1,2,1),\ \beta = (3,4,1,1), \gamma = (1,3,4,1), \theta = (3,3,1,4)\}$ with corresponding extreme points *Ext*($\mathcal{W}_\Lambda$) $= \{(1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1)\}$ 106

4.2 Mean and standard deviation of the truncated Gaussian distribution used to sample the demand of a component with respect to each category. 109

4.3 Intervals of the uniform distributions used to sample the mean cost of components with respect to each category. 109

4.4 Discount intervals per component category and quantity ordered. 110

4.5 Experimental results with $|\mathrm{I}| = 10$. Bold values represent the best result among the three methods in that row, with respect to time in seconds (for the first set of three columns), or number of queries (for the second set of three columns). 111

4.6 Experimental results with $|\mathrm{I}| = 20$. Bold values represent the best result among the three methods in that row, with respect to time in seconds (for the first set of three columns), or number of queries (for the second set of three columns). 111

4.7 Experimental results with $|\mathrm{I}| = 30$. Bold values represent the best result among the three methods in that row, with respect to time in seconds (for the first set of three columns), or number of queries (for the second set of three columns). 112

5.1 Average computation time in seconds to compute $\mathit{SMMR}^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT, LP SAT, EPI BF and LP BF over 20 repetitions varying k and p with an input set of 50 undominated alternatives and $\mathcal{W} = \mathcal{U}$. 131

5.2 Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT over 20 repetitions varying the number of user preferences Λ with $|\mathrm{UD}_{\mathcal{W}}(\mathrm{A})| = 500$, $\mathcal{W} = \mathcal{U}$, $k = 2$ and $p = 4$. $|\mathcal{W}'|$ represents the average number of user preference models used to evaluate subsets of A with SAT, and UD represents the algorithm to filter out dominated alternatives. 132

5.3 Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT over 20 repetitions varying the size of the input set $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ of undominated alternatives with $|\Lambda| = 0$ (i.e., $\mathcal{W} = \mathcal{U}$), $k = 2$, $p = 4$. $|\mathcal{W}'|$ represents the average number of user preference models used to evaluate subsets of A with SAT. . . . 133

5.4 Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT over 20 repetitions varying p with $|\mathrm{UD}_{\mathcal{W}}(\mathrm{A})| = 100$, $|\Lambda| = 0$ (so $\mathcal{W} = \mathcal{U}$) and $k = 4$. $|\mathcal{W}'|$ represents the average number of user preference models used to evaluate subsets of A with SAT. 134

5.5 Computation of $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with the databases considered in the experimental results of [VB20]. The first four columns show information regarding the input databases. The fifth and the sixth columns show the performances of filtering out dominated elements. The last two columns show the time performance our algorithm EPI SAT and the method used in [VB20] whose results are shown in Table 8. 134

B.1 Gaussian distribution parameters to sample the quantity of a component for an order with respect to component categories. . B4

I, Federico Toffano, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere.

Federico Toffano

Abstract

In a decision-making problem, there can be uncertainty regarding the user preferences concerning the available alternatives. Thus, for a decision support system, it is essential to analyse the user preferences to make personalised recommendations. In this thesis we focus on *Multiattribute Utility Theory* (MAUT) which aims to define user preference models and elicitation procedures for alternatives evaluated with a vector of a fixed number of conflicting criteria. In this context, a preference model is usually represented with a real value function over the criteria used to evaluate alternatives, and an elicitation procedure is a process of defining such value function. The most preferred alternative will then be the one that maximises the value function.

With MAUT models, it is common to represent the uncertainty of the user preferences with a parameterised value function. Each instantiation of this parameterisation then represents a user preference model compatible with the preference information collected so far. For example, a common linear value function is the weighted sum of the criteria evaluating an alternative, which is parameterised with respect to the set of weights. We focus on this type of preference models and in particular on value functions evaluating sets of alternatives rather single alternatives. These value functions can be used for example to define if a set of alternatives is preferred to another one, or which is the worst-case loss in terms of utility units of recommending a set of alternatives.

We define the concept of *setwise minimal equivalent subset* (SME) and algorithms for its computation. Briefly, SME is the subset of an input set of alternatives with equivalent value function and minimum cardinality. We generalise standard preference relations used to compare single alternatives with the purpose of comparing sets of alternatives. We provide computational procedures to compute SME and evaluate preference relations with particular focus on linear value functions.

We make extensive use of the *Minimax Regret* criterion, which is a common method to evaluate alternatives for potential questions and recommendations with uncertain value functions. It prescribes an outcome that minimises the worst-case loss with respect to all the possible parameterisation of the value function. In particular, we focus on its setwise generalisation, namely *Setwise Minimax Regret* (SMR), which is the worst-case loss of recommending a set of

alternatives. We provide a novel and efficient procedure for the computation of the SMR when supposing a linear value function.

We also present a novel incremental preference elicitation framework for a supplier selection process, where a realistic medium-size factory inspires constraints and objectives of the underlying optimization problem. This preference elicitation framework applies for generic multiattribute combinatorial problems based on a linear preference model, and it is particularly useful when the computation of the set of Pareto optimal alternatives is practically unfeasible.

To my mother and my father, with love and gratitude.

Acknowledgements

My deepest gratitude goes to my supervisor Nic Wilson, who patiently guided me throughout my whole PhD with valuable comments, encouragement, advice and critics. I feel very privileged to have had his supervision, which helped me grow both as a researcher and as a person.

I would like to thank Michele Garaffa, Steve Prestwich and Paolo Viappiani for the fruitful collaborations, which led to scientific publications and valuable content for this thesis. I would like to thank the examiners of this thesis, Jérôme Lang and Barry O'Sullivan, for their valuable feedback and other reviewers of the here presented publications. Furthermore, this thesis would not have been possible without the financial support of Science Foundation Ireland (Grant No. SFI/12/RC/2289), United Technologies Research Center, Cork, and the European project LOGISTAR.

I would like to thank also the admin office, Ann O'Brien, Eleanor O'Riordan, Linda O'Sullivan and Caitríona Walsh for all their help handling the administrative tasks, and Peter MacHale for addressing all my IT needs.

I also wish to thank my friends and colleagues Diego Carraro and Andrea Visentin. We met 12 years ago, and together we did our bachelors, masters and PhDs. They have been my best companions during these years, supporting me in both my private and my work life. A special thanks also goes to Carla Barcellos for her support and encouragement when writing my thesis, also to Stefano Ghidoni, Enrico Mattarolo, Jessica Susinni and Ivanka Walsh for their very genuine friendship throughout these years. I am also grateful to all current and previous Insight members for the many interesting discussions and all the pleasant moments of daily PhD life; these years in Cork have been full of positive experiences and amazing people.

Last but definitely not least, a grateful thought goes to my mother Edi Braido who always gave me support, advice and strength. She raised me alone but she ensured that I did not lack anything I needed, and this achievement is also a fruit of her strength and determination. Unfortunately I did not have the chance to meet my father Alessandro Toffano. However, I also wish to thank him since he has been a source of inspiration through my mother.

Chapter 1

Introduction

Multi-criteria decision-making (MCDM) (or multi-criteria decision analysis) [VNM47, Sav51, Fis70, Rai68] is a field of research which aim is to support a decision-maker (DM) in a decision-making process where the alternatives are evaluated with different and typically conflicting criteria. In this context it may not be obvious when an alternative is better than another one. For example, suppose that we have two alternatives α and β evaluated with respect to cost and quality. If α has a better cost, and β has better quality, then the best alternative depends on the DM's *preferences*. For this reason, in an MCDM context, the concept of optimal alternative is often replaced with the set of undominated (or non-dominated) alternatives with respect to the available DM's preference information. However, the set of undominated alternatives may be too complex to be presented to the DM; MCDM methods help on these situations in choosing ranking or sorting these type of alternatives.

Several approaches have been studied in the context of MCDM which can be applied in many academic disciplines, such as Finance, Statistics, Telecommunications and Economic. For an overview of MCDM methods see, e.g., the state of the art surveys book [FGE$^+$05b], or a more generic survey such as [ZTK14]. In this thesis, particular attention will be dedicated to a branch of MCDM theory called multi-attribute utility theory (MAUT) [Rai68]. MAUT involves numerical representations of the DM's preferences with respect to a set of alternatives which are evaluated with a fixed number of conflicting criteria. MAUT models assume the existence of an unknown DM's *utility function* representing the DM's preferences, where an alternative α is preferred to another alternative β if and only α has higher value according to the DM's utility function; this can then be

used for recommending appropriate alternatives for the DM.

Example 1. *Suppose that we have a set of apartments. Each apartment is evaluated with a vector of three reals values* (x_1, x_2, x_3) *representing monthly rent, apartment size and garden size. The DM's utility function* $u(x_1, x_2, x_3)$ *aggregates the criteria* (x_1, x_2, x_3) *returning a real number* $r = u(x_1, x_2, x_3) \in \mathbb{R}$ *representing a score of the corresponding apartment. The most preferred apartment will then be the one with highest score with respect to the DM's utility function.*

The process of learning the DM's utility function is called *preference elicitation*. A classical preference elicitation method is to precisely define the DM's utility function with elaborated interview techniques [Rai68]. However, experiments with real users have shown that this process can be a difficult and error-prone task [Sim55, TK74, PFT03]. From the 1980s, artificial intelligence has been widely applied in MAUT contexts to develop more robust preference elicitation systems. A common approach in modern MAUT preference models is to consider parameterised utility functions where the parameterisation represents the uncertainty with respect to the user preferences (see, e.g., [WSD84]). Each feasible parameter of a such parametrisation corresponds then to a specific preference model, and two different parameters could then lead to two different most preferred alternatives.

The aim of the work in this thesis relates to the development of methods for supporting decision-makers, based on parameterised preference models. These methods include:

- Reduction of the alternatives of a decision problem without reducing the maximum utility achievable.
- Preference elicitation approaches to reduce the uncertainty regarding the DM's preferences.
- Computational procedures to evaluate sets of alternatives with uncertain preferences.

In particular, suppose that $u_w(\alpha)$ is a real utility function over alternatives α where the parameter w defines a possible DM's preference model. The value of a set A of alternatives supposing a preference model w can be defined as $\max_{\alpha \in \mathrm{A}}(u_w(\alpha))$. Given two sets of alternatives A and B, we will consider the following related questions:

1. Are there elements of A that can be eliminated unproblematically? In

particular, is there a strict subset A′ of A that is equivalent to A for every consistent preference model w?

2. Given a choice between one situation, in which the available alternatives are A, and another situation, in which alternatives B are available, is A at least as good as B for every consistent preference model w?

3. Given a linear utility function $u_w(\alpha)$, how can we generate an efficient query set A for an interactive preference elicitation process in a complex combinatorial problem?

4. Given a linear utility function $u_w(\alpha)$ and a list of alternatives, how can we efficiently compute an optimal recommendation set A?

We will provide methods addressing the above questions.

Although some of the theoretical contributions apply to generic preference models, we will focus mainly on parameterised preference models where reducing the uncertainty of the utility function means reducing the set of parameters. In particular, we will consider a weighted sum of the criteria evaluating the alternatives as a preference model. In this case, the set of weights vectors can be the parameterisation of the utility function, and each weights vector defines then a different preference model. For example, a preference elicitation procedure for this preference model could aim to incrementally reduce the set of possible weights vectors until we get a good enough approximation of the DM's preferences according to a predefined stopping criterion.

To evaluate alternatives with uncertain DM's preferences, we will consider methods based on the max regret criterion [Sav72, KY13]. The max regret is the worst-case utility loss of an alternative with respect to all the feasible parameters of the DM's utility function. In other words, the max regret is a real value evaluating the maximum loss of an alternative in terms of utility units with respect to a set of possible preference models. The max regret can then be used to rank a set of alternatives in case of uncertainty regarding the DM's preferences. In particular, we will consider its setwise generalisation, i.e., the setwise max regret criterion, which is used to evaluate the worst-case loss of a set of alternatives. In this case then the setwise max regret can be used to rank different sets of alternatives when dealing with uncertain preferences. We will provide algorithms for the computation of the setwise max regret where alternatives are evaluated with linear utility functions. In particular, we will

present a novel method to compute the setwise minimax regret, which is the minimum setwise max regret among a set of sets of alternatives with a specific cardinality. The latter is a very important measure since it can be used to compute an optimal recommendation set, but also to compute a myopically optimal query set for elicitation purposes [VB09, VB20].

1.1 Outline and Contributions

This thesis is organised as follows.

Chapter 2: Background and Related Work This chapter gives an overview of MAUT models defining standard concepts such as weak orders, multi-attribute utility functions, Pareto dominance and preferential independence. We discuss some related works on common preference elicitation with particular attention to the minimax criterion which is used to evaluate alternatives with uncertain preferences. We also define preference relations representing parameterised utility functions and standard optimality classes for sets of alternatives.

Chapter 3: Minimality and Comparison of Sets of Multi-Attribute Vectors The main purpose of this chapter is to define methods to exclude alternatives from a decision problem without reducing the maximum utility achievable in all DM's preference scenarios. In particular, we define different dominance and equivalence relations for sets of alternatives including properties and evaluation procedures. We define the concept of *minimal equivalent subset* of a set of alternatives. We show the connection between the minimal equivalent subset and standard optimality classes of alternatives, and in particular, we show that for a large variety of utility functions the minimal equivalent subset is unique and corresponds to the set of possibly strictly optimal alternatives. We also define procedures to compute the minimal equivalent subset and to evaluate dominance and equivalence relations for preference models based on linear utility functions. These algorithms are based on linear programming approaches and on methods which make use of the epigraph of the utility function. The latter is a novel approach which can also be used to compute the setwise max regret of a set of alternatives, and it seems to outperform standard methods based on linear programming when up to six criteria used to evaluate alternatives.

The main contributions of this chapter are:

- A generalisation of standard preference relations to evaluate dominance and equivalence with uncertain preference models of sets of alternatives rather than single alternatives. This allows simplification of the decision space, by showing some parts are irrelevant.
- The concept of setwise minimal equivalent subset, and relative conditions of uniqueness. This allows the set of alternatives to be reduced as far as possible, without any loss of utility.
- Novel computational approaches to filter and evaluate dominance and setwise regret for sets of alternatives when supposing a weighted sum utility function as a preference model.

Chapter 4: A Multi-objective Supplier Selection Framework based on User Preferences Here we present a novel incremental preference elicitation framework for a supplier selection process to satisfy the demand for a set of products. Although it is designed for a specific problem, the underlying structure of our framework is suitable for preference elicitation in complex combinatorial problems. We suppose a linear utility function with four objectives: minimisation of cost, lateness and lead time, and maximisation of the suppliers' reputation. The framework iteratively interacts with the DM asking to compare two alternatives consistent with the DM's preference collected so far, and the interaction terminates when a stopping criterion is satisfied. We define two query strategies based on a novel measure that we call discrepancy which is strictly related to the max regret criterion, and we compare them with myopically optimal queries. As shown in our experimental results, our novel query strategies have the benefit of being simple to be computed whilst keeping a good value of information.

The main contributions of this chapter are:

- A framework to support a DM in a supplier selection process to satisfy the demand for a set of products.
- A preference elicitation procedure for problems where alternatives represented with complex combinatorial problems when supposing a weighted sum utility function as a preference model.
- Fast query selection strategies based on the max regret criterion.

Chapter 5: An exact algorithm to compute the Setwise Minimax Regret This chapter introduces a novel efficient algorithm to compute the setwise minimax regret, i.e., the minimum setwsie max regret of subsets with a specific cardinality of an input list of alternatives. The setwise minimax regret can be used to compute optimal recommendation set and myopically optimal query set. The algorithm makes use of a SAT solver to evaluate the setwise regret of several sets simultaneously, and applies to linear additive utility function. Our experimental results show that this algorithm is much faster than the current state of the art.

The main contribution of this chapter are:

- A novel efficient algorithm to compute the setwise minimax regret when supposing a weighted sum utility function as a preference model. This may allow the use of the setwise minimax regret criterion in real-time applications.

Chapter 6: Conclusions In the conclusion, we summarise the work presented in this thesis highlighting significant results and possible future works.

1.2 Publications

Chapters 2 and 4 are based on the following published papers which have been subject to peer review.

1. Federico Toffano, and Nic Wilson. Minimality and Comparison of Sets of Multi-Attribute Vectors. In *Proc. European Conference on Artificial Intelligence (ECAI) 2020*, 2020.

2. Federico Toffano, Paolo Viappiani, and Nic Wilson. Efficient Exact Computation of Setwise Minimax Regret for Interactive Preference Elicitation. In *Proc. Autonomous Agents and Multi-Agent Systems (AAMAS) 2021*, 2021.

The following is a paper based on Chapter 4 which is currently being evaluated by an editor for the journal *Annals of Operational Research*.

1. Federico Toffano, Michele Garraffa, Yiqing Lin, Steven Prestwich, Helmut Simonis, and Nic Wilson. A Multi-objective Supplier Selection Framework based on User-Preferences.

The following are further papers published during my PhD which are not related to this thesis.

1. Federico Toffano, and Nic Wilson. Balancing schedules using maximum leximin. In *Proc. European Conference on Symbolic and Quantitative Approaches with Uncertainty (ECSQARU) 2019*, Pages 492-503, 2019.
2. Steven D Prestwich, Federico Toffano, and Nic Wilson. A probabilistic programming language for influence diagrams. In *Proc. International Conference on Scalable Uncertainty Management (SUM) 2017*, Pages 252-265, 2017.

Chapter 2

Background and Related Work

In this chapter, we provide the background material for the thesis, including an overview of preference elicitation methods. In Section 2.1 we introduce the main concepts of multi-attribute utility theory (MAUT) and user preferences which are the basis of this thesis. In Section 2.2 we show a connection between MAUT and classic recommender systems, and we discuss some classical methods for preference elicitation such as complete elicitation of the utility function, and interactive optimisation including Bayesian methods. In Section 2.3 we discuss parameterised utility functions. We present a classical work on imprecisely specified multi-attribute utility theory (ISMAUT) which is one of the first attempts to deal with uncertain preference information in a MAUT context. We also define the concept and properties of the minimax regret criterion, including its setwise generalisation, which will be considered in all the chapters of this thesis. In particular, we will show how it can be used within an incremental elicitation process and how to compute it with linear utility functions. In Section 2.4 we define relations and optimality classes concerning uncertain utility functions which will be extended in the next chapter to evaluate sets of alternatives. Section 2.5 concludes the chapter.

2.1 Multi-Attribute Utility Theory

Let $\alpha \in \Omega$ be an *alternative* for the decision-maker (DM) and let Ω be a (possibly infinite) set of alternatives. In a MAUT model, alternatives are evaluated with p *evaluators*, or *criteria*, $X_i : \Omega \to \mathbb{R}$ for all $i \in [1, p]$. Each alternative α is then associated with a vector $X(\alpha) = (X_1(\alpha), \ldots, X_p(\alpha))$ which is called the *outcome* of α. Thus, we can think of the p evaluators as a mapping $X : \Omega \to \mathbb{R}^p$. We

define $\mathcal{M}$ to be the set of finite non-empty subsets of Ω.

Example 2. *Let $A \in \mathcal{M}$ be a list of apartments for rent. Suppose, for example, that the criteria used to evaluate each apartment are monthly rent in euro (X_1), size in squared metres (X_2) and distance from the city center in kilometres (X_3). The outcome of an apartment $\alpha \in \Omega$ could then be for example $X(\alpha) = (1400, 60, 2)$.*

In a MAUT context, the goal of the DM is to select her[1] preferred alternative α among a set of possible alternatives Ω. The DM is associated with a utility function defining the DM's trade-offs among the conflicting criteria used to evaluate an alternative. The preferences of the DM can then be represented by a utility function $u : \Omega \to \mathbb{R}$ which measures the value of outcomes of alternatives and can be used to find an alternative $\alpha \in \Omega$ that leads to the best outcome with respect to the DM's preferences. Note that the precise function of the DM is usually unknown, and one of the main purposes of MAUT methods is to define it or approximate it with a preference elicitation method. Such procedures typically consider a predefined *parameterised* structure of the DM's utility function, where the parameterisation reflects the uncertainty of the system concerning the DM's preferences. The goal of preference elicitation procedures is then to reduce the uncertainty of the DM's utility function.

2.1.1 Preferences under certainty

Let α and β be two alternatives in Ω. The notation $\alpha \succcurlyeq \beta$ means that a DM *weakly* prefers α to β, i.e., α is at least as good as β. The DM's preference relation $\succcurlyeq$ is assumed to be a *total preorder* or (*weak order*) which is defined by the following two rationality properties:

- Completeness: $\forall \alpha, \beta \in \Omega$, $\alpha \succcurlyeq \beta \vee \beta \succcurlyeq \alpha$. This means that a DM can always say whether or not she prefers one choice to another.
- Transitivity: $\forall \alpha, \beta, \gamma \in \Omega$, if $\alpha \succcurlyeq \beta$ and $\beta \succcurlyeq \gamma$, then $\alpha \succcurlyeq \gamma$. This, in conjunction with completeness, implies that a DM can always order choices from best to worst, allowing ties.

Weak preference can be seen also as a union of two relations:

- Indifference relation: $\alpha \equiv \beta \iff (\alpha \succcurlyeq \beta \wedge \beta \succcurlyeq \alpha)$, which reads "the DM is indifferent between α and β". $\equiv$ is an equivalence relation (i.e., it is reflexive, symmetric, and transitive).

[1] We will be using feminine pronouns for the decision-maker.

- Strict preference relation: $\alpha \succ \beta \iff (\alpha \succcurlyeq \beta \wedge \beta \not\succcurlyeq \alpha)$, which reads "the DM strictly prefers α to β". $\succ$ is a strict order (i.e., it is acyclic and transitive).

For A $\in \mathcal{M}$, a *utility-representation theorem* [Deb59a] gives necessary and sufficient conditions under which a weak preference relation $\succcurlyeq$ can be represented with a utility function, i.e., such that for any $\alpha, \beta \in$ A:

$$u(\alpha) \geq u(\beta) \iff \alpha \succcurlyeq \beta. \tag{2.1}$$

A utility function defining a weak preference relation with Equation 2.1 is called an *ordinal utility function*. For a given A $\in \mathcal{M}$, ordinal utility functions are equivalent up to a monotonic transformation. More precisely, a weak preference relation $\succcurlyeq$ can be represented with two ordinal utility functions $u :$ A $\to \mathbb{R}$ and $u' :$ A $\to \mathbb{R}$ such that for any $\alpha, \beta \in$ A:

$$\begin{aligned} &u(\alpha) \geq u(\beta) \iff \alpha \succcurlyeq \beta \\ &\text{and} \\ &u'(\alpha) \geq u'(\beta) \iff \alpha \succcurlyeq \beta \end{aligned} \tag{2.2}$$

if and only if $u(\cdot) = h(u'(\cdot))$, where $h(\cdot)$ is a monotonic increasing function (see e.g. [Fis70] or [BC19] for more details and proofs). Thus, ordinal utility functions define a ranking over the outcome space but they do not contain information on how much better an alternative is with respect to another.

2.1.2 Pareto Dominance

If an alternative α is at least as good in every criteria and strictly better in at least one criterion with respect to another alternative β, then we say that α *Pareto dominates* β. Pareto dominance is an important concept since one can discard Pareto-dominated alternatives from a recommendation process without reducing the quality of a recommendation. Formally, let α and β be two alternatives with outcomes $x = (x_1, \ldots, x_p)$ and $y = (y_1, \ldots, y_p)$ respectively, where $x_i = X_i(\alpha)$ and $y_i = X_i(\beta)$ $\forall i \in \{1, \ldots, p\}$. We say that α Pareto dominates β if and only if

$$x_i \geq y_i \qquad \text{for all } i \tag{2.3}$$

and

$$x_i > y_i \qquad \text{for some } i. \tag{2.4}$$

The preference relation $\succ_P$ induced by the Pareto dominance is then defined as $\alpha \succ_P \beta$ if and only if α Pareto dominates β. An alternative $\alpha \in \Omega$ is said to be Pareto optimal if and only if there does not exist an alternative $\beta \in \Omega$ such that $\beta \succ_P \alpha$.

The *Pareto frontier* (also known as *efficient set* or *admissible set*) is the set of Pareto optimal alternatives, i.e., it is composed of all the alternatives that are not Pareto dominated. Thus, an alternative can be part of the Pareto frontier even if it does not dominate other alternatives.

Example 3. *consider three apartments α, β and γ with outcomes $x = (1400, 60, 2)$, $y = (1200, 60, 2)$ and $z = (1400, 70, 2)$ respectively, where the criteria represent monthly rent (the smaller the value, the better), size and number of rooms (the higher the value, the better). α can be ignored since it has the same size and number of bedrooms, but higher monthly rent than β, i.e. α is Pareto dominated by β. On the other hand, β and γ are both Pareto optimal. Thus, in this example, the choice set of the DM can be reduced to β and γ since β has lower monthly rent and γ has a larger size.*

A utility function $u : \mathbb{R}^p \rightarrow \mathbb{R}$ is said to be *monotonic* if and only if for all $\alpha, \beta \in \Omega$:

$$\alpha \succ_P \beta \implies u(\alpha) \geq u(\beta). \tag{2.5}$$

In the context of preference elicitation, imposing monotonicity means ensuring Pareto-optimality.

2.1.3 Preferential independence

Let $I \subseteq \{1, \ldots, p\}$ be an index set. we define X_I to be a mapping from a set of alternatives A $\in \mathcal{M}$ to a sub-outcome state space $\mathbb{R}^{|I|}$ restricted to criteria indexed by I. Let $I^C = \{1, \ldots, p\} \setminus I$ be the complement of I. Thus, we can partition an outcome $x \in \mathbb{R}^p$ as $x = (x_I, x_{I^C})$, where $x_I \in \mathbb{R}^{|I|}$ and $x_{I^C} \in \mathbb{R}^{|I^C|}$.

We say that $y_I \in \mathbb{R}^{|I|}$ is *conditionally preferred or indifferent to* $z_I \in \mathbb{R}^{|I|}$ *given*

$x_{I^C} \in \mathbb{R}^{|I^C|}$ if and only if

$$(y_I, x_{I^C}) \succcurlyeq (z_I, x_{I^C}). \tag{2.6}$$

We say that criteria indexed by I are *preferentially independent* of the remaining criteria indexed by I^C if and only if the conditionally preference relation between any pair of suboutcome $y_I, z_I \in \mathbb{R}^{|I|}$ given $x_{I^C} \in \mathbb{R}^{|I^C|}$ does not depend on x_{I^C}, i.e., for some $x_{I^C} \in \mathbb{R}^{|I^C|}$

$$\begin{aligned}&(y_I, x_{I^C}) \succcurlyeq (z_I, x_{I^C}) \iff (y_I, y_{I^C}) \succcurlyeq (z_I, y_{I^C})\\ &\text{for all } y_I, z_I \in \mathbb{R}^{|I|} \text{ and } y_{I^C} \in \mathbb{R}^{|I^C|}.\end{aligned} \tag{2.7}$$

When preferential independence does hold, then we know that efforts made to elicit the DM's preferences of criteria indexed by I, when fixing the remaining criteria indexed by I^C, does not have to be repeated for other values of the criteria indexed by I^C.

If for each possible index set $I \subseteq \{1, \ldots, p\}$ the criteria indexed by I are *preferentially independent* with respect to the remaining criteria indexed by I^C, then we say that the criteria $X_1, \ldots, X_p$ are *mutually preferentially independent*. Note that the concept of preferential independence then has nothing to do with stochastic (statistical) independence; in fact, in this case, the independence refers to the relation between criteria used to evaluate an alternative.

2.1.4 Additive utility function

The criteria $X_1, \ldots, X_p$ used to evaluate an alternative $\alpha \in \Omega$ are mutually preferentially independent if and only if there exists an additive ordinal utility function such that:

$$u(\alpha) = \sum_{i=1}^{p} u_i(X_i(\alpha)) \tag{2.8}$$

where each u_i is a single-criterion *subutility functions* defined for criterion X_i [Deb59b, Fis65, Fis70]. *Additive independence* is often assumed in multi-criteria optimisation problems because of its simple structure.

Weighted sum utility function: The weighted sum is one of the most common models used to aggregate the multi-criteria outcomes of alternatives. It has been widely adopted in a preference elicitation context such as [Sar78, KP84, AP97], and also in recent works such as [KS14, KVVA17, BL19]. With this utility function the DM's preferences are expressed as a vector of weights $w = (w_1, \dots, w_p)$ such that $\sum_{i=1}^{p} w_i = 1$ and $w_i \geq 0$ for all $i \in \{1, \dots p\}$. Each weight w_i quantifies a value related to the importance for the DM of the criterion x_i used to evaluate an alternative. More precisely, the value of an alternative $\alpha \in \Omega$ with outcome $X(\alpha) = (X_1(\alpha), \dots, X_p(\alpha)) \in \mathbb{R}^p$ is defined as follows:

$$u(\alpha) = w \cdot X(\alpha) = \sum_{i=1}^{p} w_i X_i(\alpha) \tag{2.9}$$

where $\cdot$ is the dot product. Thus, by definition, the preference relation $\succcurlyeq_w$ induced by the weighted sum utility function with respect to the parameter w is defined as:

$$\alpha \succcurlyeq_w \beta \iff w \cdot X(\alpha) \geq w \cdot X(\beta). \tag{2.10}$$

Note that often the weights are wrongly interpreted as the importance that DM gives to each criterion. For example, consider a utility function $u_w(\alpha) = w_1 \cdot cost_e(\alpha) + w_2 \cdot time_s(\alpha)$, where $cost_e(\alpha)$ evaluate the cost of α in euro and $time_s(\alpha)$ evaluate the time of α in seconds. In this case, it is easy to see that in general the meaning of the weights vector $w = [w_1 = 0.5, w_2 = 0.5]$ does not correspond to an equal importance of the two objectives because of the different unit of measure. In fact, if we decide to consider a new utility function $u'_w(\alpha) = w_1 \cdot cost_e(\alpha) + w_2 \cdot time_h(\alpha)$, where $time_h(\alpha)$ evaluates the time of α in hours, the new weak preference relation associated with $w = [w_1 = 0.5, w_2 = 0.5]$ would be different.

Suppose we normalize the outcome space associated to an input set A $\in \mathcal{M}$, i.e., we define the evaluators as $X'_i : \text{A} \to [0, 1]$ for each $i \in \{1, \dots p\}$. In that case, we may get an interpretation of the weights vector that is more similar to the concept of importance. However, it may still not be the correct interpretation. This because the weak preference relation associated with a weights vector w would be strictly dependent on the lower and upper bounds of each criterion used to normalize the outcome space with respect to A. Thus, different input

sets could lead to different bounds and then different weak preference relations associated with a specific weights vector. However, we can say that the weights are somehow related to the importance that DM gives to each criterion, since increasing the value of a specific weight means increasing the importance that the DM gives to the corresponding criterion.

2.1.5 Generalised additive independence utility model

Generalised additive independence (GAI) models have been first defined by Fishburn [Fis67a, Fis70]. GAI models are a generalisation of additive models where preferential independence is related to sets of criteria rather than each single criterion. Recent works based on this utility model include [BG13, BBB13, BB12, Bra12].

Let $I_j \subseteq \{1,\ldots,p\}$ be a set of indexes with $I_j \in \{1,\ldots,M\}$ such that $I_1 \cup,\ldots,\cup I_M = \{1,\ldots,p\}$. Let $X_{I_j}(\alpha) \in I\!R^{|I_j|}$ be the partial outcome of α representing the values of the criteria indexed by I_j. The sets of criteria indexed by $I_1,\ldots,I_M$ used to evaluate an alternative $\alpha \in \Omega$ are mutually preferentially independent if and only if there exists an additive ordinal utility function such that:

$$u(\alpha) = \sum_{j=1}^{M} u_j(X_{I_j}(\alpha)) \tag{2.11}$$

where each u_j is a subutility function defined over the criteria indexed by I_j [BG95, GP04]. This enables GAI models to capture preferentially-dependent criteria.

GAI models have also been represented with graphical structures called GAI networks [GP04] which is similar to junction graphs used for Bayesian networks [CDLS06]. A GAI network is an undirected graph where each node represents a criterion and to each clique corresponds a factor F_I. Such a graph keep track of all the dependencies between the different components of the subutility function.

2.2 Classical Methods for Preference Elicitation

Preference elicitation refers to the process of assessing the DM's preferences. The DM's preferences can be used, for example, by a recommender system

capable of supporting the DM in a decision process (see, e.g., [CP04] for some examples of recommender systems). Some of the most common preference elicitation approaches adopted by recommeneder systems are classified as follows [LWM+15, A+16]:

1. *Content-based* (see, e.g., [PB07]): recommendations are based on similarities between the descriptions of the available alternatives and the user preferences recorded from previous interactions with the system.
2. *Collaborative filtering* (see, e.g., [SKKR01]) recommendations are computed considering past similar items liked by the user (item-based) or considering items liked by a set of users of the system with similar interests (user-based).
3. *Knowledge-based* (see, e.g., [Bur02]) recommendations are based on knowledge about users, items or their relationships. In this case, the user specifies constraints and preferences over alternatives that will be used to refine the user preference model and generate recommendations.

The main difference between knowledge-based systems with respect to content-based and collaborative filtering systems is that the former usually does not require a big amount of data to compute a recommendation. Thus, knowledge-based systems are well suited for a cold-start scenario, i.e., when we do not have any information about the DM's preferences. In this thesis we will focus on knowledge-based recommender systems. In particular, we assume a preference model based on MAUT, which supposes the existence of an unknown DM's utility function with a predefined structure. Regarding the preference elicitation procedures instead, we will focus on systems which reduce the uncertainty of the DM's utility function through an iterative interaction with the DM.

The following are a brief summary of some of the main approaches used to infer the DM's preference [Bra12, Ben17] in a MAUT context. Note that these approaches are not mutually exclusive.

2.2.1 Complete elicitation of the utility function

The goal of the classical approach is to completely specify the DM's utility function through a series of questions and answers [Fis67b, Rai68, Far84].

This elicitation process can be summarised with the following steps:

1. Determination of the property of the underlying utility function such as

number of criteria, outcome space and preferential independence.

2. Complete assessment of the parameters of the utility function using elaborate interview techniques.

3. Check of the consistency of the DM's responses and sensitive analysis. In case of inconsistencies, the DM is asked to revise her responses. The sensitivity analysis evaluates the sensitivity of the output with respect to different inputs.

Experiments with real users have shown that this process can be a difficult and error-prone task [Sim55, TK74, PFT03]. Furthermore, it is difficult to apply this approach in a combinatorial domain since it can rapidly become expensive in terms of questions for the DM. In fact, with multi-criteria outcomes, often the number of possible alternatives grows exponentially with respect to the number of criteria considered. Thus, learning each point of the utility function may not be feasible in practice.

2.2.2 Interactive optimisation

Interactive optimisation is an approach widely studied in a multi-criteria decision-making context. It allows the exploration of Pareto-optimal alternatives based on different interactions with the DM and without listing all the available alternatives (see, e.g., [GDF72, SS78, Kor05, BLL20]). With this approach, it is common to adopt a parametrised form of DM's utility function. The parameterisation represents different DM's preference scenarios, and the restrictions on the parameters represent the information obtained by the interaction with the DM. Common utility functions used in this context are linear scalarising functions such as the weighted sum utility function (see Section 2.1.4), or the Chebyshev type scalarising functions, i.e., utility functions that consider the distance from an ideal utility vector (see, e.g., [Wie80]).

Methods that iteratively interact with the DM to reduce the uncertainty about a parameterised preference models are also called *Incremental* [BP15b, BG15, LB11b]. The purpose of such methods is to recommend alternatives to the DM without defining a precise utility function for the DM using methods such as the minimax regret criterion (see Section 2.3.2).

A typical interactive approach consists of the following steps:

1. *Computation*: generate some solutions belonging to the Pareto frontier.

2. *Interaction*: show to the DM some of the generated solution asking to input new preference information.
3. *Termination*: interruption of the elicitation process by the DM or if some specified stopping criterion has been satisfied.

Different approaches have been explored to generate new queries for the DM (see, e.g., [SR91]). A classical interactive approach is based on a comparison of alternatives (see, e.g., [ZW76, SC83]), i.e., the preference elicitation system asks the DM to specify her preference between a set of alternatives, and the response is used to reduce the uncertainty of the preference model. Other approaches are for example based on queries for the DM asking to define *aspiration levels* or *interval judgments* Aspiration levels are a reference point of the value of the criteria evaluating an alternative (see, e.g., [Wie80, LSZ92]). Interval judgments define trade-offs between different criteria (see, e.g., [SH92, KM97]).

A drawback of interactive optimisation is that it may require many interactions with the DM in order to find a solution satisfying the DM's expectations. Also, the higher the number of queries, the higher the risk of collecting inconsistent preference information with respect to the (unknown) real DM's preference. However, in the literature we can find interactive methods dealing with inconsistent user preferences; the most common (presented in the next two sections) are based on a Bayesian representation of the preference state space, or focused on the minimisation of an error function. Alternatively, with a preference elicitation method such as [DTP18], we can start the learning process with a predefined instantiation of the utility function parameters, and update it at each DM's response.

Automatic systems implementing an interactive elicitation procedure require also a user interface to interact with the DM in order to collect her preferences (see, e.g., [KL86, Kli92, Bly02]). Such an interface may be used also to show the current state of the system, since detailed information displayed with a graphical interface may also increase the DM's awareness with respect to the consequences of her input preferences. Furthermore, if the system detects an inconsistency in the input preferences, with a user interface it could be directly notified to the DM, asking for a resolution.

2.2.3 Bayesian preference elicitation

Standard Bayesian approaches for preference elicitation represent the uncertainty of the DM's utility function with a prior probability distribution which can be, for example, computed using data collected from other users with similar preferences. Without any other information about the DM's preference, it seems to be reasonable to recommend the alternative that maximises the expectation of the DM's utility function. On the other hand, if there is the possibility of interacting with the DM, then we can consider asking her some questions to try to improve the accuracy of the utility function. In fact, Bayesian approaches are often used along with an incremental preference elicitation process interacting with the DM until a stopping criteria is satisfied (see, e.g., [CKP00, BB06]).

In this context, the value of information of a query is associated with the expectation of the utility function. Thus, a response to a query with a high value of information comes with a high increase of the expectation of the DM's utility function. However, finding the query with the highest value of information is in practice often unfeasible. This is because to compute the exact value of information, we should consider all the possible future queries and responses (see, e.g., [Bou02]), and this may well be very computationally expensive. Therefore, it may be more practical to adopt a myopic strategy (see, e.g., [VB10]), where the value of information of a query is computed comparing the expected utility before and after the DM's answer.

It has been shown that a Bayesian approach for an interactive preference elicitation allows one to formulate relevant and personalised recommendations with a reasonable amount of interaction with the DM (see, e.g., [GS10a, VB10]). Also, an advantage of this approach relates to its intrinsic property of being robust with respect to inconsistent DM's preferences given that an input preference does not exclude preference models; instead, it redefines the probability distribution over preference state space. However, the complexity of the process of query selection is usually very high, also adopting a myopic strategy. Thus in some cases, such methods are approximated in practice using, for example, Monte Carlo methods [Via12, VLHB19].

2.2.4 Minimisation of an error function

In the literature, we can find methods that consider an input set of preferences of a specific DM and try to estimate a precise utility function minimising an

error function. For example, there is a class of methods called *Utilié Additive* (UTA) (e.g., [JLS82, SGM05, SGMP18]) where the purpose is to assess an additive utility function from a set of DM's preferences given as input. The input preferences could, for example, express a weak ordering of a subset of alternatives, or specify if some criteria are more important than others. In this case, a common approach is to estimate a utility function solving a constraint optimisation problem and minimising the utility error (see, e.g., [MR05]). A weak point of this approach lies in the choice of the function to optimise (e.g., quadratic error [MR05] or variance [Koj07]), which sometimes seems rather arbitrary.

A similar approach makes use of a support vector machine (SVM) algorithm [MW16] to estimate a weighted sum utility function parameterised with respect to the set of weights. Briefly, SVM can be used for binary classification in $\mathbb{R}^p$ defining a hyperplane that maximises the distance of the points belonging to two different classes. This preference learning method is based on an interactive approach with binary queries of the form "is alternative α better than alternative β?". Each query is represented as a point in a p-dimensional space and the hyperplane learned with the SVM algorithm will divide queries which answer is yes from queries which answer is no. Such a hyperplane can then be used to rank an input set of alternatives.

A more general approach is based on *robust ordinal regression* [GSFM10, GMS14] which has been introduced to take into account all the sets of parameters of a preference model compatible with the preference information given as input. This method is a generalisation of UTA where the preference information is used to define a set of linear constraints representing conditions on the compatible utility functions. However, without a predefined structure for the utility function, we may need more input data to get a good approximation of the DM's preferences.

2.3 Parameterised Preferences

As we discussed in the previous section, the precise definition of the DM's utility function is liable to be a difficult and error-prone task. Thus, modern MAUT preference elicitation systems consider different preference models parameterised by a set $\mathcal{U}$ of *scenarios*. Each scenario $w \in \mathcal{U}$, i.e., a specific configuration of the parameters of the utility function defines an ordinal utility

function u_w which can be used to compare different alternatives. Suppose that, in a particular situation, A is the set of alternatives that are available to the decision-maker. If we knew that w was the true scenario, so that u_w represents the DM's preferences over alternatives, then we would be able to choose the best element of A with respect to u_w. More precisely, we assume here that the utility function u_w of a scenario $w \in \mathcal{U}$ represents a weak preference relation $\succcurlyeq_w$ defined over the space of alternatives. Thus, w is viewed as a model of the DM's preferences that is consistent with the preference information we know. However, the situation can be ambiguous, given a non-singleton set $\mathcal{U}$ of possible user models or scenarios. The main purpose of preference elicitation system based on parameterised utility functions is then to reduce the uncertainty of the DM's preferences represented by the set $\mathcal{U}$ of possible parameters. Common utility functions used in preference elicitation context are such as the weighted sum presented in Section 2.1.4, ordered weighted average (see, e.g., [Yag88]) and Choquet integral (see, e.g., [GPS10]).

In the context of social choice, utility functions are usually called *social choice functions* or *rank aggregation functions* (see, e.g., [LB11a]). In this context the preferences of each voter are usually given as input and represented as a ranking of the available alternatives. The main purpose of social choice is to define the function used to aggregate the input preferences of the voters, which can be used to find a *consensus alternative* or a *consensus ranking*. Common social choice functions are such as the Plurality Rule or the Borda Count (see, e.g., [FSST17]). In Chapter3 we will define the concepts of minimal equivalent subset, and it is worth noting that in the context of social choice this corresponds to the set of alternatives which are not Pareto dominated with respect to the vectors of agents scores associated with each alternative. Similarly, the concept of dominance for a set of alternatives reduces in this context to the Pareto dominance.

A major division in recent work on parameterised preference models is whether a Bayesian model is assumed over the parameters, or if there is a purely qualitative (logical) representation of the uncertainty. Bayesian approaches include for example [CKP00, Bou02, VB10]. Work involving a qualitative uncertainty representation includes [BPPS06, VB09, BB07, MRW13, BPV17a]. In particular, qualitative imprecise preference models based on the weighted sum value function have been considered in work such as [SH10, MRW12, KVVA17], including in a conversational recommender system context [BR07, VB09]. As we have discussed in the previous section, Bayesian approaches have the advan-

tage of being more robust with respect to inconsistent input preferences at the expense of an increased computational complexity. Qualitative methods instead are in general more efficient in terms of computational complexity, but inconsistent query responses can compromise the quality of the recommendation. In this thesis we will focus mainly on the latter type of parameterised preference models.

In the following two sections we describe in detail *imprecisely specified multi-attribute utility theory* (ISMAUT) and the minimax regret criterion (MMR). ISMAUT is one of the earliest attempt to deal with parameterised utility information, and it is based on the weighted sum value function. MMR is a method to rank alternatives when there is uncertainty on the DM's utility function, and it will be considered in all the chapters of this thesis.

2.3.1 ISMAUT

The main idea behind ISMAUT [WSD84] is to translate the DM's preferences into linear inequalities reducing the set of feasible parameters of the utility function, and then reducing also the set of alternatives to those that are not dominated by any other alternative. Related research such as [Haz86] and [Web87], deals with similar issues. ISMAUT methods consider a finite set of alternatives A, and an additive utility function parameterised with respect to the weights vector $w = (w_1, \ldots, w_p)$ and evaluators $X = (X_i, \ldots, X_p)$, i.e., the DM's utility function of an alternative $\alpha \in$ A is defined to be:

$$u_{w,X}(\alpha) = \sum_{i=1}^{p} w_i X_i(\alpha) = w \cdot X(\alpha) \tag{2.12}$$

where $w \in \{w \in I\!R^p : w_i \geq 0, \sum_{i=1}^{p} = 1\}$ is a point of the *unit* $(p-1)$*-simplex*, and X : A $\rightarrow I\!R^p$ defines the p-dimensional vector of evaluators for alternative α. The idea behind ISMAUT is to interact with the DM in order to collect a set of preference information used to reduce the set of possible utility functions. Such input preferences can be categorised as follows:

1. Weights constraints: the DM can define tradeoff between the weights (e.g. $w_{\text{cost}} \geq w_{\text{time}}$) or bounds on their value (e.g. $w_{\text{time}} \in [0, 0.5]$). These constraints define the subset of feasible weights vector $\mathcal{W} \subseteq \{w \in I\!R^p : w_i \geq 0, \sum_{i=1}^{p} = 1\}$.

2. Constraints on evaluators: supposing that the domain of criteria can be

labelled, the DM can specify a preference between different values of a criterion (e.g. $X_{\text{cost}}(\text{cheap}) \geq X_{\text{cost}}(\text{expensive})$), or bounds on the utility value of a criterion value (e.g. $X_{\text{cost}}(\text{cheap}) \in [0, 0.2]$). These constraints define the set $\mathcal{X}$ of feasible evaluators $X = (X_1, ..., X_p)$.

3. Pairwise comparison of alternatives: If the DM prefers an alternative α to another alternative β, then we can translate the preference $\alpha \succcurlyeq \beta$ into the inequality $w \cdot X(\alpha) \geq w \cdot X(\beta)$ that can be used to further reduce the set of possible parameters. Let $\Lambda = \{(\alpha, \beta) \in \mathsf{A} \times \mathsf{A} : \alpha \succcurlyeq \beta\}$ be the set of such pairwise comparisons.

This input preference information is used to define the set of feasible scenarios $\langle w, X \rangle \in \mathcal{U}$ and the corresponding utility function $u_{w,X}$. More precisely, the tuple $\langle w, X \rangle \in \mathcal{U}$ if and only if:

1. $w \in \mathcal{W}$
2. $X \in \mathcal{X}$
3. $w \cdot X(\alpha) \geq w \cdot X(\beta)$, for all $(\alpha, \beta) \in \Lambda$.

The goal of classical ISMAUT approaches is to reduce the uncertainty of the feasible set $\mathcal{U}$ of utility functions in order to reduce also the set of undominated alternatives to a manageable size. In this context, an alternative α dominates another alternative β if and only if:

$$\min_{\langle w,X \rangle \in \mathcal{U}} w \cdot (X(\alpha) - X(\beta)) \geq 0. \tag{2.13}$$

The parameterisation of the utility function of classical ISMAUT refers to the weights vectors w and single-criterion utility functions X. Thus, to check dominance between alternatives, we need to solve a quadratic problem.

In works such as [ILC01b] and [GK03] the authors consider an incremental approach based on the ISMAUT model but limiting the paramtrisation of the utility function to the weights vector w, i.e., assuming fixed input single-criterion utility functions. The idea is to ask a series of queries binary queries such as "Do you prefer the alternative α or the alternative β?", and to translate the response into a linear constraint with the purpose of reducing the state space of the feasible weights vectors. In Chapter 4, Chapter 5, and in the experimental results of Chapter 3, we will focus on this preference model supposing then the evaluators given as input.

2.3.2 Minimax Regret decision criterion

The *Minimax Regret* [Sav72, KY13] criterion is generally used to solve decision problem under uncertainty. More recently, it has been used in the context of artificial intelligence to evaluate alternatives for potential questions and recommendations, where the uncertainty refers to the parameters of the decision-model [SH01, Bou02, BPPS06, BB06]. It prescribes an outcome that minimises the worst-case loss with respect to the utility function uncertainty represented by the set of feasible scenarios $\mathcal{U}$, and it is, therefore, a reasonable criterion used to evaluate alternatives in decision support systems. Applications include, for example, the elicitation of multi-attribute utilities (e.g., [WB03, BB07, BP15b], or the elicitation of preferences for ranking and voting problems (e.g., [LB11b, BDDPV16, BPV17b]. The practical effectiveness of this approach has been proven in numerous works (e.g., [WB03, Bra12, Ben17] and in particular during a study carried out with real users [BB10].

In this thesis we will consider the minimax regret criterion to evaluate alternatives with respect to parameterised utility functions. For simplicity we suppose a set of parameters $\mathcal{U} \subset \mathbb{R}^p$ closed and bounded, and for any $w \in \mathcal{U}$ a continuous utility function $u_w(\alpha)$ with respect to w for any $\alpha \in \Omega$. However, the following definitions can be generalised to generic sets of parameters $\mathcal{U}$ and utility functions u_w replacing $\max_{w\in\mathcal{U}}$ with $\sup_{w\in\mathcal{U}}$.

Pairwise Max Regret (*PMR*): The *PMR* in $\mathcal{W} \subseteq \mathcal{U}$ of an alternative $\alpha \in \Omega$ with respect to another alternative $\beta \in \Omega$ is defined by:

$$PMR_{\mathcal{W}}(\alpha, \beta) = \max_{w\in\mathcal{U}}(u_w(\beta) - u_w(\alpha)). \tag{2.14}$$

The $PMR_{\mathcal{W}}$ of α with respect to β is then the worst-case loss among all the feasible utility functions u_w with respect to $\mathcal{W}$ of recommending α instead of β. Note that if $PMR_{\mathcal{W}}(\alpha, \beta) \leq 0$ then $u_w(\alpha) \geq u_w(\beta)$ for any $w \in \mathcal{W}$ which means that alternative α is at least as good as alternative β with respect to the available preference information.

Max Regret (*MR*): The *MR* in $\mathcal{W} \subseteq \mathcal{U}$ of an alternative α with respect to a finite set of alternatives $\mathrm{A} \subseteq \Omega$ is defined by:

$$MR_{\mathcal{W}}(\alpha, \mathrm{A}) = \max_{\beta\in\mathrm{A}} PMR_{\mathcal{W}}(\alpha, \beta) \tag{2.15}$$

The $MR_{\mathcal{W}}$ of α with respect to A is then the worst-case loss among all the feasible utility functions u_w with respect to $\mathcal{W}$ of recommending α instead of any other $\beta \in$ A. In other words, it is the maximum loss of an alternative in terms of utility units with respect to all the alternatives in A and all the possible scenarios $\mathcal{W}$.

The following two lemmas state well-known properties of max regret; proofs are included for completeness.

Lemma 2.3.1. *Let $A \subseteq \Omega$ be a finite set of alternatives and $\alpha \in A$. For $\mathcal{W} \subseteq \mathcal{U}$, $MR_{\mathcal{W}}(\alpha, A) \geq 0$.*

Proof. Since $\alpha \in$ A and we are maximising w.r.t. any $\beta \in$ A, with $\beta = \alpha$ we get $u_w(\alpha) = u_w(\beta) = 0$ for any $w \in \mathcal{W}$. Thus:

$$\begin{aligned} MR_{\mathcal{W}}(\alpha, \mathrm{A}) &= \max_{\beta \in \mathrm{A}} PMR_{\mathcal{W}}(\alpha, \beta) \\ &= \max_{\beta \in \mathrm{A}} \max_{w \in \mathcal{W}} (u_w(\beta) - u_w(\alpha)) \\ &\geq \max_{\beta \in \{\alpha\}} \max_{w \in \mathcal{W}} (u_w(\beta) - u_w(\alpha)) = 0 \end{aligned} \tag{2.16}$$

□

Lemma 2.3.2. *Let $A \subseteq \Omega$ be a finite set of alternatives and $\alpha \in A$. For $\mathcal{W} \subseteq \mathcal{U}$, $MR_{\mathcal{W}}(\alpha, A) = 0$ if and only if $u_w(\alpha) \geq u_w(\beta)$ for any $w \in \mathcal{W}$ and for any $\beta \in A$.*

Proof. Since $\alpha \in$ A, From Lemma 2.3.1 it follows that $MR_{\mathcal{W}}(\alpha, \mathrm{A}) \geq 0$. Therefore, $MR_{\mathcal{W}}(\alpha, \mathrm{A}) = \max_{\beta \in \mathrm{A}} \max_{w \in \mathcal{W}} (u_w(\beta) - u_w(\alpha)) = 0$ if and only if $\arg\max_{\beta \in \mathrm{A}} (\max_{w \in \mathcal{W}} (u_w(\beta) - u_w(\alpha)) = \alpha$ which is if and only if $u_w(\alpha) \geq u_w(\beta)$ for any $w \in \mathcal{W}$ and for any $\beta \in$ A. □

Lemma 2.3.2 can be used to recommend optimal alternatives with respect to the DM's preferences represented by $\mathcal{W} \subseteq \mathcal{U}$, since if alternative α is such that $MR_{\mathcal{W}}(\alpha, \mathrm{A}) = 0$, then it must be one of the most preferred alternatives with respect to any scenario $w \in \mathcal{W}$.

Minimax Regret (*MMR*): The *MMR* in $\mathcal{W} \subseteq \mathcal{U}$ of a finite set of alternatives A $\subseteq \Omega$ is defined by:

$$MMR_{\mathcal{W}}(\mathrm{A}) = \min_{\alpha \in \mathrm{A}} MR_{\mathcal{W}}(\alpha, \mathrm{A}) \tag{2.17}$$

The value $MMR_{\mathcal{W}}(\mathrm{A})$ is then the minimum max regret we can get from

alternatives in A. An alternative $\alpha \in$ A that minimises $MMR_{\mathcal{W}}(\text{A})$ is then an optimal recommendation in $\mathcal{W}$ with respect to the minimax regret criterion and it can be interpreted as the best worst-case loss recommendation. By recommending to the DM an alternative associated with minimax regret, i.e., alternative $\alpha^* \in \arg\max_{\alpha \in \text{A}} MR_{\mathcal{W}}(\alpha, \text{A})$, we provide robustness in face of uncertainty (due to not knowing the DM's utility function).

When u_w is a linear function with respect to a set of possible parameters $\mathcal{U} \subseteq \mathbb{R}^p$, the regret of an alternative (and the minimax regret) is maximised in one of the extreme points of $\mathcal{W}$ (see, e.g., [KVVA17]). This is a simple but very important result since it allows an efficient computation of the *MMR*.

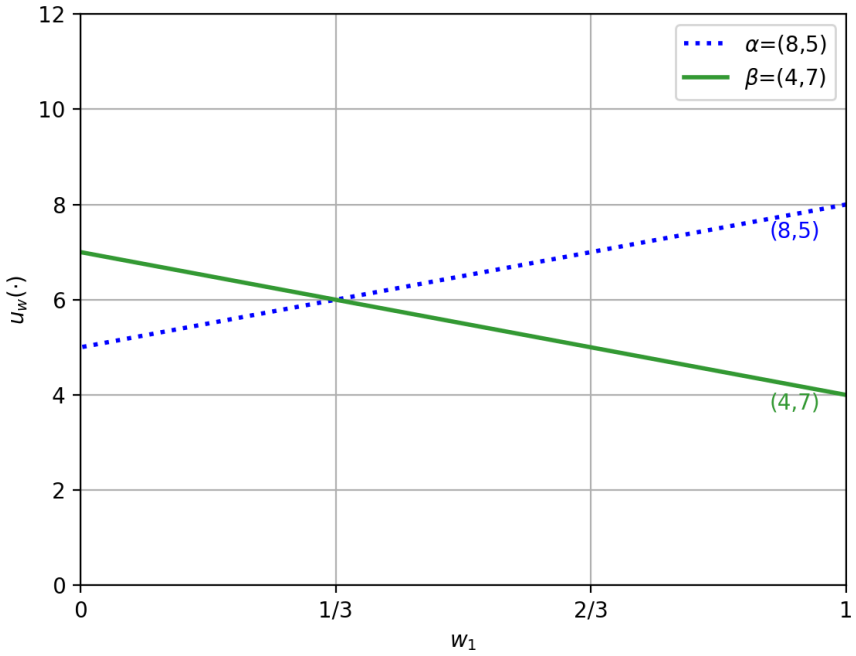


Figure 2.1: Plot of the linear utility functions $u_w(\cdot)$ of the alternatives $\alpha = (8, 5)$ (blue dotted) and $\beta = (4, 7)$ (green solid) with respect to the scenarios $\mathcal{U} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$.

Example 4. *Consider a linear utility function $u_w(\alpha) = w \cdot \alpha$ parameterised with respect to $w \in \mathcal{U}$, where $\alpha = (\alpha_1, \alpha_2)$ is an alternative evaluated with two criteria $\alpha_1, \alpha_2 \in \mathbb{R}$ and $\mathcal{U} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$ is the set of weights vectors representing all the possible parameters of u_w. (Note that since $w_1 = 1 - w_2$, the utility function u_w can be expressed as function of a single weight, i.e., $u_w = w_1 \cdot \alpha_2 + (1 - w_1 \cdot \alpha_2)$.*

Figure 2.1 shows the utility function $u_w(\cdot)$ (vertical axis) of the two alternatives

$\alpha = (8, 5)$ and $\beta = (4, 7)$ for each possible scenario w (horizontal axis). Let $A = \{\alpha, \beta\}$.

The regret in $\mathcal{U}$ of α and β with respect to A is maximised in $w_1 = 0$ and $w_1 = 1$ respectively, i.e.:

- $MR_{\mathcal{U}}(\alpha, A) = \max_{w \in \mathcal{U}}(u_w(\beta) - u_w(\alpha)) = (0, 1) \cdot ((4, 7) - (8, 5)) = 2$
- $MR_{\mathcal{U}}(\beta, A) = \max_{w \in \mathcal{U}}(u_w(\alpha) - u_w(\beta)) = (1, 0) \cdot ((8, 5) - (4, 7)) = 4$

The minimax regret of the set of alternatives A is minimised in $w_1 = 0$ by the alternative α, i.e.:

- $MMR_{\mathcal{U}}(A) = \min(MR_{\mathcal{U}}(\alpha, A), MR_{\mathcal{U}}(\beta, A)) = 2$.

2.3.2.1 Setwise Minimax Regret decision criterion

In many applications, it is desirable to produce a *recommendation set*, and not just a single recommendation, allowing the decision-maker to pick the alternative (among those of the recommendation set) that provides the most value to her. Intuitively, by providing several recommendations, it is more likely that at least one of them will have high utility value to the decision-maker. As originally observed by Price and Messinger [PM05] it is, therefore, a good idea to show *diverse* recommendations that have high value for different parts of the parameter space $\mathcal{U}$.

In [VB09] and [VB20] the authors generalised the concept of Minimax Regret defining the *Setwise max Regret* (*SMR*) which is used to evaluate a set of alternatives rather than a single alternative, and the setwise minimax regret (*SMMR*) which is used to select an optimal set with respect to *SMR*. This provides a principled method for capturing the idea of recommendation sets. Assume that, when we provide A as recommendation set, the DM is able to pick the most preferred item (the one with the highest value) in A. The DM then perceives a value $\text{Val}_\text{A}(w) = \max_{\alpha \in \text{A}} u_w(\alpha)$ where the *true* utility function is dictated by the parameter $w \in \mathcal{U}$.

Utility function of A $\in \mathcal{M}$: For a set of alternatives A $\in \mathcal{M}$, we define the DM's utility function of A with respect to a parametersation $w \in \mathcal{U}$ as:

$$\text{Val}_\text{A}(w) = \max_{\alpha \in \text{A}} u_w(\alpha) \tag{2.18}$$

The regret of a set A with respect to another set B in w is the difference between the utility of the best item under w in B and the utility of the best item with respect to w in the set A; that is, $\text{Val}_\text{B}(w) - \text{Val}_\text{A}(w)$. As for the max regret, with the definition of the setwise max regret we suppose that the set of parameters $\mathcal{U}$ is closed and bounded and for any $w \in \mathcal{U}$ the utility function $u_w(\alpha)$ is continuous with respect to w for any $\alpha \in \Omega$. However, replacing $\max_{w\in\mathcal{U}}$ with $\sup_{w\in\mathcal{U}}$ we get a generalised definition which is suitable for generic sets of parameters $\mathcal{U}$ and utility functions u_w.

Setwise Max Regret (*SMR*): The *SMR* in $\mathcal{W}$ of a finite set of alternatives $\text{A} \subseteq \Omega$ with respect to another finite set of alternatives $\text{B} \subseteq \Omega$ is defined by:

$$SMR_{\mathcal{W}}(\text{A}, \text{B}) = \max_{w\in\mathcal{W}}(\text{Val}_\text{B}(w) - \text{Val}_\text{A}(w)) \tag{2.19}$$

The value $SMR_{\mathcal{W}}(\text{A}, \text{B})$ is then the worst-case loss of recommending the best alternative $\alpha \in \text{A}$ instead of the best alternative $\beta \in \text{B}$ supposing that the DM's utility function parameter could be any $w \in \mathcal{W}$. From the definition it follows immediately that if $SMR_{\mathcal{W}}(\text{A}, \text{B}) < 0$, then A is strictly better than B is every scenario $w \in \mathcal{W}$, i.e., for any $w \in \mathcal{W}$ and $\beta \in \text{B}$ there exists $\alpha \in \text{A}$ such that $u_w(\alpha) > u_w(\beta)$.

The following lemmas state well-known properties of setwise max regret; proofs are included for completeness.

Lemma 2.3.3. *Let $B \subseteq \Omega$ be a finite set of alternatives and $A \subseteq B$. For $\mathcal{W} \in \mathcal{U}$, $SMR_{\mathcal{W}}(A, B) \geq 0$.*

Proof. Since $\text{Val}_\text{B}(w) = \max_{\beta\in\text{B}} u_w(\beta)$, $\text{Val}_\text{A}(w) = \max_{\alpha\in\text{A}} u_w(\alpha)$ and $\text{A} \subseteq \text{B}$, then $\text{Val}_\text{B}(w) \geq \text{Val}_\text{A}(w)$. Thus, $SMR_{\mathcal{W}}(\text{A}, \text{B}) = \max_{w\in\mathcal{W}}(\text{Val}_\text{B}(w) - \text{Val}_\text{A}(w)) \geq \max_{w\in\mathcal{W}}(\text{Val}_\text{B}(w) - \text{Val}_\text{B}(w)) = 0$. □

Lemma 2.3.4. *Let $B \subseteq \Omega$ be a finite set of alternatives and $A \subseteq B$. For $\mathcal{W} \subseteq \mathcal{U}$, $SMR_{\mathcal{W}}(A, B) = 0$ if and only if for any $w \in \mathcal{W}$ and for any $\beta \in B$ there exists $\alpha \in A$ such that $u_w(\alpha) \geq u_w(\beta)$*

Proof. Since $\text{A} \subseteq \text{B}$, from Lemma 2.3.3 it follows that $SMR_{\mathcal{W}}(\text{A}, \text{B}) \geq 0$. Therefore, $SMR_{\mathcal{W}}(\text{A}, \text{B}) = \max_{w\in\mathcal{W}}(\text{Val}_\text{B}(w) - \text{Val}_\text{A}(w)) = 0$ if and only if $\text{Val}_\text{B}(w) - \text{Val}_\text{A}(w) = 0$ for all $w \in \mathcal{W}$ which is if and only if for any $w \in \mathcal{W}$ and for any $\beta \in \text{B}$ there exists $\alpha \in \text{A}$ such that $u_w(\alpha) \geq u_w(\beta)$. □

Lemma 2.3.5. *Let $A, B \subseteq \Omega$ be two finite sets of alternatives. For $\mathcal{W} \subseteq \mathcal{U}$, $SMR_{\mathcal{W}}(A, B) = \max_{\beta \in B} SMR_{\mathcal{W}}(A, \{\beta\})$*

Proof. $\max_{\beta \in \mathrm{B}} SMR_{\mathcal{W}}(\mathrm{A}, \{\beta\}) = \max_{\beta \in \mathrm{B}} \max_{w \in \mathcal{W}} SMR_{\{w\}}(\mathrm{A}, \{\beta\})$, which equals $\max_{w \in \mathcal{W}} \max_{\beta \in \mathrm{B}} SMR_{\{w\}}(\mathrm{A}, \{\beta\}) = \max_{w \in \mathcal{W}} SMR_{\{w\}}(\mathrm{A}, \mathrm{B}) = SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$. □

Minimax Setwise Regret (*SMMR*): The *SMMR* in $\mathcal{W} \subseteq \mathcal{U}$ of all the subsets A of cardinality k of a finite set of alternatives $\mathrm{B} \subseteq \Omega$ is defined by:

$$SMMR^{k}_{\mathcal{W}}(\mathrm{B}) = \min_{\mathrm{A} \subseteq \mathrm{B}: |\mathrm{A}| = k} SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) \tag{2.20}$$

The value $SMMR^{k}_{\mathcal{W}}(\mathrm{B})$ is then the minimum setwise max regret we can get from all the possible subset of alternatives A with cardinality k of B with respect to any parameter $w \in \mathcal{W}$. A subset A of B that minimises $SMMR^{k}_{\mathcal{W}}(\mathrm{B})$ is an optimal set of recommendation with cardinality k with respect to the Minimax criterion. Recommendation sets can be used in elicitation, where they are treated as *choice queries* (i.e., questions of the kind *"Among a, b, and c, which one do you prefer?"*) with the goal of reducing uncertainty to improve the quality of future recommendations; that is, reducing minimax regret. It turns out [VB09, VB20] that optimal recommendation sets w.r.t. *SMMR* are also myopically optimal in an elicitation sense, as they ensure the highest worst-case (with respect to the possible query's responses) reduction of minimax regret *a posteriori* (see Section 2.3.2.2 below for details).

Example 5. *Consider the linear utility function u_w defined in Example 4. Figure 2.2 shows the utility function $u_w(\cdot)$ (vertical axis) of three alternatives $\alpha = (2, 8)$, $\beta = (8, 2)$ and $\gamma = (6, 6)$ for each possible scenario w (horizontal axis). Let $\Omega = \{\alpha, \beta, \gamma\}$, $A = \{\alpha, \beta\}$, $B = \{\alpha, \gamma\}$, $C = \{\beta, \gamma\}$. The dash-dotted red line in Figure 2.2 represents the function $Val_A(w) = \max(u_w(\alpha), u_w(\beta))$ used to compute the setwise max regret of A with respect to the set of scenarios $\mathcal{U} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$*

The setwise max regret of A, B and C in $\mathcal{U}$ with respect to A is maximised in $w_1 = 0.5$, $w_1 = 1$ and $w_1 = 0$ respectively i.e.:

- $SMR_{\mathcal{U}}(A, \Omega) = \max_{w \in \mathcal{U}}(Val_{\Omega}(w) - Val_A(w)) = (0.5, 0.5) \cdot ((6, 6) - (8, 2)) = 1$
- $SMR_{\mathcal{U}}(B, \Omega) = \max_{w \in \mathcal{U}}(Val_{\Omega}(w) - Val_B(w)) = (1, 0) \cdot ((8, 2) - (6, 6)) = 2$
- $SMR_{\mathcal{U}}(C, \Omega) = \max_{w \in \mathcal{U}}(Val_{\Omega}(w) - Val_C(w)) = (0, 1) \cdot ((2, 8) - (6, 6)) = 2$

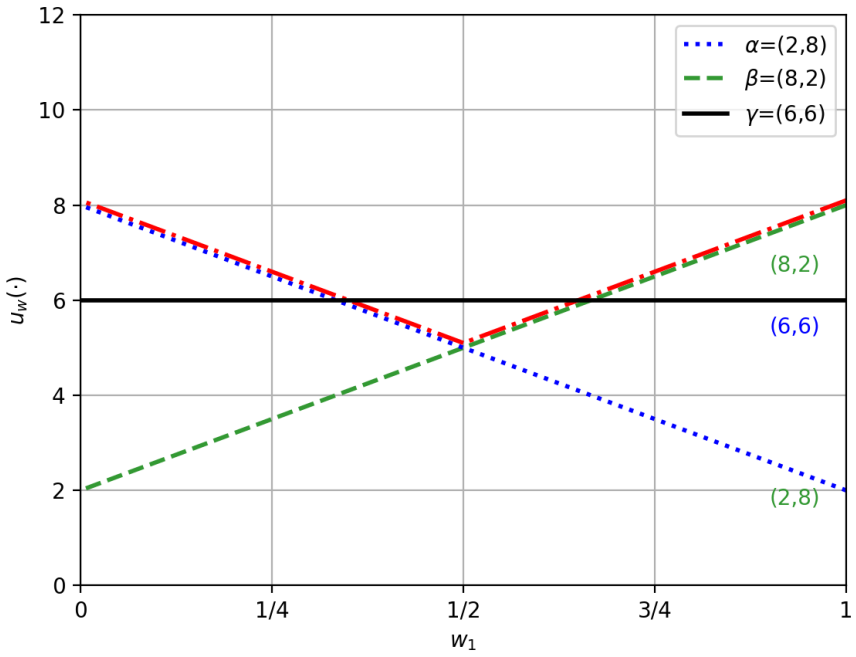


Figure 2.2: Plot of the linear utility functions $u_w(\cdot)$ for the alternatives $\alpha = (2, 8)$ (blue dotted), $\beta = (8, 2)$ (green dashed) and $\gamma = (6, 6)$ (black solid), and $\text{Val}_{\{\alpha,\beta\}}(w) = \max(u_w(\alpha), u_w(\beta))$ (red dash-dotted) with respect to the set of scenarios $\mathcal{U} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$

The setwise minimax regret of all the subsets of Ω *of cardinality* 2 *is minimised in* $w = (0.5, 0.5)$ *by the subset A, i.e.:*

- $SMMR_{\mathcal{U}}(\Omega) = \min(SMR_{\mathcal{U}}(A, \Omega), SMR_{\mathcal{U}}(B, \Omega), SMR_{\mathcal{U}}(C, \Omega) = 1$

2.3.2.2 Incremental elicitation based on Minimax Regret

Using the Minimax regret criterion to select an alternative without any knowledge about the DM's preferences may be too risky. In fact, when the *MMR* has a high value, the quality of an optimal alternative with respect to the Minimax criterion may be too low. If possible, it may be then useful to elicit some of the DM's preferences to reduce the *MMR* and increase then the quality of the optimal alternatives. A common elicitation strategy used along with the Minimax regret criterion is based on an iterative interaction with the DM, where at each iteration we ask an *informative* query, i.e. a query whose answer is guaranteed to ensure the reduction of the uncertainty on the DM's preferences. In such context, the Minimax criterion can be used as a stopping criterion, i.e., if the value *MMR* is less or equal than a specific threshold $\epsilon \geq 0$, then we stop

the interaction, recommending an alternative that minimises *MMR*. In fact, a reduction of set $\mathcal{U}$ representing the uncertainty of the user preferences to a new set $\mathcal{W} \subseteq \mathcal{U}$ ensures that the new *MMR* cannot increase.

Lemma 2.3.6. *For any $\mathcal{W} \subseteq \mathcal{U} \subseteq \mathbb{R}^p$:*

(i) $PMR_{\mathcal{W}}(\alpha, \beta) \leq PMR_{\mathcal{U}}(\alpha, \beta)$ *for any* $\alpha, \beta \in \Omega$

(ii) $MR_{\mathcal{W}}(\alpha, A) \leq MR_{\mathcal{U}}(\alpha, A)$ *for any* $\alpha \in \Omega$ *and* $A \in \mathcal{M}$

(iii) $SMR_{\mathcal{W}}(A, B) \leq SMR_{\mathcal{U}}(A, B)$ *for any* $A, B \in \mathcal{M}$

Proof. (i): Since $\mathcal{W} \subseteq \mathcal{U}$, $PMR_{\mathcal{W}}(\alpha, \beta) = \max_{w \in \mathcal{W}}(u_w(\beta) - u_w(\alpha)) \leq \max_{w \in \mathcal{U}}$ $(u_w(\beta) - u_w(\alpha)) = PMR_{\mathcal{U}}(\alpha, \beta)$.

(ii): Since $\mathcal{W} \subseteq \mathcal{U}$, $MR_{\mathcal{W}}(\alpha, \mathrm{A}) = \max_{\beta \in \mathrm{A}} \max_{w \in \mathcal{W}}(u_w(\beta) - u_w(\alpha)) \leq \max_{\beta \in \mathrm{A}}$ $\max_{w \in \mathcal{U}}(u_w(\beta) - u_w(\alpha)) = MR_{\mathcal{U}}(\alpha, \mathrm{A})$.

(iii): Since $\mathcal{W} \subseteq \mathcal{U}$, $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) = \max_{w \in \mathcal{W}}(\mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w)) \leq \max_{w \in \mathcal{U}}$ $(\mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w)) = SMR_{\mathcal{U}}(\mathrm{A}, \mathrm{B})$.

□

The monotonic nature of the Minimax regret criterion with respect to queries for the DM makes this method well-suited for incremental elicitation processes. Furthermore, as we have shown before with Lemma 2.3.2, if the max regret in $\mathcal{W} \subseteq \mathcal{U}$ of an alternative $\alpha \in \mathrm{A}$ equals zero, then α has to be one of the most preferred alternatives in A for the DM according to the preference information represented by $\mathcal{W}$.

To minimise the number of interactions with the DM, we need to carefully choose the queries to reduce the uncertainty as fast as possible. Ideally, evaluating a question at a given iteration should take into account all future questions and possible responses (e.g., [Bou02, HWI03]). However, in practice, this evaluation is often carried out *myopically*. Let $\mathrm{A} \subseteq \mathrm{B}$ be a *query set* defined as a subset of the a finite set $\mathrm{B} \subseteq \Omega$ of available alternatives, and let the DM's response to a query set be her most preferred outcome $\gamma \in \mathrm{A}$. For $\mathcal{W} \subseteq \mathcal{U}$ and $\gamma \in \mathrm{A}$, we define $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$ to be the subset of $w \in \mathcal{W}$ such that $u_w(\gamma) \geq u_w(\alpha)$ for each $\alpha \in \mathrm{A}$. $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$ is then the subset of the DM's utility function prameterisations $\mathcal{W}$ consistent with the user response γ to a query set A. With following definitions we describe the *myopic value of information* and how it can be used to compute an optimal query set in these terms with respect to the Minimax regret criterion [VB09, VB20]:

Max a posteriori regret (*MPR*): Given a finite set of alternatives $\mathrm{B} \subseteq \Omega$, the *MPR* in $\mathcal{W} \subseteq \mathcal{U}$ of a query set $\mathrm{A} = \{\alpha_1, \ldots, \alpha_n\} \subseteq \mathrm{B}$ is defined by:

$$\mathit{MPR}_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) = \max_{\alpha \in \mathrm{A}} \mathit{MMR}_{\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)}(\mathrm{B}). \tag{2.21}$$

MPR is defined to be the myopic value of information of the query $\mathrm{A} \subseteq \mathrm{B}$ in $\mathcal{W}$ with respect to the minimax regret criterion.

Minimax a posteriori regret (*MMPR*) : The *MMPR* in $\mathcal{W}$ of all the possible query sets of cardinality k of a finite set of alternatives $\mathrm{B} \subseteq \Omega$ is defined by:

$$\mathit{MMPR}^{k}_{\mathcal{W}}(\mathrm{B}) = \min_{\mathrm{A} \subseteq \mathrm{B}: |\mathrm{A}| = k} \mathit{MPR}_{\mathcal{W}}(\mathrm{A}, \mathrm{B}). \tag{2.22}$$

A query set A of cardinality k that minimise $\mathit{MPR}_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$ is defined to be myopically optimal with respect to the minimax regret criterion. In Chapter 3 we show that every outcome of a query set A should to be optimal in at least one scenario, i.e., for each $\alpha \in \mathrm{A}$ there should exist $w \in \mathcal{W}$ such that $u_w(\alpha) \geq u_w(\beta)$ for any $\beta \in \mathrm{B} \setminus \{\alpha\}$. Otherwise, a reduction of the uncertainty of the utility function and the consistency of the preference information are not guaranteed.

In [VB09, VB20], the authors show that an optimal recommendation set of size k of a set of alternatives B is also a myopically optimal query set of size k with respect to the minimax regret criterion, and $\mathit{MMPR}^{k}_{\mathcal{W}}(\mathrm{B}) = \mathit{SMMR}^{k}_{\mathcal{W}}(\mathrm{B})$. This makes it compelling to display an optimal set of items with respect to *SMR* with a combined elicitation and recommendation purpose: the system proposes a set of recommended items, the DM picks the one she prefers, then the system updates the model and shows a new set of items; this proceeds until a termination condition (max regret lower than a threshold, or when the user is satisfied) is met. However, setwise regret is computationally very demanding to optimise.

To the best of our knowledge, the state of the art of the algorithms for the computation of the *SMMR* are presented in [VB20]. These algorithms are based on a linear utility model and differ with respect to two classes of alternatives representation, namely, configuration problems and database problem.

Configuration problems: The alternatives are defined by a set of variables and hard constraints where the optimal configuration depends on the DM's

preferences. For example, the set of alternatives may be given by the possible configurations of computer parts for a customised laptop, where each part corresponds to a vector of reals representing, e.g., price and quality. The total value of a laptop could then be the overall price and quality given by the combinations of the corresponding components. The procedure to compute the *SMMR* for this type of problems is encoded with a mixed-integer program and solved by techniques such as Bender's decomposition and constraints generation. Examples of configuration problems in a regret based context can be found in [BPPS06, BB07, BL19].

Database problem: The alternatives are enumerated and represented with an explicit list of multi-attribute outcomes. For example, the set of alternatives may corresponds to a catalogue of laptops already assembled, where each laptop is associated with a vector of reals representing, e.g., price and quality. In this case, the *SMMR* computation is based on the generation of all the possible sets of a specific size k and the corresponding maximum regret. Examples of configuration problems in a regret based context can be found in [BB07, BB10, BPV17b]

Given the complexity of the computation of an optimal query set using the *SMMR* criterion, in [VB09, VB20] have also been proposed heuristics named setwise chain of adversaries strategy (SCAS) and Query iteration strategy (QI) that can be used for both configuration problems and database problems.

Given the complexity of the computation of an optimal query set using the *SMMR* criterion, in [VB20] the authors propose two heuristics, namely, setwise chain of adversaries strategy (SCAS) and Query iteration strategy (QI). These heuristics can be used for both configuration problems and database problems.

2.3.2.3 Minimax Regret with linear utility function

Of particular interest in this thesis is the weighted sum utility function $u_w(\alpha) = \alpha \cdot w$ parameterised with respect to a sets of weights $\mathcal{W} \subseteq \mathcal{U} = \{w \in I\!R^p : w_i \geq 0, \sum_{i=1}^{p} = 1\}$. Given a finite set $\Lambda = \{\lambda_i : i = 1, \ldots, k\}$ of vectors in $I\!R^p$, and corresponding real numbers r_i, we can define $\mathcal{W}_\Lambda$ as the set of $w \in \mathcal{U}$ such that for all $i = 1, \ldots, k$, $w \cdot \lambda_i \geq r_i$. In particular, such linear inequalities can arise from input preferences of the form α *is preferred to* β, leading to the constraint $w \cdot (\alpha - \beta) \geq 0$.

Considering the above model and a set of alternatives A $\in \mathcal{M}$, the max regret

$MR_{\mathcal{W}_\Lambda}(\alpha, \mathrm{A})$ of an alternative $\alpha \in \mathrm{A}$ can be computed as the maximum regret of α with respect to the set $Ext(\mathcal{W}_\Lambda)$ of extreme points of $\mathcal{W}_\Lambda$ (see, e.g., [KVVA17]). The pseudocode is shown in Algorithm 1. The pseudocode of the minimax regret of a set A is shown in Algorithm 2, which is an iterative computation of the max regret for each $\alpha \in \mathrm{A}$.

Algorithm 1 Max Regret

```
1: procedure MR_{W_Λ}(α, A)
2:     MR ← −∞
3:     for γ ∈ A do
4:         for w ∈ W_Λ do
5:             MR ← max((γ − α) · w, MR)
6:     return MR
```

Algorithm 2 Minimax Regret

```
1: procedure MMR_{W_Λ}(A)
2:     MMR ← ∞
3:     for α ∈ A do
4:         min(MR_{W_Λ}(α, A), MMR)
5:     return MMR
```

Regarding the computation of the setwise regret $SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \mathrm{B})$ of a set $\mathrm{A} \in \mathcal{M}$ with $\mathrm{A} \subseteq \mathrm{B}$, a standard method consists of the evaluation of a linear programming problem for each $\beta \in \mathrm{B}$ (see, e.g., [VB20]). Since $SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \{\beta\}) = \max_{w \in \mathcal{W}_\Lambda}(\beta \cdot w - \mathrm{Val}_{\mathrm{A}}(w))$, we can compute $SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \{\beta\})$ as the maximum value δ subject to the constraints $w \in \mathcal{W}_\Lambda$, and $(\beta - \alpha) \cdot w \geq \delta$ for each $\alpha \in \mathrm{A}$. Thus, since $SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \mathrm{B}) = \max_{\beta \in \mathrm{B}} SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \{\beta\})$ (see Lemma 2.3.5), we can compute $SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \mathrm{B})$ solving |B| linear programming problems as shown in Algorithm 3. The setwise minimax regret $SMMR^k_{\mathcal{W}_\Lambda}(\mathrm{B})$ will then be the minimum setwise regret computed for each $\mathrm{A} \in \mathcal{A}^k$, where $\mathcal{A}^k$ is the set of all the subsets of B with cardinality k. The pseudocode for the computation of $SMMR^k_{\mathcal{W}_\Lambda}(\mathrm{B})$ is shown in Algorithm 14 in Section 5.4 of Chapter 5.

In Section 3.6 of Chapter 3 we will present a novel method to compute $SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \mathrm{B})$ and in Chapter 5 a novel method to compute $SMMR^k_{\mathcal{W}_\Lambda}(\mathrm{B})$.

2.4 Relations and Optimality Classes

In this section we define reltaions and optimality classes for alternatives arising from parameterised utility functions.

Algorithm 3 Setwise Max Regret

procedure $SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \mathrm{B})$
  $SMR \leftarrow -\infty$
  for β in B **do**
    $\delta_M \leftarrow$ Maximize δ subject to

$$\left\{ \begin{array}{ll} (\beta - \alpha) \cdot w \geq \delta & \forall \alpha \in \mathrm{A} \\ w \in \mathcal{W}_\Lambda & \end{array} \right\}$$

    if $\delta_M > SMR$ **then**
      $SMR \leftarrow \delta_M$
  return *SMR*

Algorithm 4 Setwise Minimax Regret

procedure $SMMR^k_{\mathcal{W}_\Lambda}(\mathrm{B})$
  $SMMR \leftarrow \infty$
  for $\mathrm{A} \in \mathcal{A}^k$ **do**
    $SMMR \leftarrow \min(SMR_{\mathcal{W}_\Lambda}(\mathrm{A}, \mathrm{B}), SMMR)$
  return *SMMR*

2.4.1 Relations

Recall that $\mathcal{U}$ is the set of parameters of a parameterised utility function representing a set of possible DM's preference models. With the following relation we represent a weak order (see Section 2.1.1) corresponding to a parameterised utility function whose parameter w is known.

Relation $\succcurlyeq_w$: To each parameter $w \in \mathcal{U}$ is associated a utility function $u_w : \Omega \to I\!R$; this gives rise to a total pre-order $\succcurlyeq_w$ on a set of alternatives Ω given by $\alpha \succcurlyeq_w \beta \iff u_w(\alpha) \geq u_w(\beta)$, for $\alpha, \beta \in \Omega$. We define $\succ_w$ to be the strict part of $\succcurlyeq_w$, i.e., $\alpha \succ_w \beta$ if and only if $\alpha \succcurlyeq_w \beta$ and $\neg(\beta \succcurlyeq_w \alpha)$, which is if and only if $u_w(\alpha) > u_w(\beta)$. We define equivalence relation $\equiv_w$ to be the symmetric part of $\succcurlyeq_w$, given by $\alpha \equiv_w \beta$ if and only if $\alpha \succcurlyeq_w \beta$ and $\beta \succcurlyeq_w \alpha$, which is if and only if $u_w(\alpha) = u_w(\beta)$.

The following relation instead is derived from utility functions parameterised with respect to a set of parameter $\mathcal{W} \subseteq \mathcal{U}$ compatible with the available preference information. The main difference with respect to $\succcurlyeq_w$ is that it does not define a total preorder since the property of completeness does not hold, i.e., there could exist $\alpha, \beta \in \Omega$ such that $\alpha \not\succcurlyeq_{\mathcal{W}} \beta$ and $\beta \not\succcurlyeq_{\mathcal{W}} \alpha$.

Relation $\succcurlyeq_{\mathcal{W}}$: For $\mathcal{W} \subseteq \mathcal{U}$ we define relation $\succcurlyeq_{\mathcal{W}}$ on Ω by $\alpha \succcurlyeq_{\mathcal{W}} \beta$ if and only if for all $w \in \mathcal{W}$, $\alpha \succcurlyeq_w \beta$. Thus, $\alpha \succcurlyeq_{\mathcal{W}} \beta$ if and only if α is at least as good as β in every parameter in $\mathcal{W}$, i.e., $u_w(\alpha) \geq u_w(\beta)$ for any $w \in \mathcal{W}$. We define $\succ_{\mathcal{W}}$ to be the strict part of $\succcurlyeq_{\mathcal{W}}$, i.e., for $\alpha, \beta \in \Omega$, $\alpha \succ_{\mathcal{W}} \beta$ if and only if $\alpha \succcurlyeq_{\mathcal{W}} \beta$ and $\beta \not\succcurlyeq_{\mathcal{W}} \alpha$, which is if and only if $u_w(\alpha) \geq u_w(\beta)$ for any $w \in \mathcal{W}$ and there exists $w' \in \mathcal{W}$ such that $u_{w'}(\alpha) > u_{w'}(\beta)$. Thus, $\alpha \succ_{\mathcal{W}} \beta$ if and only if α is at least as good as β in every parameter in $\mathcal{W}$, and strictly better in at least one parameter in $\mathcal{W}$. Relation $\succ_{\mathcal{W}}$ is transitive and acyclic. We define equivalence relation $\equiv_{\mathcal{W}}$ to be the symmetric part of $\succcurlyeq_{\mathcal{W}}$, given by $\alpha \equiv_{\mathcal{W}} \beta$ if and only if $\alpha \succcurlyeq_{\mathcal{W}} \beta$ and $\beta \succcurlyeq_{\mathcal{W}} \alpha$, i.e., $u_w(\alpha) = u_w(\beta)$ for any $w \in \mathcal{W}$.

The following are definitions of dominance corresponding to the parameterised preference relation defined above.

Weakly dominance: We say that an alternative $\alpha \in \Omega$ *weakly dominates* an alternative $\beta \in \Omega$ with respect to $\mathcal{W} \subseteq \mathcal{U}$ if and only if $\alpha \succcurlyeq_{\mathcal{W}} \beta$.

Dominance: We say that an alternative $\alpha \in \Omega$ *dominates* an alternative $\beta \in \Omega$ with respect to $\mathcal{W} \subseteq \mathcal{U}$ if and only if $\alpha \succ_{\mathcal{W}} \beta$.

Equivalence: We say that an alternative $\alpha \in \Omega$ *is equivalent to* an alternative $\beta \in \Omega$ with respect to $\mathcal{W} \subseteq \mathcal{U}$ if and only if $\alpha \equiv_{\mathcal{W}} \beta$.

In Chapter 3 we will introduce relations and definitions of dominance with respect to set of alternatives rather than single alternatives.

2.4.2 Optimality classes

Over the years, parameterised utility functions led to different notion of optimality that can be used to classify alternatives (see, e.g., [WO11]). The following is a summary of those of interest with respect to this thesis. Let A be a finite set of alternatives.

Optimal alternative: An alternative $\alpha \in$ A is defined to be *optimal* with respect to a specific parameter $w \in \mathcal{U}$ if and only if it maximises the corresponding utility function u_w, i.e., $u_w(\alpha) \geq u_w(\beta)$ for any $\beta \in$ A.

Undominated alternative: An alternative $\alpha \in \mathrm{A}$ is defined to be *undominated* (or *nondominated*) with respect to a feasible set of parameters $\mathcal{W} \subseteq \mathcal{U}$ if and only if there does not exists $\beta \in \mathrm{A}$ that dominates α, i.e., $\beta \not\succ_{\mathcal{W}} \alpha$ for any $\beta \in \mathrm{A}$.

The Undominated operator $\mathrm{UD}_{\mathcal{W}}$: For $\mathcal{W} \subseteq \mathcal{U}$ we define $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ to be the set of $\alpha \in \mathrm{A}$ such that there does not exist $\gamma \in \mathrm{A}$ such that $\gamma \succ_{\mathcal{W}} \alpha$. Thus, the alternative α of A is not in $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ if and only if there exists some $\gamma \in \mathrm{A}$ such that γ is at least as good as α in every scenario, and strictly better in at least one scenario. The set $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ is a natural generalisation of the Pareto-optimal elements, and is sometimes referred to as the set of *undominated* elements in A.

Possibly optimal alternative: An alternative $\alpha \in \mathrm{A}$ is defined to be *possibly optimal* with respect to a feasible set of parameters $\mathcal{W} \subseteq \mathcal{U}$ if and only if there exists $w \in \mathcal{W}$ in which α is optimal, i.e., $u_w(\alpha) \geq u_w(\beta)$ for all $\beta \in \mathrm{A}$.

Possibly Optimal operator $\mathrm{PO}_{\mathcal{W}}(\mathbf{A})$: for each $w \in \mathcal{U}$ we define $\mathrm{O}_w(\mathrm{A})$ to be all alternatives α of A that are optimal in A in scenario w, i.e., such that for all $\beta \in \mathrm{A}$, $\alpha \succcurlyeq_w \beta$. For $\mathcal{W} \subseteq \mathcal{U}$ we define $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$ to be $\bigcup_{w \in \mathcal{W}} \mathrm{O}_w(\mathrm{A})$, the set of alternatives that are optimal in some parameter, i.e., optimal for some consistent user preference model.

Possibly strictly optimal alternative: An alternative $\alpha \in \mathrm{A}$ is defined to be *possibly strictly optimal* with respect to a feasible set of parameters $\mathcal{W} \subseteq \mathcal{U}$ if and only if there exists $w \in \mathcal{W}$ in which α is strictly optimal, i.e., $u_w(\alpha) > u_w(\beta)$ for all $\beta \in \mathrm{A}$.

Possibly Strictly Optimal operator $\mathrm{PSO}_{\mathcal{W}}(\mathbf{A})$: For each $w \in \mathcal{W}$ with $\mathcal{W} \subseteq \mathcal{U}$, we define $\mathrm{SO}_w^{\mathcal{W}}(\mathrm{A})$ to be all alternatives α of A such that $\alpha \succ_w \beta$, for all $\beta \in \mathrm{A}$ with $\beta \not\equiv_{\mathcal{W}} \alpha$. These alternative α are said to be *strictly optimal in parameter* w. We define $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, the set of *possibly strictly optimal* elements, to be $\bigcup_{w \in \mathcal{W}} \mathrm{SO}_w^{\mathcal{W}}(\mathrm{A})$, i.e., all the elements that are strictly optimal for some parameter in $\mathcal{W}$. If there do not exists $\alpha, \beta \in \mathrm{A}$ such that $\beta \equiv_{\mathcal{W}} \alpha$, then $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ consists of all alternatives $\alpha \in \mathrm{A}$ which are uniquely optimal in some parameter $w \in \mathcal{W}$ (i.e., $\mathrm{O}_w(\mathrm{A}) = \{\alpha\}$).

Necessarily optimal alternative: An alternative $\alpha \in \mathrm{A}$ is defined to be *necessarily optimal* with respect to a feasible set of parameters $\mathcal{W} \subseteq \mathcal{U}$ if and only if it is optimal for any $w \in \mathcal{W}$, i.e., $\alpha \succcurlyeq_{\mathcal{W}} \beta$ for any $\beta \in \mathrm{A}$.

Necessarily Optimal operator $\mathrm{NO}_{\mathcal{W}}(\mathbf{A})$: for each $w \in \mathcal{U}$ and A $\in \mathcal{M}$ we define $\mathrm{NO}_{\mathcal{W}}$(A), the set of *necessarily optimal* elements, to be $\bigcap_{w\in\mathcal{W}} \mathrm{O}_w$(A), i.e., all the elements that are optimal in at least one scenario $w \in \mathcal{W}$. If there do not exists $\alpha, \beta \in$ A such that $\beta \equiv_{\mathcal{W}} \alpha$ and $\mathrm{NO}_{\mathcal{W}}(\mathrm{A}) \neq \emptyset$, then $\mathrm{NO}_{\mathcal{W}}$(A) is a singleton.

The set of undominated alternatives $\mathrm{UD}_{\mathcal{W}}$(A) is a natural class of optimality when considering uncertain preference models and it appears in numerous contexts (see, e.g., [WSD84, MRW13, KVVA17]). In [Haz86, Web87] we can find the firsts attempt to define a different class of alternatives with respect to a non singleton set of parameters of a utility function; the authors noted that for $\alpha \in \mathrm{UD}_{\mathcal{W}}$(A) there could exists a subset B of A such that for all $w \in \mathcal{W}$ there exists $\beta \in$ B such that $u_w(\beta) > u_w(\alpha)$ which means that an undominated alternative may not be possibly optimal. Later, possibly optimal alternatives $\mathrm{PO}_{\mathcal{W}}$(A) (also known as *potentially optimal*) have been considered in many other publications, such as [AP97, GPR$^+$10, WRM15, BP15a, BP17] becoming a fundamental concept of MAUT. The possibly strictly optimal set $\mathrm{PSO}_{\mathcal{W}}$(A) and the necessarily optimal set $\mathrm{NO}_{\mathcal{W}}$(A) instead has been considered much less (see, e.g, [WO11, OW13]). However, as we will show in Chapter 3, alternatives composing a choice query should be possibly strictly optimal. Otherwise, there is the risk of modelling the DM's preference with respect to events with zero probability. Necessarily optimal alternatives instead can be used as stopping criteria for an iterative preference learning approach. This is because a necessarily optimal alternative maximises a parameterised utility function for any admissible parameter in $\mathcal{W}$. The concepts of possibly optimal alternatives and necessarily optimal alternatives have also been considered in voting problems [KL05, XC11]. In this context, several preference profiles concerning a set of candidates are considered. If a candidate wins for at least one preference profile, then it is defined as a *possible winner*; if a candidate wins for every feasible preference profile, it is defined as a *necessary winner*.

Example 6. *Consider the Figure 2.3 showing the utility function $u_w(\cdot)$ of the alternatives of the sets $A' = \{(11,1),(7,5),(6,6)\}$ and $A'' = \{(10,4),(4,7)\}$, with $w \in \mathcal{U} = \{(w_1,w_2) : w_1 + w_2 = 1, w_1 \geq 0\}$. Let $A = A' \cup A''$. Suppose that we get as input the preference information $w_1 \leq \frac{2}{3}$. This leads to a new set of feasible parameters $\mathcal{W} = \{(w_1,w_2) : w_1 + w_2 = 1, 0 \leq w_1 \leq \frac{2}{3}\}$ (white background). The set of undominated alternative in $\mathcal{W}$ is then $\mathrm{UD}_{\mathcal{W}}(A) = \{(10,4),(7,5),(6,6),(4,7)\}$. From Figure 2.3 we can easily see that $\mathrm{UD}_{\mathcal{W}}(A)$ does not contain the alternative $(11,1)$ since the alternative $(10,4)$ has better utility for every parameter $w \in \mathcal{W}$. Abbreviating w to just its first component w_1 we can*

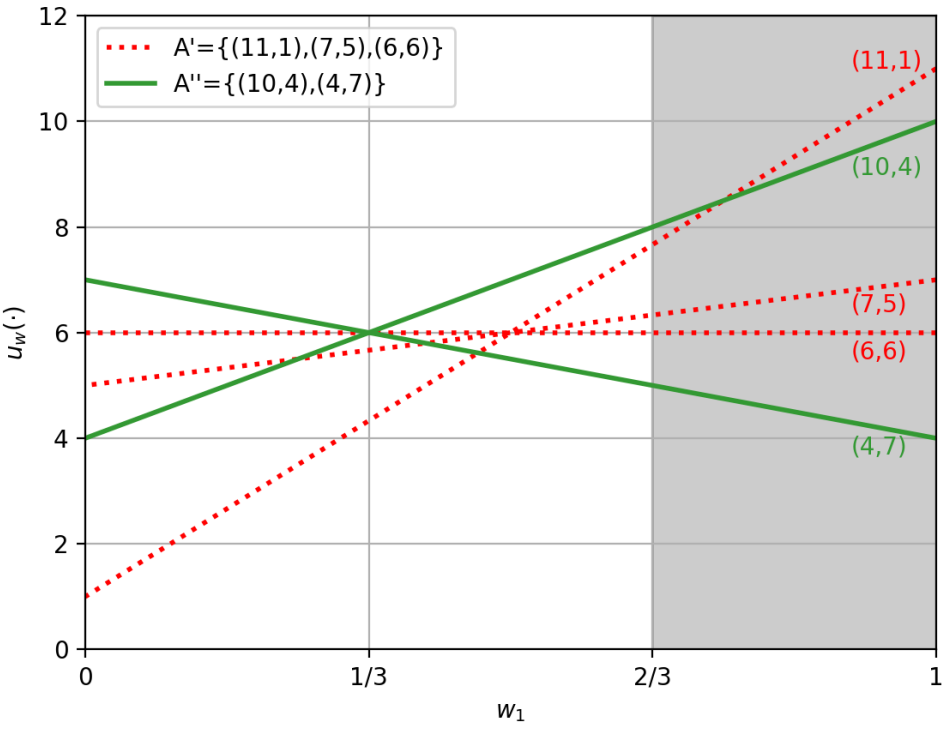


Figure 2.3: Utility function $u_w(\cdot)$ for each alternative in $\mathrm{A} = \mathrm{A}' \cup \mathrm{A}'' = \{(11,1), (7,5), (6,6), (10,4), (4,7)\}$, where $w \in \mathcal{U} = \{(w_1, w_2) : w_1 + w_2 = 1, w_1 \geq 0\}$.

represent the set of possible parameters as $\mathcal{W} = [0, \frac{2}{3}]$. *Let* $\mathrm{Opt}^A_{\mathcal{W}}(\alpha)$ *be the set of parameters in* $\mathcal{W}$ *in which alternative* α *is optimal.* $\mathrm{Opt}^A_{\mathcal{W}}((10,4)) = [\frac{1}{3}, \frac{2}{3}]$; $\mathrm{Opt}^A_{\mathcal{W}}((6,6)) = \{\frac{1}{3}\}$, $\mathrm{Opt}^A_{\mathcal{W}}((4,7)) = [0, \frac{1}{3}]$ *and* $\mathrm{Opt}^A_{\mathcal{W}}(11,1) = \mathrm{Opt}^A_{\mathcal{W}}(7,5) = \emptyset$. *Thus,* $\mathrm{PO}_{\mathcal{W}}(A) = \{(10,4), (6,6), (4,7)\}$ *since these alternatives have the best utility for at least one parameter* $w \in \mathcal{W}$, *and* $\mathrm{PSO}_{\mathcal{W}}(A) = \{(10,4), (4,7)\}$ *since these alternatives have strictly better utility with respect to the remaining alternatives in A for at least one parameter* $w \in \mathcal{W}$. *The* $\mathrm{PSO}_{\mathcal{W}}$ *operator thus leads here to stronger filtering than the* $\mathrm{PO}_{\mathcal{W}}$ *operator since* $(6,6)$ *is optimal only in* $w = \frac{1}{3}$ *along with* $(10,4)$ *and* $(4,7)$.

2.5 Conclusions

In this chapter, we summarised the main related works in the topics of interest to our research, especially in the area of multi-attribute utility theory and preference elicitation. We also introduced parameterised utility functions representing different scenarios of the DM's preferences. Parameterised utility function in a MAUT setting will be the type of preference model considered in

this thesis.

We have defined in details the minimax regret criterion and its setwise generalisation to evaluate alternatives with parameterised value functions, and we have shown how max regret methods can be used in a preference elicitation context. The minimax regret criterion will be considered in all the chapters of this thesis: in chapter 3 we define a new method for the computation of the setwise max regret, and in chapter 4 we will define a new method for the computation of the setwise minimax regret.

We concluded the chapter defining relations and optimality classes for alternatives evaluated with parametrised utility functions, which will be generalised to evaluate set of alternatives in the next chapter.

Chapter 3

Minimality and Comparison of Sets of Multi-Attribute Vectors

In this chapter we provide definitions and prove general properties of preference relations for sets of alternatives evaluated with uncertain utility functions. We define the concept of setwise minimal equivalent set with respect to generic parametrised utility functions, in particular, we show that for important classes of preference models, the set of possibly strictly optimal alternatives is the unique minimal equivalent subset. We also discuss potential issues of sets of alternatives used as query sets for incremental preference elicitation methods, and how to avoid them. We derive mathematical results that allow different computational techniques to evaluate relations and to compute the setwise minimal equivalent subset. We focus especially on alternatives represented as multi-attribute utility vectors, with a user preference model based on the weighted sum utility function. The main computational procedures presented in this chapter are based on linear programming (LP), or, alternatively, a novel method using the extreme points of the epigraph of the utility function (which we abbreviate to EEU). These approaches can be used to compute both the setwise minimax regret and the set of strictly possibly optimal elements supposing a weighted sum utility function. We validate our methods with some experimental results showing that EEU outperforms LP up to a certain number of criteria used to evaluate alternatives.

3.1 Introduction

Let $\mathcal{W}$ be a set of scenarios, where, associated with each scenario $w \in \mathcal{W}$, is a utility function u_w over alternatives. As we discussed in 2.3.2, given a finite set of alternatives A, the best alternative of a decision-maker supposing that w were the true scenario lead to a utility function $\text{Val}_\text{A}(w) = \max_{\alpha \in \text{A}} u_w(\alpha)$. In this chapter we consider the following related pair of questions raised in the introduction of this thesis:

(1) Are there elements of A that can be eliminated unproblematically? In particular, is there a strict subset A′ of A that is equivalent to A?

(2) Given a choice between one situation, in which the available alternatives are A, and another situation, in which alternatives B are available, is A at least as good as B in every scenario?

Regarding (1), we need to be able to eliminate unimportant choices, to make the list of options manageable, in particular, if we want to display the alternatives to the user. We interpret this as finding a minimal subset A′ of A such that $\text{Val}_\text{A}(w) = \text{Val}_{\text{A}'}(w)$ for every scenario $w \in \mathcal{W}$.

Question (2) concerns a case in which the user may have a choice between (I) being able to obtain any of the set of alternatives A, and (II) any alternative in B (and thus, the user could obtain any alternative in $\text{A} \cup \text{B}$). Sets A and B may correspond to different choices $Y = a$ and $Y = b$ of a fundamental variable Y, and determining that A dominates B may lead us to exclude $Y = b$, thus simplifying the problem. For instance, A may correspond to hotels in Barcelona, and B to hotels in Valencia, for a potential weekend away. We want to be able to determine if one of these clearly dominates the other; if, for instance, A dominates B, then there may be no need for the system and the user to further consider B, and, for example, may focus on Barcelona rather than Valencia. We interpret this task as determining if in every scenario the utility A is at least that for B, i.e., $\text{Val}_\text{A}(w) \geq \text{Val}_\text{B}(w)$ for all $w \in \mathcal{W}$.

The rest of the chapter is organised as follows. Section 3.2 defines preference relations for sets of alternatives along with some basic properties. Section 3.3 considers the problem of reducing the size of a set A, whilst maintaining equivalence. Section 3.4 defines a form of maximum regret in this context, shows how it relates to dominance, and gives properties that will be useful for computation. Section 3.5 discusses the importance of the possibly optimal and possibly

strictly optimal alternatives in incremental preference elicitation. Section 3.6 describes the EEU method. Section 3.7 brings together the computational techniques for the weighted sum utility function. Sections 3.8 and 3.9 describe the implementation and experimental testing, and Section 3.10 concludes.

3.2 Preference Relations for Set of Alternatives

Based on a set $\mathcal{W}$ of scenarios, and the corresponding set of relations $\succcurlyeq_{\mathcal{W}}$, and for $w \in \mathcal{W}$, $\succcurlyeq_w$ (see section 2.4.1), we will consider different relations on $\mathcal{M}$, the set of finite subsets of Ω.

Dominance relation between sets: For subset $\mathcal{W}$ of $\mathcal{U}$, we define binary relation $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ on $\mathcal{M}$ as follows. Consider any $\mathrm{A}, \mathrm{B} \in \mathcal{M}$.

- $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ if and only if for all $w \in \mathcal{W}$ and for all $\beta \in \mathrm{B}$ there exists $\alpha \in \mathrm{A}$ such that $\alpha \succcurlyeq_w \beta$. Since each relation $\succcurlyeq_w$ is a total pre-order, and A is finite (as is B), we have $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ if and only if if for each scenario $w \in \mathcal{W}$, there exists an alternative in A that weakly dominates all the alternatives in B. We define $\equiv^{\mathcal{W}}_{\forall\forall\exists}$ to be the symmetric part of $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$, with $\mathrm{A} \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ if and only if $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ and $\mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$.

One can also consider a (strong form of) strict dominance $\mathrm{A} \gg^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ defined as for all $w \in \mathcal{W}$, $\mathrm{Val}_{\mathrm{A}}(w) > \mathrm{Val}_{\mathrm{B}}(w)$; this corresponds with the dominance relation defined in Definition 2 of [BP15b].

Relation $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ and its corresponding symmetric part $\equiv^{\mathcal{W}}_{\forall\forall\exists}$ are the main foci of attention in this chapter. However, for computational reasons we consider two variations, which allow computationally efficient sufficient conditions for $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$. We define $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ and $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ on $\mathcal{M}$ as follows.

- $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall} \mathrm{B}$ if and only if for all $\beta \in \mathrm{B}$ there exists $\alpha \in \mathrm{A}$ such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$ (i.e., for all $w \in \mathcal{W}$, $\alpha \succcurlyeq_w \beta$). Thus, $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall} \mathrm{B}$ if and only if for all the alternatives $\beta \in \mathrm{B}$ there exists at least one alternative $\alpha \in \mathrm{A}$ that weakly dominates β in all the possible scenarios $\mathcal{W}$. We define $\equiv^{\mathcal{W}}_{\forall\exists\forall}$ to be the symmetric part of $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$, with $\mathrm{A} \equiv^{\mathcal{W}}_{\forall\exists\forall} \mathrm{B}$ if and only if $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall} \mathrm{B}$ and $\mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall} \mathrm{A}$.
- $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall\forall} \mathrm{B}$ if and only if there exists $\alpha \in \mathrm{A}$ such that for all $\beta \in \mathrm{B}$, $\alpha \succcurlyeq_{\mathcal{W}} \beta$ which is if and only if there exists $\alpha \in \mathrm{A}$ such that for all $w \in \mathcal{W}$ $u_w(\alpha) \geq \mathrm{Val}_{\mathrm{B}}(w)$. Thus, $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall\forall} \mathrm{B}$ if and only if there exists at least one

alternative in A that weakly dominates all the alternatives in B in all the possible scenarios $\mathcal{W}$. We define $\equiv^{\mathcal{W}}_{\exists\forall\forall}$ to be the symmetric part of $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$, with A $\equiv^{\mathcal{W}}_{\exists\forall\forall}$ B if and only if A $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ B and B $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ A.

Example 7. *Consider the sets of utility vectors* $A = \{(10,4),(4,7)\}$, $B = \{(11,2),(8,5)\}$ *and* $C = \{(11,1),(7,5))\}$ *associated with hotels in Barcelona, Valencia and Málaga respectively. Suppose that the first value of each utility vector is a score evaluating the distance from the beach and the second value is a score evaluating the distance from the city center, where the higher the score, the better. We assume a linear utility functions with* $u_w(\alpha) = w_1\alpha_1 + w_2\alpha_2$, *where* $w \in \mathcal{U} = \{(w_1,w_2) : w_1,w_2 \geq 0 \;\&\; w_1+w_2=1\}$ *represents the possible preference scenarios and* $\alpha = (\alpha_1,\alpha_2)$ *is the utility vector of an apartment. We assume that the user has an associated (unknown) weights vector* w^* *and we want to recommend to the user a trip to Barcelona or Valencia or Málaga based on some input preference information. Suppose then that we ask the user her preference between the hotel with utility* $(10,4)$ *and the hotel with utility* $(11,2)$. *An input preference of* $(10,4)$ *over* $(11,2)$ *implies* $w\cdot(10,4) \geq w\cdot(11,2)$ *and so* $2w_2 \geq w_1$ *and thus,* $w_1 \leq \frac{2}{3}$. *This leads to the new set of preference scenarios* $\mathcal{W} = \{(w_1,w_2) : w_1+w_2=1 \;\&\; 0 \leq w_1 \leq \frac{2}{3}\}$. *This example is illustrated in Figure 3.1, and supposing this input preference information, it is easy to see that in this case A* $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ *B and A* $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ *C since for* $0 \leq w_1 \leq \frac{1}{3}$ *there is no line above the line associated to* $(4,7) \in$ *A, and for* $\frac{1}{3} \leq w_1 \leq \frac{2}{3}$ *there is no line above the line associated to* $(10,4) \in$ *A, i.e.,* $\nexists\beta \in$ *B s.t.* $u_w(\beta) > Val_A(w)$ *and* $\nexists\gamma \in$ *C s.t.* $u_w(\gamma) > Val_A(w)$ *for any* $w \in \mathcal{W}$. *Therefore, the optimal recommendation would be a trip to Barcelona since it has an optimal hotel for each preference scenario. Note that, A* $\not\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ *B, A* $\not\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ *B, A* $\not\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ *C and A* $\not\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ *C. This can be easily seen from Figure 3.1, since for each alternative of A there exists at least one scenario* $w \in \mathcal{W}$ *in which at least one alternative of B and C have better utility values* $u_w(\cdot)$. *On the other hand, B* $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ *C, since for each alternative of C there is a better alternative in B for all the possible scenarios in* $\mathcal{W}$. *Reducing the set* $\mathcal{W}$ *to* $\mathcal{W}' = \{(w_1,w_2) : w_1+w_2=1 \;\&\; 0 \leq w_1 \leq \frac{1}{3}\}$, *we also get that B* $\succcurlyeq^{\mathcal{W}'}_{\exists\forall\forall}$ *C since* $(8,5) \in$ *B is the best alternative with respect to B and C for all the possible scenarios in* $\mathcal{W}'$.

We now give some properties of the relations defined above.

The following result shows that the relation $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ and its corresponding equivalence relation $\equiv^{\mathcal{W}}_{\forall\forall\exists}$ can be expressed in terms of the utility function. Part (iii) gives another representation of the relation $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ that allows efficient

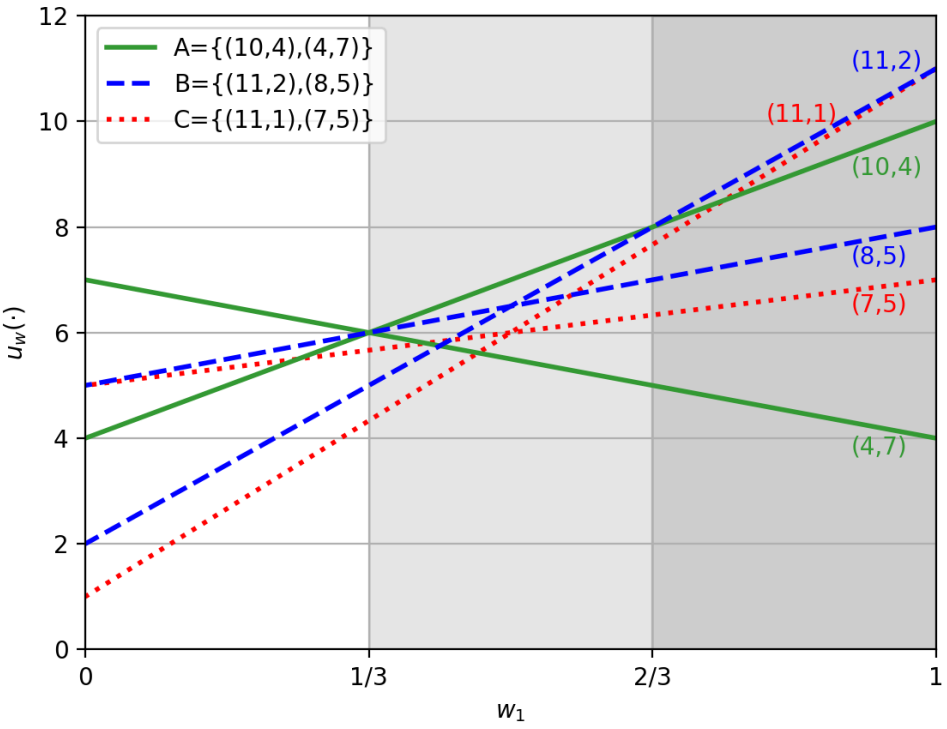


Figure 3.1: utility function $u_w(\alpha) = w \cdot \alpha$ for each alternative of the sets, A $= \{(10,4),\ (4,7)\}$, B $= \{(11,2),(8,5)\}$ and C $= \{(11,1),(7,5))\}$ where $w \in \mathcal{U} = \{(w_1, w_2) : w_1, w_2 \geq 0\ \&\ w_1 + w_2 = 1\}$.

computation.

Lemma 3.2.1. *Consider any $\mathcal{W} \subseteq \mathcal{U}$ and $A, B \in \mathcal{M}$.*

(i) $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \iff$ *for all* $w \in \mathcal{W}$, $Val_A(w) \geq Val_B(w)$.

(ii) $A \equiv^{\mathcal{W}}_{\forall\exists} B \iff$ *for all* $w \in \mathcal{W}$, $Val_A(w) = Val_B(w)$.

(iii) $A \succcurlyeq^{\mathcal{W}}_{\exists\forall} B$ *if and only if there exists* $\alpha \in A$ *such that for all* $w \in \mathcal{W}$, $u_w(\alpha) \geq Val_B(w)$.

Proof: (i): For all $w \in \mathcal{W}$, $\text{Val}_\text{A}(w) \geq \text{Val}_\text{B}(w)$, if and only if for all $w \in \mathcal{W}$, $\max_{\beta\in\text{B}} u_w(\beta) \leq \max_{\alpha\in\text{A}} u_w(\alpha)$, which is if and only if for all $w \in \mathcal{W}$, for all $\beta \in$ B, $u_w(\beta) \leq \max_{\alpha\in\text{A}} u_w(\alpha)$. This holds if and only if for all $w \in \mathcal{W}$ and $\beta \in$ B there exists $\alpha \in$ A such that $u_w(\alpha) \geq u_w(\beta)$, which is if and only if A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B.

(ii): A $\equiv^{\mathcal{W}}_{\forall\exists}$ B holds if and only if A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B and B $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ A, which by (i) is if and only if for all $w \in \mathcal{W}$, $\text{Val}_\text{B}(w) \leq \text{Val}_\text{A}(w)$ and $\text{Val}_\text{B}(w) \geq \text{Val}_\text{A}(w)$, which is if and only if $\text{Val}^{\mathcal{W}}_\text{B} = \text{Val}^{\mathcal{W}}_\text{A}$.

(iii): For $\alpha \in \Omega$, $\{\alpha\} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{B} \iff \{\alpha\} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \text{B}$. Then, $\text{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{B}$ if and only if there exists $\alpha \in \text{A}$ such that $\{\alpha\} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{B}$, which is if and only if there exists $\alpha \in \text{A}$ such that for all $w \in \mathcal{W}$, $u_w(\alpha) \geq \text{Val}_\text{B}(w)$, using part (i). □

Lemma 3.2.2. *Consider any $\mathcal{W} \subseteq \mathcal{U}$ and $A, B \in \mathcal{M}$. $\succcurlyeq^{\mathcal{W}}_{\exists\forall} \subseteq \succcurlyeq^{\mathcal{W}}_{\forall\exists} \subseteq \succcurlyeq^{\mathcal{W}}_{\forall\forall}$, i.e.:*

(i) $A \succcurlyeq^{\mathcal{W}}_{\exists\forall} B \implies A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$

(ii) $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \implies A \succcurlyeq^{\mathcal{W}}_{\forall\forall} B$

Proof. (i): If $\text{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{B}$ then there exists $\alpha' \in \text{A}$ such that for all $\beta \in \text{B}$, $\alpha' \succcurlyeq_{\mathcal{W}} \beta$. Thus, $\text{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \text{B}$ since for all $\beta \in \text{B}$ there exists $\alpha \in \text{A}$, i.e., α', such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$.

(ii): If $\text{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \text{B}$ then for all $\beta \in \text{B}$ there exists $\alpha' \in \text{A}$ such that $\alpha' \succcurlyeq_{\mathcal{W}} \beta$. Thus, $\text{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall} \text{B}$ since for all $w \in \mathcal{W}$ and for all $\beta \in \text{B}$ there exists $\alpha \in \text{A}$, i.e., α', such that $\alpha \succcurlyeq_w \beta$.

□

The following result, states transitivity and chaining properties of the three relations. These are valuable, for instance, if we are comparing a number of sets A_i, $i = 1, \ldots, K$, since if we determine that $\text{A}_i \succcurlyeq^{\mathcal{W}}_{\forall\forall} \text{A}_j$ and $\text{A}_j \succcurlyeq^{\mathcal{W}}_{\forall\forall} \text{A}_k$, then we do not need to check that $\text{A}_i \succcurlyeq^{\mathcal{W}}_{\forall\forall} \text{A}_k$, since it is implied.

Lemma 3.2.3. *Each of the relations, $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$, $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ and $\succcurlyeq^{\mathcal{W}}_{\exists\forall}$ on $\mathcal{M}$, is transitive. Furthermore, we have the following chaining properties:*

(i) If $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$ and $B \succcurlyeq^{\mathcal{W}}_{\exists\forall} C$ then $A \succcurlyeq^{\mathcal{W}}_{\exists\forall} C$.

(ii) If $\succcurlyeq$ is any of the relations $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$, $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ and $\succcurlyeq^{\mathcal{W}}_{\exists\forall}$, then $A \succcurlyeq^{\mathcal{W}}_{\exists\forall} B$ and $B \succcurlyeq C$ implies $A \succcurlyeq^{\mathcal{W}}_{\exists\forall} C$.

Proof. (i): Assume that $\text{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \text{B}$ and $\text{B} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{C}$. Since $\text{B} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{C}$, there exists an alternative β in B such that $\beta \succcurlyeq_{\mathcal{W}} \gamma$ for all $\gamma \in \text{C}$. Such β exists since $\text{B} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{C}$. Since $\text{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \text{B}$, there exists $\alpha \in \text{A}$ such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$. Since $\succcurlyeq_{\mathcal{W}}$ is transitive on Ω and $\alpha \succcurlyeq_{\mathcal{W}} \beta$ and $\beta \succcurlyeq_{\mathcal{W}} \gamma$ for all $\gamma \in \text{C}$, then $\alpha \succcurlyeq_{\mathcal{W}} \gamma$ for all $\gamma \in \text{C}$, which implies $\text{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{C}$.

(ii): From Lemma 3.2.2 it follows that $\text{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \text{C} \implies \text{B} \succcurlyeq^{\mathcal{W}}_{\forall\forall} \text{C}$ and $\text{B} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{C} \implies \text{B} \succcurlyeq^{\mathcal{W}}_{\forall\forall} \text{C}$. Thus, if (ii) is true when $\succcurlyeq$ equals $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$, then it is true also for the other two cases. We then need to prove that $\text{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{B}$ and $\text{B} \succcurlyeq^{\mathcal{W}}_{\forall\forall} \text{C}$ implies $\text{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{C}$: since there exists $\alpha \in \text{A}$ such that for all $\beta \in \text{B}$ $\alpha \succcurlyeq_{\mathcal{W}} \beta$ ($\text{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \text{B}$), and for all $\gamma \in \text{C}$ there exists $\beta \in \text{B}$ such that $\beta \succcurlyeq_{\mathcal{W}} \gamma$

(B $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ C), then, by the transitivity of $\succcurlyeq_{\mathcal{W}}$, there exists $\alpha \in$ A such that such that $\alpha \succcurlyeq_{\mathcal{W}} \gamma$ for all $\gamma \in$ C, i.e., A $\succcurlyeq^{\mathcal{W}}_{\exists\forall}$ C.

Transitivity of $\succcurlyeq^{\mathcal{W}}_{\exists\forall}$ on $\mathcal{M}$ is implied by (ii).

Transitivity of $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ on $\mathcal{M}$: we have transitivity of $\succcurlyeq^{\{w\}}_{\exists\forall}$ on $\mathcal{M}$, which is the same as $\succcurlyeq^{\{w\}}_{\forall\exists}$ on $\mathcal{M}$, and thus transitivity of $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ on $\mathcal{M}$, as the intersection of an arbitrary set of transitive relations is transitive.

Transitivity of $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ on $\mathcal{M}$: Suppose If A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B and B $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ C and consider any $\gamma \in$ C. Because B $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ C, there exists $\beta \in$ B such $\beta \succcurlyeq_{\mathcal{W}} \gamma$. A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B implies there exists $\alpha \in$ A such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$, and thus for the transitivity of $\succcurlyeq_{\mathcal{W}}$, $\alpha \succcurlyeq_{\mathcal{W}} \gamma$ for all $\gamma \in$ C, proving that A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ C.

□

For $\succcurlyeq$ being either $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ or $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$, to determine if A $\succcurlyeq$ B it is sufficient to check that A $\succcurlyeq \{\beta\}$ holds for each $\beta \in$ B. In more detail: we say that relation $\succcurlyeq$ on $\mathcal{M}$ satisfies the Right Decomposition property if A $\succcurlyeq$ B if and only if A $\succcurlyeq \{\beta\}$ holds for each $\beta \in$ B. $\succcurlyeq$ is *reflexive* if for all A $\subseteq \Omega$, A $\succcurlyeq$ A. As well as being reflexive, relations $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ and $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ satisfy the Right Decomposition property, which is useful computationally, since it means that, for $\succcurlyeq$ being either $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ or $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$, to determine if A $\succcurlyeq$ B it is sufficient to check that A $\succcurlyeq \{\beta\}$ holds for each $\beta \in$ B.

Lemma 3.2.4. *For any $\mathcal{W} \subseteq \mathcal{U}$. Let $\succcurlyeq^{\mathcal{W}}$ be any of $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$, $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ and $\succcurlyeq^{\mathcal{W}}_{\exists\forall}$, and let $A', B \in \mathcal{M}$ and let $A \subseteq A'$ and let $B' \subseteq B$, and let $\mathcal{W}' \subseteq \mathcal{W}$.*

(i) If $A \succcurlyeq^{\mathcal{W}} B$ then $A' \succcurlyeq^{\mathcal{W}'} B'$.

(ii) Relations $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ and $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ satisfy Right Decomposition.

(iii) Relations $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ and $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ are reflexive.

(iv) If $A \supseteq B$ then $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$ and $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$.

Proof. (i): since A′ $\supseteq$ A then for any $\alpha \in$ A, $\alpha \in$ A′.

Assume A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B, and consider any $w \in \mathcal{W}'$ and $\beta \in$ B′; since $\mathcal{W}' \subseteq \mathcal{W}$ and B′ $\subseteq$ B, we have $w \in \mathcal{W}$ and $\beta \in$ B, and thus, since A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B, there exists $\alpha \in$ A such that $\alpha \succcurlyeq_w \beta$. This proves that A′ $\succcurlyeq^{\mathcal{W}'}_{\forall\exists}$ B′

Assume A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B, and consider any $\beta \in$ B′; since B′ $\subseteq$ B, we have $\beta \in$ B, and thus, since A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B, there exists $\alpha \in$ A such that $\alpha \succcurlyeq_w \beta$ for all $w \in \mathcal{W}$, and thus also for all $w \in \mathcal{W}'$ since $\mathcal{W}' \subseteq \mathcal{W}$. This proves that A′ $\succcurlyeq^{\mathcal{W}'}_{\forall\exists}$ B′.

Assume $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$; since $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$, there exists $\alpha \in A$ such that $\alpha \succcurlyeq_w \beta$ for all $w \in \mathcal{W}$ and for all $\beta \in B$, and thus also for all $w \in \mathcal{W}'$ since $\mathcal{W}' \subseteq \mathcal{W}$, and for all $\beta \in B'$ since $B' \subseteq B$. This proves that $A' \succcurlyeq^{\mathcal{W}'}_{\forall\exists} B'$.

(ii): (i) implies that if $A \succcurlyeq^{\mathcal{W}} B$, then $A \succcurlyeq^{\mathcal{W}} \{\beta\}$ for any $\beta \in B$. Regarding the converse for $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$, if $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} \{\beta\}$ for any $\beta \in B$, then for all $\beta \in B$ there exists $\alpha \in A$ and $w \in \mathcal{W}$ such that $\alpha \succcurlyeq_w \beta$, thus showing $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$. Regarding the converse for $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$, if $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} \{\beta\}$ for any $\beta \in B$, then for all $\beta \in B$ there exists $\alpha \in A$ such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$, and so we have $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$.

(iii): If for all $\beta \in B$ there exists $\alpha \in A$ such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$, then $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$. Thus for $B = A$ we can take $\alpha = \beta$ and we get that $\alpha \succcurlyeq_{\mathcal{W}} \beta$ for all $\beta \in B$, and so we have $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} A$. Regarding $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$, from Lemma 3.2.2(ii) it follows that if $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} A$ then $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} A$.

(iv): We prove the result for $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$; the result for $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ follows in exactly the same way. Let $A = B \cup C$. (iii) implies $B \succcurlyeq^{\mathcal{W}}_{\forall\forall} B$ and (i) implies that if $A \supseteq B$ and $B \succcurlyeq^{\mathcal{W}}_{\forall\forall} B$ then $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} B$.

□

Since the relations $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$ and $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ are reflexive and transitive, the symmetric parts, $\equiv^{\mathcal{W}}_{\forall\forall}$ and $\equiv^{\mathcal{W}}_{\forall\exists\forall}$, of these relations are equivalence relations. Also, Lemma 3.2.2 implies that $\equiv^{\mathcal{W}}_{\forall\exists} \subseteq \equiv^{\mathcal{W}}_{\forall\forall}$.

Clearly, $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$ and $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ determine the corresponding equivalence relations; conversely, $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$ and $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ can be expressed in terms of their corresponding equivalence relations:

Lemma 3.2.5. *For all $A, B \in \mathcal{M}$, $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} B \iff A \equiv^{\mathcal{W}}_{\forall\forall} A \cup B$; and $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \iff A \equiv^{\mathcal{W}}_{\forall\exists} A \cup B$.*

Proof: We prove the result for $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$; the result for $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ follows in exactly the same way. Lemma 3.2.4 implies that $A \cup B \succcurlyeq^{\mathcal{W}}_{\forall\forall} A$ holds for any $A, B \in \mathcal{M}$. It is then sufficient to show $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} B \iff A \succcurlyeq^{\mathcal{W}}_{\forall\forall} A \cup B$. From Lemma 3.2.4(ii) (Right Decomposition property) it follows that $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} A \cup B$ if and only if $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} A$ and $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} B$. Since $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$ is reflexive, $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} A$ is true for any $A \in \mathcal{M}$. Thus, $A \succcurlyeq^{\mathcal{W}}_{\forall\forall} A \cup B \iff (A \succcurlyeq^{\mathcal{W}}_{\forall\forall} A) \wedge (A \succcurlyeq^{\mathcal{W}}_{\forall\forall} B) \iff A \succcurlyeq^{\mathcal{W}}_{\forall\forall} B$. □

3.3 Filtering A and Minimal Equivalent Subsets

In this section we consider the question, raised in the introduction of the chapter, regarding replacing A with an equivalent subset of A, i.e., filtering out elements of A that are redundant.

Equivalence-free: we say that A ($\in \mathcal{M}$) is $\equiv_{\mathcal{W}}$-free (or *equivalence-free*) if for all $\alpha, \beta \in$ A, $\alpha \not\equiv_{\mathcal{W}} \beta$. One can reduce any A to an equivalence-free set A′ by including exactly one element in A′ of each $\equiv_{\mathcal{W}}$-equivalence class in A.

Setwise-minimal equivalent subsets: We define $\text{SME}_{\mathcal{W}}(\text{A})$ to be the set of subsets B of A that are setwise-minimal equivalent to A, i.e., such that B $\equiv^{\mathcal{W}}_{\forall\exists}$ A and there does not exist any strict subset C of B such that C $\equiv^{\mathcal{W}}_{\forall\exists}$ A.

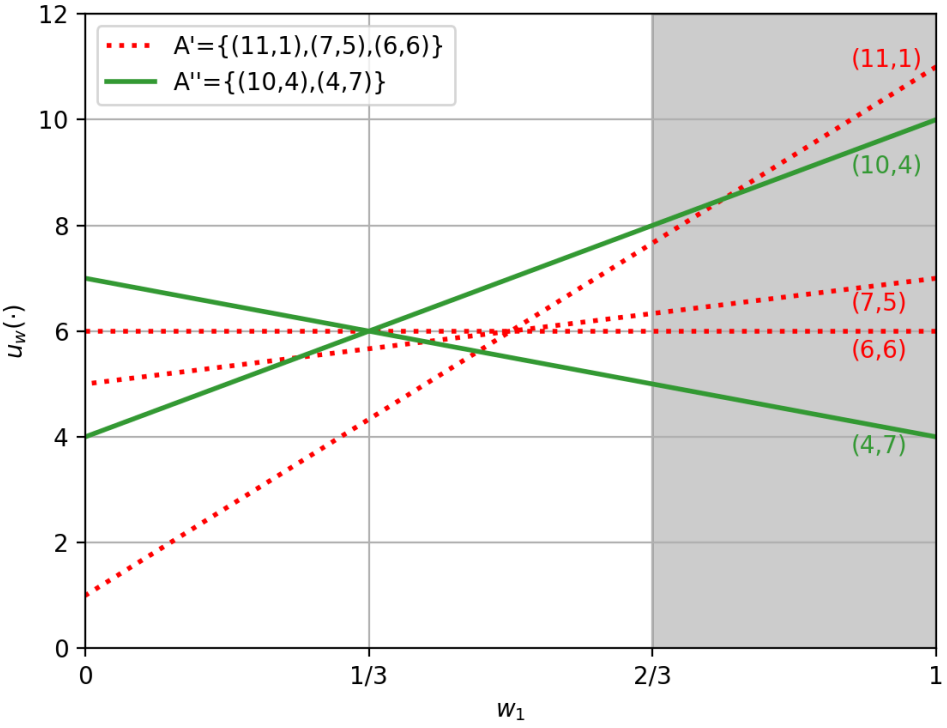


Figure 3.2: utility function $u_w(\alpha) = w \cdot \alpha$ for each alternative of the sets $\text{A}' = \{(11,1),(7,5),(6,6)\}$ and $\text{A}'' = \{(10,4),(4,7)\}$ where $w \in \mathcal{W} = \{(w_1, w_2) : w_1 + w_2 = 1 \;\&\; w_1 \in [0, \frac{2}{3}]\}$.

Example 8. *Consider the set of alternatives $A = A' \cup A''$, where $A' = \{(11,1),(7,5),(6,6)\}$ and $A'' = \{(10,4),(4,7)\}$. The utility function $u_w(\alpha) = w \cdot \alpha$ of the alternatives $\alpha \in A$ is shown in Figure 3.2. If we suppose $\mathcal{W} = \{(w_1, w_2)$:*

$w_1 + w_2 = 1$ & $w_1 \in [0, \frac{2}{3}]\}$, *then the unique setwise minimal equivalent subset of A is A″ since for each* $w \in \mathcal{W}$ *and* $\alpha \in A$, $Val_{A''}(w) \geq w \cdot \alpha$.

For the computation and the characterisation of $\mathrm{SME}_{\mathcal{W}}(\mathrm{A})$, we will consider some of the operators for filtering defined in Section 2.4, namely, $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, which removes dominated alternatives from A, $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$, which removes alternatives that are not possibly optimal, and $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ which removes alternatives that are not possibly strictly optimal. With Theorem 3.3.6 defined in Section 3.3.1 we determine when $\mathrm{SME}_{\mathcal{W}}(\mathrm{A})$ is a singleton, and in Section 3.3.2 we give a simple method to compute it. In section 3.3.3 we will also define the operators $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$ and $\mathrm{SMPO}_{\mathcal{W}}(\mathrm{A})$ that will be used to characterize $\mathrm{SME}_{\mathcal{W}}(\mathrm{A})$ more precisely under specific circumstances, and with Theorem 3.3.15 we show that for analytic utility functions $\mathrm{SME}_{\mathcal{W}}(\mathrm{A})$ equals $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$.

3.3.1 Operators for set of alternatives

Here we briefly recall the definitions of the operators $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$ and $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ defined in Section 2.4, and we define some related properties with respect to the setwise preference relations introduced in this chapter. In particular, we show that the operators $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ and $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$ can be used filter out elements whilst maintaining setwise equivalence, and with Theroem 3.3.6 we show the connection between $\mathrm{PSO}_{\mathcal{W}}$ and $\mathrm{SME}_{\mathcal{W}}$.

Operator $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$**:** Recall that $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ is the set of $\alpha \in \mathrm{A}$ such that there does not exist $\gamma \in \mathrm{A}$ such that $\gamma \succ_{\mathcal{W}} \alpha$.

We give a simple fundamental property of the set $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, which is used to prove e.g., Lemma 3.3.2 below.

Lemma 3.3.1. *Consider any* $A \in \mathcal{M}$.

(i) If $\alpha \in A \setminus \mathrm{UD}_{\mathcal{W}}(A)$ *then there exists* $\gamma \in \mathrm{UD}_{\mathcal{W}}(A)$ *such that* $\gamma \succ_{\mathcal{W}} \alpha$.

(ii) If $\alpha \in A$ *then there exists* $\gamma \in \mathrm{UD}_{\mathcal{W}}(A)$ *such that* $\gamma \succcurlyeq_{\mathcal{W}} \alpha$.

Proof: (i): Consider any $\alpha \in \mathrm{A} \setminus \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$. By the definition of $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, for any $\beta \in \mathrm{A} \setminus \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ there exists $\beta' \in \mathrm{A}$ such that $\beta' \succ_{\mathcal{W}} \beta$. Let $\alpha_1 = \alpha$. We construct a sequence $\alpha_1, \alpha_2, \ldots$, where for each $i = 1, 2, \ldots$, we have $\alpha_{i+1} \succ_{\mathcal{W}} \alpha_i$, where we stop the sequence when we reach an element α_i such that either (a) α_i has appeared earlier in the sequence, or (b) $\alpha_i \in \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$. Because A is finite,

there must be a last element α_k in the sequence. Transitivity of $\succ_{\mathcal{W}}$ implies that if $1 \leq i < k$ then $\alpha_k \succ_{\mathcal{W}} \alpha_i$, so, in particular, $\alpha_k \succ_{\mathcal{W}} \alpha$. If (a) $\alpha_k = \alpha_i$ for some $i < k$ then $\alpha_k \succ_{\mathcal{W}} \alpha_k$ which contradicts the fact that $\succ_{\mathcal{W}}$ is irreflexive. Thus, we have (b) $\alpha_k \in \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ and $\alpha_k \succ_{\mathcal{W}} \alpha$, showing part (i).

(ii): Consider any $\alpha \in \mathrm{A}$. If $\alpha \in \mathrm{A} \setminus \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ then, by part (i), there exists $\gamma \in \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ such that $\gamma \succcurlyeq_{\mathcal{W}} \alpha$. Otherwise, $\alpha \in \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, and, by reflexivity of $\succcurlyeq_{\mathcal{W}}$ we have $\alpha \succcurlyeq_{\mathcal{W}} \alpha$, so we can let $\gamma = \alpha$. □

Example 9. *Consider the running example in Figure 3.2 with* $A = A' \cup A'' = \{(11,1), (7,5), (6,6), (10,4), (4,7)\}$ *and* $\mathcal{W} = \{(w_1, w_2) : w_1 + w_2 = 1 \ \& \ w_1 \in [0, \frac{2}{3}]\}$. *We have that* $\mathrm{UD}_{\mathcal{W}}(A) = A \setminus \{(11,1)\}$. *As an example of Lemma 3.3.1, we have that* $(10,4) \succ_{\mathcal{W}} (11,1)$ *and, for example,* $(10,4) \succcurlyeq_{\mathcal{W}} (10,4)$.

Lemma 3.3.2. *Assume that* $\mathcal{W} \subseteq \mathcal{U}$ *and* $A \in \mathcal{M}$. *Then,* $\mathrm{UD}_{\mathcal{W}}(A)$ *is non-empty and the following hold.*

(i) $\mathrm{UD}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\exists} A$ *and* $\mathrm{UD}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\forall} A$.

(ii) For $B \in \mathcal{M}$ *and, for* $\succcurlyeq$ *being any of* $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$, $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ *or* $\succcurlyeq^{\mathcal{W}}_{\exists\forall}$, *we have* $A \succcurlyeq B \iff \mathrm{UD}_{\mathcal{W}}(A) \succcurlyeq \mathrm{UD}_{\mathcal{W}}(B)$.

Proof. (i): $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A}$ if and only if $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ and $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A}$. Since $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{A}$, from Lemma 3.2.4 follows that $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$. Regarding the converse, let $\mathrm{B} = \mathrm{A} \setminus \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, so that $\mathrm{A} = \mathrm{B} \cup \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$. From Lemma 3.3.1(i) it follows that for all $\alpha \in \mathrm{B}$ there exists $\gamma \in \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ such that $\gamma \succ_{\mathcal{W}} \alpha$, which is if and only if $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B}$. Thus, from Lemma 3.2.5 follows $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B} \cup \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, i.e., $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A}$.

Since $\equiv^{\mathcal{W}}_{\forall\forall} \supseteq \equiv^{\mathcal{W}}_{\forall\exists}$, then $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A}$ implies $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall} \mathrm{A}$.

(ii): Part (i) implies part (ii) when $\succcurlyeq$ is either $\succcurlyeq^{\mathcal{W}}_{\forall\forall}$ or $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$. This is because, if $\equiv$ is the corresponding equivalence relation, then $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \equiv \mathrm{A}$ and $\mathrm{UD}_{\mathcal{W}}(\mathrm{B}) \equiv \mathrm{B}$. Then, $\mathrm{A} \succcurlyeq \mathrm{B}$ implies $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \equiv \mathrm{A} \succcurlyeq \mathrm{B} \equiv \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$, and thus, $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$ by transitivity of $\succcurlyeq$. Similarly, $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$ implies $\mathrm{A} \succcurlyeq \mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq \mathrm{UD}_{\mathcal{W}}(\mathrm{B}) \succcurlyeq \mathrm{B}$ and thus, $\mathrm{A} \succcurlyeq \mathrm{B}$.

Regarding relation $\succcurlyeq^{\mathcal{W}}_{\exists\forall}$, we have, by part (i), $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ and $\mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$. First suppose, $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \mathrm{B}$. We have $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$. Applying parts (i) and (ii) of Lemma 3.2.3 implies $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\exists\forall} \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$. Now, assume that $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\exists\forall} \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$. We have $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists}$

$\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\exists\forall} \mathrm{UD}_{\mathcal{W}}(\mathrm{B}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B}$. Applying again parts (i) and (ii) of Lemma 3.2.3 we obtain $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\exists\forall} \mathrm{B}$. □

Operator $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$**:** Recall that $\mathrm{O}_w(\mathrm{A})$ is the set of elements α of A such that for all $\beta \in \mathrm{A}$, $\alpha \succcurlyeq_w \beta$. $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) = \bigcup_{w\in\mathcal{W}} \mathrm{O}_w(\mathrm{A})$.

Definition of $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$**:** We define, for $\alpha \in \mathrm{A}$, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ to consist of all scenarios $w \in \mathcal{W}$ in which α is optimal, i.e., $\alpha \in \mathrm{O}_w(\mathrm{A})$. Thus, $\alpha \in \mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \iff \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \neq \emptyset$.

Lemma 3.3.3. *Let $\mathcal{W} \subseteq \mathcal{U}$ and let $A \in \mathcal{M}$. For $B \subseteq A$, $B \equiv^{\mathcal{W}}_{\forall\exists} A$ if and only if $\bigcup_{\beta\in B} \mathrm{Opt}^{A}_{\mathcal{W}}(\beta) = \mathcal{W}$. In particular, $\bigcup_{\alpha\in A} \mathrm{Opt}^{A}_{\mathcal{W}}(\alpha) = \mathcal{W}$.*

Proof: We first prove that $\bigcup_{\alpha\in\mathrm{A}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathcal{W}$. Clearly, $\bigcup_{\alpha\in\mathrm{A}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq \mathcal{W}$. Now, consider any $w \in \mathcal{W}$. $\mathrm{O}_w(\mathrm{A})$ is clearly non-empty (since A is finite and $\succcurlyeq_w$ is a total pre-order), so let α be some element of it. Then $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \ni w$, so $\bigcup_{\alpha\in\mathrm{A}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \supseteq \mathcal{W}$.

Since $\mathrm{B} \subseteq \mathrm{A}$, we have $\mathrm{B} \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A} \iff \mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A} \iff$ for all $w \in \mathcal{W}$ and for all $\alpha \in \mathrm{A}$ there exists $\beta \in \mathrm{B}$ such that $\beta \succcurlyeq_w \alpha$. This holds if and only if for all $w \in \mathcal{W}$ there exists $\beta \in \mathrm{B}$ such that for all $\alpha \in \mathrm{A}$, $\beta \succcurlyeq_w \alpha$, i.e., for all $w \in \mathcal{W}$ there exists $\beta \in \mathrm{B}$ such that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta) \ni w$, which is equivalent to $\bigcup_{\beta\in\mathrm{B}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta) = \mathcal{W}$. □

Example 10. *Consider the running example in Figure 3.2 with $A = A' \cup A'' = \{(11,1),\ (7,5),\ (6,6),\ (10,4),\ (4,7)\}$ and $\mathcal{W} = \{(w_1, w_2) : w_1 + w_2 = 1\ \&\ w_1 \in [0, \frac{2}{3}]\}$. We have that $\mathrm{O}_w(A) = \{(4,7)\}$ for $w \in [0, \frac{1}{3})$, $\mathrm{O}_w(A) = \{(10,4),(4,7),(6,6)\}$ for $w \in [\frac{1}{3}, \frac{1}{3}]$, $\mathrm{O}_w(A) = (\{10,4\})$ for $w \in (\frac{1}{3}, \frac{2}{3}]$, and then $\bigcup_{w\in\mathcal{W}} \mathrm{O}_w(A) = \mathrm{PO}_{\mathcal{W}}(A) = \{(10,4),(4,7),(6,6)\}$. Also, as an example of Lemma 3.3.3, we have that $\mathrm{PO}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\exists} A$ since $\mathrm{Opt}^{A}_{\mathcal{W}}((4,7)) = [0, \frac{1}{3}]$, $\mathrm{Opt}^{A}_{\mathcal{W}}((10,4)) = [\frac{1}{3}, \frac{2}{3}]$, $\mathrm{Opt}^{A}_{\mathcal{W}}((6,6)) = [\frac{1}{3}, \frac{1}{3}]$, and then $\bigcup_{\beta\in\mathrm{PO}_{\mathcal{W}}(A)} \mathrm{Opt}^{A}_{\mathcal{W}}(\beta) = [0, \frac{2}{3}] = \mathcal{W}$.*

Lemma 3.3.4. *Assume that $\mathcal{W} \subseteq \mathcal{U}$ and $A \in \mathcal{M}$. Then, the following all hold.*

(i) $\mathrm{PO}_{\mathcal{W}}(A)$ *is non-empty, and for all $w \in \mathcal{W}$,* $\mathrm{O}_w(A)$ *is non-empty.*

(ii) For all $w \in \mathcal{W}$ and for all $\alpha \in A \setminus \mathrm{O}_w(A)$ there exists $\gamma \in \mathrm{O}_w(A)$ such that $\gamma \succ_w \alpha$, and thus, there exists $\gamma \in \mathrm{PO}_{\mathcal{W}}(A)$ such that $\gamma \succ_w \alpha$.

(iii) $\mathrm{PO}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\exists} A$.

(iv) For $B \subseteq \Omega$ we have $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \iff \mathrm{PO}_{\mathcal{W}}(A) \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$; and $B \succcurlyeq^{\mathcal{W}}_{\forall\exists} A \iff B \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PO}_{\mathcal{W}}(A)$. Thus, if B is also finite then $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \iff \mathrm{PO}_{\mathcal{W}}(A) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PO}_{\mathcal{W}}(B)$.

Proof. (i): If A is non-empty and finite, then we can compute $\alpha^* = \arg\max_{\alpha\in \mathrm{A}} u_w(\alpha)$. By definition of $\succcurlyeq_w$, $\alpha^* \succcurlyeq_w \alpha$ for all $\alpha \in$ A which is if and only if $\alpha^* \in \mathrm{O}_w(\mathrm{A})$. Therefore $\mathrm{O}_w(\mathrm{A})$ is non-empty and then $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) = \bigcup_{w\in\mathcal{W}} \mathrm{O}_w(\mathrm{A})$ is non-empty.

(ii): From (i) it follows that $\mathrm{O}_w(\mathrm{A})$ is not empty, then if $\alpha \in \mathrm{A} \setminus \mathrm{O}_w(\mathrm{A})$ then there exists $\gamma \in \mathrm{O}_w(\mathrm{A})$ such that $u_w(\gamma) > u_w(\alpha)$ which is if and only if $\gamma \succ_w \alpha$. Thus, since $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) = \bigcup_{w\in\mathcal{W}} \mathrm{O}_w(\mathrm{A})$, $\gamma \in \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$.

(iii): $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A}$ if and only if $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$ and $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A}$. Since $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{A}$, from Lemma 3.2.4 follows that $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$. Regarding the converse, let $\mathrm{B} = \mathrm{A} \setminus \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$, so that $\mathrm{A} = \mathrm{B} \cup \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$. From (ii) it follows that for all $\alpha \in \mathrm{B}$ and for all $w \in \mathcal{W}$ there exists $\gamma \in \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$ such that $\gamma \succ_w \alpha$, which implies $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B}$. Thus, from Lemma 3.2.5 we get that $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B} \cup \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$, i.e., $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A}$.

(iv): From (iii) follows that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A}$ and $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. Thus, for the transitivity of $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$, we have $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B} \iff \mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B}$; and $\mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A} \iff \mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$. Suppose $\mathrm{B} \in \mathcal{M}$. If $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B}$ then $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$, and if $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$ then $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PSO}_{\mathcal{W}}(\mathrm{B}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B}$. Thus, $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B} \iff \mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{PO}_{\mathcal{W}}(\mathrm{B})$.

□

Operator $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$**:** Recall that $\mathrm{SO}^{\mathcal{W}}_w(\mathrm{A})$ is the set of elements α of A such that $\alpha \succ_w \beta$, for all $\beta \in \mathrm{A}$ with $\beta \not\equiv_{\mathcal{W}} \alpha$. $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) = \bigcup_{w\in\mathcal{W}} \mathrm{SO}^{\mathcal{W}}_w(\mathrm{A})$.

Example 11. *Consider the running example in Figure 3.2 with $A = A' \cup A'' = \{(11,1), (7,5), (6,6), (10,4), (4,7)\}$ and $\mathcal{W} = \{(w_1, w_2) : w_1 + w_2 = 1 \;\&\; w_1 \in [0, \frac{2}{3}]\}$. We have that $\mathrm{SO}_w(A) = \{(4,7)\}$ for $w \in [0, \frac{1}{3})$, $\mathrm{SO}_w(A) = \emptyset$ for $w \in [\frac{1}{3}, \frac{1}{3}]$, $\mathrm{SO}_w(A) = (\{10,4\})$ for $w \in (\frac{1}{3}, \frac{2}{3}]$, and then $\bigcup_{w\in\mathcal{W}} \mathrm{SO}_w(A) = \mathrm{PSO}_{\mathcal{W}}(A) = \{(10,4), (4,7)\}$. Note that in this case $\bigcup_{\beta\in\mathrm{PSO}_{\mathcal{W}}(A)} \mathrm{Opt}^A_{\mathcal{W}}(\beta) \neq \mathcal{W}$*

Lemma 3.3.5. *Consider any $\mathcal{W} \subseteq \mathcal{U}$ and an $A \in \mathcal{M}$.*

(i) For $B \subseteq A$, if $B \equiv^{\mathcal{W}}_{\forall\exists} A$ then for all $\alpha \in \mathrm{PSO}_{\mathcal{W}}(A)$ there exists $\beta \in B$ with $\beta \equiv_{\mathcal{W}} \alpha$. In particular, if A is $\equiv_{\mathcal{W}}$-free then $B \supseteq \mathrm{PSO}_{\mathcal{W}}(A)$.

(ii) If $\alpha \in A \setminus \mathrm{PSO}_{\mathcal{W}}(A)$ *then* $\bigcup_{\beta \in A \setminus \{\alpha\}} \mathrm{Opt}^{A}_{\mathcal{W}}(\beta) = \mathcal{W}$ *and so* $A \setminus \{\alpha\} \equiv^{\mathcal{W}}_{\forall\exists} A$.

Proof: (i) Assume that $B \equiv^{\mathcal{W}}_{\forall\exists} A$. By Lemma 3.3.3, $\bigcup_{\beta \in B} \mathrm{Opt}^{A}_{\mathcal{W}}(\beta) = \mathcal{W}$. Consider any $\alpha \in \mathrm{PSO}_{\mathcal{W}}(A)$. Then there exists $w \in \mathcal{W}$ such that $\mathrm{SO}^{\mathcal{W}}_{w}(A) \ni \alpha$. Also, there exists $\beta \in B$ such that $\mathrm{Opt}^{A}_{\mathcal{W}}(\beta) \ni w$, and so, $\mathrm{O}_{w}(A) \ni \beta$. The definition of $\mathrm{SO}^{\mathcal{W}}_{w}(A)$ implies that $\beta \equiv_{\mathcal{W}} \alpha$. In particular, if A is $\equiv_{\mathcal{W}}$-free then for all $\alpha \in \mathrm{PSO}_{\mathcal{W}}(A)$, $\alpha \in B$, so $B \supseteq \mathrm{PSO}_{\mathcal{W}}(A)$.

(ii): Suppose that $\alpha \in A \setminus \mathrm{PSO}_{\mathcal{W}}(A)$ and consider any $w \in \mathcal{W}$. Then $\mathrm{O}_{w}(A) \neq \{\alpha\}$ so there exists $\beta \in \mathrm{O}_{w}(A) \setminus \{\alpha\}$, and thus, $\mathrm{Opt}^{A}_{\mathcal{W}}(\beta) \ni w$. This shows that $\bigcup_{\beta \in A \setminus \{\alpha\}} \mathrm{Opt}^{A}_{\mathcal{W}}(\beta) = \mathcal{W}$, and thus, by Lemma 3.3.3, $A \setminus \{\alpha\} \equiv^{\mathcal{W}}_{\forall\exists} A$. □

Theorem 3.3.6 below gives some relationships between PSO, SME and the dominance relation $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$, for equivalence-free A. Any setwise-minimal equivalent subset of A contains $\mathrm{PSO}_{\mathcal{W}}(A)$, the set of possibly strictly optimal elements. The latter set is equivalent to A if and only if there is a unique minimal equivalent subset, which is thus equal to $\mathrm{PSO}_{\mathcal{W}}(A)$.

The condition that $\mathrm{PSO}_{\mathcal{W}}(A)$ is equivalent to A holds in the linear multi-objective case considered in Section 3.7 below (see Theorem 3.3.15), and so then $\mathrm{PSO}_{\mathcal{W}}(A)$ is the unique minimal equivalent subset of A. Part (ii) implies that the relation $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ can be used for computing $\mathrm{PSO}_{\mathcal{W}}(A)$.

Theorem 3.3.6. *Assume that A* ($\in \mathcal{M}$) *is* $\equiv_{\mathcal{W}}$*-free and let* $\mathcal{W} \subseteq \mathcal{U}$. *Then the following hold:*

(i) $\bigcap_{B \in \mathrm{SME}_{\mathcal{W}}(A)} B = \mathrm{PSO}_{\mathcal{W}}(A)$;

(ii) $\mathrm{PSO}_{\mathcal{W}}(A)$ *is the set of all* $\alpha \in A$ *such that* $A \setminus \{\alpha\} \not\succcurlyeq^{\mathcal{W}}_{\forall\exists} \{\alpha\}$;

(iii) $\mathrm{PSO}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\exists} A$ *if and only if* $\mathrm{SME}_{\mathcal{W}}(A)$ *is a singleton, which is if and only if* $\mathrm{PSO}_{\mathcal{W}}(A)$ *is the unique setwise-minimal equivalent subset for A.*

Proof: (i) First consider any $B \in \mathrm{SME}_{\mathcal{W}}(A)$, and thus, $B \equiv^{\mathcal{W}}_{\forall\exists} A$. Lemma 3.3.5 implies that $B \supseteq \mathrm{PSO}_{\mathcal{W}}(A)$. Hence, $\bigcap_{B \in \mathrm{SME}_{\mathcal{W}}(A)} B \supseteq \mathrm{PSO}_{\mathcal{W}}(A)$. Conversely, consider any $\alpha \in A \setminus \mathrm{PSO}_{\mathcal{W}}(A)$. Lemma 3.3.5 implies $A \setminus \{\alpha\} \equiv^{\mathcal{W}}_{\forall\exists} A$. Since A is finite, there exists a subset C of $A \setminus \{\alpha\}$ that is setwise-minimal equivalent to A, and so $C \in \mathrm{SME}_{\mathcal{W}}(A)$ and $C \not\ni \alpha$, which implies that $\bigcap_{B \in \mathrm{SME}_{\mathcal{W}}(A)} B \not\ni \alpha$. This proves that $\bigcap_{B \in \mathrm{SME}_{\mathcal{W}}(A)} B \subseteq \mathrm{PSO}_{\mathcal{W}}(A)$.

(ii): Consider any $\alpha \in A \setminus \mathrm{PSO}_{\mathcal{W}}(A)$. Lemma 3.3.5 implies $A \setminus \{\alpha\} \equiv^{\mathcal{W}}_{\forall\exists} A$, and

thus $\mathsf{A} \setminus \{\alpha\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$. Conversely, if $\mathsf{A} \setminus \{\alpha\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$ then for each $w \in \mathcal{W}$ there exists $\gamma \in \mathsf{A}$ such that $\gamma \succcurlyeq_w \alpha$, i.e., $\gamma \notin \mathrm{PSO}_{\mathcal{W}}(\mathsf{A})$.

(iii): Now let us assume that $\mathrm{SME}_{\mathcal{W}}(\mathsf{A})$ is a singleton, say $\{\mathsf{B}\}$. By definition, $\mathsf{B} \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathsf{A}$, and, by (i), $\mathsf{B} = \mathrm{PSO}_{\mathcal{W}}(\mathsf{A})$, showing that $\mathrm{PSO}_{\mathcal{W}}(\mathsf{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathsf{A}$. Conversely, assume that $\mathrm{PSO}_{\mathcal{W}}(\mathsf{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathsf{A}$, which implies that there exists some subset C of $\mathrm{PSO}_{\mathcal{W}}(\mathsf{A})$ such that $\mathsf{C} \in \mathrm{SME}_{\mathcal{W}}(\mathsf{A})$. Using (i) we have $\mathrm{PSO}_{\mathcal{W}}(\mathsf{A}) = \bigcap_{\mathsf{B}\in\mathrm{SME}_{\mathcal{W}}(\mathsf{A})} \mathsf{B} \subseteq \mathsf{C} \subseteq \mathrm{PSO}_{\mathcal{W}}(\mathsf{A})$. Thus, $\mathsf{C} = \mathrm{PSO}_{\mathcal{W}}(\mathsf{A}) = \bigcap_{\mathsf{B}\in\mathrm{SME}_{\mathcal{W}}(\mathsf{A})} \mathsf{B}$ and so $\mathrm{PSO}_{\mathcal{W}}(\mathsf{A}) \in \mathrm{SME}_{\mathcal{W}}(\mathsf{A})$, and any element of $\mathrm{SME}_{\mathcal{W}}(\mathsf{A})$ contains $\mathrm{PSO}_{\mathcal{W}}(\mathsf{A})$. By definition of $\mathrm{SME}_{\mathcal{W}}(\mathsf{A})$ this implies that $\mathrm{SME}_{\mathcal{W}}(\mathsf{A}) = \{\mathrm{PSO}_{\mathcal{W}}(\mathsf{A})\}$.

We conclude this subsection introducing two further operators, namely, the Maximally Possibly Optimal Set ($\mathrm{MPO}_{\mathcal{W}}(\mathsf{A})$) and the Strictly Maximal Possibly Optimal Set ($\mathrm{SMPO}_{\mathcal{W}}(\mathsf{A})$). These will be used to further characterize minimal equivalent subsets.

Operator $\mathrm{MPO}_{\mathcal{W}}(\mathsf{A})$**:** Let us define the maximally possibly optimal elements to be those that are optimal in a maximal set of scenarios. For $\mathsf{A} \in \mathcal{M}$, we define $\mathrm{MPO}_{\mathcal{W}}(\mathsf{A})$ to consist of all $\gamma \in \mathsf{A}$ such that there exists no $\alpha \in \mathsf{A}$ such that $\mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\alpha) \supsetneqq \mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\gamma)$.

Lemma 3.3.7. *Assume that $\mathcal{W} \subseteq \mathcal{U}$ and $\mathsf{A} \in \mathcal{M}$. $\mathrm{MPO}_{\mathcal{W}}(\mathsf{A}) \subseteq \mathrm{UD}_{\mathcal{W}}(\mathsf{A})$.*

Proof. From Lemma 3.3.1 it follows that if $\gamma \notin \mathrm{UD}_{\mathcal{W}}(\mathsf{A})$ then there exists $\alpha \in \mathrm{UD}_{\mathcal{W}}(\mathsf{A}) \subseteq \mathsf{A}$ such that $\alpha \succ_{\mathcal{W}} \gamma$, thus $\mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\alpha) \supsetneqq \mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\gamma)$, i.e., $\gamma \notin \mathrm{MPO}_{\mathcal{W}}(\mathsf{A})$. □

Operator $\mathrm{SMPO}_{\mathcal{W}}(\mathsf{A})$**:** Let us define $\mathrm{SMPO}_{\mathcal{W}}(\mathsf{A})$ (the *Strictly Maximal Possibly Optimal* elements of A) to consist of all $\gamma \in \mathsf{A}$ such that for all $\alpha \in \mathsf{A}$ if $\mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\alpha) \supseteq \mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\gamma)$ then $\alpha \equiv_{\mathcal{W}} \gamma$.

Lemma 3.3.8. *Assume that $\mathcal{W} \subseteq \mathcal{U}$ and $\mathsf{A} \in \mathcal{M}$. $\mathrm{SMPO}_{\mathcal{W}}(\mathsf{A}) \subseteq \mathrm{MPO}_{\mathcal{W}}(\mathsf{A})$.*

Proof. $\mathrm{SMPO}_{\mathcal{W}}(\mathsf{A}) \subseteq \mathrm{MPO}_{\mathcal{W}}(\mathsf{A})$: If $\gamma \notin \mathrm{MPO}_{\mathcal{W}}(\mathsf{A})$ then there exists $\alpha \in \mathsf{A}$ such that $\mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\alpha) \supsetneqq \mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\beta)$. Thus from the definition of $\mathrm{SMPO}_{\mathcal{W}}(\mathsf{A})$ it follows that $\beta \notin \mathrm{SMPO}_{\mathcal{W}}(\mathsf{A})$ since $\mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\alpha) \supseteq \mathrm{Opt}^{\mathsf{A}}_{\mathcal{W}}(\beta)$ but $\alpha \not\equiv_{\mathcal{W}} \gamma$. □

The following result states relationships between the different operators, and shows that we can replace A and B by (for instance) $\mathrm{PO}_{\mathcal{W}}(\mathsf{A})$ and $\mathrm{PO}_{\mathcal{W}}(\mathsf{B})$, respectively, in testing $\mathsf{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathsf{B}$.

Lemma 3.3.9. *Assume that $\mathcal{W} \subseteq \mathcal{U}$ and $\mathsf{A} \in \mathcal{M}$. Then the following hold:*

(i) $\mathrm{PSO}_{\mathcal{W}}(A) \subseteq \mathrm{SMPO}_{\mathcal{W}}(A) \cap \mathrm{UD}_{\mathcal{W}}(A) \subseteq \mathrm{SMPO}_{\mathcal{W}}(A) \subseteq \mathrm{MPO}_{\mathcal{W}}(A) \subseteq \mathrm{PO}_{\mathcal{W}}(A) \subseteq A$.

(ii) $\mathrm{OP}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\forall\exists} A$ *if* $\mathrm{OP}_{\mathcal{W}}(A)$ *is any of the following:* $\mathrm{MPO}_{\mathcal{W}}(A) \cap \mathrm{UD}_{\mathcal{W}}(A)$, $\mathrm{MPO}_{\mathcal{W}}(A)$, $\mathrm{PO}_{\mathcal{W}}(A) \cap \mathrm{UD}_{\mathcal{W}}(A)$, *or* $\mathrm{PO}_{\mathcal{W}}(A)$. *Thus,* $A \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} B \iff \mathrm{OP}_{\mathcal{W}}(A) \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{OP}_{\mathcal{W}}(B)$.

Proof: (i):$\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{SMPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{SMPO}_{\mathcal{W}}(\mathrm{A})$: If $\gamma \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, then there exists $w \in \mathcal{W}$ such that $\gamma \succ_w \alpha$ for any $\alpha \in \mathrm{A}$. Thus, $\gamma \in \mathrm{SMPO}_{\mathcal{W}}(\mathrm{A})$ since there does not exists $\alpha \in \mathrm{A}$ such that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \supseteq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$. $\mathrm{SMPO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ follows from Lemma 3.3.7 and Lemma 3.3.8. □

$\mathrm{SMPO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$: This follows from Lemma 3.3.8.

$\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$: If $\gamma \notin \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$, then $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma) = \emptyset$. Thus, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \supsetneq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$ for any $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ which implies $\gamma \notin \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$.

$\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{A}$: $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$ is by definition a subset of A.

(ii): we show that $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$. Since $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{A}$ we have $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$. We need to show the converse, that $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$. Consider any $w \in \mathcal{W}$ and $\alpha \in \mathrm{A}$; it is sufficient to show that there exists $\beta \in \mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ with $\beta \succcurlyeq_w \alpha$. Finiteness of A implies that there exists $\gamma \in \mathrm{A}$ such that $\gamma \in \mathrm{O}_w(\mathrm{A})$, i.e., $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$, and there does not exist $\delta \in \mathrm{A}$ with $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\delta) \supsetneq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$, and thus, $\gamma \in \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$ which by (i) implies $\gamma \in \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$. From Lemma 3.3.1(ii) it follows that there exists $\beta \in \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ with $\beta \succcurlyeq_{\mathcal{W}} \gamma$, which implies that $\beta \in \mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ and $\beta \succcurlyeq_w \alpha$. Now, if $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{B} \subseteq \mathrm{A}$ then $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$ and so $\mathrm{A} \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$. Using part (i), we thus have $\mathrm{OP}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$ if $\mathrm{OP}_{\mathcal{W}}(\mathrm{A})$ is any of the following: $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$, $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, or $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$.

We also define here the following technical lemma which is used later, to prove Corollary 3.3.16.1.

Lemma 3.3.10. *Let* $A \in \mathcal{M}$ *and let* $\mathcal{W} \subseteq \mathcal{U}$. *Assume that* $\mathrm{PSO}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\forall\exists} A$ *and that for any* $B \subseteq A$, $\mathrm{SMPO}_{\mathcal{W}}(B) = \mathrm{PSO}_{\mathcal{W}}(B)$. *Then* $\mathrm{MPO}_{\mathcal{W}}(A) = \mathrm{PSO}_{\mathcal{W}}(A)$.

Proof: Suppose otherwise, and so there exists some $\alpha \in \mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) \setminus \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. Let $\mathrm{B} = \mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \cup \{\alpha\}$. Now, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is not contained in (or equal to) $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$ for any $\beta \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, since if $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$ then $\alpha \in \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$ implies $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$, which would then imply $\alpha \equiv_{\mathcal{W}} \beta$

(since $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$), which contradicts $\alpha \notin \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. Thus, $\alpha \in \mathrm{SMPO}_{\mathcal{W}}(\mathrm{B})$, which equals $\mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$ by the hypothesis. Now, $\mathrm{PSO}_{\mathcal{W}}(\mathrm{B}) \supseteq \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, since $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{B}$, so each element of $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is still possibly strictly optimal in the reduced set B. Thus, $\mathrm{PSO}_{\mathcal{W}}(\mathrm{B}) = \mathrm{B}$, and we have $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \subsetneqq \mathrm{PSO}_{\mathcal{W}}(\mathrm{B}) = \mathrm{B} \subseteq \mathrm{A}$, and $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$ and so, $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$. But Lemma 3.3.5(i) implies that there exists $\gamma \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ with $\gamma \equiv_{\mathcal{W}} \alpha$ (since $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$ and $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$), which implies that $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, which is the contradiction required. □

3.3.2 Filtering

A simple way of generating a minimal equivalent subset of A is to sequentially delete elements α of A that are not needed for maintaining equivalence, i.e., are such that $\mathrm{A} \setminus \{\alpha\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$, since then $\mathrm{A} \setminus \{\alpha\} \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$. This is what is done in the operation *Filter*$_\sigma(\mathrm{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$ defined below, to produce a minimal equivalent subset of A.

For $\alpha \in \mathrm{A}$, define *Filter*$(\mathrm{A}, \alpha; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$ to be $\mathrm{A} \setminus \{\alpha\}$ if $\mathrm{A} \setminus \{\alpha\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$; otherwise it equals A.

More generally, for $\mathrm{B} \subseteq \mathrm{A}$, we define *Filter*$(\mathrm{A}, \mathrm{B}; \succcurlyeq)$ to be $\mathrm{A} \setminus \mathrm{B}$ if $\mathrm{A} \setminus \mathrm{B} \succcurlyeq \mathrm{B}$; otherwise it equals A.

Let us label A as $\alpha_1, \ldots, \alpha_n$, where $n = |\mathrm{A}|$. Formally the labelling is a bijection σ from $\{1, \ldots, n\}$ to A (so that $\sigma(i) = \alpha_i$), and let Λ be the set of all labellings. We define *Filter*$_\sigma(\mathrm{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$ iteratively as follows. We set $\mathrm{A}^0 = \mathrm{A}$. For $i = 1, \ldots, n$, we set $\mathrm{A}^i =$ *Filter*$(\mathrm{A}^{i-1}, \alpha_i; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$. We then define *Filter*$_\sigma(\mathrm{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$ to be A^n, i.e., the set remaining after iteratively deleting elements from A that are dominated with respect to relation $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$.

Example 12. *Consider the set of alternatives* $A^0 = \{(10,4), (11,1), (4,7), (7,5), (6,6)\}$*, where the utility function* $u_w(\alpha) = w \cdot \alpha$ *of the alternatives* $\alpha \in A$ *with* $w \in \mathcal{W} = \{(w_1, w_2) : w_1 + w_2 = 1 \;\&\; w_1 \in [0, \frac{2}{3}]\}$ *shown in Figure 3.2. In this case,* $A^1 =$ *Filter*$_\sigma(A^0, (6,6); \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) = A^0 \setminus \{(6,6)\}$ *since* $A^0 \setminus \{(6,6)\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{(6,6)\}$. $A^2 = A^1 \setminus \{(7,5)\}$ *since* $A^1 \setminus \{(7,5)\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{(7,5)\}$. $A^3 = A^2$ *since* $A^2 \setminus \{(4,7)\} \not\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{(4,7)\}$. $A^4 = A^3 \setminus \{(11,1)\}$ *since* $A^3 \setminus \{(11,1)\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{(11,1)\}$. $A^5 = A^4$ *since* $A^4 \setminus \{(10,4)\} \not\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{(10,4)\}$. *Thus, we get Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) = \{(10,4), (4,7)\}$.

We define *Filter*$(\mathrm{A}, \alpha; \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall})$ and *Filter*$_\sigma(\mathrm{A}; \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall})$ analogously.

Lemma 3.3.11. *Let* $\succcurlyeq$ *be either* $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ *or* $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$*, and let* $\equiv$ *be the corresponding*

equivalence relation. Let $A, A' \in \mathcal{M}$ and let σ be any labelling of A. Then we have:

(i) If $A' \subseteq A$ and $A' \succcurlyeq A \setminus A'$ then $A \equiv A'$;

(ii) For $B \subseteq A$, Filter$(A, B; \succcurlyeq) \equiv A$.

(iii) $A \equiv$ Filter$_\sigma(A; \succcurlyeq) \subseteq A$.

(iv) Filter$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\exists}) \subseteq \mathrm{UD}_{\mathcal{W}}(A)$.

(v) SME$_{\mathcal{W}}(A) = \{$Filter$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) : \sigma \in \Lambda\}$.

(vi) Filter$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) \subseteq \mathrm{PO}_{\mathcal{W}}(A)$.

(i): From Lemma 3.2.5 it follows that $A' \succcurlyeq A \setminus A' \iff A' \equiv A' \cup (A \setminus A')$, i.e. $A' \equiv A$.

(ii): By definition, if $A \setminus B \not\succcurlyeq B$ then *Filter*$(A, B; \succcurlyeq) = A$, and thus *Filter*$(A, B; \succcurlyeq) \equiv A$. On the other hand, if $A \setminus B \succcurlyeq B$ then *Filter*$(A, B; \succcurlyeq) = A \setminus B$, but from Lemma 3.2.5 it follows that if $A \setminus B \succcurlyeq B$ with $B \subseteq A$, then $A \setminus B \equiv A$, i.e., *Filter*$(A, B; \succcurlyeq) \equiv A$.

(iii): From (ii) it follows that for any $i \in \{1, \ldots, n\}$, $A^i =$ *Filter*$(A^{i-1}, \{\alpha_i\}; \succcurlyeq) \equiv A^{i-1}$. Thus, *Filter*$_\sigma(A; \succcurlyeq) = A^n \equiv \cdots \equiv A^0$, i.e., $A \equiv$ *Filter*$_\sigma(A; \succcurlyeq)$. Since $A^i \subseteq A^{i-1}$ for any $i \in \{1, \ldots, n\}$, then *Filter*$_\sigma(A; \succcurlyeq) = A^n \subseteq \cdots \subseteq A^0$, i.e., *Filter*$_\sigma(A; \succcurlyeq) = A^n \subseteq A$.

(iv): Suppose that $\alpha_i \in A \setminus$ *Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\exists})$. Then, using the notation above, $A^i \not\ni \alpha_i$, i.e., $\alpha \notin$ *Filter*$(A^{i-1}, \alpha_i; \succcurlyeq^{\mathcal{W}}_{\forall\exists})$, and thus, there exists $\gamma \in A^{i-1}$ with $\gamma \succcurlyeq_{\mathcal{W}} \alpha$. If $\alpha_i \in \mathrm{UD}_{\mathcal{W}}(A)$ this implies that $\gamma \equiv_{\mathcal{W}} \alpha$. Applying this iteratively, we see that if $\alpha_i \in \mathrm{UD}_{\mathcal{W}}(A) \setminus$ *Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\exists})$ then there exists $\gamma \in$ *Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\exists})$ with $\gamma \equiv_{\mathcal{W}} \alpha_i$.

Now assume that $\alpha_i \in A \setminus \mathrm{UD}_{\mathcal{W}}(A)$; then, by Lemma 3.3.1, there exists $\beta \in \mathrm{UD}_{\mathcal{W}}(A)$ with $\beta \succ_{\mathcal{W}} \alpha_i$; by the above argument, there exists $\gamma \in$ *Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\exists})$ with $\gamma \equiv_{\mathcal{W}} \beta$ and thus, $\gamma \succ_{\mathcal{W}} \alpha_i$. We have $\gamma \in$ *Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\exists}) \subseteq A^{i-1}$, which implies that $\alpha_i \notin A^i$, and thus, $\alpha_i \notin$ *Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\exists})$.

(v): Firstly, we observe that if $\alpha \in$ *Filter*$_\sigma(A; \succcurlyeq)$ then *Filter*$_\sigma(A; \succcurlyeq) \setminus \alpha \not\succcurlyeq \{\alpha\}$. (This follows using the fact that if $\alpha_i = \alpha$ then $A^i =$ *Filter*$(A^{i-1}, \alpha_i; \succcurlyeq) \ni \alpha_i$ in the sequence of sets, i.e., $A^{i-1} \setminus \{\alpha_i\} \not\succcurlyeq \{\alpha_i\}$, which, by monotonicity, implies *Filter*$_\sigma(A; \succcurlyeq) \setminus \{\alpha_i\} \not\succcurlyeq \{\alpha_i\}$, since *Filter*$_\sigma(A; \succcurlyeq) \subseteq A^{i-1}$.) This implies, using Lemma 3.2.5, that no strict subset of *Filter*$_\sigma(A; \succcurlyeq)$ is equivalent to A. In particular, for the case in which $\succcurlyeq$ equals $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$, we obtain that *Filter*$_\sigma(A; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$

$) \in \text{SME}_{\mathcal{W}}(\text{A})$.

Conversely, for $\text{B} \in \text{SME}_{\mathcal{W}}(\text{A})$; to complete the proof we will show that there exists $\sigma \in \Lambda$ such that $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) = \text{B}$. We choose σ to list the elements of B last. Now, $\text{B} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \text{A}$, and so, $\text{B} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$ for each $\alpha \in \text{A} \setminus \text{B}$. This implies that $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) \subseteq \text{B}$. Since, we have $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \text{A}$, and $\text{B} \in \text{SME}_{\mathcal{W}}(\text{A})$, by definition of $\text{SME}_{\mathcal{W}}(\text{A})$ we have $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) = \text{B}$.

(vi): From (v) it follows that for any $\alpha \in \mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$ there must be w such that $\alpha \in \text{Opt}_{w}(\text{A})$, i.e., $\alpha \in \text{PO}_{\mathcal{W}}(\text{A})$, otherwise $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) \setminus \{\alpha\}$ would be equivalent to A, i.e., $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) \notin \text{SME}_{\mathcal{W}}(\text{A})$. Thus, $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) \subseteq \text{PO}_{\mathcal{W}}(\text{A})$.

As the proposition below states, when applying the filtering operation $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$, (i) equivalence is always maintained; and (ii) we always obtain a minimal equivalent subset, and any such subset can be achieved for some ordering. Part (iii) implies that for any labelling σ we have $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) = \text{PSO}_{\mathcal{W}}(\text{A})$ if $\text{PSO}_{\mathcal{W}}(\text{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \text{A}$.

Proposition 1 immediately follows, using Lemma 3.3.11, and with part (iii) also using Theorem 3.3.6.

Proposition 1. *Let $A \in \mathcal{M}$ and let σ be any labelling of A. Then we have:*

(i) $A \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathit{Filter}_{\sigma}(A; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) \subseteq A$.

(ii) $\mathit{SME}_{\mathcal{W}}(A) = \{\mathit{Filter}_{\sigma}(A; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) : \sigma \in \Lambda\}$.

(iii) If A is $\equiv_{\mathcal{W}}$-free and $\text{PSO}_{\mathcal{W}}(A) \equiv^{\mathcal{W}}_{\forall\forall\exists} A$ then $\mathit{Filter}_{\sigma}(A; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) = \text{PSO}_{\mathcal{W}}(A)$ for any labelling σ.

Proof. (i): It follows from Lemma 3.3.11(iii).

(ii): See Lemma 3.3.11(v).

(iii): If A is $\equiv_{\mathcal{W}}$-free, Theorem 3.3.6 implies that $\text{SME}_{\mathcal{W}}(\text{A}) = \{\text{PSO}_{\mathcal{W}}(\text{A})\}$, so part (ii) then implies that $\mathit{Filter}_{\sigma}(\text{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}) = \text{PSO}_{\mathcal{W}}(\text{A})$ for every labelling σ. □

Let $\gg^{\mathcal{W}}_{\forall\forall\exists}$ be the strongly strict version of the dominance relation $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$, i.e., $\text{A} \gg^{\mathcal{W}}_{\forall\forall\exists} \text{B}$ if and only if for all $w \in \mathcal{W}$, $\text{Val}_{\text{A}}(w) > \text{Val}_{\text{B}}(w)$. One can define $\mathit{Filter}_{\sigma}(\text{A}; \gg^{\mathcal{W}}_{\forall\forall\exists})$ analogously with the result being $\text{PO}_{\mathcal{W}}(\text{A})$, irrespective of σ, and without requiring any conditions on A.

Proposition 2. *For $A \in \mathcal{M}$, $\mathit{Filter}_{\sigma}(A; \gg^{\mathcal{W}}_{\forall\forall\exists}) = \text{PO}_{\mathcal{W}}(A)$.*

For a proof of Proposition 2 see Theorem 1 of [BP15b].

3.3.3 $\mathrm{PSO}_{\mathcal{W}}(\mathbf{A})$ as unique minimal equivalent set

In order to show the uniqueness of a minimal equivalent set in certain circumstances, we first prove some lemmas.

Lemma 3.3.12. *For each $\alpha, \beta \in \Omega$, let $H_{\alpha \geq \beta} = \{w \in \mathcal{U} : u_w(\alpha) \geq u_w(\beta)\}$. Let $\mathcal{W} \subseteq \mathcal{U}$ and let $A \in \mathcal{M}$.*

(i) For $\alpha \in A$, $\mathrm{Opt}^A_{\mathcal{W}}(\alpha) = \mathcal{W} \cap \bigcap_{\beta \in A} H_{\alpha \geq \beta}$.

(ii) Suppose that for each $\alpha \in \Omega$, $u_w(\alpha)$ is a continuous function over $w \in \mathcal{U}$. Then for each $\alpha, \beta \in \Omega$, $H_{\alpha \geq \beta}$ is a topologically closed subset of $\mathcal{U}$, and $\mathrm{Opt}^A_{\mathcal{W}}(\alpha)$ is a topologically closed subset of $\mathcal{W}$.

Proof: (i): $w \in \mathcal{W} \cap \bigcap_{\beta \in \mathrm{A}} H_{\alpha \geq \beta}$ if and only if $w \in \mathcal{W}$ and for all $\beta \in \mathrm{A}$, $u_w(\alpha) \geq u_w(\beta)$, i.e., $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$.

(ii): The function $G : \mathcal{U} \to I\!R$, given by $G(w) = u_w(\alpha) - u_w(\beta)$, is continuous. Then $H_{\alpha \geq \beta} = \{w \in \mathcal{U} : G(w) \geq 0\}$. Since $[0, \infty)$ is a closed subset of the reals, $H_{\alpha \geq \beta}$ is a closed subset of $\mathcal{U}$, and therefore $H_{\alpha \geq \beta} \cap \mathcal{W}$ is a closed subset of $\mathcal{W}$. Using (i) $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is an intersection of closed subsets of $\mathcal{W}$, and so is closed. □

Lemma 3.3.13. *Let $A \in \mathcal{M}$ and let $\mathcal{W} \subseteq \mathcal{U} = I\!R^p$. Assume that for each $\alpha \in A$, $u_w(\alpha)$ is a continuous function of w. Consider any measure on $\mathcal{W}$ such that $\mathrm{Opt}^A_{\mathcal{W}}(\alpha)$ is measurable for each $\alpha \in A$ and such that $\mathcal{W}$ has non-zero measure. Define $\widehat{A}$ to be elements α of A such that $\mathrm{Opt}^A_{\mathcal{W}}(\alpha)$ has non-zero measure.*

(i) If $\mathcal{W}$ is such that, for any open ball S in $I\!R^p$ intersecting with $\mathcal{W}$, $S \cap \mathcal{W}$ has non-zero measure then $\bigcup_{\alpha \in \widehat{A}} \mathrm{Opt}^A_{\mathcal{W}}(\alpha) = \mathcal{W}$.

(ii) Assume that for any $\alpha, \beta \in A$, if $\mathcal{W}_{\alpha=\beta} = \{w \in \mathcal{W} : u_w(\alpha) = u_w(\beta)\}$ then either $\mathcal{W}_{\alpha=\beta} = \mathcal{W}$ or $\mathcal{W}_{\alpha \geq \beta}$ has measure zero. If α and β are elements of $\widehat{A}$ with $\alpha \not\equiv_{\mathcal{W}} \beta$ then $\mathrm{Opt}^A_{\mathcal{W}}(\alpha) \cap \mathrm{Opt}^A_{\mathcal{W}}(\beta)$ has measure zero.

Proof: (i): Assume that $\mathcal{W}$ is such that, for any open ball S in $I\!R^p$ intersecting with $\mathcal{W}$, $S \cap \mathcal{W}$ has non-zero measure. To show that $\bigcup_{\alpha \in \widehat{\mathrm{A}}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathcal{W}$ we proceed by contradiction, so suppose that $\bigcup_{\alpha \in \widehat{\mathrm{A}}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \neq \mathcal{W}$. Then there exists $w \in \mathcal{W}'' = \mathcal{W} \setminus \bigcup_{\alpha \in \widehat{\mathrm{A}}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$. Since $\bigcup_{\alpha \in \mathrm{A}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathcal{W}$, $\mathcal{W}'' \subseteq \bigcup_{\alpha \in \mathrm{A} \setminus \widehat{\mathrm{A}}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$, with the latter, by definition of $\widehat{\mathrm{A}}$, being of measure zero. Thus $\mathcal{W}''$ is of measure zero. Now, by Lemma 3.3.12, for any $\alpha \in \mathrm{A}$,

$\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is a topologically closed subset of $\mathcal{W}$. Since $\widehat{\mathrm{A}}$ is finite, $\bigcup_{\alpha\in\widehat{\mathrm{A}}}\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is closed, so $\mathcal{W}''$ is an open subset of $\mathcal{W}$. Then there exists an open ball S in $\mathbb{R}^p$ with $w \in S\cap\mathcal{W} \subseteq \mathcal{W}''$. By the hypothesis, $S\cap\mathcal{W}$ has non-zero measure, which contradicts $\mathcal{W}''$ having zero measure.

(ii): Let α and β be elements of $\widehat{\mathrm{A}}$ with $\alpha \not\equiv_{\mathcal{W}} \beta$. Let $\mathcal{W}' = \{w \in \mathcal{W} : u_w(\alpha) = u_w(\beta)\}$. Since $\alpha \not\equiv_{\mathcal{W}} \beta$, $\mathcal{W}' \neq \mathcal{W}$. By the hypothesis, we assume that $\mathcal{W}'$ has measure zero. Let $\mathcal{W}'' = \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \cap \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$. Then for all $w \in \mathcal{W}''$, $u_w(\alpha) = u_w(\beta)$ so $\mathcal{W}'' \subseteq \mathcal{W}'$, and thus, $\mathcal{W}''$ has measure zero.

Lemma 3.3.14. *Let $A \in \mathcal{M}$ and let $\mathcal{W} \subseteq \mathcal{U} = \mathbb{R}^p$. Assume that for each $\alpha \in A$, $u_w(\alpha)$ is a continuous function of w. Consider any measure on $\mathcal{W}$ such that $\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha)$ is measurable for each $\alpha \in A$ and such that $\mathcal{W}$ has non-zero measure. We assume two further properties:*

(a) $\mathcal{W}$ is such that, for any open ball S in $\mathbb{R}^p$ intersecting with $\mathcal{W}$, $S\cap\mathcal{W}$ has non-zero measure; and

(b) for any $\alpha, \beta \in A$, if $\mathcal{W}' = \{w \in \mathcal{W} : u_w(\alpha) = u_w(\beta)\}$ then either $\mathcal{W}' = \mathcal{W}$ or $\mathcal{W}'$ has measure zero.

Then $\mathrm{PSO}_{\mathcal{W}}(A)$ equals the set of elements of A such that $\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha)$ has non-zero measure. If A is $\equiv_{\mathcal{W}}$-free then there exists a unique setwise-minimal equivalent subset for A, i.e., $\mathrm{SME}_{\mathcal{W}}(A)$ is a singleton, and this equals $\mathrm{PSO}_{\mathcal{W}}(A)$.

Proof: Again, let $\widehat{\mathrm{A}}$ be the set of elements α of A such that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has non-zero measure. Lemma 3.3.13(i) implies that $\bigcup_{\alpha\in\widehat{\mathrm{A}}}\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathcal{W}$. By Lemma 3.3.3, this implies that $\widehat{\mathrm{A}} \equiv^{\mathcal{W}}_{\forall\forall\exists}$ A.

Consider any $\alpha \in \widehat{\mathrm{A}}$. Let B_α be the set of $\beta \in$ A such that $\alpha \not\equiv_{\mathcal{W}} \beta$, and let C_α be $\bigcup_{\beta\in\mathrm{B}_\alpha}\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$. Lemma 3.3.13(ii) implies that for $\beta \in \mathrm{B}_\alpha$, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \cap \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$ has measure zero, and thus, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \cap \mathrm{C}_\alpha$ has measure zero. Since $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has non-zero measure, this implies that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \not\subseteq \mathrm{C}_\alpha$. Choose some $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \setminus$ C, which implies that $\beta \in \mathrm{O}_w$(A) (if and) only if $\alpha \equiv_{\mathcal{W}} \beta$, and thus, $\alpha \in \mathrm{PSO}_{\mathcal{W}}$(A). We have then shown that $\widehat{\mathrm{A}} \subseteq \mathrm{PSO}_{\mathcal{W}}$(A).

Also, $\widehat{\mathrm{A}} \equiv^{\mathcal{W}}_{\forall\forall\exists}$ A implies, using Lemma 3.3.5, that for all $\alpha \in \mathrm{PSO}_{\mathcal{W}}$(A) there exists $\beta \in \widehat{\mathrm{A}}$ with $\beta \equiv_{\mathcal{W}} \alpha$. The definition of $\widehat{\mathrm{A}}$ then ensures that $\alpha \in \widehat{\mathrm{A}}$, so we have $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \subseteq \widehat{\mathrm{A}}$. Therefore we have $\widehat{\mathrm{A}} = \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. Hence, $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists}$ A, so, by Theorem 3.3.6, if A is $\equiv_{\mathcal{W}}$-free then $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is the unique setwise-minimal equivalent subset for A. □

We show that in certain very important classes of problem we do have $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A}$, leading (by Theorem 3.3.6) to $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ being the unique setwise-minimal equivalent subset for equivalence-free A. The result below covers the linear case in which $u_w(\alpha) = w \cdot \alpha$, but also much more general forms of utility function. This contrasts with the general case in which $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ may well not be equivalent to A; it is even easy to construct small discrete examples in which $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is empty; see e.g., Table 2 of [WO11].

Theorem 3.3.15. *Let $\Omega = \mathcal{U} = I\!R^p$ and let $\mathcal{W}$ be a convex subset of $\mathcal{U}$. Assume that for each $\alpha \in \Omega$, $\{u_w(\alpha) : w \in \mathcal{W}'\}$ is an analytic function of w, where $\mathcal{W}'$ is the smallest affine space containing $\mathcal{W}$. Assume that A ($\in \mathcal{M}$) is $\equiv_{\mathcal{W}}$-free. Then there exists a unique setwise-minimal equivalent subset for A, i.e., $SME_{\mathcal{W}}(A)$ is a singleton, and this equals $\mathrm{PSO}_{\mathcal{W}}(A)$, which equals the set of elements α of A such that $\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha)$ has the same dimension as $\mathcal{W}$.*

The assumption that $\mathcal{W}$ is convex can be very much weakened. For instance, we might assume that $\mathcal{W}$ is a subset of affine space $\mathcal{W}'$, and $\mathcal{W}$ is a subset of the closure of its interior.

As well as the formal proof, we include a proof sketch which gives the basic idea behind the proof for the linear case (and which extends to the general case).

Sketch of proof: Let *dim*($\mathcal{W}$) be the dimension of $\mathcal{W}$. For any B $\subseteq$ A, $\mathcal{W} \setminus \bigcup_{\alpha \in \mathrm{B}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is an open subset of $\mathcal{W}$, which implies that it is either empty or has dimension *dim*($\mathcal{W}$). It follows that $\bigcup_{\alpha \in \widehat{\mathrm{A}}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathcal{W}$, where $\widehat{\mathrm{A}}$ is the set of elements α of A such that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has dimension *dim*($\mathcal{W}$). By Lemma 3.3.3, we have $\widehat{\mathrm{A}} \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A}$, and Theorem 3.3.6 then implies that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \subseteq \widehat{\mathrm{A}}$.

We next show that $\widehat{\mathrm{A}} \subseteq \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. Let $\alpha \in \widehat{\mathrm{A}}$, and let C_α be $\bigcup_{\beta \in \mathrm{A} \setminus \{\alpha\}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$. For $\beta \in \mathrm{A} \setminus \{\alpha\}$ let $\mathcal{W}_{\alpha=\beta} = \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \cap \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$, so that $w \cdot (\alpha - \beta) = 0$ for all $w \in \mathcal{W}_{\alpha=\beta}$. If it were the case that *dim*($\mathcal{W}_{\alpha=\beta}$) = *dim*($\mathcal{W}$) then $w \cdot (\alpha - \beta) = 0$ for all $w \in \mathcal{W}$, and so $\alpha \equiv_{\mathcal{W}} \beta$, which contradicts A being $\equiv_{\mathcal{W}}$-free. Hence, *dim*($\mathcal{W}_{\alpha=\beta}$) < *dim*($\mathcal{W}$). This implies that *dim*($\mathrm{C}_\alpha \cap \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$) $\leq \max_{\beta \in \mathrm{A} \setminus \{\alpha\}}$ *dim*($\mathcal{W}_{\alpha=\beta}$) < *dim*($\mathcal{W}$) = *dim*($\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$), and thus, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \not\subseteq \mathrm{C}_\alpha$, so there exists w in $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \setminus \mathrm{C}_\alpha$. α is strictly optimal in scenario w, and thus, strictly possibly optimal, i.e., $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, showing that $\widehat{\mathrm{A}} \subseteq \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$.

Therefore, $\widehat{\mathrm{A}} = \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. and thus, $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A}$, so, by Theorem 3.3.6, $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is the unique setwise-minimal equivalent subset for A. □

Proof: By considering a standard measure on the smallest affine space con-

taining $\mathcal{W}$, we have that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is measurable for each $\alpha \in \mathrm{A}$ and $\mathcal{W}$ has non-zero measure. We will use Lemma 3.3.14. Also, (a) for any open ball S in $\mathbb{R}^p$ intersecting with $\mathcal{W}$, $S \cap \mathcal{W}$ has non-zero measure. Regarding condition (b) of Lemma 3.3.14, consider for any $\alpha, \beta \in \mathrm{A}$, the set $\mathcal{W}_{\alpha=\beta} = \{w \in \mathcal{W} : u_w(\alpha) = u_w(\beta)\}$. Let $g(w) = u_w(\alpha) - u_w(\beta)$ for $w \in \mathcal{W}'$, which is an analytic function on $\mathcal{W}'$. If $\mathcal{W}_{\alpha=\beta}$ has non-zero measure, then it contains an open subset of $\mathcal{W}'$ and so has dimension the same as $\mathcal{W}$. Thus, g is zero on an open subset of $\mathcal{W}'$ so is the zero function, since it is analytic. Thus, $\mathcal{W}_{\alpha=\beta} = \mathcal{W}$.

Lemma 3.3.14 implies that there exists a unique setwise-minimal equivalent subset for A, i.e., $\mathrm{SME}_{\mathcal{W}}(\mathrm{A})$ is a singleton, and this equals $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, which equals the set of elements of A such that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has non-zero measure, i.e., with dimension equal to that of $\mathcal{W}$. □

Lemma 3.3.16. *Assume that $\mathcal{W}$ is a convex subset of $\mathbb{R}^p$, and consider $A \in \mathcal{M}$ and for $w \in \mathbb{R}^p, \alpha \in \mathbb{R}^p$, $u_w(\alpha) = w \cdot \alpha$. Then* $\mathrm{SMPO}_{\mathcal{W}}(A) = \mathrm{PSO}_{\mathcal{W}}(A)$.

Proof: $\mathrm{SMPO}_{\mathcal{W}}(\mathrm{A}) \supseteq \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ follows from Lemma 3.3.9. Regarding the converse, consider any $\gamma \in \mathrm{SMPO}_{\mathcal{W}}(\mathrm{A})$ and let A′ consist of all $\alpha \in \mathrm{A}$ such that $\alpha \not\equiv_{\mathcal{W}} \gamma$. Then, by definition of $\mathrm{SMPO}_{\mathcal{W}}(\mathrm{A})$, for all $\alpha \in \mathrm{A}'$ there exists $w_\alpha \in \mathcal{W}$ such that $w_\alpha \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma) \setminus \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$. Let w equal $\frac{1}{|\mathrm{A}'|}\sum_{\alpha\in\mathrm{A}'} w_\alpha$. Consider any $\beta \in \mathrm{A}'$. Since, for any $\alpha \in \mathrm{A}'$, $w_\alpha \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$, $w_\alpha \cdot \gamma - w_\alpha \cdot \beta \geq 0$, and this is strictly positive if $\alpha = \beta$, since $w_\alpha \notin \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$. Thus, $w \cdot \gamma - w \cdot \beta = \frac{1}{|\mathrm{A}'|}\sum_{\alpha\in\mathrm{A}'}(w_\alpha \cdot \gamma - w_\alpha \cdot \beta) > 0$. This implies that $w \notin \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$ for all $\beta \in \mathrm{A}'$, and $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\gamma)$, showing that $\gamma \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. We have shown that $\mathrm{SMPO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, and hence, $\mathrm{SMPO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. □

From the theorem we can prove the following further equality, which is useful computationally. In particular, it implies Corollary 3.3.16.2, that $\mathrm{MPO}_{\mathcal{W}}$ is an Optimality Operator in the sense defined in [WRM15], since $\mathrm{MPO}_{\mathcal{W}}$ always satisfies the first and third axiom, and $\mathrm{PSO}_{\mathcal{W}}$ satisfies the first and second axiom, so all three axioms are satisfied if $\mathrm{PSO}_{\mathcal{W}}$ equals $\mathrm{MPO}_{\mathcal{W}}$. This enables a good deal of flexibility in the use of incremental algorithms for computing $\mathrm{MPO}_{\mathcal{W}}$. Proposition 2 of [WRM15] then implies, for instance, that a bottom-up incremental algorithm can be used to compute $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$.

Corollary 3.3.16.1. *With the conditions of Theorem 3.3.15, and with linear utility functions,* $\mathrm{PSO}_{\mathcal{W}}(A) = \mathrm{MPO}_{\mathcal{W}}(A)$.

Proof: Theorem 3.3.15 implies that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists}$ A and Lemma 3.3.16

implies that that for any B $\subseteq$ A, $\mathrm{SMPO}_{\mathcal{W}}(\mathrm{B}) = \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$. Lemma 3.3.10 then shows that $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. □

Corollary 3.3.16.2. *With the conditions of Theorem 3.3.15, and with linear utility functions,* $\mathrm{PSO}_{\mathcal{W}}$ *is an Optimality Operator on* $\mathcal{M}$ *(in the sense of [WRM15]), i.e., for all* $A, B \in \mathcal{M}$*:*

(i) $\mathrm{PSO}_{\mathcal{W}}(A) \subseteq A$

(ii) If $B \subseteq A$ *then* $\mathrm{PSO}_{\mathcal{W}}(A) \cap B \subseteq \mathrm{PSO}_{\mathcal{W}}(B)$

(iii) If $\mathrm{PSO}_{\mathcal{W}}(A) \subseteq B \subseteq A$ *then* $\mathrm{PSO}_{\mathcal{W}}(B) = \mathrm{PSO}_{\mathcal{W}}(A)$

Proof. (i): $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$ is by definition a subset of A.

(ii): From Theorem 3.3.6(ii) follows that $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ if and only if $\mathrm{A} \setminus \{\alpha\} \not\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$. Thus, for any subset B of A, we get that if $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ then $\mathrm{B} \setminus \{\alpha\} \not\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$, which implies that if α is also member of B, then $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$. Therefore, if $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ and $\alpha \in \mathrm{B}$, then $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$, i.e., $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \cap \mathrm{B} \subseteq \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$.

(iii): From Theorem 3.3.15 follows that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is the unique minimal equivalent subset of A. Thus, since $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{B} \subseteq \mathrm{A}$, $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is the unique minimal equivalent subset also for B, i.e., $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{PSO}_{\mathcal{W}}(\mathrm{B})$.

□

3.4 Setwise Max Regret

The condition $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ states that in every scenario, the set of alternatives A is at least as good as the set B. A natural related numerical measure is setwise max regret $\mathit{SMR}_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$ (See Section 2.3.2.1) since we have $\mathit{SMR}_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) \leq 0$ if and only if $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ (see Proposition 3.4.1 below).

The definitions and results from earlier sections (apart from Section 3.3.3), regarding $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$, SME, PO, PSO and UD, depended only on the orderings $\succcurlyeq_w$, for $w \in \mathcal{W}$, and so were ordinal, in the sense that they are not affected by any strictly monotonic transformations of each function u_w (which can be different for each w). However, this is not the case for *SMR*, which has much weaker invariance properties.

In contrast with Section 2.3.2.1, Here we consider an alternative definition of setwise minimax regret by replacing $\max_{w\in\mathcal{U}}$ with $\sup_{w\in\mathcal{U}}$ in Equation 2.19:

$$SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) = \sup_{w\in\mathcal{W}} (\mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w)) \tag{3.1}$$

We say that $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$ *is achieved* if there exists $w \in \mathcal{W}$ such that $\mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w) = SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$, so that then $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) = \max_{w\in\mathcal{W}} \mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w)$ which corresponds to the definition of setwise max regret in Section 2.3.2.1; this always happens, for instance, if for each $\alpha \in \Omega$, $u_w(\alpha)$ is a continuous function of w, and $\mathcal{W}$ is compact. The reason to consider a more generic definition is to highlight properties that are valid only if $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$ is achieved.

We give some further basic properties of the setwise maximum regret function below. Parts (i) and (ii) give decomposability properties, with (i) being more useful computationally. (ii) is a slight generalisation of Observation 4 in [VB09]. (iii) relates the function $SMR_{\mathcal{W}}$ with the relation $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$, and (iv) with the Possibly Optimal operator $\mathrm{PO}_{\mathcal{W}}$, and (v) with the Possibly Strictly Optimal operator $\mathrm{PSO}_{\mathcal{W}}$. Property (vi) enables pre-processing of the sets A and B.

Proposition 3. *Consider $A, B \in \mathcal{M}$ and $\mathcal{W} \subseteq \mathcal{U}$.*

(i) $SMR_{\mathcal{W}}(A,B) = \max_{\beta\in B} SMR_{\mathcal{W}}(A,\{\beta\})$

(ii) $SMR_{\mathcal{W}}(A,B) = \max_{\alpha\in \mathrm{PO}_{\mathcal{W}}(A)} SMR_{\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha)}(\{\alpha\},B)$.

(iii) $SMR_{\mathcal{W}}(A,B) \leq 0$ *if and only if* $A \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} B$.

(iv) If $SMR_{\mathcal{W}}(A,B)$ *is achieved then* $SMR_{\mathcal{W}}(A,B) \geq 0$ *if and only if* $\mathrm{PO}_{\mathcal{W}}(A\cup B)\cap B \neq \emptyset$.

(v) For equivalence-free A, and $\alpha \in A$, $SMR_{\mathcal{W}}(A\setminus\{\alpha\},\{\alpha\}) > 0$ *if and only if* $\mathrm{PSO}_{\mathcal{W}}(A) \ni \alpha$.

(vi) If $A' \equiv^{\mathcal{W}}_{\forall\forall\exists} A$ *and* $B' \equiv^{\mathcal{W}}_{\forall\forall\exists} B$ *then* $SMR_{\mathcal{W}}(A',B') = SMR_{\mathcal{W}}(A,B)$.

This proposition follows immediately from parts of the following lemma.

Lemma 3.4.1. *(i)* $SMR_{\mathcal{W}}(A,B)$ *is monotonically decreasing in A, monotonically increasing in B and monotonically increasing in* $\mathcal{W}$*, i.e., if* $A' \supseteq A$*, and* $B' \subseteq B$*, and* $\mathcal{W}' \subseteq \mathcal{W}$ *then* $SMR_{\mathcal{W}'}(A',B') \leq SMR_{\mathcal{W}}(A,B)$.

(ii) $SMR_{\mathcal{W}}(A,B) = \max_{\beta\in B} SMR_{\mathcal{W}}(A,\{\beta\})$

(iii) $SMR_{\mathcal{W}}(A,B) \leq 0$ *if and only if* $A \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} B$.

(iv) *If* $SMR_{\mathcal{W}}(A,B)$ *is achieved then* $SMR_{\mathcal{W}}(A,B) \geq 0$ *if and only if* $\mathrm{PO}_{\mathcal{W}}(A \cup B) \cap B \neq \emptyset$.

(v) *If* $A' \equiv^{\mathcal{W}}_{\forall\exists} A$ *and* $B' \equiv^{\mathcal{W}}_{\forall\exists} B$ *then* $SMR_{\mathcal{W}}(A',B') = SMR_{\mathcal{W}}(A,B)$.

(vi) *If* $A' \subseteq A$ *and* $B' \subseteq B$ *and* $A' \succcurlyeq^{\mathcal{W}}_{\forall\exists} A \setminus A'$ *and* $B' \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \setminus B'$ *then* $SMR_{\mathcal{W}}(A',B') = SMR_{\mathcal{W}}(A,B)$.

(vii) *If* $B' \subseteq B$ *and* $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \setminus B'$ *and* $SMR_{\mathcal{W}}(A,B) \geq 0$ *then* $SMR_{\mathcal{W}}(A,B') = SMR_{\mathcal{W}}(A,B)$.

(viii) $SMR_{\mathcal{W}}(A,B) = \max_{\alpha \in \mathrm{PO}_{\mathcal{W}}(A)} SMR_{\mathrm{Opt}^A_{\mathcal{W}}(\alpha)}(\{\alpha\},B)$.

(ix) *For equivalence-free A, and* $\alpha \in A$, $SMR_{\mathcal{W}}(A \setminus \{\alpha\}, \{\alpha\}) > 0$ *if and only if* $\alpha \in \mathrm{PSO}_{\mathcal{W}}(A)$.

Proof: (i): The fact that $\mathrm{A}' \supseteq \mathrm{A}$ and $\mathrm{B}' \subseteq \mathrm{B}$ implies that for all $w \in \mathcal{U}$, $SMR_w(\mathrm{A}',\mathrm{B}') \leq SMR_w(\mathrm{A},\mathrm{B})$. Then $SMR_{\mathcal{W}'}(\mathrm{A},\mathrm{B}) = \sup_{w \in \mathcal{W}'} SMR_w(\mathrm{A}',\mathrm{B}') \leq \sup_{w \in \mathcal{W}} SMR_w(\mathrm{A}',\mathrm{B}') \leq \sup_{w \in \mathcal{W}} SMR_w(\mathrm{A},\mathrm{B}) = SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$.

(ii): We have that $\max_{\beta \in \mathrm{B}} SMR_{\mathcal{W}}(\mathrm{A},\{\beta\}) = \max_{\beta \in \mathrm{B}} \sup_{w \in \mathcal{W}} SMR_w(\mathrm{A},\{\beta\})$, which equals $\sup_{w \in \mathcal{W}} \max_{\beta \in \mathrm{B}} SMR_w(\mathrm{A},\{\beta\}) = \sup_{w \in \mathcal{W}} SMR_w(\mathrm{A},\mathrm{B}) = SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$.

(iii): $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) \leq 0$, i.e., $\sup_{w \in \mathcal{W}} \mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w) \leq 0$, if and only if $\mathrm{Val}_{\mathrm{B}}(w) \leq \mathrm{Val}_{\mathrm{A}}(w)$ for all $w \in \mathcal{W}$, which is if and only if $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B}$, by Lemma 3.2.1.

(iv): Assume that $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$ is achieved; then $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) \geq 0$ if and only if there exists $w \in \mathcal{W}$ such that $SMR_w(\mathrm{A},\mathrm{B}) \geq 0$. Now, $SMR_w(\mathrm{A},\mathrm{B}) \geq 0$ if and only if $\max_{\beta \in \mathrm{B}} u_w(\beta) - \max_{\alpha \in \mathrm{A}} u_w(\alpha) \geq 0$, which is if and only if there exists $\beta \in \mathrm{B}$ such that for all $\gamma \in \mathrm{A} \cup \mathrm{B}$, $u_w(\beta) \geq u_w(\gamma)$. This implies that $SMR_w(\mathrm{A},\mathrm{B}) \geq 0$ if and only if i.e., $\mathrm{B} \cap \mathrm{O}_w(\mathrm{A} \cup \mathrm{B})$ is non-empty. Thus, $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) \geq 0$ if and only if $\mathrm{B} \cap \bigcup_{w \in \mathcal{W}} \mathrm{O}_w(\mathrm{A} \cup \mathrm{B})$ is non-empty, i.e., $\mathrm{PO}_{\mathcal{W}}(\mathrm{A} \cup \mathrm{B}) \cap \mathrm{B} \neq \emptyset$.

(v): From Lemma 3.2.1(ii) follows that $\mathrm{A} \equiv^{\mathcal{W}}_{\forall\exists} \mathrm{A}' \iff$ for all $w \in \mathcal{W}$, $\mathrm{Val}_{\mathrm{A}}(w) = \mathrm{Val}_{\mathrm{A}'}(w)$. Thus, $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) = \sup_{w \in \mathcal{W}} \mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w) = \sup_{w \in \mathcal{W}} \mathrm{Val}_{\mathrm{B}'}(w) - \mathrm{Val}_{\mathrm{A}'}(w) = SMR_{\mathcal{W}}(\mathrm{A}',\mathrm{B}')$.

(vi): From Lemma 3.3.11(i) follows that if $\mathrm{A}' \subseteq \mathrm{A}$ and $\mathrm{A}' \succcurlyeq \mathrm{A} \setminus \mathrm{A}'$ then $\mathrm{A} \equiv \mathrm{A}'$. Thus, from (v) follows that if $\mathrm{A}' \subseteq \mathrm{A}$ and $\mathrm{B}' \subseteq \mathrm{B}$ and $\mathrm{A}' \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{A} \setminus \mathrm{A}'$ and $\mathrm{B}' \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B} \setminus \mathrm{B}'$, then $SMR_{\mathcal{W}}(\mathrm{A}',\mathrm{B}') = SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$.

(vii): Assume that $\mathrm{B}' \subseteq \mathrm{B}$ and $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists} \mathrm{B} \setminus \mathrm{B}'$ and $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) \geq 0$. (ii) implies that $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}) = \max(SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B}'), SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B} \setminus \mathrm{B}'))$, and (iii) implies

that $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B} \setminus \mathrm{B}') \leq 0$. Thus, $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}') \geq SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B} \setminus \mathrm{B}')$ and so $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}') = SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$.

(viii): $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) = \sup_{w \in \mathcal{W}} \mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w)$, which can be written as $\max_{\alpha \in \mathrm{PO}_{\mathcal{W}}(\mathrm{A})} \sup_{w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)} \mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w)$, since $\bigcup_{\alpha \in \mathrm{PO}_{\mathcal{W}}(\mathrm{A})} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \bigcup_{\alpha \in \mathrm{A}} \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathcal{W}$, using Lemma 3.3.3. For $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$, $\mathrm{Val}_{\mathrm{A}}(w) = u_w(\alpha)$, so $\sup_{w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)} \mathrm{Val}_{\mathrm{B}}(w) - \mathrm{Val}_{\mathrm{A}}(w) = \sup_{w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)} \mathrm{Val}_{\mathrm{B}}(w) - u_w(\alpha)$, which equals $SMR_{\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)}(\{\alpha\}, \mathrm{B})$, showing the result.

(ix): Consider equivalence-free A, and $\alpha \in \mathrm{A}$. Then, $SMR_{\mathcal{W}}(\mathrm{A} \setminus \{\alpha\}, \{\alpha\}) > 0 \iff \sup_{w \in \mathcal{W}} u_w(\alpha) - \mathrm{Val}_{\mathrm{A} \setminus \{\alpha\}}(w) > 0$, which is if and only if there exists $w \in \mathcal{W}$ such that for all $\gamma \in \mathrm{A} \setminus \{\alpha\}$, $u_w(\alpha) > u_w(\gamma)$, which, since A is equivalence-free, holds if and only if $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \ni \alpha$. □

The following result shows that we can pre-process A and B using $\mathrm{UD}_{\mathcal{W}}$ and $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$. It follows easily using Proposition 3.

Lemma 3.4.2. *Consider any $A, B \in \mathcal{M}$.*

(i) $SMR_{\mathcal{W}}(\mathrm{UD}_{\mathcal{W}}(A), \mathrm{UD}_{\mathcal{W}}(B)) = SMR_{\mathcal{W}}(A, B)$.

(ii) If $B' \subseteq B$ and $A \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall} B \setminus B'$ and $SMR_{\mathcal{W}}(A, B) \geq 0$ then $SMR_{\mathcal{W}}(A, B') = SMR_{\mathcal{W}}(A, B)$.

Proof: (i): By Lemma 3.3.2, $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$ and $\mathrm{UD}_{\mathcal{W}}(\mathrm{B}) \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$. Then Lemma 3.4.1(v) implies the result.

(ii): Assume $\mathrm{B}' \subseteq \mathrm{B}$ and $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall} \mathrm{B} \setminus \mathrm{B}'$ and $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) \geq 0$. Lemma 3.4.1(ii) implies $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) = \max(SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}'), SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B} \setminus \mathrm{B}'))$. Because $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\exists\forall} \mathrm{B} \setminus \mathrm{B}'$, we have by Lemma 3.4.1(iii) that $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B} \setminus \mathrm{B}') \leq 0$. Thus, $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B} \setminus \mathrm{B}') \leq SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}')$ and therefore, $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}') = SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$. □

3.5 Implication for Incremental Preference Elicitation

Let α and β be alternatives. Preference model w is said to satisfy a preference statement $\alpha \geq \beta$ if $u_w(\alpha) \geq u_w(\beta)$, i.e., α is at least as good as β given w. For set of alternatives A the preference statement $\alpha \geq \mathrm{A}$ means $\alpha \geq \beta$ for all $\beta \in \mathrm{A}$. Thus, for $\alpha \in \mathrm{A}$, w satisfies $\alpha \geq \mathrm{A}$ if and only if (given w) α is a most preferred

element in A, $\alpha \in \mathrm{O}_w(\mathrm{A})$, i.e., w makes α optimal in A. This holds if and only if $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$.

In incremental elicitation a common strategy is to generate a small set of alternatives A, and to ask the user which element of A is most preferred. If they reply "α" then this is interpreted as $\alpha \geq \mathrm{A}$. We will then update $\mathcal{W}$ to the set of all $w \in \mathcal{W}$ such that α is a most preferred option in A, i.e., we update $\mathcal{W}$ to $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$.

There can be forms of inconsistency, of different kinds, between the user answers and the model we have of the user.

Feasible answer: We say that, given set of preference models $\mathcal{W}$, alternative α is a *feasible answer to query A* if $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is non-empty, i.e., there exists some user preference model in $\mathcal{W}$ under which α is optimal in A.

Strongly feasible answer: We say that α is a *strongly feasible answer to query A* (given $\mathcal{W}$) if $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has the same dimension as $\mathcal{W}$.

Example 13. *In the example in Figure 3.3, with the query* $A = \{(10,4),(4,7),(6,6),(5,5)\}$*, the alternatives* $(10,4)$ *and* $(4,7)$ *are strongly feasible answers, and* $(6,6)$ *is a feasible answer, and* $(5,5)$ *is not a feasible answer.*

The following result, which follows from Theorem 3.3.15, characterises *strongly* feasible answers to queries.

Proposition 4. *Consider* $A \in \mathcal{M}$ *and* $\mathcal{W} \subseteq \mathcal{U}$*.*

(i) α *is a feasible answer to query A given* $\mathcal{W}$ *if and only if* $\alpha \in \mathrm{PO}_{\mathcal{W}}(A)$*.*

(ii) Under the conditions of Theorem 3.3.15 on Ω*,* $\mathcal{U}$*,* $\mathcal{W}$ *and* u *we have that* α *is a strongly feasible answer to query A given* $\mathcal{W}$ *if and only if* $\alpha \in \mathrm{PSO}_{\mathcal{W}}(A)$*.*

Proof: (i): α is a feasible answer to query A given $\mathcal{W}$ if and only if $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is non-empty, i.e., $\alpha \in \mathrm{PO}_{\mathcal{W}}(\mathrm{A})$.

(ii): α is a strongly feasible answer to query A given $\mathcal{W}$ if and only if $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has the same dimension as $\mathcal{W}$. If A is equivalence-free then, by Theorem 3.3.15, this is if and only if $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$.

More generally, it is sufficient to show that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has the same dimension as $\mathcal{W}$ if and only if $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$.

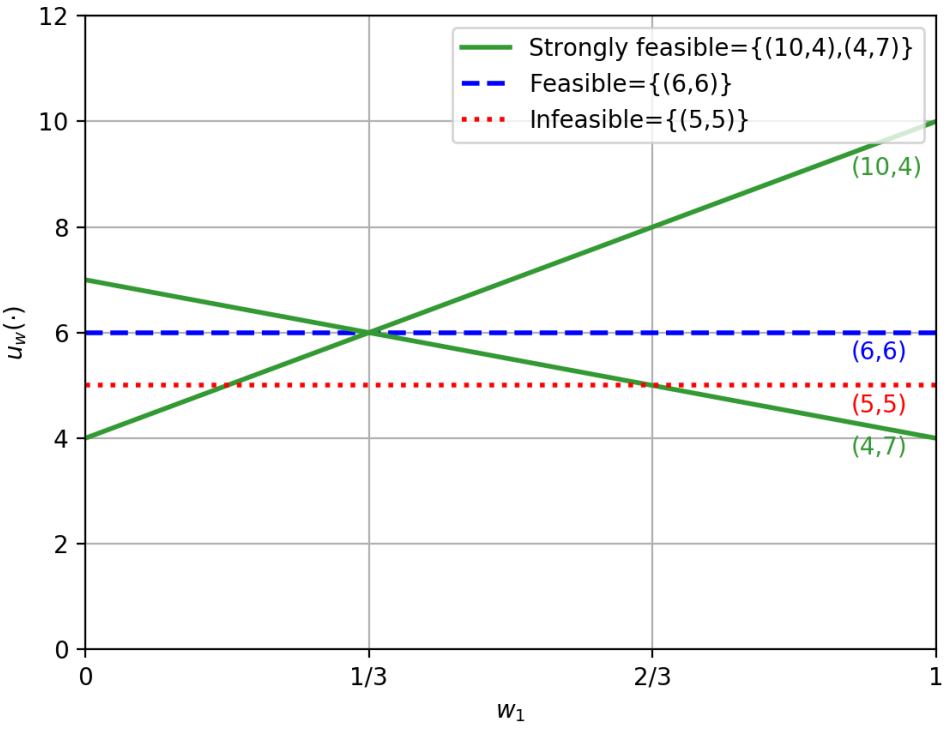


Figure 3.3: utility function $u_w(\cdot)$ for each alternative in A $= \{(10,4),\ (4,7),\ (6,6),\ (5,5)\}$, where $w \in \mathcal{U} = \{(w_1, w_2) : w_1 + w_2 = 1\}$. The alternative $(10,4)$ and $(4,7)$ are strongly feasible answers, and $(6,6)$ is a feasible answer, and $(5,5)$ is not a feasible answer

Choose an equivalence-free subset A′ of A containing α such that every element of A is equivalent with some element of A′. Then $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathrm{Opt}^{\mathrm{A}'}_{\mathcal{W}}(\alpha)$. Hence, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has the same dimension as $\mathcal{W}$ if and only if $\mathrm{Opt}^{\mathrm{A}'}_{\mathcal{W}}(\alpha)$ has the same dimension as $\mathcal{W}$, which, by Theorem 3.3.15, is if and only if $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A}')$, which is if and only if $\alpha \in \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, as required. □

If the user chooses α from A, and α is not a feasible answer to A, then we get an inconsistency since the updated $\mathcal{W}$ will be empty. Suppose now, on the other hand, α is not a strongly feasible answer to A. We can still consistently update $\mathcal{W}$, so this is a less strong kind of inconsistency; however, such an answer would still be seriously troubling. For instance, suppose $\mathcal{W} \subseteq \mathbb{R}^p$, and consider any probability distribution over $\mathcal{W}$, regarding which is the true user model w, such that (as one would expect) the probability distribution is compatible with the measure of the sets. If α is not a strongly feasible answer to query A then the probability that w is such that $\alpha \geq$ A holds would be zero (since $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ has then measure zero in $\mathcal{W}$, being of lower dimension than $\mathcal{W}$). A choice, by the

user, of α would hence correspond with an event of probability zero.

To ensure that every answer to a query A is feasible, we thus require that $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{A}$. And, to ensure that every answer to A is strongly feasible, we require that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{A}$, i.e., that every element of A is strictly possibly optimal in A.

We thus argue that the standard methods for generating queries in incremental preference learning should be modified to ensure that every element in the query set is strictly possibly optimal. Learning an inconsistency could in theory be useful information, allowing the potential of updating the model in some way to restore consistency; however, this would probably have a heavy computational cost, and in a practical application, one will want to avoid the incremental elicitation procedure breaking down. Since Theorem 3.3.15 implies that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is non-empty, (and indeed equivalent to A) we can therefore replace a potential query A by $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$.

As we have discussed in Section 2.3.2.2, choosing the subset A of a specific cardianlity k, of the set of available alternatives B, that minimises the setwise regret *SMR*$_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$ is a desirable and well-founded choice for an informative query. However, it can easily happen that, for such a query A, we have $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) \neq \mathrm{A}$ and even $\mathrm{PO}_{\mathcal{W}}(\mathrm{A}) \neq \mathrm{A}$. Such a choice of A is then in danger of leading to an inconsistency, as described above. Fortunately, one can easily solve this problem by replacing A by $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, since if A maximises setwise regret then $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ also maximises setwise regret (under the conditions in Theorem 3.3.15 on Ω, $\mathcal{U}$, $\mathcal{W}$ and u) because *SMR*$_{\mathcal{W}}(\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}), \mathrm{B}) =$ *SMR*$_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$, by Theorem 3.3.15 and Proposition 3.4.1.

3.6 EEU Method for Testing A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B and Computing *SMR*$_{\mathcal{W}}$(A, B)

Computing the extreme points of $\mathcal{W}$ can lead for the linear case to an easy way of testing if $\alpha \succcurlyeq_{\mathcal{W}} \beta$ (for $\alpha, \beta \in \mathbb{R}^p$): it is easy to see that $\alpha \succcurlyeq_{\mathcal{W}} \beta$ holds if and only if for each extreme point w of $\mathcal{W}$, we have $w \cdot (\alpha - \beta) \geq 0$ [KVVA17]. Similarly, it follows immediately that standard maximum regret over the convex polytope $\mathcal{W}$ can be computed using the extreme points of $\mathcal{W}$, as observed e.g., in [Tim13]. However, for setwise max regret it is not sufficient to consider the extreme points of $\mathcal{W}$. Here we develop a novel extreme points

method for testing A $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ B and computing $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$, by moving to a higher dimensional space.

Epigraph $\gamma(\mathcal{W},\mathbf{A})$ of the utility function $\mathbf{Val_A}$ on $\mathcal{W}$: Given $\mathcal{W}$, the utility function $\mathrm{Val}_\mathrm{A}(w)$ (over $w \in \mathcal{W}$) can be viewed as a subset of $\mathcal{W} \times \mathbb{R}$, and we can test A $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ B by considering such subsets. Let us define $\gamma(\mathcal{W},\mathrm{A}) \subseteq \mathcal{W} \times \mathbb{R} \subseteq \mathbb{R}^p \times \mathbb{R}$ to be $\{(w,r) : w \in \mathcal{W}, r \geq \mathrm{Val}_\mathrm{A}(w)\}$, i.e., the *epigraph* [BV04] of the utility function Val_A on $\mathcal{W}$. If $\mathcal{W}$ is convex and compact and for all $\alpha \in$ A, $u_w(\alpha)$ is a convex and continuous function of $w \in \mathcal{W}$, then $\gamma(\mathcal{W},\mathrm{A})$ is a closed convex set. We write *Ext*$(\gamma(\mathcal{W},\mathrm{A}))$ for the extreme points of $\gamma(\mathcal{W},\mathrm{A})$. For any $\mathcal{W} \subseteq \mathcal{U}$, and any $\alpha \in \mathbb{R}^p$ we define $J^\alpha_\mathcal{W}$ to be the set $\{(w,r) : w \in \mathcal{W} \;\&\; r \geq u_w(\alpha)\}$.

We first give two basic lemmas involving the set $J^\alpha_\mathcal{W}$ defined above.

Lemma 3.6.1. *Define $J^\alpha_\mathcal{W} = \{(w,r) : w \in \mathcal{W}, r \geq u_w(\alpha)\}$.*

(i) If $\mathcal{W}$ is closed and $u_w(\alpha)$ is continuous with respect to w then $J^\alpha_\mathcal{W}$ is a closed subset of $\mathbb{R}^p \times \mathbb{R}$.

(ii) If $\mathcal{W}$ is convex and $u_w(\alpha)$ is a convex function of w then $J^\alpha_\mathcal{W}$ is convex. ($u_w(\alpha)$ is a convex function means that for any $s \in (0,1)$, and any $w_1, w_2 \in \mathbb{R}^p$, $u_w(\alpha) \leq s u_{w_1}(\alpha) + (1-s)u_{w_2}(\alpha)$ where $w = sw_1 + (1-s)w_2$.)

Proof: (i): Showing $J^\alpha_\mathcal{W}$ is closed if $\mathcal{W}$ is closed and $u_w(\alpha)$ is continuous: Suppose $(w,r) \notin J^\alpha_\mathcal{W}$. Either (a) $w \notin \mathcal{W}$ or (b) $r < u_w(\alpha)$. If (a) $w \notin \mathcal{W}$ then the fact that $\mathcal{W}$ is closed implies that there's an open ball S in $\mathbb{R}^p$ containing w and such that $S \cap \mathcal{W} = \emptyset$. Let $S' = S \times (a,b)$ for any open interval (a,b) of $\mathbb{R}$ containing r. Then S' is an open ball in $\mathbb{R}^p \times \mathbb{R}$, and $(w,r) \in S'$ and $S' \cap J^\alpha_\mathcal{W} = \emptyset$.

If (b) $r < u_w(\alpha)$: let $\epsilon = \frac{1}{3}(u_w(\alpha) - r)$. By continuity of $u_w(\alpha)$, there exists $\delta > 0$ such that for all $w' \in \mathbb{R}^p$ with $|w' - w| < \delta$ we have $|u_{w'}(\alpha) - u_w(\alpha)| < \epsilon$. Let $S_\delta = \{w' \in \mathbb{R}^p : |w' - w| < \delta\}$. Let $S'' = S_\delta \times (r-\epsilon, r+\epsilon)$ which contains (w,r). For any $(w',r') \in S''$, $|u_{w'}(\alpha) - u_w(\alpha)| < \epsilon$ and $|r' - r| < \epsilon$ so $|u_{w'}(\alpha) - r'| > \epsilon$, since $|u_w(\alpha) - r| = 3\epsilon$. Thus, S'' is an open set containing (w,r) that is disjoint from $J^\alpha_\mathcal{W}$.

We've shown that for any $(w,r) \notin J^\alpha_\mathcal{W}$ there exists an open set containing (w,r) that is disjoint from $J^\alpha_\mathcal{W}$, which proves that $(\mathbb{R}^p \times \mathbb{R}) \setminus J^\alpha_\mathcal{W}$ is an open subset of $\mathbb{R}^p \times \mathbb{R}$, and thus, $J^\alpha_\mathcal{W}$ is a closed subset.

(ii): Showing $J^\alpha_\mathcal{W}$ is convex if $\mathcal{W}$ is convex and $u_w(\alpha)$ is a convex function: Consider any $(w_1,r_1),(w_2,r_2) \in J^\alpha_\mathcal{W}$, let $s \in (0,1)$ and let $(w_0,r_0) = s(w_1,r_1) +$

$(1-s)(w_2,r_2)$, so that $w_0 = sw_1 + (1-s)w_2$ and $r_0 = sr_1 + (1-s)r_2$. Since, $(w_1,r_1),(w_2,r_2) \in J^{\alpha}_{\mathcal{W}}$ we have $r_1 \geq u_{w_1}(\alpha)$ and $r_2 \geq u_{w_2}(\alpha)$. Since $u_w(\alpha)$ is a convex function of w we have $u_{w_0}(\alpha) \leq su_{w_1}(\alpha)+(1-s)u_{w_2}(\alpha) \leq sr_1+(1-s)r_2 = r_0$. Thus, $(w_0,r_0) \in J^{\alpha}_{\mathcal{W}}$, proving that $J^{\alpha}_{\mathcal{W}}$ is convex. □

Lemma 3.6.2. *Consider any finite subset A of $\mathbb{R}^p$, and any $\mathcal{W} \subseteq \mathcal{U}$, and any $\alpha \in \mathbb{R}^p$. Define $J^{\alpha}_{\mathcal{W}} = \{(w,r) : w \in \mathbb{R}^p, r \geq u_w(\alpha)\}$. Then:*

(i) $\gamma(\mathcal{W},\{\alpha\}) = \{(w,r) : w \in \mathcal{W} \;\&\; r \geq u_w(\alpha)\} = (\mathcal{W} \times \mathbb{R}) \cap J^{\alpha}_{\mathcal{W}}$;

(ii) $\gamma(\mathcal{W},A) = \bigcap_{\alpha\in A} \gamma(\mathcal{W},\{\alpha\}) = (\mathcal{W} \times \mathbb{R}) \cap \bigcap_{\alpha\in A} J^{\alpha}_{\mathcal{W}}$.

(iii) If $\mathcal{W}$ is closed and for all $\alpha \in A$, $u_w(\alpha)$ is continuous then $\gamma(\mathcal{W},A)$ is closed.

(iv) If $\mathcal{W}$ is convex and for all $\alpha \in A$, $u_w(\alpha)$ is a convex function of w then $\gamma(\mathcal{W},A)$ is convex.

(v) If $\mathcal{W}$ is a convex polytope and for $w \in \mathbb{R}^p, \alpha \in \mathbb{R}^p$, $u_w(\alpha) = w \cdot \alpha$ then $\gamma(\mathcal{W},A)$ is a convex polytope.

Proof: (i): $(w,r) \in \gamma(\mathcal{W},\{\alpha\})$ if and only if $w \in \mathcal{W}$ and $r \geq \text{Val}_{\{\alpha\}}(w) = u_w(\alpha)$, so $\gamma(\mathcal{W},\{\alpha\}) = (\mathcal{W} \times \mathbb{R}) \cap J^{\alpha}_{\mathcal{W}}$.

(ii): Given $w \in \mathcal{W}$, we have $(w,r) \in \gamma(\mathcal{W},\text{A})$ if and only if and $r \geq \text{Val}_{\text{A}}(w) = \max_{\alpha\in\text{A}} u_w(\alpha)$ if and only if for all $\alpha \in$ A, $r \geq u_w(\alpha)$, which, by part (i), is if and only if for all $\alpha \in$ A, $(w,r) \in \gamma(\mathcal{W},\{\alpha\})$. Thus, $\gamma(\mathcal{W},\text{A}) = \bigcap_{\alpha\in\text{A}} \gamma(\mathcal{W},\{\alpha\}) = (\mathcal{W} \times \mathbb{R}) \cap \bigcap_{\alpha\in\text{A}} J^{\alpha}_{\mathcal{W}}$.

(iii): Assume that $\mathcal{W}$ is closed and for all $\alpha \in$ A, $u_w(\alpha)$ is continuous. Lemma 3.6.1(i) implies that for all $\alpha \in$ A, $J^{\alpha}_{\mathcal{W}}$ is closed, which implies that $\gamma(\mathcal{W},\text{A})$ is closed using (ii), since an intersection of closed sets is closed.

(iv): Assume that $\mathcal{W}$ is convex and for all $\alpha \in$ A, $u_w(\alpha)$ is a convex function of w. Then Lemma 3.6.1(ii) implies that for all $\alpha \in$ A, $J^{\alpha}_{\mathcal{W}}$ is convex, which implies that $\gamma(\mathcal{W},\text{A})$ is convex using (ii), since an intersection of convex sets is convex.

(v): Assume that $\mathcal{W}$ is a convex polytope and for $w \in \mathbb{R}^p, \alpha \in \mathbb{R}^p$, $u_w(\alpha) = w\cdot\alpha$. Then, for each $\alpha \in$ A, $J^{\alpha}_{\mathcal{W}}$ is a closed half-space and thus a convex polytope. Therefore, using (ii), $\gamma(\mathcal{W},\text{A})$ is a convex polytope, since it is the intersection of a finite number of convex polytopes. □

The following technical lemma is important in proving Lemmas 3.6.4 and 3.6.5 and thus Theorem 3.6.6.

For $E \subseteq \mathbb{R}^p$ we let $CH(E)$ be the convex hull of E.

Lemma 3.6.3. *Assume that $\mathcal{W}$ is a compact and convex subset of $\mathbb{R}^p$, and for all $\alpha \in A$, $u_w(\alpha)$ is a convex and continuous function of $w \in \mathcal{W}$. There exists real value N such that for all $w \in \mathcal{W}$, $N > Val_A(w)$. Define $\gamma_N(\mathcal{W},A)$ to be the intersection of $\gamma(\mathcal{W},A)$ with the half-space $\{(w,r) : r \leq N\}$. Assume that $\mathcal{W}$ is a convex polytope. Then*

(i) $\gamma_N(\mathcal{W},A)$ *is compact and convex.*

(ii) $CH(Ext(\gamma_N(\mathcal{W},A))) = \gamma_N(\mathcal{W},A)$.

(iii) If $(w,r) \in Ext(\gamma(\mathcal{W},A))$ *then* $r = Val_A(w)$.

(iv) $Ext(\gamma(\mathcal{W},A)) \subseteq \gamma_N(\mathcal{W},A)$.

(v) $Ext(\gamma(\mathcal{W},A)) \subseteq Ext(\gamma_N(\mathcal{W},A))$.

(vi) If $(w,r) \in Ext(\gamma_N(\mathcal{W},A)) \setminus Ext(\gamma(\mathcal{W},A))$ *then* $r = N$.

Proof: Firstly, $\mathrm{Val_A}(w)$ is a continuous function on compact set $\mathcal{W}$ so is bounded; we can therefore choose N and M such that for all $w \in \mathcal{W}$, $M < \mathrm{Val_A}(w) < N$.

(i): $\gamma_N(\mathcal{W},\mathrm{A})$ is bounded and thus compact, since it is a subset of the cylinder $\mathcal{W} \times [M,N]$, and $\mathcal{W}$ is compact. $\gamma_N(\mathcal{W},\mathrm{A})$ is therefore a compact and convex.

(ii): Using a well-known property of extreme points (e.g., the Krein–Milman theorem [Fan63]), (i) implies $CH(Ext(\gamma_N(\mathcal{W},\mathrm{A}))) = \gamma_N(\mathcal{W},\mathrm{A})$.

(iii): This follows from the fact that the line $\{(w,r') : r' \geq \mathrm{Val_A}(w)\}$ is a subset of $\gamma(\mathcal{W},\mathrm{A})$.

(iv): Since $N > \mathrm{Val_A}(w)$, from (iii) follows that $Ext(\gamma(\mathcal{W},\mathrm{A})) \subseteq \gamma_N(\mathcal{W},\mathrm{A})$.

(v): From (iv) follows that if it were the case that $(w,r) \in Ext(\gamma(\mathcal{W},\mathrm{A})) \setminus Ext(\gamma_N(\mathcal{W},\mathrm{A}))$, then there would exist a line segment containing (w,r) as inner point, within $\gamma_N(\mathcal{W},\mathrm{A})$, and thus, within $\gamma(\mathcal{W},\mathrm{A})$, which contradicts $(w,r) \in Ext(\gamma(\mathcal{W},\mathrm{A}))$. Therefore, $Ext(\gamma(\mathcal{W},\mathrm{A})) \subseteq Ext(\gamma_N(\mathcal{W},\mathrm{A}))$.

(vi): Assume that $(w,r) \in Ext(\gamma_N(\mathcal{W},\mathrm{A})) \setminus Ext(\gamma(\mathcal{W},\mathrm{A}))$. Now, $(w,r) \in \gamma(\mathcal{W},\mathrm{A}) \setminus Ext(\gamma(\mathcal{W},\mathrm{A}))$, so there exists a line segment L within $\gamma(\mathcal{W},\mathrm{A})$ which has (w,r) as an internal point. Let L' be the line segment intersected with the half-space $\{(w,r) : r \leq N\}$. Then $L' \subseteq \gamma_N(\mathcal{W},\mathrm{A})$. Since $(w,r) \in Ext(\gamma_N(\mathcal{W},\mathrm{A}))$, (w,r) cannot be an internal point of L', and so r must equal N. □

Lemma 3.6.4. *Assume that $\mathcal{W}$ is a compact and convex subset of $\mathbb{R}^p$, and for all $\alpha \in A$, $u_w(\alpha)$ is a convex and continuous function of $w \in \mathcal{W}$. Let $\Theta(\mathcal{W},A) = \{(w, Val_A(w)) : w \in \mathcal{W}\}$. Then CH(Ext($\gamma(\mathcal{W},A)$)) contains $\Theta(\mathcal{W},A)$, i.e., for all $w \in \mathcal{W}$, CH(Ext($\gamma(\mathcal{W},A)$)) contains $(w, Val_A(w))$.*

Proof: Consider any $w \in \mathcal{W}$. By definition $(w, \mathrm{Val_A}(w))$ is a point in $\gamma(\mathcal{W}, \mathrm{A})$. Lemma 3.6.3 implies the existence of N such that for all $w \in \mathcal{W}$, $N > \mathrm{Val_A}(w)$ and with $(w, \mathrm{Val_A}(w)) \in \gamma_N(\mathcal{W}, \mathrm{A})$. Using Lemma 3.6.3(ii), *CH*(*Ext*($\gamma_N(\mathcal{W}, \mathrm{A})$)) $\ni$ $(w, \mathrm{Val_A}(w))$ so we can write $(w, \mathrm{Val_A}(w))$ as $\sum_{j=1}^{J} \tau_j q_j$ where $q_j \in$ *Ext*($\gamma_N(\mathcal{W}, \mathrm{A})$), and the τ_j are non-negative reals that sum to 1. We will show that $\tau_j = 0$ unless $q_j \in$ *Ext*($\gamma(\mathcal{W}, \mathrm{A})$). This then implies that $(w, \mathrm{Val_A}(w)) \in$ *CH*(*Ext*($\gamma(\mathcal{W}, \mathrm{A})$)), proving Lemma 3.6.4.

So, suppose that there exists k with $\tau_k > 0$ and $q_k \in$ *CH*(*Ext*($\gamma_N(\mathcal{W}, \mathrm{A})$)) $\setminus$ *Ext*($\gamma(\mathcal{W}, \mathrm{A})$). By Lemma 3.6.3(vi), $q_k = (w', N)$ for some $w' \in \mathcal{W}$ and $N > \mathrm{Val_A}(w')$. Let $q' = \sum_{j=1}^{J} \tau_j q'_j$, where $q'_k = (w', \mathrm{Val_A}(w'))$, and $q'_j = q_j$ for $j \neq k$. Then for all j, $q'_j \in \gamma(\mathcal{W}, \mathrm{A})$ and so, by convexity of $\gamma(\mathcal{W}, \mathrm{A})$, $q' \in \gamma(\mathcal{W}, \mathrm{A})$. Also, q' can be written as (w, r') for some r', and $r' < \mathrm{Val_A}(w)$. The definition of $\gamma(\mathcal{W}, \mathrm{A})$ implies that $(w, r') \notin \gamma(\mathcal{W}, \mathrm{A})$, giving the required contradiction. □

The following result states that $\gamma(\mathcal{W}, \mathrm{A})$ is determined by its extreme points, even though it is not compact.

Lemma 3.6.5. *Consider any finite subsets A and B of $\mathbb{R}^p$, and any compact and convex subset $\mathcal{W}$ of $\mathbb{R}^p$, and assume that for all $\alpha \in A \cup B$, $u_w(\alpha)$ is a convex and continuous function of $w \in \mathcal{W}$. Then $\gamma(\mathcal{W},A) = \gamma(\mathcal{W},B) \iff$ Ext($\gamma(\mathcal{W},A)$) = Ext($\gamma(\mathcal{W},B)$).*

Proof: If $\gamma(\mathcal{W}, \mathrm{A}) = \gamma(\mathcal{W}, \mathrm{B})$ then obviously *Ext*($\gamma(\mathcal{W}, \mathrm{A})$) = *Ext*($\gamma(\mathcal{W}, \mathrm{B})$). Regarding the converse, assume that *Ext*($\gamma(\mathcal{W}, \mathrm{A})$) = *Ext*($\gamma(\mathcal{W}, \mathrm{B})$). For any $w \in \mathcal{W}$, $(w, \mathrm{Val_A}(w)) \in \gamma(\mathcal{W}, \mathrm{A})$ and $(w, r) \in \gamma(\mathcal{W}, \mathrm{A})$ if and only if $r \geq \mathrm{Val_A}(w)$. Now, Lemma 3.6.4 implies that $(w, \mathrm{Val_B}(w))$ is in the convex hull of *Ext*($\gamma(\mathcal{W}, \mathrm{B})$), and thus, in the convex hull of *Ext*($\gamma(\mathcal{W}, \mathrm{A})$); hence, $(w, \mathrm{Val_B}(w)) \in \gamma(\mathcal{W}, \mathrm{A})$, which shows that $\mathrm{Val_B}(w) \geq \mathrm{Val_A}(w)$, which holds for an arbitrary element w of $\mathcal{W}$. This implies $\gamma(\mathcal{W}, \mathrm{A}) \supseteq \gamma(\mathcal{W}, \mathrm{B})$. Switching the roles of A and B in the argument shows also $\gamma(\mathcal{W}, \mathrm{A}) \subseteq \gamma(\mathcal{W}, \mathrm{B})$, and thus, $\gamma(\mathcal{W}, \mathrm{A}) = \gamma(\mathcal{W}, \mathrm{B})$. □

The following result leads to two different ways of testing A $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ B. Firstly, we can compute the extreme points of both $\gamma(\mathcal{W}, \mathrm{A})$ and $\gamma(\mathcal{W}, \mathrm{A} \cup \mathrm{B})$; by (ii), these two sets of extreme points are equal if and only if A $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ B. Alternatively,

we can test A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B, using part (iii), after computing *Ext*$(\gamma(\mathcal{W},\mathrm{A}))$. We can compute the pairwise max regret $SMR_{\mathcal{W}}(\mathrm{A},\mathrm{B})$ as $\max_{\beta\in\mathrm{B}} SMR_{\mathcal{W}}(\mathrm{A},\{\beta\})$ (see Lemma 3.4.1), and use part (iv) below.

Theorem 3.6.6. *Consider any finite subsets A and B of $\mathbb{R}^p$, any $\beta \in \mathbb{R}^p$, and any compact and convex subset $\mathcal{W}$ of $\mathbb{R}^p$, and assume that for all $\alpha \in A \cup B \cup \{\beta\}$, $u_w(\alpha)$ is a convex and continuous function of $w \in \mathcal{W}$.*

(i) $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B \iff \gamma(\mathcal{W},A) \subseteq \gamma(\mathcal{W},B) \iff \gamma(\mathcal{W},A) = \gamma(\mathcal{W},A\cup B)$.

(ii) $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$ *if and only if* *Ext*$(\gamma(\mathcal{W},A)) =$ *Ext*$(\gamma(\mathcal{W},A\cup B))$.

(iii) $A \succcurlyeq^{\mathcal{W}}_{\forall\exists} B$ *holds if and only if for all* $(w,r) \in$ *Ext*$(\gamma(\mathcal{W},A))$ *and for all* $\beta \in B$ *we have* $u_w(\beta) \le r$.

(iv) $SMR_{\mathcal{W}}(A,\{\beta\}) = \max\{u_w(\beta) - r : (w,r) \in$ *Ext*$(\gamma(\mathcal{W},A))\}$.

Proof: Regarding (i): Lemma 3.2.1(i) implies that A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B $\iff$ for all $w \in \mathcal{W}$, $\mathrm{Val}_{\mathrm{A}}(w) \ge \mathrm{Val}_{\mathrm{B}}(w)$, which is if and only if $\gamma(\mathcal{W},\mathrm{A}) \subseteq \gamma(\mathcal{W},\mathrm{B})$. Using Lemma 3.6.2, $\gamma(\mathcal{W},\mathrm{A}\cup\mathrm{B}) = \gamma(\mathcal{W},\mathrm{A})\cap\gamma(\mathcal{W},\mathrm{B})$. This implies $\gamma(\mathcal{W},\mathrm{A}) \subseteq \gamma(\mathcal{W},\mathrm{B}) \iff \gamma(\mathcal{W},\mathrm{A}) = \gamma(\mathcal{W},\mathrm{A}\cup\mathrm{B})$.

(ii): By (i), A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B if and only if $\gamma(\mathcal{W},\mathrm{A}) = \gamma(\mathcal{W},\mathrm{A}\cup\mathrm{B})$, which, by Lemma 3.6.5 is if and only if *Ext*$(\gamma(\mathcal{W},\mathrm{A})) =$ *Ext*$(\gamma(\mathcal{W},\mathrm{A}\cup\mathrm{B}))$.

(iii): Using (i), A $\succcurlyeq^{\mathcal{W}}_{\forall\exists} \{\beta\}$ holds if and only if $\gamma(\mathcal{W},\mathrm{A}) \subseteq \gamma(\mathcal{W},\{\beta\})$. Recall that $\Theta(\mathcal{W},\mathrm{A}) = \{(w,\mathrm{Val}_{\mathrm{A}}(w)) : w \in \mathcal{W}\}$. Now, $\gamma(\mathcal{W},\mathrm{A}) \subseteq \gamma(\mathcal{W},\{\beta\}) \iff \Theta(\mathcal{W},\mathrm{A}) \subseteq \gamma(\mathcal{W},\{\beta\})$ which is if and only if *CH*(*Ext*$(\gamma(\mathcal{W},\mathrm{A}))) \subseteq \gamma(\mathcal{W},\{\beta\})$ because $\Theta(\mathcal{W},\mathrm{A}) \subseteq$ *CH*(*Ext*$(\gamma(\mathcal{W},\mathrm{A}))) \subseteq \gamma(\mathcal{W},\mathrm{A})$, by Lemma 3.6.4. Because, by Lemma 3.6.2(iv), $\gamma(\mathcal{W},\{\beta\})$ is convex, *CH*(*Ext*$(\gamma(\mathcal{W},\mathrm{A}))) \subseteq \gamma(\mathcal{W},\{\beta\})$ holds if and only if *Ext*$(\gamma(\mathcal{W},\mathrm{A})) \subseteq \gamma(\mathcal{W},\{\beta\})$, which, using Lemma 3.6.2(i), is if and only if for all $(w,r) \in$ *Ext*$(\gamma(\mathcal{W},\mathrm{A}))$, $u_w(\beta) \le r$. Then A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B holds if and only if for all $\beta \in$ B, A $\succcurlyeq^{\mathcal{W}}_{\forall\exists} \{\beta\}$, which holds if and only if for all $(w,r) \in$ *Ext*$(\gamma(\mathcal{W},\mathrm{A}))$ and for all $\beta \in$ B we have $u_w(\beta) \le r$.

(iv): For $S \subseteq \gamma(\mathcal{W},\mathrm{A})$, let us define $K(S) = \max\{u_w(\beta) - r : (w,r) \in S\}$. First we show that $K($*CH*$(S)) = K(S)$. To see this, consider $(w_1,r_1),(w_2,r_2) \in S$ and let (w,r) be a convex combination of them, i.e., $(w,r) = s(w_1,r_1) + (1-s)(w_2,r_2)$ for some $s \in (0,1)$. Convexity of $u_w(\beta)$ implies that $u_w(\beta) \le su_{w_1}(\beta) + (1-s)u_{w_2}(\beta)$, so $u_w(\beta) - r = u_w(\beta) - sr_1 - (1-s)r_2 \le s(u_{w_1}(\beta) - r_1) + (1-s)(u_{w_2}(\beta) - r_2)$, so $u_w(\beta) - r \le \max(u_{w_1}(\beta) - r_1, u_{w_2}(\beta) - r_2)$. This shows that adding convex combinations of elements of S does not

change $K(S)$ and thus, $K(CH(S)) = K(S)$. Proving (iv) amounts to proving that $SMR_{\mathcal{W}}(\mathrm{A},\{\beta\}) = K(Ext(\gamma(\mathcal{W},\mathrm{A})))$. Next, note that $K(\gamma(\mathcal{W},\mathrm{A})) = K(\Theta(\mathcal{W},\mathrm{A}))$, where $\Theta(\mathcal{W},\mathrm{A}) = \{(w,\mathrm{Val}_{\mathrm{A}}(w)) : w \in \mathcal{W}\}$. Also, $K(\Theta(\mathcal{W},\mathrm{A})) = \max_{w\in\mathcal{W}} u_w(\beta) - \mathrm{Val}_{\mathrm{A}}(w)$, which equals $SMR_{\mathcal{W}}(\mathrm{A},\{\beta\})$. We have $K(Ext(\gamma(\mathcal{W},\mathrm{A}))) = K(CH(Ext(\gamma(\mathcal{W},\mathrm{A}))))$, and, using Lemma 3.6.4, we have $\Theta(\mathcal{W},\mathrm{A}) \subseteq CH(Ext(\gamma(\mathcal{W},\mathrm{A}))) \subseteq \gamma(\mathcal{W},\mathrm{A})$, which implies $K(Ext(\gamma(\mathcal{W},\mathrm{A}))) = K(\gamma(\mathcal{W},\mathrm{A})) = K(\Theta(\mathcal{W},\mathrm{A})) = SMR_{\mathcal{W}}(\mathrm{A},\{\beta\})$, as required. □

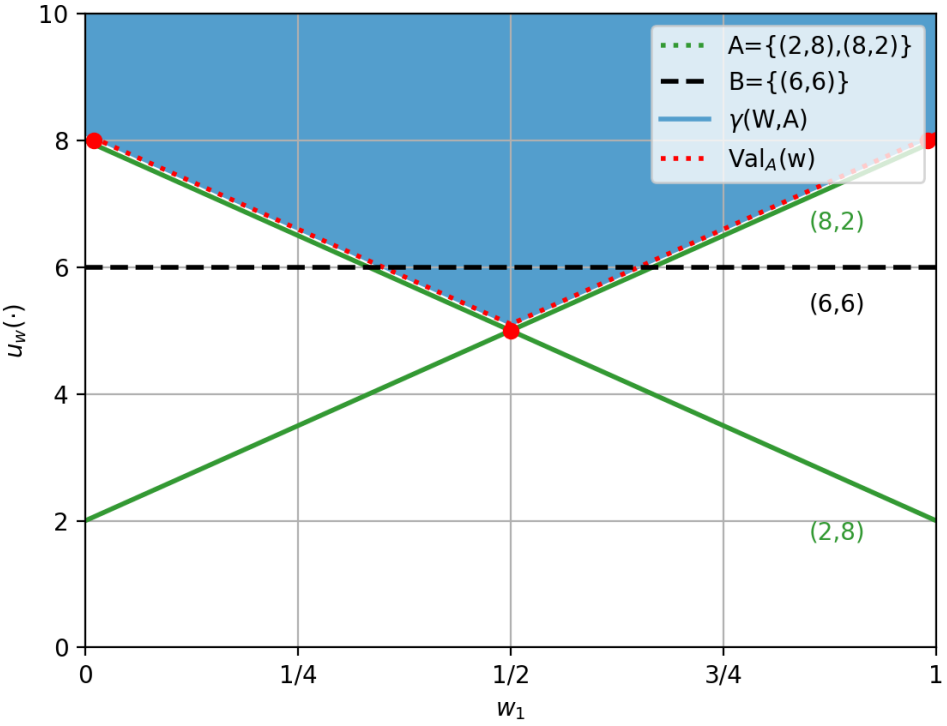


Figure 3.4: Utility function $u_w(\cdot)$ for each alternative in $\mathrm{A} = \{(2,8),(8,2)\}$ (green solid) and $\mathrm{B} = \{(6,6)\}$ (black dashed), where $w \in \mathcal{U} = \{(w_1,w_2) : w_1 + w_2 = 1\}$. The blue area represents the epigraph $\gamma(\mathcal{W},\mathrm{A}) = \{(w,r) : w \in \mathcal{W}, r \geq \mathrm{Val}_{\mathrm{A}}(w)\}$ of A and the red dotted line represents $\mathrm{Val}_{\mathrm{A}}(w)$.

Example 14. *In the example in Figure 3.4, $Ext(\gamma(\mathcal{W},A)) = \{(0,8),(\frac{1}{2},5)(1,8)\}$, where we are again abbreviating w to just its first component w_1, so that e.g., $(\frac{1}{2},5)$ represents the pair $(w,Val_A(w))$ with $w = (\frac{1}{2},\frac{1}{2})$. From Theorem 3.6.6(ii) it follows that $A \not\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} B$ since $Ext(\gamma(\mathcal{W},A\cup B)) = \{(0,8),(\frac{1}{3},6),(\frac{2}{3},6),(1,8)\} \neq Ext(\gamma(\mathcal{W},A))$. This can be verified also with Theorem 3.6.6(iii) since $u_{(\frac{1}{2},\frac{1}{2})}((6,6)) = 6 > 5$. Using Theorem 3.6.6(iv), $SMR_{\mathcal{W}}(A,B) = \max(-2,1,-2) = 1 > 0$; for instance, the middle term in the max equals $u_w((6,6)) - 5 = \frac{1}{2}\cdot 6 + \frac{1}{2}\cdot 6 - 5 = 1$. Therefore, $A \not\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{(6,6)\}$, by Proposition 3.4.1. This illustrates the fact that it is*

not sufficient to just consider the extreme points of $\mathcal{W}$ that in this case are $(0, 1)$ *and* $(1, 0)$.

From Lemma 3.4.1 and Theorem 3.6.6 follows that $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$ can be computed as $\max_{\beta \in \mathrm{B}} \max \{u_w(\beta) - r \,:\, (w, r) \in Ext(\gamma(\mathcal{W}, \mathrm{A}))\}$

3.7 The Case of Multi-Attribute Utility Vectors

We now consider the situation in which we are especially interested, where the alternatives in Ω are multi-attribute utility vectors evaluated with a parameterised weighted sum utility function $u_w(\alpha) = \alpha \cdot w$ defined in 2.1.4, with $\Omega = I\!R^p$, $\alpha \in \Omega$ and $w \in \mathcal{U} = \{w \in I\!R^p : w_i \geq 0, \sum_{i=1}^{p} = 1\}$. The utility function of a set of alternatives A is then $\mathrm{Val}_{\mathrm{A}}(w) = \max_{\alpha \in \mathrm{A}} w \cdot \alpha$.

We will assume $\mathcal{W}$ to be a closed polytope in $I\!R^p$, which can be defined using a finite set of linear inequalities. Given a finite set $\Lambda = \{\lambda_i \,:\, i = 1, \ldots, k\}$ of vectors in $I\!R^p$, and corresponding real numbers r_i, we can define $\mathcal{W}$ to be the set of $w \in \mathcal{U}$ such that for all $i = 1, \ldots, k$, $w \cdot \lambda_i \geq r_i$. In particular, such linear inequalities can arise from input preferences of the form α *is preferred to* β, leading to the constraint $w \cdot (\alpha - \beta) \geq 0$.

With these assumptions, the epigraph $\gamma(\mathcal{W}, \mathrm{A}) = \{(w, r) \,:\, w \in \mathcal{W}, r \geq \mathrm{Val}_{\mathrm{A}}(w)\}$ $= \{(w, r) \in I\!R^{p+1} \,:\, w \in \mathcal{W}, r \geq w \cdot \alpha, \forall \alpha \in \mathrm{A}\}$ of the value function $\mathrm{Val}_{\mathrm{A}}(w)$ is a polytope in $I\!R^{p+1}$ since it is defined by the intersection of $|\mathrm{A}|$ $(p + 1)$-dimensional half-spaces $r \geq w \cdot \alpha$ with $r \in I\!R$ and $w \in I\!R^p$. Some of our algorithms require the enumeration of the extreme points $Ext(\gamma(\mathcal{W}, \mathrm{A}))$ of the epigraph of $\mathrm{Val}_{\mathrm{A}}(w)$, and the computational complexity of this operation is $O(p^{|\mathrm{A}|})$ (see, e.g.,[Dye83]). However, as we will see in our experimental results, for small values of p these algorithms are faster than algorithms based on the solution of $|\mathrm{A}|$ linear programming problems which have a polynomial computational complexity with respect to a number of constraints $|\mathrm{A}|$ and a number of variables p. For example, the algorithm proposed in [Vai89] to solve linear programming problems has a computational complexity $O((p+|\mathrm{A}|)^{1.5}|\mathrm{A}|)$, but recently faster algorithms have been published (see, e.g., [vdB20]). To compute the extreme points $\mathcal{W}^0$ of $\mathcal{W}$ we use the same procedure used to enumerate of the extreme points of the epigraph, which in this case has a computational complexity $O(p^{|\Lambda|})$. This is in general a faster operation since usually the number of input preferences Λ is much lower than the number of input alternatives A, and $\mathcal{W}$ is one dimension lower than $\gamma(\mathcal{W}, \mathrm{A})$.

This form of preferences has been studied a great deal; for instance, $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ consists of the non-dominated alternatives in A for a multiobjective program (MOP) given a cone (with the cone generated as the dual of $\mathcal{W}$) [Yu74, Wie07, EW05]. Without any additional preferences, so that $\mathcal{W}$ is just the unit $(p-1)$-simplex, $\succcurlyeq_{\mathcal{W}}$ is the Pareto ordering on alternatives, and $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ is set of Pareto-optimal alternatives, with the supported alternatives being also in $\mathrm{PO}_{\mathcal{W}}(\mathrm{A})$.

Testing A $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ B and A $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ B

Recall that for $E \subseteq I\!R^p$, $CH(E)$ is the convex hull of E. The follow simple result is useful for computing relations $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ and $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$.

Lemma 3.7.1. *Assume that for $w \in I\!R^p, \alpha \in I\!R^p$, $u_w(\alpha) = w \cdot \alpha$. Let $\mathcal{W}, \mathcal{W}' \subseteq I\!R^p$.*

(i) If $CH(\mathcal{W}) = CH(\mathcal{W}')$ then $\succcurlyeq_{\mathcal{W}} = \succcurlyeq_{\mathcal{W}'}$. In particular, if $\mathcal{W}$ is a compact subset of $I\!R^p$ and $\mathcal{W}^0 = Ext(\mathcal{W})$ is the set of extreme points of $\mathcal{W}$ then $\succcurlyeq_{\mathcal{W}} = \succcurlyeq_{\mathcal{W}^0}$.

(ii) Binary relations $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ and $\succcurlyeq^{\mathcal{W}^0}_{\forall\exists\forall}$ are equal; and $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ equals $\succcurlyeq^{\mathcal{W}^0}_{\exists\forall\forall}$, and A $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ B if and only if there exists $\alpha \in$ A such that for all $w \in \mathcal{W}^0$, $u_w(\alpha) \geq Val_B(w)$.

Proof: (i): Let $\mathcal{W}'' = CH(\mathcal{W}) = CH(\mathcal{W}')$. We will show that $\succcurlyeq_{\mathcal{W}}$ equals $\succcurlyeq_{\mathcal{W}''}$; the same proof will show $\succcurlyeq_{\mathcal{W}'}$ equals $\succcurlyeq_{\mathcal{W}''}$, and thus, $\succcurlyeq_{\mathcal{W}} = \succcurlyeq_{\mathcal{W}'}$. Since $\mathcal{W}'' \supseteq \mathcal{W}$, we have $\succcurlyeq_{\mathcal{W}} \subseteq \succcurlyeq_{\mathcal{W}''}$; to prove the converse, suppose that $\alpha \succcurlyeq_{\mathcal{W}} \beta$ and consider any $w \in \mathcal{W}''$. Then, there exists $w_i \in \mathcal{W}$ and strictly positive reals r_i, for $i = 1 \ldots, k$, such that $w = \sum_{i=1}^{k} r_i w_i$. Because $\alpha \succcurlyeq_{\mathcal{W}} \beta$, for each $i = 1 \ldots, k$, $w_i \cdot (\alpha - \beta) \geq 0$, and thus $w \cdot (\alpha - \beta) \geq 0$, showing that $w \cdot \alpha \geq w \cdot \beta$. This shows that $\alpha \succcurlyeq_{\mathcal{W}''} \beta$, and thus $\succcurlyeq_{\mathcal{W}}$ equals $\succcurlyeq_{\mathcal{W}''}$, and hence, $\succcurlyeq_{\mathcal{W}} = \succcurlyeq_{\mathcal{W}'}$. Since $\mathcal{W}$ is compact and $\mathcal{W}^0$ is the set of extreme points of $\mathcal{W}$ we have $CH(\mathcal{W}^0) = \mathcal{W}$ and so $\succcurlyeq_{\mathcal{W}} = \succcurlyeq_{\mathcal{W}^0}$.

(ii): A $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ B if and only if for all $\beta \in$ B there exists $\alpha \in$ A such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$. And A $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ B if and only if there exists $\alpha \in$ A such that for all $\beta \in$ B, $\alpha \succcurlyeq_{\mathcal{W}} \beta$. The first part then implies $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ and $\succcurlyeq^{\mathcal{W}^0}_{\forall\exists\forall}$ are equal; and $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ equals $\succcurlyeq^{\mathcal{W}^0}_{\exists\forall\forall}$. And we also have that A $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ B if and only if there exists $\alpha \in$ A such that for all $w \in \mathcal{W}^0$, $u_w(\alpha) \geq \mathrm{Val}_{\mathrm{B}}(w)$. □

Because of Lemma 3.7.1, there is a simple way of testing if $\alpha \succcurlyeq_{\mathcal{W}} \beta$ (for $\alpha, \beta \in I\!R^p$): $\alpha \succcurlyeq_{\mathcal{W}} \beta$ holds if and only if for each extreme point w of $\mathcal{W}$, we have $w \cdot (\alpha - \beta) \geq 0$.

This can then be used for the relations $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ and $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$, using, for example, A $\succcurlyeq^{\mathcal{W}}_{\forall\exists\forall}$ B if and only if for all $\beta \in$ B there exists $\alpha \in$ A such that $\alpha \succcurlyeq_{\mathcal{W}} \beta$.

In Section 3.6 we gave an EEU method for computing $SMR_{\mathcal{W}}$ and testing dominance; in Section 3.7.1 we give a straight-forward LP method related to the approaches used in [VB09, BP15b, BP16]. In Section 3.7.2 we give a result that enables one to compute the minimal equivalent subset using the extreme points of the epigraph.

3.7.1 Linear programming for $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$, and A $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ B

The definitions easily imply that $SMR_{\mathcal{W}}(\mathrm{A}, \{\beta\})$ equals $\max_{w\in\mathcal{W}} u_w(\beta) - \mathrm{Val}_{\mathrm{A}}(w)$. Thus, for real-valued x, we have $SMR_{\mathcal{W}}(\mathrm{A}, \{\beta\}) \geq x$ if and only if there exists $w \in \mathcal{W}$ such that for all $\alpha \in$ A, $w \cdot (\beta - \alpha) \geq x$. This leads to the following characterisation.

$SMR_{\mathcal{W}}(\mathrm{A}, \{\beta\})$ is equal to the maximum value of x such that there exists $w \in \mathbb{R}^p$ satisfying the constraints:

(i) $w \in \mathcal{W}$.

(ii) For all $\alpha \in$ A, $w \cdot (\beta - \alpha) \geq x$.

Since $\mathcal{W}$ is a closed polytope, we can use a linear programming solver to compute $SMR_{\mathcal{W}}(\mathrm{A}, \{\beta\})$. Applying this for each $\beta \in$ A allows us to compute $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$, and thus, to test if A $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$ B, using Lemma 3.4.1.

3.7.2 Using extreme points of epigraph to compute minimal equivalent subset

We first prove some key properties relating to the the sets $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ and their relationship with the extreme points of the epigraph of the utility function.

Lemma 3.7.2. *Assume that $\mathcal{W}$ is a convex subset of $\mathbb{R}^p$, and that for $w \in \mathbb{R}^p, \alpha \in \mathbb{R}^p$, $u_w(\alpha) = w \cdot \alpha$. Consider A $\in \mathcal{M}$, $\alpha \in$ A, $w \in \mathcal{W}$. Let $I_\alpha = \{(w, r) \in \mathbb{R}^p \times \mathbb{R} : r = w \cdot \alpha\}$. For $K \subseteq \mathbb{R}^p \times \mathbb{R}$ we write $K^{\downarrow}$ for the projection of K to $\mathbb{R}^p$, i.e., $K^{\downarrow} = \{w \in \mathbb{R}^p : (w, r) \in K\}$.*

(i) For any $\alpha \in$ A, $\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha)$ is a convex subset of $\mathcal{W}$.

(ii) If $\mathcal{W}$ is compact and $\alpha, \beta \in$ A, then $\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{A}_{\mathcal{W}}(\beta) \iff$ Ext$(\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha)) \subseteq$ Ext$(\mathrm{Opt}^{A}_{\mathcal{W}}(\beta))$.

(iii) $\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha) = (\gamma(\mathcal{W},A) \cap I_\alpha)^{\downarrow}$.

(iv) $\mathit{Ext}((\gamma(\mathcal{W},A) \cap I_\alpha)^{\downarrow}) = (\mathit{Ext}(\gamma(\mathcal{W},A) \cap I_\alpha))^{\downarrow}$.

(v) $\mathit{Ext}(\gamma(\mathcal{W},A) \cap I_\alpha) = \mathit{Ext}(\gamma(\mathcal{W},A)) \cap I_\alpha$.

Proof: (i): Lemma 3.3.12(i) implies that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathcal{W} \cap \bigcap_{\beta \in \mathrm{A}} H_{\alpha \geq \beta}$, where $H_{\alpha \geq \beta} = \{w \in I\!R^p : w \cdot \alpha \geq w \cdot \beta\}$. Since each set $H_{\alpha \geq \beta}$ is convex and $\mathcal{W}$ is convex then their intersection is convex.

(ii): Assume $\mathcal{W}$ is compact. Then $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \mathit{CH}(\mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha))$, and $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta) = \mathit{CH}(\mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta))$. Thus, if $\mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) \subseteq \mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta))$ then $\mathit{CH}(\mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) \subseteq \mathit{CH}(\mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta))$, and hence, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$.

For the converse, assume that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$, and, proceeding with proof by contradiction, let us assume that there exists some $w \in \mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) \setminus \mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta))$. We have $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ and thus $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$, so there exists some $w_1, w_2 \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$ and $s \in (0,1)$ such that $w = sw_1 + (1-s)w_2$. Because $w_1, w_2 \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$ for $i = 1, 2$, $w_i \cdot \beta \geq w_i \cdot \alpha$, i.e., $w_i \cdot (\beta - \alpha) \geq 0$. Also, since $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$, $w \cdot (\beta - \alpha) = 0$. Thus, $0 = w \cdot (\beta - \alpha) = sw_1 \cdot (\beta - \alpha) + (1-s)w_2 \cdot (\beta - \alpha)$, and so, since both terms are non-negative, they are both zero: $w_1 \cdot (\beta - \alpha) = w_2 \cdot (\beta - \alpha) = 0$, which implies that $w_1, w_2 \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$. This contradicts w being an extreme point of $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$.

(iii): $(w, r) \in (\gamma(\mathcal{W}, \mathrm{A}) \cap I_\alpha)$ if and only if $r \geq \mathrm{Val}_{\mathrm{A}}(w)$ and $w \cdot \alpha = r$, which is if and only if $r = w \cdot \alpha = \mathrm{Val}_{\mathrm{A}}(w)$, which is if only if $w \in \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ and $r = w \cdot \alpha$. Thus, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = (\gamma(\mathcal{W}, \mathrm{A}) \cap I_\alpha)^{\downarrow}$.

(iv): Let us write $K = \gamma(\mathcal{W}, \mathrm{A}) \cap I_\alpha$. First we prove, by contradiction, that $\mathit{Ext}(K^{\downarrow}) \subseteq (\mathit{Ext}(K))^{\downarrow}$. So, suppose that there exists some $w \in \mathit{Ext}(K^{\downarrow}) \setminus (\mathit{Ext}(K))^{\downarrow}$. Now, $w \in K^{\downarrow}$, so $(w, w \cdot \alpha) \in K$. However, $w \notin (\mathit{Ext}(K))^{\downarrow}$ implies that $(w, w \cdot \alpha) \notin \mathit{Ext}(K)$. Thus, there exists some $(w_1, r_1), (w_2, r_2) \in K$ and $s \in (0,1)$ such that $(w, w \cdot \alpha) = s(w_1, r_1) + (1-s)(w_2, r_2)$. We must have $r_1 = w_1 \cdot \alpha$ and $r_2 = w_2 \cdot \alpha$. $w_1, w_2 \in (K)^{\downarrow}$ and $w = sw_1 + (1-s)w_2$, which contradicts $w \in \mathit{Ext}(K^{\downarrow})$.

Conversely, we prove, by contradiction, that $\mathit{Ext}(K^{\downarrow}) \supseteq (\mathit{Ext}(K))^{\downarrow}$. Suppose there exists some $w \in (\mathit{Ext}(K))^{\downarrow} \setminus \mathit{Ext}(K^{\downarrow})$. Now, because $w \in (\mathit{Ext}(K))^{\downarrow}$, there exists r such that $(w, r) \in \mathit{Ext}(K)$. Then $(w, r) \in K$ implies $r = w \cdot \alpha$, and thus, $(w, w \cdot \alpha) \in \mathit{Ext}(K)$. Since $w \in K^{\downarrow}$ and $w \notin \mathit{Ext}(K^{\downarrow})$, there exists $w_1, w_2 \in K^{\downarrow}$ $s \in (0,1)$ such that $w = sw_1 + (1-s)w_2$. Since $w_1, w_2 \in K^{\downarrow}$, there exists r_1 and r_2 such that $(w_1, r_1), (w_2, r_2) \in K$. Then, $r_1 = w_1 \cdot \alpha$ and $r_2 = w_2 \cdot \alpha$, and so

$(w, w \cdot \alpha) = s(w_1, r_1) + (1-s)(w_2, r_2)$, which contradicts $(w, w \cdot \alpha) \in \mathit{Ext}(K)$.

(v): For any convex subsets P and Q of some set we have $\mathit{Ext}(P) \cap Q \subseteq \mathit{Ext}(P \cap Q)$. To show this, we proceed with proof by contradiction: suppose that $x \in \mathit{Ext}(P) \cap Q$ and $x \notin \mathit{Ext}(P \cap Q)$. Since $x \in P \cap Q$ but is not in $\mathit{Ext}(P \cap Q)$, there exists a line segment in $P \cap Q$ that contains x as an internal point. But this line segment is also in P, contradicting $x \in \mathit{Ext}(P)$.

The above result shows that $\mathit{Ext}(\gamma(\mathcal{W}, \mathrm{A}) \cap I_\alpha) \supseteq \mathit{Ext}(\gamma(\mathcal{W}, \mathrm{A})) \cap I_\alpha$. To prove the converse, we proceed by contradiction and assume that there exists some element $(w, r) \in \mathit{Ext}(\gamma(\mathcal{W}, \mathrm{A}) \cap I_\alpha)$ with $(w, r) \notin \mathit{Ext}(\gamma(\mathcal{W}, \mathrm{A})) \cap I_\alpha$. Since $(w, r) \in \gamma(\mathcal{W}, \mathrm{A}) \setminus \mathit{Ext}(\gamma(\mathcal{W}, \mathrm{A}))$ there exist some $(w_1, r_1), (w_2, r_2) \in \gamma(\mathcal{W}, \mathrm{A})$ and $s \in (0, 1)$ such that $(w, r) = s(w_1, r_1) + (1-s)(w_2, r_2)$. Now, $(w, r) \in I_\alpha$ implies that $r = w \cdot \alpha$; and $(w_1, r_1), (w_2, r_2) \in \gamma(\mathcal{W}, \mathrm{A})$ implies that $r_1 \geq \mathrm{Val}_{\mathrm{A}}(w_1) \geq w_1 \cdot \alpha$ and $r_2 \geq \mathrm{Val}_{\mathrm{A}}(w_2) \geq w_2 \cdot \alpha$. We have $r = sr_1 + (1-s)r_2$ and so $0 = r - (w \cdot \alpha) = s(r_1 - w_1 \cdot \alpha) + (1-s)(r_2 - w_2 \cdot \alpha)$. Since the two parts of the right-hand-side are non-negative, they must be zero, showing that $r_1 = w_1 \cdot \alpha$ and $r_2 = w_2 \cdot \alpha$. This implies that $(w_1, r_1), (w_2, r_2) \in \gamma(\mathcal{W}, \mathrm{A}) \cap I_\alpha$, which contradicts $(w, r) \in \mathit{Ext}(\gamma(\mathcal{W}, \mathrm{A}) \cap I_\alpha)$. □

For the linear case, with $\alpha \in \mathrm{A}$, the set $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$, consisting of all w in $\mathcal{W}$ that make α optimal in A (see Section 3.3.1), is convex; we abbreviate its set of extreme points $\mathit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha))$ to $E^{\mathrm{A}}_{\mathcal{W}}(\alpha)$. Theorem 3.3.15 implies that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is the unique minimal equivalent subset of an (equivalence-free) set $\mathrm{A} \in \mathcal{M}$, which can be shown to consist of all $\alpha \in \mathrm{A}$ such that there does not exist $\beta \in \mathrm{A}$ such that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta) \supsetneqq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$. The following result shows that the condition $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta) \supsetneqq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is (perhaps surprisingly) equivalent to $E^{\mathrm{A}}_{\mathcal{W}}(\beta) \supsetneqq E^{\mathrm{A}}_{\mathcal{W}}(\alpha)$, and that $E^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ can be computed by projecting the extreme points of the epigraph, $\mathit{Ext}(\gamma(\mathcal{W}, \mathrm{A}))$. This is the basis of our method, described in Section 3.8.1(II) below, for efficiently computing the minimal equivalent set $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$.

(In more detail: Corollary 3.3.16.1 implies that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$, and, by definition, $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$ consists of all $\alpha \in \mathrm{A}$ such that there does not exist $\beta \in \mathrm{A}$ such that $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta) \supsetneqq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$.)

Proposition 5. *Assume that $\mathcal{W}$ is a convex subset of $\mathbb{R}^p$, and that for $w \in \mathbb{R}^p, \alpha \in \mathbb{R}^p$, $u_w(\alpha) = w \cdot \alpha$. Consider $A \in \mathcal{M}$, $w \in \mathcal{W}$, and $\alpha, \beta \in A$.*

(i) $\mathrm{Opt}^{A}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{A}_{\mathcal{W}}(\beta) \iff E^{A}_{\mathcal{W}}(\alpha) \subseteq E^{A}_{\mathcal{W}}(\beta)$.

(ii) $E^A_{\mathcal{W}}(\alpha) = \{w \in I\!R^p : (w, w \cdot \alpha) \in \textit{Ext}(\gamma(\mathcal{W}, A))\}$.

(iii) If $\mathcal{W}$ *is compact then* $\textit{dim}(\mathrm{Opt}^A_{\mathcal{W}}(\alpha)) < |E^A_{\mathcal{W}}(\alpha)|$.

Proof: (i): It is part (ii) of Lemma 3.7.2.

(ii): Using parts (iii), (iv) and (v) of Lemma 3.7.2, we have $\textit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) = (\textit{Ext}(\gamma(\mathcal{W}, \mathrm{A})) \cap I_\alpha)^\downarrow = \{w \in I\!R^p : (w, w \cdot \alpha) \in \textit{Ext}(\gamma(\mathcal{W}, \mathrm{A}))\}$.

(iii): Since $\mathcal{W}$ is compact, then $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ is compact since it is a closed subset of compact set $\mathcal{W}$, and so, $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) = \textit{CH}(E^{\mathrm{A}}_{\mathcal{W}}(\alpha))$. This implies that $\textit{dim}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) < |E^{\mathrm{A}}_{\mathcal{W}}(\alpha)|$. □

Using Theorem 3.3.15 and Proposition 5 we can compute $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$. In fact, for each $\alpha \in \mathrm{A}$ we can compute $\textit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha))$ using Proposition 5(ii). Because of Theorem 3.3.15, we know that any elements α such that $|\textit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha))| - 1 < \textit{dim}(\mathcal{W})$ are not in $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ (since $\textit{dim}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) \leq |\textit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha))| - 1$). If for any $\alpha, \beta \in \mathrm{A}$ with $\alpha \not\equiv_{\mathcal{W}} \beta$, $\textit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) \subseteq \textit{Ext}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta))$, (so then $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$) then we know that $\alpha \notin \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, and if those sets are equal we know also that $\beta \notin \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. Now, any of these elements can then be deleted from A, because of a nice incremental property of $\mathrm{MPO}_{\mathcal{W}}$ (see an earlier comment). This then gives a nice incremental approach to computing $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$.

Example 15. *Let* $A' = \{(2,8),(8,2)\}$, $A'' = \{(5,5),(3,3)\}$ *and* $A = A' \cup A''$. *In the example in Figure 3.5 we can see that* $\mathrm{Opt}^A_{\mathcal{W}}((3,3)) = \emptyset$, $\mathrm{Opt}^A_{\mathcal{W}}((5,5)) = \{(\frac{1}{2},\frac{1}{2})\}$, $\mathrm{Opt}^A_{\mathcal{W}}((8,2)) = \{w \in \mathcal{W} : w_1 \in [0,\frac{1}{2}], w_2 = 1 - w_1\}$ *and* $\mathrm{Opt}^A_{\mathcal{W}}((2,8)) = \{w \in \mathcal{W} : w_1 \in [\frac{1}{2},1], w_2 = 1 - w_1\}$. *Thus,* $E^A_{\mathcal{W}}((3,3)) = \emptyset$, $E^A_{\mathcal{W}}((5,5)) = \{(\frac{1}{2},\frac{1}{2})\}$, $E^A_{\mathcal{W}}((8,2)) = \{(\frac{1}{2},\frac{1}{2}),(1,0)\}$ *and* $E^A_{\mathcal{W}}((2,8)) = \{(\frac{1}{2},\frac{1}{2}),(0,1)\}$. *Therefore,* $(3,3)$ *and* $(5,5)$ *are not possibly strictly optimal alternatives of A with respect to* $\mathcal{W}$ *since* $|E^A_{\mathcal{W}}((3,3))| - 1 = -1 < 1 = \textit{dim}(\mathcal{W})$ *and* $|E^A_{\mathcal{W}}((5,5))| - 1 = 0 < 1 = \textit{dim}(\mathcal{W})$. *On the other hand,* $(8,2)$ *and* $(2,8)$ *are possibly strictly optimal alternatives of A with respect to* $\mathcal{W}$ *since* $|E^A_{\mathcal{W}}((8,2))| - 1 = |E^A_{\mathcal{W}}((2,8))| - 1 = 1 \geq 1 = \textit{dim}(\mathcal{W})$.

3.8 The Structure of the Algorithms

In this section we make use of mathematical results in previous sections in developing computational methods for computing the minimal equivalent set $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ and testing dominance between sets, for the case of multi-attribute

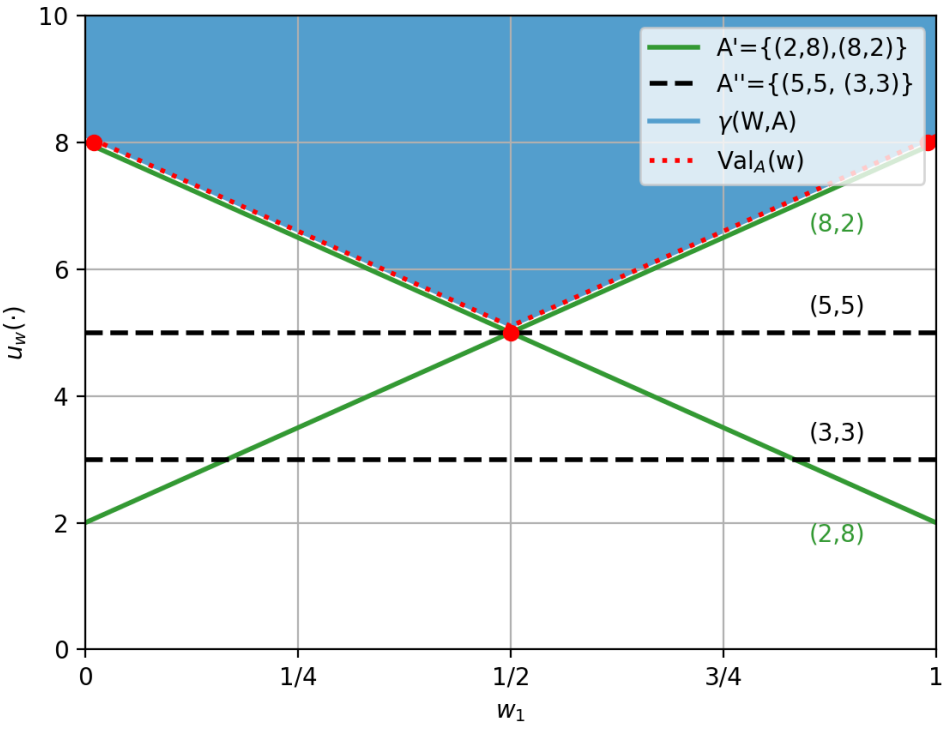


Figure 3.5: Utility function $u_w(\cdot)$ for each alternative in $\mathrm{A}' = \{(2,8),(8,2)\}$ (green solid) and $\mathrm{A}' = \{(5,5),(3,3)\}$ (black dashed), where $w \in \mathcal{U} = \{(w_1,w_2) : w_1 + w_2 = 1\}$. The blue area is the epigraph $\gamma(\mathcal{W},\mathrm{A}) = \{(w,r) : w \in \mathcal{W}, r \geq \mathrm{Val}_{\mathrm{A}}(w)\}$, where $\mathrm{A} = \mathrm{A}' \cup \mathrm{A}''$, and the red dotted line represents $\mathrm{Val}_{\mathrm{A}}(w)$.

utility vectors, with the set of scenarios $\mathcal{W}$ being a convex polytope, and with linear utility functions.

3.8.1 Computing minimal equivalent set

Given $\mathrm{A} \in \mathcal{M}$, we aim to generate $\mathrm{A}' \subseteq \mathrm{A}$ with $\mathrm{A}' \equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$, and such that for strict subset A'' of A', $\mathrm{A}'' \not\equiv^{\mathcal{W}}_{\forall\forall\exists} \mathrm{A}$.

First we pre-process by eliminating elements of A not in $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$. At the same time we can make A equivalence-free (Algorithm 5). This operation requires the computation of $O(|\mathrm{A}|^2)$ dot products.

Theorem 3.3.15 implies that there exists a minimal equivalent set, i.e., $\mathrm{SME}_{\mathcal{W}}(\mathrm{A})$ has a unique element, say, A', and this equals $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. We have two methods for then computing A'.

(I) we use the approach $\mathit{Filter}_{\sigma}(\mathrm{A}; \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists})$ defined in Section 3.3.2. This

Algorithm 5 UD

```
1: procedure UD(A, W^0)
2:     P ← ∅
3:     for α_i ∈ A do
4:         P ← P ∪ {α_i}
5:         for α_j ∈ A \ P do
6:             if α_i ≽_{W^0} α_j and α_j ⋡_{W^0} α_i then
7:                 A ← A \ {α_j}
8:             if α_j ≽_{W^0} α_i and α_i ⋡_{W^0} α_j then
9:                 A ← A \ {α_i}
10:                break
11:            if α_i ≡_{W^0} α_j then
12:                A ← A \ {α_j}
13:    return return A
```

involves multiple (i.e., |A|) tests of the form $\mathrm{A} \setminus \{\alpha\} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \{\alpha\}$, which can be achieved using a similar approach to 3.8.2 below, using a linear programming solver (Algorithm 6). This filtering operation requires then solving $O(|\mathrm{A}|)$ linear programming problems with $O(|\mathrm{A}|)$ constraints and p variables.

Algorithm 6 PSO based on linear programming

```
1: procedure PSO_LP(A, W)
2:     for α_1 in A do
3:         x_M ← Maximize x subject to
4:                 { (α_1 − α_2) · w ≥ x      ∀α_2 ∈ A \ {α_1} }
                   { w ∈ W                                     }
5:         if x_M ≤ 0 then
6:             A ← A \ α_1
7:     return A
```

(II) For each $\alpha \in \mathrm{A}$ we compute $E^{\mathrm{A}}_{\mathcal{W}}(\alpha)$ using Proposition 5(ii), by computing the extreme points of the epigraph. We can eliminate any element α such that $|E^{\mathrm{A}}_{\mathcal{W}}(\alpha)| \leq \textit{dim}(\mathcal{W})$, since Proposition 5(iii) would then imply that $\textit{dim}(\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha)) < \textit{dim}(\mathcal{W})$, and thus, $\alpha \notin \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, by Theorem 3.3.15. If for any $\alpha, \beta \in \mathrm{A}$ with $\alpha \not\equiv_{\mathcal{W}} \beta$, $E^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq E^{\mathrm{A}}_{\mathcal{W}}(\beta)$, (so then $\mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\alpha) \subseteq \mathrm{Opt}^{\mathrm{A}}_{\mathcal{W}}(\beta)$) then we know that $\alpha \notin \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$, and if those sets are equal we know also that $\beta \notin \mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$. Now, any of these elements can then be deleted from A, because of an incrementality property of $\mathrm{PSO}_{\mathcal{W}}$. We then continue until for all remaining elements $\alpha, \beta \in \mathrm{A}$ we have

$E^{\mathrm{A}}_{\mathcal{W}}(\alpha) \not\subseteq E^{\mathrm{A}}_{\mathcal{W}}(\beta)$, and then $\mathrm{A} = \mathrm{A}'$, the unique element of $\mathrm{SME}_{\mathcal{W}}(\mathrm{A})$, the set of possibly strictly optimal elements (Algorithm 7). This procedure requires enumerating the extreme points (w, r) of the epigraph of A which is an operation with exponential computational complexity with respect to $|\mathrm{A}|$.

Algorithm 7 PSO based on epigraph

```
1: procedure PSO_EP(A, W)
2:     for α in A do
3:         if |E^A_W(α)| ≤ dim(W) then
4:             A ← A \ α
5:     for α_1 in A do
6:         for α_2 in A do
7:             if E^A_W(α_2) ⊆ E^A_W(α_1) then
8:                 A ← A \ α_2
9:             if E^A_W(α_2) = E^A_W(α_1) then
10:                A ← A \ α_1
11:                break
12:    return A
```

In more detail: Corollary 3.3.16.2 implies the following property of $\mathrm{MPO}_{\mathcal{W}}$: if $\mathrm{C} \cap \mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) = \emptyset$ then $\mathrm{MPO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{MPO}_{\mathcal{W}}(\mathrm{A} \setminus \mathrm{C})$. Since $\mathrm{MPO}_{\mathcal{W}} = \mathrm{PSO}_{\mathcal{W}}$ by Corollary 3.3.16.2, if $\beta \notin \mathrm{PSO}_{\mathcal{W}}(\mathrm{A}) = \mathrm{MPO}_{\mathcal{W}}(\mathrm{A})$, we can delete β from A.

3.8.2 Testing $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$

Our algorithm includes three steps of increasing complexity:

(1) Efficiently testing (a) a necessary condition $\mathrm{A} \succcurlyeq^{\mathcal{W}^0}_{\forall\forall\exists} \mathrm{B}$, where $\mathcal{W}^0 = \mathit{Ext}(\mathcal{W})$ is the set of extreme points of $\mathcal{W}$; and (b) a sufficient condition, whether there exists $\alpha \in \mathrm{A}$ such that for all $w \in \mathcal{W}^0$, $u_w(\alpha) \geq \mathrm{Val}_{\mathrm{B}}(w)$; (the conditions can be tested together, by first computing $\mathrm{Val}_{\mathrm{B}}(w)$ for each $w \in \mathcal{W}^0$) (Algorithm 8). If (a) is false then we know that $\mathrm{A} \not\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ (because of monotonicity with respect to $\mathcal{W}$); if (b) is true then we know that $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ holds. If the necessary condition is false, or the sufficient condition is true, then we need go no further. The complexity of this step is $O(|\mathcal{W}^0|(|\mathrm{A}| + |\mathrm{B}|))$ with the most complex operation being the dot product.

(2) Pre-processing by reducing the sets A and B. We replace A by $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ and B by $\mathrm{UD}_{\mathcal{W}}(\mathrm{B})$ (Algorithm 5). We then eliminate all elements β from B

Algorithm 8 Necessary and sufficient condition for $\mathrm{A} \succcurlyeq^{\mathcal{W}^0}_{\forall\forall\exists} \mathrm{B}$

```
1:  procedure NSc(A, B, W^0)
2:      Val_A : W^0 → IR, Val_B : W^0 → IR
3:      Necessary ← true, Sufficient ← false
4:      A' ← A
5:      for w ∈ W^0 do
6:          Val_A(w) ← −∞, Val_B(w) ← −∞
7:          for β ∈ B do
8:              Val_B(w) ← max(Val_B(w), β · w)
9:          for α ∈ A do
10:             if α · w < Val_B(w) then
11:                 A' ← A' \ {α}
12:             Val_A(w) ← max(Val_A(w), α · w)
13:         if Val_A(w) < Val_B(w) then
14:             Necessary ← false
15:             break
16:     if Necessary then
17:         Sufficient ← |A'| > 0
18:     return Necessary, Sufficient
```

such that for some $\alpha \in \mathrm{A}$, $\alpha \succcurlyeq_{\mathcal{W}} \beta$ (Algorithm 9). If B becomes empty then we can stop, since we then have $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$. This step has computational complexity $O(|\mathcal{W}^0||\mathrm{A}||\mathrm{B}|)$ with the most complex operation being the dot product.

Algorithm 9 Filtering using relation $\succcurlyeq^{\mathcal{W}^0}_{\forall\exists\forall}$

```
1:  procedure Filt_∀∃∀(A, B, W^0)
2:      for α ∈ A do
3:          for β ∈ B do
4:              if α ≽_{W^0} β then
5:                  B ← B \ {β}
6:      return B
```

(3) We determine whether $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ holds using one of the methods in Sections 3.7.1 and 3.6, i.e., doing either (a), (b) or (c) below:

   (a) Using linear programming, as described in Section 3.7.1 (Algorithm 10). This operation requires solving $|\mathrm{B}|$ linear programming problems with $O(|\mathrm{A}|)$ constraints and p variables.

   (b) Using Theorem 3.6.6(ii) and testing $\mathit{Ext}(\Gamma(\mathcal{W}, \mathrm{A})) = \mathit{Ext}(\Gamma(\mathcal{W}, \mathrm{A} \cup \mathrm{B}))$ (Algorithm 11). In this case we then need to enumerate the extreme

Algorithm 10 Testing dominance using linear programming

procedure $\mathrm{T}_{\mathrm{LP}}(\mathrm{A}, \mathrm{B}, \mathcal{W})$
  SMR $\leftarrow -\infty$
  for β in B **do**
    $x_M \leftarrow$ Maximize x subject to

$$\left\{ \begin{array}{ll} (\beta - \alpha) \cdot w \geq x & \forall \alpha \in \mathrm{A} \\ w \in \mathcal{W} & \end{array} \right\}$$

    if $x_M >$ *SMR* **then**
      SMR $\leftarrow x_M$
  return *SMR* ≤ 0

points of $\Gamma(\mathcal{W}, \mathrm{A})$ and $\Gamma(\mathcal{W}, \mathrm{A} \cup \mathrm{B})$, which is an operation with computational complexity exponential in $|\mathrm{A} \cup \mathrm{B}|$.

Algorithm 11 Testing dominance using epigraph - method 1

procedure $\mathrm{T}_{\mathrm{EPU}}(\mathrm{A}, \mathrm{B}, \mathcal{W})$
  $\Gamma(\mathcal{W}, \mathrm{A}) \leftarrow \bigcap_{\alpha \in \mathrm{A}} \{(w, r) \,:\, w \in \mathcal{W} \;\&\; r \geq w \cdot \alpha\}$
  Ext$(\Gamma(\mathcal{W}, \mathrm{A})) \leftarrow$ extreme points of $\Gamma(\mathcal{W}, \mathrm{A})$
  $\Gamma(\mathcal{W}, \mathrm{A} \cup \mathrm{B}) \leftarrow \bigcap_{\gamma \in \mathrm{A} \cup \mathrm{B}} \{(w, r) \,:\, w \in \mathcal{W} \;\&\; r \geq w \cdot \gamma\}$
  Ext$(\Gamma(\mathcal{W}, \mathrm{A} \cup \mathrm{B})) \leftarrow$ extreme points of $\Gamma(\mathcal{W}, \mathrm{A} \cup \mathrm{B})$
  return *Ext*$(\Gamma(\mathcal{W}, \mathrm{A})) =$ *Ext*$(\Gamma(\mathcal{W}, \mathrm{A} \cup \mathrm{B}))$

(c) Using Theorem 3.6.6(iii) and testing if $u_w(\beta) \leq r$ for all $\beta \in \mathrm{B}$ and for all $(w, r) \in$ *Ext*$(\gamma(\mathcal{W}, \mathrm{A}))$ (Algorithm 12). With this procedure we need to enumerate the extreme points of $\Gamma(\mathcal{W}, \mathrm{A})$, which is an operation with computational complexity exponential in $|\mathrm{A}|$.

Algorithm 12 Testing dominance using epigraph - method 2

procedure $\mathrm{T}_{\mathrm{EEU}}(\mathrm{A}, \mathrm{B}, \mathcal{W})$
  SMR $\leftarrow -\infty$
  $\Gamma(\mathcal{W}, \mathrm{A}) \leftarrow \bigcap_{\alpha \in \mathrm{A}} \{(w, r) \,:\, w \in \mathcal{W} \;\&\; r \geq w \cdot \alpha\}$
  Ext$(\Gamma(\mathcal{W}, \mathrm{A})) \leftarrow$ extreme points of $\Gamma(\mathcal{W}, \mathrm{A})$
  for $\beta \in \mathrm{B}$ **do**
    for $(w, r) \in$ *Ext*$(\Gamma(\mathcal{W}, \mathrm{A}))$ **do**
      if $\beta \cdot w - r >$ *SMR* **then**
        SMR $\leftarrow \beta \cdot w - r$
  return *SMR* ≤ 0

Although we focus on non-strict dominance, the same algorithms can also be used to test the strongly strict dominance $\mathrm{A} \gg^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ given as for all $w \in \mathcal{W}$, $\mathrm{Val}_{\mathrm{A}}(w) > \mathrm{Val}_{\mathrm{B}}(w)$. In particular, under the conditions of Theorem 3.6.6,

we have $\mathrm{A} \gg^{\mathcal{W}}_{\forall\exists} \mathrm{B} \iff \mathit{SMR}_{\mathcal{W}}(\mathrm{A},\mathrm{B}) < 0$, which is if and only if for all $(w,r) \in \mathit{Ext}(\gamma(\mathcal{W},\mathrm{A}))$ and for all $\beta \in \mathrm{B}$ we have $u_w(\beta) < r$.

3.9 Experimental Testing

All experiments were performed on a computer facilitated by a Core i5 2.70 GHz processor and 8 GB RAM. We used CPLEX 12.8 [ILO17] as the linear programming solver, and we used the Python library pycddlib [Tro18] for computing the extreme points of a polytope.

We consider the linear case, where $\mathcal{W}$ is a subset of the unit $(p-1)$-simplex which is an intersection of T half-spaces. Specifically, we choose T (consistent) random user preferences of the form $aw_i + bw_j \geq cw_k$ (meaning that the user prefers a units of w_i and b units of w_j to c units of w_k), like in [MRW12]. The alternatives in the sets A and B are integer utility vectors. See Appendix A.1 for details about our random problem generator.

The pre-processing steps based on the $\mathrm{UD}_{\mathcal{W}}$ filtering were very worthwhile, for both computing the minimal set in Section 3.8.1, and in 3.8.2(2) for dominance; they reduce the sizes of the sets A and B very considerably (see e.g., Table 3.2), making the algorithms much faster overall, e.g., by an order of magnitude.

For cases in which $\mathit{dim}(\mathcal{W}) < 7$, the EEU method 3.8.1(I) to compute $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ was on average faster, and scaled better with the cardinalities of sets A and B, than the LP method 3.8.1(II). However, the situation dramatically reverses for $\mathit{dim}(\mathcal{W}) \geq 7$; this may well be because the number of extreme points is much larger then. This is illustrated by Table 3.1 along with the performance of the $\mathrm{UD}_{\mathcal{W}}$ filtering, where each figure is an average over 100 random instances. We also tested our algorithms with larger A such as $|\mathrm{A}| = 20{,}000$, with $\mathit{dim}(\mathcal{W}) = 5$ and four user preferences giving an average execution time over 100 experiments of 13 seconds for the $\mathrm{UD}_{\mathcal{W}}$ filtering, 22 seconds for the LP-based method and 6 seconds for the EEU method.

We also tested our EEU approach against the standard LP approach to compute $\mathrm{PO}_{\mathcal{W}}$ and also in this case it looks like that EEU is faster for cases in which $\mathit{dim}(\mathcal{W}) \leq 6$. The performances of EEU to compute $\mathrm{PO}_{\mathcal{W}}$ are very similar to the performances of EEU to compute $\mathrm{PSO}_{\mathcal{W}}$ shown in Table 3.1. With sets generated with our random problem generator we observed that the $\mathrm{PSO}_{\mathcal{W}}$

Table 3.1: Execution times (in seconds) of methods to compute $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ (Section 3.8.1), $\mathrm{UD}_{\mathcal{W}}$ filtering, EEU (I) and LP (II) (and number of extreme points of the epigraph), with respect to $dim(\mathcal{W})$ with $|\mathrm{A}| = 500$ and 4 user preferences.

$dim(\mathcal{W})$	$\mathrm{UD}_{\mathcal{W}}$[s]	LP[s]	EEU[s]	# extreme points
2	0.014	0.057	0.001	13.24
3	0.036	0.192	0.005	57.52
4	0.116	0.548	0.039	248.14
5	0.268	1.467	0.310	1024.74
6	0.439	3.062	2.103	3667.36
7	0.683	5.943	15.630	13483.87

Table 3.2: Number of elements of $\mathrm{A}' = \mathrm{UD}_{\mathcal{W}}(\mathrm{A})$, $\mathrm{B}' = \mathrm{UD}_{\mathcal{W}}(\mathrm{B})$ and $\mathrm{B}'' = \{\beta \in \mathrm{B}' : \forall \alpha \in \mathrm{A},\ \alpha \not\succcurlyeq_{\mathcal{W}} \beta\}$ with respect to $dim(\mathcal{W})$ with $|\mathrm{A}| = |\mathrm{B}| = 500$ and 4 user preferences.

$dim(\mathcal{W})$	$\lvert\mathrm{A}'\rvert$	$\lvert\mathrm{B}'\rvert$	$\lvert\mathrm{B}''\rvert$
2	10.08	7.99	2.44
3	23.89	22.04	5.11
4	48.70	19.53	11.93
5	92.23	86.94	15.45
6	144.78	142.89	41.15
7	206.93	210.11	64.28

filtering removes around 5% more elements than $\mathrm{PO}_{\mathcal{W}}$.

In all our experimental testing we have $\mathcal{W}$ of maximal dimension, i.e., $dim(\mathcal{W}) = p-1$, so that $\mathcal{W}$ is of the same dimension as the unit $(p-1)$-simplex, $\mathcal{U}$.

Tables 3.2 and 3.3 give results for testing $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ (Section 3.8.2), where each figure is an average over 100 instances in which the initial test 3.8.2(1) was inconclusive (i.e., failed to determine whether or not $\mathrm{A} \succcurlyeq^{\mathcal{W}}_{\forall\forall\exists} \mathrm{B}$ holds), and the size of the set B after the $\mathrm{UD}_{\mathcal{W}}$ filtering 3.8.2(2) was greater than zero.

Table 3.2 shows how the input sets A and B were reduced by the $\mathrm{UD}_{\mathcal{W}}$ filtering 3.8.2 (2). As we can see, increasing the size of $dim(\mathcal{W})$, the number of undominated elements increase and therefore the number of elements removed by the $\mathrm{UD}_{\mathcal{W}}$ filtering reduces.

Table 3.3 gives average execution time of the preliminaries steps and the methods 3(a), 3(b) and 3(c) of Section 3.8.2. The checking of the necessary and the sufficient condition in 3.8.2(1) were very effective: on approximately 94%

Table 3.3: Execution time of methods for testing the dominance A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B (Section 3.8.2), i.e., testing the necessary and the sufficient condition (NSc) (1), $\mathrm{UD}_{\mathcal{W}}$ filtering (2) and algorithms T_{LP} 3(a), $\mathrm{T}_{\mathrm{EPU}}$ 3(b) and $\mathrm{T}_{\mathrm{EEU}}$ 3(c) for testing A $\succcurlyeq^{\mathcal{W}}_{\forall\exists}$ B, with respect to $dim(\mathcal{W})$ with $|\mathrm{A}| = |\mathrm{B}| = 500$ and 4 user preferences.

$dim(\mathcal{W})$	NSc[s]	$\mathrm{UD}_{\mathcal{W}}$[s]	T_{LP}[s]	$\mathrm{T}_{\mathrm{EPU}}$[s]	$\mathrm{T}_{\mathrm{EEU}}$[s]
2	0.008	0.028	0.016	0.001	0.001
3	0.010	0.070	0.038	0.003	0.002
4	0.013	0.223	0.131	0.015	0.013
5	0.015	0.504	0.238	0.105	0.088
6	0.015	0.858	0.967	1.179	1.028
7	0.016	1.487	2.232	24.452	14.97

of the problems generated with our random problem generator, the necessary condition failed, or the sufficient condition succeeded, allowing the algorithm to stop in advance. On average, method 3(c) seems to be faster than method 3(b), and the LP method seems to the fastest for $dim(\mathcal{W}) \geq 6$. As for the previous case, the EEU methods are much worse for the case of $dim(\mathcal{W}) = 7$.

3.10 Conclusions

We defined natural notions of equivalence and dominance for a general model of sets of multi-attribute utility, and proved general properties. Computationally we focused especially on the linear (weighted sum) case and we proved that there is a unique setwise-minimal equivalent subset of any (equivalence-free) set of utility vectors A. This set then equals the set of possibly strictly optimal alternatives $\mathrm{PSO}(\mathrm{A})$, and is a compact representation of the utility function for A, giving the utility achievable with A for each scenario. We show that filtering a query with the PSO operator avoids the potential of inconsistency in the user response. Along with pre-processing techniques we developed a linear programming method for generating $\mathrm{PSO}(\mathrm{A})$, and a method based on computing the extreme points of the epigraph of the utility function (EEU), as well as related methods for testing dominance. We implemented the approaches and our testing on random problems showed that both methods scaled to substantially sized problems, with the EEU method being better for lower dimensions.

Chapter 4

A Multi-objective Framework based on User-Preferences

This chapter presents a novel incremental preference elicitation framework for a supplier selection process based on a weighted sum utility function parameterised with respect to the set of weights. A realistic medium-size factory inspires constraints and objectives of the underlying optimisation problem. However, the preference elicitation framework applies to generic multi-criteria configuration problems (see, e.g., [BPPS06, BB07, BL19]).

The main idea is to solve a combinatorial problem multiple times optimally with different weights assigned to the objectives. Afterwards, a pair of solutions among those computed is selected through a particular query generation strategy, and the user expresses a preference between them. These two steps are repeated in a preference learning loop that stops when we find a solution with worst-case loss below a certain threshold. As query generation strategies, we will consider the setwise minimax regret and two other methods based on a novel measure that we call discrepancy, strongly related to the max regret measure.

With our computational experiments, we show that our framework is suitable for large instances. We compare the performance of three different query generation strategies in terms of the number of iterations in the learning loop and computation time.

4.1 Introduction

Supplier selection is the process of determining the best suppliers for acquiring the necessary materials for the activities of a firm. Nowadays, this is a key point of a business strategy, in the case of both a small activity like a restaurant and a large one like a corporation. Although the decision-makers still proceed manually in several contexts, many automated methods and tools have been adopted to solve the problem. This is not just to do with reducing processing time or optimising the cost; it is a much complex problem in which different criteria need to be considered in order to sharpen the company's competitiveness. For example, common criteria used to evaluate a supplier are such as lead time, product quality, resilience and reputation (see, e.g., [Abd13]). A recent review on the topic [Ste17] gives a qualitative ranking of the most used criteria. Still, it is not easy to quantify the relative importance of each criterion, and thus a preference elicitation process seems to suit this context well.

Here we consider a supplier selection process inspired by a real-world problem where the alternatives are resented as a configuration problem; evaluation criteria, constraints and instance structure come from a medium-sized factory. This work lies between two research areas. On the one hand, it provides an alternative perspective to the solution of supplier selection problems. On the other hand, it presents an interactive preference elicitation approach with novel query selection strategies.

In Section 4.2 we provide a literature review including work on supplier selection and user preferences. The assumptions made in relation to the problem definition are discussed in Section 4.3. The notation used across the chapter is presented in Section 4.4. The structure of the framework is described in Section 4.5. The two main blocks of the framework are:

- a Mixed Integer Linear Programming model used for the solution of the combinatorial optimisation problem (described in Section 4.5.1);
- Interactive Preference Elicitation strategies based on the minimax regret criterion for computing the queries posed to the user (described in Section 4.5.2).

The main measures that we consider for evaluating the performance of the framework are the number of interactions with the user and the computational

time required to execute the whole loop. Section 4.6 presents some computational results showing how the framework performs with respect to these measures, analysing the impact of the different user-preference elicitation strategies. Finally, Section 4.7 provides some conclusions about this work and future developments.

4.2 Literature review

Supplier selection literature is very rich. In particular, many techniques have been developed and tailored to solve specific versions of the problem, with different constraints/objectives. A few surveys, such as [WCB91, AHH07, WSB12, ZFS16], provide a deep introduction on quantitative and qualitative methods used. Recently, a pair of papers [CLN13, CN20] analyse advancements in the area of supplier selection from 2008 to 2012 and from 2013 to 2018, respectively. The second paper includes possible future trends aiming at making supplier selection a more interdisciplinary field, in which economic theory, big data analysis, risk analysis and game theory interact.

The most common techniques used to solve supplier selection problems belong to the following areas: Multi-Criteria Decision-Making (MCDM), Mathematical Programming (MP), Meta-heuristics, and Artificial Intelligence (AI). The approach used in our framework involves MCDM, MP and AI methods, following the interdisciplinary future for supplier selection depicted in [CN20]. MCDM approaches are generally used to evaluate a set of discrete options according to several conflicting criteria. Many different methods belong to this family, such as the Analytical Hierarchy Process (AHP) [T.L88] which is based on the ranking of the evaluation criteria, and Analytical Network Process (ANP) [Saa08] which can be used instead of AHP when the evaluation criteria are not preferentially independent. See [RS16] for a review on MCDM supplier selection approaches, and [FGE05a] for surveys on MCDM methods.

MP approaches for supplier selection solve a mathematical model in which one or more objective functions are specified, and the solutions have to respect a set of constraints. The nature of the approach used depends on the type of model. Many supplier selection models are based on Mixed Integer Linear Programming (MILP), due to the effectiveness of modern solvers. MILP models are linear models with potentially both continuous and discrete variables and only one objective. It should be noted that the single objective can be a weighted

sum of different linear objectives, where the weights are considered as parameters. Many MILP based approaches have been developed for supplier selection, such as [RD11, CBGVTG15, APW+18, AMD19]. Non-Linear Programming (NLP) models have also been exploited [WLX11, AV15, AV18, KCM17]. Among other MP techniques used for supplier selection, Data Envelopment Analysis [KD14], Fuzzy Programming [SSYT12] and Goal Programming [HY16] have been widely studied. Finally, Stochastic Programming has received increasing attention recently, since different stochastic aspects, such as demand uncertainty [ACAL+16], currency fluctuation [HTF14] and disruption risk [HGM14], can be considered in a stochastic supplier selection model.

Meta-heuristic approaches are often used when the mathematical problem is too complex to be solved exactly. The price that decision-makers pay is to lose solution optimality (in the case of a single objective problem) or to approximate the Pareto frontier (in the case of multi-objective problems). Some of the meta-heuristics used to tackle supplier selection problems are Genetic Algorithms [DGH+15] and Particle Swarm optimisation [Che17].

AI techniques have also been widely used. In particular, Neural Networks have been exploited to evaluate suppliers according to a set of performance data [TFDCSA16]. Other AI techniques used in supplier selection are Bayesian Networks [HB16, NY15] and Rough Set Theory [CL14].

Recently, a similar framework has been developed in [BL19] to deal with complex combinatorial problems in which computing an optimal solution with respect to known user preference weights vector is computational demanding. The main difference with respect to our general framework is in the methods tested for query selection; see the discussion at the end of Section 4.5.2.3.

Note that we assume the correctness of the user responses, which means that an incorrect answer could well reduce the quality of the final recommendation. In the literature we can find utility models defined to deal with noisy user responses [HWI03, VB10, TPV16, DTP18] at the expense of increased time complexity. Because our problem is computationally challenging even for optimisation with a known objective function, we focus on a model using a standard utility representation.

There is some previous work on preference elicitation approaches for supplier selection. For example, in [CN15] the authors investigate how to elicit human preferences using a *hesitant fuzzy preference relation* in order to deal with

ambiguous opinions of several decision-makers. In works such as [CPRV11, CR11] the authors consider a linear programming model for the problem of supplier selection for multiple collaborating businesses. An advantage of our approach over these previous approaches is that the latter involve the elicitation of a potentially large number of numerical values. These can be very time-consuming and difficult to assess. In contrast, our approach involves intuitive comparison queries and attempts to limit the number of queries asked to the user. Also, to the best of our knowledge, there are no works describing an iterative preference-based multi-criteria supplier selection problem to satisfy the demand for a set of products.

4.3 Problem Requirements

The problem requirements around which our framework is designed come from a real-world study. More specifically, we interacted with the supply chain management of a medium-sized manufacturing factory by asking for some information about their internal supplier selection process. As a result of this interaction, we included in our problem formulation a set of suppliers' evaluation criteria and constraints. Furthermore, although the instances considered in Section 4.6 are artificially generated, they are aligned with this real-world scenario.

The problem consists of computing the quantities to be ordered for a certain time horizon from each supplier to satisfy the demand. Upper and lower limits on the number of suppliers per component are given as input. This is because the decision-maker wants to have control on defining some backup suppliers in case of unexpected disruptions. A catalogue of available suppliers is also given as input, including price and availability of each component.

Four different evaluation criteria are considered in the factory's supplier selection process. The first supplier evaluation criterion considered is cost, including both the direct costs for all the materials and the activation costs of establishing business relationships with suppliers. The price breaks [CFZ93] discount scheme is adopted, meaning that the unit cost is defined depending on how many components of the same type are ordered from the same supplier. This is the standard mechanism adopted by the factory's suppliers to determine the unit costs for a certain material enquiry. The second and third criteria are the supplier lead time and lateness that relate to past experience with

that supplier. They represent the time agreed with a supplier to provide the materials and the lateness with respect to the due date, respectively. The last criterion is the supplier reputation. This is a score assigned by internal experts to each supplier, by considering different aspects, such as disruption risk, the relationship between the company and the supplier, and the strategic vision of the firm.

4.4 Terminology and Definitions

In this section we present the key notations used in this chapter, some of which have been already defined in Chapter 2 and we recall here for the sake of clarity. Let $\mathcal{P}$ be a combinatorial maximisation problem, A $\in \mathcal{M}$ be the set of its feasible solutions (or alternatives) and α be an element of A. Let us define $\mathcal{U}$ to be the initial user preferences state space $\{w \in I\!R^p : \sum_{i=1}^{p} w_i = 1, w_i \geq 0, \forall i = 1, \ldots, p\}$, i.e., the set of all the normalised non-negative weights vectors w. We consider p evaluators, X_i : A $\rightarrow I\!R\ \forall i \in \{1, \ldots, p\}$ over A defining each of the p criteria used to evaluate an alternative, and define the vector $(X_1(\alpha), \ldots, X_p(\alpha))$ as the utility vector of solution α. Note that, in contrast with the previous chapters, an alternative is not directly represented as a vector of reals; as we will show in the next section, it is instead a feasible configuration of the variables of a MILP problem. The utility function of $\alpha \in$ A with respect to $w \in \mathcal{U}$, which is also the objective function of $\mathcal{P}$, is a weighted sum $u_w(\alpha) = w \cdot X(\alpha)$ parameterised with respect to w. Here we define the outcome $X(\alpha)$ of an alternative as $X(\alpha) = (\text{sign}_1 X_1(\alpha), \ldots, \text{sign}_p X_p(\alpha))$ where $\text{sign}_i \in \{1, -1\}$ with $i \in \{1, \ldots, p\}$ defines the sign of the i-th evaluator. The function sign_i relates to whether one is minimising rather than maximising the corresponding objective. We indicate with $\alpha_w \in$ A an optimal solution of $\mathcal{P}$ with respect to the weights vector $w \in \mathcal{U}$, that is a solution α_w such that $u_w(\alpha_w) \geq u_w(\alpha)$ for any $\alpha \in$ A.

Let V_Λ be a convex polyhedron in $I\!R^p$ defined by a set of non-strict linear inequalities Λ; we define $\mathcal{W}_\Lambda$ as the convex and closed (and thus compact) polytope $\mathcal{W}_\Lambda = \mathcal{U} \cap V_\Lambda$. The linear inequalities in Λ can arise from input preferences of the form β *is preferred to* γ, leading to the linear constraint $w \cdot (X(\beta) - X(\gamma)) \geq 0$.

Let *Ext*$(\mathcal{W}_\Lambda)$ be the set of extreme points of a user preference state space $\mathcal{W}_\Lambda$. For each extreme point w we choose an optimal solution α_w, and we define $\mathcal{S}$ to be the set $\{\alpha_w : w \in$ *Ext*$(\mathcal{W}_\Lambda)\}$. We say that $\mathcal{S}$ *is a set of optimal solutions with*

respect to Ext($\mathcal{W}_\Lambda$).

4.5 The Structure of the Framework

We denote by $\mathcal{P}$ the combinatorial optimisation problem given by the MILP model in Section 4.5.1 below. Recall from the previous section that A is the finite set of all the feasible solutions of $\mathcal{P}$. The objective function considered in $\mathcal{P}$ is a weighted sum of four functions $X_1(\alpha)$, $X_2(\alpha)$, $X_3(\alpha)$, $X_4(\alpha)$, associating a measure of the cost, lateness, lead time and reputation with a feasible solution $\alpha \in$ A. The analytic form of these functions is provided in Section 4.5.1.

The weighted sum used as the objective function of $\mathcal{P}$ is $u_w(\alpha) = -w_1X_1(\alpha) - w_2X_2(\alpha) - w_3X_3(\alpha) + w_4X_4(\alpha)$, where $w_i \in [0,1]$ (for all $i \in \{1,2,3,4\}$) is the weight of the i-th function. The first three signs are negative because the first three functions have to be minimised, in contrast with $X_4(\alpha)$ that has to be maximised. The parameters of the MILP model come from different sources. Data such as tariffs and components availability of each supplier come from a supplier catalogue which in our experimental testing is randomly generated, as is the demand of each component. Finally, a lateness/lead time predictor defined in Appendix B.3 is used to predict supplier performances, providing coefficients to be used in $X_2(\alpha)$ and $X_3(\alpha)$. The predictions are performed by relying on a database of components orders, whose entries are referring to a series of orders related to the past. The method used for random generation of the database of past orders used in the framework is described in Appendix B.2.

With the learning loop depicted in Figure 4.1, we iteratively ask questions to the Decision-Maker (DM) in order to estimate a preference vector $w^* = (w_1^*, w_2^*, w_3^*, w_4^*) \in \mathcal{U}$ representing her true preferences. Let us consider as a query $\mathcal{Q}$ a subset of A, associated with a question of the form: which solution do you prefer among the solutions in $\mathcal{Q}$? For example, if $\mathcal{Q} = \{\beta, \gamma\}$, the query amounts to: do you prefer solution β or γ? This is the type of query that we use in the framework to learn about w^*. Each answer to a query implies an inequality of the type: $w \cdot (X(\beta) - X(\gamma)) \geq 0$ or $w \cdot (X(\beta) - X(\gamma)) \leq 0$, depending on the DM's preference between β and γ. At each iteration of the framework, Λ is the polyhedron defined as the set of inequalities derived from the user answers to the queries. Such inequalities reduce the user preference space state $\mathcal{U}$ to $\mathcal{W}_\Lambda$, as indicated in Section 4.4. The framework is based on computing the optimal solutions associated with the extreme points of $\mathcal{W}_\Lambda$,

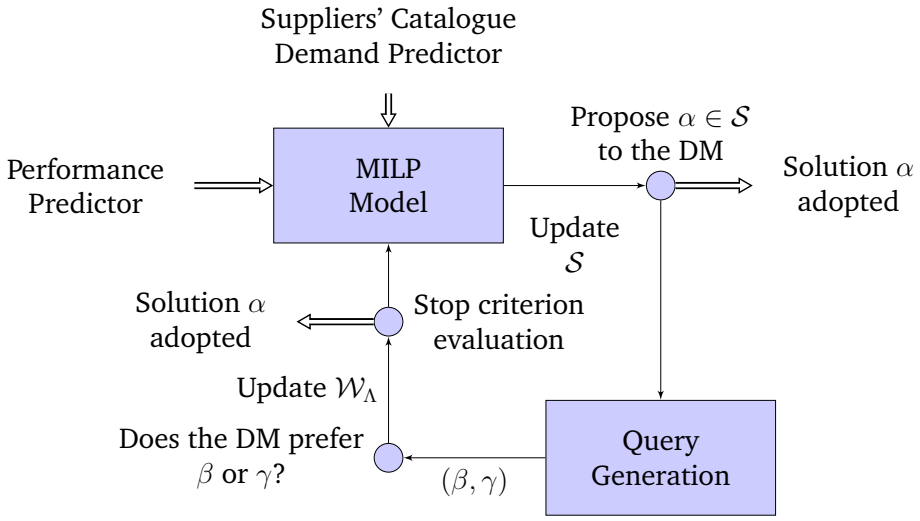


Figure 4.1: Structure of the proposed framework

by constructing the set $\mathcal{S}$ as defined in the previous section. This is because computing the set A is usually intractable due to its size.

Algorithm 13 Supplier Selection Framework

```
1:  procedure SupplierSelectionLearningLoop
2:      Compute MILP parameters
3:          Run performance predictors
4:          Retrieve data from the suppliers' catalogue
5:      Λ ← ∅
6:      W_Λ ← U
7:      repeat
8:          Update S by running the MILP model on the new vertexes of W_Λ
9:          α ← SelectRecommendedSolution(S)
10:         if DM accepts α then return α
11:         (β, γ) ← ComputeQuery(W_Λ, S)
12:         Question to the DM: Do you prefer β or γ?
13:         Update Λ according to the user's answer
14:         Update W_Λ
15:     until StopCriterion(W_Λ, S)
16:     return α
```

The following lines describe how the framework works in practice, referring to the block diagram in Figure 4.1 and the pseudocode depicted in Algorithm 13. The first step is to execute the performance predictors in order to compute lateness and lead time estimation for each supplier (line 3 of Algorithm 13). Furthermore, the components cost and availability per supplier need to be retrieved from the suppliers' catalogue (line 4). These are input parameters for the MILP model described in Section 4.5.1.

The next step is to initialize the set of constraints Λ to $\emptyset$ and thus $\mathcal{W}_\Lambda$ to $\mathcal{U}$ (lines 5 and 6). The MILP model is then solved for each weights vector $w \in \mathit{Ext}(\mathcal{W}_\Lambda)$ (line 8). The first time, this means solving the combinatorial problem four times by optimizing with respect to each single function $X_i(\alpha)$, $i = 1, \ldots, 4$. Recall that $\mathcal{S}$ is the set of solutions generated, following the definition in Section 4.4. A solution $\alpha \in \mathcal{S}$ is selected and proposed to the user, by means of the function SELECTRECOMMENDEDSOLUTION($\mathcal{S}$) called at line 9. If the user accepts the solution, the algorithm stops and provides α as an output (line 10). A pair of solutions (β, γ), with $\beta, \gamma \in \mathcal{S}$, is chosen with a user-preference elicitation strategy, implemented by the function COMPUTEQUERY($\mathcal{W}_\Lambda$,$\mathcal{S}$) at line 11. The user then answers the question (line 12): do you prefer solution β or solution γ? The answer then leads to an update of Λ and $\mathcal{W}_\Lambda$ (lines 13-14). A stopping criterion is then checked by calling the function STOPCRITERION($\mathcal{W}_\Lambda$,$\mathcal{S}$) (line 15), which determines if $\mathcal{W}_\Lambda$ allows one to approximate w^* with a certain accuracy. If the function returns true, the solution α is provided as an output. Otherwise, line 8 is executed again by considering the updated Λ and $\mathcal{W}_\Lambda$, and the MILP model will run on the extreme points of $\mathcal{W}_\Lambda$ that have not been considered in the previous iterations.

As we can see in Figure 4.1, the main components of our framework are the MILP model and the query generation. Section 4.5.1 and Section 4.5.2 describe these components. The description of the functions SELECTRECOMMENDED-SOLUTION($\mathcal{S}$), COMPUTEQUERY($\mathcal{W}_\Lambda$,$\mathcal{S}$), STOPCRITERION($\mathcal{W}_\Lambda$,$\mathcal{S}$) is included in Section 4.5.2.

4.5.1 The Mixed Integer Linear Programming model

Let us consider a set of suppliers I and a set of components C. A set of components C_i is defined for each supplier $i \in \mathrm{I}$, consisting of all the components $j \in \mathrm{C}$ that can be provided by supplier i. Our MILP model generates an optimal solution given a fixed weights vector $w = (w_1, w_2, w_3, w_4)$ which will then be an input parameter. Multiple unit costs are provided by each supplier with respect to a certain part, depending on the quantity bought. A unit cost is associated with a certain quantity interval, meaning that the unit cost is the same for any quantity in the interval. The set $\mathrm{T}_{i,j}$ is the set of all the disjoint quantity intervals for supplier $i \in \mathrm{I}$ and component $j \in \mathrm{C}_i$, whose union covers the set $\mathbb{N}$. Let us define the parameter $m_{i,j,t} \in \mathbb{N}$ as the minimum amount of component $j \in \mathrm{C}_i$ to be ordered from supplier $i \in \mathrm{I}$ in the quantity interval

$t \in \mathrm{T}_{i,j}$. As a consequence, $\mathrm{T}_{i,j} = (\cup_{t=1}^{|\mathrm{T}_{i,j}|-1}[m_{i,j,t}, m_{i,j,t+1} - 1]) \cup [m_{i,j,|\mathrm{T}_{i,j}|}, +\infty]$, where $m_{i,j,1} = 0$. The unit cost associated with a quantity interval $t \in \mathrm{T}_{i,j}$ defines a certain tariff and it is indicated with $c_{i,j,t}$. The value $a_i \in \mathbb{R}_+$ indicates the activation cost of a supplier $i \in \mathrm{I}$. Note that all the parameters mentioned so far, regarding components cost and availability, are coming from the suppliers' catalogue of the factory.

The parameters $l_{i,j,t} \in \mathbb{R}_+$ and $\delta_{i,j,t} \in \mathbb{R}_+$ respectively represent the expected lead time and the expected lateness of component $j \in \mathrm{C}_i$ ordered from $i \in \mathrm{I}$ in the quantity interval $t \in \mathrm{T}_{i,j}$. These parameters are computed by the lateness/lead time predictor. The value $r_j \in \{1, .., 100\}$ is the reputation of supplier $i \in \mathrm{I}$. This value is assigned by internal experts, as mentioned in Section 4.3. The values $\lambda_{j,min}, \lambda_{j,max} \in \mathbb{N}$ are bounds on the number of suppliers for component $j \in \mathrm{C}$. Finally, $D_j \in \mathbb{N}$ is the estimated demand of component $j \in \mathrm{C}$.

Our MILP model is based on the following integer decision variables:

- $\phi_{i,j,t} \in \mathbb{N}$ is the number of component $j \in \mathrm{C}_i$ ordered from supplier $i \in \mathrm{I}$ in the quantity interval $t \in \mathrm{T}_{i,j}$
- $\rho_{i,j,t} \in \{0, 1\}$ is equal to 1 if a positive quantity of component $j \in \mathrm{C}_i$ is ordered from $i \in \mathrm{I}$ in the quantity interval $t \in \mathrm{T}_{i,j}$, 0 otherwise
- $\tau_i \in \{0, 1\}$ is equal to 1 if at least one component is ordered from the supplier $i \in \mathrm{I}$, 0 otherwise
- $\theta_1, \theta_2, \theta_3, \theta_4 \in \mathbb{R}_+$ are auxiliary variables used to model the min-max/max-min formulations of the objectives.

Note that the variables $\phi_{i,j,t}$ and $\rho_{i,j,t}$ have three indexes in order to take into account different costs, lead time and lateness for each triple of supplier i, component j and quantity interval t.

A feasible solution $\alpha \in \mathrm{A}$ is determined by a feasible assignment to all these variables. The four functions $X_1(\alpha)$, $X_2(\alpha)$, $X_3(\alpha)$, $X_4(\alpha)$ are defined as follows. First, the cost is computed as:

$$X_1(\alpha) = \sum_{i \in \mathrm{I}, j \in \mathrm{C}, t \in \mathrm{T}_{i,j}} c_{i,j,t}\phi_{i,j,t} + \sum_{i \in \mathrm{I}} a_i \tau_i \tag{4.1}$$

thus both direct costs and suppliers' activation costs are taken into account. The

first goal is to minimise this quantity. The second and third objectives are:

$$X_2(\alpha) = \max_{i \in \mathrm{I}, j \in \mathrm{C}_i} \sum_{t \in \mathrm{T}_{i,j}} l_{i,j,t} \rho_{i,j,t} \tag{4.2}$$

$$X_3(\alpha) = \max_{i \in \mathrm{I}, j \in \mathrm{C}_i} \sum_{t \in \mathrm{T}_{i,j}} \delta_{i,j,t} \rho_{i,j,t}. \tag{4.3}$$

They represent the maximum expected lead time and the maximum expected lateness related to a certain component and supplier, which are considered as measures of the quality of service. Our goal is to minimise these quantities. The fourth and last objective is

$$X_4(\alpha) = \min_{i \in \mathrm{I}} r_i \tau_i \tag{4.4}$$

which we want to maximise, since it indicates the minimum reputation among the suppliers considered in the solution.

The complete MILP model is as follows:

$$\max -w_1\theta_1 - w_2\theta_2 - w_3\theta_3 + w_4\theta_4 \tag{4.5}$$

$$\sum_{i\in \mathrm{I}} \sum_{t\in \mathrm{T}_{i,j}} \phi_{i,j,t} \geq D_j \qquad \forall j \in \mathrm{C} \tag{4.6}$$

$$\phi_{i,j,t} \geq m_{i,j,t} y_{i,j,t} \qquad \forall i \in \mathrm{I}, j \in \mathrm{C}, t \in \mathrm{T}_{i,j} \tag{4.7}$$

$$\phi_{i,j,t} \leq M_1 \rho_{i,j,t} \qquad \forall i \in \mathrm{I}, j \in \mathrm{C}, t \in \mathrm{T}_{i,j} \tag{4.8}$$

$$\sum_{t\in \mathrm{T}_{i,j}} \rho_{i,j,t} \leq 1 \qquad \forall j \in \mathrm{C}, i \in \mathrm{I} \tag{4.9}$$

$$\sum_{i\in \mathrm{I}, t\in \mathrm{T}_{i,j}} \rho_{i,j,t} \geq \lambda_{j,min} \qquad \forall j \in \mathrm{C} \tag{4.10}$$

$$\sum_{i\in \mathrm{I}, t\in \mathrm{T}_{i,j}} \rho_{i,j,t} \leq \lambda_{j,max} \qquad \forall j \in \mathrm{C} \tag{4.11}$$

$$\theta_1 = \sum_{j\in \mathrm{C}, i\in \mathrm{I}, t\in \mathrm{T}_{i,j}} c_{i,j,t}\phi_{i,j,t} + \sum_{i\in \mathrm{I}} a_i \tau_i \tag{4.12}$$

$$\theta_2 \geq \sum_{t\in \mathrm{T}_{i,j}} l_{i,j,t}\rho_{i,j,t} \qquad \forall j \in \mathrm{C}, i \in \mathrm{I} \tag{4.13}$$

$$\theta_3 \geq \sum_{t\in \mathrm{T}_{i,j}} \delta_{i,j,t}\rho_{i,j,t} \qquad \forall j \in \mathrm{C}, i \in \mathrm{I} \tag{4.14}$$

$$\theta_4 \leq M_2(1-\tau_i) + r_i\tau_i \qquad \forall i \in \mathrm{I} \tag{4.15}$$

$$\sum_{j\in \mathrm{C}, t\in \mathrm{T}_{i,j}} \rho_{i,j,t} \leq M_3 \tau_i \qquad \forall i \in \mathrm{I} \tag{4.16}$$

$$\sum_{j\in \mathrm{C}, t\in \mathrm{T}_{i,j}} \rho_{i,j,t} \geq \tau_i \qquad \forall i \in \mathrm{I} \tag{4.17}$$

$$\tag{4.18}$$

where $M_1, M_2, M_3 \in \mathbb{R}_+$ are large enough ("big-M") constants and the other variables/parameters are defined previously. In our implementation, we set M_1 as the maximum expected demand with respect to all the components of the catalogue, M_2 as the maximum reputation with respect to all the suppliers of the catalogue, and M_3 as the cardinality of C. The objective function (4.5) is the weighted sum of the auxiliary variables $\theta_1, \theta_2, \theta_3, \theta_4$, where the signs are minus for the functions to be minimised and plus for the ones to be maximised. Constraint (4.6) imposes the condition that the demand per part has to be satisfied. Constraints (4.7) and (4.8) are linking constraints between $\phi_{i,j,t}$ and $\rho_{i,j,t}$, which state that $\rho_{i,j,t}$ is active if and only if $\phi_{i,j,t}$ is greater than the minimum quantity $m_{i,j,t}$ to unlock the tariff. Constraint (4.9) forces that only one tariff is used when we order a certain quantity from a supplier. Constraints

(4.10) and (4.11) impose the bounds on the number of suppliers to be selected for each component. Constraint (4.12) links θ_1 with the analytical expression of $X_1(\alpha)$. Constraints (4.13) and (4.14) are used for the min-max formulations, so that the auxiliary variables θ_2, θ_3 are linked to $X_2(\alpha), X_3(\alpha)$ when the model is solved. Analogously, Constraints (4.15) is used for the max-min formulation regarding $X_4(\alpha)$. In fact, the expression $M_2(1-\tau_i)+r_i\tau_i$ is equal to r_i in the case the supplier is selected, and equal to M_2 otherwise, meaning that the constraint (4.16) is disabled in the latter case. This expression is then linked to θ_4. Finally, Constraints (4.15) and (4.16) are linking constraints for $\rho_{i,j,t}$ and τ_i, imposing that a certain supplier is active if and only if one component is ordered from it.

4.5.2 User-preference elicitation approach

A key point for a good user experience is to reduce the number of interactions with the user by asking informative queries. In this section, we define different strategies for the query generation, in order to study their impact on the number of iterations required by the framework to converge towards a stopping criterion. In Section 4.5.2.1 we recall the main concepts of the minimax criterion and we define the function SELECTRECOMMENDEDSOLUTION($\mathcal{S}$). Section 4.5.2.2 introduces the *discrepancy measure* which is a measure related to the minimax regret criterion that will be used for query selection. Section 4.5.2.3 presents different query generation strategies, then different implementations of the function COMPUTEQUERY($\mathcal{W}_\Lambda$,$\mathcal{S}$). Finally, Section 4.5.2.4 defines the stopping criterion used in the framework, that is the implementation of STOPCRITERION($\mathcal{W}_\Lambda$,$\mathcal{S}$).

4.5.2.1 Max regret

Recall from Section 2.3.2 that the maximum regret of a feasible solution $\alpha \in \mathrm{A}$ with respect to the user preference state space $\mathcal{W}_\Lambda$ is given by:

$$MR_{\mathcal{W}_\Lambda}(\alpha, \mathrm{A}) = \max_{\beta \in \mathrm{A}} \max_{w \in \mathcal{W}_\Lambda} (w \cdot (X(\beta) - X(\alpha))). \tag{4.19}$$

As mentioned earlier, computing the set A of feasible solutions is not practically feasible. However, the following lemma (based on a well known property of maximum regret see Section 2.3.2.3) allows us to compute the maximum regret of a solution $\alpha \in \mathrm{A}$ with respect to any $w \in \mathcal{W}_\Lambda$ and $\beta \in \mathrm{A}$ using just the set *Ext*($\mathcal{W}_\Lambda$) of extreme points of $\mathcal{W}_\Lambda$ and the corresponding set $\mathcal{S}$ of optimal

solutions.

Lemma 4.5.1. *Let A be the set of all the feasible solutions with respect to $\mathcal{W}_\Lambda$, let α be an element of A and let $\mathcal{S}$ be a set of optimal solutions with respect to Ext($\mathcal{W}_\Lambda$). Then $MR_{\mathcal{W}_\Lambda}(\alpha, A) = MR_{Ext(\mathcal{W}_\Lambda)}(\alpha, \mathcal{S})$*

The function SELECTRECOMMENDEDSOLUTION($\mathcal{S}$) of Algorithm 13 selects a solution in A that minimise the worst-case loss with respect to $\mathcal{W}_\Lambda$, i.e., by Lemma 4.5.1, a solution that minimise $MR_{Ext(\mathcal{W}_\Lambda)}(\alpha, \mathcal{S})$.

Let $\text{Val}_\text{A}(w)$ be $\max_{\alpha \in \text{A}}(u_w(\alpha))$ (i.e., the maximum scalar utility we can get from solutions $\alpha \in$ A supposing that the weights vector is $w \in \mathcal{W}_\Lambda$). Recall from Section 2.3.2.1 that the setwise maximum regret (SMR) for a subset $\mathcal{Q} \subseteq$ A with respect to the user preference state space $\mathcal{W}_\Lambda$ is defined as:

$$\begin{aligned} SMR_{\mathcal{W}_\Lambda}(\mathcal{Q}, \text{A}) &= \max_{\beta \in \text{A}} \max_{w \in \mathcal{W}_\Lambda} (w \cdot X(\beta) - \max_{\alpha \in \mathcal{Q}} (w \cdot X(\alpha))) \\ &= \max_{w \in \mathcal{W}_\Lambda} (\text{Val}_\text{A}(w) - \text{Val}_\mathcal{Q}(w)). \end{aligned} \tag{4.20}$$

The setwise max regret will be the base of one of the three strategies for query generation.

4.5.2.2 Discrepancy measure

Recall from Section 4.4 that $\alpha_w \in$ A is a an optimal solution computed from the discrete optimisation problem with respect to $w \in \mathcal{W}_\Lambda$. We define the *discrepancy* of $\alpha \in$ A with respect to w as

$$D_w(\alpha) = w \cdot (X(\alpha_w) - X(\alpha)). \tag{4.21}$$

This is a measure of how good the solution α is, supposing that the user weights vector is w. Note that $D_w(\alpha) \geq 0$ for any $\beta \in$ A since α_w is an optimal solution with respect to w, i.e., $w \cdot X(\alpha_w) \geq w \cdot X(\alpha)$ for any $\alpha \in$ A. We will use this measure in order to select a query composed by two solutions $\alpha_{w_1}, \alpha_{w_2} \in \mathcal{S}$ with high values $D_{w_1}(\alpha_{w_2})$ and $D_{w_2}(\alpha_{w_1})$. The idea is to ask to the user her preference between two optimal solutions that are maximally different with respect to the corresponding weights vectors in order to get a high value of information from her answer.

Since $\mathcal{S}$ is the set of optimal solutions with respect to the extreme points *Ext*($\mathcal{W}_\Lambda$) of the user preference state space $\mathcal{W}$, by Lemma 4.5.1, $MR_{\mathcal{W}_\Lambda}(\alpha, \text{A}) =$

$MR_{Ext(\mathcal{W}_\Lambda)}(\alpha, \mathcal{S}) = \max_{w \in Ext(\mathcal{W}_\Lambda)} \max_{\beta \in \mathcal{S}}(w \cdot (X(\beta) - X(\alpha)))$, which can be written as $\max_{w \in Ext(\mathcal{W}_\Lambda)}(w \cdot (X(\alpha_w) - X(\alpha)))$. Thus, the maximum regret of a solution can be expressed using the discrepancy function:

$$MR_{\mathcal{W}_\Lambda}(\alpha, \mathrm{A}) = MR_{\mathcal{W}_\Lambda}(\alpha, \mathcal{S}) = \max_{w \in Ext(\mathcal{W}_\Lambda)} D_w(\alpha). \tag{4.22}$$

4.5.2.3 Query generation

Recall from Section 2.4.2 that a solution $\beta \in \mathcal{S}$ is *undominated* in $\mathcal{S}$ with respect to $\mathcal{W}_\Lambda$, i.e., $\beta \in \mathrm{UD}_{\mathcal{W}_\Lambda}(\mathcal{S})$, if there does not exist $\gamma \in \mathcal{S}$ such that (i) $w \cdot X(\gamma) \geq w \cdot X(\beta)$ for all $w \in \mathcal{W}_\Lambda$, and (ii) $w \cdot X(\gamma) > w \cdot X(\beta)$ for at least one $w \in \mathcal{W}_\Lambda$. Also, recall from Section 3.3.1 that $\mathcal{S}$ is *equivalence-free* with respect to $\mathcal{W}_\Lambda$ if and only if $\beta \not\equiv_{\mathcal{W}_\Lambda} \gamma$ for all $\beta, \gamma \in \mathcal{S}$ with , which is if and only if there do not exist $\beta, \gamma \in \mathcal{S}$ such that $w \cdot X(\beta) = w \cdot X(\gamma)$ for all $w \in \mathcal{W}_\Lambda$.

To ensures the consistency of the DM's preference model after a response to a binary query $\mathcal{Q} = \{\beta, \gamma\}$, the alternatives β and γ must be strongly feasible answers given $\mathcal{W}_\Lambda$ (see Section 3.5). In our context this can be ensured making $\mathcal{S}$ equivalence-free and replacing it with $\mathrm{UD}_{\mathcal{W}_\Lambda}(\mathcal{S})$. In this case, in fact, for any query $\mathcal{Q} = \{\beta, \gamma\}$ we have that γ does not dominate β and $\beta \not\equiv_{\mathcal{W}_\Lambda} \gamma$. Thus, there exists $w_1 \in \mathcal{W}_\Lambda$ with $w_1 \cdot X(\beta) > w_1 \cdot X(\gamma)$. Similarly, there exists also $w_2 \in \mathcal{W}_\Lambda$ with $w_2 \cdot X(\gamma) > w_2 \cdot X(\beta)$. This means that $\beta, \gamma \in \mathrm{PSO}_{\mathcal{W}_\Lambda}(\mathcal{Q})$, which is, by Proposition 4, if and only if β and γ are strongly feasible answers since $\mathcal{Q}$ is equivalence-free.

Note that $\mathrm{UD}_{\mathcal{W}_\Lambda}(\mathcal{S}) = \mathrm{UD}_{Ext(\mathcal{W}_\Lambda)}(\mathcal{S})$ since the scalar utility of a solution is a linear function with respect to $w \in \mathcal{W}_\Lambda$. Thus we can compute $\mathrm{UD}_{\mathcal{W}_\Lambda}(\mathcal{S})$ and at the same time make $\mathcal{S}$ equivalence-free as follows. If it is the case that $w \cdot (X(\beta) - X(\gamma)) = 0$ for all $w \in Ext(\mathcal{W}_\Lambda)$, then we remove either β or γ. Thereafter, we remove all $\gamma \in \mathcal{S}$ such that there exists $\beta \in \mathcal{S}$ with $w \cdot X(\gamma) \leq w \cdot X(\beta)$ for all $w \in Ext(\mathcal{W}_\Lambda)$.

Once we make $\mathcal{S}$ equivalence-free and devoid of dominated elements, we can then proceed with the query selection process ensuring the consistency of the preference model. We consider the following three methods to generate a binary query $\mathcal{Q} = \{\alpha_{w_1}, \alpha_{w_2}\}$ from $\mathcal{S}$ (with their relative performance being compared in Section 4.6):

1. Setwise min max regret (SMMR): select a query $\mathcal{Q} \subseteq \mathcal{S}$ with $|\mathcal{Q}| = 2$ that minimises $SMR_{\mathcal{W}_\Lambda}(\mathcal{Q}, \mathcal{S})$.

2. Max min discrepancy (MMD): select a query $\mathcal{Q} \subseteq \mathcal{S}$ with $|\mathcal{Q}| = 2$ that maximises $\min(D_{w_2}(\alpha_{w_1}), D_{w_1}(\alpha_{w_2}))$.
3. Max discrepancy sum (MDS): select a query $\mathcal{Q} \subseteq \mathcal{S}$ with $|\mathcal{Q}| = 2$ that maximises $D_{w_2}(\alpha_{w_1}) + D_{w_1}(\alpha_{w_2}) = (w_1 - w_2) \cdot (X(\alpha_{w_2}) - X(\alpha_{w_1}))$.

Each of these methods can be used to implement COMPUTEQUERY($\mathcal{W}_\Lambda$,$\mathcal{S}$) used in Algorithm 13.

SMMR combines the quality of solutions with being myopically optimal (see Section 2.3.2.2). This ensures a good diversity of solutions shown to the user, but computing the query that minimises the setwise maximum regret is quite expensive since we need to solve $O(|\mathcal{S}|^3)$ linear programming problems. This is because we have to evaluate the SMR of each possible query $\mathcal{Q}$, and for each query $\mathcal{Q}$ we need to solve $O(|\mathcal{S}|)$ linear programming problems (see Section 2.3.2.3 for the computational complexity of solving linear programming problems). MDS and MMD are two simpler methods we developed that consider only the two weights vectors associated with the solutions composing the query rather than the whole user preference state space $\mathcal{W}_\Lambda$. The aim is still to be maximally informative but with a lower complexity for the evaluation of each query. In fact, with these two methods we need to compute $O(|\mathcal{S}|^2)$ dot products. Furthermore, we can store and reuse the value of a query for subsequent iterations in cases when the corresponding extreme points are not removed by the preference elicitation process.

Example 16. *Consider the set of alternatives* $\mathcal{S} = \{\alpha = (4,1,2,1),\ \beta = (3,4,1,1), \gamma = (1,3,4,1), \theta = (3,3,1,4)\}$ *associated with the extreme points* $Ext(\mathcal{W}_\Lambda) = \{(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1)\}$. *In Table 4.1 are shown the values computed by the three query selection criteria with respect to each possible query. In this example the myopically optimal query* $\mathcal{Q} = \{\gamma, \theta\}$ *is also selected by the two methods based on the discrepancy measure.*

A very recent paper [BL19] proposes a similar interactive preference elicitation procedure. As in our framework, queries for the user are computed using the solutions associated with the extreme points of the polytope representing the preferences learned so far. The experimental results show that their best method for query selection is *Max-Dist* where the query is composed of a pair of solutions that maximise the corresponding Euclidean distance. During the development of our framework, we considered this method but discarded it because our initial experimental results indicated that it was not performing

Table 4.1: Values computed by the three query selection criteria with respect to each possible query selected from the set $\mathcal{S} = \{\alpha = (4,1,2,1),\ \beta = (3,4,1,1), \gamma = (1,3,4,1), \theta = (3,3,1,4)\}$ with corresponding extreme points $\mathit{Ext}(\mathcal{W}_\Lambda) = \{(1,0,0,0), (0,1,0,0),\ (0,0,1,0), (0,0,0,1)\}$

$\mathcal{Q}$	$\mathit{SMR}_{\mathcal{W}_\Lambda}(\mathcal{Q},\mathcal{S})$	$\min(D_{w_2}(\alpha_{w_1}), D_{w_1}(\alpha_{w_2}))$	$D_{w_2}(\alpha_{w_1}) + D_{w_1}(\alpha_{w_2})$
$\{\alpha,\beta\}$	3	1	4
$\{\alpha,\beta\}$	3	2	5
$\{\alpha,\theta\}$	2	1	4
$\{\beta,\gamma\}$	3	1	4
$\{\beta,\theta\}$	3	1	4
$\{\gamma,\theta\}$	1	3	6

well compared to the other methods we are presenting in this chapter. We believe that the poor efficacy of such method applied to our context is due to its high sensitivity to the (somewhat arbitrary choice of) scales of the objectives of the utility function to be optimised. Note that the idea behind our MDS method is somewhat similar, since we select a pair of solutions that maximise $(w_1 - w_2) \cdot (X(\alpha_{w_2}) - X(\alpha_{w_1}))$, i.e., the dot product between (i) the difference between the corresponding utility vectors, and (ii) the difference of the utilities corresponding extreme points. It may well be that MDS performs better in our context because it is much less sensitive to changes in the particular choices of utility scales.

4.5.2.4 Stopping criterion

Recall from Section 2.4.2 that $\mathrm{NO}_{\mathcal{W}_\Lambda}(\mathrm{A})$ is the set of the *necessarily optimal* solutions of A with respect to $\mathcal{W}_\Lambda$, i.e., the set of solutions $\beta \in$ A such that $w \cdot (X(\beta) - X(\gamma)) \geq 0$ for any $\gamma \in$ A and for any $w \in \mathcal{W}_\Lambda$. Note that usually there are not any necessarily optimal solutions, unless $\mathcal{W}_\Lambda$ is a small set. Also, if there is more than one necessarily optimal element then they are all equivalent. If there exists a solution $\beta \in$ A such that $D_{w_2}(\beta) = 0$ for all $w_2 \in \mathit{Ext}(\mathcal{W}_\Lambda)$ then, since $\mathcal{W}_\Lambda$ is a convex and compact set, there is no solution better than β with respect to the user preference state space $\mathcal{W}_\Lambda$, i.e., $\beta \in \mathrm{NO}_{\mathcal{W}_\Lambda}(\mathrm{A})$. Also, as is well known (see e.g., [Tim13] or [BP19b]), $\beta \in \mathrm{NO}_{\mathcal{W}_\Lambda}(\mathrm{A})$ if and only if $\mathit{MR}_{\mathcal{W}_\Lambda}(\beta, \mathrm{A}) = 0$. These equivalences are expressed more formally by the following proposition.

Proposition 6. *Let $\alpha \in A$ be a feasible solution, then the following statements are equivalent:*

(a) $D_{w_2}(\alpha) = 0$ *for all* $w_2 \in$ *Ext*$(\mathcal{W}_\Lambda)$*;*

(b) $\alpha \in \mathrm{NO}_{\mathcal{W}_\Lambda}(A)$*;*

(c) $MR_{\mathcal{W}_\Lambda}(\alpha, A) = 0$.

Proof: (a)⇒(b): If $D_{w_2}(\alpha) = 0$ for each $w_2 \in$ *Ext*$(\mathcal{W}_\Lambda)$, then $w_2 \cdot (X(\alpha_{w_2}) - X(\alpha)) = 0$ for each $w_2 \in$ *Ext*$(\mathcal{W}_\Lambda)$. Therefore, since α_{w_2} is an optimal solution with respect to w_2, also α is optimal for all $w_2 \in$ *Ext*$(\mathcal{W}_\Lambda)$ and then $w \cdot (X(\alpha) - X(\beta)) \geq 0$ for each $w \in$ *Ext*$(\mathcal{W}_\Lambda)$ and for any $\beta \in$ A. Since $\mathcal{W}_\Lambda$ is convex and compact, any $w' \in \mathcal{W}_\Lambda$ can be expressed as a convex combination of extreme points in *Ext*$(\mathcal{W}_\Lambda) = \{w_1, \ldots, w_n\}$, i.e., $w' = \sum_{i=1}^n \lambda_i w_i$ for some $\lambda_i \in [0, 1]$ such that $\sum_{i=1}^n \lambda_i = 1$, then $w' \cdot (X(\alpha) - X(\beta)) = \sum_{i=1}^n \lambda_i w_i (X(\alpha) - X(\beta)) \geq 0$, and then α is optimal for any $w' \in \mathcal{W}_\Lambda$, i.e., $\alpha \in \mathrm{NO}_{\mathcal{W}_\Lambda}$(A).

(b)⇒(c) If $\alpha \in \mathrm{NO}_{\mathcal{W}_\Lambda}$(A), then $w \cdot (X(\beta) - X(\alpha)) \leq 0$ for any $\beta \in$ A and for any $w \in \mathcal{W}_\Lambda$. Therefore $MR_{\mathcal{W}_\Lambda}(\alpha, \mathcal{S}) = \max_{\beta \in \mathcal{S}} \max_{w \in \mathcal{W}_\Lambda}(w \cdot (X(\beta) - X(\alpha))) \leq 0$, but since $MR_{\mathcal{W}_\Lambda}(\alpha, \mathcal{S}) \geq 0$, then $MR_{\mathcal{W}_\Lambda}(\alpha, \mathcal{S}) = 0$.

(c)⇒(a) If $MR_{\mathcal{W}_\Lambda}(\alpha, \mathcal{S}) = 0$, since $MR_{\mathcal{W}_\Lambda}(\alpha, \mathcal{S}) = \max_{w_2 \in Ext(\mathcal{W}_\Lambda)}(D_{w_2}(\alpha))$ (see Section 4.5.2.2) and $D_{w_2}(\alpha) \geq 0$ for any $w_2 \in \mathcal{W}_\Lambda$, then $D_{w_2}(\alpha) = 0$ for all $w_2 \in$ *Ext*$(\mathcal{W}_\Lambda)$.

□

Because of Proposition 6, if we find a solution $\alpha \in \mathcal{S}$ such that $D_{w_2}(\alpha) = 0$ for each $w_2 \in$ *Ext*$(\mathcal{W}_\Lambda)$, then we can stop the algorithm and recommend α to the user since it will be an optimal solution with respect to any $w \in \mathcal{W}_\Lambda$.

Our iterative procedure could be repeated until we find a necessarily optimal solution in $\mathcal{S}$. However, if there are several similar solutions which are optimal for at least one DM's preference model $w \in \mathcal{W}$, we may need too many interactions with the user to find a necessarily optimal solution. We then use an alternative stopping criterion which stops the interaction with the user when the minimax regret is below a certain threshold. More precisely, we implement the function STOPCRITERION($\mathcal{W}_\Lambda$,$\mathcal{S}$) defined in Algorithm 13 as follows. We compute the maximum regret $MR_{\mathcal{W}_\Lambda}(\alpha, \mathcal{S})$ of each solution $\alpha \in \mathcal{S}$ and if there is at least one solution with max regret lower than a specific threshold ϵ, then we return *true*, *false* otherwise. Thus, if the stopping criterion is satisfied, i.e., the function STOPCRITERION($\mathcal{W}_\Lambda$,$\mathcal{S}$) returns *true*, then we stop the algorithm and we recommend the solution with minimum max regret.

4.6 Computational Experiments

The aim of this section is to assess the computational effectiveness of the framework by considering the three different preference elicitation strategies described in Section 4.5.2.3. Two different performance measures are considered: the number of queries generated and the overall computational time required to achieve convergence to the stopping criterion. The number of queries generated is equivalent to the number of interactions with the user, which is an important measure of the framework usability. Contrary to the computational time, such performance measure focuses on measuring the quality of the user preferences strategy adopted, and it does not depend on the approach used to solve the combinatorial problem.

We tested our framework with randomly generated data since we could not use the real data provided by the factory with which we have been collaborating during the development of this framework. The data provided had too many inconsistencies such as unrealistic delivery time and quantities ordered. Also, our random generated experiments allow a detailed comparison of how the algorithm scales with the various parameters, such as the cardinalities of components and suppliers, and the number of components supplied by each supplier.

The computational experiments are performed on randomly generated instances that represent realistic scenarios, as described in Section 4.6.1. With Section 4.6.2 we present the computational results and we discuss how the framework performs under different conditions.

4.6.1 Instances structure

Each instance considered is generated by considering as an input the number of suppliers $|\mathrm{I}|$, the number of components $|\mathrm{C}|$ and the density parameter $\rho \in \mathbb{R}$, where the latter enforces that the total number of pairs (i, j), (where supplier $i \in \mathrm{I}$ can provide component $j \in \mathrm{C}$) is equal to $\rho \cdot |\mathrm{I}| \cdot |\mathrm{C}|$ rounded to the nearest integer. The component availability of each supplier is randomly assigned such that the overall density ρ is enforced, by using the procedure described in Appendix B.1.

The instances are structured in order to reflect a scenario in which the firm needs a large number of low price components and a small number of expensive ones. Bearing this in mind, the set of components C is partitioned into three

categories: *Cheap*, *Average* and *Expensive*, which include, respectively, 75%, 20% and 5% of the overall number of components. The demand D_j of each component $j \in \mathrm{C}$ depends on its category. It is sampled from a Gaussian distribution with mean μ_j^d and standard deviation σ_j^d (discarding values that are less than or equal to zero), using the values reported in Table 4.2.

Table 4.2: Mean and standard deviation of the truncated Gaussian distribution used to sample the demand of a component with respect to each category.

	Cheap	**Average**	**Expensive**
μ_j^d	3000	600	90
σ_j^d	750	150	22.5

The unit cost of each component depends on its category, the supplier, and the quantity ordered. An average cost μ_j^c per component $j \in \mathrm{C}$ is computed by considering a uniform distribution over the interval associated with the component category, as defined in Table 4.3. The unit cost of a component provided by a supplier $i \in \mathrm{I}$ is then sampled with a uniform distribution on the interval $[0.9\mu_j^c, 1.1\mu_j^c]$. Finally, a random discount is considered for computing the costs, by sampling uniformly on the intervals indicated in Table 4.4, which depend on the quantity ordered. The lower limits on the quantities indicated in such table represent the coefficients $m_{i,j,t}$ of Equation 4.7. By following the steps described above, the unit cost parameters $c_{i,j,t}$ are computed.

Table 4.3: Intervals of the uniform distributions used to sample the mean cost of components with respect to each category.

Cheap	**Average**	**Expensive**
[0.05, 3]	[4, 30]	[50, 200]

The activation costs a_i (for $i \in \mathrm{I}$) are defined such that the impact on the overall cost function is of the same order of magnitude as the direct costs. Let $\mu_{j,TOT}^c = \sum_{j \in \mathrm{C}} D_j \cdot \mu_j^c$ be the average total cost to satisfy the whole demand of all components. Assuming that we rely on only $\frac{|\mathrm{I}|}{2}$ suppliers, the average amount of direct costs per supplier is equal to $\frac{2\mu_{j,TOT}^c}{|\mathrm{I}|}$. We sample each activation cost a_i from a uniform distribution on the interval $[0.8\frac{2\mu_{j,TOT}^c}{|\mathrm{I}|}, 1.2\frac{2\mu_{j,TOT}^c}{|\mathrm{I}|})$.

Table 4.4: Discount intervals per component category and quantity ordered.

Cheap		Average		Expensive	
Quantity	**Discount**	**Quantity**	**Discount**	**Quantity**	**Discount**
1-750	0%	1-150	0%	1-20	0%
751 - 1000	[3,7]%	151 - 200	[3,7]%	21 - 30	[3,7]%
1001 - 1250	[8,12]%	201 - 250	[8,12]%	31 - 40	[8,12]%
>1251	[13,17]%	>251	[13,17]%	>41	[13,17]%

The parameters $\lambda_{j,min}$ (Equation 4.10) and $\lambda_{j,max}$ (Equation 4.11) representing the bounds on the number of component per supplier j are sampled by using a discrete uniform distribution on the set of integers $\{1, 2\}$ and on $\{\lambda_{j,min}, \ldots, 5\}$, respectively. The parameters representing the expected lead time $l_{i,j,t}$ and expected delay $\delta_{i,j,t}$ in Equation 4.13 and Equation 4.14 are computed by means of a supplier performance predictor (see Appendix B.3) based on a database of past orders (see Appendix B.2). Finally, the reliability r_i (Equation 4.15) of each supplier i is defined by sampling a discrete uniform distribution on the set $\{1, \ldots, 100\}$.

4.6.2 Experimental results

The framework was implemented in Python 3.7, including the MILP model generation and the different preference elicitation strategies. CPLEX 12.8 [ILO17] was used as a MILP and LP solver, while the Python library pycddlib [Tro18] was used to compute the extreme points of the user-preference polytope. All the experiments described below were performed on an Intel(R) Xeon(R) E5620 2.40 GHz processor with 32 GB of RAM.

The instances considered are randomly generated as described in Section 4.6.1. We generated 20 instances for each triple $(|\mathrm{I}|, |\mathrm{C}|, \rho)$, such that $|\mathrm{I}| \in \{10, 20, 30\}$, $|\mathrm{C}| \in \{30, 40, 50, 60\}$ and $\rho \in \{0.2, 0.3, 0.4, 0.5\}$. As a result, the overall set of instances is formed by $20 \cdot 3 \cdot 4 \cdot 4 = 960$ elements.

Table 4.5, Table 4.6 and Table 4.7 show the performance of the different strategies SMMR, MMD and MDS concerning time and number of queries for experiments with $|\mathrm{I}| = 10$, $|\mathrm{I}| = 20$ and $|\mathrm{I}| = 30$ respectively. The first three columns of both tables contain the values of the parameters $|\mathrm{I}|$, $|\mathrm{C}|$ and ρ, while the fourth column gives the percentage η of instances where the convergence to the stopping criterion was achieved within the time limit of 2 hours. The

Table 4.5: Experimental results with $|\mathrm{I}| = 10$. Bold values represent the best result among the three methods in that row, with respect to time in seconds (for the first set of three columns), or number of queries (for the second set of three columns).

$\|\mathrm{I}\|$	$\|\mathrm{C}\|$	ρ	η	**SMMR** μ_{time}	**MMD** μ_{time}	**MDS** μ_{time}	**SMMR** μ_{query}	**MMD** μ_{query}	**MDS** μ_{query}
10	30	0.2	100.0%	1.06	**0.73**	0.76	**4.15**	4.3	4.3
10	30	0.3	100.0%	6.53	**3.13**	4.3	**5.6**	6.1	6.85
10	30	0.4	100.0%	19.35	**6.57**	10.94	7.1	**6.45**	8.25
10	30	0.5	100.0%	36.29	**16.82**	24.95	**7.85**	8.15	9.2
10	40	0.2	100.0%	1.2	**1.0**	1.01	**4.05**	4.3	4.5
10	40	0.3	100.0%	3.05	**2.39**	2.44	**4.75**	5.15	5.05
10	40	0.4	100.0%	28.2	**15.01**	21.68	7.35	**7.25**	10.65
10	40	0.5	100.0%	36.41	**25.51**	27.83	**8.05**	**8.05**	8.5
10	50	0.2	100.0%	1.44	0.86	**0.82**	**3.65**	3.7	**3.65**
10	50	0.3	100.0%	3.45	2.6	**2.36**	5.0	5.3	**4.9**
10	50	0.4	100.0%	14.62	**10.39**	16.27	**6.55**	7.15	7.8
10	50	0.5	100.0%	74.2	**40.39**	60.56	8.7	**8.15**	9.0
10	60	0.2	100.0%	1.75	1.38	**1.11**	4.1	4.3	**3.95**
10	60	0.3	100.0%	2.16	1.53	**1.48**	**3.65**	3.7	3.75
10	60	0.4	100.0%	39.63	**15.12**	16.58	**6.7**	6.8	7.25
10	60	0.5	100.0%	78.25	**40.92**	45.83	8.45	**8.15**	8.45
Average values				1.779	**1.027**	1.213	**1.017**	1.038	1.116

Table 4.6: Experimental results with $|\mathrm{I}| = 20$. Bold values represent the best result among the three methods in that row, with respect to time in seconds (for the first set of three columns), or number of queries (for the second set of three columns).

$\|\mathrm{I}\|$	$\|\mathrm{C}\|$	ρ	η	**SMMR** μ_{time}	**MMD** μ_{time}	**MDS** μ_{time}	**SMMR** μ_{query}	**MMD** μ_{query}	**MDS** μ_{query}
20	30	0.2	100.0%	16.07	**9.17**	12.07	7.1	**6.95**	7.75
20	30	0.3	100.0%	94.69	**63.48**	78.58	9.85	**9.65**	10.1
20	30	0.4	100.0%	197.66	**99.33**	144.52	9.85	**9.65**	10.7
20	30	0.5	100.0%	476.25	364.75	**276.31**	11.65	**11.45**	13.05
20	40	0.2	100.0%	17.36	15.92	**12.97**	**6.0**	6.95	6.25
20	40	0.3	100.0%	158.36	107.66	**95.94**	10.45	10.45	**10.25**
20	40	0.4	100.0%	241.47	200.07	**195.73**	**9.95**	10.25	10.25
20	40	0.5	100.0%	309.45	**223.2**	226.74	9.75	**9.35**	9.95
20	50	0.2	100.0%	23.04	**12.83**	16.68	**6.65**	6.9	8.0
20	50	0.3	100.0%	174.25	**99.49**	164.75	9.9	**9.1**	10.2
20	50	0.4	100.0%	216.03	194.69	**187.28**	**8.5**	9.05	9.05
20	50	0.5	100.0%	1060.72	**545.77**	729.69	11.05	**10.15**	11.35
20	60	0.2	100.0%	23.51	**20.76**	24.09	**6.9**	7.4	7.2
20	60	0.3	100.0%	217.26	**146.5**	175.08	10.1	**9.55**	11.05
20	60	0.4	95.0%	673.92	**441.33**	509.27	10.0	**9.89**	10.84
20	60	0.5	100.0%	748.75	**332.97**	627.92	10.0	**9.05**	10.75
Average values				1.6	**1.046**	1.232	1.031	**1.024**	1.096

Table 4.7: Experimental results with $|\mathrm{I}| = 30$. Bold values represent the best result among the three methods in that row, with respect to time in seconds (for the first set of three columns), or number of queries (for the second set of three columns).

$\|\mathrm{I}\|$	$\|\mathrm{C}\|$	ρ	η	**SMMR**	**MMD**	**MDS**	**SMMR**	**MMD**	**MDS**
				μ_{time}	μ_{time}	μ_{time}	μ_{query}	μ_{query}	μ_{query}
30	30	0.2	100.0%	179.3	**131.18**	145.89	**10.25**	10.9	11.6
30	30	0.3	100.0%	324.08	**187.3**	239.96	10.8	**10.05**	10.25
30	30	0.4	100.0%	1250.47	760.46	**554.27**	11.75	**10.35**	10.55
30	30	0.5	90.0%	649.68	**396.93**	1092.39	10.28	**9.94**	12.45
30	40	0.2	100.0%	266.6	163.43	**158.97**	**11.0**	11.45	11.4
30	40	0.3	100.0%	558.93	417.02	**343.79**	**10.2**	10.8	10.3
30	40	0.4	90.0%	1457.82	**867.42**	1157.63	11.83	**10.56**	12.44
30	40	0.5	90.0%	2024.71	1075.7	**920.72**	12.23	**11.39**	12.67
30	50	0.2	100.0%	137.72	**88.14**	120.45	8.55	**8.4**	9.45
30	50	0.3	100.0%	580.08	**548.1**	570.5	**8.85**	9.5	10.85
30	50	0.4	100.0%	838.59	**774.24**	887.99	**10.15**	10.5	10.9
30	50	0.5	75.0%	909.26	954.31	**661.15**	**9.66**	9.93	10.14
30	60	0.2	100.0%	492.32	**236.87**	285.56	9.95	**9.7**	9.75
30	60	0.3	90.0%	1022.41	**641.09**	676.74	10.34	10.05	**9.89**
30	60	0.4	80.0%	**298.74**	922.73	780.58	**7.56**	9.88	10.56
30	60	0.5	95.0%	1815.33	**1126.28**	1493.48	10.53	**10.31**	11.16
Average values				1.596	**1.207**	1.327	**1.034**	1.039	1.107

remaining columns show the average μ_{time} of the computational time in seconds and the average μ_{query} of the number of queries, for each of the proposed strategies. The results reported for these last six columns take into account only the instances where convergence was achieved within the time limit.

We need a common measure to compare the rows of Table 4.5, Table 4.6 and Table 4.7, and summarize the performances of the three methods; a simple mean for each column would strongly bias the results towards the larger instances. Instead, for each result (i.e., average time or average number of queries) we compute a score that we call *ratio with the best method* (RWB), dividing the result by the corresponding best result among the three methods in that row; for example, the RWB value for SMMR query time for the first row is equal to $1.06/0.73$. We then consider the mean of these values over all 48 rows; these values are recorded in the last row of the tables.

The 20 instances generated for each triple $(|\mathrm{I}|, |\mathrm{C}|, \rho)$ have a different unknown user preference vector, generated randomly by means of the procedure described below. The first aspect to consider when defining this procedure is the different scales of the four objective functions. For example, a user preference

vector $w_u = (0.25, 0.25, 0.25, 0.25)$ does not necessarily describe a case in which the same importance is given to each of the four objectives since the choice of scales of the objectives can be somewhat arbitrary. Because of the difference in scales, a vector of $(0.25, 0.25, 0.25, 0.25)$ might implicitly be giving much higher importance to, e.g., the first objective. For this reason, we chose not to sample w_u with a uniform distribution (which could lead to, e.g., the first objective being the most important one for almost all the instance), but with a distribution that gives higher probability to more extreme vectors. More precisely, we use the following method:

1. Solve the MILP problem using the extreme points $w1 = (1, 0, 0, 0)$, $w2 = (0, 1, 0, 0)$, $w3 = (0, 0, 1, 0)$ and $w4 = (0, 0, 0, 1)$ of the initial weights vector state space $\mathcal{U}$, and let β_{wi} be the solution computed with weights vector wi where $i \in \{1, \ldots, 4\}$.
2. Compute a value $k_i = \frac{rnd[0,1)}{\beta_{wi}(i) - \min_{j \in \{1,\ldots,4\}}(\beta_{wj}(i))}$ for each $i \in \{1, \ldots, 4\}$.
3. Set the weights vector to $w_p = \frac{1}{\sum_{j=1}^{4} k_j} \cdot (k_1, k_2, k_3, k_4)$

The idea is to try to define an approximation of the range of each objective in order to re-scale a random vector with respect to the ranges of the objective functions.

The bar chart in Figure 4.2 counts the number of times in which each of the three methods for query selection achieved the best average performances given a triple $(|\mathrm{I}|, |\mathrm{C}|, \rho)$ of Table 4.5, Table 4.6 and Table 4.7, and with respect to the number of queries and the total computational time in seconds. More specifically, the frequency in this bar chart is based on giving a score to each strategy. This score is based on summing up 1 unit in the case the strategy is the only method achieving the best performances, a half a unit in the case of a tie between two strategies, and a third of a unit in the case of a three-way tie.

As we can see from Figure 4.2 and the last row of Table 4.5, Table 4.6 and Table 4.7, it looks like that MMD is on average better than the other two methods in terms of total time, better than MDS in terms of number of queries, and (perhaps surprisingly) roughly equivalent to SMMR in terms of number of queries.

In Figure 4.3 and Figure 4.4 is shown the average CPLEX execution time per iteration and the average query computation time per iteration for the three methods for query selection for two different experiment configurations, i.e.,

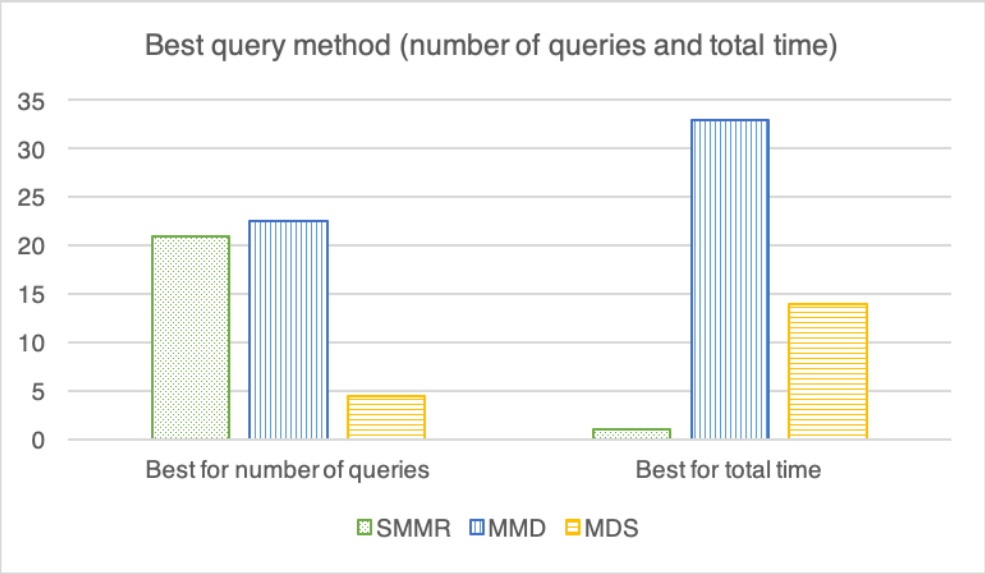


Figure 4.2: Number of experiments in which the three methods for query selection achieved the best performances with respect to number of queries and CPLEX time.

10 suppliers, 20 components and 0.5 density, and 30 suppliers, 50 components and 0.5 density, respectively. The average CPLEX execution time per iteration is computed as the sum of the total CPLEX time for each instance divided by the sum of the number of iterations for each instance. Similarly, the average query computation time per iteration is computed as the sum of the total query time for each repetition divided by the sum of the total number of iterations for each instance. As we can see in Figure 4.3 and Figure 4.4, the query time is much higher for the method SMMR. This is not surprising since SMMR has a worse computational complexity than MMD and MDS. It is interesting to see that the choice of the query selection method has a substantial impact on the total time for small instances (see Figure 4.3). On the contrary, the time taken by the query selection methods is negligible when the size of the instances is large enough (see Figure 4.4).

Generally speaking, the results show that the strategies SMMR and MMD look better than MDS in terms of number of queries generated. A possible explanation is that the discrepancy sum computed in MDS, which drives the query generation process, can be high even if one of the two solutions in the selected pair $(\alpha_{w_1}, \alpha_{w_2})$ has a discrepancy value close to zero. In such scenario, it may happen that the region of the polytope $\mathcal{W}_\Lambda$ in which $w \cdot (X(\alpha_{w_2}) - X(\alpha_{w_1})) \geq 0$ holds is very small. Thus, if the user prefers α_{w_1} to α_{w_2}, the cut induced by the user answer is not highly informative and does not reduce the region

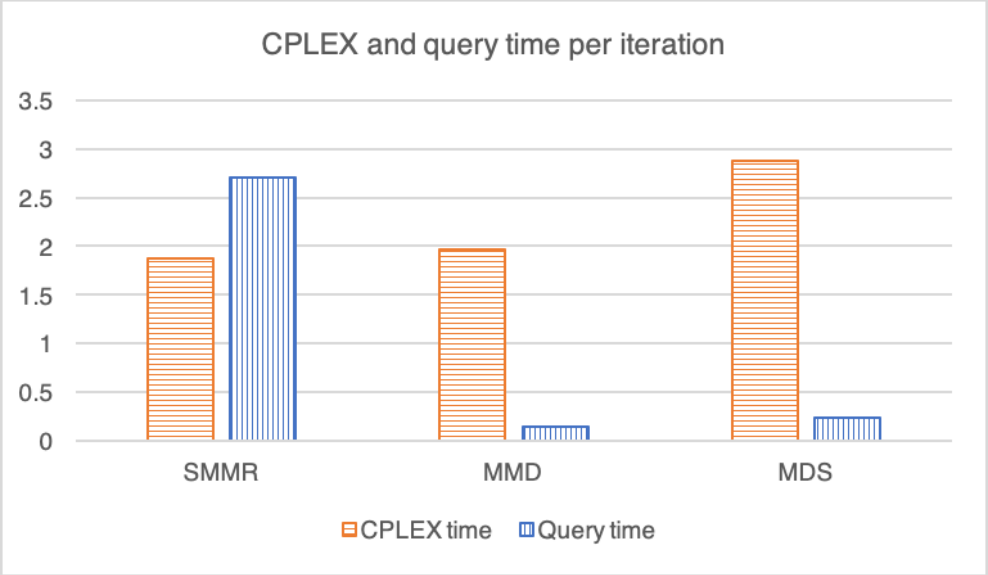


Figure 4.3: Average CPLEX and query computation time in seconds per iteration for the three methods for query selection. The graph shows an average of 20 instances where $|\mathrm{I}| = 10$, $|\mathrm{C}| = 20$ and $\rho = 0.5$.

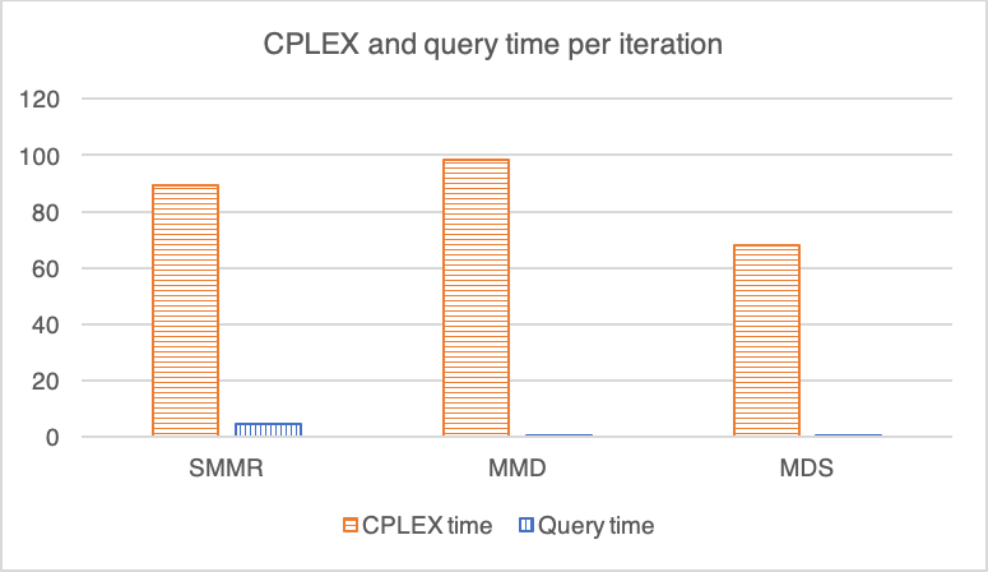


Figure 4.4: Average CPLEX and query computation time in seconds per iteration for the three methods for query selection. The graph shows an average of 20 instances where $|\mathrm{I}| = 30$, $|\mathrm{C}| = 50$ and $\rho = 0.5$.

$\mathcal{W}_\Lambda$ significantly. For this reason, min-max based methods such as SMMR and MMD achieve better performance, since they aim at computing queries that are informative whichever the user answer is.

As we have discussed in Section 2.3.2.2, the SMMR method generates a myopically optimal query with respect to $\mathcal{W}_\Lambda$ if we consider all the optimal solutions associated with $\mathcal{W}_\Lambda$. But at each iteration of our framework, we consider only the solutions associated with the extreme points *Ext*$(\mathcal{W}_\Lambda)$ of $\mathcal{W}_\Lambda$ and then the query computed by SMMR is the most informative only with respect to the user preferences *Ext*$(\mathcal{W}_\Lambda)$. Thus, we cannot guarantee the optimality of the whole sequence of queries since different greedy methods (such as MMD) might generate a different set of extreme points from which we might extract more informative queries.

With MMD we evaluate the minimum worst-case loss of a pair of solutions α_{w_1} and α_{w_2} composing a query only on the corresponding extreme points $w_1, w_2 \in \mathcal{W}_\Lambda$. On the other hand, with SMMR we evaluate the worst-case loss of the query rather than of the single solutions composing the query, and with respect to to the whole set of extreme points *Ext*$(\mathcal{W}_\Lambda)$. Thus, it is interesting to see that even if MMD is a simplification of SMMR, it is an effective method since we can reach the stopping criteria with a similar number of queries.

Finally, the computational results presented clearly show that the framework is very scalable with respect to the number of queries computed to achieve convergence. In fact, such measure grows very slowly as soon as the instance size grows (see Table 4.5, Table 4.6 and Table 4.7). This suggests a practical usability of the framework designed in the context of supplier selection.

4.7 Conclusions

In this chapter we presented a general framework to guide decision-makers via a query generation mechanism in a multi-criteria supplier selection process inspired by a real-world scenario. Our computational results show the performance of three preference elicitation strategies based on the setwise minimax regret criterion, with two of the three being novel. The usability of the framework has been shown by highlighting the moderate number of queries needed to achieve convergence.

Although we tackled a specific problem, our framework can be applied to

generic configuration problems (see, e.g., [BPPS06, BB07, BL19]). In fact, the preference elicitation module is independent of the specific problem that we have to solve. Our MILP model can be replaced by any other configuration problem as long as the objective function is a weighted sum of a fixed number of criteria, and the weights vector represents the user preferences with respect to these criteria. Some examples of domains of application are such as chemical process engineering [RFH20], flow shop scheduling [MIT96], inventory control [TC17] and maintenance planning [ABSK+19]. Our framework can be particularly useful when computing a solution to a configuration problem is time-consuming also with certain preference information. This because we face the preference uncertainty solving the problem for fixed preference models, and thus we do not increase the complexity of the configuration problem to solve. Also, it can be useful when we do not have any preference information since with our fast query generation methods we can compute highly informative queries that can rapidly reduce the uncertainty of the decision-maker's preferences.

Chapter 5

An exact algorithm to compute the Setwise Minimax Regret

In this chapter we define a novel efficient approach to compute the setwise minimax regret, i.e., the minimum setwise max regret, of subsets with a specific cardinality, of an input set of alternatives. We focus on a preference model based on the weighted sum utility function parameterised with respect to the set of weights vectors, and on database problems where alternatives are enumerated and represented with an explicit list of multi-attribute outcomes. The main idea behind our approach is to approximately evaluate the setwise max regret of several subsets simultaneously considering only a discrete subset of the continuous space representing the DM's preferences; then, if the latter evaluation leads to a lower bound of the corresponding setwise max regret above a specific threshold, we compute the setwise max regret with respect to all the DM's preferences consistent with our preference model. The approximated evaluation of the setwise max regret will be performed with a novel approach based on a SAT solver, and the exact computation of the setwise max regret will be performed with algorithms based on concepts presented in Chapter 3.

5.1 Introduction

On the process of incremental preference elicitation based on queries for the DM, minimising the number of interactions with the DM is a key point for a successful preference elicitation system. To evaluate the value of information

of a query for the DM we should consider all future queries and possible responses (see, e.g., [Bou02]), but this may be computationally prohibitive even for a small set of alternatives. As we have discussed in Section 2.3.2.2, a simplification widely used is a myopic evaluation based on the worst-case regret after a single query response. In regret based frameworks for preference elicitation such as [SH01, BPPS06, BB07, BPV14], to minimise the number of interactions with the DM we then need to reduce the worst-case regret at each iteration as much as possible. In [VB09, VB20] the authors have shown that a query defined as a comparison of alternatives and selected using the Setwise Minimax Regret (*SMMR*) criterion is myopically optimal. Furthermore, such a query is also an optimal recommendation set. An important consequence of this result is that there is no exploration-exploitation tradeoff with the Minimax Setwise Regret criterion. On the other hand, this method can be very computationally demanding despite its myopic nature.

Here we focus on database problems, i.e., on problem based on an explicit representation of the alternatives (see Section 2.3.2.2). To the best of our knowledge, the only algorithm in the literature to compute *SMMR* for database problems has been proposed in [VB20] and it is based on the generation of all the possible sets of a specific size k and the corresponding maximum regret. Our method instead relies on search; nodes in the search tree correspond to sets of alternatives with cardinality up to k, and leaves correspond to sets with cardinality exactly k. Pruning is done when we are sure that no extension of the current set can beat the previously found solution; to check this condition we use a SAT solver with cardinality constraints to prune the search space. This idea is combined with our method to compute the setwise max regret based on the extreme points of the epigraph presented in Section 3.7.2. The resulting algorithm is a method that is much faster than the state of the art, as is shown in our experimental tests.

This chapter is organised as follows. In Section 5.2 we state our general assumptions, we recall the main concepts of setwise max regret defined in Section 2.3.2, we describe the utility function considered in this chapter, and provide some basic properties. In Section 5.3 we describe the main ideas behind our algorithm and its main components; while in Section 5.4 we provide a detailed description of the main algorithm to compute the *SMMR*. We provide some experimental results to validate our approach in Section 5.5, and conclude with a final discussion in Section 5.6.

5.2 Setwise Max Regret

We now give some general background and recall some definitions and lemmas of interest for this chapter.

We assume a database problem with a MAUT setting (see 2.3.2.2) where the task is to choose one among a finite and explicit list of alternatives $\mathrm{A} \in \mathcal{M}$. The Decision-Maker (DM) is assumed to be endowed with a utility or utility function u_w, mapping from A to $\mathbb{R}$; w denote the parameters of the utility function (a specific choice of w uniquely determines the utility function). The goal is to pick $\arg\max_{x \in \mathrm{A}} u_w(x)$; however we assume that we (i.e., taking the point of view of a recommender system tasked to support decision-making) do not have access to the DM's true utility function. The problem is to make recommendations under utility function uncertainty; we suppose that our knowledge about the DM's preferences is such that we can identify $\mathcal{U}$ as the set of scenarios representing all the consistent parametrisations w of a DM's utility function u_w.

Let $\mathcal{W} \subseteq \mathcal{U}$ be the set of scenarios representing all the parametrisation w of a DM's utility function u_w consistent with previous input information. We define $\mathrm{Val}_{\mathrm{A}}(w) = \max_{\alpha \in \mathrm{A}} u_w(\alpha)$ to be the utility function of a finite set of alternatives A defined as the maximum utility value that we can get from any alternative $\alpha \in \mathrm{A}$ with respect to the utility function u_w. The setwise max regret (*SMR*) in $\mathcal{W}$ of a finite set of alternatives A with respect to another finite set of alternatives B is defined by:

$$\mathit{SMR}_{\mathcal{W}}(\mathrm{B}, \mathrm{A}) = \max_{w \in \mathcal{W}} (\mathrm{Val}_{\mathrm{A}}(w) - \mathrm{Val}_{\mathrm{B}}(w)) \tag{5.1}$$

The value $\mathit{SMR}_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$ is then the worst-case loss of recommending the best alternative $\beta \in \mathrm{B}$ instead of the best alternative $\alpha \in \mathrm{A}$ supposing that the DM's utility function parameterisation could be any $w \in \mathcal{W}$. The setwise minimax regret (*SMMR*) in $\mathcal{W}$ of all the subsets B of cardinality k of a finite set of alternatives A is defined by:

$$\mathit{SMMR}^{k}_{\mathcal{W}}(\mathrm{A}) = \min_{\mathrm{B} \subseteq \mathrm{A} : |\mathrm{B}| = k} \mathit{SMR}_{\mathcal{W}}(\mathrm{B}, \mathrm{A}) \tag{5.2}$$

The value $\mathit{SMMR}^{k}_{\mathcal{W}}(\mathrm{A})$ is then the minimum setwise max regret we can get from all the possible subsets B of alternatives with cardinality k of A.

Basic properties of setwise regret Recall from Section 2.4.2 that $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}) \subseteq \mathrm{A}$ is the set of undominated elements of A with respect to $\mathcal{W}$. We recall also from Section 2.4.2 the following basic properties of setwise regret that we will use later:

1. (Lemma 3.4.1(i)) $\mathit{SMR}_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$ is monotonically decreasing in B, and monotonically increasing in $\mathcal{W}$, i.e., if $\mathrm{B}' \supseteq \mathrm{B}$ and $\mathcal{W}' \subseteq \mathcal{W}$, then $\mathit{SMR}_{\mathcal{W}'}(\mathrm{B}', \mathrm{A}) \leq \mathit{SMR}_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$.
2. (Lemma 3.4.2(i)) $\mathit{SMR}_{\mathcal{W}}(\mathrm{B}, \mathrm{A}) = \mathit{SMR}_{\mathcal{W}}(\mathrm{UD}_{\mathcal{W}}(\mathrm{B}), \mathrm{UD}_{\mathcal{W}}(\mathrm{A}))$ and, for any $k \geq 1$, $\mathit{SMMR}^k_{\mathcal{W}}(\mathrm{A}) = \mathit{SMMR}^k_{\mathcal{W}}(\mathrm{UD}_{\mathcal{W}}(\mathrm{A}))$.

Utility functions While the previously introduced concepts of this chapter are quite general and apply to any kind utility function, also in this chapter we focus on a preference model based on the weighted sum utility function presented in Section 2.1.4. Here an alternative α is represented with a vector of p reals, with each component corresponding to a criterion, and α_i being the evaluation of α with respect to the i-th criterion. We define $u_w(\alpha) = \alpha \cdot w$ $(= \sum_{i=1}^{p} w_i \alpha_i)$ to be the utility function parametrised with respect to w, where $\alpha \in \mathrm{A} \subset \mathbb{R}^p$, $w \in \mathcal{U}$ and

$$\mathcal{U} = \{w \in \mathbb{R}^p : w_i \geq 0, \sum_{i=1}^{p} w_i = 1\}. \tag{5.3}$$

Our algorithmic approach assumes that $\mathcal{W}$ is a compact convex polytope subset of $\mathcal{U}$. More precisely, we define $\mathcal{W}_\Lambda$ as the set of elements of $\mathcal{U}$ consistent with an input set Λ of DM's input preferences, where a DM's input preference of alternative α over β leading to the constraint $\alpha \cdot w \geq \beta \cdot w$. Thus, $\mathcal{W}_\Lambda$ is the set of elements of $\mathcal{U}$ that satisfy constraints induced by Λ.

5.3 An Efficient Algorithm to Compute Setwise Minimax Regret

The main idea behind our algorithm is to use a depth-first search over subsets of A, with setwise max regret computations at leaf nodes of the search tree, and with a method of pruning branches that reduces the number of setwise max regret computations. More precisely, for a given subset C of A with cardinality less than k, we use a method that determines, for a particular discrete subset $\mathcal{W}'$ of $\mathcal{W}$, if $\mathit{SMR}_{\mathcal{W}'}(\mathrm{B}, \mathrm{A}) \geq \bar{r}$ holds for all supersets B of C with cardinality

k, where $\overline{r}$ is the current upper bound of $SMMR^k_{\mathcal{W}}(\mathrm{A})$. If this holds then, by Lemma 3.4.1(i), $SMR_{\mathcal{W}}(\mathrm{B},\mathrm{A}) \geq \overline{r}$ for all such B, enabling us to backtrack at this point of the search.

In the next paragraph we define how we represent subsets of A; then we define how to evaluate the setwise max regret of a set of subsets of A simultaneously with a Boolean satisfiability (SAT) problem.

Search space: We consider the set of Boolean strings of length at most $n = |\mathrm{A}|$ as a representation of the search space over subsets of A with cardinality less or equal to k. For string x, let $Len(x)$ be the length of x. Let us label A as $\alpha_1, \ldots, \alpha_n$, where $n = |\mathrm{A}|$. We say that a string is *complete* if it is of length n, and otherwise it is *partial*. Each complete string x corresponds to a subset B_x of A, where B_x is the set of all $\alpha_i \in \mathrm{A}$ such that x has a one at its i-th position. We say that complete string x *is of cardinality* k if it contains k *ones*, i.e., if the corresponding subset B_x is of cardinality k. If x and y are Boolean strings then we say that y *extends* x if $Len(y) \geq Len(x)$ and the first $Len(x)$ places of y are the same as those of x. We say that y is a *complete extension of* x if y extends x and y is a complete string. Each partial string x represents a set $\mathcal{Y}_x$ of subsets of A, i.e., all those subsets of cardinality k that correspond to extensions of x. $\mathcal{Y}_x$ is thus the set of all sets B_y for complete extensions y of x of cardinality k. In Section 5.3.3 we define how to generate strings x in turn.

Example 17. *Let $A = \{\alpha_1, \ldots, \alpha_5\}$ be a set of $n = 5$ elements, and let $k = 3$. The complete string $z = 01101$ represents the subset $B_z = \{\alpha_2, \alpha_3, \alpha_5\}$. The partial string $x = 011$ represents the subsets $\mathcal{Y}_x = \{\{\alpha_2, \alpha_3, \alpha_4\}, \{\alpha_2, \alpha_3, \alpha_5\}\}$, where the complete extensions of x are $y = 01101$ and $y' = 01110$.*

5.3.1 Pruning the search space using SAT

Evaluating subsets of A: Given a partial string x, if a set $\mathrm{B}_y \in \mathcal{Y}_x$ is such that $SMR_{\mathcal{W}}(\mathrm{B}_y,\mathrm{A}) < \overline{r}$, then B_y has to contain at least one alternative with worst-case regret lower than $\overline{r}$ for each $w \in \mathcal{W}'$. This concept is formally defined with the following lemma and it will be used to check if there could exists a set in $\mathcal{Y}_x$ improving the current upper bound $\overline{r}$ of $SMMR^k_{\mathcal{W}}(\mathrm{A})$.

Lemma 5.3.1. *Let $\overline{r}$ be an upper bound of $SMMR^k_{\mathcal{W}}(A)$, $\mathcal{W}' \subseteq \mathcal{W}$ and $B \subseteq A$. For $w \in \mathcal{W}$, let Γ_w be the set of $\alpha \in A$ such that $Val_A(w) - u_w(\alpha) < \overline{r}$. Then $SMR_{\mathcal{W}'}(B,A) < \overline{r}$ if and only if for all $w \in \mathcal{W}'$, there exists $\alpha \in B$ such that*

$\alpha \in \Gamma_w$.

Proof. From the definition of setwise max regret it follows that $SMR_{\mathcal{W}'}(\mathrm{B}, \mathrm{A}) < \overline{r}$ if and only if $\mathrm{Val}_{\mathrm{A}}(w) - \mathrm{Val}_{\mathrm{B}}(w) < \overline{r}$ for all $w \in \mathcal{W}'$, which is if and only if for all $w \in \mathcal{W}'$ there exists $\alpha \in \mathrm{B}$ such that $\mathrm{Val}_{\mathrm{A}}(w) - u_w(\alpha) < \overline{r}$, which is if and only if $\alpha \in \Gamma_w$ since $\mathrm{B} \subseteq \mathrm{A}$. □

To check if there exists a set $\mathrm{B}_y \in \mathcal{Y}_x$ such that $SMR_{\mathcal{W}'}(\mathrm{B}_y, \mathrm{A}) < \overline{r}$, we define a SAT problem with cardinality constraint c (see, e.g., [Sin05]), where the cardinality constraint is used to define the size k of the sets in $\mathcal{Y}_x$.

Example 18. *Consider the following SAT formula:* $X = (X_1 \vee X_2) \wedge (X_1 \vee X_3)$ *with cardinality constraint* $c = 1$*, where* X_i *are* $\{0, 1\}$*-valued variable.* X_i *with* $i = \{1, 2, 3\}$ *are literals, and* $(X_1 \vee X_2)$ *and* $(X_1 \vee X_3)$ *are clauses. The cardinality constraint* $c = 1$ *means* $\sum_{i=1}^{3} X_i = 1$*. In this example,* X *is satisfiable since if* $X_1 = 1$ *then* $X = 1$*. But if for example we add the constraint* $X_1 = 0$*, then* X *is unsatisfiable since for any valid assignment of the cardinality constraint, i.e.,* $(X_1 = 0, X_2 = 1, X_3 = 0)$ *or* $(X_1 = 0, X_2 = 0, X_3 = 1)$*, we get* $X = 0$*.*

In our SAT problem, we use a $\{0, 1\}$-valued variable X_i for each $\alpha_i \in \mathrm{A}$. These are used to reason about the unknown sets B_y in $\mathcal{Y}_x$, which we want to be such that $SMR_{\mathcal{W}'}(\mathrm{B}_y, \mathrm{A}) < \overline{r}$. Then $X_i = 1$ means that $\mathrm{B}_y \ni \alpha_i$. Given a partial string x, we then define the corresponding SAT problem with cardinality constraint as follows:

(1) The cardinality constraint $|\mathrm{B}_y| = k$ is expressed as $\sum_{\alpha_i \in \mathrm{A}} X_i = k$.

(2) The constraint that y extends x is expressed as: for all $i \in \{1, \ldots, \mathit{Len}(x)\}$,
   - if $x(i) = 1$ then $X_i = 1$ (where $x(i)$ is the i-th value of x);
   - if $x(i) = 0$ then $X_i = 0$.

(3) For each $w \in \mathcal{W}'$ we define a clause $\bigvee_{\alpha_i \in \Gamma_w} X_i$, where Γ_w is the set of $\alpha \in \mathrm{A}$ such that $\mathrm{Val}_{\mathrm{A}}(w) - u_w(\alpha) < \overline{r}$.

This SAT problem is satisfiable if and only if there exists $\mathrm{B}_y \in \mathcal{Y}_x$ such that for all $w \in \mathcal{W}'$ there exists $\alpha \in \mathrm{B}_y$ such that $\alpha \in \Gamma_w$, which is (by Lemma 5.3.1) if and only if there exists $\mathrm{B}_y \in \mathcal{Y}_x$ such that $SMR_{\mathcal{W}'}(\mathrm{B}_y, \mathrm{A}) < \overline{r}$. Therefore, if the SAT problem is unsatisfiable, then for each $\mathrm{B}_y \in \mathcal{Y}_x$, $SMR_{\mathcal{W}'}(\mathrm{B}_y, \mathrm{A}) \geq \overline{r}$, and thus (by Lemma 3.4.1(i)) $SMR_{\mathcal{W}}(\mathrm{B}_y, \mathrm{A}) \geq \overline{r}$. This means that there is then no need to explore any string y extending x, so we can then backtrack from the

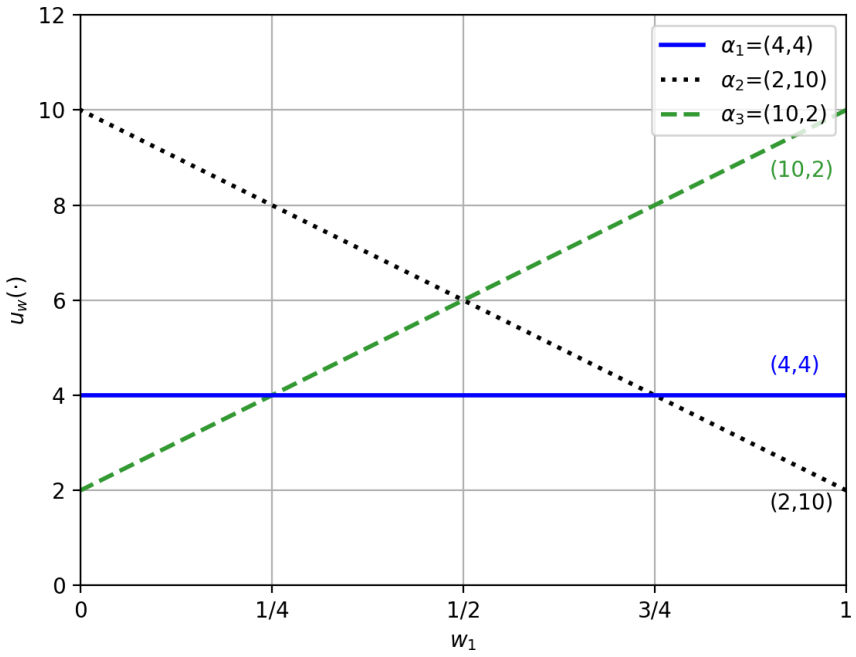


Figure 5.1: Plot of the linear utility functions $u_w(\cdot)$ for the alternatives $\alpha_1 = (4,4)$ (blue solid), $\alpha_2 = (2,10)$ (black dotted) and $\alpha_3 = (10,2)$ (green dashed) with $w \in \mathcal{W} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$.

current search node associated with x, saving us from computing $SMR_{\mathcal{W}}(\mathrm{B}_y, \mathrm{A})$ for $\mathrm{B}_y \in \mathcal{Y}_x$.

Example 19. *Consider the set of alternatives* $A = \{\alpha_1 = (4,4), \alpha_2 = (2,10), \alpha_3 = (10,2)\}$ *whose utility function* $u_w(\cdot)$ *with* $w \in \mathcal{W} = \{w \in \mathbb{R}^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$ *is shown in Figure 5.1. Let* $k = 2$, $\mathcal{W}' = \{(0,1),(0.5),(1,0)\}$, $\overline{r} = 1$, *and let* x *be the string* 1. *Thus,* $\Gamma_{(0,1)} = \{\alpha_2\}$, $\Gamma_{(0.5,0.5)} = \{\alpha_2, \alpha_3\}$, $\Gamma_{(1,0)} = \{\alpha_3\}$ *and* $\mathcal{Y}_x = \{\{\alpha_1, \alpha_2\}, \{\alpha_1, \alpha_3\}\}$ *since the complete extensions of* x *with cardinality* $k = 2$ *are* $y =$ "110" *and* $y' =$ "101". *The corresponding SAT problem is then* $X_2 \wedge (X_2 \vee X_3) \wedge X_3$ *with cardinality constraints* $c = 2$ *and* $X_1 = 1$. *It is easy to see that in this case the SAT problem is unsatisfiable; therefore we can avoid the computation of* $SMR_{\mathcal{W}}(\{\alpha_1, \alpha_2\}, A)$ *and* $SMR_{\mathcal{W}}(\{\alpha_1, \alpha_3\}, A)$. *In fact, the subset B of A cardinality* 2 *that minimises* $SMR_{\mathcal{W}}(B, A)$ *is* $B = \{\alpha_2, \alpha_3\}$.

5.3.2 Computation of setwise max regret

With linear utility functions $u_w(\cdot)$, a standard method to compute the setwise max regret $SMR_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$ of a set $\mathrm{B} \in \mathrm{A}$ consists of the evaluation of a linear programming (LP) problem for each $\alpha_i \in \mathrm{A}$ (see Section 2.3.2.3). Briefly, for

$\alpha_i \in$ A we have $SMR_{\mathcal{W}}(\mathrm{B}, \{\alpha_i\}) = \max_{w \in \mathcal{W}}(\alpha_i \cdot w - \mathrm{Val}_{\mathrm{B}}(w))$, which means that we can compute $SMR_{\mathcal{W}}(\mathrm{B}, \{\alpha_i\})$ as the maximum value δ_i subject to the constraints $w \in \mathcal{W}$, and $(\alpha_i - \beta) \cdot w \geq \delta_i$ for each $\beta \in$ B. We can then compute $SMR_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$ as $\max_{\alpha_i \in \mathrm{A}} SMR_{\mathcal{W}}(\mathrm{B}, \{\alpha\})$.

However, here we also make use of the method based on the extreme points of the epigraph of the utility function (EP) presented in Section 3.6; let $\gamma(\mathcal{W}, \mathrm{B}) = \{(w, r) : w \in \mathcal{W}, \quad \beta \cdot w \leq r \quad \forall \beta \in \mathrm{B}\}$ be the *epigraph* of the utility function $\mathrm{Val}_{\mathrm{B}}$ on $\mathcal{W}$. Briefly, from Lemma 3.4.1 and Theorem 3.6.6 follows that we can $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$ compute as $\max_{\beta \in \mathrm{B}} \max \{u_w(\beta) - r : (w, r) \in Ext(\gamma(\mathcal{W}, \mathrm{A}))\}$.

Our purpose is to find the subset B with minimum $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$. Thus, since $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B}) \geq SMR_{\mathcal{W}'}(\mathrm{A}, \mathrm{B})$, before the computation of $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$ we can first check the value of $SMR_{\mathcal{W}'}(\mathrm{A}, \mathrm{B})$ which is a much faster operation given that $\mathcal{W}'$ is a discrete set. In fact, we can compute $SMR_{\mathcal{W}'}(\mathrm{A}, \mathrm{B})$ apllying directly the definition of setwise max regret (Equation 5.1). We then compute $SMR_{\mathcal{W}}(\mathrm{A}, \mathrm{B})$ only if $SMR_{\mathcal{W}'}(\mathrm{A}, \mathrm{B}) < \overline{r}$, otherwise we already know that B cannot improve the current upper bound $\overline{r}$.

5.3.3 Generating subsets of A using depth-first search

We generate strings x representing subsets of A sequentially using a depth-first search with backtracking on a binary tree T, and with a fixed value and variable ordering (though the variable ordering depends on the value k: see Section 5.3.4). Note that we are not interested in all the possible binary strings of length n. Instead, we want to generate complete strings x with k ones, and the corresponding sub-strings, since these will represent subsets B of A with $|\mathrm{B}| \leq k$. The order in which we reach complete strings (and their associated subsets) is based on the obvious lexicographic order, i.e., ascending numerical order if the strings are viewed as binary numbers. We then define T as follows: the root represents the empty string; internal nodes represent strings of length less than n; and leaves represent strings of length n with k ones. The out-edges of an internal node pointing to the corresponding left and right children have values 0 and 1, respectively. Thus, if an internal node represents the string x, then the left child represents the string $x0$ and the right child represents the string $x1$.

We generate strings sequentially starting from the left most leaf node representing the subset $(\alpha_{n-k+1}, \ldots, \alpha_n)$. Given a generic string x_j, we define two

methods to generate the next string x_{j+1}, namely, the *backtracking case* and the *non-backtracking case*.

Backtracking case: Let *NextBT*$(x_j, n, k) = x_{j+1}$ corresponds to the backtracking case of the j-th string. With *NextBT*(x_j, n, k) we move from the current node representing x_j toward the root until we find an edge e with value zero. Let v be the parent of e. We define *NextBT*(x, n, k) as the string represented by the right child of v. We will use this method to generate the string x_{j+1} when x_j is a complete string, or when x_j is a partial string but $SMR_{\mathcal{W}}(\mathrm{B}, \mathrm{A}) \geq \overline{r}$ for all $\mathrm{B} \in \mathcal{Y}_{x_j}$. Roughly speaking, we use *NextBT*(x_j, n, k) when we want to evaluate a new set of subsets since $\mathcal{Y}_{x_j} \cap \mathcal{Y}_{x_{j+1}} = \emptyset$.

Non-backtracking case: Let *Next*$(x_j, n, k) = x_{j+1}$ be the non-backtracking case of the j-th string. With *Next*(x_j, n, k) we compute the next string following the depth-first search logic. We will use this method to generate the string x_{j+1} for the cases not covered by the backtracking case, i.e., when x_j is not a complete string and we can't ensure that $SMR_{\mathcal{W}}(\mathrm{B}, \mathrm{A}) \geq \overline{r}$ for all $\mathrm{B} \in \mathcal{Y}_{x_j}$. Roughly speaking, we use *Next*(x_j, n, k) to reduce the sets to evaluate, in fact, $\mathcal{Y}_{x_{j+1}} \subset \mathcal{Y}_{x_j}$.

In both cases, when we visit the root, and the corresponding out-edges have already been visited, we stop the search. Note that if $\mathcal{Y}_{x_{j+1}}$ is a singleton set with x_{j+1} not being a complete string, then we can speed up the computation by jumping to the leaf node corresponding to the unique set in $\mathcal{Y}_{x_{j+1}}$. This can happen when x_{j+1} can be extended only with ones or only with zeros in order to satisfy the constraint that a complete string x must have k ones.

5.3.4 Further implementation details

Generating $\mathcal{W}'$: We start with $\mathcal{W}' = \emptyset$. Then for each *SMR* computation of a subset B of A, if $SMR_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$ is greater than the current upper bound $\overline{r}$ of $SMMR^k_{\mathcal{W}}(\mathrm{A})$, we update $\mathcal{W}'$. Depending on which method we use for the computation of *SMR* (see Section 5.3.2)., we use one of the following method to update $\mathcal{W}'$:

1. Epigraph of the value function: $\mathcal{W}' = \mathcal{W}' \cup UE(\mathrm{B})$ where $UE(\mathrm{B})$ is the projection of $Ext(\gamma(\mathcal{W}, \mathrm{B}))$ to $\mathcal{W}$.
2. Linear programming: $\mathcal{W}' = \mathcal{W}' \cup_{i \in \{1,\dots,n\}} w_i$ where $w_i \in \mathcal{W}$ is the

preference model in which $SMR_{\mathcal{W}}(\mathrm{B}, \{\alpha_i\})$ is maximised.

One could update $\mathcal{W}'$ using only the point $w \in \mathcal{W}$ in which $SMR_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$ is maximised; however, collecting more points in $\mathcal{W}'$ adds more clauses to the SAT problem, and thus increases the possibility of unsatisfiability, leading to pruning of the search tree.

Incremental updating of SAT instances: For a given $\mathcal{W}'$, when the SAT problem associated with a string x is solvable, we can use the corresponding instantiation to define the SAT problem associated to a string y extending x. In fact, the SAT problem corresponding to y will be the same as that associated with x but with the additional constraints $X_i = y_i$ for all $i \in \{Len(x) + 1, \ldots, Len(y)\}$. This is particularly useful when y is a substring of the solution X found for the SAT problem for x, since in this case X is a solution also to the SAT problem associated with y, and thus we do not need to call the SAT solver. For example, suppose $n = 5$, $k = 3$ and $x = 01$, and suppose that the solution of the SAT problem associated with x is $X = 01110$. Then, if $y = 011$ and $\mathcal{W}'$ has not changed, we don't need to define a new SAT problem from scratch since the SAT problem associated with y is the same as that associated with x but with the additional constraint $X_3 = 1$. Furthermore, in this case y is also a substring of X, thus we do not need to call the SAT solver since X is a solution also for the SAT problem associated with y.

5.4 Pseudocode

In this section we combine the concepts presented in Section 5.3, defining the whole procedure for the computation of $SMMR^k_{\mathcal{W}}(\mathrm{A})$. The inputs of our algorithm are:

1. A finite set A of alternatives α_i where each alternative is represented as a p-dimensional vector of reals.
2. The DM's preference state space $\mathcal{W}$ representing the possible parametrisations w of the utility function $u_w(\cdot)$ expressed as a compact subset of $\{w \in I\!R^p : w_i \geq 0, \sum_{i=1}^{p} w_i = 1\}$.
3. An integer $k \leq |\mathrm{A}|$ representing the cardinality of the subsets of A that we want to evaluate.

We start with $\mathcal{W}' = \emptyset$, with x equal to $n - k$ zeros followed by n ones, and with

$\overline{r} = \infty$. Then we proceed as follows:

1) If x is the empty string then we stop the algorithm and return $\overline{r}$.

2) If *Len*$(x) = n$ (i.e., x is a complete string) and *SMR*$_{\mathcal{W}'}(\mathrm{B}_x, \mathrm{A}) < \overline{r}$ then we compute *SMR*$_{\mathcal{W}}(\mathrm{B}_x, \mathrm{A})$, where B_x is the set represented by x, also generating the set *UE*(B_x) (that is the projection on $\mathcal{W}$ of the set of extreme points of the epigraph $\gamma(\mathcal{W}, \mathrm{B}_x)$). If *SMR*$_{\mathcal{W}}(\mathrm{B}_x, \mathrm{A}), \overline{r}$:

   a) we update the upper bound $\overline{r}$ by $\overline{r} = \min(\overline{r}, \textit{SMR}_{\mathcal{W}}(\mathrm{B}_x, \mathrm{A}))$;

   b) we update $\mathcal{W}' = \mathcal{W}' \cup \textit{Ext}(\mathcal{W}_{\mathrm{B}_x})$;

   c) we generate a new string with the backtracking case *NextBT*(x, n, k).

3) Otherwise, *Len*$(x) < n$ (i.e., x is a partial string). We call Boolean function SAT$(x, k, \mathrm{A}, \mathcal{W}', \overline{r})$, which returns TRUE if and only if the associated SAT problem (see Section 5.3.1) is satisfiable, i.e., there exists $\mathrm{B}_y \in \mathcal{Y}_x$ such that *SMR*$_{\mathcal{W}'}(\mathrm{B}_y, \mathrm{A}) < \overline{r}$.

   a) If the SAT problem is satisfiable, we move to the next string with the non-backtracking case *Next*(x, n, k);

   b) If the SAT problem is not satisfiable, we move to the next string with the backtracking case *NextBT*(x, n, k).

When we have gone through all of the space of strings, i.e., when x is the empty string, the value of $\overline{r}$ will equal *SMMR*$^k_{\mathcal{W}}(\mathrm{A})$, i.e., the minimum value of *SMR*$_{\mathcal{W}}(\mathrm{B}, \mathrm{A})$ over all subsets B of A of cardinality k.

In Algorithm 14 we show the recursive procedure to compute *SMMR*$^k_{\mathcal{W}}(\mathrm{A})$.

Example 20. *Consider the set of alternatives* $A = \{\alpha_1 = (4,4), \alpha_2 = (2,8), \alpha_3 = (6,6), \alpha_4 = (8,2)\}$ *whose utility function* $u_w(\cdot)$ *with* $w \in \mathcal{W} = \{w \in I\!R^2 : w_i \geq 0, \sum_{i=1}^2 w_i = 1\}$ *is shown in Figure 5.1. Let* $k = 2$, $\mathcal{W}' = \emptyset$, $\overline{r} = \infty$. *We start with the string* $x =$ "0011" *representing the set* $B_{0011} = \{\alpha_3, \alpha_4\}$, *and we compute* *SMR*$_{\mathcal{W}}(B_{0011}, A)$ *which equals 2 and it is maximised in* $w = (0,1)$. *Since* *SMR*$_{\mathcal{W}}(B_{0011}, A) < \overline{r}$, *we update* $\overline{r} = 2$, *and* $\mathcal{W}' = \{(0,1), (\frac{2}{3}, \frac{1}{3}), (1,0)\}$. *We then set* $x =$ *NextBT*$(\text{"0011"}, 4, 2) =$ "01" *and we call the SAT solver. In this case we get* $\Gamma_{(0,1)} = \{\alpha_2\}$, $\Gamma_{(\frac{2}{3},\frac{1}{3})} = \{\alpha_3, \alpha_4\}$, $\Gamma_{(1,0)} = \{\alpha_4\}$. *The corresponding SAT formula is then* $X_2 \wedge (X_3 \vee X_4) \wedge X_4$ *with cardinality constraints* $c = 2$ *and* $X_2 = 1$. *With* $X_4 = 1$ *the SAT formula is satisfied, thus we generate the next string with the non-backtracking case* *Next*$(\text{"01"}, 4, 2) =$ "0101". *Since* *SMR*$_{\mathcal{W}'}(B_{0101}, A) = 0 < \overline{r}$, *we compute* *SMR*$_{\mathcal{W}}(B_{0101}, A)$ *which equals 1 and*

Algorithm 14 EPI SAT

procedure SMMR$(k, \mathrm{A}, \mathcal{W})$
  $\mathcal{W}' \leftarrow \emptyset$
  $\overline{r} \leftarrow \infty$
  $x \leftarrow$ string of $n-k$ zeros concatenated with k ones
  do
    if $Len(x) = n$ **then**
      if $\mathcal{W}' = \emptyset$ or $SMR_{\mathcal{W}'}(\mathrm{B}_x, \mathrm{A}) < \overline{r}$ **then**
        $SMR \leftarrow SMR_{\mathcal{W}}(\mathrm{B}_x, \mathrm{A})$
        if $SMR < \overline{r}$ **then**
          $\overline{r} \leftarrow SMR$
          $\mathcal{W}' \leftarrow \mathcal{W}' \cup UE(\mathrm{B}_x)$
      $x \leftarrow NextBT(x, n, k)$
    else if SAT$(x, k, \mathrm{A}, \mathcal{W}', \overline{r})$ **then**
      $x \leftarrow Next(x, n, k)$
    else
      $x \leftarrow NextBT(x, n, k)$
  while $x \neq$ empty string
  return $\overline{r}$

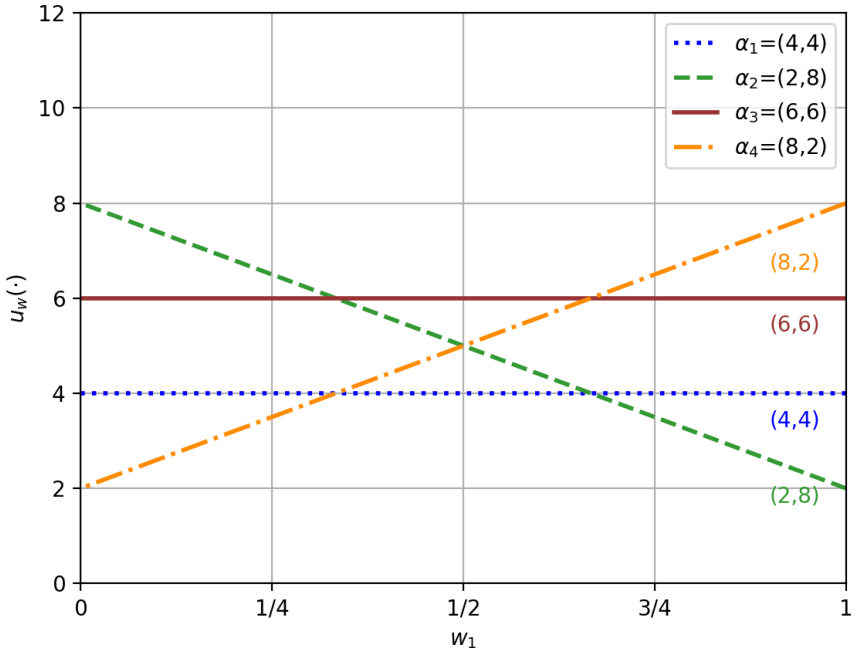


Figure 5.2: Plot of the linear utility functions $u_w(\cdot)$ for the alternatives $\alpha_1 = (4,4)$ (blue dotted), $\alpha_2 = (2,8)$ (green dashed), $\alpha_3 = (6,6)$ (red solid) and $\alpha_3 = (8,2)$ (yellow dashed-dotted) with $w \in \mathcal{W} = \{w \in I\!R^2 : w_i \geq 0, \sum_{i=1}^{2} w_i = 1\}$.

it is maximised in $(\frac{1}{2}, \frac{1}{2})$. *Since* $SMR_{\mathcal{W}}(B_{0011}, A) < \overline{r}$, *we update* $\overline{r} = 1$, *and* $\mathcal{W}' = \{(0,1), (\frac{1}{2}, \frac{1}{2}), (\frac{2}{3}, \frac{1}{3}), (1,0)\}$. *We then call NextBT*$(\text{"}0101\text{"}, 4, 2)$ *which returns* $\text{"}0110\text{"}$ *since* $\mathcal{Y}_{011} = \{\alpha_2, \alpha_3\}$ *is a singleton. We do not compute the exact setwise regret of the set* $B = \{\alpha_2, \alpha_3\}$ *since* $SMR_{\mathcal{W}'}(B_{0110}, A) = 2 < \overline{r} = 1$, *and we move then to the next string NextBT*$(\text{"}0110\text{"}, 4, 2) = 1$. *The corresponding SAT formula is* $X_2 \wedge X_3 \wedge (X_3 \vee X_4) \wedge X_4$ *with cardinality constraints* $c = 2$ *and* $X_1 = 1$, *which is not satisfiable. NextBT*$(\text{"}1\text{"}, 4, 2) = \text{""}$ *then the algorithm return the current upper bound* $\overline{r} = 1$. *Thus, the subset of A of cardniality 2 with minimum setwise regret is* $B = \{\alpha_2, \alpha_4\}$. *Note that* α_1 *is dominated by* α_3, *thus we could have discarded it filtering out the dominated alternatives before executing the algorithm.*

The most expensive operations in our algorithm are enumerating the extreme points of the epigraph on the evaluation of $SMR_{\mathcal{W}}(\mathrm{B}_x, \mathrm{A})$, and the SAT problem used to evaluate if there exists $\mathrm{B}_y \in \mathcal{Y}_x$ such that $SMR_{\mathcal{W}'}(\mathrm{B}_y, \mathrm{A}) < \overline{r}$. As we have discussed in Section 3.7, the computational complexity of enumerating the extreme points of the epigraph of the value function of a set B of k alternatives is $O(p^k)$. Regarding the SAT problem, the number of clauses equals the size of $|\mathcal{W}'|$ at the corresponding iteration, and the number of literals for each clause is $O(|\mathrm{A}|)$.

5.5 Experimental Results

In our experimental results we used CPLEX 12.8 [ILO17] as the linear programming solver, and we used the Python library pycddlib [Tro18] for computing the extreme points of the epigraph of the value function. As the SAT solver, we used Minicard implemented in the Python library Pysat [Sto19] which has a native method to set a cardinality constraint. All experiments were performed on a computer facilitated by two Intel Xeon E5620 2.40GHz processors and 32 GB RAM.

From Lemma 3.4.2(i) it follows that $SMMR^k_{\mathcal{W}}(\mathrm{A}) = SMMR^k_{\mathcal{W}}(\mathrm{UD}_{\mathcal{W}}(\mathrm{A}))$, where $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ represents the set of undominated alternatives of A in $\mathcal{W}$. In our experiments we have noticed that filtering out dominated alternatives can be a very worthwhile preliminary step. For example, generating 10 random sets A with $|\mathrm{A}| = 25000$ and $p = 4$, we got an average of $|\mathrm{UD}_{\mathcal{W}}(\mathrm{A})| = 220$ alternatives computed in roughly 10 seconds. See Appendix A.1 for details about our random problem generator.

In Table 5.1 we show the average computation time of $SMMR^k_{\mathcal{W}}(\mathrm{A})$ over 20

Table 5.1: Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT, LP SAT, EPI BF and LP BF over 20 repetitions varying k and p with an input set of 50 undominated alternatives and $\mathcal{W} = \mathcal{U}$.

k	p	Time[s] EPI SAT	Time[s] LP SAT	Time[s] EPI BF	Time[s] LP BF
2	3	0.17	8.5	4.15	413.71
2	4	0.19	6.69	5.88	424.90
3	3	0.43	17.46	76.43	6739.02
3	4	0.63	17.72	115.97	6922.82

repetitions with $k \in \{2,3\}$, $p \in \{3,4\}$, and an input set of 50 random undominated alternatives. Time SAT EPI and Time SAT LP indicate the average time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ using the SAT solver, where we compute $SMR_{\mathcal{W}}(\mathrm{B},\mathrm{A})$ using the epigraph of the value function, and a linear programming solver, respectively. Time BF EPI and Time BF LP indicate the average time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ using a straightforward algorithm evaluating all the subsets of size k, where also in this case we compute $SMR_{\mathcal{W}}(\mathrm{B},\mathrm{A})$ using the epigraph of the value function, and a linear programming solver, respectively. Thus, the results on the first column relate to our best algorithm, and the results on the last column relate to a straightforward algorithm based on the definition of $SMMR^k_{\mathcal{W}}(\mathrm{A})$, and as we can see, with our algorithm we get a very significant improvement. Also, comparing EPI SAT with LP SAT, and EPI BF with LP BF, we can see that the computation of the setwise max regret using the epigraph of the value function seems to improve the performance with respect to the linear programming method.

In Table 5.2 and Table 5.3 we show the average timing of our algorithm EPI SAT with $k = 2$ and $p = 4$ varying the number of user preferences and the size of the undominated input sets, respectively. The SMMR with $k = 2$ is of our particular interest since the corresponding set with minimum setwise max regret is a myopically optimal binary query [VB20]. Binary queries have been often used in preference elicitation systems (see, e.g., [ILC01b, Abb04, GW05, BB07, BB10, GS10b, BL19]), and our algorithm may be used as a query selection strategy in these contexts when the set of alternatives is not too large. In Table 5.2, Λ represents the set of (consistent) user preferences (corresponding to linear constraints on the user preference space $\mathcal{U}$), each being of the form $\alpha \cdot w \geq \beta \cdot w$. This constraint can be interpreted as a preference of alternative α to alternative β. Each set of constraints Λ therefore defines a subset $\mathcal{W}_\Lambda$ of $\mathcal{U}$, which is in fact a compact convex polytope. The sets of constraints Λ used in our experiments are generated supposing a

Table 5.2: Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT over 20 repetitions varying the number of user preferences Λ with $|\mathrm{UD}_{\mathcal{W}}(\mathrm{A})| = 500$, $\mathcal{W} = \mathcal{U}$, $k = 2$ and $p = 4$. $|\mathcal{W}'|$ represents the average number of user preference models used to evaluate subsets of A with SAT, and UD represents the algorithm to filter out dominated alternatives.

$\lvert\mathrm{UD}_{\mathcal{W}_\Lambda}(\mathrm{A})\rvert$	$\lvert\Lambda\rvert$	Time[s] EPI SAT	$\lvert\mathcal{W}'\rvert$	Time[s] UD
500	0	16.18	68.4	1.467
267.7	1	5.86	61.8	0.460
180.9	2	2.25	56.8	0.154
140.45	3	1.36	59	0.067
100.55	4	0.76	47.7	0.036
65.15	5	0.49	45.5	0.021
90.6	6	0.88	46.6	0.016
34.1	7	0.26	42.8	0.013
40.8	8	0.43	37.2	0.008
24.45	9	0.24	34.4	0.006

random user preference model w, and simulating an iterative elicitation process with binary queries. At each iteration, we simulate the user preference with respect to a myopically optimal binary query $Q = \{\alpha, \beta\}$ computed with our algorithm, and we use the simulated user preference model w to define the sign of the inequality associated with Q. The resulting constraint will then be added to Λ. As we can see in Table 5.2, setting for example $k = 2$ and $p = 4$ with $|\mathrm{UD}_{\mathcal{W}}(\mathrm{A})| = 500$ alternatives, the number of undominated alternatives rapidly decreases when increasing the number of constraints, with a consequent improving of the computation time of $SMMR^k_{\mathcal{W}_\Lambda}(\mathrm{A})$. In fact, as we can see in Table 5.3, the time performance of our main algorithm seems to grow exponentially with respect to the size of the input set $\mathrm{UD}_{\mathcal{W}_\Lambda}(\mathrm{A})$ of undominated alternatives. In Table 5.2 we reported also the computation time to filter out dominated alternatives, and, as we can see, it is a very fast operation in comparison with the computation of *SMMR*.

In Figure 5.3 we show how EPI SAT scales with respect to k and p. The y-axis represents the average timing of our algorithm with a logarithmic scale. The x-axis represents the number of criteria $p \in \{2, \ldots, 6\}$. Each line represents the average time performance of 20 repetitions with an input set A of 100 undominated alternatives and varying $k \in \{2, \ldots, 6\}$. As we can see, the computation times increases exponentially with respect to k, reflecting the exponential growth of the number of subsets of A of cardinality k. The computation time seems to grow exponentially also with respect to p, and this

Table 5.3: Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT over 20 repetitions varying the size of the input set $\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$ of undominated alternatives with $|\Lambda| = 0$ (i.e., $\mathcal{W} = \mathcal{U}$), $k = 2$, $p = 4$. $|\mathcal{W}'|$ represents the average number of user preference models used to evaluate subsets of A with SAT.

$\lvert\mathrm{UD}_{\mathcal{W}}(\mathrm{A})\rvert$	Time[s] EPI SAT	$\lvert\mathcal{W}'\rvert$
100	0.37	46.9
200	1.45	54.3
300	5.29	58.6
400	9.69	62.2
500	28.27	56.9
600	35.95	72.5

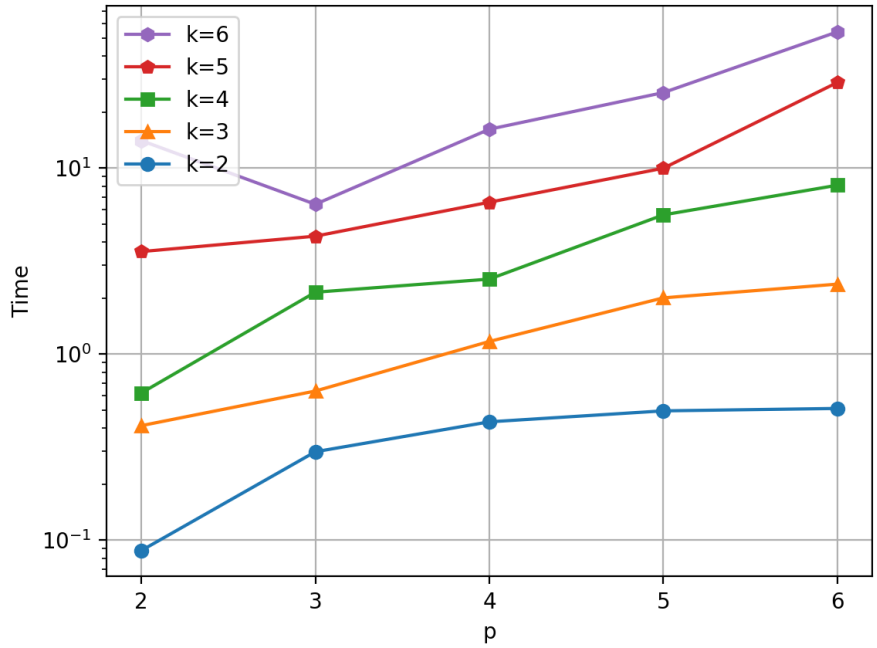


Figure 5.3: Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT (y-axis) over 20 repetitions varying k and p with an input set of 100 undominated alternatives and $\mathcal{W} = \mathcal{U}$.

may be due to the exponential growth of the number of extreme points of $\mathcal{W}$ with respect to p (cf. Table 5.4 for $k = 4$).

We also tested our algorithm with the databases used in the experimental results of [VB20]:

1. **Synthetic**: A synthetic database of 81 alternatives evaluated with $p = 12$ criteria with binary domain $\{0, 1\}$.

Table 5.4: Average computation time in seconds to compute $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with EPI SAT over 20 repetitions varying p with $|\mathrm{UD}_{\mathcal{W}}(\mathrm{A})| = 100$, $|\Lambda| = 0$ (so $\mathcal{W} = \mathcal{U}$) and $k = 4$. $|\mathcal{W}'|$ represents the average number of user preference models used to evaluate subsets of A with SAT.

p	Time[s] EPI SAT	$\lvert\mathcal{W}'\rvert$
2	0.62	20.7
3	2.15	77.6
4	2.53	152.2
5	5.61	301.9
6	8.09	507.5

Table 5.5: Computation of $SMMR^k_{\mathcal{W}}(\mathrm{A})$ with the databases considered in the experimental results of [VB20]. The first four columns show information regarding the input databases. The fifth and the sixth columns show the performances of filtering out dominated elements. The last two columns show the time performance our algorithm EPI SAT and the method used in [VB20] whose results are shown in Table 8.

A	k	p	|A|	$\lvert\mathrm{UD}_{\mathcal{W}}(\mathrm{A})\rvert$	$\mathrm{UD}_{\mathcal{W}}(\mathrm{A})$[s]	EPI SAT[s]	[VB20][s]
Synthetic	2	12	81	81	0.22	0.09	19.47
Synthetic	3	12	81	81	0.22	0.93	-
Synthetic	4	12	81	81	0.22	5.86	-
Synthetic	5	12	81	81	0.22	17.72	-
Rental	2	23	187	100	0.57	0.19	0.17
Rental	3	23	187	100	0.57	0.28	-
Rental	4	23	187	100	057	4.01	-
Rental	5	23	187	100	0.57	172.32	-
Boston	2	14	506	475	3.29	60.73	277.53
Boston	3	14	506	475	3.29	182.18	-

2. **Rental**: Real university student rental database with 187 alternatives evaluated with $p = 23$ criteria with four of them being real values in the interval $[0, 1]$ and the remaining being binary values $\{0, 1\}$.

3. **Boston**: Boston housing database [Bel82] with 506 alternatives evaluated with $p = 14$ criteria with one of them being a binary value $\{0, 1\}$ and the remaining being real values. In this case, the alternatives are preprocessed to lie within a common scale.

In Table 5.5 we compare the performance of our algorithm EPI SAT with respect to the results reported in Table 8 of [VB20] related to the three datasets described above. The precise algorithm used in the experimental results of [VB20] to compute *SMMR* is not described, and the authors show

the performance only for $k = 2$, with the computation of *SMMR* for $k > 2$ not being feasible with their algorithm. Regarding our algorithm EPI SAT, the computation of *SMMR* with $k = 4$ was unfeasible for the Boston database. However, we were able to compute it for $k \in \{2, 3\}$. With the datasets Synthetic and Rental we were able to compute *SMMR* even with $k = 5$. Furthermore, with $k = 2$, for example, our approach performed better than the heuristic approximate methods used in [VB20]. Only be able to manage up to $k = 3$ for the Boston database may be because of the large number of both alternatives and criteria. In fact, from Table 5.3 and Figure 5.3 it looks like that the time performance of our algorithm increases exponentially with respect to these parameters.

5.6 Conclusions

Interactive elicitation methods maintain a model of the user preferences which is revised incrementally by recording the answers to questions asked to the decision-maker. In particular, several works [WB03, BPPS06, Bra12, BPV14, BPV17b] have used (standard, single-item) minimax regret to provide a robust recommendation to decision-maker.

The notion of setwise regret [VB09, VB20] allows one to provide a sound and principled approach for generating, based on the current uncertainty about the decision-maker's utility function, a set of alternatives 1) to used as a recommendation set, and 2) to be used as a choice query to drive the elicitation forward. Thus it is a valuable method for generating queries and recommendation sets to help a user find a most-preferred item in the database. However, the high computational burden of this approach has limited its adoption in applications. In this chapter we have addressed this issue, providing an efficient algorithm to compute the setwise minimax regret for database problems making use of a SAT solver to prune the search.

We validated our approach in numerical experiments that showed very substantial improvement with respect to the state of the art. In particular, our algorithm may replace heuristic approaches adopted in real-time preference elicitation system with binary queries, allowing the computation of optimal query sets. The computation of a query set with more than two alternatives may instead be too slow for a real-time context since the complexity burden increases exponentially with respect to k. However, our method unlocked the possibil-

ity of computing optimal sets for some problems in which such approach was previously considered unfeasible.

Chapter 6

Conclusions & Future Work

In this chapter, we discuss the main results of this thesis, and we describe some possible future works.

6.1 Summary

Chapter 2 is a summary of the background material of this thesis and some related works. We presented the main concepts and properties of MAUT models. MAUT defines the theoretical basis for preference models represented by the weighted sum utility function defined in Section 2.1.4. This preference model has been the focus of Chapter 4 and Chapter 5, and it has been used for examples and algorithms of Chapter 3. We also discussed the connection of MAUT models with classic recommender systems, and some related works on preference elicitation. We presented parameterised preference models introducing imprecisely specified multi-attribute utility theory which is one of the first attempts to deal with parameterised utility functions. We also described in details the minimax criterion and its setwise generalisation, which has been of particular interest for all the chapters of this thesis. We concluded the chapter describing preference relations, optimality classes and operators for alternatives evaluated with parameterised value functions.

In Chapter 3 we generalised dominance and equivalence relations introduced in Chapter 2 to evaluate sets of alternatives with respect to a set of preference models $\mathcal{W}$. We defined the connections of such preference relations with some optimality classes, namely, undominated alternatives $\mathrm{UD}_{\mathcal{W}}$, possibly optimal alternatives $\mathrm{PO}_{\mathcal{W}}$ and possibly strictly optimal alternatives $\mathrm{PSO}_{\mathcal{W}}$.

An important contribution of this chapter is the definition of the concept of setwise minimal equivalent subset of a set of alternatives A, and its relation with the set of possibly optimal alternatives. In particular, with Theorem 3.3.6 we have shown that $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is the unique setwise minimal equivalent subset for A if and only if $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is equivalent to A, and with Theorem 3.3.15 we have shown that for analytic utility functions $\mathrm{PSO}_{\mathcal{W}}(\mathrm{A})$ is the unique setwise minimal equivalent subset for A. We have also shown that alternatives composing a choice query in a preference elicitation process should be possibly strictly optimal. Otherwise, there is the risk of modelling the DM's preference with respect to events with zero probability (see Proposition 4 and discussion of Section 3.5). Theorem 3.6.6 and Proposition 5 are other important contributions of this chapter. These relate to algorithms based on the epigraph of the utility function for the computation of the setwise max regret and the setwise minimal equivalent subset, and for testing dominance of sets of alternatives when supposing the weighted sum utility function as preference model. The pseudo-code for the novel algorithm to compute the setwise max regret and to test the dominance of sets of alternatives is shown in Algorithm 12. The pseudo-code for the novel algorithm to compute the set of possibly strictly optimal alternatives is shown in Algorithm 7. These methods outperform standard methods based on linear programming when up to seven criteria used to evaluate alternatives.

In Chapter 4 we presented a preference elicitation framework for a multi-criteria supplier selection problem inspired by a real-world scenario. We supposed a preference model based on the weighted sum utility function with the criteria evaluating the alternatives being cost, lateness, lead time and reputation. Although we tackled a specific problem, our framework can be applied to generic configuration problems based on a linear preference model. This work lies between two research areas. On the one hand, it provides an alternative perspective on the solution of supplier selection problems. On the other hand, it presents an interactive preference elicitation approach with novel query selection strategies. An advantage of our approach with respect to other supplier selection frameworks preference-based is that the latter involve the elicitation of a potentially large number of numerical values. These can be very time-consuming and difficult to assess. In contrast, our approach involves intuitive comparison queries and attempts to limit the number of queries asked to the user. Also, to the best of our knowledge, there are no works describing an iterative preference-based multi-criteria supplier selection problem to satisfy

the demand of a set of products. Regarding the query selection strategies, our framework makes use of novel methods based on a discrepancy measure which is strictly related to the max regret (see Equation 4.22 and Proposition 6). To evaluate our approach, we compared our query selection strategies with myopically optimal queries generated with the setwise minimax regret criterion. Our experimental results show that our query selection strategies have roughly the same performances as myopically optimal queries in terms of number of interactions to achieve convergence, but have significantly better performance in terms of computational time.

Chapter 5 is devoted to the presentation of a novel algorithm to compute the setwise minimax regret; also in this case we supposed a preference model based on the weighted sum value function. Our algorithm is based on a SAT solver which evaluates the setwise max regret of different sets of alternatives of a specific cardinality simultaneously. This evaluation is made with respect to a discrete set of parameters representing different DM's preference models, and the purpose is to exclude several sets simultaneously without computing the setwise max regret for each of these sets. Lemma 5.3.1 states the fundamental property on which our algorithm is based. We presented two variations of the algorithm, which differ with respect to the method used for the computation of the setwise max regret. The first is based on the epigraph of the value function presented in Section 3.6; the second is based on a standard linear programming approach presented in Section 2.4. Given our experimental results shown in Table 3.3, the latter may be more suitable for problems in which alternatives are evaluated with more than six criteria. This may well be due to the exponential growth of the number of extreme points of the epigraph of the utility function with respect to the number of criteria. Our experimental results of Section 5.5 show the efficiency of our method for the computation of the setwise minimax regret, which seems to be largely outperforming the current state of the art independently of the method used to compute the setwise max regret. For example, the computation of the setwise minimax regret of size $k = 3$ with $p = 4$ criteria used to evaluate the alternatives outperforms the current state of the art for more than two orders of magnitude in terms of computational time.

6.2 Possible Future Works

In Chapter 3 we focus on the weighted sum value function for the computation of the minimal equivalent subset. However, it would be natural to to develop

computational procedures to compute the minimal equivalent subset for more general linear cases, such as GAI networks, OWA and Choquet integral, based on our more general characterisation results, such as Theorems 3.3.15 and 3.6.6 and Lemma 3.4.1. Alternatively, one could evaluate the feasibility of our algorithms relaxing the constraints on the structure of the DM's utility function as in [GSFM10]. Our methods for the computation of the minimal equivalent subset could be directly applied to reduce the set of utility vectors derived for a multi-objective influence diagram [MRW12] or a multi-objective optimisation problem [MRW13]. In the latter cases, the set of alternatives is derived from a combinatorial structure. Thus one could also consider to further develop the computational techniques that make use of such combinatorial structures. A further natural application of our model and methods is for computing the Value of Information [DJ97] for a multi-objective influence diagram. Each observable variable generates a Value of Information function which is a utility function $\mathrm{Val}_{\mathrm{A}}$, so different observable variables can be compared using the relation $\succcurlyeq^{\mathcal{W}}_{\forall\forall\exists}$.

Regarding the preference elicitation framework described in Chapter 4, future studies may be conducted in order to adapt the framework to the case where the combinatorial problem is solved by means of some heuristic algorithm with no optimality guarantee. This should speed up the computation of the solutions used to generate the queries, and thus it could make the framework suitable for larger problems. Also, it would be interesting to compare the performance in terms of execution time of the proposed query selection strategies with respect to the novel algorithm to compute the setwise minimax regret presented in Chapter 5, or with respect to the query selection strategies proposed in [BLL20].

Future works related to the novel algorithm presented in Chapter 5 to compute the setwise minimax regret could involve testing the performances of our algorithm computing an initial upper bound of the setwise minimax regret with a heuristic such as those in [VB20]. This could well speed up the initial iterations of the algorithm since an upper bound close to the minimax regret value has the potential to reduce the number of exact computations of setwise max regret. Also, our implementation gives a proof of concept, using an algorithm of quite a simple structure. However, it can probably be speeded up a lot using, for example, parallel evaluation of different branches whilst keeping track of a common upper bound. It would also be interesting to explore the feasibility of a constraint programming approach replacing the call to the SAT solver; this may enable propagation of literals to reduce the search space further. Also, it could be interesting evaluating the computational complexity

of computing the setwise minimax regret. Another direction for future work, also related to Chapter Chapter 3, is to consider the definition of a procedure to compute a subset that is not equivalent to the input set A but has a worst-case loss that is below a given threshold. This can be useful since a small regret can be acceptable if it helps significantly reduce the size of the input set. For example, one could perform a search over the cardinality parameter k starting from the size of a minimal equivalent subset M of A and decreasing this value until we get $\mathit{SMMR}^k_{\mathcal{W}}(M) > \epsilon$. In this case the algorithm based on linear programming may be faster since the computational complexity of enumerating the extreme points is $O(p^k)$ and starting with $k = |M|$ we may well have high values of k. Alternatively, one could also start with $k = 1$ and increase this value until we get $\mathit{SMMR}^k_{\mathcal{W}}(M) < \epsilon$. Note that the number of subsets of cardinality k is $\binom{|M|}{k}$ and then the most complex problem to solve is with $k = \frac{|M|}{2}$. Thus it may be worth it to use first an heuristic such as those presented in [VB20] to compute an upper bound of the size k^* we are looking for. This could help on deciding if starting with $k = 1$ or $k = |M|$.

Appendix A

A.1 Random Problem Generator

This appendix includes a description of the random problem generator that is used for the experiments in Section 5.5 and Section 3.9.

Let $j \in \{1, \ldots, J\}$ be the index of each set of alternatives A_j that we are going to generate. Consider now any j. For each criterion $i \in \{1, \ldots, p\}$ we pick random parameters $\mu_j(i)$ and $\delta_j(i)$, and each of the n elements α of A_j is picked independently as follows: for each criterion i, choose value $\alpha(i)$ uniformly in range $[\mu_j(i) - \delta_j(i), \mu_j(i) + \delta_j(i)]$. Choosing random parameters $\mu_j(i)$ and $\delta_j(i)$ for A_j: first we randomly pick μ_j uniformly in range $[-\mu, \mu]$, and θ_j is chosen uniformly in range $[0, 2\theta]$, and δ_j is chosen uniformly in range $[0, 2\delta]$. Then, for each criterion i, $\mu_j(i)$ is chosen uniformly in range $[\mu_j - \theta_j, \mu_j + \theta_j]$ and $\delta_j(i)$ is chosen uniformly in range $[0, 2\delta_j]$. The pseudocode is shown in Algorithm 15; the generated sets will contain only undominated elements if the Boolean input u is True.

Note that if θ and δ are both very small, then each θ_j and δ_j will be very small, so each $\mu_j(i)$ will be very close to μ_j. Generating two sets A and B (i.e. $J = 2$), this would lead to a case in which we will tend to almost always get that A $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ B or B $\succcurlyeq^{\mathcal{W}}_{\exists\forall\forall}$ A (since the elements of e.g., A will be very similar to each other). This will also tend (to a somewhat lesser extent) to be the case if just θ is very small. On the other hand, if θ and δ are relatively large, then we will tend to get less dominance. We tried different values of the input parameters, obtaining the lowest rate of dominance ($\sim 96\%$) with $\mu = 10$, $\theta = 50$ and $\delta = 60$.

In our random problem generator we also randomly generate T user preferences of the form $aw_i + bw_j \geq cw_k$ meaning that the user prefers a units of w_i and b units of w_j to c units of w_k. We ensure the consistency of such constraints

by first randomly picking a normalised vector w, and only including constraints that are consistent with this vector (e.g., if a constraint randomly generated that is not consistent with this, then we change the sign of the constraint to make it consistent).

Algorithm 15 Random instance generator

procedure $RandomProblem(n, p, J, T, \mu, \theta, \delta, u)$
  Input: $n \in \mathbb{N}$: sets cardinality, $p \in \mathbb{N}$: number of criterion, $J \in \mathbb{N}$: number of sets, $J \geq 2$, $T \in \mathbb{N}$: number of user preferences, $\theta \in \mathbb{R}$, $\delta \in \mathbb{R}$, $\mu \in \mathbb{R}$, UD: Boolean value
  $\mathcal{W} \leftarrow \{w \in \mathbb{R}^p : \sum_{i=1}^{n} w[i] = 1, w[i] \geq 0\}$
  $j \leftarrow 0$
  while $j < J$ **do**
    $\mu_j \leftarrow rndReal[-\mu, \mu]$
    $\theta_j \leftarrow rndReal[0, 2\theta]$
    $\delta_j \leftarrow rndReal[0, 2\delta]$
    $i \leftarrow 0$
    while $i < p$ **do**
      $\mu_{ji} \leftarrow rndReal[\mu_j - \theta_j, \mu_j + \theta_j]$
      $\delta_{ji} \leftarrow rndReal[0, 2\delta_j]$
      $i \leftarrow i + 1$
      $\mathrm{A}_j \leftarrow \emptyset$
      while $|\mathrm{A}_j| < n$ **do**
        $\alpha \leftarrow$ vector in $\mathbb{R}^p$ with $\alpha[i] = rndReal[\mu_{ji} - \delta_{ji}, \mu_{ji} + \delta_{ji}]$
        if $\neg u$ or (u and $\mathrm{UD}_{\mathcal{W}}(\mathrm{A}_j \cup \alpha) = \mathrm{A}_j \cup \alpha$) **then**
          $\mathrm{A}_j \leftarrow \mathrm{A}_j \cup \alpha$
      $j \leftarrow j + 1$
  $t \leftarrow 0$
  $\omega \leftarrow$ random vector $w \in \mathcal{W}$
  while $t < T$ **do**
    $a, b, c \leftarrow$ three $rndReal[0, 1)$
    $i, j, k \leftarrow$ three distinct $rndInteger[0, d - 1]$
    if $a\omega[i] + b\omega[j] - c\omega[k] \geq 0$ **then**
      $\mathcal{W} \leftarrow \mathcal{W} \cap \{w \in \mathbb{R}^p : aw[i] + bw[j] - cw[k] \geq 0\}$
    else
      $\mathcal{W} \leftarrow \mathcal{W} \cap \{w \in \mathbb{R}^p : aw[i] + bw[j] - cw[k] \leq 0\}$
  $t \leftarrow t + 1$
return A, B, $\mathcal{W}$

Appendix B

B.1 Random Catalogue Generation

This appendix describes how to generate a suppliers' catalogue for the experiments in Section 4.6. The algorithm assigns to each supplier a certain set of components, such that an overall density ρ is enforced, a minimum number of components $\lambda_{j,min} = 2$ is provided by each supplier, and each component is provided by one supplier at least.

The suppliers' catalogue is represented by a $|\mathrm{I}| \times |\mathrm{C}|$ matrix Ψ where each element (i, j) is equal to 1 if supplier i can provide component j, 0 otherwise. As previously indicated in Section 4.5.1, C_i is the set of components supplied by supplier i. Similarly, let I_j be the set of suppliers providing the component j. The following is the procedure used to randomly generate the matrix Ψ:

1. Set each element (i, j) of Ψ to 0
2. For each supplier i in I choose a random component j in C, add j to C_i, add i to I_j and set the (i, j)-th element of Ψ to 1
3. For each component j in C such that $|\mathrm{I}_j| = 0$ choose two different random suppliers i and i', add j to C_i and $\mathrm{C}_{i'}$, add $\{i, i'\}$ to I_j, set the (i, j)-th and the (i', j)-th elements of Ψ to 1
4. For each component j in C such that $|\mathrm{I}_j| = 1$, let $\mathrm{I}_j = \{i'\}$, choose random supplier $i \neq i'$, add j to C_i, add i to I_j, and set the (i, j)-th element of Ψ to 1
5. Let K be the value $\rho \cdot |\mathrm{C}| \cdot |\mathrm{I}|$ rounded to the nearest integer
6. Let Δ be the number of elements of Ψ equal to 1
7. Let $k = K - \Delta$

8. While $k > 0$, pick a random $i \in \mathrm{I}$ and $j \in \mathrm{C}$. If the (i, j)-th component of Ψ is equal to 0, then set such element to 1 and decrease k by 1 unit.

B.2 Random Database Generator

This appendix describes how to compute a random database of past orders for the experiments in Section 4.6. This is used in the framework to simulate the possibility of predicting the lead time $l_{i,j,t}$ and lateness $\delta_{i,j,t}$ parameters of the MILP model by means of real data, as a function of the triple supplier i, component j and tariff t. We assume that each entry of the database is a random order o_k represented by a tuple $\langle i(o_k), j(o_k), q(o_k), l(o_k), \delta(o_k)\rangle$, meaning that supplier $i(o_k)$ received an order of quantity $q(o_k)$ of component $j(o_k)$, and provided the components with lead time $l(o_k)$ and lateness $\delta(o_k)$.

The number of orders generated for each component $j \in \mathrm{C}$ supplied by supplier $i \in \mathrm{I}$ is sampled from a discrete uniform distribution on the set $\{5, \ldots, 15\}$. The quantity of each order o_k related to a component $j(o_k)$ is the nearest integer of a value sampled from a Gaussian distribution (where negative and null values are discarded) whose parameters μ_q and σ_q are shown in Table B.1 and depend on the category of $j(o_k)$.

Table B.1: Gaussian distribution parameters to sample the quantity of a component for an order with respect to component categories.

	Cheap	**Average**	**Expensive**
μ_q	1000	200	30
σ_q	250	50	7.5

Five different values RD_i, $RV1_i$, $RV2_i$, $RV3_i$ and $RV4_i$ are assigned to each supplier in order to model its ability to deliver in time and compute the delay and lateness of its orders. These values are computed as follows:

- RD_i is sampled by using a uniform distribution on the interval $[0, 1)$
- $RV1_i$ and $RV2_i$ are sampled by using a discrete uniform distribution on the set $\{10, \ldots, 30\}$
- $RV3_i$ and $RV4_i$ are sampled by using a discrete uniform distribution on the set $\{1, \ldots, 10\}$

The lead time $l(o_k)$ of an order o_k assigned to a supplier $i(o_k)$ and of a quantity $q = q(o_k)$ is then computed by sampling a value from each of the following

distributions:

- a discrete uniform distribution on the set $\{2, \ldots, 20\}$
- a Gamma distribution with mean $RV1_i \cdot max(\log_{10}(10 \cdot q), 1)$ and standard deviation $\sigma_{l(o_k)} = RV2_i \cdot max(\log_{10}(10 \cdot q), 1)$

and summing them.

The lateness $\delta(o_k)$ of a random order o_k supplied by supplier $i(o_k)$ and of a quantity $q = q(o_k)$ is 0 if a random number sampled between 0 and 1 is less than RD_i. This models the case where the order is not late. Otherwise, $\delta(o_k)$ is computed by a sample of a Gamma distribution with mean $\mu_{\delta(o_k)} = RV3_i \cdot max(\log_{10}(10 \cdot q), 1)$ and standard deviation $\sigma_{\delta(o_k)} = RV4_i \cdot max(\log_{10}(10 \cdot q), 1)$. Please note that the term $max(\log_{10}(10 \cdot q), 1)$ is used in the computation of both $l(o_k)$ and $\delta(o_k)$ in order to increase mean and standard deviation for orders with high quantities.

B.3 Lead-time and Lateness Predictor

This appendix describes a predictor for the experiments in Section 4.6 to compute expected lead time $l_{i,j,t}$ (Equation 4.13) and expected delay $\delta_{i,j,t}$ (Equation 4.14) of a triple supplier i, component j and quantity interval t given a database of past orders. Let us first suppose that we have an *objective order* $o_0 = \langle i(o_0), j(o_0), q(o_0), l(o_0), \delta(o_0) \rangle$ where $i(o_0)$, $j(o_0)$ and $q(o_0)$ are known and we want to estimate $l(o_0)$ and $\delta(o_0)$. As in Appendix B.2, we indicate with $o_k = \langle i(o_k), j(o_k), q(o_k), l(o_k), \delta(o_k) \rangle$ a past order, i.e., the k-th order of a database. The idea is computing $l(o_0)$ and $\delta(o_0)$ as a weighted average of lead times and delays of past orders, where each weight depends on the similarity of the corresponding past order o_k with o_0.

Two types of similarity measures are considered:

- similarity between the quantities $Sim_q(q(o_k), q(o_0)) = \frac{\min(q(o_k), q(o_0))}{\max(q(o_k), q(o_0))}$
- similarity between the components $Sim_j(j(o_k), j(o_0))$, defined to be 1 if $j(o_k) = j(o_0)$, defined to be 0.5 if the category of $j(o_k)$ and $j(o_0)$ is the same, and $Sim_j(j(o_k), j(o_0)) = 0.1$, otherwise.

These two similarity measures are used to compute two *sub-weights* for each past order o_k:

$$w_q^k = \frac{Sim_q(q(o_k), q(o_0))}{\sum_{p=1}^{n} Sim_q(q(o_p), q(o_0))} \text{ and } w_j^k = \frac{Sim_j(j(o_k), j(o_0))}{\sum_{p=1}^{n} Sim_j(j(o_p), j(o_0))},$$

where $p \in [1, n]$ are the indexes of all the past orders stored in the database. After computing w_q^k and w_j^k for each past order o_k, the lead time $l(o_0)$ and the delay $\delta(o_0)$ of the objective order o_0 are estimated as $l(o_0) = \sum_{k=1}^{n}(0.5w_q^k + 0.5w_j^k)l(o_k)$ and $\delta(o_0) = \sum_{k=1}^{n}(0.6w_q^k + 0.4w_j^k)\delta(o_k)$, where $l(o_k)$ and $\delta(o_k)$ are delay and lead time of the past order o_k. The weight of w_q^k in the formula used to compute $\delta(o_0)$ is set to 0.6 in order to give slightly more importance to past orders with similar quantities rather than past orders with similar components.

Note that $l_{i,j,t}$ and $\delta_{i,j,t}$ represent an expectation of lead time and delay given a specific quantity interval, while the method described computes an estimated lead time and delay given a specific quantity. We manage this issue by estimating lead time and delay of two objective orders o_0' and o_0'', where the quantities $q(o_0')$ and $q(o_0'')$ are lower and upper bound of the range of quantities defining the quantity interval t (see Table 4.4 in Sect. 4.6.1). The values of $l_{i,j,t}$ and $\delta_{i,j,t}$ are then computed by averaging the values predicted for o_0' and o_0'' as follows: $l_{i,j,t} = (l(o_0') + l(o_0''))/2$ and $\delta_{i,j,t} = (\delta(o_0') + \delta(o_0''))/2$. Since the upper bound of the last quantity intervals in Table 4.4 are not defined, we consider these values to be 1500, 300 and 50 for category cheap, average and expensive, respectively.

References

[A+16] Charu C Aggarwal et al. *Recommender systems*, volume 1. Springer, 2016.

[Abb04] Ali Abbas. Entropy methods for adaptive utility elicitation. *IEEE Transactions on Systems, Science and Cybernetics*, 34(2):169–178, 2004.

[Abd13] Mohammad Abdolshah. A review of quality criteria supporting supplier selection. *Journal of Quality and Reliability Engineering*, 06 2013.

[ABSK+19] Zaharah Allah Bukhsh, Irina Stipanovic, Giel Klanker, Alan O'Connor, and Andre G Doree. Network level bridges maintenance planning using multi-attribute utility theory. *Structure and infrastructure engineering*, 15(7):872–885, 2019.

[ACAL+16] Pedro Amorim, Eduardo Curcio, Bernardo Almada-Lobo, Ana P.F.D. Barbosa-Póvoa, and Ignacio E. Grossmann. Supplier selection in the processed food industry under uncertainty. *European Journal of Operational Research*, 252(3):801 – 814, 2016.

[AHH07] Najla Aissaoui, Mohamed Haouari, and Elkafi Hassini. Supplier selection and order lot sizing modeling: A review. *Computers & Operations Research*, 34(12):3516 – 3540, 2007. Operations Research and Outsourcing.

[AMD19] Christina Arampantzi, Ioannis Minis, and Georgios Dikas. A strategic model for exact supply chain network design and its application to a global manufacturer. *International Journal of Production Research*, 57(5):1371–1397, 2019.

[AP97] Antreas D Athanassopoulos and Victor V Podinovski. Dominance and potential optimality in multiple criteria decision analysis

with imprecise information. *Journal of the Operational research Society*, 48(2):142–150, 1997.

[APW+18] Adi Wicaksono Purnawan, Pujawan I Nyoman, Widodo Erwin, Sutrisno, and Izzatunnisa Laila. Mixed integer linear programming model for dynamic supplier selection problem considering discounts. *MATEC Web Conf.*, 154:01071, 2018.

[AV15] Hamza Adeinat and José A. Ventura. Determining the retailer's replenishment policy considering multiple capacitated suppliers and price-sensitive demand. *European Journal of Operational Research*, 247(1):83 – 92, 2015.

[AV18] Hamza Adeinat and Jose A. Ventura. Integrated pricing and supplier selection in a two-stage supply chain. *International Journal of Production Economics*, 201:193 – 202, 2018.

[BB06] Darius Braziunas and Craig Boutilier. Preference elicitation and generalized additive utility. In *Proceedings of AAAI*, volume 21, 2006.

[BB07] Darius Braziunas and Craig Boutilier. Minimax regret based elicitation of generalized additive utilities. In *Proceedings of UAI*, pages 25–32, 2007.

[BB10] Darius Braziunas and Craig Boutilier. Assessing regret-based preference elicitation with the utpref recommendation system. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 219–228, 2010.

[BB12] Darius Braziunas and Craig Boutilier. Local utility elicitation in gai models. *arXiv preprint arXiv:1207.1361*, 2012.

[BBB13] Craig Boutilier, Fahiem Bacchus, and Ronen I Brafman. UCP-networks: A directed graphical representation of conditional utilities. *arXiv preprint arXiv:1301.2259*, 2013.

[BC19] Silvia Bacci and Bruno Chiandotto. *Introduction to Statistical Decision Theory: Utility Theory and Causal Analysis*. CRC Press, 2019.

[BDDPV16] Nawal Benabbou, Serena Di Sabatino Di Diodoro, Patrice Perny, and Paolo Viappiani. Incremental preference elicitation in multi-

attribute domains for choice and ranking with the borda count. In *Proceedings of International Conference on Scalable Uncertainty Management*, pages 81–95. Springer, 2016.

[Bel82] David E Bell. Regret in decision making under uncertainty. *Operations research*, 30(5):961–981, 1982.

[Ben17] Nawal Benabbou. *Procédures de décision par élicitation incrémentale de préférences en optimisation multicritère, multi-agents et dans l'incertain*. PhD thesis, 2017.

[BG95] Fahiem Bacchus and Adam Grove. Graphical models for preference and utility. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 3–10, 1995.

[BG13] Fahiem Bacchus and Adam J Grove. Graphical models for preference and utility. *arXiv preprint arXiv:1302.4928*, 2013.

[BG15] Tim Baarslag and Enrico H Gerding. Optimal incremental preference elicitation during negotiation. In *Proceedings of Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[BL19] Nawal Benabbou and Thibaut Lust. An interactive polyhedral approach for multi-objective combinatorial optimization with incomplete preference information. In *Proceedings of International Conference on Scalable Uncertainty Management, (SUM 2019) Compiègne, France, December 16-18*, pages 221–235. Springer, 2019.

[BLL20] Nawal Benabbou, Cassandre Leroy, and Thibaut Lust. An interactive regret-based genetic algorithm for solving multi-objective combinatorial optimization problems. In *Proceedings of Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20)*, 2020.

[Bly02] Jim Blythe. Visual exploration and incremental utility elicitation. In *Proceedings of AAAI/IAAI*, pages 526–532, 2002.

[Bou02] Craig Boutilier. A POMDP formulation of preference elicitation problems. In *Proceedings of AAAI/IAAI*, pages 239–246, 2002.

[BP15a] N. Benabbou and P. Perny. On possibly optimal tradeoffs in multicriteria spanning tree problems. In *Proceedings of ADT 2015*, volume 9346 of *Lecture Notes in Computer Science*, pages 322–337. Springer, 2015.

[BP15b] Nawal Benabbou and Patrice Perny. Incremental weight elicitation for multiobjective state space search. In *Proceedings of Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[BP16] Nawal Benabbou and Patrice Perny. Solving multi-agent knapsack problems using incremental approval voting. In *Proceedings of the Twenty-second European Conference on Artificial Intelligence*, pages 1318–1326. IOS Press, 2016.

[BP17] N. Benabbou and P. Perny. Adaptive elicitation of preferences under uncertainty in sequential decision making problems. In *Proceedings of IJCAI 2017*, pages 4566–4572, 2017.

[BP19a] N. Bourdache and P. Perny. Active preference elicitation based on generalized gini functions: Application to the multiagent knapsack problem. In *Proceedings of AAAI-2019*, 2019.

[BP19b] Nadjet Bourdache and Patrice Perny. Active preference learning based on generalized gini functions: Application to the multiagent knapsack problem. In *Proceedings of Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019)*, 2019.

[BPPS06] Craig Boutilier, Relu Patrascu, Pascal Poupart, and Dale Schuurmans. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence*, 170(8-9):686–713, 2006.

[BPV14] Nawal Benabbou, Patrice Perny, and Paolo Viappiani. Incremental elicitation of Choquet capacities for multicriteria decision making. In *Proceedings of ECAI*, pages 87–92, 2014.

[BPV17a] N. Benabbou, P. Perny, and P. Viappiani. Incremental elicitation of Choquet capacities for multicriteria choice, ranking and sorting problems. *Artificial Intelligence*, 246:152–180, 2017.

[BPV17b] Nawal Benabbou, Patrice Perny, and Paolo Viappiani. Incremental elicitation of Choquet capacities for multicriteria choice, rank-

ing and sorting problems. *Artificial Intelligence*, 246:152–180, 2017.

[BR07] D. G. Bridge and F. Ricci. Supporting product selection with query editing recommendations. In *Proceedings of RecSys 2007*, pages 65–72. ACM, 2007.

[Bra12] Darius Braziunas. *Decision-theoretic elicitation of generalized additive utilities*. PhD thesis, 2012.

[Bur02] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, England, 2004.

[CBGVTG15] Leopoldo Eduardo Cárdenas-Barrón, José Luis González-Velarde, and Gerardo Treviño-Garza. A new approach to solve the multi-product multi-period inventory lot sizing with supplier selection problem. *Computers & Operations Research*, 64:225 – 232, 2015.

[CDLS06] Robert G Cowell, Philip Dawid, Steffen L Lauritzen, and David J Spiegelhalter. *Probabilistic networks and expert systems: Exact computational methods for Bayesian networks*. Springer Science & Business Media, 2006.

[CFZ93] Sohail S. Chaudhry, Frank G. Forst, and James L. Zydiak. Vendor selection with price breaks. *European Journal of Operational Research*, 70(1):52 – 66, 1993.

[Che17] Z.H. Che. A multi-objective optimization algorithm for solving the supplier selection problem with assembly sequence planning and assembly line balancing. *Computers & Industrial Engineering*, 105:247 – 259, 2017.

[CKP00] Urszula Chajewska, Daphne Koller, and Ronald Parr. Making rational decisions using adaptive utility elicitation. In *Proceedings of AAAI/IAAI*, pages 363–369, 2000.

[CL14] Junyi Chai and James Liu. A novel believable rough set approach for supplier selection. *Expert Systems with Applications: An International Journal*, 41:92–104, 01 2014.

[CLN13] Junyi Chai, James N.K. Liu, and Eric W.T. Ngai. Application of decision-making techniques in supplier selection: A systematic review of literature. *Expert Systems with Applications*, 40(10):3872 – 3885, 2013.

[CN15] Junyi Chai and Eric WT Ngai. Multi-perspective strategic supplier selection in uncertain environments. *International Journal of Production Economics*, 166:215–225, 2015.

[CN20] Junyi Chai and Eric W.T. Ngai. Decision-making techniques in supplier selection: Recent accomplishments and what lies ahead. *Expert Systems with Applications*, 140:112903, 2020.

[CP04] Li Chen and Pearl Pu. Survey of preference elicitation methods. Technical report, 2004.

[CPRV11] Gianluca Campanella, Alfredo Pereira, Rita A Ribeiro, and Maria Leonilde R Varela. Collaborative dynamic decision making: A case study from b2b supplier selection. In *Proceedings of Euro Working Group Workshop on Decision Support Systems*, pages 88–102. Springer, 2011.

[CR11] Gianluca Campanella and Rita A Ribeiro. A framework for dynamic multiple-criteria decision making. *Decision Support Systems*, 52(1):52–60, 2011.

[Deb59a] Gerard Debreu. *Theory of value: An axiomatic analysis of economic equilibrium*. Number 17. Yale University Press, 1959.

[Deb59b] Gerard Debreu. Topological methods in cardinal utility theory. Cowles Foundation Discussion Papers 76, Cowles Foundation for Research in Economics, Yale University, 1959.

[DGH+15] Baigang Du, Shunsheng Guo, Xiaorong Huang, Yibing Li, and Guo Jun. A pareto supplier selection algorithm for minimum the life cycle cost of complex product system. *Expert Systems with Applications*, 42, 06 2015.

[DJ97] S. L. Dittmer and F. V. Jensen. Myopic value of information in influence diagrams. In *Proceedings of UAI '97:*, pages 142–149, 1997.

[DTP18] Paolo Dragone, Stefano Teso, and Andrea Passerini. Constructive preference elicitation over hybrid combinatorial spaces. In *Proceedings of Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[Dye83] Martin E Dyer. The complexity of vertex enumeration methods. *Mathematics of Operations Research*, 8(3):381–402, 1983.

[EW05] M. Ehrgott and M. M. Wiecek. *Mutiobjective Programming*, pages 667–708. Springer New York, New York, NY, 2005.

[Fan63] Ky Fan. On the krein-milman theorem. *Convexity*, 7:211–220, 1963.

[Far84] Peter H Farquhar. State of the art—utility assessment methods. *Management science*, 30(11):1283–1300, 1984.

[Fei98] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[FGE05a] J. Figueira, S. Greco, and M. Ehrgott. *Multiple Criteria Decision Analysis—State of the Art Surveys*. Springer International Series in Operations Research and Management Science Volume 76, 2005.

[FGE+05b] JosÉ Figueira, Salvatore Greco, Matthias Ehrogott, et al. Multiple criteria decision analysis: State of the art surveys. *International Series in Operations Research and Management Science*, 2005.

[Fis65] Peter C Fishburn. Independence in utility theory with whole product sets. *Operations research*, 13(1):28–45, 1965.

[Fis67a] Peter C Fishburn. Interdependence and additivity in multivariate, unidimensional expected utility theory. *International Economic Review*, 8(3):335–342, 1967.

[Fis67b] Peter C Fishburn. Methods of estimating additive utilities. *Management science*, 13(7):435–453, 1967.

[Fis70] Peter C Fishburn. Utility theory for decision making. Technical report, Research analysis corp McLean VA, 1970.

[FSST17] Piotr Faliszewski, Piotr Skowron, Arkadii Slinko, and Nimrod Talmon. Multiwinner voting: A new challenge for social choice theory. *Trends in computational social choice*, 74:27–47, 2017.

[GDF72] Arthur M Geoffrion, James S Dyer, and A Feinberg. An interactive approach for multi-criterion optimization, with an application to the operation of an academic department. *Management science*, 19(4-part-1):357–368, 1972.

[GK03] Soumyadip Ghosh and Jayant Kalagnanam. Polyhedral sampling for multiattribute preference elicitation. In *Proceedings of Proceedings of the 4th ACM conference on Electronic Commerce*, pages 256–257, 2003.

[GMS14] Salvatore Greco, Vincent Mousseau, and Roman Słowiński. Robust ordinal regression for value functions handling interacting criteria. *European Journal of Operational Research*, 239(3):711–730, 2014.

[GP04] Christophe Gonzales and Patrice Perny. Gai networks for utility elicitation. *KR*, 4:224–234, 2004.

[GPR+10] M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Elicitation strategies for soft constraint problems with missing preferences: Properties, algorithms and experimental studies. *Artificial Intelligence*, 174(3-4):270–294, 2010.

[GPS10] Lucie Galand, Patrice Perny, and Olivier Spanjaard. Choquet-based optimisation in multiobjective shortest path and spanning tree problems. *European Journal of Operational Research*, 204(2):303–315, 2010.

[GS10a] Shengbo Guo and Scott Sanner. Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries. In *Proceedings of Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 289–296, 2010.

[GS10b] Shengbo Guo and Scott Sanner. Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries. In *Proceedings of the Thirteenth International Conference*

on Artificial Intelligence and Statistics, (AISTATS-10), pages 289–296, Chia Laguna Resort, Sardinia, Italy, 2010.

[GSFM10] Salvatore Greco, Roman Słowiński, José Rui Figueira, and Vincent Mousseau. Robust ordinal regression. In *Proceedings of Trends in multiple criteria decision analysis*, pages 241–283. Springer, 2010.

[GW05] Krzysztof Gajos and Daniel S. Weld. Preference elicitation for interface optimization. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST-05)*, pages 173–182, Seattle, WA, USA, 2005.

[Haz86] Gordon B Hazen. Partial information, dominance, and potential optimality in multiattribute utility theory. *Operations research*, 34(2):296–310, 1986.

[HB16] Seyedmohsen Hosseini and Kash Barker. A Bayesian network model for resilience-based supplier selection. *International Journal of Production Economics*, 180:68 – 87, 2016.

[HGM14] F. Hamdi, A. Ghorbel, and F. Masmoudi. Supplier selection under disruption risks using stochastic mixed linear programming techniques. In *Proceedings of 2014 International Conference on Advanced Logistics and Transport (ICALT)*, pages 368–373, May 2014.

[HKBR99] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1999*, pages 230–237. Association for Computing Machinery, Inc, 1999.

[HTF14] Ramzi Hammami, Cecilia Temponi, and Yannick Frein. A scenario-based stochastic model for supplier selection in global context with multiple buyers, currency fluctuation uncertainties, and price discounts. *European Journal of Operational Research*, 233(1):159 – 170, 2014.

[HWI03] Hillary A Holloway and Chelsea C White Iii. Question selection for multi-attribute decision-aiding. *European Journal of Opera-*

tional Research, 148(3):525–533, 2003.

[HY16] Kuo-Jen Hu and Vincent F. Yu. An integrated approach for the electronic contract manufacturer selection problem. *Omega*, 62:68 – 81, 2016.

[ILC01a] Vijay S Iyengar, Jon Lee, and Murray Campbell. Evaluating multiple attribute items using queries. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 144–153. ACM, 2001.

[ILC01b] Vijay S. Iyengar, Jon Lee, and Murray Campbell. Q-Eval: Evaluating multiple attribute items using queries. pages 144–153, Tampa, FL, USA, 2001.

[ILO17] IBM ILOG. IBM ILOG CPLEX Optimization Studio, V12.8.0, 2017.

[JLS82] Eric Jacquet-Lagreze and Jean Siskos. Assessing a set of additive utility functions for multicriteria decision-making, the uta method. *European journal of operational research*, 10(2):151–164, 1982.

[KCM17] Dinçer Konur, James F. Campbell, and Sepideh A. Monfared. Economic and environmental considerations in a stochastic inventory control model with order splitting under different delivery schedules among suppliers. *Omega*, 71:46 – 65, 2017.

[KD14] E. Karsak and Mehtap Dursun. An integrated supplier selection methodology incorporating qfd and dea with imprecise data. *Expert Systems with Applications*, 41:6995–7004, 11 2014.

[KL86] Pekka J Korhonen and Jukka Laakso. A visual interactive method for solving the multiple criteria problem. *European Journal of Operational Research*, 24(2):277–287, 1986.

[KL05] Kathrin Konczak and Jérôme Lang. Voting procedures with incomplete preferences. In *Proceedings of Proceedings IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling*, volume 20. Citeseer, 2005.

[Kli92] Ronald Klimberg. Grads: A new graphical display system for visualizing multiple criteria solutions. *Computers & operations research*, 19(7):707–711, 1992.

[KM97] I Kaliszewski and W Michalowski. Efficient solutions and bounds on tradeoffs. *Journal of Optimization Theory and Applications*, 94(2):381–394, 1997.

[Koj07] Ivan Kojadinovic. Minimum variance capacity identification. *European Journal of Operational Research*, 177(1):498–514, 2007.

[Kor05] Pekka Korhonen. Interactive methods. In *Proceedings of Multiple criteria decision analysis: state of the art surveys*, pages 641–661. Springer, 2005.

[KP84] Zbigniew Wawrzyniec Kmietowicz and AD Pearman. Decision theory, linear partial information and statistical dominance. *Omega*, 12(4):391–399, 1984.

[KS14] Gokhan Kirlik and Serpil Sayın. A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. *European Journal of Operational Research*, 232(3):479–488, 2014.

[KVVA17] S. Kaddani, D. Vanderpooten, J. M. Vanpeperstraete, and H. Aissi. Weighted sum model with partial preference information: Application to multi-objective optimization. *European Journal of Operational Research*, 260(2):665–679, 2017.

[KY13] Panos Kouvelis and Gang Yu. *Robust discrete optimization and its applications*, volume 14. Springer Science & Business Media, 2013.

[LB11a] Tyler Lu and Craig Boutilier. Budgeted social choice: From consensus to personalized decision making. In *Twenty-Second International Joint Conference on Artificial Intelligence*. Citeseer, 2011.

[LB11b] Tyler Lu and Craig Boutilier. Robust approximation and incremental elicitation in voting protocols. In *Proceedings of Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[LSZ92] Vahid Lotfi, Theodor J Stewart, and Stanley Zionts. An aspiration-level interactive model for multiple criteria decision making. *Computers & Operations Research*, 19(7):671–681, 1992.

[LWM+15] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: a survey. *Decision Support Systems*, 74:12–32, 2015.

[MIT96] Tadahiko Murata, Hisao Ishibuchi, and Hideo Tanaka. Multi-objective genetic algorithm and its applications to flowshop scheduling. *Computers & industrial engineering*, 30(4):957–968, 1996.

[MR05] Patrick Meyer and Marc Roubens. Choice, ranking and sorting in fuzzy multiple criteria decision aid. In *Proceedings of Multiple criteria decision analysis: State of the art surveys*, pages 471–503. Springer, 2005.

[MRW12] R. Marinescu, A. Razak, and N. Wilson. Multi-objective influence diagrams. In *Proceedings of UAI-2012*, pages 574–583, 2012.

[MRW13] R. Marinescu, A. Razak, and N. Wilson. Multi-objective constraint optimization with tradeoffs. In *Proceedings of CP-2013*, pages 497–512, 2013.

[MW16] Mojtaba Montazery and Nic Wilson. Learning user preferences in matching for ridesharing. In *Proceedings of ICAART (2)*, pages 63–73, 2016.

[NY15] Bimal Nepal and Om Prakash Yadav. Bayesian belief network-based framework for sourcing risk analysis during supplier selection. *International Journal of Production Research*, 53(20):6114–6135, 2015.

[OW13] C. O'Mahony and N. Wilson. Sorted-pareto dominance and qualitative notions of optimality. In *Proceedings of ECSQARU'2013*, pages 449–460, 2013.

[PB07] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *Proceedings of The adaptive web*, pages 325–341. Springer, 2007.

[PFT03] Pearl Pu, Boi Faltings, and Marc Torrens. User-involved preference elicitation. Technical report, 2003.

[PM05] Robert Price and Paul R. Messinger. Optimal recommendation sets: Covering uncertainty over user preferences. In *Proceedings,*

The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA, pages 541–548, 2005.

[Rai68] H. Raiffa. *Decision analysis*. Addison-Wesley, 1968.

[RD11] Jafar Rezaei and Mansoor Davoodi. Multi-objective models for lot-sizing with supplier selection. *International Journal of Production Economics*, 130(1):77 – 86, 2011.

[RFH20] Gade Pandu Rangaiah, Zemin Feng, and Andrew F Hoadley. Multi-objective optimization applications in chemical process engineering: Tutorial and review. *Processes*, 8(5):508, 2020.

[RS16] K. Renganath and M. Suresh. Supplier selection using fuzzy mcdm techniques: A literature review. In *Proceedings of 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pages 1–6, 2016.

[Saa08] Thomas Saaty. The analytic network process. *Encyclopedia of Operations Research and Management*, 1, 04 2008.

[Sar78] Rakesh Kumar Sarin. Elicitation of subjective probabilities in the context of decision-making. *Decision Sciences*, 9(1):37–48, 1978.

[Sav51] Leonard J Savage. The theory of statistical decision. *Journal of the American Statistical association*, 46(253):55–67, 1951.

[Sav72] Leonard J Savage. *The foundations of statistics*. Courier Corporation, 1972.

[SC83] Ralph E Steuer and Eng-Ung Choo. An interactive weighted tchebycheff procedure for multiple objective programming. *Mathematical programming*, 26(3):326–344, 1983.

[SGM05] Yannis Siskos, Evangelos Grigoroudis, and Nikolaos F Matsatsinis. Uta methods. In *Proceedings of Multiple criteria decision analysis: State of the art surveys*, pages 297–334. Springer, 2005.

[SGMP18] Olivier Sobrie, Nicolas Gillis, Vincent Mousseau, and Marc Pirlot. Uta-poly and uta-splines: additive value functions with polynomial marginals. *European Journal of Operational Research*, 264(2):405–418, 2018.

[SH92] Ahti A Salo and Raimo P Hämäläinen. Preference assessment by imprecise ratio statements. *Operations Research*, 40(6):1053–1061, 1992.

[SH01] Ahti A Salo and Raimo P Hamalainen. Preference ratios in multiattribute evaluation (prime)-elicitation and decision procedures under incomplete information. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 31(6):533–545, 2001.

[SH10] A. Salo and R. P. Hämäläinen. Preference programming – multicriteria weighting models under incomplete information. In *Proceedings Handbook of Multicriteria Analysis*, pages 167–187. Springer Berlin Heidelberg, 2010.

[Sim55] Herbert A Simon. A behavioral model of rational choice. *The quarterly journal of economics*, 69(1):99–118, 1955.

[Sin05] Carsten Sinz. Towards an optimal cnf encoding of boolean cardinality constraints. In *Proceedings of International conference on principles and practice of constraint programming*, pages 827–831. Springer, 2005.

[SKKR01] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of Proceedings of the 10th international conference on World Wide Web*, pages 285–295, 2001.

[SR91] Wan S Shin and Arunachalam Ravindran. Interactive multiple objective optimization: Survey i—continuous case. *Computers & Operations Research*, 18(1):97–114, 1991.

[SS78] Ralph E Steuer and Albert T Schuler. An interactive multiple-objective linear programming approach to a problem in forest management. *Operations Research*, 26(2):254–269, 1978.

[SSYT12] Krishnendu Shaw, Ravi Shankar, Surendra S. Yadav, and Lakshman S. Thakur. Supplier selection using fuzzy AHP and fuzzy multi-objective linear programming for developing low carbon supply chain. *Expert Systems with Applications*, 39(9):8182 – 8192, 2012.

[Ste17] Željko Stević. Criteria for supplier selection: A literature review. *International Journal of Engineering, Business and Enterprise Applications*, 1:23–27, 02 2017.

[Sto19] Russell Stoneback. Python satellite data analysis toolkit 2.1.0, 2019.

[TC17] Shing Chih Tsai and Sin Ting Chen. A simulation-based multi-objective optimization framework: a case study on inventory management. *Omega*, 70:148–159, 2017.

[TFDCSA16] Madjid Tavana, Alireza Fallahpour, Debora Di Caprio, and Francisco J. Santos-Arteaga. A hybrid intelligent fuzzy predictive model with simulation for supplier evaluation and selection. *Expert Systems with Applications*, 61:129–144, 2016.

[Tim13] Mikhail Timonin. Robust optimization of the Choquet integral. *Fuzzy sets and systems*, 213:27–46, 2013.

[TK74] Amos Tversky and Daniel Kahneman. Judgment under uncertainty: Heuristics and biases. *science*, 185(4157):1124–1131, 1974.

[T.L88] Saaty T.L. What is the analytic hierarchy process? *Mathematical Models for Decision Support*, 48:109 – 121, 1988.

[TPV16] Stefano Teso, Andrea Passerini, and Paolo Viappiani. Constructive preference elicitation by setwise max-margin learning. *arXiv preprint arXiv:1604.06020*, 2016.

[Tro18] Matthias C. M. Troffaes. Python wrapper for komei fukuda's cddlib, 2018.

[Vai89] Pravin M Vaidya. Speeding-up linear programming using fast matrix multiplication. In *30th annual symposium on foundations of computer science*, pages 332–337. IEEE Computer Society, 1989.

[VB09] Paolo Viappiani and Craig Boutilier. Regret-based optimal recommendation sets in conversational recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 101–108. ACM, 2009.

[VB10] Paolo Viappiani and Craig Boutilier. Optimal Bayesian recommendation sets and myopically optimal choice query sets. In *Proceedings of Advances in neural information processing systems*, pages 2352–2360, 2010.

[VB20] Paolo Viappiani and Craig Boutilier. On the equivalence of optimal recommendation sets and myopically optimal query sets. *Artificial Intelligence*, page 103328, 2020.

[vdB20] Jan van den Brand. A deterministic linear program solver in current matrix multiplication time. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 259–278. SIAM, 2020.

[Via12] Paolo Viappiani. Monte carlo methods for preference learning. In *Proceedings of International Conference on Learning and Intelligent Optimization*, pages 503–508. Springer, 2012.

[VLHB19] Ivan Vendrov, Tyler Lu, Qingqing Huang, and Craig Boutilier. Gradient-based optimization for Bayesian preference elicitation. *arXiv preprint arXiv:1911.09153*, 2019.

[VNM47] John Von Neumann and Oskar Morgenstern. Theory of games and economic behavior, 2nd rev. 1947.

[WB03] Tianhan Wang and Craig Boutilier. Incremental utility elicitation with the minimax regret decision criterion. In *Proceedings of IJCAI*, volume 3, pages 309–316, 2003.

[WCB91] Charles A. Weber, John R. Current, and W.C. Benton. Vendor selection criteria and methods. *European Journal of Operational Research*, 50(1):2 – 18, 1991.

[Web87] Martin Weber. Decision making with incomplete information. *European journal of operational research*, 28(1):44–57, 1987.

[Wie80] Andrzej P Wierzbicki. The use of reference objectives in multiobjective optimization. In *Proceedings of Multiple criteria decision making theory and application*, pages 468–486. Springer, 1980.

[Wie07] M. Wiecek. Advances in cone-based preference modeling for decision making with multiple criteria. *Decision Making in Manufacturing and Services*, 1(1-2):153–173, 2007.

[WLX11] Zhoujing Wang, Kevin W. Li, and Jianhui Xu. A mathematical programming approach to multi-attribute decision making with interval-valued intuitionistic fuzzy assessment information. *Expert Systems with Applications*, 38(10):12462 – 12469, 2011.

[WO11] N. Wilson and C. O'Mahony. The relationships between qualitative notions of optimality for decision making under logical uncertainty. In *Proceedings of AICS-2011*, 2011.

[WRM15] N. Wilson, A. Razak, and R. Marinescu. Computing possibly optimal solutions for multi-objective constraint optimisation with tradeoffs. In *Proceedings of IJCAI-2015*, 2015.

[WSB12] Nilesh Ware, Surya Prakash Singh, and D. Banwet. Supplier selection problem: A state-of-the-art review. *Management Science Letters*, 2:1465–1490, 07 2012.

[WSD84] Chelsea C White, Andrew P Sage, and Shigeru Dozono. A model of multiattribute decisionmaking and trade-off weight determination under uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics*, (2):223–229, 1984.

[XC11] Lirong Xia and Vincent Conitzer. Determining possible and necessary winners given partial orders. *Journal of Artificial Intelligence Research*, 41:25–67, 2011.

[Yag88] Ronald R Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on systems, Man, and Cybernetics*, 18(1):183–190, 1988.

[Yu74] P. Yu. Cone convexity, cone extreme points, and nondominated solutions in decision problems with multiobjectives. *Journal of Optimization Theory and Applications*, 14(3):319–377, 1974.

[ZFS16] Konrad Zimmer, Magnus Fröhling, and Frank Schultmann. Sustainable supplier management – a review of models supporting sustainable supplier selection, monitoring and development. *International Journal of Production Research*, 54(5):1412–1442, 2016.

[ZTK14] Edmundas Kazimieras Zavadskas, Zenonas Turskis, and Simona Kildienė. State of art surveys of overviews on mcdm/madm

methods. *Technological and economic development of economy*, 20(1):165–179, 2014.

[ZW76] Stanley Zionts and Jyrki Wallenius. An interactive programming method for solving the multiple criteria problem. *Management science*, 22(6):652–663, 1976.