Representation and exploitation of conditional preferences within a possibilistic framework : comparison withother approaches and extensions

Syrine Saidi

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*
Cotutelle internationale *Université de Tunis: Institut Supérieur de Gestion de Tunis*

Présentée et soutenue le *24/11/2022* **par :**

Syrine SAIDI

Représentation et exploitation des préférences conditionnelles dans un cadre possibiliste : Comparaison avec d'autres approches et extensions

JURY

Salem BENFERHAT	Professeur, Labo. CRIL, Lens	Rapporteur
Sébastien DESTERCKE	HdR, CRNS, labo. Heudiasyc, Compiègne	Rapporteur
Agnès RICO	Maître de Conférences, Université Claude	Examinatrice
Zied ELOUEDI	Professeur, ISG Tunis	Président
Nahla BEN AMOR	Professeur, ISG Tunis	Directrice de thèse
Henri PRADE	DR Emérite, CNRS, IRIT	Directeur de thèse
Didier DUBOIS	DR Emérite, CNRS, IRIT	Invité
Florence DUPIN de SAINT CYR	Maître de Conférences, IRIT	Invitée

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle

Unité de Recherche :
Institut de Recherche en Informatique de Toulouse (UMR 5505)

Directeur(s) de Thèse :
Nahla BEN AMOR et *Henri PRADE*

Rapporteurs :
Salem BENFERHAT et *Sébastien DESTERCKE*

Representing and exploiting conditional preferences within a possibilistic framework : Comparison with other approaches and extensions

Acknowledgement

Without the support and collaboration of some people, this thesis could not have been completed. Thus, I would like to thank all those who have contributed in any way to the progress of my work.

For more than three years, I had the chance to be supervised by incredibly gifted and professional supervisors through whom I was able to develop skills and a love for research to finally be able to complete this thesis in appropriate conditions. I would first like to express my deepest gratitude to my thesis director *Henri Prade* for his patience, his help and his precious advises through all this journey. I would also like to thank my thesis supervisor *Nahla Ben Amor* for her guidance and support. A special thank to *Didier Dubois* for taking part in my work and accepting to support my thesis. Without your immense knowledge, your encouragement and especially your faith in me, this work would have never been achieved. Thank you all for your daily commitment and guidance from the beginning of my academic research to this day.

I would also like to express my gratitude and sincere respect to all the members of the jury who agreed to evaluate my work.

My earnest thanks to all members of my family and specially to my mother for her valuable support. Her dedicated efforts contributed a lot for completion of my thesis. A special thanks to the person who always believed in me and supported me in the difficult times. Your unfailing love and support gave me the strength and patience to work and continue through all these years.

Finally, I extend my deepest thanks to everyone who contributed to the success of this work.

Summary

π-pref nets, CP-nets and possibly LP-trees are able to encode specifications of the form "In the context of u, I prefer a to its negation" which is quite similar to the piece of knowledge "If u then generally a". This rule can be encoded by possibility logic. In some works, researchers have been particularly interested in reasoning with default rules for representing some state of affairs in a possibilistic framework. Because of the similarity between a user preference and a default rule, this work has caught our attention and led us to question whether interpreting a collection of user preference statements as default rules and using some informational principles permit to construct the same ordering as induced by a given graphical preference representation.

One of the main goals of this manuscript is also to compare the expressive power of CP-nets, LP-trees and π-pref nets. Using a possibilistic framework, specifications of a user may be also encoded as default rules on which several reasoning approaches are applied to therefore compare their induced orderings. The work is restricted to Boolean variables.

The dissertation is divided in seven chapters. The first two chapters are dedicated to provide the background knowledge. On the one hand, they review the state of the art on conditional preference representations and on the other hand the basis of the possibility theory . The first chapter deals with qualitative graphical models, namely CP-nets, their extension TCP-nets and LP-trees. We provide independence assumption of each model, their induced orderings over complete configurations, in addition to explaining queries that can be performed over them. Chapter 2 is devoted to possibility theory and its use for representing preferences in different formats such as possibility distributions, logical bases or graphical networks. The first part of Chapter 3 gives a brief background about possibilistic preference networks (π-pref nets) and discusses their expressiveness and consistency with regard to CP-nets. The second part of the chapter introduces new variants of π-pref nets by using different scales for encoding preference degrees. Besides, some researchers have proposed to deal with default knowledge formalized by means of constraints expressed in the setting of the possibility theory. This is the aim of the chapter 4, we will apply a similar approach for modeling preferences in the aim of finding an order-ranking over solutions of a given preference problem thus handled by means of default rules. The resulting orderings are compared to those obtained by different order approaches and particularly the Pareto order. Chapter 5 discusses repairs and refinements of the complete pre-orders obtained from preferences encoded as default-like rules. Chapter 6 discusses our last goal that consists on studying the expressive and representative power of LP-trees compared to

π-pref nets. It also discusses the procedures for transforming an LP-tree into a π-pref net. Finally and before concluding, Chapter 7 presents an implemented toolbox that supports CP-nets and π-pref nets as graphical structures in addition to the default rule-based algorithms discussed in previous chapters.

Keywords

Graphical preference modeling, possibilistic preference network, default-like preferences, Pareto order, ceteris paribus preferences, lexicographic order.

Résumé

Les réseaux π-pref-nets, les CP-nets et éventuellement les LP-trees sont capables d'encoder des spécifications de la forme "Dans le contexte de u, je préfère a à sa négation" qui est assez similaire à une règle par défaut "Si u alors généralement a". Cette règle peut être codée en théorie des possibilités. Dans certains travaux, des chercheurs se sont particulièrement intéressés au raisonnement avec des règles par défaut pour représenter un certain état de choses dans un cadre possibiliste. En raison de la similarité entre une préférence d'utilisateur et une règle par défaut, ce travail a attiré notre attention et nous a conduit à nous demander si l'interprétation d'une collection d'énoncés de préférences d'utilisateurs comme des règles par défaut et l'utilisation de certains principes informationnels permettent de construire le même ordre que celui induit par une représentation graphique donnée des préférences.

L'un des principaux objectifs de ce manuscrit est aussi de comparer le pouvoir expressif des CP-nets, LP-trees et des π-pref nets. En utilisant un cadre possibiliste, les spécifications d'un utilisateur peuvent aussi être encodées comme des règles par défaut sur lesquelles plusieurs approches de raisonnement sont appliquées pour ainsi comparer leurs ordonnancements induits. Le travail est limité aux variables booléennes.

La thèse est divisée en sept chapitres. Les deux premiers chapitres sont consacrés à fournir les connaissances de base. D'une part, ils passent en revue l'état de l'art sur les représentations des préférences conditionnelles et d'autre part, les bases de la théorie des possibilités. Le premier chapitre traite des modèles graphiques qualitatifs, à savoir les CP-nets, leur extension TCP-nets et les LP-trees. Nous indiquons l'hypothèse d'indépendance de chaque modèle, leurs ordonnancements induits sur des configurations complètes, en plus d'expliquer les requêtes qui peuvent être effectuées sur eux. Le chapitre 2 est consacré à la théorie des possibilités et à son utilisation pour représenter les préférences sous différents formats tels que les distributions de possibilités, les bases logiques ou les réseaux graphiques. La première partie du chapitre 3 donne un bref aperçu des réseaux de préférences possibilistes (π-pref nets) et discute de leur expressivité et de leur cohérence par rapport aux CP-nets. La deuxième partie du chapitre présente de nouvelles variantes des réseaux π-pref en utilisant différentes échelles pour encoder les degrés de préférence. Par ailleurs, des chercheurs ont proposé de traiter la connaissance par défaut formalisée au moyen de contraintes exprimées dans le cadre de la théorie des possibilités. C'est l'objet du chapitre 4, où nous appliquerons une approche similaire pour modéliser les préférences et pour trouver un ordre de classement sur les solutions d'un problème de préférence donné ainsi traité au moyen de règles par défaut. Les classements obtenus sont comparés à ceux

obtenus par différentes approches d'ordre et notamment l'ordre de Pareto. Le chapitre 5 traite des réparations et des raffinements des préordres complets obtenus à partir de préférences codées comme des règles par défaut. Le chapitre 6 traite de notre dernier objectif qui consiste à étudier le pouvoir expressif et représentatif des LP-trees par rapport aux π-pref nets. Il aborde également les procédures de transformation d'un arbre LP-tree en un réseau π-pref nets. Enfin, avant de conclure, le chapitre 7 présente une boîte à outils implémentée qui prend en charge les CP-nets et les π-pref nets en tant que structures graphiques en plus des algorithmes basés sur des règles par défaut abordés dans les chapitres précédents.

Mots clés

Modélisation graphique des préférences, réseau de préférences possibiliste, préférences par défaut, ordre de Pareto, préférence ceteris paribus, ordre lexicographique.

Contents

List of Figures

List of Tables

General Introduction

For several decades, modeling human preferences has been regarded as a particularly promising research field of great interest in decision analysis. Going from economics, e-commerce [Ribeiro et al., 2018], recommender systems [Wang et al., 2018], computer science and psychology to space, medicine, and politics, fields of applications in this area are many to be counted. Handling preferences requires to go through three main steps: data collection, formal representation and model querying. Preference specifications can be gathered using elicitation techniques or machine learning methods. In fact, the former practice requires knowledge to be processed directly from human beings, e.g., surveys, observation or interviews. The latter one seeks to acquire new knowledge or function approximation in order to derive an unknown model based on input data sets. The second step, which is preference representation, consists on encoding these preferences into logical or graphical models that can encode qualitative or numerical preferences. Possibilistic logic [Benferhat et al., 2001c], propositional languages [Coste-Marquis et al., 2004] and modal logic [van Benthem et al., 2009] are some of the logical frameworks that allow to model preferences. Graphical models have been motivated by the need of a compact representation of user preferences. A general overview of graphical preference representations can be found in [Ben Amor et al., 2016a]. Figure 1 is a timeline that sums up the main preference approach that exist in the literature. The upper part of the scale designates the graphical representations, while the lower part designates logical representations. Rectangles in thick lines represent qualitative graphical models, those with strong lines indicate quantitative graphical models, while those in dashed lines can be classified as both qualitative and quantitative models. Once the preference model is constructed, the last step consists on exploiting information retrieved from these models to answer some given queries such as finding a dominance relation between alternatives or finding the most satisfying solution.

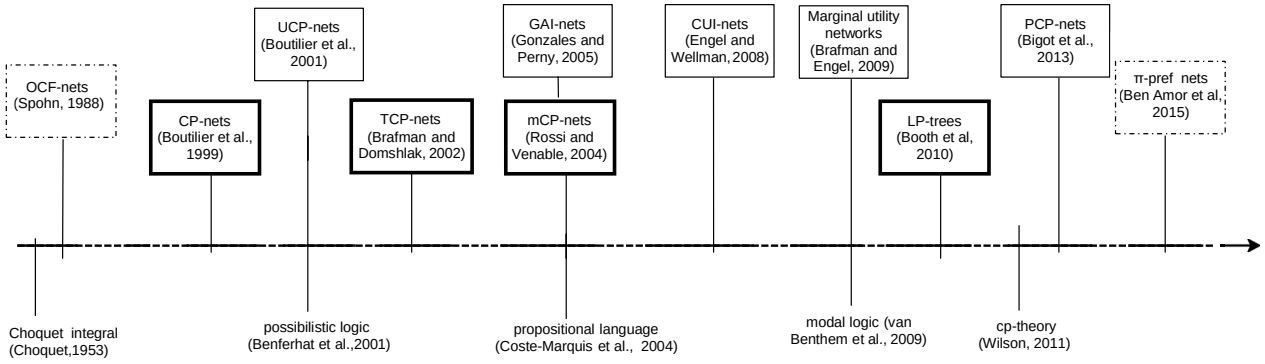


Figure 1: Timeline of some known preference representations

Given any decision analysis task, comparing all conceivable configurations of the universe of discourse comes down to determine a specific function that manages to rank order them. No need to mention that it is obviously unreasonable to require a human being to specify an explicit preference ordering over a prohibitive number of solutions. This process leads to a high computational cost, then, to an impossibility to construct dominance relations between configurations. In fact, humans are more eager to express their preferences in a contextual manner than generally. For these reasons, managing preferences using graphical structures appears to be an emerging challenge, since in addition to its computational efficiency, it provides a succinct and compact tool for data collection and modeling. Graphical preference representations are divided into two main categories: qualitative and quantitative models. A user can actually express its preferences by providing numerical or ordinal rankings. Ordinal Conditional function networks [Spohn, 1988], Utility CP-nets (UCP-nets) [Boutilier et al., 2001], Generalized Additive Independence networks (GAI-nets) [Gonzales and Perny, 2004] and marginal utility networks [Brafman and Engel, 2009a] are some of many models where preferences are expressed by means of numerical values. When it comes to ordinal representations of preferences, we cite Conditional preference networks (CP-nets) [Boutilier et al., 1999], Tradeoffs-enhanced CP-nets (TCPnets) [Brafman and Domshlak, 2002], Lexicographic Preference trees (LP-trees) [Booth et al., 2010], etc. In Figure 1, quantitative models are depicted by bold line rectangles, qualitative models are depicted by thin line rectangles and semi-qualitative models are drawn by dotted lines rectangles.

In this work, we will mainly focus on qualitative representation frameworks for representing conditional preferences, such as CP-nets, LP-trees and the π-pref nets (for *possibilistic preference networks*) [Ben Amor et al., 2018a] which have been more recently introduced. Even though CP-nets and LP-trees have been introduced to compactly represent (conditional) preferences, motivations and perspectives in which au-

thors placed themselves when inventing them are not the same. In fact, CP-nets, which appeared already in 1999 and which experienced their great development between 2000 and 2005, were motivated by a representation concern and were more focused on reasoning. Instead, LP-trees came from several groups of researchers often motivated by learning concerns. Still CP-nets have also been investigated for learning purposes. Even though they were not designed for this end, authors [Chevaleyre et al., 2010] [Fürnkranz and Hüllermeier, 2010] [Liu et al., 2018] have used the CP-nets format for learning preferences.

LP-trees require the user to specify a total order over the domain of variable(s) composing each node of the graph. This seems to be restrictive and much demanding to the user especially as the number of grouped variables of same importance increases. CP-nets use the *ceteris paribus* assumption to infer a partial order on complete configurations where an implicit priority on preferences associated with father nodes seems to be enforced without being explicitly specified by the user. A π-pref net is a graphical model that compactly represents conditional preferences. It seems to offer an interesting tool that enables to avoid the cumbersome task of the elicitation process imposed by LP-trees and the skewed effect of the ceteris paribus property on variables importance. Actually, the CP-net structure was inspired by Bayesian networks such that decision variables are associated with local tables that contain conditional preferences. π-pref nets were inspired from possibilistic networks where variables are associated with possibility distributions expressed with symbolic degrees encoding an ordinal ranking between values of the variable in question. Symbolic degrees take values in an ordinal scale and can be instantiated by numerical values. π-pref nets come thus halfway between quantitative and qualitative representations.

π-pref nets, CP-nets and possibly LP-trees are able to encode specifications of the form "In the context of u, I prefer a to its negation" which is quite similar to the piece of knowledge "If u then generally a". This rule can be encoded by possibility logic. In some of their works Benferhat and his colleagues [Benferhat et al., 1992] have been particularly interested in reasoning with default rules for representing some state of affairs in a possibilistic framework. Because of the similarity between a user preference and a default rule, this work has caught our attention and led us to question whether interpreting a collection of user preference statements as default rules and using some informational principles permit to construct the same ordering as induced by a given graphical preference representation.

One of the main goals of this manuscript is to compare the expressive power of CP-nets, LP-trees and π-pref nets. Using a possibilistic framework, specifications of

a user may be also encoded as default rules on which several reasoning approaches are applied to therefore compare their induced orderings. The work is restricted to Boolean variables.

The dissertation is divided in seven chapters. The first two chapters are dedicated to provide the background knowledge. On the one hand, they review the state of the art on conditional preference representations and on the other hand the basis of the possibility theory . The first chapter deals with qualitative graphical models, namely CP-nets, their extension TCP-nets and LP-trees. We provide independence assumption of each model, their induced orderings over complete configurations, in addition to explaining queries that can be performed over them. Chapter 2 is devoted to possibility theory and its use for representing preferences in different formats such as possibility distributions, logical bases or graphical networks. The first part of Chapter 3 gives a brief background about possibilistic preference networks (π-pref nets) and discusses their expressiveness and consistency with regard to CP-nets. The second part of the chapter introduces new variants of π-pref nets by using different scales for encoding preference degrees. Besides, Benferhat and his colleagues have proposed to deal with default knowledge formalized by means of constraints expressed in the setting of the possibility theory. In chapter 4, we will apply a similar approach for modeling preferences in the aim of finding an order-ranking over solutions of a given preference problem thus handled by means of default rules. The resulting orderings are compared to those obtained by different order approaches and particularly the Pareto order. Chapter 5 discusses repairs and refinements of the complete pre-orders obtained from preferences encoded as default-like rules. Chapter 6 discusses our last goal that consists on studying the expressive and representative power of LP-trees compared to π-pref nets. It also discusses the procedures for transforming an LP-tree into a π-pref net. Finally and before concluding, Chapter 7 presents an implemented toolbox that supports CP-nets and π-pref nets as graphical structures in addition to the default rule-based algorithms discussed in previous chapters.

Papers summarizing the main contributions of this thesis are:

- [Ben Amor et al., 2019]: Nahla Ben Amor, Didier Dubois, Henri Prade and Syrine Saidi. Revisiting conditional preferences: from defaults to graphical representations. In Proceedings of the 15th International Symbolic and Quantitative Approaches to Reasoning with Uncertainty, pages 187-198, (ECSQARU 2019) Belgrade, Serbia. This paper is presented in Chapter 4;
- [Ben Amor et al., 2021a]: Nahla Ben Amor, Didier Dubois, Henri Prade and Syrine Saidi. Conditional Preference Networks - Refining Solution Orderings

Beyond Pareto Dominance. In Proceedings of the 34th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pages 447-459, (IEA/AIE 2021), Kuala Lumpur, Malaysia. This paper is presented in Chapter 5;

- [Ben Amor et al., 2021b]: Nahla Ben Amor, Didier Dubois, Henri Prade and Syrine Saidi. Réseaux possibilistes de préférences et arbres de préférences lexicographiques – Une comparaison. Actes de la Rencontres Francophones sur la Logique Floue et ses Applications, pages 209-216 (LFA 2021), Paris, France. This paper is presented in Chapter 6;

- [Ben Amor et al., 2022]: Nahla Ben Amor, Didier Dubois, Henri Prade and Syrine Saidi. Possibilistic preference networks and lexicographic preference trees – A comparison. In Proceedings of the 19th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, pages 581-592, (IPMU 2022), Milan, Italy. This work is the English version of the paper in [Ben Amor et al., 2021b].

Chapter 1

Qualitative Graphical Representations of Preferences: CP-nets and LP-trees

1.1 Introduction

Reasoning about user preferences requires to specify three components : a language that encodes information provided by the user about the decision problem, a formal model presenting the order retrieved from user specifications based assumptions, and finally queries to reason or question the model [Domshlak, 2008].

Consider a set of binary-valued decision variables $\mathcal{X}$. In our work, a language consists of a total order (that may be weak) over values of each variable $X \in \mathcal{X}$. This relation can be conditioned by the value of a set of depending variable(s) that are different from X. The language enables to translate preference statements into a formal definition while keeping the elicitation process as simple as possible without ambiguity or loss of information. For instance, a claim of the form $c : a \succ \bar{a}$ means that, in the context of c, the user prefers a to its negation, the conditioning part being optional. This claim is called a *generalized* statement[1]. Concepts correspond to a set of postulates or informational properties that allow to concretize a language into a logical interpretation generating a possible arrangement over complete configurations which correspond to a conjunction of the value of each decision variable. A language, along with its assumptions, describes a model. The latter can subsequently be exploited to answer a number of questions, such as finding the optimal configuration, the top k configurations or compare configurations. Most preference models are mainly composed

[1]All specifications considered in models presented later in this work are considered *generalized* statements.

of two parts: a graphical structure, consisting of a graph (directed in most cases), and an informational component.

Modeling preferences graphically may represent a simple task. However, getting complete configurations ordered is not trivial, since their number is exponential in the number of variables of choice in $\mathcal{X}$. To find the dominance relation between configurations, we generally associate assumptions to a graphical structure such as ceteris paribus [Boutilier et al., 2004], lexicographic order [Booth et al., 2010] or other order semantics, e.g., Pareto, Minimum, Leximin, etc. Conditional specifications of a user may also be encoded differently by assigning some degrees or rankings to values of variables. For instance, we cite ordinal conditional functions [Spohn, 1988], utility functions [Brafman and Engel, 2009b], belief functions [Wang et al., 2018] and possibility distributions [Ben Amor et al., 2014].

In this chapter, we mainly focus on *CP-nets* [Boutilier et al., 1999], their extension *TCP-nets* [Brafman and Domshlak, 2002] and LP-trees [Booth et al., 2010]. CP-nets, composed of directed acyclic graphs (DAGs)[2], obey to the ceteris paribus assumption. LP-trees, that restrict their graphical structure to directed trees, are based on the lexicographic order. TCP-nets, depicted by graphs with directed and un-directed edges, enhance the expressiveness of a CP-net by permitting the expression of (conditional) importance relation between variables.

This chapter is organized as follows. Section 1.2 defines the general structure composing graphical preference models later introduced. Section 1.3 presents one of the most used models for representing conditional preferences, nameley *CP-nets*. The section also addresses an extension of CP-nets that permits to express importance relations between variables, entitled *TCP-nets*. Section 1.5 exposes another graphical preference model called *LP-trees* that is most used for learning purposes. As for TCP-nets, LP-trees are able to encode two types of relations: conditional preference dependencies and importance relations between decision variables. LP-trees however differ by their graphical representation and their semantics. Finally, *CP-theories*, which offer a logical framework for encoding preferences, are introduced in Section 1.6 as a tool for comparing the expressiveness of previously discussed models.

[2]In the original work [Boutilier et al., 1999], CP-net are presented as conditional preference network that can be cyclic, however we limit our work to acyclic graphs.

1.2 Conditional preference networks structure

The basis of graphical preference models that will be dealt with throughout our work is composed of a DAG relating decision variables along with a set of conditional local tables each encoding a total order between values of variables in the context of its parents in the DAG. We consider these components as elements of what we call a *Conditional preference network structure*. Each decision problem is depicted by a *conditional preference network structure* where nodes correspond to features / decision variables , e.g.„ meal, product, mean of transport, etc. and arcs reflect dependencies between them.

Before going any further, we first introduce some basic notations:

- $\mathcal{X} = \{X_1, \ldots, X_N\}$ denotes the set of N decision variables;
- $\forall X_i \in \mathcal{X}$, $\underline{X_i}$ denotes the domain of possible values of X_i. We limit our work to the particular case of Boolean variables, i.e., $\forall X_i \in \mathcal{X}$, $\underline{X_i} = \{x_i, \bar{x_i}\}$;
- The set of parents of node X_i is denoted by U_{X_i};
- $\forall u \in \underline{U_{X_i}}$, the user specification "*If u is true, I prefer x_i to $\bar{x_i}$*" expresses the choice of the agent over X_i in the context of u. This statement is formally written $u : x_i \succ \bar{x_i}$, where $\succ$ is the strict part of $\succeq$, $\sim$ the indifference part of $\succeq$;
- A configuration $\omega = \{x_1^* \wedge x_2^* \wedge \cdots \wedge x_N^*\}$ [3] is a complete assignment of all decision variables in $\mathcal{X}$ such that $\underline{X_i} = \{x_i, \bar{x_i}\}$ for $i = [0, N]$. For a matter of simplicity, we write $\omega = x_1^* x_2^* \ldots x_N^*$ instead of $\omega = \{x_1^* \wedge x_2^* \wedge \cdots \wedge x_N^*\}$;
- $\Omega = \underline{X_1} \times \cdots \times \underline{X_N} = \{\omega_0, \ldots, \omega_{2^N-1}\}$ denotes the universe of discourse composed of 2^N configurations;
- $\forall \omega \in \Omega, \omega[X_i]$ denotes the projection of ω on the variable X_i;
- $\omega \models x$ means that $\exists X \in \mathcal{X}$ such that $\omega[X] = x$.

Definition 1.1 (Conditional preference network structure) *A Conditional preference network structure $\mathcal{P} = \langle \mathcal{G}, CPT \rangle$ is composed of two components:*

(i) a Directed Acyclic Graph (DAG) $\mathcal{G} = (\mathcal{X}, E)$ where $\mathcal{X} = \{X_1, \ldots, X_N\}$ is a set of N decision variables and E is the set of arcs representing preference dependencies between them;

[3] $x_i^* = x_i$ or $x_i^* = \bar{x_i}$

(ii) a set of conditional preference tables $CPT = \{CPT(X_1), \ldots, CPT(X_N)\}$ *where* $CPT(X_i)$ *is the local table attached to* X_i *and composed of statements of the form* $u_i : X_i \succ \neg X_i$*, such that* $X_i = x_i$ *or* $X_i = \bar{x}_i$*, that expresses a strict total order* $\succ$ *over values of* X_i *in the context of each instance* u_i *in* $\underline{U_{X_i}}$*.*

A conditional preference network structure makes it easy to find the optimal and worst outcomes which respectively correspond to configurations having all of their variables put at their best, resp. worst, assignments in the context of their parents. Finding these configurations can easily be done graphically by sweeping though the network from root to leaves and associating to each node its preferred resp. worst value. The complexity of the optimisation query is linear at the number of decision variables, it is the same complexity for finding the worst configuration.

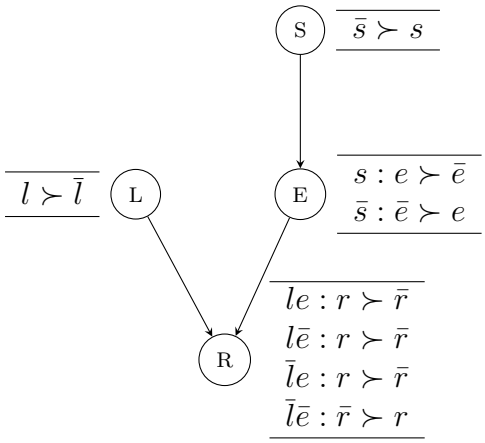


Figure 1.1: Conditional preference network structure

Example 1.1 *Let us consider the following specifications expressed by a user:*

- *I'm confused about buying or renting a car. I am more keen on making long trips* ($\bar{s}$) *rather than short ones* (s) *i.e.* $\bar{s} \succ s$.
- *I prefer to drive a luxury car* (l) *rather than a modest one* ($\bar{l}$) *i.e.* $l \succ \bar{l}$.
- *Even though vehicles with electric propulsion systems are easier on the environment, if I am traveling a long distance* ($\bar{s}$)*, I prefer a gasoline car* ($\bar{e}$)*, since finding a charging station and recharging is often difficult and may take a while i.e.* $\bar{s} : \bar{e} \succ e$*. From the other hand, if I am traveling short distances* (s)*, I prefer to drive an electric engine* (e) *because its maintenance requirements are low and electricity is cheaper than gasoline i.e.* $s : e \succ \bar{e}$*.*

- *My preferences on car ownership are conditional. If it is a luxury car (l), whatever is its propulsion system (e or $\bar{e}$), I'd prefer to rent (r) rather to purchase ($\bar{r}$) the car mainly because of its expensiveness i.e. $le : r \succ \bar{r}$ and $l\bar{e} : r \succ \bar{r}$. If it is a modest vehicle ($\bar{l}$) with an electric motor (e), I still prefer to rent rather than buy, mainly to avoid maintenance fees i.e. $\bar{l}e : r \succ \bar{r}$. However, if the vehicle is modest ($\bar{l}$) and equipped with a gasoline motor ($\bar{e}$), I prefer to buy since the car price is still acceptable and spare parts are cheaper than electric ones i.e. $\bar{l}\bar{e} : \bar{r} \succ r$.*

This example involves four decision variables, S: distance, L: category, E: propulsion system and R: ownership, s.t. $\underline{S} = \{s, \bar{s}\}$, $\underline{L} = \{l, \bar{l}\}$, $\underline{E} = \{e, \bar{e}\}$ and $\underline{R} = \{r, \bar{r}\}$. The vehicle category L and distance trip S are not conditioned by other variables. The preferences over the car motor E is conditioned by the trip distance S and owning or renting the car R depends of its category L and propulsion system E. The conditional preference network structure is given by Figure 1.1.

The set Ω contains $2^4 = 16$ configurations, i.e., $\Omega = \{\omega_0 = sler, \omega_1 = sle\bar{r}, \omega_2 = sl\bar{e}r, \omega_3 = sl\bar{e}\bar{r}, \omega_4 = s\bar{l}er, \omega_5 = s\bar{l}e\bar{r}, \omega_6 = s\bar{l}\bar{e}r, \omega_7 = s\bar{l}\bar{e}\bar{r}, \omega_8 = \bar{s}ler, \omega_9 = \bar{s}le\bar{r}, \omega_{10} = \bar{s}l\bar{e}r, \omega_{11} = \bar{s}l\bar{e}\bar{r}, \omega_{12} = \bar{s}\bar{l}er, \omega_{13} = \bar{s}\bar{l}e\bar{r}, \omega_{14} = \bar{s}\bar{l}\bar{e}r, \omega_{15} = \bar{s}\bar{l}\bar{e}\bar{r}\}$. The optimal solution is $\omega_{10} = \bar{s}l\bar{e}r$ and the worst one is $\omega_6 = s\bar{l}\bar{e}r$.

1.2.1 Pareto semantic for ordering configurations

Each decision variable in $\mathcal{X}$ has a polarity that describes its value, it can take either $(+)$ for the good or preferred assignment or $(-)$ for the bad or rejected one. A configuration ω can thus be described by a *quality* vector composed of N symbols ρ_i such that $i = [1, N]$ and $\rho_i \in \{+, -\}$. Given the preference statement $u : X \succ \neg X$, the good assignment corresponds to the preferred value X and the bad one corresponds to the rejected value $\neg X$. A natural way of ranking configurations is to say that $\omega \succ \omega'$ if for all decision variables ω is as good as ω' and at least one decision variable such that $\omega[X] \succ \omega'[X]$. ω defines the *dominating* configuration and ω' the *dominated* one. This ranking corresponds to the *Pareto* order on Boolean variables (see Definition 1.2). It permits to entail a partial order on configurations leaving some outcomes incomparable. This incomparability case happens when for the pair of solutions (ω, ω'), $\exists\{X_i, X_j\} \in \mathcal{X}$ such that $\omega[X_i] \succ \omega'[X_i]$ and $\omega'[X_j] \succ \omega[X_j]$.

Definition 1.2 (Pareto) *$\forall \omega \neq \omega' \in \Omega$ associated to the distinct quality vectors $\vec{\omega} = (\rho_1, \ldots, \rho_N)$ and $\vec{\omega'} = (\rho'_1, \ldots, \rho'_N)$ such that ρ_i is the polarity of variable X_i for $i =$*

$[1, N]$ *and* $\rho_i, \rho'_i \in \{+, -\}$, *then* $\omega \succ_{Pareto} \omega'$ *iff* $\forall i = [1, N]$

(i) either $\rho_i = \rho'_i$ *or* $\rho_i = +$ *and* $\rho'_i = -$;

(ii) for some $\ell, \rho_\ell = +$ *and* $\rho'_\ell = -$.

The Pareto order can be depicted by a directed graph (see Definition 1.3) such that nodes correspond to configurations attached to quality vectors and an arc from ω to ω' reflects a dominance relation in favor to ω.

Definition 1.3 (Pareto graph) *Given a conditional preference network with N decision variables, a Pareto graph is a DAG structure $< \mathcal{N}, E >$ such that $\mathcal{N} = \{W_0, \ldots, W_{2^N-1}\}$ is the set of nodes and E is the set of arcs that connects them.*

- *Each node W_i is composed of the configuration reference ω_i, its complete instantiation $x_1^* x_2^* \ldots x_N^*$ and its associated quality vector $\vec{\omega_i} = (\rho_1, \rho_2, \ldots, \rho_N)$;*
- *An arc $W_i \rightarrow W_j$ means that ω_i Pareto dominates ω_j ($\omega_i \succ_{Pareto} \omega_j$).*

Example 1.2 *Let us consider again network in Figure 1.1. Using Pareto semantic, the ordering on configurations of Ω is presented by the graph in Figure 1.2. The outcome ω_1, associated to the quality vector $(- + +-)$, is dominated by the outcome ω_0, associated to the quality vector $(-+++)$, because, all other variables equally valued, $\omega_0[R] \succ \omega_1[R]$.*

1.2.2 Cardinality order

Another natural way of ranking configurations is by considering the number of its variables put in their least preferred value[4], say $Card(.)$. The larger $Card(\omega)$, the worst the configuration ω. Thus, each configuration ω can be associated with a cardinality degree $Card(\omega)$ that describes its preference degree with regard to the user's specifications. Indeed, a configuration ω is preferred to ω based on the cardinality order, simply $\omega \succ_{Card} \omega'$, if $Card(\omega) < Card(\omega')$, respectively if the number of variables with polarity $(+)$ of ω is greater than that of ω. Unlike the Pareto order which

[4]Note that the least preferred value of a variable X_i refers here to the violated preference attached to X_i in the context of X_i's parent assignment.

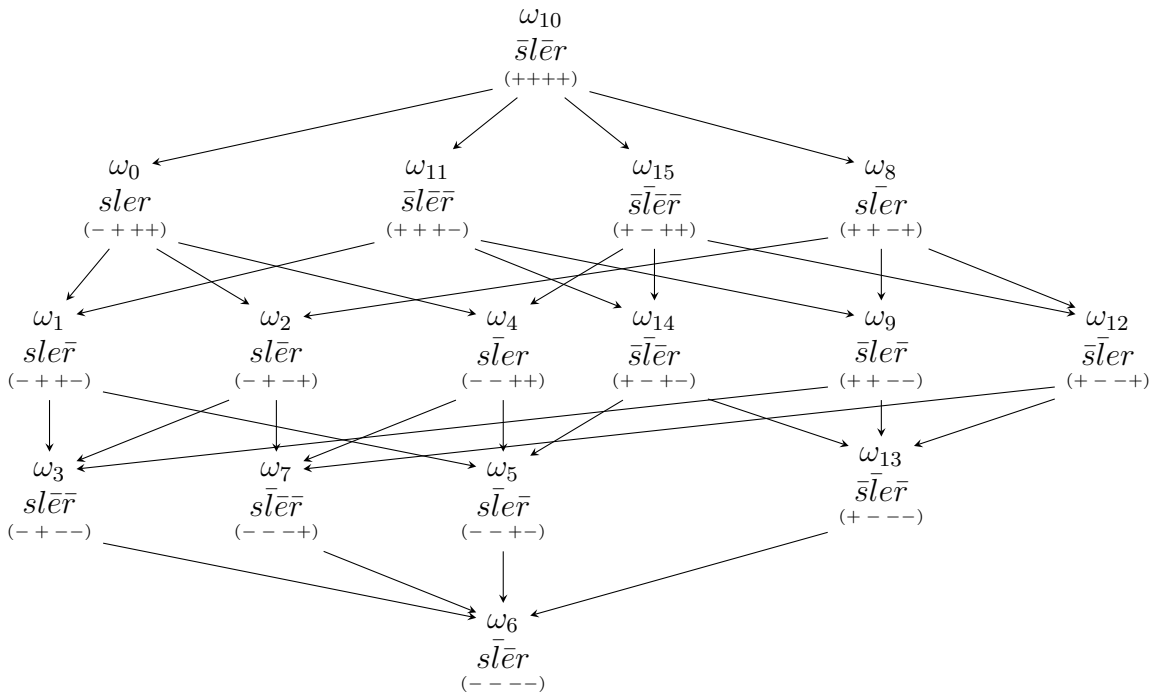


Figure 1.2: Pareto graph relative to network in Figure 1.1

produces a partial order over Ω, the cardinality order yields a total pre-order. Actually, The Pareto order states that if $\omega \succ_{Pareto} \omega'$ then, every variable polarity else being equal, there should exist at least one variable $X \in \mathcal{X}$ with a polarity equal to $(+)$ in $\vec{\omega}$ and equal to $(-)$ in $\vec{\omega'}$. If $\omega \succ_{Pareto} \omega'$ then $Card(\omega) < Card(\omega')$. This means that the cardinality order refines the Pareto order, more formally, $\forall(\omega, \omega') \in \Omega$, $\omega \succ_{Pareto} \omega' \Rightarrow Card(\omega) < Card(\omega')$. Obviously, the reverse entailment is wrong. If $\omega \succ_{Card} \omega'$, the Pareto order may fail to capture this relation, ω and ω' being incomparable. For instance, let $\vec{\omega} = (+ + +-)$ and $\vec{\omega'} = (- + -+)$, then $Card(\omega') = 2 > Card(\omega) = 1$ which means that $\omega \succ_{Card} \omega'$. However, ω and ω' are incomparable based on Pareto order.

Example 1.3 *Consider the network structure in Figure 1.1. Table 1.1 associates to each configuration in Ω the set of its violated decision variables and its cardinality. We depict the inferred ordering by a well-ordered partition of $N+1$ layers, each composed of equally preferred configurations having the same cardinality. See Table 1.2 for details. The first layer encompasses the best outcome while the last one corresponds to the worst. Configurations of a given layer are preferred to those of the next layer.*

Ω	Violated variables set	$Card(\omega)$
$\omega_0 = sler$	$\{S\}$	1
$\omega_1 = sle\bar{r}$	$\{S, R\}$	2
$\omega_2 = sl\bar{e}r$	$\{S, E\}$	2
$\omega_3 = sl\bar{e}\bar{r}$	$\{S, E, R\}$	3
$\omega_4 = s\bar{l}er$	$\{S, L\}$	2
$\omega_5 = s\bar{l}e\bar{r}$	$\{S, L, R\}$	3
$\omega_6 = s\bar{l}\bar{e}r$	$\{S, L, E, R\}$	4
$\omega_7 = s\bar{l}\bar{e}\bar{r}$	$\{S, L, E\}$	3
$\omega_8 = \bar{s}ler$	$\{E\}$	1
$\omega_9 = \bar{s}le\bar{r}$	$\{E, R\}$	2
$\omega_{10} = \bar{s}l\bar{e}r$	$\emptyset$	0
$\omega_{11} = \bar{s}l\bar{e}\bar{r}$	$\{R\}$	1
$\omega_{12} = \bar{s}\bar{l}er$	$\{L, E\}$	1
$\omega_{13} = \bar{s}\bar{l}e\bar{r}$	$\{L, E, R\}$	3
$\omega_{14} = \bar{s}\bar{l}\bar{e}r$	$\{L, R\}$	2
$\omega_{15} = \bar{s}\bar{l}\bar{e}\bar{r}$	$\{L\}$	1

Table 1.1: Configurations of network in Figure 1.1 and their cardinality degree

$Card(\omega)$	Partition
0	$\{\omega_{10}\}$
1	$\{\omega_0, \omega_8, \omega_{11}, \omega_{15}\}$
2	$\{\omega_1, \omega_2, \omega_4, \omega_9, \omega_{12}, \omega_{14}\}$
3	$\{\omega_3, \omega_5, \omega_7, \omega_{13}\}$
4	$\{\omega_6\}$

Table 1.2: Cardinality order relative to network in Figure 1.1

1.3 CP-nets

Conditional Preference Networks or simply *CP-nets* [Boutilier et al., 1999] were introduced to compactly represent conditional preferences of a user over multivariate decision problems. They initially were inspired from Bayesian networks, but unlike the latter, they offer a tool for representing preferences in a qualitative manner. Each user specification encoded by CP-nets define a ranking over values of single variables in the context of fixed assignments of features that influence them. Thus, a CP-net is composed of two components: a graphical and an informational structure. Indeed, decision variables are depicted by nodes connected with each other by means of arcs that reflect dependencies between them. Each node is associated with a local table that contains an ordinal relation on values of the variable in question. A CP-net $\mathcal{C}$ is thus composed of a directed graph that involves a set of decision variables depicted by nodes that are connected by means of directed edges. Each arc from a node X to

a node $\bar{x}$ reflects the dependency of $\bar{x}$ in regards of X. If we restrict features to be binary-valued, each decision variable $X \in \mathcal{X}$ is associated with a collection of statements of the form $u : X \succ \neg X$ where $u \in \underline{U_X}$ and $\{x, \bar{x}\} \in \underline{X}$. Such a statement is interpreted by "In the context of u, I strictly prefer X to $\neg X$". The collection of conditional statements associated with each node forms the second component of a CP-net namely *Conditional Preference Tables* (CPT).

In order to detail semantics of CP-nets and give a complete definition of the model, we first need to define some basic notions.

Definition 1.4 (Worsening flip) *Let ω, ω' be two configurations in Ω that differ by a single variable assignment over $X \in \mathcal{X}$. X is called the* swapped variable. *There exists a* worsening flip *from ω to ω' if and only if there exists a statement $u : X \succ \neg X$ where $X = x$ or $X = \bar{x}$ such that*

(i) $\omega, \omega' \models u$,

(ii) $\omega \models X$,

(iii) $\omega' \models \neg X$,

(iv) $\omega, \omega' \models y$ *such that* $y \in \underline{Y}$ *and* $\mathcal{Y} = \mathcal{X} \setminus \{\{X\} \cup U_X\}$.

Definition 1.5 (Worsening flip sequence) *Let ω, ω' be two configurations in Ω. There exists a* worsening flip sequence *from ω to ω' if and only if there exists a sequence $(\omega, \omega_1, \ldots, \omega_K, \omega')$ such that $\forall k \in \{1, \ldots, K-1\}$ there exists a worsening flip from ω_k to ω_{k+1}.*

CP-nets are based on the *ceteris paribus* independence property, which enables the preference over values of X in the context of a fixed instantiation of its parents U_X to be extended to complete configurations assuming that the remaining set of variables $\mathcal{Y} = \mathcal{X} \setminus \{X\} \setminus U_X$ takes the same value. In other words, this assumption enables us to compare a pair of configurations that differ by single flip value on the swapped variable *every thing else being equal*.

Definition 1.6 (Ceteris paribus) *Let $\mathcal{Y}$ be a set of variables s.t. $\mathcal{Y} = \mathcal{X} \setminus \{X\} \setminus U_X$. The* ceteris paribus *assumption states that $\omega = yux$ is preferred to $\omega' = yu\bar{x}$ s.t. $y \in \underline{\mathcal{Y}}$ and $u \in \underline{U_X}$ if and only if one of the following conditions are satisfied:*

(i) there exists a worsening flip from ω to ω'

(ii) ω' is obtained from ω via a worsening flip sequence.

In fact, a CP-net[5] can be defined as being a combination of a Conditional preference network structure $\mathcal{P}$ and an independence assumption, namely ceteris paribus which allows to construct a dominance relation between configurations by means of a transitive closure of the worsening flip relation. Indeed, given a set of variables $\mathcal{X}$, a CP-net has the same graphical structure as defined in section 1.2 which consists of a directed acyclic graph expressing dependency between variables and a set of conditional preference tables. Note that the graphical structure of a CP-net could be cyclic. This case can yield to an unsatisfiable ranking on configurations ($\exists \omega, \omega' \in \Omega$ with $\omega \neq \omega'$ such that $\omega \succ \omega'$ and $\omega' \succ \omega$). This case is discussed later.

Definition 1.7 (CP-net) *A Conditional Preference network $\mathcal{C} = \langle \mathcal{G}, CPT \rangle$, denoted by CP-net, is a Conditional Preference network as defined in Definition 1.1 that obeys to the ceteris paribus preferential independence property.*

Given a configuration ω we can subsequently either improve a flip on one of its variable's value or worsen it to reach another configuration ω'. Given statements in conditional tables of a CP-net $\mathcal{C}$ along with the ceteris paribus assumption, a sequence of improving flips from one configuration to another confirms that these configurations are comparable. Accordingly, they are incomparable if and only if there exists no flipping sequence between them. In other words, ω is preferred to ω' ($\omega \succ \omega'$), if and only if there exists a *improving flip sequence* from ω to ω'. Every *improving flip* can be depicted by an arc from ω to ω' expressing that $\omega \succ_{\mathcal{C}} \omega'$. The collection of flip sequences between all pairs of configurations in Ω forms a DAG called *induced preference graph* composed of a unique root node corresponding to the best configuration and a single leaf node consisting of a sink corresponding to the worst one. Mind that one could also start by the worst configuration as a root and subsequently deteriorate values of variables. The obtained ordering remains unchanged.

The optimal configuration is found by assigning the best value to each decision variable in the context or value of variables from which they depend. In a similar manner, the worst configuration is obtained by considering the least preferred value of each variable in the context of its parents. Finding the dominance relation between all possible solutions is performed by means of the transitive closure on worsening flips.

[5]The abbreviation CP-nets refer to Conditional Preference networks introduced by [Boutilier et al., 2004]. To avoid confusion with Conditional Preference network structures described in Section 1.2, we use $\mathcal{C}$ as a reference to the former model and $\mathcal{P}$ to the latter.

In most cases, the induced CP-net ordering corresponds to a strict partial order since some pairs of configurations may remain incomparable.

Example 1.4 *Let us re-consider Example 1.1 expressing preferences about renting or buying a car. We consider network in Figure 1.1 which has the same structure of a CP-net. Configurations* $\omega_0 = sler$ *and* $\omega_1 = sle\bar{r}$ *differ by the value of node* R *which corresponds to the swapped variable. There exists a* worsening flip *from* ω_0 *to* ω_1 *since they both model the same assignment* le *of* R*'s parents:* $U_R = \{L, E\}$*, and also the same assignment* s *for the remaining variables* $\mathcal{Y} = \mathcal{X} \setminus \{R\} \setminus U_R = S$*. Thus,* $\omega_0 = sler$ *is preferred to* $\omega_1 = sle\bar{r}$ *according to the ceteris paribus principle since* $le : r \succ \bar{r}$*.*

Let us now consider the pair of configurations $\omega_0 = sler$ *and* $\omega_7 = s\bar{l}\bar{e}\bar{r}$ *that differ by the value of more than one decision variable. A* worsening flip sequence *from* ω_0 *to* ω_7 *can be constructed by subsequently considering the worsening flips :* $\omega_0 = sler \rightarrow \omega_1 = sle\bar{r}$*,* $\omega_1 = sle\bar{r} \rightarrow \omega_3 = sl\bar{e}\bar{r}$ *and finally* $\omega_3 = sl\bar{e}\bar{r} \rightarrow \omega_7 = s\bar{l}\bar{e}\bar{r}$*. Hence,* $\omega_0 = sler$ *is preferred to* $\omega_7 = s\bar{l}\bar{e}\bar{r}$ *w.r.t. the ceteris paribus property since there exists a* worsening flip sequence *from a configuration to the other.*

The set of worsening flip sequences between all pairs of configurations is depicted by Figure 1.3. The optimal configuration ω_{10}*, which assigns to each variable its preferred value, is in top of the graph while the worst one* ω_6 *is in the bottom. We count* 21 worsening flips *between pairs of configurations. The induced order is partial. For instance,* ω_9 *and* ω_{13} *are incomparable since there exists no worsening flip sequence between them.*

1.3.1 Ceteris paribus order and Pareto order

Let $\mathcal{X}'(\omega, \omega') \subseteq \mathcal{X}$ be the set of variables on which configurations ω and ω' differ. Given a CP-net $\mathcal{C}$, we say that ω *locally dominates* ω', simply $\omega \succ_{LD} \omega'$ if and only if for all variables X in $\mathcal{X}'$, $\omega[X] \succ \omega'[X]$. Wilson et al. [Wilson et al., 2019] have proved that the ceteris paribus order $\succ_{\mathcal{C}}$ is nothing more than the transitive closure of the order relation $\succ_{LD}$ on $\mathcal{C}$. This is due to the fact that a worsening flip sequence from ω to ω' implies a *local dominance* from ω to ω'. Thus, the local dominance relations are included in the induced ceteris paribus relations, formally $\succ_{LD} \subseteq \succ_{\mathcal{C}}$.

Definition 1.8 (Local dominance) *Let* $\mathcal{C}$ *be a CP-net, the subset* $\mathcal{X}'(\omega, \omega') \subseteq \mathcal{X}$ *encompasses variables on which* ω *and* ω' *differ.* ω locally dominates ω'*, formally* $\omega \succ_{LD} \omega'$ *iff* $\forall X \in \mathcal{X}'$, $\omega[X] \succ \omega'[X]$.

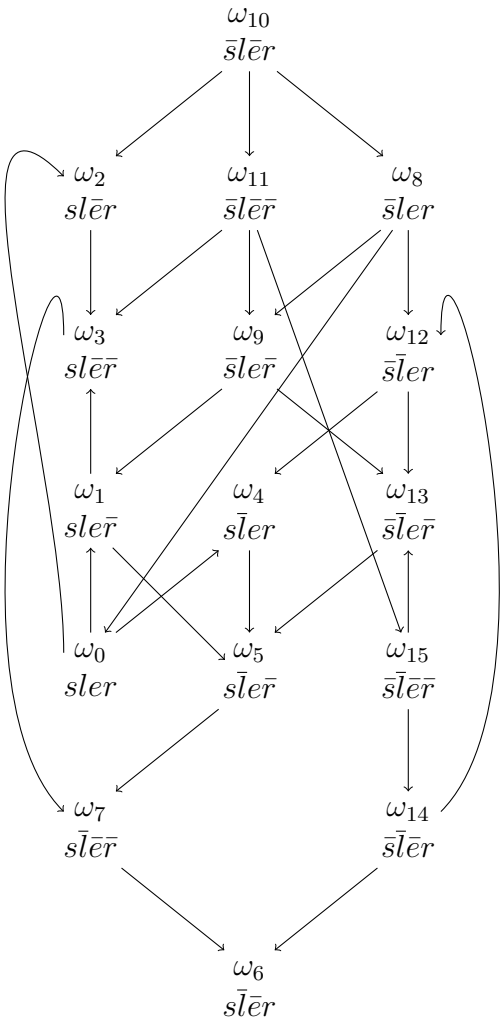


Figure 1.3: Induced worsening flip graph of CP-net of Example 1.4

Example 1.5 *Let us pursue with same example in 1.1. Consider configurations* $\omega_{11} = \bar{s}l\bar{e}\bar{r}$ *and* $\omega_5 = s\bar{l}e\bar{r}$. *they differ on values of variables* $\mathcal{X}' = \{S, L, E\}$. ω_{11} *locally dominates* ω_5 *since, w.r.t local tables in CPT of CP-net depicted by the structure in Figure 1.1,* $\omega_{11}[S] = \bar{s} \succ \omega_5[S] = s$, $\omega_{11}[L] = l \succ \omega_5[L] = \bar{l}$ *and* $\omega_{11}[E] \succ \omega_5[E]$. *Formally* $\omega_{11} \succ_{LD} \omega_5$. *We can also check that* $\omega_5 \succ_{LD} \omega_7$ *since for each variable* X *in* $\mathcal{X}'(\omega_5, \omega_7) = \{E\}$ *the statement* $\omega_5[X] \succ \omega_7[X]$ *is verified. Thus, we can conclude that* $\omega_{11} \succ_{LD} \omega_7$ *by transitivity. This order relation is entailed by CP-net semantics since there exists a worsening flip sequence from* ω_{11} *to* ω_7 *(see Figure 1.3). One can check that all order relations induced by the transitive closure of local dominance relation are entailed by the ceteris paribus assumption and conversely.*

Wilson et al. [Wilson et al., 2019] have then proved that the Pareto order is contained in $\succ_{LD}$, which means that if $\omega \succ_{Pareto} \omega'$ then $\omega \succ_{LD} \omega'$. Indeed, considering a CP-net $\mathcal{C}$, if $\omega \succ_{Pareto} \omega'$ then for all variables $X \in \mathcal{X}'$, $\omega[X] \succ \omega'[X]$ all else be-

ing equal. This claim corresponds to the definition of a local dominance relation $\succ_{LD}$. Thus, $\succ_{Pareto} \subseteq \succ_{LD} \subseteq \succ_{\mathcal{C}}$ [Wilson et al., 2019], which stipulates that the ceteris paribus dominance relation refines the Pareto order with the local dominance relation being between the two. This proposition is also true for multi-valued decision variables.

1.3.2 Implicit importance given by ceteris paribus

[Boutilier et al., 2004] acknowledge the fact that

"Violating the preference constraints for a parent variable is less preferred than violating the preference constraints for any of its children".[Boutilier et al., 2004]

This can be seen in a very simple example like in Figure 1.4. It is clear that this effect of *ceteris paribus* is debatable since it may happen that preferences associated with child node are more important than those associated with father nodes.

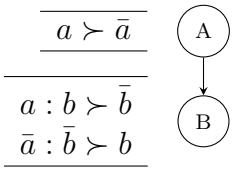


Figure 1.4: Example of CP-net

Example 1.6 *The fact that the ceteris paribus assumption implies that parent preferences have priority to child ones is easily detected by a CP-net with two nodes. Consider CP-net $\mathcal{C}$ in Figure 1.4 and its induced order: $ab \succ_{\mathcal{C}} a\bar{b} \succ_{\mathcal{C}} \bar{a}\bar{b} \succ_{\mathcal{C}} \bar{a}b$. The configuration that violates the root node A, namely $\bar{a}\bar{b}$, is considered less satisfactory than the one that violates its child B, namely $a\bar{b}$. Thus, violating A is more penalizing than violating B, while the semantics of the CP-net does not explicitly specify an importance relation between them.*

The previous Example is minimalist but is sufficient for an illustration of the implicit relative priority allocated to parent nodes in CP-nets. Even after admitting that semantics of CP-nets give more priority to parents nodes over their descendants, the ceteris paribus order remains questionable, and open to criticism. By means of the following example, inspired from [Dubois et al., 2013], we bring the light on some doubtful incompatibilities entailed by ceteris paribus semantic using a CP-net that involves variables depending from multiple father nodes.

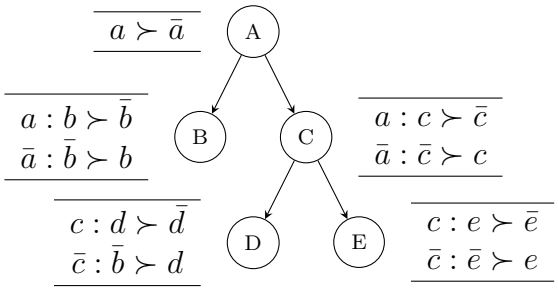


Figure 1.5: Example of CP-net

Example 1.7 *Consider CP-net $\mathcal{C}$ in Figure 1.5 and configurations $\omega = ab\bar{c}d\bar{e}$ and $\omega' = a\bar{b}\bar{c}\bar{d}e$. Both ω and ω' share same assignments on parent nodes, namely $\omega[A] = \omega'[A] = a$ and $\omega[C] = \omega'[C] = \bar{c}$. In the context of a, ω assigns a more preferred value to B than ω', i.e., $a : b \succ \bar{b}$. Similarly, In the context of $\bar{c}$, ω assigns a more preferred value to E than ω', i.e., $\bar{c} : \bar{e} \succ e$. Thus, CP-nets semantics never imply that $\omega' \succ_{\mathcal{C}} \omega$ and one can check that there exist no worsening flip sequence from ω' to ω. The CP-net semantics entail the preference relations: $\omega \succ_{\mathcal{C}} \omega' \succ_{\mathcal{C}} \omega''$ and $\omega \succ_{\mathcal{C}} \omega'''$, such that $\omega = abcd\bar{e}$, $\omega' = a\bar{b}\bar{c}\bar{d}\bar{e}$, $\omega'' = \bar{a}\bar{b}\bar{c}\bar{d}\bar{e}$, $\omega''' = ab\bar{c}\bar{d}e$. We divide $\mathcal{X}$ into three subsets: parents $\{A\}$, children $\{B, C\}$ and grand-children $\{D, E\}$. From the relation $\omega \succ_{\mathcal{C}} \omega'$, we notice that violating constraints of children nodes is more discriminant than violating constraints of grand-children, since violated nodes of ω are D and E, and those violated by ω' are B and C. Furthermore, violating multiple child and grand-children nodes is less penalizing than violating one parent node. This can be exemplified by the relation $\omega \succ_{\mathcal{C}} \omega' \succ_{\mathcal{C}} \omega''$, where ω'' violates A. However, it is troublesome that the ceteris paribus is not able to rank order the configuration ω''' with respect to ω' and ω'', even though ω'' violates a parent node, ω' violates a child node and ω''' violates a child and a grandchild node.*

In the previous example 1.7, we can see that one configuration can be ranked by ceteris paribus as preferred over another even if they violate the same number of variables. In the next section, we discuss a ordering semantics that considers the number of variable violations as a ranking strategy.

1.3.3 Ceteris paribus order vs. cardinality order

As previously explained, given a CP-net, we can use what is called a cardinality order on the set of alternative Ω. Each configuration is ranked based on the number of its violated variables. The preference property of CP-nets is able to produce a ranking between a pair of configurations that both violate the same number of variables. How-

ever, given a a configuration ω that violates less variables than ω', i.e., $\omega \succ_{Card} \omega'$, it is not necessary the case that the ceteris paribus ordering acknowledges it.

Example 1.8 *Consider network in Figure 1.1. Configuration* $\omega_3 = sl\bar{e}\bar{r}$ *and* $\omega_9 = \bar{s}le\bar{r}$ *share the same number of violated variables but the preferential property of CP-nets succeeds to rank order them. Indeed, the configuration* ω_3 *violates the set of variables* $\{S, R\}$ *and* ω_9 *violates* $\{E, R\}$*. Figure 1.3 confirms that there exists a worsening flip sequence between them yielding the relation* $\omega_9 \succ_{\mathcal{C}} \omega_3$*. The configuration* $\omega_{13} = \bar{s}\bar{l}e\bar{r}$ *has a cardinality equals to* 3 *for the violated nodes L,E and R. Semantics of CP-nets fail to rank order* ω_3 *and* ω_{13} *even though the latter violates more variables than the former.*

In general, the CP-net relation is in accordance with Pareto relation, but not with the cardinality order. In fact, the importance accorded by CP-net semantics in favor of parents are quite strong since violating a parent node can be more penalizing than violating multiple children. This is in contradiction with the cardinality order since the ceteris paribus ranks a configuration that violates multiple nodes as being preferred to a configuration that violates a single node. Thus two configurations can be compared by ceteris paribus and by cardinality orders in an opposite ways. Example 1.9 provides an illustration case.

Example 1.9 *Consider CP-net* $\mathcal{C}$ *in Figure 1.6 with one root variable, namely A, that has two child nodes, namely B and C. Consider configurations (*$\omega = a\bar{b}\bar{c}$, $\omega' = \bar{a}\bar{b}\bar{c}$*) that differ on a single variable flip on A.* $\omega \succ_{\mathcal{C}} \omega'$ *since there exists a preference statement in* $CPT(A)$ *stipulating that* $a \succ \bar{a}$*. The pair of configurations are respectively associated with quality vectors* $\vec{\omega} = (+--)$ *and* $\vec{\omega'} = (-++)$*. Since* ω' *violates less variables than* ω*, then, the cardinality order confirms that* $\omega' \succ \omega$*, which is in contradiction with the CP-net order.*

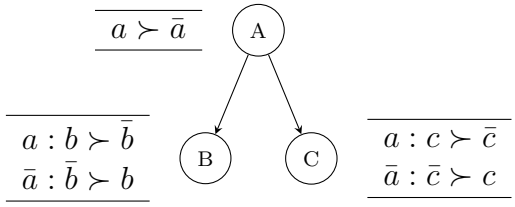


Figure 1.6: Example of CP-net

1.3.4 Satisfiability of a CP-net

The satisfiability of a CP-net is actually related to its structure. The graphical component of a CP-net can either be cyclic or acyclic. In this section, in order to explain some concepts, we consider the both graphical structures. Most research works consider acyclic CP-nets since the acyclicity characteristic confers to the network the property of being satisfiable, which is not the case for cyclic networks [Boutilier et al., 2004].

Definition 1.9 (Satisfiability of a CP-net) *A CP-net $\mathcal{C}$ is satisfiable by an order $\succ_{\mathcal{C}}$ iff it satisfies each of the conditional preference statements in the collection of local tables CPT using the ceteris paribus assumption.*

Let $\succ'_{\mathcal{C}}$ and $\succ''_{\mathcal{C}}$ be two orderings that satisfy $\mathcal{C}$. If $\succ'_{\mathcal{C}}$ ranks $\omega \succ'_{\mathcal{C}} \omega'$, and $\succ''_{\mathcal{C}}$ ranks $\omega' \succ''_{\mathcal{C}} \omega''$ then the assertion $\omega \succ_{\mathcal{C}} \omega''$ should be valid for all orderings that satisfy $\mathcal{C}$. This means that all induced partial orderings that satisfies $\mathcal{C}$ are transitive. Note that not all cyclic CP-nets yield to an inconsistent ordering over configurations. In fact, Domshlak and Brafman have studied the consistency of binary-valued cyclic CP-nets and have identified a wide class of satisfiable networks [Domshlak and Brafman, 2002]. The following example illustrates the problem encountered by cyclic graphs, where one may have a worsening path that is cyclic.

Example 1.10 *Consider network $\mathcal{C}$ in Figure 1.7(a). The graph structure is cyclic since the preference over variable A depends on B and vise versa. The first specification $b : a \succ \bar{a}$ in $CPT(A)$ induces the preference relation $ab \succ_{\mathcal{C}} \bar{a}b$ which is represented by an arc $ab \rightarrow \bar{a}b$ in the right-most worsening flip graph in Figure 1.7 (b). The second specification $\bar{b} : \bar{a} \succ a$ in $CPT(A)$ induces the preference relation $\bar{a}\bar{b} \succ_{\mathcal{C}} a\bar{b}$ which is depicted by an arc $\bar{a}\bar{b} \rightarrow a\bar{b}$. Following the same reasoning, $CPT(B)$ entails the order rankings $a\bar{b} \succ_{\mathcal{C}} ab$ and $\bar{a}b \succ_{\mathcal{C}} \bar{a}\bar{b}$. The induced worsening flip graph corresponding to $\mathcal{C}$ is cyclic and entails an inconsistent ranking on configurations, namely $ab \succ_{\mathcal{C}} \bar{a}b \succ_{\mathcal{C}} \bar{a}\bar{b} \succ_{\mathcal{C}} a\bar{b} \succ_{\mathcal{C}} ab$ (see Figure 1.7(b)) . Note that if we modify the local table $CPT(B)$ to hold statements $a : b \succ \bar{b}$ and $\bar{a} : \bar{b} \succ b$ then we end up with a consistent partial order namely $\{ab, \bar{a}\bar{b}\} \succ_{\mathcal{C}} \bar{a}b$, $\{ab, \bar{a}\bar{b}\} \succ_{\mathcal{C}} a\bar{b}$.*

Indifference and satisfiability

Most CP-nets are based on generalized statements that define a strict total order relation over instantiation of variables in question. However, CP-nets offer to the

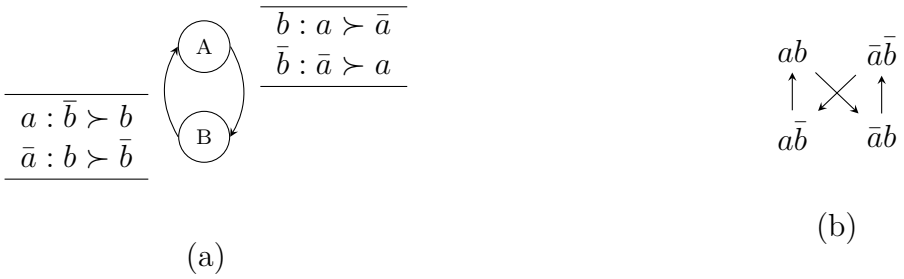


Figure 1.7: Cyclic CP-net (a) and its induced worsening flip graph (b)

user the flexibility to express indifference between values of a given variable, e.g., $u : x \sim \bar{x}$ which is interpreted by the claim "In the context of u, x and $\bar{x}$ are equally preferred". The preference order induced by CP-nets that allow indifference between variables values is not always consistent [Boutilier et al., 2004]. In fact, if indifference is allowed, then the CP-net must obey a precise restriction in order to generate a consistent ordering. Consider a CP-net $\mathcal{C}$ composed of nodes $\mathcal{X} = \{U_X, X, Y, Ch_X\}$ where U_X denotes parents of X, Ch_X denotes any children of X and Y is the set of parents of Ch_X excluding X. Suppose that for a given $u \in \underline{U_X}$, we have $u : x \sim \bar{x}$. Then, $\mathcal{C}$ is satisfiable as long as the following technical condition holds, for a fixed value y in $\underline{Y}$, the preference over values of Ch_X is the same e.g $xy : c \succ \bar{c}$ for $x \in \underline{X}$ and $\{c, \bar{c}\} \in \underline{Ch_X}$. For the remainder of the manuscript, we will only assume the case where each generalized preference statement composing a CP-net holds a strict total order relation over values of the variable in question. The following example shows the problem created by indifference when the above technical condition does not hold.

To avoid dealing with inconsistent orderings, we prohibit indifference and we will only consider acylic structures for the remaining of this work.

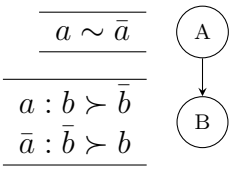


Figure 1.8: Example of CP-net where indifference is allowed

Example 1.11 *Consider CP-net $\mathcal{C}$ in Figure 1.8. Statements in $CPT(B)$ entail the following order rankings: $ab \succ_{\mathcal{C}} a\bar{b}$ and $\bar{a}\bar{b} \succ_{\mathcal{C}} \bar{a}b$. The indifference between values of A yield to an inconsistent ordering over configurations namely $ab \succ_{\mathcal{C}} a\bar{b} \succeq_{\mathcal{C}} \bar{a}b \succ_{\mathcal{C}} \bar{a}b \succeq_{\mathcal{C}} ab$, making the CP-net $\mathcal{C}$ unsatisfiable.*

1.3.5 Querying CP-nets

The ceteris paribus semantic of CP-nets allows to exploit information contained in conditional preference tables to either compare a pair of configurations or find the optimal outcome in accordance with the network. In the following, we introduce the most used queries on CP-nets, namely the *optimization query* for determining the best configuration and the *dominance query* for finding a preference relation between two given configurations.

Optimization Query

Given a CP-net $\mathcal{C}$, sweeping through its conditional preference tables while picking for each variable its preferred value in the context of parents assignment is a simple task that allows to determine the unique optimal solution in accordance with $\mathcal{C}$. This procedure exploits both ceteris paribus informational property along with the compact graphical modeling of preference statements to easily determine the best solution in time linear with respect to the number of alternative choices [Boutilier et al., 2004]. One can also be interested in finding the worst configuration which can be done by sweeping though the CP-net and setting each variable to its less preferred assignment given parents context.

The optimization query is outlined by Algorithm 1.1. Function *Search_context* (ρ, X_i) takes as parameters a partial configuration ρ and a node X_i and determines the assignment u of U_{X_i} such that $u \models \rho$. Function *Values_given_context* (X_i, u) return a pair of values $(pref, \overline{pref})$ corresponding to respectively the most and less preferred assignments of X_i. Similarly, Algorithm 1.1 can generate the worst configuration by modifying the instruction of the 10^{th} line of the algorithm and concatenate the result of previous iterations with the least preferred value of the variable in question namely $\overline{pref}$.

Example 1.12 *Let us continue with Example 1.4. Consider that the set $\mathcal{X}$ is ordered starting from root nodes to leaves, e.g., $\mathcal{X} = \{S, L, E, R\}$.*

- *In order to answer the* optimization query *"find the optimal configuration", we need to associate to each variable its preferred value. We begin by root nodes $\{S, L\}$ making $\omega_{opt} = \bar{s}l$, we then continue by node E that stipulates that in the context of $\bar{s} : \bar{e} \succ e$, making $\omega_{opt} = \bar{s}l\bar{e}$, to finally assign the value r to ω_{opt} since in $CPT(R)$ $l\bar{e} : r \succ \bar{r}$. The optimal configuration thus corresponds to $\omega_{opt} = \bar{s}l\bar{e}r$.*

Algorithm 1.1: Optimization query given a CP-net $\mathcal{C}$

Input: $\mathcal{C} = \langle \mathcal{G} = (T, E), CPT \rangle$
Output: ω_{opt}

```
T = table of nodes          /* Nodes in Ω ordered in a topological order */
for i = 1 to i = |T| do
    X ← T[i]
    if U_X ≠ ∅ then
        u ← Search_context (ω_opt, X)
    else
        u ← ∅
    end
    (pref, pref̄) ← Values_given_context (X, u)
    Concatenate (ω_opt, pref)
end
return ω_opt
```

- *Therefore, the worst case configuration is obtained by sweeping through nodes of the graph and assigning each variable its least preferred value. The worst configuration is* $s\bar{l}\bar{e}r$.

Complexity of optimization query Considering a partial configuration ρ, a variant of the optimization query consists on finding the configuration that models ρ and assigns to the non instantiated variables their best value. Given any topological order on decision variables, and similarly to the classical optimization query, finding the optimal configuration in the completions set of ρ such that $\omega \in \Omega$ is done by a forward sweep procedure by sweeping through the network from ancestors to descendant while assigning to each node its preferred value. The complexity of dominance query with enforced conjunctive constraints is also linear in the number of features [Boutilier et al., 2004].

Dominance Query

The second most interesting query with respect to this model is to figure out if a configurations dominates another with respect to the ceteris paribus assumption or not. This query is called the *preferential comparison between outcomes*, better known as the *dominance query*. Let $\omega, \omega' \in \Omega$ be two configurations, three situations are possible:

- ω dominates ω': $\omega \succ \omega'$,
- ω' dominates ω: $\omega' \succ \omega$,

- ω and ω' are incomparable: $\omega \bowtie \omega'$. This relation is deduced if and only if $\omega \not\succ \omega'$ and $\omega' \not\succ \omega$:

Complexity of dominance query The preferential independence property of CP-nets allows to compare pairs of configurations by simply consulting information depicted in the network structures. In binary-valued decision variables, the complexity of this query performed over the set of possible solutions is generally NP-complete. This complexity actually depends on the network structure. In acyclic CP-nets where the graph corresponds to a tree structure, the complexity of this query is quadratic in the number of variables. It is polynomial in the size of decision variables for polytrees. For directed-path singly connected DAGs [6] the complexity is NP-complete [Boutilier et al., 2004].

Figuring out if a configuration dominates another given a CP-net $\mathcal{C}$ is done by searching for a worsening flip sequence between them. Considering a configuration ω as a departure point, this can be seen as a planning problem for optimizing the value of each node $X \in \mathcal{X}$ until reaching the configuration ω' which represents the goal.

Indeed, each preference statement $u_i : X_i \succ \neg X_i$ is translated into a generalized statement of the form $u_i : x_i^j \succ x_i^{j+1}$ where x_i is the preferred value of X_i and $u_i \in \underline{U_{X_i}}$ (the algorithm in [Boutilier et al., 2004] is not restricted to Boolean variables). Each generalized statement is converted into a planning operator of the form $u_i \wedge x_i^j$ which defines the set of *Preconditions*. In order to perform any action, a proposition in the set of *Preconditions* must be satisfied. The set of *Postconditions* is obtained after performing two operations: *Delete list:* x_i^j and *Add list:* x_i^{j+1} which consists on removing x_i^j from propositions in the set of *Preconditions* and replace it by x_i^{j+1}. This corresponds to the action of worsening x_i^j to x_i^{j+1} in the context of u_i. Considering any comparison of the form $\omega \succ_{\mathcal{C}} \omega'$ between a pair of different configurations, the idea is to treat ω as the starting state and ω' as the goal. Thus, $\omega \succ_{\mathcal{C}} \omega'$ holds true if and only if the planning problem converges and can generate a plan. It is obvious to see that the problem solution corresponds to a set of worsening flips that compose a worsening flip sequence from ω to ω'. For binary-valued acyclic CP-nets, this problem is known to be PSPACE since it is reducible to a specific variant of STRIPS planning problems with unary operators which instantiate one variable at a time [Brafman and Domshlak, 2011].

Example 1.13 *Consider preference statements defined by local tables of network in Figure 1.1. Each statement of the form* $u : X \succ \neg X$ *is transformed into the operator or*

[6] Given a pair of nodes, there exists at most one path that connects them with each other.

precondition $u \wedge x_1$ *for* $u \wedge x$. *Statements of the considered example are thus transformed into the following set of planning operators :*
$\bar{s} \succ s \rightarrow \emptyset \wedge s^1$ *such that* $s^1 = \bar{s}$,
$l \succ \bar{l} \rightarrow \emptyset \wedge l^1$ *such that* $l^1 = l$,
$s : e \succ \bar{e} \rightarrow s \wedge e^1$ *such that* $e^1 = e$,
$\bar{s} : \bar{e} \succ e \rightarrow \bar{s} \wedge e^1$ *such that* $e^1 = \bar{e}$,
$\{le : r \succ \bar{r}, l\bar{e} : r \succ \bar{r}, \bar{l}e : r \succ \bar{r}\} \rightarrow \{le \wedge r^1, l\bar{e} \wedge r^1, \bar{l}e \wedge r^1\}$ *such that* $r^1 = r$,
$\bar{l}\bar{e} : \bar{r} \succ r \rightarrow \{\bar{l}\bar{e} \wedge r^1\}$ *such that* $r^1 = \bar{r}$.

We aim to find a plan that corresponds to a worsening flip sequence from $\omega_{11} = \bar{s}l\bar{e}\bar{r}$ *to* $\omega_5 = s\bar{l}e\bar{r}$. *The first step consists on worsening the value of* E. *We note that* $\omega_{11} \models \bar{s} \wedge \bar{e}$, *a proposition equivalent to the precondition* $\bar{s} \wedge e^1$ *such that* $e^1 = \bar{e}$. *We can thus change the value of variable* E *from* $e^1 = \bar{e}$ *to* $e^2 = e$. *This corresponds to a worsening flip on* E *resulting to the configuration* $\omega_9 = \bar{s}le\bar{r}$. *The second flip consists on modifying the value of* L *since* $\omega_9 \models l$ *which verifies the precondition* $\emptyset \wedge l^1$ *such that* $l^1 = l$. *This leads to attain the configuration* $\omega_{13} = \bar{s}\bar{l}e\bar{r}$. *The last step consists on worsening the value of* S *accordingly to the operator* $\emptyset \wedge s^1$ *such that* $s^1 = \bar{s}$. *We end up reaching the configuration* $\omega_5 = s\bar{l}e\bar{r}$. *A plan is thus found from* ω_{11} *to* ω_5 *which permits to conclude that* $\omega_{11} \succ \omega_5$ *according to preference specifications provided by the user. The corresponding worsening flip sequence is* $\omega_{11} = \bar{s}l\bar{e}\bar{r} \rightarrow \omega_9 = \bar{s}le\bar{r} \rightarrow \omega_{13} = \bar{s}\bar{l}e\bar{r} \rightarrow \omega_5 = s\bar{l}e\bar{r}$.

Ordering query

Another less exploited query named *ordering query* in the literature evaluates if there is a dominance relation or not between a given pair of configurations. Indeed, the optimization query evaluates if $\omega \succ \omega'$ is true making the reverse order relation false, namely $\omega' \not\succ \omega$. The *ordering query* is a weaker interrogation that verifies if $\omega \not\succ \omega'$ and $\omega' \not\succ \omega$ are both true. In other words, it verifies if there exists a strict dominance relation between two configurations. It is a yes or no query checking if two configurations are comparable or not. Actually, a user may be satisfied by simply knowing that a configuration can consistently be ranked as preferred to another. The following example gives an illustration of the *ordering* query.

Example 1.14 *Consider CP-net* $\mathcal{C}$ *in Figure 1.1. Configurations* $\omega_1 = sle\bar{r}$ *and* $\omega_3 = sl\bar{e}\bar{r}$ *differ by the value of* E *but satisfy the same set* $\{S\}$ *of ancestors. For the context* s, *there exists a statement* $s : e \succ \bar{e}$, *then we can conclude that* $\omega_3 \succ_{\mathcal{C}} \omega_1$ *does not hold. Consider now configurations* $\omega_0 = sler$ *and* $\omega_9 = \bar{s}le\bar{r}$ *that differ by values of*

nodes S and R. Conditional tables associated with these variables stipulate that $\bar{s} \succ s$ and le : $r \succ \bar{r}$ which means that $\omega_9[S] \succ \omega_0[S]$ and $\omega_0[R] \succ \omega_9[R]$. This leads to the conclusion that $\omega_0 \succ_{\mathcal{C}} \omega_9$ and $\omega_9 \succ_{\mathcal{C}} \omega_0$ do not hold which means that ω_0 and ω_9 are not comparable.

Complexity of ordering query Let $\mathcal{C}$ be a CP-net and (ω, ω') be a pair of configurations that differ by the value of X but have the same instantiations over all ancestors of X in $\mathcal{X}$. If there exists a statement $u : X \succ \neg X$ in $CPT(X)$ such that $X \models \omega$ and $\neg X \models \omega'$[7] then the assertion $\omega' \succ_{\mathcal{C}} \omega$ is false. Consider now a different pair of configurations (ω, ω') over variables $\mathcal{X}$ of CP-net $\mathcal{C}$. The complexity of determining the truth of that the assertion $\omega' \succ_{\mathcal{C}} \omega$ (or $\omega \succ_{\mathcal{C}} \omega'$) do not hold is linear in the number of decision variables (O(N)) [Boutilier et al., 2004]. The idea of the proof is to make a top-down traversal of variables of the network and verify if, for a given context u, $\exists X \in \mathcal{X}$ such that $\omega[X] \succ \omega'[X]$. Thus, we can confirm that $\omega' \succ_{\mathcal{C}} \omega$ does not hold. If configurations differ by more than a variable, for instance on $\{X_i, X_j\} \in \mathcal{X}$ then there must exist statements such that given a context u, $\omega[X_i] \succ \omega'[X_i]$ and $\omega'[X_j] \succ \omega[X_i]$. This implies that both assertions $\omega \succ_{\mathcal{C}} \omega'$ and $\omega' \succ_{\mathcal{C}} \omega$ are false.

1.4 TCP-nets

Expressiveness of CP-nets can be enhanced by introducing information about the relative importance between decision variables. In that regard *Tradeoff-enhanced CP-nets* [Brafman and Domshlak, 2002], TCP-nets for short, are an extension of CP-nets that encodes both preference statements of a user as well as conditional importance relation between variables. Thus, in addition of encoding specifications of the form "In the context of u, I strictly prefer x to $\bar{x}$", this model can also take into account statements like "In context u, the preference associated with X is more important than the one associated with another variable X'".

In case of acyclic graphs, this leads to a richer expressive power than CP-nets providing a more refined partial ordering between outcomes. For their graphical structure, TCP-nets are represented by a DAG with three types of edges, namely we add two type edges corresponding to conditional or unconditional importance relations between preferences associated with two variables. If variables importance is conditioned by the value of some other variables, then the encoding edge is undirected, labeled with

[7] This notation means that the preference associated with variable X is satisfied by the configuration ω. This means that the preference associated with X is not satisfied by ω'.

the conditioning variables and associated to a table defining the importance relation between variables in the context of what is defined by a so-called *selector set*. The notation $X \triangleright X'$ means that X is more important than X'.

TCP-nets still obey the independence property ceteris paribus leading to more comparisons than the original network. Their structure can be seen as basically being composed of a *Conditional preference network structure* $\mathcal{P}$ completed by two informational principles respectively ceteris paribus semantic and attribute importance relation that is depicted by additional types of edges. They are formally defined by

Definition 1.10 (TCP-net) *A Tradeoff-enhanced Conditional Preference network* $\mathcal{N} = \langle \mathcal{G}, CPT, CIT \rangle$*, denoted by TCP-net is composed of three components:*

(i) a graph $\mathcal{G} = (\mathcal{X}, E)$ *where* $\mathcal{X} = \{X_1, \ldots, X_N\}$ *is a set of* N *decision variables and* E *the set of (directed) edges representing preference dependencies and importance relation between them. Each arc in* E *between* X_i *and* X_j *can be associated with one of the three following types:*

- **cp-arcs** *standing for conditional preference, which are directed edges of the form* $X_i \rightarrow X_j$ *expressing that the preferred value of* X_j *depends on the value of* X_i*.*
- **i-arcs** *standing for importance relation, which are directed dashed edges depicted by* $X_i \dashrightarrow X_j$ *expressing that* X_i *is more important than* X_j*.*
- **ci-arcs** *standing for conditional importance relation, which are undirected edges of the form* X_i—X_j*, labeled with a* selector set $\mathcal{S} \subseteq \mathcal{X} \setminus \{X_i, X_j\}$*.*

(ii) a set of conditional preference tables $CPT = \{CPT(X_1), \ldots, CPT(X_N)\}$ *where* $CPT(X_i)$ *is the conditional preference table attached to* X_i *and composed of statements of the form* $u_i : x_i \succ x_i'$ *expressing a strict total order* $\succ$ *over values of* X_i *in the context of each instance* u_i *of* X_i*'s parents* U_{X_i}*.*

(iii) a set of conditional importance tables $CIT = \{CIT(\mathcal{S}_1), \ldots, CIT(\mathcal{S}_J)\}$ *that express the relative importance between a pair of nodes given assignments (not necessarily all) of the* selector set $\mathcal{S}_i$*.*

TCP-nets obey to the ceteris paribus preferential independence property.

We now give an example of TCP-net with the three kinds of arcs.

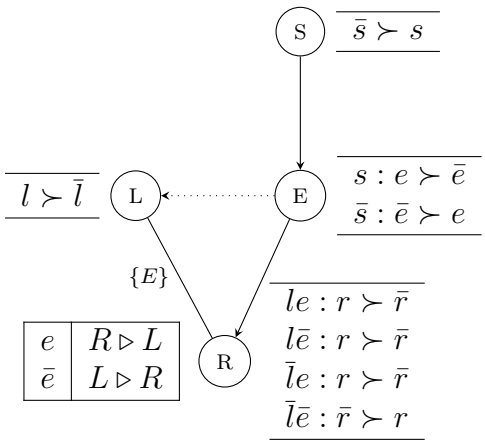


Figure 1.9: TCP-net

Example 1.15 *Let us continue Example 1.1, by considering that the user's preferences over the car propulsion system has actually higher importance than her preferences over the vehicle category since she is more concerned by the car reliability. This information is represented by a* i-arc *from E to L drawn in dotted lines. Moreover, given an electric motor, deciding on whether to purchase or rent a car is more important than the car category ($e : R \triangleright L$). However, if the car is mechanical, the user is more likely to choose car category before deciding of its ownership ($\bar{e} : L \triangleright R$). An edge* ci-arc *between L and E labeled with a conditional importance table or $CIT(E)$ expressing the previously enunciated statements is drawn. These information are sumed up by Figure 1.9 that illustrates an extension of the CP-net associated with items of the graphical structure in Figure 1.1 now capturing relative importance relation between decision variables.*

However, the implicit priority enforced in favor of father nodes cannot be reversed by the use of TCP-net.

1.4.1 Satisfiability of a TCP-net

Semantics of a TCP-net $\mathcal{N}$ is defined in term of strict partial orders consistent with the set of generalized preference statements in CPT and with the importance relation encoded by the network. In fact, each type of arcs depicted by a TCP-net entail a dominance relation. Definitions below formally define satisfiability with respect to each type of arc, leading to the global satisfiability of a TCP-net.

Definition 1.11 (Order satisfiability) [8] *A strict partial order $\succ_{\mathcal{N}}$ satisfies a TCP-net $\mathcal{N}$ iff*

*(i) For **cp-arcs**: $\succ_{\mathcal{N}}$ satisfies generalized preference statements in $CPT(X_i)$, $\forall X_i \in \mathcal{X}$. This means that for each dominance relation, e.g., $\omega \succ_{\mathcal{N}} \omega'$ between pairs of configurations that differ by the value of X_i, there should exist a statement in $CPT(X_i)$ stipulating that $\omega[X_i] \succ \omega'[X_i]$ for a given context $u \in \underline{U_{X_i}}$.*

*(ii) For **i-arcs**: $\succ_{\mathcal{N}}$ satisfies the importance relation $X_i \triangleright X_j$. This means that for each dominance relation, e.g., $\omega \succ_{\mathcal{N}} \omega'$ between pairs of configurations that differ by the value of exactly two variables X_i and X_j, there exists a statement in $CPT(X_i)$ stipulating that, for the most important variable X_i, $\omega[X_i] \succ \omega'[X_i]$ for a given context $u \in \underline{U_{X_i}}$.*

*(iii) For **ci-arcs**: $\succ_{\mathcal{N}}$ satisfies the importance relation $X_i \triangleright_s X_j$ for a given assignment s of the selector set $\mathcal{S}$. This means that for each dominance relation, e.g., $\omega \succ_{\mathcal{N}} \omega'$ between pairs of configurations that differ by the value of exactly two variables X_i and X_j, there exists a statement in $CPT(X_i)$ stipulating that, for the most important variable X_i, $\omega[X_i] \succ \omega'[X_i]$ for a given context $u \in \underline{U_{X_i}}$.*

A TCP-net $\mathcal{N}$ is satisfiable if and only if there exists a strict transitive partial order $\succ_{\mathcal{N}}$ that satisfies it. This means that if $\omega \succ_{\mathcal{N}} \omega'$ with respect to $\mathcal{N}$ then all preference orders entailed by $\mathcal{N}$ verify the assertion $\omega \succ \omega'$.

In the sequel, we give an example of a TCP-net along with its induced ordering.

Example 1.16 *Consider TCP-net $\mathcal{N}$ in Figure 1.9.* cp-arcs *yield to dominance relation entailed by the ceteris paribus property. Details about order relation induced from* i-arcs *and* ci-arcs *are described in Table 1.3. Besides of preferences induced from CP-net (see Figure 1.3), additional dominance relations entailed by the TCP-net $\mathcal{N}$ are written in bold.*

1.4.2 Querying TCP-nets

The two main queries that can be performed on the original CP-nets are *optimisation* and *dominance* query. The relative importance relation does not play a role in determining the best (resp. worst) configuration of a TCP-net. To answer the *optimization*

[8] Taking into account the importance relation between variables.

Arc type	Arc	Induced importance relation	Entailed preferences	
i-arcs	$E \dashrightarrow L$	$E \triangleright L$	$sler \succ_{\mathcal{N}} s\bar{l}\bar{e}r$	$\omega_0 \succ_{\mathcal{N}} \omega_6$
			$sle\bar{r} \succ_{\mathcal{N}} s\bar{l}\bar{e}\bar{r}$	$\omega_1 \succ_{\mathcal{N}} \omega_7$
			$\boldsymbol{\bar{s}\bar{l}\bar{e}r \succ_{\mathcal{N}} \bar{s}ler}$	$\boldsymbol{\omega_{14} \succ_{\mathcal{N}} \omega_8}$
			$\boldsymbol{\bar{s}\bar{l}\bar{e}\bar{r} \succ_{\mathcal{N}} \bar{s}le\bar{r}}$	$\boldsymbol{\omega_{15} \succ_{\mathcal{N}} \omega_9}$
			$\boldsymbol{s\bar{l}er \succ_{\mathcal{N}} sl\bar{e}r}$	$\boldsymbol{\omega_4 \succ_{\mathcal{N}} \omega_2}$
			$\boldsymbol{s\bar{l}e\bar{r} \succ_{\mathcal{N}} sl\bar{e}\bar{r}}$	$\boldsymbol{\omega_5 \succ_{\mathcal{N}} \omega_3}$
			$\bar{s}l\bar{e}r \succ_{\mathcal{N}} \bar{s}\bar{l}er$	$\omega_{10} \succ_{\mathcal{N}} \omega_{12}$
			$\bar{s}l\bar{e}\bar{r} \succ_{\mathcal{N}} \bar{s}\bar{l}e\bar{r}$	$\omega_{11} \succ_{\mathcal{N}} \omega_{13}$
ci-arcs	E—L	$R \triangleright_e L$	$sler \succ_{\mathcal{N}} sle\bar{r}$	$\omega_0 \succ_{\mathcal{N}} \omega_1$
			$\bar{s}ler \succ_{\mathcal{N}} \bar{s}le\bar{r}$	$\omega_8 \succ_{\mathcal{N}} \omega_9$
		$L \triangleright_{\bar{e}} R$	$sl\bar{e}r \succ_{\mathcal{N}} sl\bar{e}\bar{r}$	$\omega_2 \succ_{\mathcal{N}} \omega_3$
			$\bar{s}l\bar{e}r \succ_{\mathcal{N}} \bar{s}l\bar{e}\bar{r}$	$\omega_{10} \succ_{\mathcal{N}} \omega_{11}$

Table 1.3: Preferences derived from i-arcs and ci-arcs

query we need to go through the network from root nodes to leaves and assign at each step the preferred (resp. rejected) value of the node in the context of parents. However, the process of finding the dominance relation between a pair of configurations is extended by considering, in addition to *worsening flip sequences*, another type of flipping sequences called *importance flip sequences* based on the importance relation of attributes.

Consider the pair of configurations (ω, ω') that differ by two variable values. If the TCP-net $\mathcal{N}$ stipulates that $\omega[X_i] \succ \omega'[X_i]$ and $\omega'[X_j] \succ \omega[X_j]$ given some context (assignment of parents U_{X_i} and U_{X_j}), then, an *importance worsening flip* from ω to ω' exists if there is a priority of X_i over X_j conditioned (or not) by the selector set $\mathcal{S}$ such that $z \models \omega$ and $z \models \omega'$ for $z \in \underline{\mathcal{S}}$. A configuration ω dominates ω' with respect to a TCP-net $\mathcal{N}$, formally $\omega \succ_{\mathcal{N}} \omega'$, if there exists a *worsening flip sequence* between them. This is now more formally stated by the two following definitions.

Definition 1.12 (Importance worsening flip) *Let ω, ω' be two configurations in Ω that differ by exactly two variables values X_i and $X_j \in \mathcal{X}$. There exists an importance worsening flip from ω to ω' if and only if the following conditions are satisfied*

(i) $X_i \triangleright_z X_j$ given an assignment z of the selector set $\mathcal{S}$ such that $\omega, \omega' \models z$ and $\mathcal{S} \subseteq \mathcal{X} \setminus \{X_i, X_j\}$.

(ii) There should be statements in CPT such that $\omega[X_i] \succ \omega'[X_i]$ for the same context U_{X_i}, and $\omega'[X_j] \succ \omega[X_j]$ given the same context U_{X_j}.

Definition 1.13 (Importance worsening flip sequence) *The sequence $\omega_0 \succ_{\mathcal{N}} \omega_1 \succ_{\mathcal{N}}$*

$\cdots \succ_{\mathcal{N}} \omega_m$ is an importance worsening flip sequence with respect to a TCP-net $\mathcal{N}$ such that $k \in [0, m]$, if and only if one of the following conditions is satisfied

(i) Configurations ω_k and ω_{k+1} differ by a single flip of variable $X \in \mathcal{X}$, and there exists a worsening flip from ω_k to ω_{k+1},

(ii) Configurations ω_k and ω_{k+1} differ by the value of two variables and there exists an importance worsening flip from ω_k to ω_{k+1}.

Considering all worsening flip sequences induced by a CP-net, finding the additional dominance relations induced by adding *i-arcs* and *ci-arcs* to this structure comes down to determine all importance flipping sequences between pairs that differ by the value of two variables following the second condition in Definition 1.13. Example 1.17 exhibits the ordering relation between configurations for the Example of Figure 1.9.

Example 1.17 *Figure 1.10 represents the induced graph of TCP-net $\mathcal{N}$ in Figure 1.9. Due to the importance relation $E \triangleright L$, we detect four additional comparisons or* importance worsening flips *consisting of $\omega_4 \succ_{\mathcal{N}} \omega_2$, $\omega_{14} \succ_{\mathcal{N}} \omega_8$, $\omega_5 \succ_{\mathcal{N}} \omega_3$ and $\omega_{15} \succ_{\mathcal{N}} \omega_9$ (see Table 1.3). For instance, $\omega_4 = s\bar{l}er$ and $\omega_2 = sl\bar{e}r$ differ by two values of variables L and E. For the more important variable E, $\omega_4[E] = e \succ \omega_2[E] = \bar{e}$ given the context s. For the less important variable namely L, $\omega_2[L] = l \succ \omega_4[L] = \bar{l}$ where $\omega_2, \omega_4 \models r$ for a fixed value $r \in \underline{R}$. This preference entailment corresponds to an* importance worsening flip.

There exists a sequence of worsening flips between ω_4 and ω_1 described by the strict partial order $\omega_4 \succ_{\mathcal{N}} \omega_2 \succ_{\mathcal{N}} \omega_3$. In fact, configurations ω_4 ω_2 are ranked by means of the second condition in Definition 1.13, while configurations ω_2 and ω_3 are ranked by means of the first condition in Definition 1.13.

Complexity of dominance query Methods developed for searching for worsening flip sequences with respect to CP-nets [Domshlak and Brafman, 2002] [Boutilier et al., 2004] can also be used for TCP-nets. Generally the dominance testing given a TCP-net $\mathcal{N}$ is NP-hard [Brafman et al., 2006].

1.5 LP-trees

Lexicographic preference trees (LP-trees) have been introduced by[Booth et al., 2010]. They were proposed originally for learning purposes and more specifically for learning

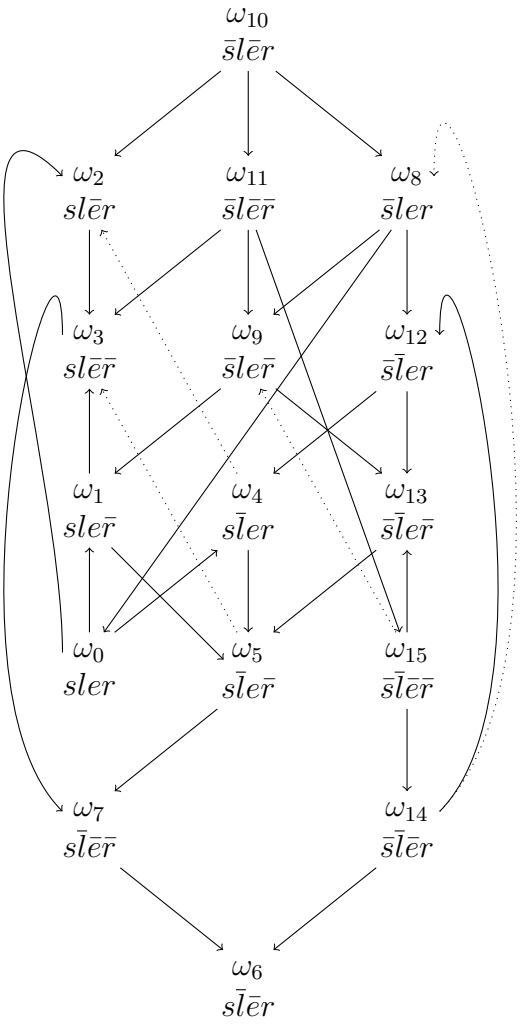


Figure 1.10: Induced graph of TCP-net of Figure 1.3. Arrows in straight line are deduced from ceteris paribus assumption and those in dotted represent additional comparisons deduced from the importance relation between variables.

ordinal preferences. The concept of a lexicographic order over preferences involves a set of attributes $\mathcal{X}$ and a strict total importance order relation $\triangleright$ over them. It can be depicted by rooted tree where nodes correspond to decision variables and edges indicate the relative importance between them. An arc $X_i \to X_j$ s.t. $X_i, X_j \in \mathcal{X}$, implies that X_i is more important than X_j formally encoded by the *importance relation statement*: $X_i \triangleright X_j$. This relation can be contextual, i.e., X_i is more important than X_j given $X_i = x_i$, this case is formalized by $X_i \triangleright_{x_i} X_j$. LP-trees allow to express two types of relations. In regards to a Conditional preference structure, LP-trees exhibit importance relations between variables by generalizing the lexicographic order, in addition of permitting conditioning on variable's preference relations. Both information

are compactly reproduced by a graphical structure depicted by a directed tree and a set of local tables.

As a representation and learning model, researchers have introduced different definitions of LP-trees [Booth et al., 2010] [Fargier et al., 2018] [Fargier and Mengin, 2021a]. In the sequel we propose a definition that subsumes all others (see definition 1.14). In Chapter 5, we propose to structure representations of classes and extensions of this model by providing well-detailed definitions and reasoning procedures.

Definition 1.14 (General LP-tree) *A Lexicographic Preference Tree $\mathcal{T} = \langle \mathcal{A}, PT \rangle$ denoted by LP-tree involves two components :*

(i) a directed tree $\mathcal{A} = \langle \mathcal{X}, E \rangle$ where $\mathcal{X}$ is a set of decision variables and E is a set of edges of the form $X_i \rightarrow X_j$ such that $\{X_i, X_j\} \in \mathcal{X}$, indicating that the preference over X_i is more important than X_j.

(ii) a set of local preference tables $PT = \{PT(X_1), \ldots, PT(X_N)\}$ where $PT(X_i)$ is the conditional preference table attached to $X_i \in \mathcal{X}$. $PT(X_i)$ contains either conditional or unconditional specifications implying a strict total order over assignments of X_i.

A LP-tree should respect the following statements:

1. *Each variable $X_i \in \mathcal{X}$ appears at most once in each branch of the tree.*
2. *Each non-leaf node $X_i \in \mathcal{X}$ has either one single unlabeled outgoing edge or two outgoing edges respectively labeled by x_i and $\bar{x_i}$(the two possible values of X_i).*

The following example gives an illustration of a general LP-tree.

Example 1.18 *Let us reconsider the decision problem about choosing to rent or buy a car with a restriction on three variables, i.e., $\mathcal{X} = \{L, E, R\}$ where L: category, E: propulsion system, R: ownership, $\underline{L} = \{l, \bar{l}\}$, $\underline{E} = \{e, \bar{e}\}$ and $\underline{R} = \{r, \bar{r}\}$. Suppose that the user's preferences are:*

- *I prefer to drive a luxury vehicle (l) rather then a modest one ($\bar{l}$)*
- *Given a luxury car (l), preferences over the propulsion system (E) are more important than those about the ownership (R), and reversely when the car is modern ($\bar{l}$).*

- *When the car is luxury, I prefer it to be equipped with a thermal propulsion system* ($\bar{e}$) *since I like the sound it makes.*
- *It would be more reasonable to rent* (r) *it mainly to avoid maintenance fees.*
- *Alternatively, when the car is modest, I prefer to own* ($\bar{r}$) *it and in this context, I would choose a thermal motor since, conversely, maintenance fees are too expensive. Otherwise, if I am renting the vehicle I prefer to have an electric motor* (e) *just to discover the new technology with minor cost.*

Figure 1.11 sums up the relative importance of decision variables encoded by the specified statements: $L \rhd_l E$, $E \rhd_l R$ *for the leftmost branch of the tree and* $L \rhd_{\bar{l}} R$, $R \rhd_{\bar{l}} E$ *for the opposite side. It also reveals dependencies between variables where preferences can be unconditional, e.g., leaf node of the leftmost branch of the network, or conditional, e.g., leaf node of the rightmost branch.*

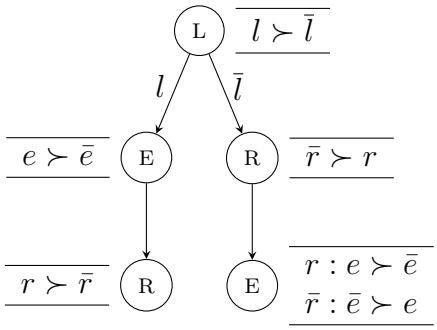


Figure 1.11: Example of a general LP-tree

Considering an LP-tree model, we can perform the standard queries of CP-nets namely the dominance and optimization queries. Finding the optimal configuration consists on sweeping through the tree from top to bottom while assigning for each node of the graph its preferred value, i.e., same procedure as for CP-nets. In order to compare a pair of configurations, we first need to select variables for which they have same assignments. We then trace along the tree structure and consider labels in edges until reaching the *decisive* variable X_i on which configurations differ. The dominance relation $\succ_{Lex}$ between the pair of configuration is deduced from the local table associated with X_i. The following example suggests how configurations are compared (see Chapter 5 for more details).

Example 1.19 *Let us pursue with the Example 1.18. Consider configurations* $(\bar{l}er, \bar{l}\bar{e}\bar{r})$ *that differ by the value of* E *and* R*. Since both configurations model* $\bar{l}$*, we need to trace*

down the right-most branch of the tree. The decisive variable is R which stipulates that $\bar{r} \succ r$*. This leads to conclude that the dominance relation between configurations is* $\bar{l}e\bar{r} \succ_{Lex} \bar{l}\bar{e}r$*.*

Complexity of dominance query In case of complete LP-trees (see Section 6.2.6) where all priority relations and all preferences are provided, the LP-tree yield to a strict total order. The search for a dominance relation between a pair of configurations can be done in a linear time with respect to the size of the graph. However, the search for the dominance relation between all pairs of configurations is done in polynomial time [Booth et al., 2010].

1.6 On the consistency between graphical representations and cp-theories

In 2011, Nic Wilson [Wilson, 2011] has introduced a logical encoding of preferences that permit to capture order relations induced by graphical models detailed in this chapter. Indeed, conditional preference theories (cp-theories) offer a logical framework for representing conditional preferences. In this section, we aim to briefly discuss the relative expressiveness of CP-nets, TCP-nets and LP-trees by means of cp-theories. We also notice that, generally, a CP-net or TCP-net ordering cannot be reproduced by an LP-tree ordering.

A cp-theory Φ is composed of a collection of *conditional preference statements* or *CP statements* that define an order relation between solutions. A CP statement ϕ over a subset $\mathcal{U} \subseteq \mathcal{X}$ is formalized by $\phi = u|\mathcal{V} : x \succ \bar{x}$, where u is an assignment of variable in set $\mathcal{U}$, $x, \bar{x}$ are values in $\underline{X}$, $\mathcal{U}$ and $\mathcal{V}$ are disjoint variable subsets s.t. $\mathcal{U} \subseteq \mathcal{X} \setminus \{X\}$, $\mathcal{V} \subseteq \mathcal{X} \setminus \mathcal{U}$. It expresses the specification "Given the context u, I prefer x to $\bar{x}$ independently from the value of variables in $\mathcal{V}$", more formally

$$\text{if} \quad \phi = u|\mathcal{V} : x \succ \bar{x} \quad \text{then} \quad yuxv \succ_{\Phi} yu\bar{x}v' \quad \text{with} \quad \mathcal{Y} = \mathcal{X} \setminus \{\{X\} \cup \mathcal{U} \cup \mathcal{V}\}. \tag{1.1}$$

When $\mathcal{V} = \mathcal{X} \setminus \{X\}$ then this means that X is more important than the remaining variables in $\mathcal{X}$. We define:

- $\mathcal{U}$ as the conditioning part or context variables;
- $\mathcal{V}$ the independent or free part;

- X the swapped variable.

For the sake of simplicity, when $\mathcal{V}$ is empty and a CP statement is always true, we simply write $\omega \succ \omega'$ instead of $\top|\{\emptyset\} : \omega \succ \omega'$. A subset Φ of such statements over $\mathcal{X}$ forms a *CP language*.

Given a set Φ of CP statements over variables in $\mathcal{X}$, we can construct what is called a *dependency graph* $\mathcal{H}_\Phi$ encoding dependency relations between variables, which actually coincides with the conditional preference network structure previously introduced in Section 1.2. For each statement $u|\mathcal{V} : x \succ \bar{x}$, $\forall U \in \mathcal{U}$, an arc $U \to X$ is drawn indicating that the preference of X is conditioned by the value of U. Given a cp-theory, $\mathcal{H}_\Phi$ can be computed in polynomial time [Wilson, 2011]. Adding arcs $X \to V$ to $\mathcal{H}_\Phi$ for all $V \in \mathcal{V}$ comes down to construct a more general network which indicates not only the dependency relation between variables but also their relative importance.

Definition 1.15 (Dependency graph) *Given a set Φ of CP statements over attributes in $\mathcal{X}$, a dependency graph $\mathcal{H}_\Phi$ is composed of a graphical structure consisting of a DAG $\mathcal{G} = (\mathcal{X}, E)$ where the set E holds* d-arcs *expressing preference dependency between variables s.t. for each statement $u|\mathcal{V} : x \succ \bar{x}$, $\forall U \in \mathcal{U}$ an arc $U \to X$ is drawn indicating that the preference of X depends on the value of U.*

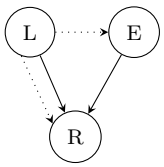


Figure 1.12: Example of a dependency graph $\mathcal{H}$ if only arcs in straight lines are considered, dotted lines reflect importance relation between variables

Example 1.20 *Let us re-consider the car renting example by restricting the set $\mathcal{X}$ to three variable namely: the car category (L), its propulsion system (E) and the ownership (R). All else being equal, an electric (e) vehicle is always preferred to thermal one ($\bar{e}$). Irrespective of the propulsion system (E) and the car ownership (R), a luxury vehicle (l) is preferred to a modest one ($\bar{l}$). Finally, if the car is modest ($\bar{l}$) and is equipped with a thermal engine ($\bar{e}$), purchasing the car ($\bar{r}$) is preferred to renting it (r). These three claims can respectively be transformed into the following CP statements $\phi_1 = e \succ \bar{e}$, $\phi_2 = \top|\{ER\} : l \succ \bar{l}$ and $\phi_3 = \bar{l}\bar{e}|\{\emptyset\} : \bar{r} \succ r$. The latter statements allow to construct the dependency graph in Figure 1.12 by depicting d-arcs $L \to R$ and $E \to R$. Statement ϕ_2 stipulates that there is an importance relation between decision variables s.t. L is more important than E and R.*

Any pair of configurations that differs by more than one flip are comparable if there exists a *CP worsening flipping sequence* from one to another, which is a generalization of flipping sequences for CP-nets.

Definition 1.16 (CP worsening flip) *Let Φ be a set of CP statements and (ω, ω') be two configurations that differ by a single flip on X. There exists a CP worsening flip from ω to ω' with respect to Φ iff there exists a CP statement $\phi = u|\mathcal{V} : x \succ \bar{x}$ in Φ s.t. $\omega, \omega' \models u$, $\omega[Y] = \omega'[Y]$ and $\omega[X] = x$, $\omega'[X] = \bar{x}$ with $u \in \mathcal{U}$ and $Y \in \mathcal{Y}$ s.t. $\mathcal{Y} = \mathcal{X} \setminus \{X\} \cup \mathcal{U} \cup \mathcal{V}$.*

A partial order between configurations is constructed by using the transitive closure on preference constraints entailed from CP statements. A cp-theory Φ is consistent if and only if there exists a strict *acyclic* total order on Ω that satisfies all CP statements in Φ. The following example details ordering entitlements derived from a set of CP statements.

Example 1.21 (Example 1.20 continued) *From the CP statement $\phi_3 = \bar{l}\bar{e}|\{\emptyset\} : \bar{r} \succ r$, we can deduce that given a luxury car (l) or an electric engine (e), renting the vehicle (r) is preferred to buying ($\bar{r}$) it, formally $\phi_4 = l|\{\emptyset\} : r \succ \bar{r}$ and $\phi_5 = e|\{\emptyset\} : r \succ \bar{r}$. From each CP statement we derive an ordering over some configurations in Ω. Table 1.4 outlines constraints associated with each CP statement.*

Let Φ be the cp-theory that contains all CP statements previously stated. There is a worsening flip sequence from $\bar{l}er$ to $\bar{l}\bar{e}\bar{r}$ since ϕ_5 entails $\bar{l}er \succ_\Phi \bar{l}e\bar{r}$ and ϕ_1 entails $\bar{l}e\bar{r} \succ_\Phi \bar{l}\bar{e}\bar{r}$, thus $\bar{l}er \succ_\Phi \bar{l}\bar{e}\bar{r}$. Using the transitive closure, the induced partial order on configurations in Ω is $ler \succ_\Phi \{le\bar{r}, l\bar{e}r\} \succ_\Phi l\bar{e}\bar{r} \succ_\Phi \bar{l}er \succ_\Phi \bar{l}e\bar{r} \succ_\Phi \bar{l}\bar{e}\bar{r} \succ_\Phi \bar{l}\bar{e}r$ with $le\bar{r}$ and $l\bar{e}r$ not being ordered.

In following sections, we are interested about which extent orderings induced by a given CP-net, TCP-net or LP-tree can be represented by a cp-theory.

1.6.1 Expressing CP-nets by cp-theories

[Wilson, 2011] has proven that CP statements can actually cover much more information than the ceteris paribus or the "everything else being equal" assumption . Mapping a CP-net $\mathcal{C}$ into a cp-theory $\Phi_\mathcal{C}$ can be done by transforming each statement $u : X \succ \neg X$, where $u \in \underline{U_X}$, of each conditional preference table $CPT(X)$ into the CP statement $u|\{\emptyset\} : X \succ \neg X$. The cp-theory $\Phi_\mathcal{C}$ corresponds to the union of CP

Φ	CP statement	Entailed preferences
ϕ_1	$e \succ_\Phi \bar{e}$	$ler \succ_\Phi l\bar{e}r$ $le\bar{r} \succ_\Phi l\bar{e}\bar{r}$ $\bar{l}er \succ_\Phi \bar{l}\bar{e}r$ $\bar{l}e\bar{r} \succ_\Phi \bar{l}\bar{e}\bar{r}$
ϕ_2	$\top \vert \{ER\} : l \succ_\Phi \bar{l}$	$\{ler, le\bar{r}, l\bar{e}r, l\bar{e}\bar{r}\} \succ_\Phi \{\bar{l}er, \bar{l}e\bar{r}, \bar{l}\bar{e}r, \bar{l}\bar{e}\bar{r}\}$
ϕ_3	$\bar{l}\bar{e} \vert \{\emptyset\} : \bar{r} \succ_\Phi r$	$\bar{l}\bar{e}\bar{r} \succ_\Phi \bar{l}\bar{e}r$
ϕ_4	$l \vert \{\emptyset\} : r \succ_\Phi \bar{r}$	$ler \succ_\Phi le\bar{r}$ $l\bar{e}r \succ_\Phi l\bar{e}\bar{r}$
ϕ_5	$e \vert \{\emptyset\} : r \succ_\Phi \bar{r}$	$ler \succ_\Phi le\bar{r}$ $\bar{l}er \succ_\Phi \bar{l}e\bar{r}$

Table 1.4: Preferences derived from CP statements given a CP-net

statements associated to each variable $X \in \mathcal{X}$. The transformation leads to the exact same ordering induced by a CP-net[9].

Example 1.22 *Consider the CP-net depicted by Figure 1.1. To construct its associated cp-theory, each preference statement is transformed into a CP statement leading to the cp-theory* $\Phi = \{\phi_1 = \bar{s} \succ s, \phi_2 = l \succ \bar{l}, \phi_3 = s \vert \{\emptyset\} : e \succ \bar{e}, \phi_4 = \bar{s} \vert \{\emptyset\} : \bar{e} \succ e, \phi_5 = le \vert \{\emptyset\} : r \succ \bar{r}, \phi_6 = l\bar{e} \vert \{\emptyset\} : r \succ \bar{r}, \phi_7 = \bar{l}e \vert \{\emptyset\} : r \succ \bar{r}, \phi_8 = \bar{l}\bar{e} \vert \{\emptyset\} : \bar{r} \succ r\}$. *After inferring their entailed preference constraints, the partial order induced by the cp-theory* Φ *on configurations of* Ω *is* $\omega_{10} \succ_\Phi \omega_8 \succ_\Phi \omega_0 \succ_\Phi \{\omega_2, \omega_{11}\} \succ_\Phi \omega_{15} \succ_\Phi \omega_{14} \succ_\Phi \{\omega_9, \omega_{12}\} \succ_\Phi \{\omega_1, \omega_4, \omega_{13}\} \succ_\Phi \{\omega_3, \omega_5\} \succ_\Phi \omega_7 \succ_\Phi \omega_6$. *We can check that the ordering is in accordance with the induced graph in Figure 1.3 but adds a lot of information.*

1.6.2 Expressing TCP-nets by cp-theories

A CP-net has directed arcs describing dependence relations between decision variables, which are called cp-arcs in TCP-net structures. TCP-net also encodes an additional type of information which consists of an importance relation between variables. If the relation is unconditional, then it can be depicted by i-arcs. Otherwise, it is represented by ci-arcs associated with conditional tables that express the the importance relation between variables given a context s of the selector set $\mathcal{S}$ values. Note that ci-arcs can simply be formalized by *ci-statements* of the form $X \rhd_s Y$ indicating that X is more important than Y given s. In cp-theories, arcs of a TCP-net structure are transformed into CP statements as follows:

[9]The induced ordering of the CP-net must be acyclic

- a **cp-arc** $U \rightarrow X$ is transformed into cp-statements $u|\{\emptyset\} : x \succ \bar{x}$ such that $u \in \underline{U}$ and $x \succ \bar{x}$ with respect to $CPT(X)$ for $\{x, \bar{x}\} \in \underline{X}$. Each cp-statement entails a preference constraint of the form $zux \succ_\Phi zu\bar{x}$ where $\mathcal{Z} = \mathcal{X} \setminus U \setminus \{X\}$, $z \in \underline{\mathcal{Z}}$.

- an **i-arc** $X \dashrightarrow Y$ is transformed into cp-statements $u|\{Y\} : x \succ \bar{x}$ such that $u \in \underline{U}$ and $x \succ \bar{x}$ with respect to $CPT(X)$ for $\{x, \bar{x}\} \in \underline{X}$. Each cp-statement entails a preference constraint of the form $zxy \succ_\Phi z\bar{x}y'$ where $\mathcal{Z} = \mathcal{X} \setminus \{X, Y\}$, $z \in \underline{\mathcal{Z}}$ and $\{y, y'\} \in \underline{Y}$.

- a **ci-statement** $X \rhd_s Y$ is transformed into cp-statements $qs|\{Y\} : x \succ \bar{x}$ where $q \in \underline{\mathcal{Q}}$ such that $\mathcal{Q} = U_X \setminus \mathcal{S}$ and $x \succ \bar{x}$ with respect to $CPT(X)$ for $\{x, \bar{x}\} \in \underline{X}$. Each cp-statement entails a preference constraint of the form $zsxy \succ_\Phi zs\bar{x}y'$ such that $z \in \underline{\mathcal{Z}}$ and $\mathcal{Z} = \mathcal{X} \setminus S \setminus \{X, Y\}$.

Any TCP-net can be converted into a set of CP statements that infers the same dominance relation between configurations in Ω [Wilson, 2011]. See the following example for illustration.

Example 1.23 *Table 1.5 show details of transforming arcs of TCP-net in Figure 1.9 into a cp-theory. Each arc is translated into a set of CP statements from which an ordering over subsets of configurations is entailed.*

Type	Arc	CP statement	$\mathcal{Z}$	Entailed preferences
cp-arcs	$S \rightarrow E$	$\phi_3 = s\|\{\emptyset\} : e \succ \bar{e}$	$\{L, R\}$	$\{sler, sle\bar{r}, s\bar{l}er, s\bar{l}e\bar{r}\} \succ_\Phi \{sl\bar{e}r, sl\bar{e}\bar{r}, s\bar{l}\bar{e}r, s\bar{l}\bar{e}\bar{r}\}$
		$\phi_4 = \bar{s}\|\{\emptyset\} : \bar{e} \succ e$	$\{L, R\}$	$\{\bar{s}l\bar{e}r, \bar{s}l\bar{e}\bar{r}, \bar{s}\bar{l}\bar{e}r, \bar{s}\bar{l}\bar{e}\bar{r}\} \succ_\Phi \{\bar{s}ler, \bar{s}le\bar{r}, \bar{s}\bar{l}er, \bar{s}\bar{l}e\bar{r}\}$
	$E \rightarrow R$	$\phi_5 = le\|\{\emptyset\} : r \succ \bar{r}$	$\{S\}$	$\{sler, \bar{s}ler\} \succ_\Phi \{sle\bar{r}, \bar{s}le\bar{r}\}$
		$\phi_6 = l\bar{e}\|\{\emptyset\} : r \succ \bar{r}$	$\{S\}$	$\{sl\bar{e}r, \bar{s}l\bar{e}r\} \succ_\Phi \{sl\bar{e}\bar{r}, \bar{s}l\bar{e}\bar{r}\}$
		$\phi_7 = \bar{l}e\|\{\emptyset\} : r \succ \bar{r}$	$\{S\}$	$\{s\bar{l}er, \bar{s}\bar{l}er\} \succ_\Phi \{s\bar{l}e\bar{r}, \bar{s}\bar{l}e\bar{r}\}$
		$\phi_8 = \bar{l}\bar{e}\|\{\emptyset\} : \bar{r} \succ r$	$\{S\}$	$\{s\bar{l}\bar{e}\bar{r}, \bar{s}\bar{l}\bar{e}\bar{r}\} \succ_\Phi \{s\bar{l}\bar{e}r, \bar{s}\bar{l}\bar{e}r\}$
i-arcs	$L \dashrightarrow E$	$\phi_9 = s\|\{L\} : e \succ \bar{e}$	$\{S, R\}$	$\{sler, sle\bar{r}\} \succ_\Phi \{s\bar{l}\bar{e}r, s\bar{l}\bar{e}\bar{r}\}$ $\{s\bar{l}er, s\bar{l}e\bar{r}\} \succ_\Phi \{sl\bar{e}r, sl\bar{e}\bar{r}\}$
		$\phi_{10} = \bar{s}\|\{L\} : \bar{e} \succ e$	$\{S, R\}$	$\{\bar{s}l\bar{e}r, \bar{s}l\bar{e}\bar{r}\} \succ_\Phi \{\bar{s}\bar{l}\bar{e}r, \bar{s}\bar{l}\bar{e}\bar{r}\}$ $\{\bar{s}\bar{l}\bar{e}r, \bar{s}\bar{l}\bar{e}\bar{r}\} \succ_\Phi \{\bar{s}ler, \bar{s}le\bar{r}\}$
ci-statements	$R \rhd_e L$	$\phi_{11} = el\|\{L\} : r \succ \bar{r}$	$\{S\}$	$\{sler, \bar{s}ler\} \succ_\Phi \{sle\bar{r}, \bar{s}le\bar{r}\}$
		$\phi_{12} = e\bar{l}\|\{L\} : r \succ \bar{r}$	$\{S\}$	$\{s\bar{l}er, \bar{s}\bar{l}er\} \succ_\Phi \{s\bar{l}e\bar{r}, \bar{s}\bar{l}e\bar{r}\}$
	$L \rhd_{\bar{e}} R$	$\phi_{13} = \bar{e}\|\{R\} : l \succ \bar{l}$	$\{S\}$	$\{sl\bar{e}r, \bar{s}l\bar{e}r\} \succ_\Phi \{s\bar{l}\bar{e}r, \bar{s}\bar{l}\bar{e}r\}$ $\{sl\bar{e}\bar{r}, \bar{s}l\bar{e}\bar{r}\} \succ_\Phi \{s\bar{l}\bar{e}r, \bar{s}\bar{l}\bar{e}r\}$

Table 1.5: Preferences derived from CP statements given a TCP-net in Figure 1.9

1.6.3 Expressing lexicographic orders by cp-theories

Consider that X dominates $\mathcal{V}$ and $\mathcal{Z}$ be a subset of variables such that $\mathcal{Z} = \mathcal{X} \backslash \{X\} \cup \mathcal{V}$. Let (ω, ω') be a pair of configurations that agree on the value of $\mathcal{Z}$ with $\omega[X] \succ \omega'[X]$ then ω lexicographically dominates ω'. In fact, any lexicographic order $\succ_{\mathcal{L}}$ over decision variables $\mathcal{X}$ can be mapped into a cp-theory Φ such that its associated order $\succ_{\Phi}$ equals $\succ_{\mathcal{L}}$. Suppose that elements in $\mathcal{X}$ are ranked from the most to the least important variable. Each attribute $X_i = \{x_i, x_i'\} \in \mathcal{X}$ s.t. $x_i \succ x_i'$ can be associated to a set Φ_{X_i} composed of formulas of the form $\top | [X_{i+1}, \ldots, X_N] : x_i \succ x_i'$. Let Φ be the union of cp-theories associated to each variable, then it has been proven that orderings $\succ_{\Phi}$ and $\succ_{\mathcal{L}}$ are the same [Wilson, 2011].

Indeed, a lexicographic order represents a stronger form of preference statement than ceteris paribus (for which a value is preferred independently of the other variables). If the cp-theory language is able to represent a CP-net ordering, then obviously it can encode lexicographic orders. To do so, an importance relation statement of the form $X \triangleright Y$ is converted into the CP statement $u | \{Y\} : x \succ \bar{x}$ for $\{x, \bar{x}\} \in \underline{X}$ and $u \in \underline{U_X}$.

Let Φ be a cp-theory that encodes a ceteris paribus ordering, and (ω, ω') be a pair of configurations that differ by the value of X. Assume $\omega \succ \omega'$ with respect to Φ, then there exists a worsening flip from ω to ω'. Which means that, without making use of the transitive closure, two configurations are comparable if and only if they differ by a single flip value. Let us now assume the preference importance relation $A \triangleright B \triangleright C$ over three variables A, B and C with respective preference statements $a \succ \bar{a}$, $b \succ \bar{b}$ and $c \succ \bar{c}$. The configuration $a\bar{b}\bar{c}$ is comparable to $\bar{a}bc$ leading to the dominance relation $a\bar{b}\bar{c} \succ \bar{a}bc$. In fact, we have directly compared two configurations that differ by more than one flip without using the transitive closure over $\succ_{\mathcal{L}}$. This cannot be done using the ceteris paribus assumption. This is why lexicographic orders cannot generally be captured by a CP-net (TCP-net) which only yields partial order. However, there exists some exceptions as shown by the following example.

Example 1.24 *Assume we have two variables A, B with $A \triangleright B$ and the preferences $a \succ \bar{a}$, $a : b \succ \bar{b}$, $\bar{a} : \bar{b} \succ b$. The CP-net ordering is $ab \succ_{\mathcal{C}} a\bar{b} \succ_{\mathcal{C}} \bar{a}\bar{b} \succ_{\mathcal{C}} \bar{a}b$. It can be checked that it agrees with the importance statement $A \triangleright B$.*

1.7 Conclusion

In order to specify preferences over combinatorial alternatives, an expressive but concise representation is needed. Existing preference formalism are often categorized into logical models and graphical models. In this chapter, we have addressed two of the major qualitative graphical preference models: conditional preference networks *CP-nets* [Boutilier et al., 1999] and Lexicographic Preference trees *LP-trees* [Booth et al., 2010].

Beyond their graphical appeal, CP-nets have given rise to several research works. One of their most important extensions are TCP-nets which which allow to explicitly specify priority relations between nodes, contrarily to CP-nets which implicitly impose that a parent's preferences is more important than a child's one. This extension may add additional importance specifications between variables but cannot reverse the priority in favor of parents entailed by CP-nets. CP-nets and their extension TCP-nets permit indifference between values of a variables but may yield to insatisfiable partial orders in this case. To overcome this barrier, explicit restrictions should be taken into consideration.

Like TCP-nets, LP-trees allow to compactly encode (conditional) preferences and importance relations between variables. Even though they are able to express the same informational properties: independence and priority relation between variables, TCP-nets and LP-trees were initially introduced for different purposes. LP-trees are more fitted to learning preferences of a user, while CP-nets and their extensions are motivated by a concern of preference representation and reasoning. Mind that, TCP-nets and LP-trees permit indifference on importance relations.

The last part of this chapter was dedicated to study the consistency between the qualitative graphical representations: CP-nets, TCP-nets and LP-trees using cp-theories. Actually, the cp-theory language offers a bridge to compare the expressiveness of CP-nets, TCP-nets and LP-trees. Indeed, all three mentioned models can be represented by a cp-theory that entails their respective exact same orders. However, orderings induced from CP-nets or TCP-nets cannot generally be represented by an LP-tree.

A summary inspired from [Gouider, 2017] detailing the main differences between the models[10] discussed in this chapter is given in Table 1.6.

Apart from their direct extensions, CP-nets have inspired many theoretical exten-

[10]The recapitulation is only valid for binary-valued decision variables and acyclic graphs.

sions. Preferences can actually be encoded by means of the possibilistic framework which gives rise to Possibilistic Preference Networks (π-pref nets) [Ben Amor et al., 2015] introduced in next chapter. A set of research studies relative to relations between π-pref nets, CP-nets and possibilistic logic [Dubois et al., 2013] [Dubois et al., 2015] [Wilson et al., 2019] is also discussed.

Model / Property	**CP-nets**	**TCP-nets**	**LP-trees**
Graphical structure	DAG	DAG	Directed tree
Informational component	- Conditional preference tables (CPT)	- Conditional preference tables (CPT) - Conditional importance relation tables (CIT)	- Conditional or unconditional preference tables (PT)
Edges	- Conditional preference arcs (cp-arcs)	- Conditional preference arcs (cp-arcs) - Importance relation arcs (i-arcs) - Conditional importance edges (ci-arcs)	- Arcs encoding conditioning over preference and importance relations
Independence property	Ceteris paribus	Ceteris paribus Lexicographic order	Lexicographic order
Ordering type	Partial order	Partial order	Total (pre)order
Optimisation query	Linear	Linear	Linear
Dominance query	- Tree : $O(N^2)$ - Poly-tree : P - Directed-DAGs singly connected : NP-complete	NP-hard	Linear
Ordering query	Linear	-	Linear

Table 1.6: Summary about the graphical preference representations: CP-nets, TCP-nets and LP-trees

Chapter 2

Possibility Theory as a Representation Setting for Preferences

2.1 Introduction

In the previous chapter, we have considered some of the most known qualitative graphical models for representing preferences. In Chapter 3, we shall discuss another representation setting called *possibilistic preference networks* (or π-pref nets for short). π-pref nets are based on possibilistic nets, and thus belong to the framework of possibility theory. Possibility theory [Zadeh, 1978], [Dubois and Prade, 1988] has been originally introduced for representing incomplete and uncertain information.

This chapter is devoted to possibility theory and its use for representing preferences in different formats (possibility distribution, logical bases, possibilistic nets). The chapter is divided in three main sections. After providing the necessary background on possibility theory in Section 2.2, Section 2.3 explains two ways of encoding conditional preferences using a possibilistic logic framework, while the next Section 2.4 defines *possibilistic networks* and ways to translate them into logical bases.

Throughout all the chapter, we use the same running example introduced in Example 1.4 in Chapter 1 (dealing with decision problem about renting or buying a car).

2.2 Background on possibility theory

A piece of information or knowledge is prone to be encoded and treated to infer other sets of information. A piece of knowledge may describe observations, facts of the real world, for example describing a patient's clinical condition. Information is likely to be pervaded by diverse types of deficiencies: if we consider that a piece of information is described by means of several attributes, a lack of the instantiation of part of these attributes leads to *incomplete* information. Besides, a piece of information is considered *uncertain* if there exists doubts about its truth or falsity.

In this context, possibility theory is a framework devoted to the handling of uncertain and incomplete information. It has first been formulated in [Zadeh, 1978] in the late seventies, and has further been studied and extended by [Dubois and Prade, 1988]). Besides of its ability to handle uncertain pieces of knowledge, possibility theory offers a valuable setting for modeling preferences. As possibility theory, probability theory is based on set-functions. Unlike the latter, possibility theory relies on the use of two dual functions: *possibility* and *necessity measures* that will be detailed in the sequel. The second distinction that differentiates the two theories is that possibility theory offers a quantitative and qualitative mathematical setting for reasoning on information while the probability theory remains quantitative.

We now detail the interpretation of a possibility distribution and set functions for dealing with preferences.

Possibility theory is based on the use of possibility distributions. Given a set of configurations Ω, a possibility distribution is a mapping π from Ω to a totally ordered scale that takes values from 0 to 1. $\pi(\omega)$ indicates how preferred is the configuration ω, i.e., to which extent it is satisfactory in regard to other configurations. A preference degree equal to 1 stipulates that the outcome is completely satisfactory, while in contrast a degree equal to 0 means that the alternative is totally rejected. Thus, the greater $\pi(\omega)$ is, the more desirable is ω.

A possibility distribution π_1 is said to be **less specific** than π_2 (in the wide sense) if $\forall \omega \in \Omega$, $\pi_1(\omega) \geq \pi_2(\omega)$. Then, the distribution π_1 is considered as less restrictive than possibility distribution π_2, since according to π_1 any ω is more possible than according to π_2. The interpretation of specificity principles applied to preferences are reported in Chapter 4.

A possibility distribution π is said to be normalized if, $\exists\, \omega \in \Omega$, such that $\pi(\omega) = 1$. This means that, considering the combinatorial domain of alternatives Ω, there is at

least one alternative ω that is fully satisfactory.

Two increasing set functions are built from a possibility distribution: a *possibility* measure Π and a *necessity* measure N which are formally defined by,

$$\Pi(P) = max_{\omega \in P} \qquad \pi(\omega) \qquad \text{s.t. } P \subseteq \Omega \tag{2.1}$$

$$N(P) = min_{\omega \notin P} \qquad \{1 - \pi(\omega)\} \qquad \text{s.t. } P \subseteq \Omega \tag{2.2}$$

Consider the proposition P that consists of a subset of configurations. $\Pi(P)$ estimates to what extent at least one configuration ω of the configuration P is satisfactory, while $N(P)$ evaluates to what extent all configurations outside P are unsatisfactory; thus, $N(P)$ can be viewed as the level of priority of P. The constraint $N(P) > N(\bar{P})$ expresses that the user is more eager to have P than $\bar{P}$. Necessity and possibility measures are associated with a duality relation, namely $N(P) = 1 - \Pi(\bar{P})$. The smaller $N(P)$ is, the larger $\Pi(\bar{P})$ is, and the more satisfactory $\bar{P}$ is. Cases of total indifference can be represented in the possibility theory, by a possibility distribution uniformly equal to 1. The constraint $N(P) = N(\bar{P}) = 0$ means that P and $\bar{P}$ have no priority at all because there are satisfactory configurations both in P and $\bar{P}$. This implies that both propositions are fully satisfactory.

When dealing with pieces of knowledge, a degree of possibility $\pi(\omega)$ expresses the level of plausibility and estimates to what extent ω is consistent with the available information. $\pi(\omega) = 1$ means that ω is totally plausible, whereas $\pi(\omega) = 0$ indicates that ω is impossible. The measure $\Pi(P)$ evaluates to what extent P is consistent with the available beliefs. The necessity measure evaluates to what extent P is entailed by the available knowledge.

As a result of the normalization property, we can verify that $\max(\Pi(P), \Pi(\bar{P})) = 1$ and respectively $\min(N(P), N(\bar{P})) = 0$, which means that if P is satisfactory to some extent then its complement $\bar{P}$ is totally rejected.

Possibility and necessity measures satisfy the maxitivity and the minitivity properties:

$$\Pi(P \cup Q) = \max(\Pi(P), \Pi(Q)) \qquad \text{s.t. } P, Q \subseteq \Omega \tag{2.3}$$

$$N(P \cap Q) = \min(N(P), N(Q)) \qquad \text{s.t. } P, Q \subseteq \Omega \tag{2.4}$$

Considering a possibility distribution π, two other sets of functions can be defined: the *guaranteed* or *strong possibility* measure $\triangle$ and its dual the *potential* or *weak necessity* measure $\triangledown$ [Dubois and Prade, 2012], formally defined by:

$$\triangle(P) = \min_{\omega \in P} \quad \pi(\omega) \quad \text{s.t. } P \subseteq \Omega \tag{2.5}$$

$$\triangledown(P) = \max_{\omega \notin P}(1 - \pi(\omega)) \quad \text{s.t. } P \subseteq \Omega \tag{2.6}$$

If we stick to a preference framework, $\triangle(P)$ estimates to what extent all models of P are satisfactory. In other words, it evaluates the degree for which the least preferred model of P is satisfactory. The assertion $\triangle(P) = 1$ implies that all models of P are fully satisfactory. In relation to the necessity measure, the evaluation of $\triangle(P)$ covers all models of P, while the possibility measure Π checks the existence of at least one configuration that models P. Thus, $\triangle$ is considered to be more demanding than Π due to the fact $\triangle(P) \leq \Pi(P)$.

The potential necessity $\triangledown$ can be derived from $\triangle$ by duality, i.e., $\triangledown(P) = 1 - \triangle(\bar{P})$. The assertion $\triangledown(P)$ estimates to what extent at least one configuration outside P has a low satisfaction degree. This constitutes a prioritized constraint for satisfying the claim $\omega \models P$. $\triangledown(P) = 1$ if and only if there exits a configuration ω s.t. $\omega \not\models P$ that is rejected.

Unlike Π and N, $\triangle$ and $\triangledown$ are decreasing functions. They however can be weakly related by the constraint 2.7 provided that π and $1 - \pi$ are both normalized[1] [Dubois and Prade, 2012].

$$\max(N(P), \triangle(P)) \leq \min(\Pi(P), \triangledown(P)) \quad \text{s.t. } P \subseteq \Omega \tag{2.7}$$

Example 2.1 *Consider a user preference to plan his vacations. Specifications relate on values of 2 binary decision features: Activities denoted by A with instantiations a = physical pastimes, $\bar{a}$ = cultural activities, and Season denoted by B with instantiations b = winter, $\bar{b}$ = summer. The set of discourse is composed of 2^2 configurations, each associated with a preference degree: $\pi(ab) = 0.3$, $\pi(a\bar{b}) = 0$, $\pi(\bar{a}b) = 1$, $\pi(\bar{a}\bar{b}) = 0.8$. Going on vacation in winter to undertake cultural activities is fully satisfactory ($\pi(\bar{a}b) = 1$). However, traveling in summer for performing physical activities is completely rejected ($\pi(a\bar{b}) = 0$). Distribution π and $1 - \pi$ are normalized. Let P define the proposition "Traveling in summer or looking for cultural events". We aim to*

[1]The claim $1 - \pi$ being normalized is equivalent to say that π is *anti-normalized* meaning that $\exists$ $\omega \in \Omega$ s.t. $\pi(\omega) = 0$.

calculate to which extent the proposition P is satisfactory. It corresponds to the maximum between preference degrees of configurations that model $\bar{a}$ or $\bar{b}$. By applying the equation 2.1, it appears that $\Pi(P) = \max(\pi(\bar{a}b), \pi(a\bar{b}), \pi(\bar{a}\bar{b})) = \max(0, 1, 0.8) = 1$. *This means that P is fully satisfactory. We seek to evaluate to which extent P is imperative. This amounts to find the priority related to P. By applying the equation 2.2, we find that* $N(P) = 1 - \pi(ab) = 0.7$. *This degree can also be calculated by means of the duality property of possibility measures, i.e.,* $N(P) = 1 - \Pi(\bar{P}) = 1 - \pi(ab) = 0.7$. *The guaranteed possibility degree of P equals* $\triangle(P) = \min(\pi(\bar{a}b), \pi(a\bar{b}), \pi(\bar{a}\bar{b})) = 0$. *This means that all models of P are not satisfactory. This verifies the inequality* $\triangle(P) \leq \Pi(P)$. *The potential necessity can be computed using the duality property, i.e.,* $\bigtriangledown(P) = 1 - \triangle(\bar{P}) = 1 - \pi(ab) = 0.7$. *The constraint 2.7, namely* $\max(N(P), \triangle(P)) \leq \min(\Pi(P), \bigtriangledown(P))$ *is satisfied, i.e.,* $\max(0.7, 0) \leq \min(1, 0.7)$.

2.3 Logical encoding of conditional preferences

Conditional preference statements can be equivalently expressed in different ways. One way consists of graphically encoding statements using networks as shown in the first chapter. The second way consists on representing statements using a set of constraints that form a possibilistic base. An agent may associate a priority degree to each proposition that indicates its will to attain it. The set of such constraints forms what is called a *prioritized base*. An agent might also express his preferences by providing levels of satisfaction associated to classes of configurations. The entailed constraints constitute a base designated by a *guaranteed possibility base*. A unique possibility distribution can be derived from each of these bases [Benferhat et al., 2002c].

Next two sections respectively discuss how to logically express preferences using the necessity and guaranteed possibility measures. They also describe how to obtain the possibility distribution underlying a possibilistic logic base encoding priorities, the possibility distribution describing satisfaction levels for different configurations. The last section details the translation of a prioritized base into a guaranteed possibility base and conversely.

2.3.1 Logical representation of possibility measures

A priority base Σ is made up of a finite set of formulas of the form (p_i, α_i), where p_i is a propositional formula and α_i is a priority degree in $(0, 1]$ [Benferhat et al., 1999b]. A weighted formula (p, α) means that p has a priority α and its counter-models are

satisfactory at most at level $1-\alpha$. The weight α is thus understood as a lower bound on the degree of priority $N(p)$. Propositions that are associated with a null degree of priority are not explicitly represented by the base. To formalize constraints about preferences of an agent, the conjunction of pairs (p_i, α_i) forms the base Σ as follows,

$$\Sigma = \{(p_i, \alpha_i), i \in [1, n]\} \tag{2.8}$$

2.3.2 From prioritized base to distribution π

A prioritized base, syntactically constituted by a set of formulas (p_i, α_i), can semantically be represented by a unique possibility distribution π_Σ. If we consider that Σ is composed of a single formula (p, α), then the preference degree associated with each configuration ω is evaluated based on the consistency of ω with p. If ω is a model p, then its preference degree should be equal to 1, i.e., $\pi(\omega) = 1$. In contrast, if ω falsifies p, which means that it satisfies $\neg p$, then it is associated to a preference degree such that the higher α is, the lower is $\pi(\omega)$. Particularly, if p has the highest priority, i.e., $(\alpha = 1)$, then the configuration ω such that $\omega \not\models p$, is rejected, i.e., $\pi(\omega) = 0$.

$$\forall \omega \in \Omega, \pi_{(p,\alpha)}(\omega) = \begin{cases} 1 & \text{if } \omega \models p \\ 1-\alpha & \text{if } \omega \not\models p \end{cases} \tag{2.9}$$

A prioritized base Σ can be seen as a well ordered partition composed of sets $S_1 \cup S_2 \cup \cdots \cup S_M$ where formulas in S_i have more priority than those in S_{i+1}. Partitions S_i for $1 \leq i \leq M$ can be attached to a prioritized base where each formula in S_i is associated with a degree α_i, such that $1 \geq \alpha_1 > \cdots > \alpha_M > 0$ [Benferhat et al., 2001a].

Generally, given $\Sigma = \{(p_i, \alpha_i), i = 1, n\}$, configurations that are in accordance with all propositions p_i are considered fully satisfactory. Otherwise, configurations are ranked with respect to the falsified proposition of the highest priority degree.

$$\forall \omega \in \Omega, \pi_\Sigma(\omega) = \begin{cases} 1 & \text{if } \omega \models p_i, \forall (p_i, \alpha_i) \in \Sigma \\ 1 - \max\{\alpha_i : (p_i, \alpha_i) \in \Sigma; \omega \models \bar{p_i}\} & \textit{otherwise} \end{cases} \tag{2.10}$$

Thus, the possibility distribution π_Σ of Σ over a configuration $\omega \in \Omega$ results from the combination of all elementary possibility distributions $\pi_{(p_i,\alpha_i)}(\omega)$ using the minimum operator [Dubois et al., 1987]. Equation 2.10 can be written more concisely as

$$\pi_\Sigma(\omega) = \min\{\pi_{\{(p_i,\alpha_i)\}}(\omega) : (p_i, \alpha_i) \in \Sigma\} \tag{2.11}$$

$\pi(\omega)$ is the greatest possibility distribution that satisfy the set of constraints $N(p_i) \geq \alpha_i$ where N is based on π.

Example 2.2 *Consider the prioritized base* $\Sigma = \{(l, 0.1), (\bar{s}, 0.4), (\bar{s} \vee e, 0.7), (s \vee \bar{e}, 0.6), (\bar{l} \vee \bar{e} \vee r, 0.9), (\bar{l} \vee e \vee r, 0.8), (l \vee \bar{e} \vee r, 0.5), (l \vee e \vee \bar{r}, 0.3)\}$. *The possibility distribution associated with* Σ *is detailed in Table 2.1. To entail elementary possibility distributions related to each formula in* Σ *we consider equation 2.10. For instance, let us consider the formula* $(\bar{l} \vee e \vee r, 0.8)$, *its associated elementary distribution over* Ω *is calculated such that configurations that model the proposition* $\bar{l} \vee e \vee r$ *are considered completely satisfactory and are attached to a preference degree equals to* 1, *while those that falsify* $\bar{l} \vee e \vee r$ *are reduced to a satisfaction degree equal to* $1 - 0.8 = 0.2$ *(see column 7 in Table 2.1). After computing possibility distributions of each formula in* Σ, *the minimum degree proposed by the elementary distributions is assigned for each configuration (see Equation 2.11) e.g.* $\pi(\omega_3 = sl\bar{e}r) = \min(1, 0.6, 0.3, 1, 1, 0.2, 1, 1) = 0.2$. *The last column in Table 2.1 contains the joint possibility distribution* π_Σ *on* Ω *based on* Σ.

Ω	$(l, .1)$	$(\bar{s}, .4)$	$(e \vee \bar{s}, .7)$	$(\bar{e} \vee s, .6)$	$(r \vee \bar{l} \vee \bar{e}$ $.9)$	$(r \vee \bar{l} \vee e, .8)$	$(r \vee l \vee \bar{e}, .5)$	$(\bar{r} \vee l \vee e, .3)$	$\pi_\Sigma(\omega)$
$\omega_0 = sler$	1	.6	1	1	1	1	1	1	.6
$\omega_1 = sle\bar{r}$	1	.6	1	1	.1	1	1	1	.1
$\omega_2 = sl\bar{e}r$	1	.6	.3	1	1	1	1	1	.3
$\omega_3 = sl\bar{e}\bar{r}$	1	.6	.3	1	1	.2	1	1	.2
$\omega_4 = s\bar{l}er$	.9	.6	1	1	1	1	1	1	.6
$\omega_5 = s\bar{l}e\bar{r}$	.9	.6	1	1	1	1	.5	1	.5
$\omega_6 = s\bar{l}\bar{e}r$	.9	.6	.3	1	1	1	1	.7	.3
$\omega_7 = s\bar{l}\bar{e}\bar{r}$	.9	.6	.3	1	1	1	1	1	.3
$\omega_8 = \bar{s}ler$	1	1	1	.4	1	1	1	1	.4
$\omega_9 = \bar{s}le\bar{r}$	1	1	1	.4	.1	1	1	1	.1
$\omega_{10} = \bar{s}l\bar{e}r$	1	1	1	1	1	1	1	1	1
$\omega_{11} = \bar{s}l\bar{e}\bar{r}$	1	1	1	1	1	.2	1	1	.2
$\omega_{12} = \bar{s}\bar{l}er$	.9	1	1	.4	1	1	1	1	.4
$\omega_{13} = \bar{s}\bar{l}e\bar{r}$	.9	1	1	.4	1	1	.5	1	.4
$\omega_{14} = \bar{s}\bar{l}\bar{e}r$	.9	1	1	1	1	1	1	.7	.7
$\omega_{15} = \bar{s}\bar{l}\bar{e}\bar{r}$	.9	1	1	1	1	1	1	1	.9

Table 2.1: Detailed computation of the possibility distribution π_Σ given a prioritized base Σ

A possibilistic base is associated with a level of inconsistency.

Definition 2.1 (Consistency and inconsistency of Σ) *The inconsistency degree of Σ is defined semantically by the equation*

$$Inc(\Sigma) = 1 - \max_{\omega \in \Omega}\{\pi_\Sigma(\omega)\} \tag{2.12}$$

and the consistency degree is obtained by complementarity

$$Cons(\Sigma) = 1 - Inc(\Sigma) = \max_{\omega \in \Omega}\{\pi_\Sigma(\omega)\} \tag{2.13}$$

Thus, in the normalized case, the consistency of Σ equals 1, i.e., $Cons(\Sigma) = 1$ when there exists at least one configuration that is fully satisfactory with $\pi(\omega) = 1$. Then, the inconsistency of the base is equal to 0, i.e., $Inc(\Sigma) = 0$. It can be shown that a priority base Σ is consistent if the classical base made of the propositions of Σ without their weights is consistent [Dubois et al., 1994]. When $Inc(\Sigma) > 0$ it means that there are *conflicting* priorities in the possibilistic base.

Example 2.3 *Consider the same prioritized base of Example 2.2. In this example we only use the formula with weights > 0.5 for simplicity. Let Σ^* this new base $\Sigma^* = \{(s \vee \bar{e}, 0.6), (\bar{s} \vee e, 0.7), (\bar{l} \vee e \vee r, 0.8), (\bar{l} \vee \bar{e} \vee r, 0.9)\}$. The possibility distribution π_{Σ^*} is constructed by considering columns 4 to 7 in Table 2.1. Preference degrees associated with configurations in Ω are constructed by combining elementary possibility distributions of formulas in question using the minimum operator : $\pi(\omega_0) = 1, \pi(\omega_1) = 0.1, \pi(\omega_2) = 0.3, \pi(\omega_3) = 0.2, \pi(\omega_4) = 1, \pi(\omega_5) = 1, \pi(\omega_6) = 0.3, \pi(\omega_7) = 0.3, \pi(\omega_8) = 0.4, \pi(\omega_9) = 0.1, \pi(\omega_{10}) = 1, \pi(\omega_{11}) = 0.2, \pi(\omega_{12}) = 0.4, \pi(\omega_{13}) = 0.4, \pi(\omega_{14}) = 1, \pi(\omega_{15}) = 1$. The prioritized base Σ^* is totally consistent since $Cons(\Sigma_*) = \max_{\omega \in \Omega} \{\pi_{\Sigma^*}(\omega)\} = 1$, its inconsistency level thus equals 0.*

2.3.3 Logical representation by guaranteed possibility measure

A guaranteed possibility base Γ is composed of a set of formulas of the form $[p_i, \alpha_i]$, where p_i is a proposition and α_i a preference degree in $[0, 1)$ that estimates the minimal degree for which p_i is satisfactory.

$$\Gamma = \{[p_i, \alpha_i], i = 1, n\} \tag{2.14}$$

The formula $[p, \alpha]$ encodes the claim: "I am satisfied with any configuration ω where p is true with a minimal degree equal to α". This means that each collection of configurations that make p true is associated with a guaranteed minimal preference degree α, i.e., $\triangle(p) \geq \alpha$. Thus ω is called *satisfaction base* since it guarantees that configurations in π are satisfactory at least to a degree α_i. Formulas expressing that an agent is not satisfied at all with a proposition, i.e., $[p_i, 0]$ are not mentioned in the base.

2.3.4 From guaranteed possibility base to a distribution π

From each formula $[p, \alpha] \in \Gamma$, a possibility distribution $\pi_{[p,\alpha]}$ can be associated. It is such that configurations ω_i that model p are satisfactory with a minimal preference degree equals to α, while those that falsify p are considered as not satisfactory, i.e., $\pi(\omega_i) = 0$.

$$\forall \omega \in \Omega, \pi_{[p,\alpha]}(\omega) = \begin{cases} \alpha & \text{if } \omega \models p \\ 0 & \text{if } \omega \not\models p \end{cases} \tag{2.15}$$

Note that this is the smallest possibility distribution that agrees with the constraint $\Delta(p) \geq \alpha$.

The resulting distribution δ_Γ is obtained as the disjunction of these elementary distributions.

$$\forall \omega \in \Omega, \pi_\Gamma(\omega) = \begin{cases} 0 & \text{if } \forall [p_i, \alpha_i] \in \Gamma, \omega \not\models p_i \\ \max\{\alpha : [p_i, \alpha_i] \in \Gamma\} & \text{if } \omega \models p_i \end{cases} \tag{2.16}$$

The generalization of Equation 2.15 to the whole base Γ is defined by the smallest distribution in agreement with constraints in $\Delta(p_i) > \alpha_i$ associated with Γ[2]. A configuration ω is satisfactory to a degree α if the highest degree of the formula $[p, \alpha]$ such that p models ω is equal to α. If ω falsifies all formulas of Γ, then ω presents no guarantee at all to be satisfactory and is associated with a degree $\pi = 0$ [Dubois et al., 1996]. The previous equation can be written more concisely by

[2]Note that the aggregation of possibility distributions in Equation 2.16 is in agreement with the modeling of satisfaction degrees by possibility distributions. Indeed, if all the models of p are satisfactory and all the models of q are satisfactory then obviously, all models of p *or* q are satisfactory.

$$\delta_\Gamma(\omega) = \max\{\pi_{\{[p_i,\alpha_i]\}}(\omega) : [p_i, \alpha_i] \in \Gamma\} \tag{2.17}$$

Example 2.4 details the procedure of inferring a guaranteed possibility base Γ from a given possibility distribution.

Example 2.4 *Let us consider the following guaranteed possibility base* $\Gamma = \{[r \vee \bar{l} \vee \bar{e}, 0.2], [r \vee \bar{l}, 0.3], [e \wedge \bar{l}, 0.4], [e \wedge r, 0.4], [\bar{s} \wedge \bar{l}, 0.4], [r \wedge \bar{s}, 0.4], [e \wedge \bar{l} \wedge s, 0.5], [e \wedge r \wedge s, 0.6], [\bar{e} \wedge r \wedge \bar{s}, 0.6], [\bar{e} \wedge \bar{l} \wedge \bar{s}, 0.7], [\bar{e} \wedge r \wedge \bar{s}, 0.7], [\bar{s} \wedge \bar{l} \wedge \bar{e} \wedge \bar{r}, 0.9], [\bar{s} \wedge l \wedge \bar{e} \wedge r, 1], [\top, 0.1]\}$*. Details about the computation of the possibility distribution inferred given* Γ *is explained in Table 2.2. For instance, the configuration* $\omega_2 = sl\bar{e}r$ *only satisfies propositions of the first, second and last formulas, namely* $\omega_2 \models r \vee \bar{l} \vee \bar{e}$, $\omega_2 \models r \vee \bar{l}$ *and* $\omega_2 \models \top$. *Thus,* $\pi_{[r \vee \bar{l} \vee \bar{e}, 0.2]}(\omega_2) = 0.2$, $\pi_{[r \vee \bar{l}(\omega_2), 0.3]} = 0.3$ *and* $\pi_{[\top, 0.1]}(\omega_2) = 0.1$. *The preference degree associated with* ω_2 *corresponds to the maximum between elementary degrees relative to each formula. Therefore,* $\delta_\Gamma(\omega_2) = \max(\pi_{[r \vee \bar{l} \vee \bar{e}, 0.2]}(\omega_2), \pi_{[r \vee \bar{l}, 0.3]}(\omega_2), \pi_{[\top, 0.1]}(\omega_2)) = 0.3$.

Ω	$r \vee \bar{l} \vee \bar{e}$	$r \vee \bar{l}$	$e \wedge \bar{l}$	$e \wedge r$	$\bar{s} \wedge \bar{l}$	$r \wedge \bar{s}$	$e \wedge \bar{l} \wedge s$	$e \wedge r \wedge s$	$\bar{e} \wedge r \wedge \bar{s}$	$\bar{e} \wedge \bar{l} \wedge \bar{s}$	$\bar{e} \wedge r \wedge \bar{s}$	$\bar{s} \wedge \bar{l} \wedge \bar{e} \wedge \bar{r}$	$\bar{s} \wedge l \wedge \bar{e} \wedge r$	$\top$,	δ_Γ
	0.2	**0.3**	**0.4**	**0.4**	**0.4**	**0.4**	**0.5**	**0.6**	**0.6**	**0.7**	**0.7**	**0.9**	**1**	**0.1**	
$\omega_0 = sler$	0.2	0.3	0	0.4	0	0	0	0.6	0	0	0	0	0	0.1	0.6
$\omega_1 = sle\bar{r}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0.1
$\omega_2 = sl\bar{e}r$	0.2	0.3	0	0	0	0	0	0	0	0	0	0	0	0.1	0.3
$\omega_3 = sl\bar{e}\bar{r}$	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0.2
$\omega_4 = s\bar{l}er$	0.2	0.3	0.4	0.4	0	0	0.5	0.6	0	0	0	0	0	0.1	0.6
$\omega_5 = s\bar{l}e\bar{r}$	0.2	0.3	0.4	0	0	0	0.5	0	0	0	0	0	0	0.1	0.5
$\omega_6 = s\bar{l}\bar{e}r$	0.2	0.3	0	0	0	0	0	0	0	0	0	0	0	0.1	0.3
$\omega_7 = s\bar{l}\bar{e}\bar{r}$	0.2	0.3	0	0	0	0	0	0	0	0	0	0	0	0.1	0.3
$\omega_8 = \bar{s}ler$	0.2	0.3	0	0.4	0	0.4	0	0	0	0	0	0	0	0.1	0.4
$\omega_9 = \bar{s}le\bar{r}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0.1
$\omega_{10} = \bar{s}l\bar{e}r$	0.2	0.3	0	0	0	0.4	0	0	0.6	0	0.7	0	1	0.1	1
$\omega_{11} = \bar{s}l\bar{e}\bar{r}$	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0.2
$\omega_{12} = \bar{s}\bar{l}er$	0.2	0.3	0.4	0.4	0.4	0.4	0	0	0	0	0	0	0	0.1	0.4
$\omega_{13} = \bar{s}\bar{l}e\bar{r}$	0.2	0.3	0.4	0	0.4	0	0	0	0	0	0	0	0	0.1	0.4
$\omega_{14} = \bar{s}\bar{l}\bar{e}r$	0.2	0.3	0	0	0.4	0.4	0	0	0.6	0.7	0.7	0	0	0.1	0.7
$\omega_{15} = \bar{s}\bar{l}\bar{e}\bar{r}$	0.2	0.3	0	0	0.4	0	0	0	0	0.7	0	0.9	0	0.1	0.9

Table 2.2: Detailed computation of the possibility distribution δ_Γ given a guaranteed possibility base Γ

2.3.5 From prioritized base to satisfaction base and back

Possibility measures can compactly represent a set of preference specifications which allows to construct a prioritized base Σ or a guaranteed possibility base Γ encoding the same possibility distribution on the set of alternatives, i.e., $\pi_\Sigma = \delta_\Gamma$ [Benferhat et al., 2002c]. In this section, we discuss the transformation of a base Σ into a base Γ, and conversely, such that both of these bases encode the same input information.

Before getting deeper into the subject, let us notice that given a guaranteed possibility base Σ that contains the formulas $[p, \alpha]$ and $[q, \beta]$, then $[p, \alpha]$ is considered as *subsumed* by $[q, \beta]$ as soon as $\alpha \leq \beta$ and $p \models q$.

From prioritized base to satisfaction base

Given a prioritized base Σ, we aim to construct a guaranteed possibility base Γ that both encode the same information, i.e., yield the same possibility distribution $\pi_\Sigma = \delta_\Gamma$.

First, let us consider a base Σ composed of a single formula : (p, α) (see Equation 2.9). Notice that all configurations have a minimal satisfaction degree equal to $1 - \alpha$ which yields the formula $[\top, 1 - \alpha]$. In order to ensure that configurations that satisfy p be associated with a maximal degree of satisfaction, we must add the formula $[p, 1]$ to Γ. Thus, the satisfaction base corresponding to $\Sigma = \{(p, \alpha)\}$ is

$$\Gamma = \{[p, 1], [\top, 1 - \alpha]\}.$$

Including another formula (q, β) in Σ such that $\alpha > \beta$ results in additional interpretations that should be considered for Γ. The preference degree associated with a configuration ω is obtained by considering the propositions that it satisfies. Three cases exists:

$$\pi_\Sigma(\omega) = \begin{cases} 1 & \text{if } \omega \models p \wedge q \\ 1 - \alpha & \text{if } \omega \models \bar{p} \\ 1 - \beta & \text{if } \omega \models p \wedge \bar{q} \end{cases}$$

All configurations have a minimal preference degree equal to $1 - \alpha$, which leads to include the formula $[\top, 1 - \alpha]$ in Γ. Solutions that verify both propositions p and q are associated with the maximal preference degree, which results in adding the formula $[p \wedge q, 1]$ in Γ. Configurations that satisfy neither p nor q are considered the

least satisfactory and are associated with the minimal satisfaction degree $1-\alpha$. This constraint is insured by the formula $[\top, 1-\alpha]$. Finally, configurations that satisfy p but not q, p being more prioritary than q, are associated with an intermediate preference degree equals to $1-\beta$, resulting in including $[p \wedge \bar{q}, 1-\beta]$ in Γ. Hence, the guaranteed possibility base corresponding to $\Sigma = \{(p,\alpha),(q,\beta)\}$ is

$$\Gamma = \{[p \wedge q, 1], [p \wedge \bar{q}, 1-\beta], [\top, 1-\alpha]\}$$

The generalization of the above equation is given in the following proposition,

Proposition 2.1 *Let $\Sigma = \{(p_i, \alpha_i) : i = 1, \ldots M\}$ be a prioritized base where each formula (p_i, α_i) corresponds to a level i where $1 \leq i \leq M$, $\alpha_i > \alpha_{i+1}$ and $\alpha_{M+1} = 0$. We define from Σ a guaranteed possibility base as follows*

$$\Gamma = \{[p_1 \wedge \cdots \wedge p_i, 1-\alpha_{i+1}] : i = 1, \ldots, M\} \cup \{[\top, 1-\alpha_1]\}. \tag{2.18}$$

In following we give an example of entailing a possibility distribution from a priority base.

Example 2.5 *Let us reconsider the prioritized base in Example 2.2 namely $\Sigma = \{(l, 0.1), (\bar{s}, 0.4), (\bar{s} \vee e, 0.7), (s \vee \bar{e}, 0.6), (\bar{l} \vee \bar{e} \vee r, 0.9), (\bar{l} \vee e \vee r, 0.8), (l \vee \bar{e} \vee r, 0.5), (l \vee e \vee \bar{r}, 0.3)\}$. The first step is to arrange the base in a decreasing order of priority degrees of its formulas. Thus, $\Sigma = \{(\bar{l} \vee \bar{e} \vee r, 0.9), (\bar{l} \vee e \vee r, 0.8), (\bar{s} \vee e, 0.7), (s \vee \bar{e}, 0.6), (l \vee \bar{e} \vee r, 0.5), (\bar{s}, 0.4), (l \vee e \vee \bar{r}, 0.3)(l, 0.1), \}$. The second step consists on applying Equation 2.18. Details are in the sequel:*

- $[\bar{l} \vee \bar{e} \vee r, 1-0.8]$
- $[(\bar{l} \vee \bar{e} \vee r) \wedge (\bar{l} \vee e \vee r), 1-0.7]$ *which yields* $[r \vee \bar{l}, 0.3]$
- $[(\bar{l} \vee \bar{e} \vee r) \wedge (\bar{l} \vee e \vee r) \wedge (e \vee \bar{s}), 1-0.6]$ *which yields* $[e \vee \bar{l}, 0.4]$, $[e \vee r, 0.4]$, $[e \vee r, 0.4]$, $[\bar{l} \vee \bar{s}, 0.4]$, $[r \vee \bar{s}, 0.4]$
- $[(\bar{l} \vee \bar{e} \vee r) \wedge (\bar{l} \vee e \vee r) \wedge (e \vee \bar{s}) \wedge (\bar{e} \vee s), 1-0.5]$ *which yields* $[\bar{e} \vee r \vee \bar{s}, 0.5]$, $[e \vee \bar{l} \vee s, 0.5]$, $[e \vee r \vee s, 0.5]$, $[e \vee r \vee s, 0.5]$, $[\bar{e} \vee \bar{l} \vee \bar{s}, 0.5]$
- $[(\bar{l} \vee \bar{e} \vee r) \wedge (\bar{l} \vee e \vee r) \wedge (e \vee \bar{s}) \wedge (\bar{e} \vee s) \wedge (r \vee l \vee \bar{e}), 1-0.4]$ *which yields* $[e \vee r \vee s, 0.6]$, $[\bar{e} \vee \bar{l} \vee \bar{s}, 0.6]$, $[\bar{e} \vee r \vee \bar{s}, 0.6]$

- $[(\bar{l} \vee \bar{e} \vee r) \wedge (\bar{l} \vee e \vee r) \wedge (e \vee \bar{s}) \wedge (\bar{e} \vee s) \wedge (r \vee l \vee \bar{e}) \wedge \bar{s}), 1 - 0.3]$ *which yields* $[\bar{e} \vee \bar{l} \vee \bar{s}, 0.7]$, $[\bar{e} \vee r \vee \bar{s}, 0.7]$

- $[(\bar{l} \vee \bar{e} \vee r) \wedge (\bar{l} \vee e \vee r) \wedge (e \vee \bar{s}) \wedge (\bar{e} \vee s) \wedge (r \vee l \vee \bar{e}) \wedge \bar{s} \wedge r \vee \bar{l} \vee \bar{e}), 1 - 0.1]$ *which yields* $[\bar{e} \wedge r \wedge l \wedge \bar{r}, 0.9]$, $[\bar{e} \wedge \bar{r} \wedge \bar{l} \wedge \bar{r}, 0.9]$

- $[(\bar{l} \vee \bar{e} \vee r) \wedge (\bar{l} \vee e \vee r) \wedge (e \vee \bar{s}) \wedge (\bar{e} \vee s) \wedge (r \vee l \vee \bar{e}) \wedge \bar{s} \wedge r \vee \bar{l} \vee \bar{e}), 1 - 0]$ *which yields* $[\bar{e} \wedge r \wedge l \wedge \bar{r}, 1]$

- $[\top, 0.1]$.

After deleting all subsumed formulas, the guaranteed possibility base associated with Σ *is* $\Gamma = \{[r \vee \bar{l} \vee \bar{e}, 0.2], [r \vee \bar{l}, 0.3], [e \wedge \bar{l}, 0.4], [e \wedge r, 0.4], [\bar{s} \wedge \bar{l}, 0.4], [r \wedge \bar{s}, 0.4], [e \wedge \bar{l} \wedge s, 0.5], [e \wedge r \wedge s, 0.6], [\bar{e} \wedge r \wedge \bar{s}, 0.6], [\bar{e} \wedge \bar{l} \wedge \bar{s}, 0.7], [\bar{e} \wedge r \wedge \bar{s}, 0.7], [\bar{s} \wedge \bar{l} \wedge \bar{e} \wedge \bar{r}, 0.9], [\bar{s} \wedge l \wedge \bar{e} \wedge r, 1], [\top, 0.1]\}$. *Both of these bases are equivalent since they infer the same possibility distribution (see Tables 2.1 and 2.2).*

From satisfaction base to prioritized base

The aim of this section is to construct prioritized base Σ from a guaranteed possibility base Γ such that both bases induce the same possibility distribution, i.e., $\pi_\Sigma = \delta_\Gamma$.

We first start by considering a base Γ composed of a single formula $[p, \alpha]$. Following the Equation 2.15, the possibility distribution associated with Γ is

$$\forall \omega \in \Omega, \delta_\Gamma(\omega) = \begin{cases} \alpha & \text{if } \omega \models p \\ 0 & \text{if } \omega \not\models p \end{cases} \tag{2.19}$$

Due to the first constraint, the collection of configurations that model p have a maximum preference degree equal to α, leading the possibility distribution π_Σ to be lower bounded by $1 - \alpha$. Therefore, this distribution is inconsistent to a level $1 - \alpha$. To unsure this property, the formula $(\bot, 1 - \alpha)$ needs to be added to the prioritized base Σ. The distribution π_Σ must hold an upper bound preference degree equal to 1 in order to guarantee that configurations that falsify p are prioritized to the highest degree. This is ensured by the formula $(p, 1)$. The prioritized base corresponding to $\Gamma = \{[p, \alpha]\}$ is

$$\Sigma = \{(p, 1), (\bot, 1 - \alpha)\}$$

Including another formula $[q, \beta]$ to Γ such that $\alpha > \beta$ results in additional interpretations that should be considered for Σ. The preference degree associated with a configuration ω is obtained by considering the propositions that it satisfies. In alike manner as for the converse transformation, three cases can take place:

$$\delta_\Gamma(\omega) = \begin{cases} \alpha & \text{if } \omega \models p \wedge q \\ \beta & \text{if } \omega \models \bar{p} \\ 0 & \text{if } \omega \models p \wedge \bar{q} \end{cases}$$

In fact, all configurations verifying both p and q have a maximum priority equal to $1 - \alpha$, which leads to include the formula $(\bot, 1 - \alpha)$ in Γ. Solutions that model p but not q are are satisfactory to a degree β, which results in adding the formula $(p \vee \bar{q}, 1 - \beta)$ in Γ. Finally, configurations that do not satisfy neither p nor q are considered totally rejected and are associated with a priority degree equal to 1. This constraint is insured by the formula $(p \vee q, 1)$. Hence, the prioritized possibility base corresponding to $\Gamma = \{[p, \alpha], [q, \beta]\}$ is

$$\Sigma = \{[p \vee q, 1], [p \vee \bar{q}, 1 - \beta], [\bot, 1 - \alpha]\} \tag{2.20}$$

The following proposition [Benferhat et al., 2002c] generalizes the above equation.

Proposition 2.2 *Let $\Gamma = \{[p_i, \alpha_i] : i = 1, \ldots M\}$ be a guaranteed possibility base where each formula $[p_i, \alpha_i]$ corresponds to a level i where $1 \leq i \leq M$, $\alpha_i > \alpha_{i+1}$ and $\alpha_{M+1} = 0$. We define from Γ a prioritized base as follows*

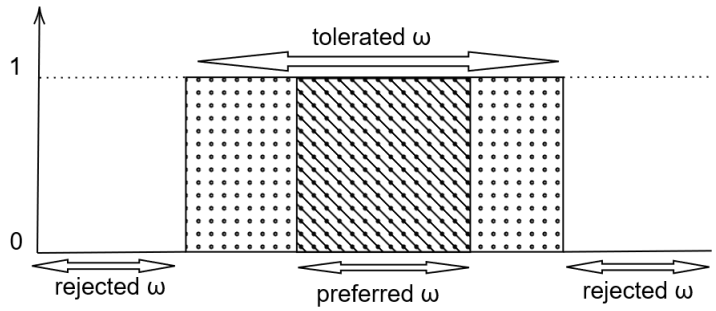
$$\Sigma = \{[p_1 \vee \cdots \vee p_i, 1 - \alpha_{i+1}] : i = 1, \ldots, M\} \cup \{[\bot, 1 - \alpha_1]\}.$$

An agent can express his preferences by means of priority constraints using the N measure or by means of satisfaction constraints using the Δ measure. Given a base of priority constraints Σ_π we can infer a distribution π on Ω from which can be entailed a satisfaction base Γ_π and vise versa. Bases Γ_π and Σ_π encode exactly the same distribution on configurations.

2.3.6 Bipolar preferences

Preferences over a set of possible configurations can often be expressed in terms of bipolar information. An agent may express his desires by associating degrees

of satisfaction to solutions, but also by stating that it is rejecting some alternatives. What is really satisfactory for the agent is thus positively assessed. This type of preferences is called *positive* and is logically encoded by Δ using constraints $Sat = \{\Delta(p_i) \geq \alpha_i : i = 1, \ldots, m\}$ where p_i is a propositional formula and α_i is its corresponding level of satisfaction. Configurations that are not explicitly declared as rejected are considered as tolerated. Solutions that are not totally rejected are some what tolerated and can be represented by a set of prioritized constraints of the form $Tol = \{N(q_j) \geq 1 - \beta_j : j = 1, \ldots, m\}$ where $1 - \beta_j$ is a priority degree. Such statements are called *negative* preferences [Benferhat et al., 2006]. Configurations which are positively preferred are should be tolerated which means that the set of *positive* preferences should be included into the set of *negative* ones (see below figure for illustration), and more generally the possibility distributions δ and π respectively associated with the sets Sat and Tol should be such that $\forall\omega, \delta(\omega) \leq \pi(\omega)$.



2.4 Possibilistic networks and the relation to possibilistic bases

In the possibility theory, preferences can be represented by logic bases or by graphs and both semantically induce the same possibility distribution on configurations. A possibility distribution can be decomposed using either the product chain rule, when using a numerical framework, or the minimum operation when preferences are qualitative. These two decompositions correspond to the two types of conditioning recalled in the next Section 2.4.1; they induce two kinds of possibilistic graphs [Benferhat et al., 2002a]. In the following, we describe the translation procedures between graphical and logical frameworks for both qualitative and quantitative networks. But first we recall possibilistic networks in the perspective of modeling uncertain knowledge. At the end of the chapter in Section 3.2, we shall present their transposing in preference networks.

2.4.1 Conditioning

Conditional possibility is defined similarly to probability theory using the Bayesian rule. It actually depends on the preference configuration range, whether it is ordinal, e.g., totally ordered chain, or numerical, e.g., a scale from 0 to 1. This leads to two different forms of conditioning for qualitative and quantitative possibility. The conditioning rule can be defined by the equation $\Pi(P \cap Q) = \Pi(Q \mid P) \square \Pi(P)$ where $\square$ is the minimum or the product operator [Dubois and Prade, 1990].

If we are dealing with qualitative information, *min-based conditioning* is used and is defined as follows:

$$\Pi(P \mid Q) = \begin{cases} 1, & \text{if } \Pi(P \cap Q) = \Pi(P) > 0. \\ \Pi(P \cap Q), & \text{otherwise.} \end{cases} \tag{2.21}$$

In a quantitative numerical setting, the *product-based conditioning*, similar to the probabilistic conditioning, is used and is defined by:

$$\Pi(P \mid Q) = \frac{\Pi(P \cap Q)}{\Pi(Q)} \tag{2.22}$$

2.4.2 Possibilistic networks

Possibilistic networks [Benferhat et al., 2002a] are counterparts of probabilistic causal networks. They are a noteworthy alternative to the latter when it comes to representing uncertain information. Possibilistic networks are based on the decomposition of a joint possibility distribution as a combination of conditional possibility distributions. Decision variables are depicted by nodes, each associated with a possibility distribution that express the agent's knowledge on the underlying variables. Local distributions must be normalized, i.e., there must exist at least one instantiation of underlying variable that is totally possible. Nodes are connected by means of edges that reflect influence links between them. These networks permit to express ignorance by associating the highest degree of possibility to all values of the variable in question. It might be more convenient for an expert to express his knowledge about the world by providing sure beliefs instead of possible ones. For this purpose, conditional local distributions can be replaced by conditional necessities, i.e., $\Pi(x|U_X) = 1 - N(\bar{x}|U_X)$, due to the duality property between possibility and necessity measures.

Definition 2.2 (Possibilistic network) *A possibilistic network over a set of variables $\mathcal{X}$ is characterized by two components*

(i) a graphical structure $\mathcal{G} = (\mathcal{X}, E)$ consisting of a DAG expressing dependency between decision variables;

(ii) a set CT of conditional tables where each node $X \in \mathcal{X}$ is attached to a conditional table $CT(X)$ that correspond to a local normalized conditional possibility distribution $\pi(X|U_X)$.

As mentioned in Section 2.4.1, there exist two kinds of conditioning, which engender two types of possibilistic graphs. The notation ΠG_m refer to the qualitative counterpart of possibilistic networks where the joint possibility distribution is calculated based on the minimum operator. A ΠG^* corresponds to a quantitative possibilistic network where the joint possibility distribution is based on the product operator. The value 0 being the absorbing element of the product operator, we assume that $0 < \pi(X|U_X) \leq 1$. Given a possibilistic network ΠG, there exists a unique joint possibility distribution π over configurations in Ω that is calculated using the following chain rule:

$$\pi(X_1, ..., X_N) = \square_{i=1}^{N} \Pi(X_i|U_{X_i}) \tag{2.23}$$

where $\square$ can take either the minimum or the product operator.

2.4.3 Encoding ΠG in possibilistic logic

The first step of translating a possibilistic network into a prioritized base is to consider each local distribution as a set of 3-tuples composed of (x, u, α) that encode the piece of knowledge $\pi(x|u) = \alpha$ with $\alpha < 1$. A possibilistic network can thus be represented by a set of triples,

$$\Pi G = \{(x_i, u_i, \alpha_i) : \alpha_i = \Pi(x_i|u_i)\}$$

where $x_i \in \underline{X_i}$, $u_i \in \underline{U_{X_i}}$ and $\forall X_i \in \mathcal{X}$. Pieces of knowledge such that $\alpha = 1$ are totally possible are not included in this set.

Each tuple (x, u, α) is translated into the formula $(x \vee \bar{u}, 1 - \alpha)$ that constitutes the logical base $\Sigma_{\Pi G}$. Given a single triple $(x, u, \alpha) \in \Pi G$, the joint distribution of a complete solution ω is

$$\forall \omega \in \Omega, \pi_{(x,u,\alpha)}(\omega) = \begin{cases} 1 & \text{if } \omega \models x \vee \bar{u} \\ \alpha & \text{otherwise} \end{cases} \tag{2.24}$$

In fact, formulas associated to each variable or node of the graph compose an elementary base. The combination of these bases form the prioritized base associated with the ΠG network.

$$\begin{aligned} \Sigma_{\Pi G} &= \{(x_i \vee \bar{u_i}, 1 - \alpha_i) : \Pi(x_i \mid u_i) = \alpha_i \in \Pi G, \alpha_i \leq 1\} \\ &= \{(x_i \vee \bar{u_i}, 1 - \alpha_i) : (x_i, u_i, \alpha_i) \in \Pi G, \alpha \leq 1\} \end{aligned} \tag{2.25}$$

Logical encoding of ΠG_m

Let ΠG_m denote the qualitative specialisation of ΠG. The joint possibility distribution π associated with ΠG_m is obtained by combining possibility distributions π_i of each tuple (x_i, u_i, α_i) by means of the minimum operator [Benferhat et al., 2002a]. Equivalently, the result of fusing all elementary bases associated with each node of the graph corresponds to the possibilistic distribution π associated with the network. Indeed, let Σ_X and $\Sigma_{X'}$ be two prioritized bases associated with possibility distributions π_X and $\pi_{X'}$, respectively for $\{X, X'\} \in \mathcal{X}$. The base $\Sigma_{XX'}$ resulting from combining Σ_X and $\Sigma_{X'}$ is [Benferhat et al., 2000]

$$\Sigma_{XX'} = \Sigma_X \cup \Sigma_{X'}$$

Given ΠG_m, the above equation is generalized as follows,

$$\begin{aligned} \Sigma_{\Pi G_m} &= \bigcup_{i=1}^{N} \Sigma_{X_i} \\ &= \{(x_i \vee \bar{u_i}, 1 - \alpha) : (x_i, u_i, \alpha) \in \Pi G_m, \alpha \neq 1\} \quad \text{s.t. } u_i \in \underline{U_{X_i}} \text{ and } X_i \in \mathcal{X} \end{aligned} \tag{2.26}$$

We now provide an example about encoding the ΠG in Figure 2.1 by a prioritized base.

Example 2.6 *Let us consider the possibilistic network in Figure 2.1. The network is written under the set of formulas* $\Pi G_m = \{(a, \emptyset, 0.4), (b, \bar{a}, 0.9), (\bar{b}, a, 0.6), (c, \bar{a}, 0.7),$

$(\bar{c}, a, 0.3), (d, \bar{b}\bar{c}, 0.6), (\bar{d}, bc, 0.7), (\bar{d}, b\bar{c}, 0.8), (\bar{d}, \bar{b}c, 0.2)\}$. *Using Equation 2.26, the prioritized base associated with the* ΠG_m *when using the minimum operator is* $\Sigma_{\Pi G_m} = \Sigma_A \bigcup \Sigma_B \bigcup \Sigma_C \bigcup \Sigma_D = \{(\bar{a}, 0.6), (\bar{a} \vee b, 0.4), (a \vee \bar{b}, 0.1), (c \vee \bar{a}, 0.7), (\bar{c} \vee a, 0.3), (d \vee \bar{b} \vee \bar{c}, 0.3), (d \vee \bar{b} \vee c, 0.2), (d \vee b \vee \bar{c}, 0.8), (\bar{d} \vee b \vee c, 0.4)\}$.

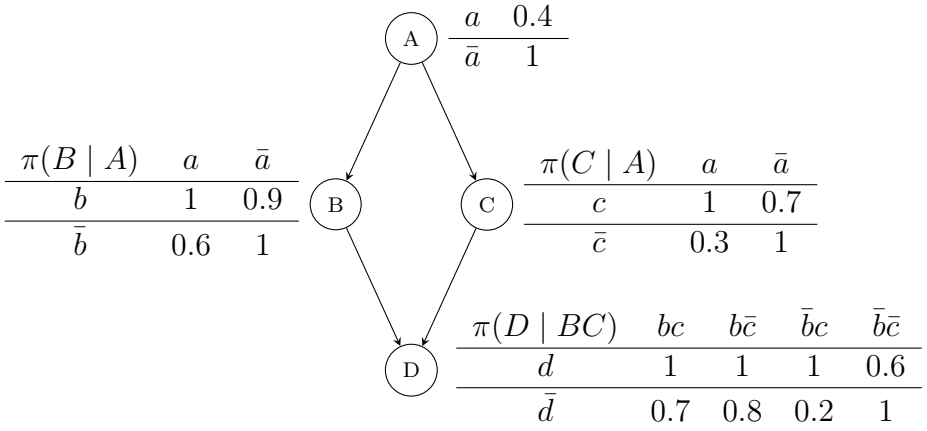


Figure 2.1: Example of a ΠG

Logical encoding of ΠG^*

As for qualitative possibilistic networks, when considering a quantitative ΠG, we are still able to construct a prioritized base that encodes the network. Following the same steps as for min-based possibilistic network, the idea is to first consider each local distribution and its entailed base. The joint possibility distribution computed from a product-based network ΠG^* corresponds to the fusion of all elementary bases using the product operator. The joint possibility distribution π^* of ΠG^* is the same as the one obtained by combining all π_{X_i} using the product [Benferhat et al., 2002a]. Let $\Sigma_{\Pi G^*}$ define the prioritized base inferred from combining Σ_X and $\Sigma_{X'}$ using the product operator. Then $\Sigma_{\Pi G^*}$ is constructed using the following equation [Benferhat et al., 2000],

$$\Sigma_{\Pi G^*} = \Sigma_X \cup \Sigma_{X'} \cup \{(x_i \vee x'_j, \alpha_i + \beta_j - \alpha_i \times \beta_j), i \in I, j \in J, x_i \vee x'_j \neq \top\} \quad (2.27)$$

The possibilistic base inferred from a min-based ΠG is obviously smaller than a base derived from a product-based ΠG. This is explained by the fact that the encoding procedure of a ΠG^* adds a set of formulas with intermediate levels to the prioritized base $\Sigma_{\Pi G^*}$, formulas expressing knowledge not explicitly mentioned in the original knowledge bases.

The following Example 2.7 illustrates the procedure of transforming product-based ΠG into a prioritized base $\Sigma_{\Pi G^*}$.

Example 2.7 *Let us reconsider the graph in Figure 2.1 that corresponds to the set of triples* $\Pi G^* = \{(a, \emptyset, 0.4), (b, \bar{a}, 0.9), (\bar{b}, a, 0.6), (c, \bar{a}, 0.7), (\bar{c}, a, 0.3), (d, \bar{b}\bar{c}, 0.6), (\bar{d}, bc, 0.7), (\bar{d}, b\bar{c}, 0.8), (\bar{d}, \bar{b}c, 0.2)\}$. *The network is associated with the elementary bases*
$\Sigma_A = \{(\bar{a}, 0.6)\}$,
$\Sigma_B = \{(\bar{a} \vee b, 0.4), (a \vee \bar{b}, 0.1)\}$,
$\Sigma_C = \{(c \vee \bar{a}, 0.7), (\bar{c} \vee a, 0.3)\}$,
$\Sigma_D = \{(d \vee \bar{b} \vee \bar{c}, 0.3), (d \vee \bar{b} \vee c, 0.2), (d \vee b \vee \bar{c}, 0.8), (\bar{d} \vee b \vee c, 0.4)\}$.
Step by step, we proceed by subsequently fusing two bases at a time. Combining Σ_A *and* Σ_B *generates the following base:*

$$\begin{aligned}\Sigma_{AB} &= \Sigma_A \cup \Sigma_B \cup \{(\bar{a} \vee b, 0.76)\} \\ &= \{(\bar{a}, 0.6), (\bar{a} \vee b, 0.4), (a \vee \bar{b}, 0.1), (\bar{a} \vee b, 0.76)\} \\ &= \{(\bar{a}, 0.6), (a \vee \bar{b}, 0.1), (\bar{a} \vee b, 0.76)\}\end{aligned}$$

The formula $(\bar{a} \vee b, 0.4)$ *is removed since it is subsumed by* $(\bar{a} \vee b, 0.76)$. *Combining* Σ_{AB} *and* Σ_C *generates the following base:*

$$\begin{aligned}\Sigma^*_{ABC} &= \Sigma_{AB} \cup \Sigma_C \cup \{(\bar{a} \vee c, 0.88), (a \vee \bar{b} \vee \bar{c}, 0.37), (\bar{a} \vee b \vee c, 0.93)\} \\ &= \{(\bar{a}, 0.6), (a \vee \bar{b}, 0.1), (\bar{a} \vee b, 0.76), (\bar{c} \vee a, 0.3), (\bar{a} \vee c, 0.88), (a \vee \bar{b} \vee \bar{c}, 0.37), \\ &\quad (\bar{a} \vee b \vee c, 0.93)\}\end{aligned}$$

Combining Σ_{ABC} *and* Σ_D *generates the prioritized base associated with the the* ΠG^* *in Figure 2.1:*

$$\begin{aligned}\Sigma^*_{\Pi G^*} = \Sigma_{ABC}, \Sigma_D = \Sigma_{ABC} \cup \Sigma_D \cup \{&(a \vee b \vee c \vee d, 0.28), (a \vee \bar{b} \vee \bar{c} \vee d, 0.56), \\ &(\bar{a} \vee b \vee c \vee d, 0.99), (\bar{a} \vee b \vee c \vee \bar{d}, 0.96), (\bar{a} \vee \bar{b} \vee c \vee d, 0.9), \\ &(\bar{a} \vee \bar{b} \vee \bar{c} \vee d, 0.72)\} \\ = \{&(\bar{a}, 0.6), (a \vee \bar{b}, 0.1), (\bar{a} \vee b, 0.76), (\bar{c} \vee a, 0.3), (\bar{a} \vee c, 0.88), \\ &(a \vee \bar{b} \vee \bar{c}, 0.37), (\bar{a} \vee b \vee c, 0.93), (a \vee b \vee c \vee d, 0.28), \\ &(a \vee \bar{b} \vee \bar{c} \vee d, 0.56), (\bar{a} \vee b \vee c \vee d, 0.99), (\bar{a} \vee b \vee c \vee \bar{d}, 0.96), \\ &(\bar{a} \vee \bar{b} \vee c \vee d, 0.9), (\bar{a} \vee \bar{b} \vee \bar{c} \vee d, 0.72)\}\end{aligned}$$

We have seen that in the possibility theory setting, we have different formats for representing information, namely a possibility distribution, a necessity-based logic base, a Δ-based logic base, a possibilistic graph ΠG with min-based conditioning and ΠG with product-based conditioning [Benferhat et al., 2001c]. We have described how to go from possibility distribution to logic bases and vice-versa and how to go from logic bases to possibility distribution in 2.3.2 and 2.3.4. We have also showed how to go from a possibilistic graph to logic bases in 2.4.3: one can go from a ΠG_m [Benferhat et al., 1999a] to a possibilistic logic base (see 2.4.3), and from a ΠG^* [Benferhat et al., 2001b] to a possibilistic logic base (see 2.4.3).

2.5 Conclusion

In possibility theory, information can logically be encoded in different formats: prioritized logic base, satisfaction logic base, possibility distribution and possibilistic networks. All of them describe the same information but some formats are more appropriate for different ways of expressing preferences or may have computational advantages [Benferhat et al., 2001c].

A remarkable feature of possibility theory is that it offers the advantage of encoding the same information in different ways. Specifications of a user can be logically encoded by possibilistic bases. The same set of statements could be compactly depicted by possibilistic networks in a graphical manner. Translating possibilistic bases into networks and conversely can be performed while preserving the same ordering on configurations. In the next chapter, we discuss π-pref nets which exploit possibilistic graphs for preference representation.

Chapter 3

Possibilistic Preference Networks: Basis, Comparisons and Variants

3.1 Introduction

This chapter is dedicated to a recently introduced model situated half-way between qualitative and quantitative representations called *possibilistic preference network* (π-pref net for short). π-pref nets are based on possibilistic networks, as CP-nets are inspired by Bayesian probabilistic nets. As we shall see, π-pref nets avoid the bias of CP-nets that privilege preferences associated with father nodes (due to ceteris paribus assumption). As CP-nets, a π-pref net structure enables us to express conditional preference statements and offers a compact model for elicitation and representation.

This chapter is organized in three main sections. It both contains a background part and also proposes new developments in Section 3.4. Next section gives background about π-pref nets and describes the various ways to exploit them. Section 3.3 discusses the expressiveness and consistency of possibilistic preference networks with regard to CP-nets. Section 3.4 introduces new variants of π-pref nets using different scales for encoding preference degrees, where the top and bottom elements 1 or 0 play or not a role.

Throughout all the chapter, we continue with the same running example, dealing with decision problem about renting or buying a car, which was introduced in Example 1.4 in Chapter 1 and already used in Chapter 2.

3.2 π-pref nets

A possibilistic preference network [Ben Amor et al., 2015] (π-pref net for short) shares the same graphical structure as a CP-net $\mathcal{C}$ (see Definition 1.7). They however differ by their informational components. In fact, to each preference statement $u : x \succ x'$ we associate a local conditional possibility distribution $\pi(X \mid U_X)$ using symbolic weights expressing an ordering between the values in $\underline{X}$. The symbolic weights are unspecified degrees, assumed to be in the real interval $(0, 1]$[1]. In each context u, there must exist a preferred instantiation of X associated with a degree equal to 1. A π-pref net can be defined as a ΠG network where possibility degrees correspond to symbolic weights.

Definition 3.1 (Possibilistic preference network) *A possibilistic preference network $\mathcal{Z} = \langle \mathcal{G}, CPT \rangle$, denoted by π-pref net, over a set of decision variables $\mathcal{X}$ is composed of*

(i) a graphical structure $\mathcal{G} = (\mathcal{X}, E)$ consisting of a DAG expressing dependency between decision variables,

(ii) a set CPT of conditional preference tables such that to each node $X \in \mathcal{X}$ is attached a conditional table $CPT(X)$ that associates preference degrees to each value $x \in \underline{X}$ in the context of each possible value u of parents U_X.

The network obeys to the Markovian assumption which stipulates that each variable X is independent from other variables in the subset $\mathcal{Y} = \{\mathcal{X} \setminus U_X \setminus \{X\}\}$ in the context of its parents (U_X). Consider a variable X with $\underline{X} = \{x, \bar{x}\}$, the possibility degree $\pi(x \mid u)$ evaluates the satisfaction degree of the value x in context u. $\pi(x \mid u) = 1$ iff x is preferred, otherwise $\pi(x \mid u)$ takes a symbolic degree α such that $\alpha < 1$. The symbolic weights appearing in different contexts have no reason to be equal: the violation of a preference may be a source of more dissatisfaction in one context than in another. However, we generally use a unique symbolic weight per value and context Apart from comparisons entailed from specifications in the network, constraints between symbolic weights can be added when available [Ben Amor et al., 2018a].

Let us illustrate the notion of π-pref net in the following Example 3.1.

[1]We are excluding the case where the symbolic weight might be equal to 0, since we intend to represent the kind of conditional preferences that are handled by CP-nets, where conditional rejection is not considered. Indeed, in CP-nets we can not express that in a given context some variables value(s) is/are not acceptable at all.

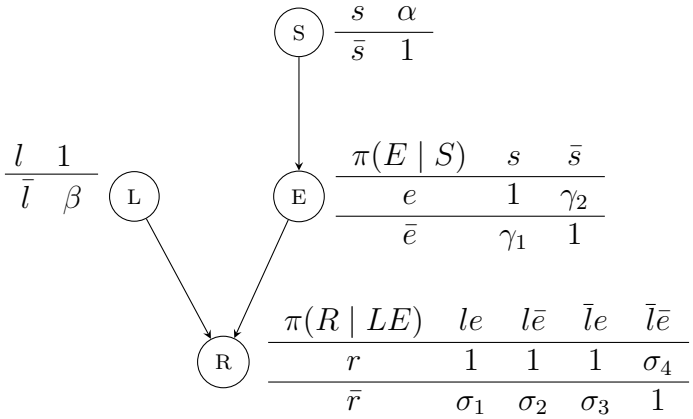


Figure 3.1: Example of a symbolic π-pref net with different symbolic weights per variable and context value

Example 3.1 *Figure 3.1 depicts an example of a π-pref net without additional constraints between symbolic weights, about the car choice problem of Example 1.1. Preferences specifications are encoded by the network by associating a preference degree equals to* 1 *for the more satisfactory value in the context of parents,and a symbolic degree otherwise. For example since the user prefers a luxury (l) car over a modest one ($\bar{l}$), therefore, $\pi(l) = 1$ and $\pi(\bar{l}) = \beta$, where β is a positive number strictly less to* 1*.*

3.2.1 Chain rule

In order to find a ranking over configurations in Ω, we need to calculate the degree of satisfaction of each $\omega_i \in \Omega$ by means of a chain rule. If we consider their quantitative counterpart, the product-based chain rule is used. It is formally expressed by:

$$\pi(X_1, ..., X_N) = \prod_{i=1}^{N} \pi(X_i|U_{X_i}) \, . \tag{3.1}$$

From the other side, if we consider their qualitative counterpart, the minimum operator is used and the chain rule is formally written as follows:

$$\pi(X_1, ..., X_N) = \min_{i=1}^{N} \pi(X_i|U_{X_i}) \tag{3.2}$$

When no additional constraints on symbolic weights are added, due to the drowning effect of the minimum operator (values above the minimum values are 'lost'), the

product-based joint possibility distribution permits to order configurations in a more discriminant and refined way than the minimum-based distribution.

We continue Example 3.1 by providing the symbolic weights computing the product-based chain rule.

Example 3.2 *Table 3.1 provides calculation details about π-pref net in Figure 3.1. For each configuration ω, Columns 2 to 5 present conditional preference degrees associated with each decision variable in the graph. The last column gives the satisfaction degree of each configuration based on the product operator.*

$\boldsymbol{\Omega}$	$\boldsymbol{\pi(S)}$	$\boldsymbol{\pi(L)}$	$\boldsymbol{\pi(E\|S)}$	$\boldsymbol{\pi(R\|LE)}$	$\boldsymbol{\vec{\omega}}$	$\boldsymbol{\pi(\omega)}$
$\omega_0 = sler$	α	1	1	1	$(\alpha, 1, 1, 1)$	α
$\omega_1 = sle\bar{r}$	α	1	1	σ_1	$(\alpha, 1, 1, \sigma_1)$	$\alpha\sigma_1$
$\omega_2 = sl\bar{e}r$	α	1	γ_1	1	$(\alpha, 1, \gamma_1, 1)$	$\alpha\gamma_1$
$\omega_3 = sl\bar{e}\bar{r}$	α	1	γ_1	σ_2	$(\alpha, 1, \gamma_1, \sigma_2)$	$\alpha\gamma_1\sigma_2$
$\omega_4 = s\bar{l}er$	α	β	1	1	$(\alpha, \beta, 1, 1)$	$\alpha\beta$
$\omega_5 = s\bar{l}e\bar{r}$	α	β	1	σ_3	$(\alpha, \beta, 1, \sigma_3)$	$\alpha\beta\sigma_3$
$\omega_6 = s\bar{l}\bar{e}r$	α	β	γ_1	σ_4	$(\alpha, \beta, \gamma_1, \sigma_4)$	$\alpha\beta\gamma_1\sigma_4$
$\omega_7 = s\bar{l}\bar{e}\bar{r}$	α	β	γ_1	1	$(\alpha, \beta, \gamma_1, 1)$	$\alpha\beta\gamma_1$
$\omega_8 = \bar{s}ler$	1	1	γ_2	1	$(1, 1, \gamma_2, 1)$	γ_2
$\omega_9 = \bar{s}le\bar{r}$	1	1	γ_2	σ_1	$(1, 1, \gamma_2, \sigma_1)$	$\gamma_2\sigma_1$
$\omega_{10} = \bar{s}l\bar{e}r$	1	1	1	1	$(1, 1, 1, 1)$	1
$\omega_{11} = \bar{s}l\bar{e}\bar{r}$	1	1	1	σ_2	$(1, 1, 1, \sigma_2)$	σ_2
$\omega_{12} = \bar{s}\bar{l}er$	1	β	γ_2	1	$(1, \beta, \gamma_2, 1)$	$\beta\gamma_2$
$\omega_{13} = \bar{s}\bar{l}e\bar{r}$	1	β	γ_2	σ_3	$(1, \beta, \gamma_2, \sigma_3)$	$\beta\gamma_2\sigma_3$
$\omega_{14} = \bar{s}\bar{l}\bar{e}r$	1	β	1	σ_4	$(1, \beta, 1, \sigma_4)$	$\beta\sigma_4$
$\omega_{15} = \bar{s}\bar{l}\bar{e}\bar{r}$	1	β	1	1	$(1, \beta, 1, 1)$	β

Table 3.1: Joint possibility distribution of configurations in Ω covered by π-pref net in Figure 3.1

As can be seen in the above example, a product of symbolic weights is associated with each configuration. This product induce only a partial order since, e.g., $\beta\gamma_2 > \beta\gamma_2\sigma_3$, but σ_2 and $\beta\gamma_2$ cannot be compared for instance. However, in case we have complementary information about the relative values of satisfaction degrees under the form of inequalities between symbolic weights, we can perform further comparisons, e.g., $\alpha\sigma_1 > \alpha\gamma_1$ if we know that $\sigma_1 > \gamma_1$.

Besides, in case we use a unique satisfaction degree for the preferences associated to a given node, i.e., in Table 3.1, $\gamma_1 = \gamma_2$, $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4$, we are able to

compare more configurations and the partial order obtained is in agreement with the cardinality order (see 1.2.2). However, the latter is a complete pre-order where configurations having the same number of violations are ties, while such configurations are incomparable in the π-pref net partial order.

In any case, each complete configuration $\omega \in \Omega$ can be described by a quality vector $\vec{\omega} = (\rho_1, \ldots, \rho_N)$, where each symbol ρ_j represents the satisfaction degree $\pi(x_j|u_j)$, such that $x_j \in \underline{X_j}$, $u_j \in \underline{U_{X_j}}$ and $j \in [1, N]$. The possibility degree $\pi(\omega)$ is just the product of components in the vector.

The following Example 3.3 illustrates the case of a π-pref net with equal symbolic weights per variable and parent context.

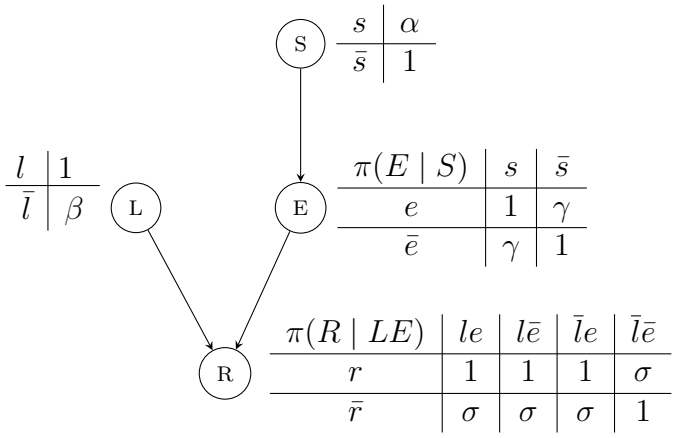


Figure 3.2: Example of a symbolic π-pref net with one symbolic weight per variable

Example 3.3 *Let us consider the π-pref net in Figure 3.2 which encodes the same preferences as the π-pref net in Figure 3.1. Nevertheless, the first network contains different symbols per variable while the second contains equal symbols per variable. Table 3.2 provides calculation details about the π-pref net in Figure 3.2. Each configuration ω is associated with a vector of symbolic weights and a satisfaction degree computed from Equation 3.1. When symbolic weights are equal in context of parents value, the chain rule induces more comparisons between configurations. This is due to the fact that there are less incomparable symbolic weights. For instance, ω_5 and ω_6 that were associated with the respective preference degrees $\alpha\beta\sigma_3$ and $\alpha\beta\gamma_1\sigma_4$ were incomparable because σ_3 and σ_4 are incomparable. They are now associated with degrees $\alpha\beta\sigma$ and $\alpha\beta\gamma\sigma$ which makes them comparable such that $\omega_5 \succ_{Prod} \omega_6$, since $\pi(\omega_5) = \alpha\beta\sigma > \pi(\omega_6) = \alpha\beta\gamma\sigma$. All comparisons of the π-pref net in Figure 3.2 are given in Figure 3.3.*

Ω		$\pi(S)$	$\pi(L)$	$\pi(E\|S)$	$\pi(R\|LE)$	$\vec{\omega}$	$\pi(\omega)$
$\omega_0 =$	$sler$	α	1	1	1	$(\alpha, 1, 1, 1)$	α
$\omega_1 =$	$sle\bar{r}$	α	1	1	σ	$(\alpha, 1, 1, \sigma)$	$\alpha\sigma$
$\omega_2 =$	$sl\bar{e}r$	α	1	γ	1	$(\alpha, 1, \gamma, 1)$	$\alpha\gamma$
$\omega_3 =$	$sl\bar{e}\bar{r}$	α	1	γ	σ	$(\alpha, 1, \gamma, \sigma)$	$\alpha\gamma\sigma$
$\omega_4 =$	$s\bar{l}er$	α	β	1	1	$(\alpha, \beta, 1, 1)$	$\alpha\beta$
$\omega_5 =$	$s\bar{l}e\bar{r}$	α	β	1	σ	$(\alpha, \beta, 1, \sigma)$	$\alpha\beta\sigma$
$\omega_6 =$	$s\bar{l}\bar{e}r$	α	β	γ	σ	$(\alpha, \beta, \gamma, \sigma)$	$\alpha\beta\gamma\sigma$
$\omega_7 =$	$s\bar{l}\bar{e}\bar{r}$	α	β	γ	1	$(\alpha, \beta, \gamma, 1)$	$\alpha\beta\gamma$
$\omega_8 =$	$\bar{s}ler$	1	1	γ	1	$(1, 1, \gamma, 1)$	γ
$\omega_9 =$	$\bar{s}le\bar{r}$	1	1	γ	σ	$(1, 1, \gamma, \sigma)$	$\gamma\sigma$
$\omega_{10} =$	$\bar{s}l\bar{e}r$	1	1	1	1	$(1, 1, 1, 1)$	1
$\omega_{11} =$	$\bar{s}l\bar{e}\bar{r}$	1	1	1	σ	$(1, 1, 1, \sigma)$	σ
$\omega_{12} =$	$\bar{s}\bar{l}er$	1	β	γ	1	$(1, \beta, \gamma, 1)$	$\beta\gamma$
$\omega_{13} =$	$\bar{s}\bar{l}e\bar{r}$	1	β	γ	σ	$(1, \beta, \gamma, \sigma)$	$\beta\gamma\sigma$
$\omega_{14} =$	$\bar{s}\bar{l}\bar{e}r$	1	β	1	σ	$(1, \beta, 1, \sigma)$	$\beta\sigma$
$\omega_{15} =$	$\bar{s}\bar{l}\bar{e}\bar{r}$	1	β	1	1	$(1, \beta, 1, 1)$	β

Table 3.2: Joint possibility distribution of configurations in Ω covered by π-pref net in Figure 3.2

3.2.2 Ordering quality vectors

Given a π-pref net, several procedures can be used for this comparing configurations, such as: Product, Minimum, Pareto, symmetric Pareto, Discrimin and Leximin orders. In the sequel, we give formal definitions of these orderings that may be used for the dominance query.

Definition 3.2 (Product) *$\forall \omega \neq \omega' \in \Omega$ associated to distinct vectors $\vec{\omega} = (\rho_1, \ldots, \rho_N)$ and $\vec{\omega'} = (\rho'_1, \ldots, \rho'_N), \omega \succeq_{Prod} \omega'$ iff $prod(\vec{\omega}) \geq prod(\vec{\omega'})$ such that $prod(\vec{\omega}) = \prod_{i=1}^{N} \rho_i$.*

Definition 3.3 (Minimum) *$\forall \omega \neq \omega' \in \Omega$ associated to distinct vectors $\vec{\omega} = (\rho_1, \ldots, \rho_N)$ and $\vec{\omega'} = (\rho'_1, \ldots, \rho'_N), \omega \succeq_{min} \omega'$ iff $\min(\vec{\omega}) \geq \min(\vec{\omega})$ such that $\min(\vec{\omega}) = \min_{i=1}^{N} \rho_i$.*

Definition 3.4 (Pareto) *$\forall \omega \neq \omega' \in \Omega$ associated to distinct vectors $\vec{\omega} = (\rho_1, \ldots, \rho_N)$ and $\vec{\omega'} = (\rho'_1, \ldots, \rho'_N), \omega \succ_{Pareto} \omega'$ iff $\forall i = 1, N, \rho_i \geq \rho'_i$ and for some $\ell, \rho_\ell = 1 > \rho'_\ell$.*

Definition 3.5 (Symmetric Pareto) *$\forall \omega \neq \omega' \in \Omega$ associated to distinct vectors $\vec{\omega} = (\rho_1, \ldots, \rho_N)$ and $\vec{\omega'} = (\rho'_1, \ldots, \rho'_N), \omega \succ_{SP} \omega'$ iff there exists a permutation f of symbolic weights positions of $\vec{\omega}$ yielding another vector $\vec{\omega_f} = (\rho_{f(1)}, \ldots, \rho_{f(N)})$ such that $\omega_f \succ_{Pareto} \omega'$.*

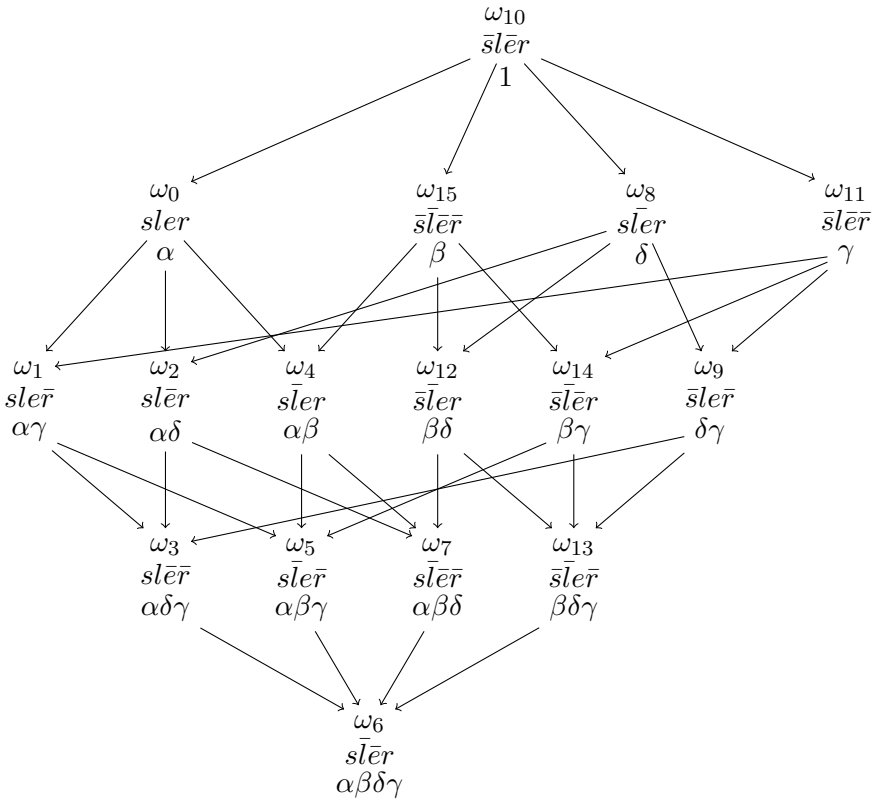


Figure 3.3: Induced graph of π-pref net in Figure 3.2 based on the product chain rule

Definition 3.6 (Discrimin) *$\forall \omega \neq \omega' \in \Omega$ associated to distinct vectors $\vec{\omega} = (\rho_1, \ldots, \rho_N)$ and $\vec{\omega'} = (\rho'_1, \ldots, \rho'_N)$, delete equal components in $\vec{\omega}, \vec{\omega'}$ such that $\rho_i = \rho'_i$. Let $d \in D$ denotes a variable index for whom $\omega[X_d] \neq \omega'[X_d]$ where D is the subset of the indices remaining in configuration vectors. $\omega \succ_{discrimin} \omega'$ iff $\min_{i \in D} \rho_i > \min_{i \in D} \rho'_i$.*

Definition 3.7 (Leximin) *$\forall \omega \neq \omega' \in \Omega$ associated to distinct vectors $\vec{\omega} = (\rho_1, \ldots, \rho_N)$ and $\vec{\omega'} = (\rho'_1, \ldots, \rho'_N), \omega \succ_{leximin} \omega'$ iff there exists a permutation f of $\vec{\omega}$'s components such that $\omega_f \succ_{discrimin} \omega'$.*

The following example considers sets of pairs of configurations and compares them using the mentioned orders.

Example 3.4 *Consider two configurations ω_0, ω_1 in the set of feasible solution Ω relative to network in Figure 3.1. $\vec{\omega_0} = (\alpha, 1, 1, 1)$ and $\vec{\omega_1} = (\alpha, 1, 1, \gamma_1)$ represent the respective vectors of weights of configurations ω_0 and ω_1. $\omega_0 \succ_{prod} \omega_1$ since $\alpha > (\alpha \times \gamma_1)$. If symbols are not instantiated, the minimum semantic can not rank order these configurations since α is less or equal than $\min(\alpha, \gamma_1)$ (drowning effect). Considering the Pareto order, $\omega_0 \succ_{Pareto} \omega_1$ since for variables (S), (L) and (E) satisfaction degrees*

are equal, except for (R) *for which* $\omega_0[R] = 1 > \omega_1[R] = \gamma_1$. *Consider configurations* ω_2 *and* ω_{15} *with respective vectors* $\vec{\omega_2} = (\alpha, 1, \delta_1, 1)$ *and* $\vec{\omega_{15}} = (1, \beta, 1, 1)$. *Deleting equal preference degrees yields vectors* $\vec{\omega_2} = (\alpha, 1, \delta_1)$ *and* $\vec{\omega_{15}} = (1, \beta, 1)$. *Suppose we know that* $\delta_1 < \beta$ *and* $\delta_1 < \alpha$. *Based on the minimum procedure* $\min(\alpha, 1, \delta_1) = \delta_1$ *while* $\min(1, \beta, 1) = \beta$ *which makes* $\omega_{15} \succ_{discrimin} \omega_2$. *Configurations* ω_3 *and* ω_7 *are encoded by vectors* $\vec{\omega_3} = (\alpha, 1, \delta_1, \gamma_2)$ *and* $\vec{\omega_7} = (\alpha, \beta, \delta_1, 0)$. *Let* $\omega_{7(f)} = (\alpha, 1, \delta_1, \beta)$ *be a permutation of weights in* $\vec{\omega_7}$ *(*f *is a permutation function).* $\omega_{7(f)} \succ_{leximin} \omega_3$ *since* $\omega_{7(f)} \succ_{discrimin} \omega_3$ *due to the fact that* $\min(\alpha, \delta_1, \beta) < \min(\alpha, \delta_1, \gamma_2)$ *if we know that* $\gamma_2 < \beta$.

π-pref nets have a qualitative and quantitative counterpart. When symbolic weights express qualitative preferences, the *Minimum* semantic corresponds to the min-based chain rule mentioned in Equation 3.2. If preferences are described by numerical degrees, the *Product* order coincides with the product-chain rule defined in Equation 3.1. With no additional constraints on weights and if symbolic degrees are not instantiated, order semantics lead to a sparsely discriminant ordering between solutions since dominance relations between weights are unknown. We say that an ordering $\succ$ over elements in Ω refines $\succ'$ if and only if for all pairs of configurations $(\omega_i, \omega_j) \in \Omega$ if $\omega_i \succ \omega_j$ then $\omega_i \succ' \omega_j$. If symbolic degrees take numerical values, [Dubois et al., 1996] have presented proofs about refinements between orderings of strategies. Figure 3.4 from [Ben Amor et al., 2016b] sums up the deduced properties. An edge from a box A to B ($A \rightarrow B$) means that A refines B.

When the satisfaction of preferences are assessed by symbolic degrees and no additional constraints are specified on them, the Pareto strategy and the product order lead to the same ordering [Ben Amor et al., 2015] [Ben Amor et al., 2016b] (see Figure 3.4(b)). This ordering corresponds to the order induced by comparing the sets of violated variables using inclusion. Indeed, let $\mathcal{V}(\omega)$ and $\mathcal{V}(\omega')$ be the set of variables that are set to their least preferred values for configurations ω and ω' configurations respectively. If $\mathcal{V}(\omega) \subset \mathcal{V}(\omega')$, then ω strictly dominates ω' based on both *Pareto* and *Product* strategies. Without any constraint on the weights, all the order strategies mentioned above lead to equivalent orders, except for the *minimum* case which is less refined [Ben Amor et al., 2016b].

When additional constraints on symbolic degrees are provided, the *product* and *symmetric Pareto* yield the same ordering. Refinements between orders are given in Figure 3.4(c). When the *symmetric Pareto* ranks a pair of configurations as incomparable, the *minimum* strategy may succeed in finding a strict dominance relationship between them. This relation is represented by a dotted arrow in Figure 3.4(c).

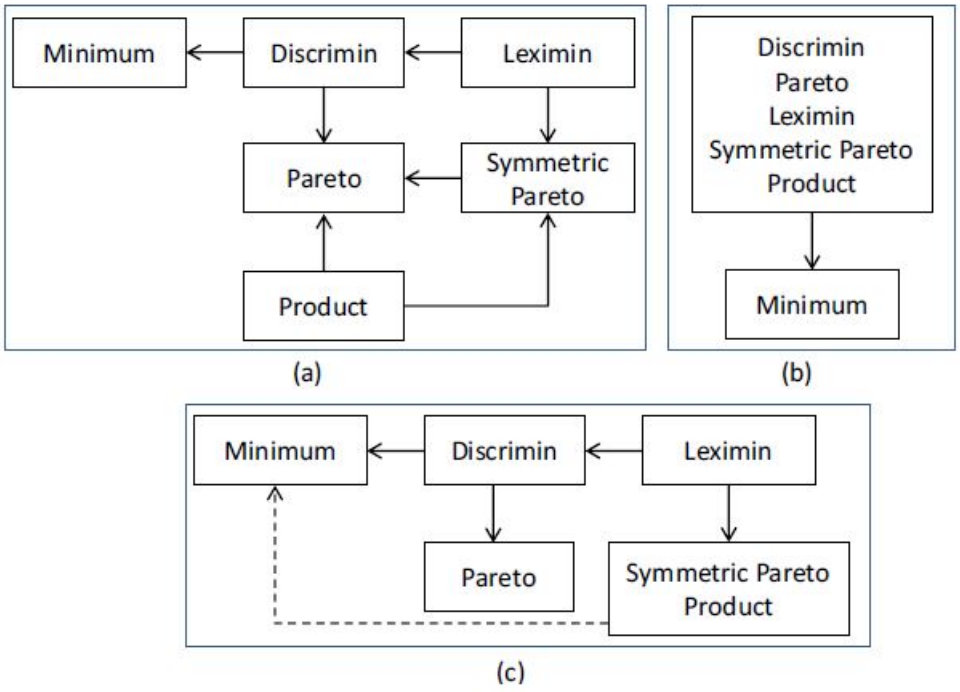


Figure 3.4: Refinements between ordering strategies (a) for instantiated numerical degrees, (b) for symbolic degrees without additional constraints and (c) for symbolic degrees and additional constraints on them

Not surprisingly, when symbolic degrees are instantiated with numerical values, the order *product* induces the most discriminant order, while the *minimum* represents the weakest order. Figure 3.4(a) summarises refinements between order semantics. Proofs are given in [Ben Amor et al., 2016b].

Consider a π-pref net model and a pair of different configuration (ω, ω'). Given a possibilistic setting, if $\omega \succ_{Prod} \omega'$ then $\omega \succ_{Pareto} \omega'$ [Ben Amor et al., 2017a]. Without additional constraints, the minimum is a poorly discriminant order strategy that yields an ordered set composed of only two levels with the optimal configuration being classified as better than all other ones in Ω [Ben Amor et al., 2016b]. Consider orderings $\succ_{Prod}$ and $\succ_{Pareto}$ entailed from a π-pref net. A configuration dominates another with respect to a π-pref net if its preference degree is higher than the one associated with the other configuration. Formally, $\omega \succ_{Prod} \omega'$ iff $\pi(\omega) \succ_{Prod} \pi(\omega')$. This case happens when one or more variable values are instantiated to a preference degree equal to 1 for ω, whereas the rest of variables are instantiated to the same symbolic degrees for both configurations. Taking in consideration Pareto as an order strategy, $\omega \succ_{Pareto} \omega'$ iff (i) there exists at least one symbolic weight $\rho > \rho'$ for $\rho \in \vec{\omega}$ and $\rho' \in \vec{\omega'}$ such that $\pi(X_i|U_{X_i}) = \rho$ and (ii) all other weights are better or equal. Since no additional constraints are considered and due to the local normalization property, then it is clear

that $\rho = 1$ and $\rho' < 1$. This represents the unique case for which ω Pareto dominates ω'. Thus we can conclude that if $\omega \succ_{Pareto} \omega'$ then $\omega \succ_{Prod} \omega'$ and vice-versa, formally

$$\omega \succ_{Pareto} \omega' \Longleftrightarrow \omega \succ_{Prod} \omega'$$

Each configuration can be characterized in term of a set of its satisfied variables. We use $\mathcal{S}(\omega)$[2] to define the described set for a given configuration $\omega \in \Omega$. Since quantitative π-pref nets reproduce the exact same ordering as Pareto, we are interested in using sets $\mathcal{S}(\omega)$ as a bridge to prove that these networks are not in dis-accordance with the ceteris paribus assumption. In fact, $\mathcal{S}(\omega)$ only depends on the set of satisfied decision variables, then if $\omega \succ_{Prod} \omega'$ according to the π-pref net $\mathcal{S}(\omega') \subset \mathcal{S}(\omega)$. Formally

$$\omega \succ_{Prod} \omega' \Longrightarrow \mathcal{S}(\omega') \subset \mathcal{S}(\omega)$$
$$\omega \succ_{Pareto} \omega' \Longrightarrow \mathcal{S}(\omega') \subset \mathcal{S}(\omega)$$

This property is necessary but not sufficient since two configurations can be comparable in term of the inclusion order between subsets $\mathcal{S}$ but incomparable based on Pareto [Ben Amor et al., 2017a]. Without additional constraints on symbolic weights, two given configurations that violate the preference of a node X given different contexts are incomparable since symbolic degrees pertaining to X are different. Formally, if $\omega[X] = \rho$ and $\omega'[X] = \rho'$ for $\rho, \rho' \in (0, 1)$, then $\omega \bowtie_{Pareto} \omega'$.

Example 3.5 *Consider configurations* ω_{13} *and* ω_{15} *with respective vectors* $\vec{\omega_{13}} = (1, \beta, \delta_2, \gamma_3)$ *and* $\vec{\omega_{15}} = (1, \beta, 1, 1)$, $\omega_{15} \succ_{Pareto} \omega_{13}$ *since all preference degrees in* ω_{15} *are higher or equal to those of* ω_{13}. *Based on the inclusion ordering between satisfied subsets,* $\mathcal{S}(\omega_{13}) = (S) \subset \mathcal{S}(\omega_{15}) = (SER)$. *The latter strategy does indeed recover the comparison engendered using Pareto semantic. However, configurations* ω_9 *and* ω_{11} *with respective vectors* $\vec{\omega_9} = (1, 1, \delta_1, \gamma_2)$ *and* $\vec{\omega_{11}} = (1, 1, \delta_2, 1)$, *are incomparable based on Pareto, since the inclusion order, unlike the Pareto order, does not take into account variable's positions but considers sets of satisfied variables as a whole. Based on inclusion, we have* $\omega_{11} \succ \omega_9$ *since* $\mathcal{S}(\omega_9) = (SL) \subset S(\omega_{11}) = (SLE)$.

[2]For a matter of representation, we use parentheses instead of braces and omit to separate variables by commas to define the subset $\mathcal{S}$.

3.2.3 Querying π-pref nets

Most used queries for interrogating a preference model are dominance and optimization queries. A product-based possibilistic network can express uncertain or incomplete information using numerical degrees. When numerical degrees over preferences can be provided, we can use such network for representing preferences and process optimisation and dominance procedures. We now describe them for symbolic π-pref nets.

Optimization query

π-pref nets allow the user to express indifference by enabling him to assign the highest satisfaction degree to both binary values of a decision variable. For that specific reason, the optimization query may return more that one configuration all associated to a possibility degree equal to 1. For this query, optimal configuration(s) are always found, since local possibility distribution associated to features are normalized by imposing to one of the variable values to be fully satisfactory. Thus, the joint possibility distribution associated to configurations covered by the network is normalized whatever the operator of the chain rule is. Graphically, finding optimal configuration(s) amounts to sweeping through the graph nodes while assigning to each variable the value associated with the highest preference degree in context of parents. This procedure is linear in the size of decision variables [Ben Amor et al., 2018a].

Example 3.6 *Considering joint possibility distribution in Table 3.1 and whatever the chain rule operator is, there exists one optimal configuration* ω_{Opt} *corresponding to a degree of satisfaction equal to* 1*, namely* ω_{10}*. Graphically, the optimal configuration is easily detected by assigning their preferred values to decision variables as follows* $S = \bar{s}$ *since* $\pi(\bar{s}) = 1$*,* $L = l$ *since* $\pi(l) = 1$*,* $E = \bar{e}$ *since* $\pi(\bar{e}|\bar{s}) = 1$ *and* $R = r$ *since* $\pi(r|l\bar{e}) = 1$*, thus* $\omega_{Opt} = \omega_{10}$*.*

Dominance query

Finding a dominance relation between a pair of configurations in a π-pref net amounts to comparing their relative vectors of weights or to compare product of symbolic weights. The dominance query in π-pref nets is linear but answering all dominance queries between all pairs of configurations to find an ordering over them has an exponential complexity of $O(N!)$ [Ben Amor et al., 2018a].

3.3 π-pref nets vs CP-nets

The induced graph in Figure 3.5 shows that π-pref nets and CP-nets do not lead to the same dominance relation over configurations of the possible states of the world. This is due to the fact that they do not share the same independence property. Although both graphical models are based on the same collection of user preference statements, however, they semantically use different strategies for comparing pairs of solutions. Indeed, the induced ordering generated from the ceteris paribus assumption is based on single flips between configurations to which is applied a transitive closure to capture all possible comparisons. On the other hand, π-pref nets use the product operator to rank order solutions and the joint possibility distribution can be calculated for getting the ordering between all configurations (quality vectors may eventually be used). In the following, we discuss the consistency between these two models and we recall inequalities between products of symbolic weights to be added in order to lead to a good approximation of a CP-net by a π-pref net [Ben Amor et al., 2018a].

3.3.1 Consistency between π-pref nets and CP-nets

Considering a CP-net $\mathcal{C}$, configurations are ranked based on a sequence of worsening flips according to ceteris paribus assumption. Let (ω, ω') be two configurations that differ by a single flip value on variable X. If $\omega \succ_{\mathcal{C}} \omega'$, then $\mathcal{S}(\omega) \subset \mathcal{S}(\omega')$ is not possible [Ben Amor et al., 2016b]. In fact, if $\omega \succ_{\mathcal{C}} \omega'$ then there exists a worsening flips from ω to ω' which means that ω holds an additional satisfied variable compared to ω' which is X. By consequence, $\mathcal{S}(\omega) \subset \mathcal{S}(\omega')$ is never true. This conclusion can be further extended to all ceteris paribus dominance relations since when considering the sequence of worsening flips, the status of each flipped variable will not be questioned by the later flips.

As recalled in Chapter 1, CP-nets agree with the Pareto ordering [Wilson et al., 2019]. Thus, if we compare a CP-net and a π-pref net (without additional constraints), induced by the same set of preference statements, then all the comparisons made by the π-pref net will be acknowledged by the CP-net. This is illustrated by the following example.

Example 3.7 *Figure 3.5 graphically represents comparisons of the π-pref net in Figure 3.1 based on the inclusion / Pareto or product order. Two adjacent configurations differ by only one flip and are described by the product of their respective weights. Each configuration is associated to its preference degree and a subset of its non-violated vari-*

ables. Considering the induced CP-net graph in Figure 1.3, solid arrows in Figure 3.5 depict comparisons induced by both ceteris paribus and inclusion, Pareto and product semantics. Arrows in dotted lines show comparisons that the CP-net gives but the inclusion ordering does not. We can compare two different solutions by comparing the product of their respective vectors e.g configuration ω_0 associated with a preference degree $\pi(\omega_0)$ equals to α dominates configuration ω_4 with $\pi(\omega_4) = \alpha\beta$ since $\alpha > \alpha\beta$, or simply $\alpha \in \{\alpha, \beta\}$.

No contradictions are observed in Figure 3.5 between orderings induced from a π-pref net and its corresponding CP-net. However, $\omega_3 = sl\bar{e}\bar{r} \succ_{\mathcal{C}} \omega_7 = s\bar{l}er$ since they differ by the value of (L) s.t. $\omega_3[L] = l$, $\omega_7[L] = \bar{l}$ and $l \succ \bar{l}$. The corresponding π-pref net fails to catch this relation. This can easily be checked by Pareto strategy where considering vectors $\vec{\omega_3} = (\alpha, 1, \delta_1, \gamma_2)$ and $\vec{\omega_7} = (\alpha, \beta, \delta_1, 1)$, we have $\omega_3[L] \succ \omega_7[L]$ while $\omega_7[R] \succ \omega_3[R]$ which leads to an incomparability case. From the other hand, $\omega_0 \succ_{Pareto} \omega_2$ but these configurations are incomparable according to the ceteris paribus assumption, which means that no flipping chain rule can be found between them.

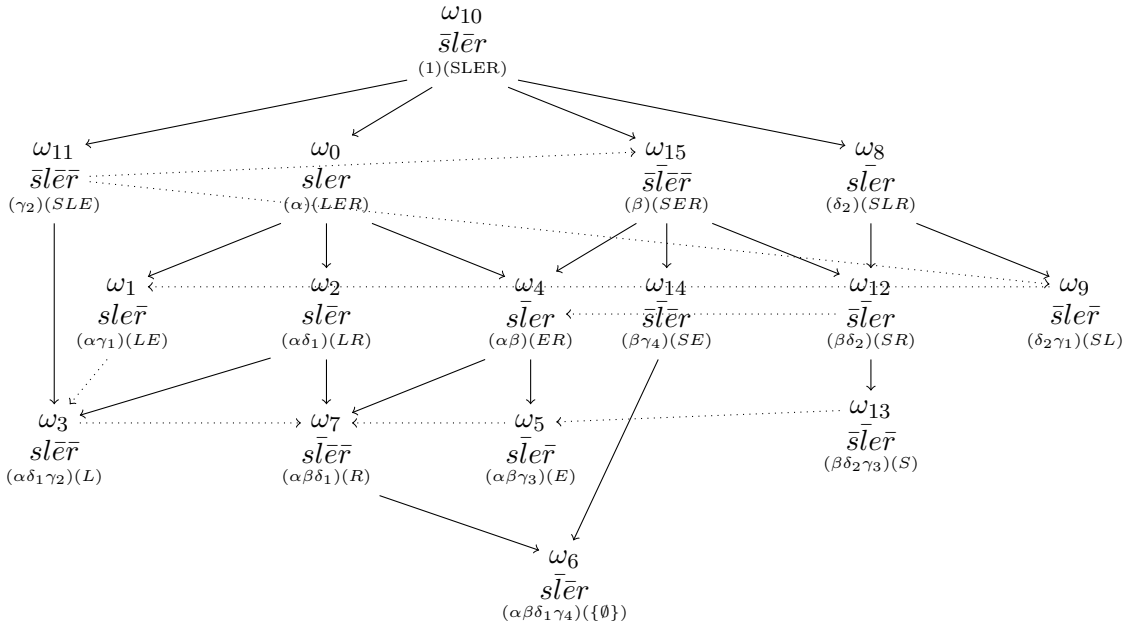


Figure 3.5: Induced graph of π-pref net in Figure 3.1 based on the product chain rule

To conclude, if a CP-net ranks two configurations as one being preferred to the other, e.g., $\omega \succ_{\mathcal{C}} \omega'$, then the corresponding product-based π-pref either entails an incomparability case $\omega \bowtie_{Prod} \omega'$ or supports the same dominance relation, but never generates a preference reversal e.g. $\omega' \succ_{Prod} \omega$ [Ben Amor et al., 2018a].

Next section details necessary constraints to recover all preference relations induced

by a CP-net.

3.3.2 Representing ceteris paribus dominance relations by π-pref nets

CP-nets underly an implicit *priority* in favor of parent nodes over their children. Thus, *satisfaction* degrees associated with parent nodes in the corresponding π-pref net must be lower than those associated with their children (for instance, in Figure 3.1, $\gamma_2 \succ \alpha$ must be verified). This property can be insured by the following constraint [Ben Amor et al., 2018a]

$$\forall i = 1, N \quad \max_{u_i \in \underline{U_{X_i}}} \alpha_{X_i|u_i} < \prod_{C \in Ch(X_i)} \min_{u_C \in \underline{U_C}} \rho_{C|u_C} \tag{3.3}$$

where $\alpha_{X_i|u_i}$ is the satisfaction degree of the bad value of X_i in the context of u_i, $\rho_{C|u_C}$ is similarly defined and $Ch(X)$ denote the set of nodes that depend on X (graphically, $Ch(X)$ includes all children nodes of X).

Consider a CP-net $\mathcal{C}$ and the corresponding π-pref net. Let (ω, ω') be a pair of configurations that differ by the value of $X \in \mathcal{X}$ and such that ω is preferred to ω' based on the ceteris paribus assumption, $\omega \succ_{\mathcal{C}} \omega'$. Let x^+ and x^- denote respectively the good and the bad values of the variable X in the context of its parent value $u_X \in \underline{U_X}$. Let $\mathcal{Y} = \mathcal{X} \setminus \{X\} \cup Ch(X) \cup U_X$ be the set of remaining variables. Configurations ω and ω' can be written under the form of the conjunction of literals such that $\omega \models x^+ \wedge u_X \wedge c^* \wedge y^*$ and $\omega' \models x^- \wedge u_X \wedge c^* \wedge y^*$ such that $y^* \in \underline{\mathcal{Y}}$, $C^* \in \underline{Ch(X)}$ provided that $Ch(X) \neq \{\emptyset\}$, $c^* \in \underline{C} = \{c, \bar{c}\}$.

In order for the π-pref net to recover the relation $\omega \succ_{\mathcal{C}} \omega'$, the constraint $\pi(\omega) > \pi(\omega')$ needs to be satisfied [Ben Amor et al., 2018a]. Note that the possibility degree of $\mathcal{Y}$ is the same for both configurations and do not depend on the value of X or its child nodes. Therefore, the constraint $\pi(\omega) > \pi(\omega')$ entails that $\pi(x^+|u_X).\prod_{C \in Ch(X)} \pi(C \mid u_C).\prod_{Y \in \mathcal{Y}} \pi(y^* \mid u_Y) > \pi(x^-|u_X).\prod_{C \in Ch(X)} \pi(C \mid u_C).\prod_{Y \in \mathcal{Y}} \pi(y^* \mid u_Y)$. Let $\pi(x^-|u_X) = \rho$, this leads to the constraint

$$\prod_{C \in Ch(X)} \pi(C \mid u_C) > \rho \cdot \prod_{C \in Ch(X)} \pi(C \mid u_C) \quad \text{with} \quad C = c \text{ or } \quad C = \bar{c} \tag{3.4}$$

For each node of the π-pref net, the above constraints must be considered in order to recover all dominance relations inferred by the corresponding CP-net $\mathcal{C}$. In

[Ben Amor et al., 2017a], Proposition (3) has established that the constraints (3.3) are enough to ensure the satisfaction of conditions (3.4), which ensures that $\omega \succ_{\mathcal{C}} \omega'$.

The following example illustrates the fact that the addition of constraints (3.3) enables us to enforce in the π-pref net all the dominance relations of a CP-net.

Example 3.8 *Consider the induced graph in Figure 3.5. In order to cover all comparisons entailed by the ceteris paribus assumption (Figure 1.3), a number of inequalities based on the constraint 3.4 must be added. The first step consist in defining nodes that are not leafs which are* $\{S, L, E\}$. *Detailed computation allowing to infer ceteris paribus constraints are detailed in Table 3.3. The set of relevant and most hard constraints are written in bold, namely* $\delta_2 > \alpha, \gamma_1 > \beta, \gamma_2 > \beta, \gamma_1 > \beta, \gamma_1 > \delta_1, \gamma_3 > \delta_1, \gamma_4 > \delta_2, \gamma_2 > \delta_2\gamma_1$ *and* $\gamma_2 > \delta_2\gamma_3$. *Therefore, we can check that all comparisons represented by dashed arrows in Figure 3.5 are recovered.*

Leaving complete freedom for a user to add constraints that express priorities between preference degrees of a π-pref net confirms the flexibility of this model compared to CP-nets. Although CP-nets share the same graphical structure and level of simplicity as π-pref nets, they do not have the same expressive power. In [Ben Amor et al., 2018a], it has been proved that CP-nets and π-pref net induce consistent orderings over solutions. Without additional constraints, π-pref nets can not capture CP-net's dominance relations induced from the priority granted to father nodes. However, by adding some constraints to a π-pref net, we are able to recover all order relations induced by a CP-net. Thus, by adding some constraints to a π-pref net, we are able to restore the ceteris paribus priorities to capture CP-nets preference relations.

3.3.3 π-pref nets vs cp-theories

As mentioned in Chapter 1, a CP statement over a subset $\mathcal{U} \subseteq \mathcal{X}$ is formalized by $\phi = u|\mathcal{V} : x \succ x'$, where $u \in \mathcal{U}$, $x, x' \in \underline{X}$ and $\mathcal{U}, \mathcal{V}$ are disjoint subsets s.t. $\mathcal{U} \subseteq \mathcal{X} \setminus \{X\}$, $\mathcal{V} \subseteq \mathcal{X} \setminus \mathcal{U}$. They express the preference specification $u : x \succ x'$ independently from values of the subset of variables $\mathcal{V}$. At first sight they seem to encode the exact definition of the Markovian independence property used by π-pref nets.

Indeed, the CP statement $\phi = u|\mathcal{V} : x \succ x'$ entails that any configuration that models $u \wedge x$ is preferred to any configuration that models $u \wedge x'$ whatever is the value of $\mathcal{V}$. In π-pref net, the statement ϕ is implicitly satisfied by the constraint $\Pi(u \wedge x) > \Pi(u \wedge x')$ inferred from the constraint $\pi(x|u) > \pi(x'|u)$ such that $u : x \succ x'$.

Node	Ch(node)	Constraints	
S	$\{E\}$	$\pi(\bar{s}).\pi(e\|\bar{s}) > \pi(s).\pi(e\|s)$	$\boldsymbol{\delta_2 > \alpha}$
		$\pi(\bar{s}).\pi(\bar{e}\|\bar{s}) > \pi(s).\pi(\bar{e}\|s)$	$1 > \alpha\delta_1$
L	$\{R\}$	$\pi(l).\pi(r\|le) > \pi(\bar{l}).\pi(r\|\bar{l}e)$	$1 > \beta$
		$\pi(l).\pi(r\|le) > \pi(\bar{l}).\pi(r\|\bar{l}\bar{e})$	$1 > \beta\gamma_4$
		$\pi(l).\pi(r\|l\bar{e}) > \pi(\bar{l}).\pi(r\|\bar{l}e)$	$1 > \beta$
		$\pi(l).\pi(r\|l\bar{e}) > \pi(\bar{l}).\pi(r\|\bar{l}\bar{e})$	$1 > \beta\gamma_4$
		$\pi(l).\pi(\bar{r}\|le) > \pi(\bar{l}).\pi(\bar{r}\|\bar{l}e)$	$\gamma_1 > \beta\gamma_3$
		$\pi(l).\pi(\bar{r}\|le) > \pi(\bar{l}).\pi(\bar{r}\|\bar{l}\bar{e})$	$\boldsymbol{\gamma_1 > \beta}$
		$\pi(l).\pi(\bar{r}\|l\bar{e}) > \pi(\bar{l}).\pi(\bar{r}\|\bar{l}e)$	$\gamma_2 > \beta\gamma_3$
		$\pi(l).\pi(\bar{r}\|l\bar{e}) > \pi(\bar{l}).\pi(\bar{r}\|\bar{l}\bar{e})$	$\boldsymbol{\gamma_2 > \beta}$
E	$\{R\}$	$\pi(e\|s).\pi(r\|le) > \pi(\bar{e}\|s).\pi(r\|l\bar{e})$	$1 > \delta_1$
		$\pi(e\|s).\pi(r\|le) > \pi(\bar{e}\|s).\pi(r\|\bar{l}\bar{e})$	$1 > \delta_1\gamma_4$
		$\pi(e\|s).\pi(r\|\bar{l}e) > \pi(\bar{e}\|s).\pi(r\|l\bar{e})$	$1 > \delta_1$
		$\pi(e\|s).\pi(r\|\bar{l}e) > \pi(\bar{e}\|s).\pi(r\|\bar{l}\bar{e})$	$1 > \delta_1\gamma_4$
		$\pi(e\|s).\pi(r\|le) > \pi(\bar{e}\|s).\pi(r\|l\bar{e})$	$\gamma_1 > \delta_1\gamma_2$
		$\pi(e\|s).\pi(r\|le) > \pi(\bar{e}\|s).\pi(r\|\bar{l}\bar{e})$	$\boldsymbol{\gamma_1 > \delta_1}$
		$\pi(e\|s).\pi(r\|\bar{l}e) > \pi(\bar{e}\|s).\pi(r\|l\bar{e})$	$\gamma_3 > \delta_1\gamma_2$
		$\pi(e\|s).\pi(r\|\bar{l}e) > \pi(\bar{e}\|s).\pi(r\|\bar{l}\bar{e})$	$\boldsymbol{\gamma_3 > \delta_1}$
		$\pi(\bar{e}\|\bar{s}).\pi(r\|l\bar{e}) > \pi(e\|\bar{s}).\pi(r\|le)$	$1 > \delta_2$
		$\pi(\bar{e}\|\bar{s}).\pi(r\|l\bar{e}) > \pi(e\|\bar{s}).\pi(r\|\bar{l}e)$	$1 > \delta_2$
		$\pi(\bar{e}\|\bar{s}).\pi(r\|\bar{l}\bar{e}) > \pi(e\|\bar{s}).\pi(r\|le)$	$\boldsymbol{\gamma_4 > \delta_2}$
		$\pi(\bar{e}\|\bar{s}).\pi(r\|\bar{l}\bar{e}) > \pi(e\|\bar{s}).\pi(r\|\bar{l}e)$	$\gamma_4 > \delta_2$
		$\pi(\bar{e}\|\bar{s}).\pi(\bar{r}\|l\bar{e}) > \pi(e\|\bar{s}).\pi(\bar{r}\|le)$	$\boldsymbol{\gamma_2 > \delta_2\gamma_1}$
		$\pi(\bar{e}\|\bar{s}).\pi(\bar{r}\|l\bar{e}) > \pi(e\|\bar{s}).\pi(\bar{r}\|\bar{l}e)$	$\boldsymbol{\gamma_2 > \delta_2\gamma_3}$
		$\pi(\bar{e}\|\bar{s}).\pi(\bar{r}\|\bar{l}\bar{e}) > \pi(e\|\bar{s}).\pi(\bar{r}\|le)$	$1 > \delta_2\gamma_1$
		$\pi(\bar{e}\|\bar{s}).\pi(\bar{r}\|\bar{l}\bar{e}) > \pi(e\|\bar{s}).\pi(\bar{r}\|\bar{l}e)$	$1 > \delta_2\gamma_3$

Table 3.3: Ceteris paribus constraints for π-pref net in Figure 3.1

It has been explained in section 3.3.2 that by adding some constraints to a π-pref net, we are able to recover all order relations induced by a CP-net. Besides, cp-theories can represent CP-nets [Wilson, 2011]. However, the representation power of cp-theories and π-pref nets have not been compared until now; see however [Ben Amor et al., 2018a] for a beginning of discussion.

In the remaining of this chapter, we explore other ways of encoding conditional preferences, slightly beyond the standard possibilistic setting.

3.4 Other ways of encoding conditional preferences

We have seen that when dealing with Boolean variables of choice, the claim "Given c, I prefer x to $\bar{x}$" can be translated into an elementary possibility distribution such that

$\pi(x|c) > \pi(\bar{x}|c)$ assuming the use of $[0, 1]$ as a scale for satisfaction degrees (still we were excluding the idea that the symbolic weight might be equal to 0). Moreover, the conditional preference tables can be turned into a possibilistic logic base expressing priorities between goals in terms of a necessity measure N associated with π as recalled in Chapter 2.

Let us observe that in the π-pref nets approach we are dealing with satisfaction constraints and violation constraints in a not similar way. Indeed, in the conditional preference tables, all the possibility degrees corresponding to the satisfaction of the preference are set to 1, while the violation situations receive a different symbolic weight for each preference. This might be justified by the fact that the important thing in the evaluation of the configuration is to take into account the preference violation. However, in the following, we shall consider an approach where we rather assign the same degree 0 in all violation cases and different symbolic degrees for the satisfaction of each preference.

A particular phenomenon takes place in the π-pref net approach. Indeed, due to the privileged role of 1, we always get a unique best configuration with satisfaction degree equal to 1, which does not violate any preference. Even if there is also a unique configuration that violates all preferences, the partial order induced in general by the π-pref net leads to incomparability of this worst configuration with some other configurations that violate less variables. Indeed, in Table 3.1, we can observe that ω_6, which violates all preferences, is incomparable for instance with ω_{13} and ω_{12}. Moreover, there are four configurations: ω_3, ω_6, ω_9 and ω_{13} that are not comparable to each other and that are not better than any other configurations.

This situation raises the question of the possibility of for instance reversing the phenomenon by having a unique worst solution maybe at the price of several non-comparable non-dominated configurations.

3.4.1 Use of the guaranteed possibility distributions

In order to privilege 0 in the encoding of preferences, we may imagine to turn the possibility distribution π into another distribution $1 - \pi$. As we shall see, this will lead us to use the set function Δ recalled in Chapter 2. Switching from the possibility to the guaranteed possibility is done by simply reversing the possibility distribution: if π is normalized to 1 then $1 - \pi$ is normalized to 0. In other words, on the one hand we compute the complement to 1 of quality vectors (e.g., $(1, \beta, \gamma)$ is changed into $(0, 1-\beta, 1-\gamma)$), and on the other hand, we may wonder what becomes the chain rule

in the transformation from π to $\delta = 1 - \pi$.

In order to define a joint distribution associated with a non-normalized π-pref net, we transform the product chain rule in Equation 3.1 to obtain the following chain rule:

$$1 - \delta(X_1, ..., X_N) = \prod_{i=1}^{N} (1 - \delta(X_i | U_{X_i})) \tag{3.5}$$

Note that the distribution δ is anti-normalized, i.e., $\exists \omega$ such that $\delta(\omega) = 0$, but may be not normalized.

In the following, we give an example of a variant of π-pref net handled in terms of Equation 3.5. We study two cases: (i) when each preference attached to a context has a particular symbolic weight for expressing the satisfaction degree when the preference is violated and, (ii) the case where we do not distinguish between the different instantiations of parent variables and where there is only a unique symbolic weight per node.

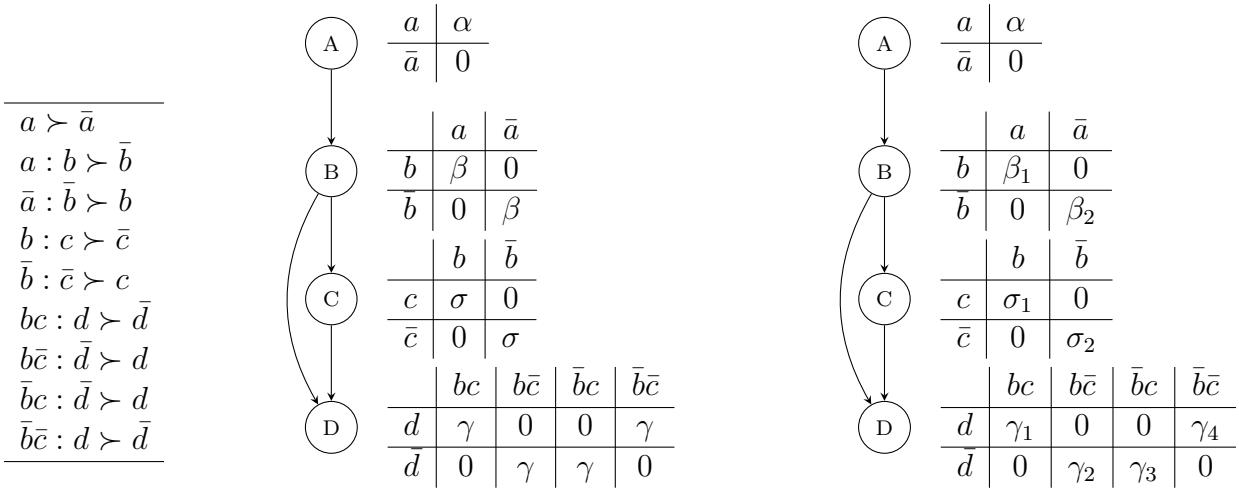


Figure 3.6: Examples of an anti-normalized π-pref nets

Example 3.9 *Figure 3.6 depicts an anti-normalized π-pref net given (a) equal and (b) different symbolic weights per variable and parents value. We consider the same set of preference specifications as in Figure 4.1. The specification $\bar{a} : \bar{b} \succ b$ is encoded by $\delta(\bar{b}|\bar{a}) > \delta(b|\bar{a})$ where $\delta(\bar{b}|\bar{a}) = \beta_1 \in (0, 1)$ and $\delta(b|\bar{a}) = 0$. Table 3.4 gives results of the product chain rule on guaranteed possibility degrees given by Equation 3.5. The worst configuration is indeed associated with the lowest joint preference degree $\pi(\omega_{10}) = 1 - \delta(\omega_{10}) = 0$ where $\delta(\omega_{10}) = (1 - \delta(A)).(1 - \delta(B|A).(1 - \delta(C|B).(1 - \delta(D|BC) =$*

$(1-0).(1-0).(1-0).(1-0) = 1$. *The induced distribution* π *on* Ω *is represented by a Pareto graph in Figure 3.7.*

Ω		$\vec{\omega}$	$1-\delta(\omega)$	$\vec{\omega}$	$1-\delta(\omega)$
$\omega_0 =$	$abcd$	$(\alpha, \beta_1, \sigma_1, \gamma_1)$	$(1-\alpha).(1-\beta_1).$ $(1-\sigma_1).(1-\gamma_1)$	$(\alpha, \beta, \sigma, \gamma)$	$(1-\alpha).(1-\beta).$ $(1-\alpha).(1-\beta)$
$\omega_1 =$	$abc\bar{d}$	$(\alpha, \beta_1, \sigma_1, 0)$	$(1-\alpha).(1-\beta_1).(1-\sigma_1)$	$(\alpha, \beta, \sigma, 0)$	$(1-\alpha).(1-\beta).(1-\sigma)$
$\omega_2 =$	$ab\bar{c}d$	$(\alpha, \beta_1, 0, 0)$	$(1-\alpha).(1-\beta_1)$	$(\alpha, \beta, 0, 0)$	$(1-\alpha).(1-\beta)$
$\omega_3 =$	$ab\bar{c}\bar{d}$	$(\alpha, \beta_1, 0, \gamma_2)$	$(1-\alpha).(1-\beta_1).(1-\gamma_2)$	$(\alpha, \beta, 0, \gamma)$	$(1-\alpha).(1-\beta).(1-\gamma)$
$\omega_4 =$	$a\bar{b}cd$	$(\alpha, 0, 0, 0)$	$(1-\alpha)$	$(\alpha, 0, 0, 0)$	$(1-\alpha)$
$\omega_5 =$	$a\bar{b}c\bar{d}$	$(\alpha, 0, 0, \gamma_3)$	$(1-\alpha).(1-\gamma_3)$	$(\alpha, 0, 0, \gamma)$	$(1-\alpha).(1-\gamma)$
$\omega_6 =$	$a\bar{b}\bar{c}d$	$(\alpha, 0, \sigma_2, \gamma_4)$	$(1-\alpha).(1-\sigma_2).(1-\gamma_4)$	$(\alpha, 0, \sigma, \gamma)$	$(1-\alpha).(1-\sigma).(1-\gamma)$
$\omega_7 =$	$a\bar{b}\bar{c}\bar{d}$	$(\alpha, 0, \sigma_2, 0)$	$(1-\alpha).(1-\sigma_2)$	$(\alpha, 0, \sigma, 0)$	$(1-\alpha).(1-\sigma)$
$\omega_8 =$	$\bar{a}bcd$	$(0, 0, \sigma_1, \gamma_1)$	$(1-\sigma_1).(1-\gamma_1)$	$(0, 0, \sigma, \gamma)$	$(1-\sigma).(1-\gamma)$
$\omega_9 =$	$\bar{a}bc\bar{d}$	$(0, 0, \sigma_1, 0)$	$(1-\sigma_1)$	$(0, 0, \sigma, 0)$	$(1-\sigma)$
$\omega_{10} =$	$\bar{a}b\bar{c}d$	$(0, 0, 0, 0)$	1	$(0, 0, 0, 0)$	1
$\omega_{11} =$	$\bar{a}b\bar{c}\bar{d}$	$(0, 0, 0, \gamma_2)$	$(1-\gamma_2)$	$(0, 0, 0, \gamma)$	$(1-\gamma)$
$\omega_{12} =$	$\bar{a}\bar{b}cd$	$(0, \beta_2, 0, 0)$	$(1-\beta_2)$	$(0, \beta, 0, 0)$	$(1-\beta)$
$\omega_{13} =$	$\bar{a}\bar{b}c\bar{d}$	$(0, \beta_2, 0, \gamma_3)$	$(1-\beta_2).(1-\gamma_3)$	$(0, \beta, 0, \gamma)$	$(1-\beta).(1-\gamma)$
$\omega_{14} =$	$\bar{a}\bar{b}\bar{c}d$	$(0, \beta_2, \sigma_2, \gamma_4)$	$(1-\beta_2).(1-\sigma_1).(1-\gamma_4)$	$(0, \beta, \sigma, \gamma)$	$(1-\beta).(1-\sigma).(1-\gamma)$
$\omega_{15} =$	$\bar{a}\bar{b}\bar{c}\bar{d}$	$(0, \beta_2, \sigma_2, 0)$	$(1-\beta_2).(1-\sigma_1)$	$(0, \beta, \sigma, 0)$	$(1-\beta).(1-\sigma)$

Table 3.4: Vectors and weights associated to configurations of the π-pref net in Figure 3.6. In the last two columns the symbolic weights associated to the violation of a preference are the same in all contexts

As we can observe on Figure 3.5, in case we use different symbolic weights according to the context, there is a unique worst configuration here $\omega_6 = s\bar{l}\bar{e}r$, while there are several non-comparable non-dominated configurations, namely $\omega_3 = sl\bar{e}\bar{r}$, $\omega_5 = s\bar{l}e\bar{r}$, $\omega_9 = \bar{s}le\bar{r}$ and $\omega_{13} = s\bar{l}\bar{e}r$. As for regular π-pref nets, when there is a unique symbolic weight per variable (see Table 3.2 and Figure 3.3), we have more comparisons between configurations using the obtained partial order, in agreement with the number of violated preferences (see discussion in Chapter 1 Section 1.2.2).

Obviously, we might have used a qualitative variant of Equation 3.5, where we use min in place of product. However, for the same reason (drowning effect) as in a regular π-pref net, product is to be preferred to the min.

3.4.2 Use of bi-valued possibility distributions

We have seen the interest of privileging either 1 or 0 in the comparison process. We may wonder what would be obtained if we just consider 1 and 0 as satisfaction degrees, i.e.,

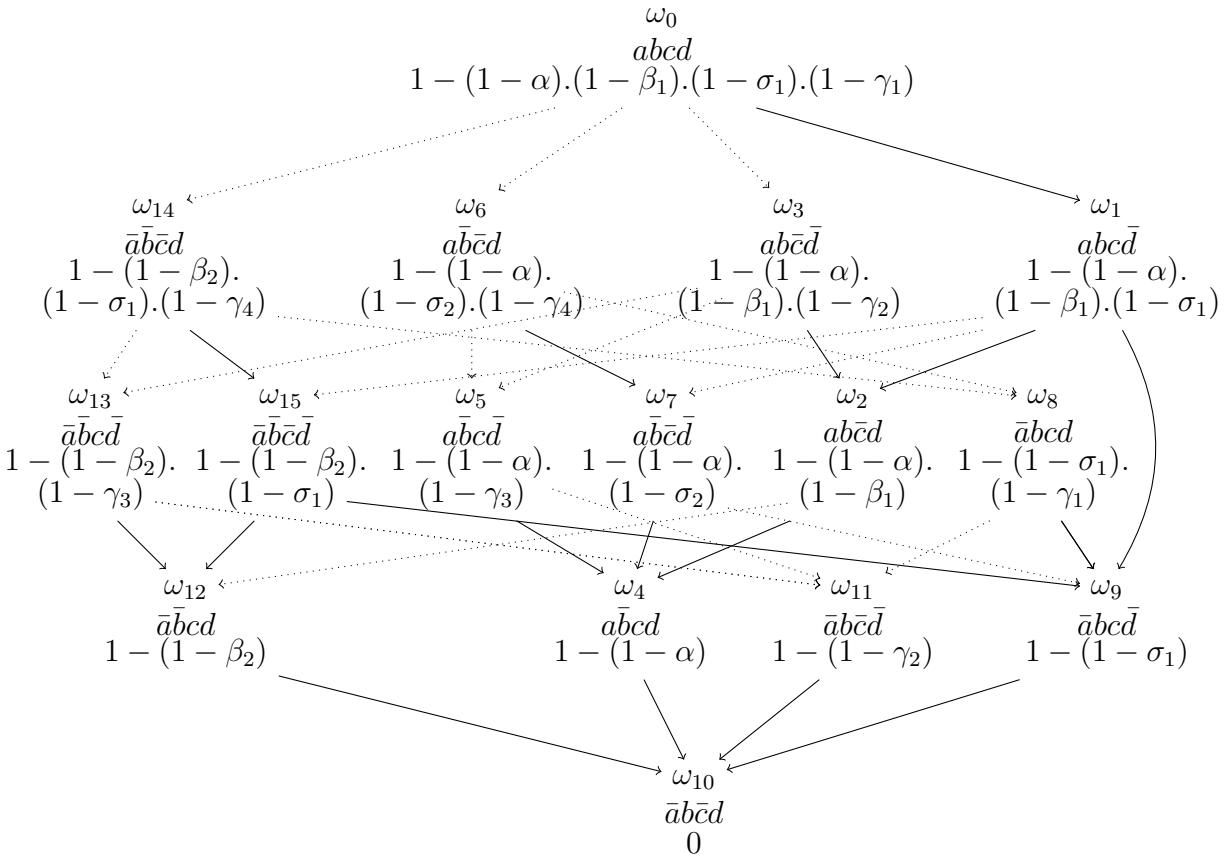


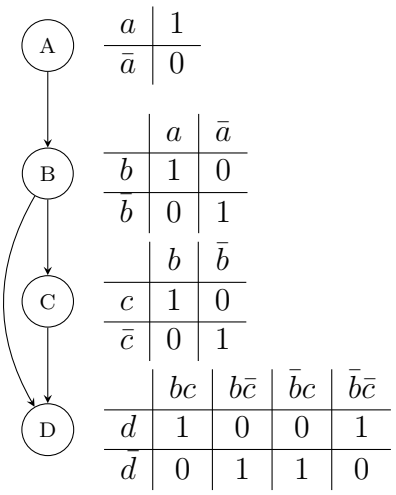
Figure 3.7: Pareto graph of π-pref nets in Figure 3.6

we assign to the preferred (or good) value of X the highest preference degree possible and to its bad value the least possible preference degree. This means that given the preference statement $u : x \succ \bar{x}$ then $\pi(x|u) = 1$ and $\pi(\bar{x}|u) = 0$.

In this case the components of quality vectors are made of 1 or 0. Then the Pareto ordering yield a partial order such that there is a unique best configuration (where all satisfaction degrees are equal to 1) and a unique worst configuration (where all satisfaction degrees are equal to 0). This situation is illustrated in Example 3.10.

Example 3.10 *Figure 3.8 shows a bi-normalized π-pref net constructed from the same set of preference specifications as in Figure 4.1. The specification $\bar{a} : \bar{b} \succ b$ is encoded by $\pi(\bar{b}|\bar{a}) > \pi(b|\bar{a})$ where $\pi(\bar{b}|\bar{a}) = 1$ and $\pi(b|\bar{a}) = 0$. The second last column of Table 3.5 gives results of the product chain rule based on Equation 3.1. There exists a unique best configuration ω_0 with $\pi(\omega_0) = 1$, while all the remaining configurations are rejected with a preference degree equal to 0. All configurations, except of ω_0, are equivalent and are ranked worst than ω_0. The same specifications can also be encoded by satisfaction degrees where π is replaced by δ. Results of the product chain rule on guaranteed possibility degrees based on Equation 3.5 are given in the last column of*

Table 3.5. All configurations, except of ω_{10}, are considered equally preferred and are ranked better than the worst configuration ω_{10}. The Pareto graph of Figure the π-pref net in Figure 3.8 is given in Figure 3.9.



	a
a	1
$\bar{a}$	0

	a	$\bar{a}$
b	1	0
$\bar{b}$	0	1

	b	$\bar{b}$
c	1	0
$\bar{c}$	0	1

	bc	$b\bar{c}$	$\bar{b}c$	$\bar{b}\bar{c}$
d	1	0	0	1
$\bar{d}$	0	1	1	0

Figure 3.8: Bi-normalized (bi-valued) π-pref net

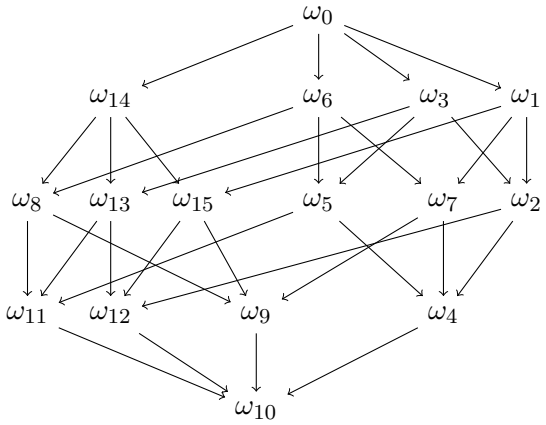


Figure 3.9: Pareto graph of π-pref net in Figure 3.8

Due to the drowning effects of degrees equal to 0 in the product chain rule of Equation 3.1, and to degrees equal to 1 in the chain rule of Equation 3.5, the ordering induced by this extension is poorly discriminant (composed of only two sets). However, since the Pareto strategy compares configurations based on comparisons between values of single variables, it captures a lot more comparisons than the product chain rule.

Let us finally remark that if we want to keep the benefits of both privileging 1 and 0 in the scale and having intermediary degrees in between, we have to handle quality vectors where we can apply, in comparisons the inequalities $1 > \alpha$ and $\beta > 0$ for any symbolic weight α and β. This situation is discussed in the next section.

Ω		$\vec{\omega}$	$\pi(\omega)$	$\Delta(\omega)$
ω_0	$abcd$	$(1,1,1,1)$	1	1
ω_1	$abc\bar{d}$	$(1,1,1,0)$	0	1
ω_2	$ab\bar{c}d$	$(1,1,0,0)$	0	1
ω_3	$ab\bar{c}\bar{d}$	$(1,1,0,1)$	0	1
ω_4	$a\bar{b}cd$	$(1,0,0,0)$	0	1
ω_5	$a\bar{b}c\bar{d}$	$(1,0,0,1)$	0	1
ω_6	$a\bar{b}\bar{c}d$	$(1,0,1,1)$	0	1
ω_7	$a\bar{b}\bar{c}\bar{d}$	$(1,0,1,0)$	0	1
ω_8	$\bar{a}bcd$	$(0,0,1,1)$	0	1
ω_9	$\bar{a}bc\bar{d}$	$(0,0,1,0)$	0	1
ω_{10}	$\bar{a}b\bar{c}d$	$(0,0,0,0)$	0	0
ω_{11}	$\bar{a}b\bar{c}\bar{d}$	$(0,0,0,1)$	0	1
ω_{12}	$\bar{a}\bar{b}cd$	$(0,1,0,0)$	0	1
ω_{13}	$\bar{a}\bar{b}c\bar{d}$	$(0,1,0,1)$	0	1
ω_{14}	$\bar{a}\bar{b}\bar{c}d$	$(0,1,1,1)$	0	1
ω_{15}	$\bar{a}\bar{b}\bar{c}\bar{d}$	$(0,1,1,0)$	0	1

Table 3.5: Vectors and weights associated with configurations of π-pref net in Figure 3.8

3.4.3 Use of non-normalized distributions

In the variant presented in this subsection, we abandon the options of putting to 1 all the weights corresponding to satisfactory situations, or to 0 all the weights corresponding to violated preferences.

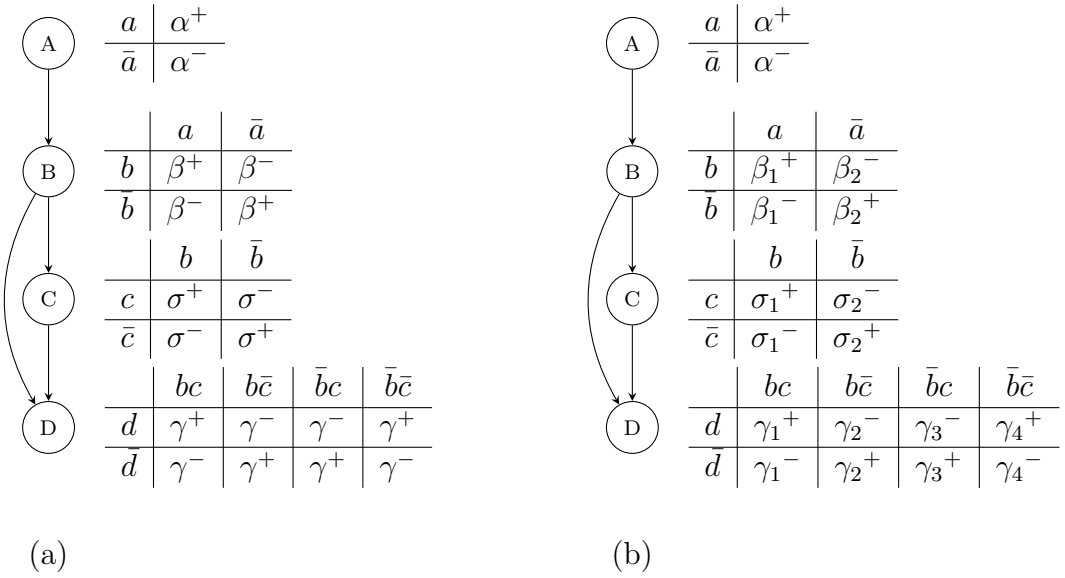
In possibility theory, the totally ordered unipolar numerical scale $[0,1]$ is used to encode preferences and information in general. A normalized π has at least an element whose satisfaction degree is equal to 1 and not necessarily another element with satisfaction degree equal to 0. A choice associated with the lowest degree expresses that the choice is completely rejected, while the highest value is neutral and does not ensure the complete satisfaction of the user about it. Similarly, an anti-normalized guaranteed possibility distribution has at least an element whose satisfaction degree is equal to 0, and has not necessarily an element with satisfaction degree equal to 1.

It is tempting to wonder what happens if we use a scale without 0 and without 1. Then, we go out of the scope of possibility theory, we have no longer any chain rule but we can still work with quality vectors. This is what we discuss now.

A simple way of encoding a conditional preference $u : x \succ \bar{x}$ is to assign to the good value of X a degree ψ strictly higher than the one assigned to its bad value, namely $\psi(x|u) = \alpha^+$ and $\psi(\bar{x}|u) = \alpha^-$ such that $\psi(x|u) > \psi(\bar{x}|u)$. We grant the agent the total freedom to instantiate these degrees with values in the open interval $(0, 1)$. Then for comparing quality vectors, we can only rely inequality of the type $\alpha^+ > \alpha^-$, $\beta^+ > \beta^-$, $\cdots$, for each symbolic weight introduced in the encoding of the various preferences. In such a case, we have no longer, in general, a unique best configuration or a unique worst configuration.

As in the two previous sections, we study both the cases where there is a unique symbolic weight (with $+$ or $-$) per node or if this weight depends on the context of the preference. The following example shows the kind of comparisons that are preserved with this approach: in case of different symbolic degrees per variable and parent value, we have obviously less comparisons than with the corresponding standard π-pref net or than with the reversed π-pref net in the sense of Section 3.4.1. Each configuration can only be compared to one other configuration. However, when symbolic degrees are unique per variable, we obtain a partial order that captures much more comparisons. These comparisons are due to the deterioration of one or many variables' values.

induced from π-pref nets in the sense of Sections 3.2 and 3.4.1.



(a) (b)

Figure 3.10: Conditional preference network with non-normalized distribution on preferences: (a) for equal symbolic degrees per variable (b) for different symbolic degrees per variable and context

Example 3.11 *Figure 3.10 shows a non-normalized preference network constructed from the same set of preference specifications as in Figure 3.6. When there is a unique*

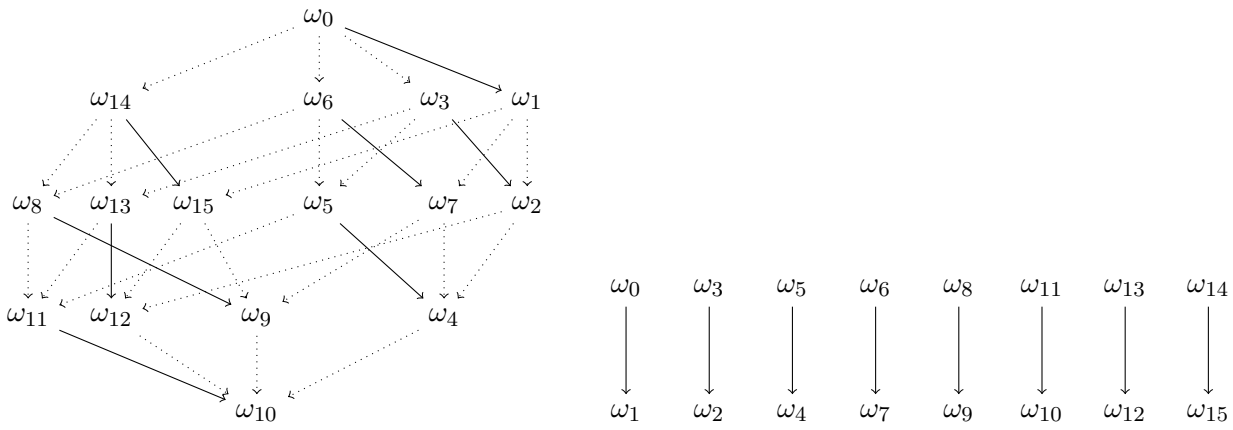
symbolic degree per variable, for instance, the specification $\bar{a} : \bar{b} \succ b$ *is encoded by* $\psi(\bar{b}|\bar{a}) > \psi(b|\bar{a})$ *where* $\psi(\bar{b}|\bar{a}) = \beta^+$ *and* $\psi(b|\bar{a}) = \beta^-$*. The same specification is encoded by* $\psi(\bar{b}|\bar{a}) = {\beta_2}^+$ *and* $\psi(b|\bar{a}) = {\beta_2}^-$ *when symbolic degrees are different per variable and context. The specification* $a : b \succ \bar{b}$ *is encoded by* $\psi(b|a) > \psi(\bar{b}|a)$ *where* $\psi(b|a) = {\beta_1}^+$ *and* $\psi(\bar{b}|a) = {\beta_1}^-$*. In the first case, i.e., equal symbols per variable, vectors associated with each configuration are given in the second last column of Table 3.6. In the other case, i.e., different symbols per variable and context, vectors associated with each configuration are given in the last column of the same table. When degrees are different, the induced Pareto graph in Figure 3.11 (b) (also depicted in Figure 3.11(a) by solid arrows) show that few pairs of configurations can be compared. All comparisons induced in this case lay between configurations that differ by a single flip value.*

$\mathbf{\Omega}$		$\vec{\boldsymbol{\omega}}$	$\vec{\boldsymbol{\omega}}$
ω_0	$abcd$	$(\alpha^+, \beta^+, \sigma^+, \gamma^+)$	$(\alpha^+, {\beta_1}^+, {\sigma_1}^+, {\gamma_1}^+)$
ω_1	$abc\bar{d}$	$(\alpha^+, \beta^+, \sigma^+, \gamma^-)$	$(\alpha^+, {\beta_1}^+, {\sigma_1}^+, {\gamma_1}^-)$
ω_2	$ab\bar{c}d$	$(\alpha^+, \beta^+, \sigma^-, \gamma^-)$	$(\alpha^+, {\beta_1}^+, {\sigma_1}^-, {\gamma_2}^-)$
ω_3	$ab\bar{c}\bar{d}$	$(\alpha^+, \beta^+, \sigma^-, \gamma^+)$	$(\alpha^+, {\beta_1}^+, {\sigma_1}^-, {\gamma_2}^+)$
ω_4	$a\bar{b}cd$	$(\alpha^+, \beta^-, \sigma^-, \gamma^-)$	$(\alpha^+, {\beta_1}^-, {\sigma_2}^-, {\gamma_3}^-)$
ω_5	$a\bar{b}c\bar{d}$	$(\alpha^+, \beta^-, \sigma^-, \gamma^+)$	$(\alpha^+, {\beta_1}^-, {\sigma_2}^-, {\gamma_3}^+)$
ω_6	$a\bar{b}\bar{c}d$	$(\alpha^+, \beta^-, \sigma^+, \gamma^+)$	$(\alpha^+, {\beta_1}^-, {\sigma_2}^+, {\gamma_4}^+)$
ω_7	$a\bar{b}\bar{c}\bar{d}$	$(\alpha^+, \beta^-, \sigma^+, \gamma^-)$	$(\alpha^+, {\beta_1}^-, {\sigma_2}^+, {\gamma_4}^-)$
ω_8	$\bar{a}bcd$	$(\alpha^-, \beta^-, \sigma^+, \gamma^+)$	$(\alpha^-, {\beta_2}^-, {\sigma_1}^+, {\gamma_1}^+)$
ω_9	$\bar{a}bc\bar{d}$	$(\alpha^-, \beta^-, \sigma^+, \gamma^-)$	$(\alpha^-, {\beta_2}^-, {\sigma_1}^+, {\gamma_1}^-)$
ω_{10}	$\bar{a}b\bar{c}d$	$(\alpha^-, \beta^-, \sigma^-, \gamma^-)$	$(\alpha^-, {\beta_2}^-, {\sigma_1}^-, {\gamma_2}^-)$
ω_{11}	$\bar{a}b\bar{c}\bar{d}$	$(\alpha^-, \beta^-, \sigma^-, \gamma^+)$	$(\alpha^-, {\beta_2}^-, {\sigma_1}^-, {\gamma_2}^+)$
ω_{12}	$\bar{a}\bar{b}cd$	$(\alpha^-, \beta^+, \sigma^-, \gamma^-)$	$(\alpha^-, {\beta_2}^+, {\sigma_2}^-, {\gamma_3}^-)$
ω_{13}	$\bar{a}\bar{b}c\bar{d}$	$(\alpha^-, \beta^+, \sigma^-, \gamma^+)$	$(\alpha^-, {\beta_2}^+, {\sigma_2}^-, {\gamma_3}^+)$
ω_{14}	$\bar{a}\bar{b}\bar{c}d$	$(\alpha^-, \beta^+, \sigma^+, \gamma^+)$	$(\alpha^-, {\beta_2}^+, {\sigma_2}^+, {\gamma_4}^+)$
ω_{15}	$\bar{a}\bar{b}\bar{c}\bar{d}$	$(\alpha^-, \beta^+, \sigma^+, \gamma^-)$	$(\alpha^-, {\beta_2}^+, {\sigma_2}^+, {\gamma_4}^-)$

Table 3.6: Vectors and weights associated with configurations of networks in Figure 3.10

In case of preference encoding using different symbolic degrees per variable and context values, the only comparisons that are possible are those between configurations that differ by a single flip of value. We have been thus interested in answering the following question: What are the comparisons that the ceteris paribus property induced but not the Pareto strategy considering an encoding of preferences with non-

normalized distributions ? Since CP-nets induce an ordering that implicitly stipulates that violating a parent node is more penalizing than violating child nodes, the answer could be that the uncovered comparisons are those induced from this property. This would confirm that such a structure is free from biased information. For instance, let us consider a conditional preference structure simply described over two decision variables. Next example treats this case.



(a) (b)

Figure 3.11: Pareto graphs of networks in Figure 3.10: (a) solid arrows represent comparisons given different symbolic degrees per variable and context, and dotted arrows represent comparisons given unique symbols per variable; (b) only comparisons induced given different symbolic degrees per variable and context

Example 3.12 *Let us consider the conditional preference network in Figure 3.12. The ceteris paribus property induces a total order described by the worsening flip sequence in Figure 3.13. When preferences are encoded with non-normalized distributions with different degrees per variable and context values, local tables associated with nodes hold the following specifications* $\psi(a) = \alpha^+$, $\psi(\bar{a}) = \alpha^-$, $\psi(b|a) = \beta_1^+$, $\psi(\bar{b}|a) = \beta_1^-$, $\psi(b|\bar{a}) = \beta_2^-$, $\psi(\bar{b}|\bar{a}) = \beta_2^+$. *The only possible comparisons based on the Pareto property are* $ab \succ_{Pareto} a\bar{b}$ *and* $\bar{a}\bar{b} \succ_{Pareto} \bar{a}b$ *(see solid arrows in Figure 3.14). When preferences are encoded with non-normalized distributions with equal degrees per variable and parent values, local tables associated with nodes hold the following specifications* $\psi(a) = \alpha^+$, $\psi(\bar{a}) = \alpha^-$, $\psi(b|a) = \psi(\bar{b}|\bar{a}) = \beta^+$, $\psi(\bar{b}|a) = \psi(b|\bar{a}) = \beta^-$. *The additional Pareto comparisons are* $ab \succ_{Pareto} \bar{a}\bar{b}$ *and* $a\bar{b} \succ_{Pareto} \bar{a}b$ *(see dotted arrows in Figure 3.14). The only comparison that is not recovered with respect to the ceteris paribus induced ordering is between configurations* $a\bar{b}$ *and* $\bar{a}\bar{b}$. *The configurations are described by respective quality vectors* $+-$ *and* $-+$ *and the ceteris paribus assumption ranks the*

one associated with $+-$ as preferred to the one associated with $-+$. It is a comparison in favor of the priority given to the parent A over the son B.

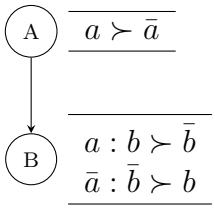


Figure 3.12: Example of a conditional preference network

Figure 3.13: Worsening flip sequence of the conditional preference network in Figure 3.12 using the ceteris paribus property

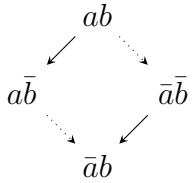


Figure 3.14: Pareto graph of conditional preference network in Figure 3.12 for preferences encoding by non-normalized distributions. Solid arrow represent comparisons for different symbols per variable and context values and dotted arrows reflect additional comparisons for equal symbolic degrees per variable and context values

3.5 Conclusion

π-pref nets offer a flexible setting for representing preferences. The use of symbolic weights and possibly additional constraints between this symbolic weights enables us to have an approach which is not more committed than permitted by the available information.

When compared with CP-nets, it has been proved that π-pref nets can recover the dominance relations induced by CP-nets orderings by means of additional constraints between product of symbolic weights (which is a way to mimic the fact that father nodes are more important than children nodes). As pointed out in [Wilson et al., 2019], π-pref nets provide a valuable way of computing in a polynomial time a good upper approximation of any CP-net. We have not mentioned OCF-networks [Eichhorn et al., 2016] which are indeed very close to π-pref nets. OCF-networks are based on Spohn's ordinal conditional functions (also called ranking function [Spohn, 2012], [Spohn, 1988], [Eichhorn et al., 2016]) can be transposed into the setting of possibility theory [Dubois and Prade, 2016].

In the last part of this chapter, we have explored variants of the π-pref net approach, showing the versatile nature of this type of approach based on possibility or possibility-like distributions and comparisons on quality vectors. We have shown that the different variants may in general yield different partial orders, which are induced by different ways of cautiously representing the agent preferences. However, it is in practice quite desirable to obtain complete pre-orders. This is the concern of the next chapter.

Besides, the ultimate purpose to compactly represent preferences by graphical structures is to find an ordering on complete solutions given conditional preferences. π-pref nets are positioned somewhat in between qualitative and quantitative models [Ben Amor et al., 2016a] due to the symbolic treatment of weights. Numerical π-pref nets may be of interest for learning purposes.

Chapter 4

Conditional Preferences as Defaults: Possibilistic Approaches

4.1 Introduction

Chapter 1 has reviewed the best known forms of modeling preferences, while Chapters 2 and 3 have represented how to handle conditional preferences in the setting of possibility theory. More precisely, specifications can be represented by graphical networks or by possibilistic logic bases. When modeling preferences with π-pref nets, the Pareto dominance (equivalent to the product chain rule order) is a natural basis for comparing solutions: one is then preferred to another as soon as the latter violates the same preferences as the first one and some others.

Until now, we have been using a chain rule or an order principle on quality vectors allow us to rank order the set of possible choices. In this chapter, we propose another handling of conditional preference statements inspired by the treatment of default rules. In the setting of possibility theory setting, default rules are represented by constraints to which one may apply two information principles: the minimum, and maximum specificity [Dubois and Prade, 2015].

The application of the minimum specificity principle, applied to preferences, amounts to saying that a configuration is considered satisfactory unless preference statements say otherwise. In contrast, the maximum specificity principle amounts to saying that a configuration is considered unsatisfactory unless preference statements say otherwise. Applying such principles on preference specifications under default rules yields a *complete pre-orders* on the set of possible alternatives.

The aim of this chapter is to investigate and discuss the diverse orderings that can be inferred from preference specifications handled as default rules.

This chapter is divided in six main sections. Section 4.2 gives the necessary background on the possibilistic approach to reason on default rules encoding knowledge. The next section details the diverse ways of reasoning on preferences in such a setting, then presents the default-like approach to preference handling, and compares orderings induced by the default rules and the Pareto order. In Section 4.4, we show that the complete pre-orders obtained have always three layers for particular structures, which may be insufficiently discriminant. In section 4.5, we present a modified algorithm that can remedy to excessive effects of minimum or maximum specificity principles regarding configurations that are not constrained. In Section 4.6, we show that there are preference statements that cannot be described neither by π-pref-nets nor by CP-nets, but still can be handled by the "default-like" method. An experimental study, reported in Section 4.7, is proposed to confirm propositions and conjectures mentioned in this chapter.

Mind that in this chapter and in the next one, we no longer use the running example of chapters 1 to 3, but a slightly more sophisticated example necessary to exhibit some behaviours of interest.

The work reported in this chapter mainly rely on [Ben Amor et al., 2019] and to some extent on [Ben Amor et al., 2021a].

4.2 Background on possibilistic approach to default rules

A *default rule* [Pearl, 1990] of the form $p \rightsquigarrow q$ where p and q are Boolean propositions and $\rightsquigarrow$ is a non-classical arrow, modeling the rule "if p then generally q". Such rule divides the set of possible interpretations Ω into three parts: those that satisfy p but falsify q ($\omega \models p \wedge \bar{q}$), the models that verify both p and q ($\omega \models p \wedge q$ or $\omega \models \bar{p} \vee q$), and the interpretations for which the rule cannot be fired, i.e., those that falsify p ($\omega \models \bar{p}$). Such a rule is actually very similar to so-called *conditional objects* denoted by $q|p$ [Dubois and Prade, 1991]. More interestingly here, it also looks like the conditional preference $p : q \succ \bar{q}$.

In this section, we recall the approach on default rules offered by the possibility framework, in order to infer a ranking on the possible states of the world.

Given a set of default rules $\mathcal{R} = \{r_i : p_i \rightsquigarrow q_i, i = 1, \ldots, k\}$ that represent a knowledge about the world, Benferhat et. al. [Benferhat et al., 1992] gave a method that rank orders the set of alternatives Ω such that the more the configuration satisfies the rules in R, the higher is its possibility degree.

Given a default rules knowledge base $\mathcal{R}$, $\mathcal{C}_\Pi$ denotes the set of constraints modeling this rule. Namely a rule $p \rightsquigarrow q$ is understood in the possibilistic setting as a constraint $\Pi(p \wedge q) > \Pi(p \wedge \bar{q})$ which means that if p is true then q true is more possible than q false. Let $\mathcal{C}_\Pi$ be the set of constraints induced by $\mathcal{R}$:

$$\mathcal{C}_\Pi = \{c_i : \Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q_i}), r_i : p_i \rightsquigarrow q_i \in \mathcal{R}\} \tag{4.1}$$

When representing knowledge in the possibilistic setting, a minimum specificity principle is applied which amounts to assessing the greatest possible possibility degree agreeing with the constraints to each interpretation. This is a least commitment principle since it does not restrict the interpretations that are possible to some extent abusively (see Section 4.3.1 for more details).

Maximizing possibility degrees of interpretations according to the minimum specificity principle is achieved via the Algorithm 4.1 which outputs a well-ordered partition composed of sets E_j of configurations [Benferhat et al., 1992].

Algorithm 4.1: Algorithm of partitioning of Ω using the minimum specificity principle

Input: The set of solutions Ω
The set of possibilistic constraints $\mathcal{C}_\Pi$
Output: A well-ordered partition E

```
1 j = 0
2 while Ω ≠ ∅ do
3 |   E_j = {ω_i, i = 0, ··· , m} s.t. ω_i does not belong to the set of configurations
  |     on the right-hand side of any constraint (ω_i is never dominated)
4 |   Ω = Ω \ E_j
5 |   Remove from C_Π all satisfied constraints (their left-hand side are consistent
  |     with solutions of E_j)
6 |   j ← j + 1
7 end
8 return E
```

Given a set of constraints, the first step consists of finding interpretations that are never dominated. They can be derived from computing the negation of the disjunction of formulas that appear on the right side of constraints of $\mathcal{C}_\Pi$. In accordance with the minimum specificity principle, the resulting interpretations are then associated to the

highest possibility degree (e.g. $\pi(\omega_i) = 1$) and are assigned to the first partition E_0. Constraints that are satisfied are then deleted from $\mathcal{C}_\Pi$. The same process is repeated until no constraints are left. In a final step, the remaining interpretations of Ω are assigned to a final last level.

This procedure is illustrated by the following example.

Example 4.1 *Let $\mathcal{R}$ be composed of the following default rules expressing some knowledge about cancer diagnosis $\{c \rightsquigarrow s,\ y \rightsquigarrow \bar{c},\ \bar{y} \wedge \bar{s} \rightsquigarrow \bar{c},\ s \rightsquigarrow y\}$. Rules are respectively interpreted by the assertions:*

$c \rightsquigarrow s$:	*People diagnosed by cancer c are generally smokers s.*
$y \rightsquigarrow \bar{c}$:	*Young people y are generally not sick $\bar{c}$.*
$\bar{y} \wedge \bar{s} \rightsquigarrow \bar{c}$:	*Old persons $\bar{y}$ that do not smoke $\bar{s}$ are generally not sick $\bar{c}$.*
$s \rightsquigarrow y$:	*Smokers s are generally young y.*

We have three bi-valued description variables C for Cancer, S for Smoking and Y for young. The universe of discourse is thus composed of 2^3 solutions, where $\Omega = \{\omega_0 = ysc, \omega_1 = ys\bar{c}, \omega_2 = y\bar{s}c, \omega_3 = y\bar{s}\bar{c}, \omega_4 = \bar{y}sc, \omega_5 = \bar{y}s\bar{c}, \omega_6 = \bar{y}\bar{s}c, \omega_7 = \bar{y}\bar{s}\bar{c}\}$. Given $\mathcal{R}$ we can infer the set of constraints $\mathcal{C}_\Pi$ (resp. $\mathcal{C}_\Delta$ by replacing the measure Π by Δ in all constraints) as follows

$$\Pi(c \wedge s) > \Pi(c \wedge \bar{s}) = \max(\pi(\omega_0), \pi(\omega_4)) > \max(\pi(\omega_2), \pi(\omega_6))$$
$$\Pi(y \wedge \bar{c}) > \Pi(y \wedge c) = \max(\pi(\omega_1), \pi(\omega_3)) > \max(\pi(\omega_0), \pi(\omega_2))$$
$$\Pi(\bar{y} \wedge \bar{s} \wedge \bar{c}) > \Pi(\bar{y} \wedge \bar{s} \wedge c) = \pi(\omega_7) > \pi(\omega_6)$$
$$\Pi(s \wedge y) > \Pi(s \wedge \bar{y}) = \max(\pi(\omega_0), \pi(\omega_1)) > \max(\pi(\omega_4), \pi(\omega_5))$$

The optimistic approach consist thus on applying algorithm 4.1. The first iteration gives the first partition $E_0 = \{\omega_1, \omega_3, \omega_7\}$ that corresponds to solutions that model $\overline{c \wedge \bar{s}} \vee \overline{y \wedge c} \vee \overline{\bar{y} \wedge \bar{s} \wedge c} \vee \overline{s \wedge \bar{y}}$. All constraints except the first one are now satisfied, since there exists at least one solution in E_0 that figures on the left-hand side of them. The second partition set is composed as follows $E_1 = \{\omega_0, \omega_4, \omega_5\}$. It groups solutions that are yet not ranked and that model $\overline{c \wedge \bar{s}}$. After satisfying and deleting the last constraint, the third iteration outputs the set $E_2 = \{\omega_2, \omega_6\}$. Furthermore, the partition obtained by applying the most specific distribution i. e. adapting a positive reasoning is $E = \{\{\omega_1, \omega_3, \omega_7\}, \{\omega_0, \omega_4, \omega_5\}, \{\omega_2, \omega_6\}\}$.

We now consider the application of the above approach to conditional preferences.

4.3 Default rules for preferences

At a semantic level, a conditional preference statement $p : q \succ \bar{q}$ means that configurations that satisfy $p \wedge q$ are preferred to configurations that satisfy $p \wedge \bar{q}$. When dealing with single conditional preference. Let ω, ω' such that $\omega \models p \wedge q$ and $\omega' \models p \wedge \bar{q}$, we obtain the following preference relation

$$\pi(\omega) > \pi(\omega') \text{ iff } p : q \succ \bar{q} \tag{4.2}$$

In the knowledge representation perspective of the previous section, the default rules were represented in terms of constrains expressed by possibility measures: we wrote that the maximum of the possibility degrees of models of $p \wedge q$ are greater than the maximum of the possibility degrees of models of $p \wedge \bar{q}$, meaning that there is at least one model of $p \wedge q$ that is more possible than any model of $p \wedge \bar{q}$. This representation has proved to be appropriate for default rules [Benferhat et al., 1997]. However, this treatment in terms of possibility measures may be found to be too liberal for the intended meaning of the preference $p : q \succ \bar{q}$. Indeed, we may think of aggregating the possibility degrees on each side of the inequality in different ways. Namely,

$$\oplus \{\pi(\omega) : \omega \models p_i \wedge q_i\} > \otimes\{\pi(\omega') : \omega' \models p_i \wedge \bar{q_i}\} \text{ iff } p_i : q_i \succ \bar{q_i} \tag{4.3}$$

where $\oplus$ and $\otimes$ correspond to either the minimum or maximum operator. This can be related to possibility set functions Π and Δ and considering conditional preferences, we thus can distinguish between four types of semantics that have been already considered in the literature [Dubois et al., 2004] [Kaci and van der Torre, 2008] [Kaci, 2012] [Ben Amor et al., 2019]. We now recall these four types of reading conditional preferences.

Definition 4.1 (Possible meanings of conditional preferences) *Let $p : q \succ \bar{q}$ be a conditional preference statement. Considering a constraint of the form $\oplus\{\pi(\omega) : \omega \models p \wedge q\} > \otimes\{\pi(\omega') : \omega' \models p \wedge \bar{q}\}$, we have the four following options when the aggregation operators are* min *or* max*:*

- ***Optimistic** : $p : q \succ \bar{q}$ encodes the claim "In the context of p, I prefer the best case in which q is true to the best case in which $\bar{q}$ is true". This is an optimistic reading focusing on best cases, formally expressed by $\Pi(p \wedge q) > \Pi(p \wedge \bar{q})$, i.e., $\oplus$ and $\otimes$ are both replaced by the* max *operator.*

- ***Pessimistic***: $p : q \succ \bar{q}$ *means that "In the context of* p*, I prefer the worst case in which* q *is true to the worst case in which* $\bar{q}$ *is true". This is a pessimistic reading focusing on worst cases, formally encoded by* $\Delta(p \wedge q) > \Delta(p \wedge \bar{q})$*, i.e.,* $\oplus$ *and* $\otimes$ *are both replaced by the* min *operator.*

- ***Opportunistic***: $p : q \succ \bar{q}$ *can correspond to the specification "In the context of* p*, I prefer the best configurations in which* q *is true to the worst configurations in which* $\bar{q}$ *is true". This statement is formally expressed by the equation* $\Pi(p \wedge q) > \Delta(p \wedge \bar{q})$*, i.e.,* $\oplus = \max$ *and* $\otimes = \min$*.*

- ***Cautious***: $p : q \succ \bar{q}$ *can express a strong preference encoding the specification and is interpreted by the claim "In the context of* p*, I prefer the worst configurations in which* q *is true to the best configurations in which* $\bar{q}$ *is true". This statement is encoded by* $\Delta(p \wedge q) > \Pi(p \wedge \bar{q})$*, i.e.,* $\oplus = \min$ *and* $\otimes = \max$*.*

Thus, the preferences of user may be understood in distinct ways, according to whether he stresses on what is rejected or on what is satisfactory. In this chapter, we only study the two first readings. Note that the optimistic and pessimistic readings cannot be compared in terms of strength, while the cautious one and the opportunistic one are respectively stronger and weaker than the optimistic and pessimistic readings. The remaining readings are either too strong or too weak for being really of interest in the modeling of conditional preferences. Indeed, the cautious reading by forcing all the configurations where $p \wedge q$ is true to be more satisfactory than any configuration where $p \wedge \bar{q}$ is true does not leave any room for exceptions; this lack of flexibility sounds undesirable for local specifications of preferences. Note that we have not such a defect with the optimistic or pessimistic readings. The opportunistic reading, on the contrary guarantees only that one configuration where $p \wedge q$ is true is more satisfactory than one configuration where $p \wedge \bar{q}$ is true; this is really weak as an understanding of the conditional preference. The reader is referred to [Kaci, 2012] for further discussions.

When dealing with the optimistic reading where we look for a possibility distribution π satisfying the constraints $\Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q_i})$, we naturally apply the minimum specificity principle since we look for the less restrictive distribution. In the pessimistic reading we have constraints of the form $\Delta(p_i \wedge q_i) > \Delta(p_i \wedge \bar{q_i})$, since the Δ functions are decreasing, they are associated with an opposite principle, namely a *maximum specificity principle*[1], which assesses the smallest possible degrees in agreement with the constraints.

[1]This principle can be applied either when the Δ function is used for representing knowledge (in this case, it usually refers to reported facts and then only these facts are regarded as possible) [Benferhat et al., 2008], or when for representing preferences [Dubois et al., 2005] [Kaci, 2012].

The minimal specificity principle corresponds thus to an *optimistic* way of reasoning by saying that if a configuration ω has not been explicitly rejected, it is considered as completely satisfactory. Thus, if some configurations are not constrained they will be associated to the highest possible preference degree, namely $\pi(\omega_i) = 1$. Thus, the minimum specificity principle means for preferences that a configuration is considered satisfactory unless preference statements say otherwise. In contrast, reasoning on preferences under a *pessimistic* view comes down to consider that a configuration is rejected ($\pi(\omega) = 0$) unless the user specifies its preference level. This minimization principle corresponds to the maximum specificity principle. This is to say that, in the possibility theory, when dealing with conditional propositions, the most (resp. least) specific distribution(s) exist and correspond to an optimistic (resp. pessimistic) approach. However, specific distributions for the cautious and opportunistic semantics do not exist.

4.3.1 Optimistic approach on default preferences

A default rule $p \rightsquigarrow q$ is represented in possibility theory by a constraint stating that having $p \wedge q$ is more satisfactory than $p \wedge \bar{q}$ that can be encoded by $\Pi(p \wedge q) > \Pi(p \wedge \bar{q})$. Thus, the default rule $p \rightsquigarrow q$ expresses the conditional preference $p : q \succ \bar{q}$ and is understood as "In the context defined by p, the best situation that models q is preferred to the best situation that models $\bar{q}$", or in other words "the best case in which $p \wedge q$ is true is preferred to the best case in which $p \wedge \bar{q}$ is true". A possibility distribution on configurations of Ω can be deduced from such constraints, based on some informational principle. Indeed, the set of conditional preference statements expressed by an agent $\{p_i : q_i \succ \bar{q}_i \mid i = 1, \ldots, k\}$ is viewed a set of default rules $\mathcal{R} = \{r_i : p_i \rightsquigarrow q_i \mid i = 1, \ldots, k\}$. Using an optimistic view, i.e, the minimum specificity principle, we construct the set of constraints $\mathcal{C}_\Pi$ derived from $\mathcal{R}$ by applying (4.1). This set is implicitly associated with a set of possibility distributions compatible with constraints in $\mathcal{C}_\Pi$.

Any possibility distribution that is in agreement with constraints in $\mathcal{C}_\Pi$ represents the conditional preference statements expressed by the agent, in agreement with the way it was done by the conditional preference tables in π-pref nets. This directly follows from the definition of the conditional possibility: $\Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q}_i)$ iff $\pi(q_i \mid p_i) = 1 > \pi(\bar{q}_i \mid p_i)$ (where $\Pi(p_i \wedge q_i) = \Pi(q_i \mid p_i) \cdot \Pi(p_i)$) .

The following proposition establishes the agreement of the Pareto ordering with the constraints in $\mathcal{C}_\Pi$ expressing the preference statements [Ben Amor et al., 2021a].

Given a π-pref net, the associated possibility distribution obtained by the chain rule and the corresponding Pareto ordering between configurations agree with the constraints in $\mathcal{C}_\Pi$ expressing the preference statements of the π-pref net.

Proposition 4.1 *Given a π-pref net, let π be the possibility distribution obtained by the product chain rule and $\succ_\pi$ its associated preference relation between configurations of Ω. Let $\succ_{Pareto}$ be the preference relation on configurations of Ω obtained using the Pareto strategy. Let $\mathcal{C}_\Pi$ be the set of default constraints expressing the preference statements $p_i : q_i \succ \bar{q}_i$ of the π-pref net. Then, $\forall \omega, \omega' \in \Omega$ such that $\omega \succ_{Pareto} \omega'$ i.e. $\omega \succ_\pi \omega'$, there exists no constraint $c_i \in \mathcal{C}_\Pi$ such that $c_i : \Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q}_i) : \omega \models p_i \wedge \bar{q}_i, \omega' \models p_i \wedge q_i$.*

Proof 4.1 *Let ω and ω' be two configurations associated with their satisfaction vectors $\vec{\omega}$ and $\vec{\omega'}$. $\pi(\omega) > \pi(\omega')$ is known to be equivalent to $\vec{\omega} \succ_{Pareto} \vec{\omega'}$ [Ben Amor et al., 2016b] assuming one weight per preference statement. It means that for each vector component (corresponding to a variable) either the two components are equal with the same satisfaction degree, or the component of $\vec{\omega}$ is 1 and the same component for $\vec{\omega'}$ is equal to some symbolic weight, say ρ, assuming $\vec{\omega} \succ_{Pareto} \vec{\omega'}$. Each inequality $1 > \rho$, corresponds to an inequality of the form $\pi(x_i \mid u_i) = 1 > \pi(\bar{x}_i \mid u'_i)$. If $u_i = u'_i$ this refers to a preference statement in a conditional table $\pi(x_i \mid u_i) = 1 > \pi(\bar{x}_i \mid u_i)$ equivalent as already said to the constraint $\Pi(u_i \wedge x_i) > \Pi(u_i \wedge \bar{x}_i)$. If $u_i \neq u'_i$, this violates no preference statement. It means that the comparison of 1 with ρ refers to different contexts and no preference statement is involved. Besides, the vector components where $\vec{\omega}$ and $\vec{\omega'}$ are equal cannot correspond to a violation of a preference statement.*

Thus the π-pref nets approach and the constraints of the "default rule" approach are in full agreement. To induce a ranking over the set of possible alternatives Ω, we apply exactly the same steps of Algorithm 4.1 which comes down to apply the minimal specificity principle to the constraints $\mathcal{C}_\Pi$. In this case, violations of the Pareto ordering may appear as shown by the following Example 4.2. For easiness of representation, we give another writing of constraints in $\mathcal{C}_\Pi$ such that each of its formula is translated into a collection of subsets $(LC(c_i), RC(c_i))$ referring to the existing configurations of the left respectively right-hand side of constraint c_i $(\Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q}_i)$. Thus $\mathcal{C}_\Pi = \{(LC(c_i), RC(c_i)\}$ where $LC(c_i) = \{\omega : \omega \models p_i \wedge q_i, r_i : p_i \rightsquigarrow q_i\}$ and $RC(c_i) = \{\omega : \omega \models p_i \wedge \bar{q}_i, r_i : p_i \rightsquigarrow q_i\}$.

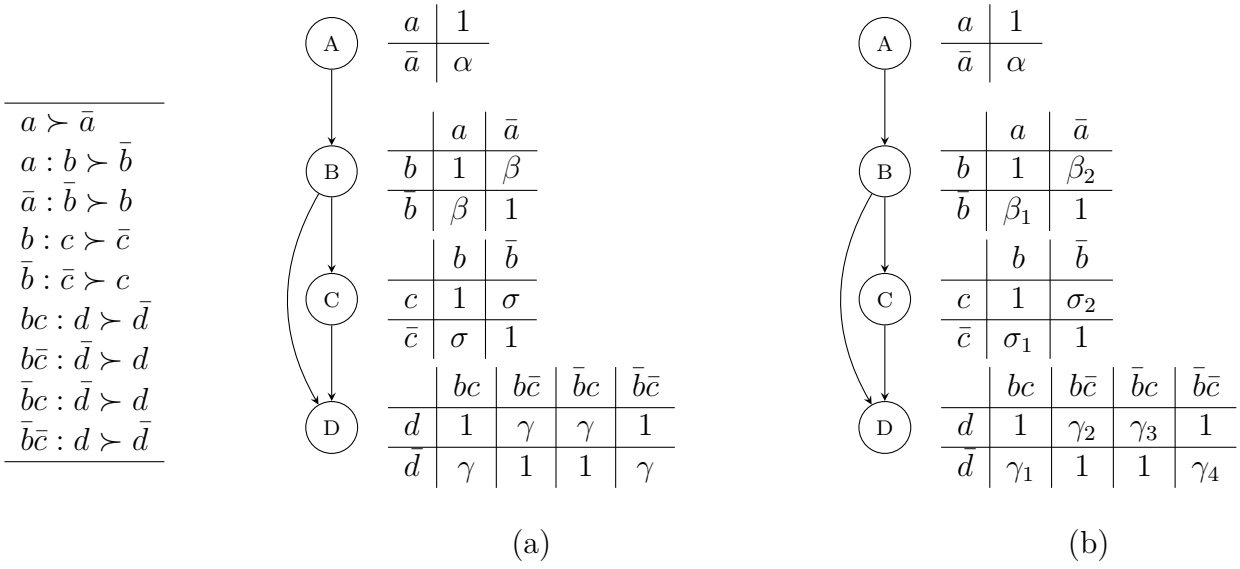


Figure 4.1: Examples of normalized π-pref nets

Example 4.2 *Figure 4.1 depicts a π-pref net with normalized local distributions: (a) for equal symbolic degrees and (b) for different degrees in context of parents. The set of the combinatorial space of configurations is* $\Omega = \{\omega_0 = abcd, \omega_1 = abc\bar{d}, \omega_2 = ab\bar{c}d, \omega_3 = ab\bar{c}\bar{d}, \omega_4 = a\bar{b}cd, \omega_5 = a\bar{b}c\bar{d}, \omega_6 = a\bar{b}\bar{c}d, \omega_7 = a\bar{b}\bar{c}\bar{d}, \omega_8 = \bar{a}bcd, \omega_9 = \bar{a}bc\bar{d}, \omega_{10} = \bar{a}b\bar{c}d, \omega_{11} = \bar{a}b\bar{c}\bar{d}, \omega_{12} = \bar{a}\bar{b}cd, \omega_{13} = \bar{a}\bar{b}c\bar{d}, \omega_{14} = \bar{a}\bar{b}\bar{c}d, \omega_{15} = \bar{a}\bar{b}\bar{c}\bar{d}\}$. *The set of preference statements represented by the left-most table of Figure 4.1 corresponds to* $\mathcal{R}$=$\{r_0 : a,\ r_1 : a \rightsquigarrow b,\ r_2 : \bar{a} \rightsquigarrow \bar{b},\ r_3 : b \rightsquigarrow c,\ r_4 : \bar{b} \rightsquigarrow \bar{c},\ r_5 : bc \rightsquigarrow d,\ r_6 : b\bar{c} \rightsquigarrow \bar{d},\ r_7 : \bar{b}c \rightsquigarrow \bar{d},\ r_8 : \bar{b}\bar{c} \rightsquigarrow d\}$. *Based on Equation 4.1, we infer the following respective constraints:*

$$
\begin{aligned}
c_0 &: \max(\pi(\omega_0), \ldots, \pi(\omega_7)) > \max(\pi(\omega_8), \ldots, \pi(\omega_{15})) \\
c_1 &: \max(\pi(\omega_0), \ldots, \pi(\omega_3)) > \max(\pi(\omega_4), \ldots, \pi(\omega_7)) \\
c_2 &: \max(\pi(\omega_{12}), \ldots, \pi(\omega_{15})) > \max(\pi(\omega_8), \ldots, \pi(\omega_{11})) \\
c_3 &: \max(\pi(\omega_0), \pi(\omega_1), \pi(\omega_8), \pi(\omega_9)) > \max(\pi(\omega_2), \pi(\omega_3), \pi(\omega_{10}), \pi(\omega_{11})) \\
c_4 &: \max(\pi(\omega_6), \pi(\omega_7), \pi(\omega_{14}), \pi(\omega_{15})) > \max(\pi(\omega_4), \pi(\omega_5), \pi(\omega_{12}), \pi(\omega_{13})) \\
c_5 &: \max(\pi(\omega_0), \pi(\omega_8)) > \max(\pi(\omega_1), \pi(\omega_9)) \\
c_6 &: \max(\pi(\omega_3), \pi(\omega_{11})) > \max(\pi(\omega_2), \pi(\omega_{10})) \\
c_7 &: \max(\pi(\omega_5), \pi(\omega_{13})) > \max(\pi(\omega_4), \pi(\omega_{12})) \\
c_8 &: \max(\pi(\omega_6), \pi(\omega_{14})) > \max(\pi(\omega_7), \pi(\omega_{15}))
\end{aligned}
$$

Constraints in $\mathcal{C}_\Pi$ of Example 4.2 are rewritten as tuples as follows:

$$
\begin{aligned}
c_0 :&\quad LC(c_0) = \{\omega_0, \ldots, \omega_7\}, RC(c_0) = \{\omega_8, \ldots, \omega_{15}\}\\
c_1 :&\quad LC(c_1) = \{\omega_0, \ldots, \omega_3\}, RC(c_1) = \{\omega_4, \ldots, \omega_7\}\\
c_2 :&\quad LC(c_2) = \{\omega_{12}, \ldots, \omega_{15}\}, RC(c_2) = \{\omega_8, \ldots, \omega_{11}\}\\
c_3 :&\quad LC(c_3) = \{\omega_0, \omega_1, \omega_8, \omega_9\}, RC(c_3) = \{\omega_2, \omega_3, \omega_{10}, \omega_{11}\}\\
c_4 :&\quad LC(c_4) = \{\omega_6, \omega_7, \omega_{14}, \omega_{15}\}, RC(c_4) = \{\omega_4, \omega_5, \omega_{12}, \omega_{13}\}\\
c_5 :&\quad LC(c_5) = \{\omega_0, \omega_8\}, RC(c_5) = \{\omega_1, \omega_9\}\\
c_6 :&\quad LC(c_6) = \{\omega_3, \omega_{11}\}, RC(c_6) = \{\omega_2, \omega_{10}\}\\
c_7 :&\quad LC(c_7) = \{\omega_5, \omega_{13}\}, RC(c_7) = \{\omega_4, \omega_{12}\}\\
c_8 :&\quad LC(c_8) = \{\omega_6, \omega_{14}\}, RC(c_8) = \{\omega_7, \omega_{15}\}
\end{aligned}
$$

When applying the Algorithm 4.1, we can see that there exists a single configuration that do not belong to any right-hand part of constraints in $\mathcal{C}_\Pi$ *which composes to the first partition set* $E_1 = \{\omega_0\}$*. Constraints* c_0, c_1, c_3, c_5 *are thus satisfied since* ω_0 *appears on the right-hand part of them namely,* $RC(c_0), RC(c_1), RC(c_3)$ *and* $RC(c_5)$*. They are thus removed from* $\mathcal{C}_\Pi$ *which is now composed of constraints:*

$$
\begin{aligned}
c_2 :&\quad LC(c_2) = \{\omega_{12}, \ldots, \omega_{15}\}, RC(c_2) = \{\omega_8, \ldots, \omega_{11}\}\\
c_4 :&\quad LC(c_4) = \{\omega_6, \omega_7, \omega_{14}, \omega_{15}\}, RC(c_4) = \{\omega_4, \omega_5, \omega_{12}, \omega_{13}\}\\
c_6 :&\quad LC(c_6) = \{\omega_3, \omega_{11}\}, RC(c_6) = \{\omega_2, \omega_{10}\}\\
c_7 :&\quad LC(c_7) = \{\omega_5, \omega_{13}\}, RC(c_7) = \{\omega_4, \omega_{12}\}\\
c_8 :&\quad LC(c_8) = \{\omega_6, \omega_{14}\}, RC(c_8) = \{\omega_7, \omega_{15}\}
\end{aligned}
$$

The left solutions to be ranked are $\Omega \setminus \{\omega_0\}$*. In a second iteration, we get* $E_1 = \{\omega_1, \omega_3, \omega_6, \omega_{14}\}$*. The verified constraints to be deleted from* $\mathcal{C}_\Pi$ *are* c_2, c_4, c_6 *and* c_8 *leaving a unique formula to satisfy which is*

$$
c_7 :\quad LC(c_7) = \{\omega_5, \omega_{13}\}, RC(c_7) = \{\omega_4, \omega_{12}\}
$$

The set Ω *is updated and now composed of configurations* $\Omega \cap \{\omega_1, \omega_3, \omega_6, \omega_{14}\}$*. Therefore, the set* E_2 *is composed of all left elements in* Ω *except of* ω_4 *and* ω_{12} *that compose* $RC(c_7)$*. We get* $E_2 = \{\omega_2, \omega_5, \omega_7, \omega_8, \omega_9, \omega_{10}, \omega_{11}, \omega_{13}, \omega_{15}\}$*. The last remaining constraint* c_7 *is now satisfied and the last partition set is composed of* $E_3 = \{\omega_4, \omega_{12}\}$*.*

Minimum specificity ordering	**Levels**	**Cardinality ordering**
$\{\omega_0\}$	1	$\{\omega_0\}$
$\{\omega_1, \omega_3, \omega_6, \omega_{14}\}$	2	$\{\omega_1, \omega_3, \omega_6, \omega_{14}\}$
$\{\omega_2, \omega_5, \omega_7, \omega_8, \omega_9, \omega_{10}, \omega_{11}, \omega_{13}, \omega_{15}\}$	3	$\{\omega_2, \omega_5, \omega_7, \omega_8, \omega_{13}, \omega_{15}\}$
$\{\omega_4, \omega_{12}\}$	4	$\{\omega_4, \omega_9, \omega_{11}, \omega_{12}\}$
	5	$\{\omega_{10}\}$

Table 4.1: well-ordered partitions based on an optimistic approach and based on the cardinality order

The default rule ordering induced by performing the partitioning algorithm is summarized by the left column of Table 4.1 . The cardinality ordering is presented in the right column of the same table as an element of comparison. Indeed, the π-pref net order is partial and cannot be put in parallel with a complete pre-order without changing incomparability into indifference.

The Pareto graph obtained by comparing quality vectors made with the symbolic weights of the π-pref net (as explained in Chapter 3) in Figure 4.1 is depicted in Figure 4.2. As in Chapter 3, we consider the case (a) where there is a unique symbolic weight per node and (b) there is a distinct symbolic weight for each conditional preference in each context. Going from the best $\omega_0 = abcd$ to the worst configuration $\omega_{10} = \bar{a}b\bar{c}d$, solid arrows represent comparisons where symbolic weights per variable are different given each context of parent, while dotted arrows depict additional comparisons where symbolic weights per variable are the same. We can check that all constraints are in agreement with comparisons induced by the Pareto order (and thus with the product chain rule).

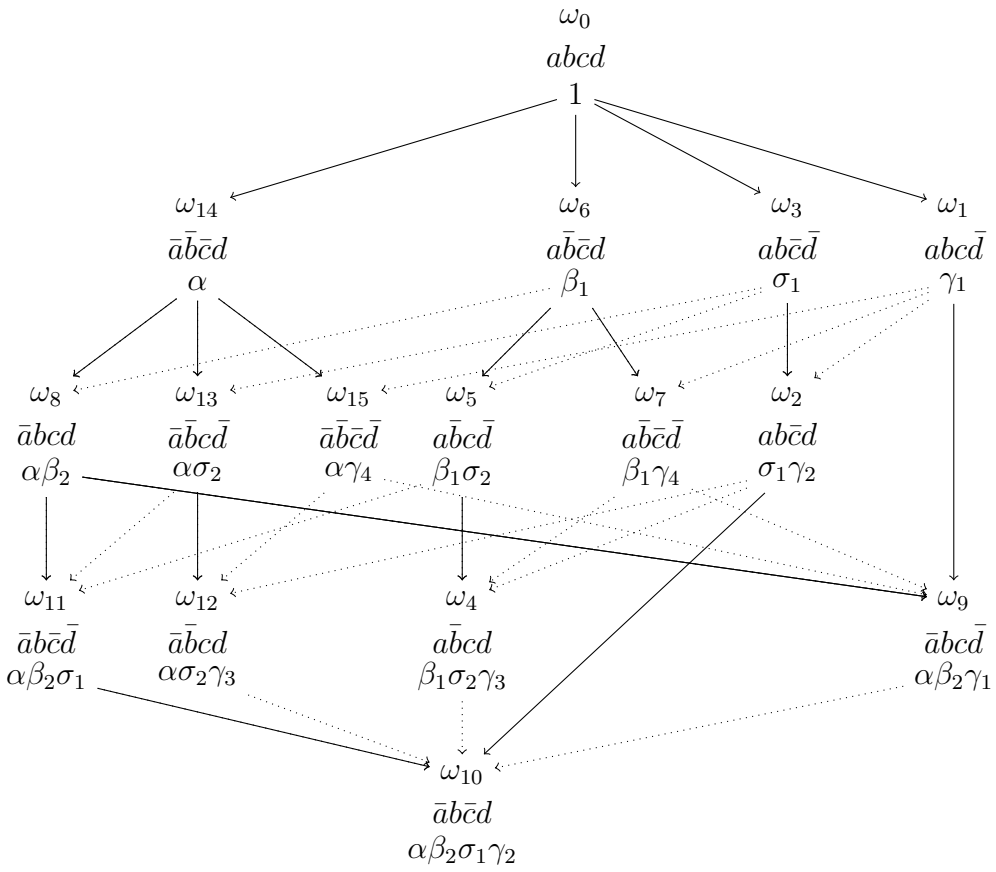


Figure 4.2: Pareto graph of π-pref nets in Figure 4.1

Observe that in the graph of Figure 4.2, the length of the longest sequence of comparisons is 5*. This means that assimilating incomparability with indifference we would*

lead to 5 *layers in a resulting well-ordered partition*[2]*. However, the order obtained with the default rules method has only* 4 *layers, which may suggest a lack of discrimination.*

Lastly, taking a closer look to the obtained optimistic default rules order, we notice that, not only the chain rule entails more preference levels than the optimistic ordering when the minimal specificity principle is applied (which is unsurprising), but also that the orderings are not in full agreement. For instance, we have $\omega_{10} \succ \omega_{12}$ *(and* $\omega_{10} \succ \omega_4$*) considering the default order while we have* $\omega_{12} \succ \omega_{10}$ *(*$\omega_4 \succ \omega_{10}$ *if symbolic degrees are equal) given the chain rule distribution (or the Pareto order).*

As mentioned before, the product chain rule agrees with the ordering of inclusion between subsets of nodes associated with violated preferences, and thus ranks alternatives according to the number of violated nodes, whereas, the specificity algorithm just finds the most compact ordering where constraints are respected. Nevertheless, the two approaches lead to distinct results that are not fully compatible, since it may be the case that $\exists\ \omega' \in \Omega$ such that for the chain rule approach $\omega' \succ \omega$ whereas for the minimum specificity based approach $\omega \succ \omega'$.

We can see that in Example 4.2, the worst configuration $\omega_{10} = \bar{a}b\bar{c}d$ is ranked on the lowest level by the product chain rule with no outgoing arrows (see Figure 4.2 and right column of Table 4.1), whereas it appears in the third level based on the minimum specificity approach being ranked as preferred to ω_4 and ω_{12}. This is due to the fact that in the third iteration the unsatisfied constraint c_7 do not prevent ω_{10} from being higher than the remaining configurations ω_4 and ω_{12}.

In Section 4.5, we shall present an improved algorithm that avoids this kind of phenomenon.

4.3.2 Pessimistic approach on default preferences

The minimum specificity Algorithm 4.1 outputs a well-ordered partition that clusters the worst configuration(s) with other less preferred ones in the same set. This can be explained by the focus on the best models of formulas. It does not provide information on the least preferred models. In order to better understand results of the optimistic preference interpretation approach, we also process preference statements using the maximum specificity principle with constraints expressed in terms of *guaranteed possibility* function $\Delta(.)$ (Equation 2.5). Let us recall that $\Delta(P) = \alpha$ expresses that all the

[2]In general, there are several ways to build such a partition

models of P are satisfactory at least at a degree α. The focus is thus directed towards the minimal degree of satisfaction over a set of choices.

In contrast, if we understand the default rule $p \rightsquigarrow q$ as the conditional constraint $\Delta(p \wedge q) > \Delta(p \wedge \bar{q})$ (pessimistic view). $p \wedge q$ is evaluated by its worst configuration [Benferhat et al., 2002b]. Such a view of a default rule can be similarly handled by the opposite principle, i.e., maximum specificity principle[3]. Let $\mathcal{C}_\Delta$ denote the set of constraints derived from the rules in $\mathcal{R}$ under a pessimistic semantics. Hence, $\mathcal{C}_\Delta$ is formally composed of

$$\mathcal{C}_\Delta = \{c_i : \Delta(p_i \wedge q_i) > \Delta(p_i \wedge \bar{q_i}), r_i : p_i \rightsquigarrow q_i \in \mathcal{R}\} \tag{4.4}$$

The following proposition establishes the agreement of the Pareto ordering with the constraints set $\mathcal{C}_\Delta$ expressing the preference statements.

Given a π-pref net, the associated possibility distribution obtained by the chain rule and the corresponding Pareto ordering between configurations agree with the constraints in $\mathcal{C}_\Delta$ expressing the preference statements of the π-pref net.

Proposition 4.2 *Given a π-pref net, let π be the possibility distribution obtained by the product chain rule and $\succ_\pi$ its associated preference relation between configurations of Ω. Let $\succ_{Pareto}$ be the preference relation on configurations of Ω obtained using the Pareto strategy. Let $\mathcal{C}_\Delta$ be the set of default constraints expressing the preference statements $p_i : q_i \succ \bar{q_i}$ of the π-pref net. Then, $\forall \omega, \omega' \in \Omega$ such that $\omega \succ_{Pareto} \omega'$ i.e. $\omega \succ_\pi \omega'$, there exists no constraint $c_i \in \mathcal{C}_\Delta$ such that $c_i : \Delta(p_i \wedge q_i) > \Delta(p_i \wedge \bar{q_i}) : \omega \models p_i \wedge \bar{q_i}, \omega' \models p_i \wedge q_i$.*

Proof 4.2 *Let ω and ω' be two configurations associated with their satisfaction vectors $\vec{\omega}$ and $\vec{\omega'}$. $\delta(\omega) > \delta(\omega')$ is equivalent to $\vec{\omega} \succ_{Pareto} \vec{\omega'}$ since each vector holds weights [Ben Amor et al., 2016b] assuming one weight per preference statement. It means that for each vector component (corresponding to a variable) either the two vectors are equal with the same satisfaction degree, or the component of $\vec{\omega'}$ is 0 and the same component for $\vec{\omega}$ is equal to some symbolic weight, say ρ, assuming $\vec{\omega'} \succ_{Pareto} \vec{\omega}$. Each inequality $\rho > 0$, corresponds to an inequality of the form $\delta(x_i \mid u_i') = \rho > (\bar{x_i} \mid u_i) = 0$. If $u_i = u_i'$ this refers to a preference statement in a conditional table $\delta(x_i \mid u_i) = \rho >$*

[3]Considering constraints $\mathcal{C}_\Delta$ associated to default rules in the knowledge Example 4.1, we would obtain the following well-ordered partition set $E = \{\{\omega_0, \omega_1, \omega_3, \omega_4, \omega_7\}, \{\omega_2, \omega_5, \omega_6\}\}$, which has less layers that the one obtained by the minimum specificity principle, which as already said is more appropriate to default reasoning.

$\delta(\bar{x_i} \mid u_i) = 0$ *equivalent as already said to the constraint* $\Delta(u_i \wedge x_i) > \Delta(u_i \wedge \bar{x}_i)$. *If* $u_i \neq u'_i$, *this violates no preference statement. It means that the comparison of* ρ *with* 0 *refers to different contexts and no preference statement is involved. Besides, the vector components where* $\vec{\omega}$ *and* $\vec{\omega'}$ *are equal cannot correspond to a violation of a preference statement.*

In Algorithm 4.1, it is enough to adapt the instruction of line 3 so that E_j involves solutions that do not appear on the *left-hand* side of constraints, i.e.,that are always dominated. In the output partition, configurations are ranked from the worst to the best ones, which means that $\omega_i \in E_j$ are less preferred than $\omega_i \in E_{j+1}$. The first set E_0 corresponds to the worst solution. The procedure for entailing a well-ordered partition from constraints in $\mathcal{C}_\Delta$ is summarized by Algorithm 4.2.

Algorithm 4.2: Algorithm of partitioning of Ω using the maximum specificity principle

Input: The set of solutions Ω
The set of possibilistic constraints $\mathcal{C}_\Delta$
Output: A well-ordered partition E

1 $j = 0$
2 **while** $\Omega \neq \emptyset$ **do**
3 $\quad E_j = \{\omega_i, i = 0, \cdots, m\}$ s.t. ω_i does not belong to the left-hand side of any constraint (ω_i is never the dominant configuration)
4 $\quad \Omega = \Omega \setminus E_j$
5 $\quad$ Remove from $\mathcal{C}_\Delta$ all satisfied constraints (their right-hand side are consistent with solutions of E_j)
6 $\quad j \leftarrow j + 1$
7 **end**
8 **return** E `/* solutions are ranked from worst to best */`

To compare with the optimistic approach, the maximum specificity principle on defaults permits to rank order configurations going from the worst to the best ones such as all constraints in $\mathcal{C}_\Delta$ are satisfied. However, as for the minimum specificity principle, the ordering obtained with the maximum specificity principle may present some contradictions with the chain rule ordering. For instance, in the following Example 4.3 which adopts a pessimistic approach to find a ranking on choices, one can check that the best configuration ω_0 is considered less preferred to ω_6 and ω_{14}. This is also due to a lack of constraints that forces ω_0 to be ranked better than ω_6 and ω_{14}.

Example 4.3 *Let us consider the same set of defaults of Example 4.2 which encodes specifications in Figure 3.6:* $\mathcal{R} = \{r_0 : a,\ r_1 : a \rightsquigarrow b,\ r_2 : \bar{a} \rightsquigarrow \bar{b},\ r_3 : b \rightsquigarrow c,\ r_4 : \bar{b} \rightsquigarrow \bar{c},$

$r_5 : bc \rightsquigarrow d$, $r_6 : b\bar{c} \rightsquigarrow d$, $r_7 : \bar{b}c \rightsquigarrow \bar{d}$, $r_8 : \bar{b}\bar{c} \rightsquigarrow \bar{d}\}$. *Following Equation 4.4, the associated constraints in* $\mathcal{C}_\Delta$ *are*

$$
\begin{array}{ll}
c_0 : a \succ \bar{a} & \min(\pi(\omega_0), \ldots, \pi(\omega_7)) > \min(\pi(\omega_8), \ldots, \pi(\omega_{15})) \\
c_1 : a : b \succ \bar{b} & \min(\pi(\omega_0), \ldots, \pi(\omega_3)) > \min(\pi(\omega_4), \ldots, \pi(\omega_7)) \\
c_2 : \bar{a} : \bar{b} \succ b & \min(\pi(\omega_{12}), \ldots, \pi(\omega_{15})) > \min(\pi(\omega_8), \ldots, \pi(\omega_{11})) \\
c_3 : b : c \succ \bar{c} & \min(\pi(\omega_0), \pi(\omega_1), \pi(\omega_8), \pi(\omega_9)) > \min(\pi(\omega_2), \pi(\omega_3), \pi(\omega_{10}), \pi(\omega_{11})) \\
c_4 : \bar{b} : \bar{c} \succ c & \min(\pi(\omega_6), \pi(\omega_7), \pi(\omega_{14}), \pi(\omega_{15})) > \min(\pi(\omega_4), \pi(\omega_5), \pi(\omega_{12}), \pi(\omega_{13})) \\
c_5 : bc : d \succ \bar{d} & \min(\pi(\omega_0), \pi(\omega_8)) > \min(\pi(\omega_1), \pi(\omega_9)) \\
c_6 : b\bar{c} : \bar{d} \succ d & \min(\pi(\omega_3), \pi(\omega_{11})) > \min(\pi(\omega_2), \pi(\omega_{10})) \\
c_7 : \bar{b}c : \bar{d} \succ d & \min(\pi(\omega_5), \pi(\omega_{13})) > \min(\pi(\omega_4), \pi(\omega_{12})) \\
c_8 : \bar{b}\bar{c} : d \succ \bar{d} & \min(\pi(\omega_6), \pi(\omega_{14})) > \min(\pi(\omega_7), \pi(\omega_{15}))
\end{array}
$$

Let us now perform Algorithm 4.2. The first partition set is composed of $E_0 = \{\omega_{10}\}$ *since it never dominates any configuration. The verified constraints are* c_0, c_2, c_3 *and* c_6. *The second partition set* $E_1 = \{\omega_4, \omega_9, \omega_{11}, \omega_{12}\}$ *satisfying constraints* c_1, c_4, c_5 *and* c_7. *Then* E_2 *is constructed holding configurations* $E_2 = \{\omega_0, \omega_1, \omega_2, \omega_3, \omega_5, \omega_7, \omega_8, \omega_{13}, \omega_{15}\}$. *The last remaining constraint* c_8 *is thus satisfied and the last level is composed of* $E_3 = \{\omega_6, \omega_{14}\}$. *The default rule ordering induced by performing the partitioning algorithm is summarized by the left column in Table 4.2 going from best to worst configuration(s). The inclusion ordering is presented in the right column of the same table.*

Maximum specificity ordering	**Levels**	**Cardinality ordering**
$\{\omega_6, \omega_{14}\}$	1	$\{\omega_0\}$
$\{\omega_0, \omega_1, \omega_2, \omega_3, \omega_5, \omega_7, \omega_8, \omega_{13}, \omega_{15}\}$	2	$\{\omega_1, \omega_3, \omega_6, \omega_{14}\}$
$\{\omega_4, \omega_9, \omega_{11}, \omega_{12}\}$	3	$\{\omega_2, \omega_5, \omega_7, \omega_8, \omega_{13}, \omega_{15}\}$
$\{\omega_{10}\}$	4	$\{\omega_4, \omega_9, \omega_{11}\omega_{12}\}$
	5	$\{\omega_{10}\}$

Table 4.2: Well-ordered partitions based on a pessimistic approach and based on the cardinality order

4.4 Well-ordered partition induced by a conditional preference graph

As explained in Section 4.3.1 of this chapter, when the user adopts an optimistic mind, a preference statement of the form $x_1x_2 \ldots x_N : x \succ \bar{x}$, where $X_1, X_2, \ldots X_N$ are parent variables, is expressed by the default preference rule $x_1x_2 \ldots x_N \rightsquigarrow x$, which

then translates into the constraint $\Pi(x_1x_2\dots x_Nx) > \Pi(x_1x_2\dots x_N\bar{x})$ simply written as the expression $x_1x_2\dots x_Nx \succ x_1x_2\dots x_N\bar{x}$. A conditional preference network is thus expressed by means of a collection of such constraints and by computing Algorithm 4.1, a ranking of configurations can then be achieved. However, in such a case whatever the number of variables the number of layers will remain equal to 3. As we shall see in this section, this behaviour is quite general for a family of graph structure, for instance a path graph structure leads also to 3 layers whatever its length. In the following, we study graph structures that always lead to 3 layers and we indicate some way of modifying the structure in order to have more layers.

First let us consider the case of a path graph, where each variable has exactly one variable as a parent (except for the root one) and the graph forms a single path (as on Figure 4.3). Hence, when variables are sorted in a topological order, conditional preference constraints are of the form $x_i : x_{i+1} > \bar{x}_{i+1}$ such that $\{x_{i+1}, \bar{x}_{i+1}\} \in \underline{X_{i+1}}$.

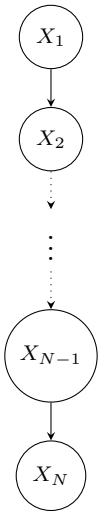


Figure 4.3: A linear DAG

Interpreting conditional preference statements as possibilistic constraints under the minimum specificity principle, any conditional preference path graph results into a well-ordered partition of solutions with exactly 3 layers.

Proposition 4.3 *Let $\mathcal{C}_\Pi = \{c_i : \Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q}_i), p_i : q_i \succ \bar{q}_i \mid i = 1, \dots, k\}$ be the set of possibilistic default constraints. Let E_Π be the well-ordered partition of solutions obtained using the minimum specificity principle on constraints of $\mathcal{C}_\Pi$. Then, any conditional preference path graph results into a partition E_Π of exactly 3 layers.*

Proof 4.3 *Let us assume a path graph $\mathcal{G}$ of N vertices namely $X_1, X_2, \dots, X_N$. The root node holds a preference constraint of the form $x_1 \succ \bar{x_1}$, whereas, for $i = 2, N$, the*

remaining nodes hold conditional preferences of the form $x_{i-1} \wedge x_i \succ x_{i-1} \wedge \overline{x_i}$ *for the preferred instantiation of the parent* X_{i-1} *and* $\overline{x_{i-1}} \wedge \overline{x_i} \succ \overline{x_{i-1}} \wedge x_i$ *for its negation. The non-dominated solution is unique and is defined by:*

$$\overline{\bar{x_1} \vee \bigvee_{i=2}^{N} (x_{i-1} \wedge \bar{x_i}) \vee \bigvee_{i=2}^{N} (\overline{x_{i-1}} \wedge x_i)} = x_1 \wedge \bigwedge_{i=2}^{N} (\overline{x_{i-1}} \vee x_i) \wedge \bigwedge_{i=2}^{N} (x_{i-1} \vee \bar{x_i}) = \wedge_{i=1}^{N} x_i$$

At the end of this iteration, the root constraint and the children constraints in the context of preferred parents configurations are satisfied by this best solution and can be deleted. The remaining constraints are $\overline{x_{i-1}} \wedge \overline{x_i} \succ \overline{x_{i-1}} \wedge x_i, i = 1, \ldots N$*. The dominated solutions are the models of* $\bigvee_{i=2}^{N} \overline{x_{i-1}} \wedge x_i$*. The non-dominated ones are thus of the form* $\bigwedge_{i=2}^{N} x_{i-1} \vee \overline{x_i}$*. This formula is consistent with the left-hand sides of the constraints* $\overline{x_{i-1}} \wedge \overline{x_i} \succ \overline{x_{i-1}} \wedge x_i$ *since they have in common the solution* $\wedge_{i=1}^{N} \overline{x_i}$*. Hence the solutions can be ranked in three levels:* $\wedge_{i=1}^{N} x_i$ *at the top forming* E_0*, and* $\bigvee_{i=2}^{N} \overline{x_{i-1}} \wedge x_i$ *at the bottom forming* E_2*, the rest being of the form* $E_1 = (\bigvee_{i=2}^{N} \overline{x_{i-1}} \wedge \overline{x_i}) \wedge \vee_{i=1}^{N} \overline{x_i}$*.*

The same result can be generalized for orders induced by the *maximum specificity principle*. Indeed, a preference statement of the form $x_1 x_2 : x \succ \bar{x}$ is translated into the constraint $\Delta(x_1 x_2 x) > \Delta(x_1 x_2 \bar{x})$, simply written by $x_1 x_2 x \succ x_1 x_2 \bar{x}$, under a pessimistic mind. The set of such constraints expressing specifications of a conditional preference network can infer a ranking of configurations (as explained in 4.3.2 to induce an ordering that verifies the same property as in Proposition 4.3.

Interpreting conditional preference statements as possibilistic constraints under the maximum specificity principle , any conditional preference path graph results into a well-ordered partition of solutions with exactly 3 layers.

Proposition 4.4 *Let* $\mathcal{C}_\Delta = \{c_i : \Delta(p_i \wedge q_i) > \Delta(p_i \wedge \bar{q_i}), p_i : q_i \succ \bar{q_i} \mid i = 1, \ldots, k\}$ *be the set of possibilistic default constraints. Let* E_Δ *be the well-ordered partition of solutions obtained using the minimum specificity principle on constraints of* $\mathcal{C}_\Delta$*. Then, any conditional preference path graph results into a partition* E_Δ *of exactly* 3 *layers.*

Proof 4.4 *Following the same reasoning as for Proof 4.3, constraints of a path graph conditional preference network are categorized in three sets of formulas:* $x_1 \succ \bar{x_i}$ *for the root node,* $x_{i-1} \wedge x_i \succ x_{i-1} \wedge \overline{x_i}$ *for the preferred context of the parent* X_{i-1} *of the remaining nodes and* $\overline{x_{i-1}} \wedge \overline{x_i} \succ \overline{x_{i-1}} \wedge x_i$ *for its negation. The first iteration of the algorithm permits to find the worst solution which is unique. It never appears on the*

left-hand side of constraints which means that it is defined by:

$$\overline{x_1 \vee \bigvee_{i=2}^{N}(x_{i-1} \wedge x_i) \vee \bigvee_{i=2}^{N}(\overline{x_{i-1}} \wedge \overline{x_i})} = \bar{x_1} \wedge \bigwedge_{i=2}^{N}(x_{i-1} \vee \overline{x_i}) \wedge \bigwedge_{i=2}^{N}(\overline{x_{i-1}} \vee x_i) = \wedge_{i=1}^{N}\overline{x_i}$$

After putting the dominated solution in the first level $E_0 = \wedge_{i=1}^{N}\overline{x_i}$*, the root constraint and the children constraints in the context of less preferred parents configurations are satisfied. The remaining constraints are* $x_{i-1} \wedge x_i \succ x_{i-1} \wedge \overline{x_i}$*. The second set of dominated solutions consist of models of the form* $\bigwedge_{i=2}^{N}\overline{x_{i-1}} \vee \overline{x_i}$*. This formula is consistent with the right-hand sides of the remaining constraints, namely the dominated solutions, which are of the form* $\bigvee_{i=2}^{N} x_{i-1} \wedge \bar{x_i}$*, since they have in common the solution* $\overline{x_i}$*. Therefore the bottom level is composed of solutions of the form* $E_2 = \bigvee_{i=2}^{N} x_{i-1} \wedge \bar{x_i}$ *and* $\bigvee_{i=2}^{N}\overline{x_{i-1}} \vee x_i$ *form the intermediate remaining level* E_1*.*

Actually, the number of layers for ordering preferences using the constraint based algorithm increases by adding edges between the grandparent nodes and those of children nodes. The following example represents an illustration that confirms this claim.

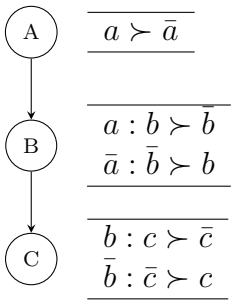


Figure 4.4: A path preference network

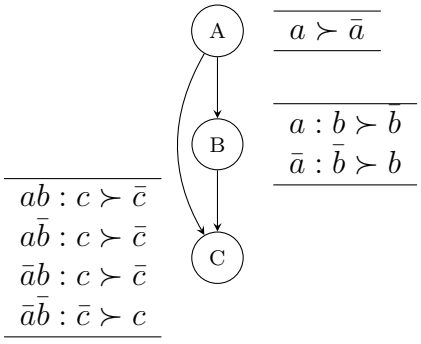


Figure 4.5: Example of a preference network

Example 4.4 *The graph on Figure 4.5 differs from graph on Figure 4.4 by an additional edge going from node A to node C. Applying the algorithm 4.1 yields 4 preference levels. Adding the edge* $A \rightarrow C$ *to the preference network of Figure 4.4, have changed statements and therefore constraints of the node C. Let* $\Omega = \{\omega_0 = abc, \omega_1 = ab\bar{c}, \omega_2 = a\bar{b}c, \omega_3 = a\bar{b}\bar{c}, \omega_4 = \bar{a}bc, \omega_5 = \bar{a}b\bar{c}, \omega_6 = \bar{a}\bar{b}c, \omega_7 = \bar{a}\bar{b}\bar{c}\}$ *be the power set of possible configurations. Adopting an optimistic mind, the set of constraints relative to the expressed preferences of Figure 4.5 is* $\mathcal{C}_\Pi = \{c_0 = \Pi(a) > \Pi(\bar{a}), c_1 = \Pi(ab) > \Pi(a\bar{b}), c_2 = \Pi(\bar{a}\bar{b}) > \Pi(\bar{a}b), c_3 = \Pi(abc) > \Pi(ab\bar{c}), c_4 = \Pi(a\bar{b}c) > \Pi(a\bar{b}\bar{c}), c_5 = \Pi(\bar{a}bc) > \Pi(\bar{a}b\bar{c}), c_6 = \Pi(\bar{a}\bar{b}c) > \Pi(\bar{a}\bar{b}\bar{c})\}$*. The induced well-ordered partition E is*

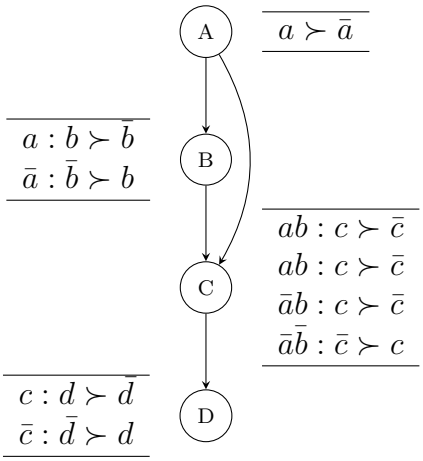
composed of the respective sets $\{\omega_0\}$, $\{\omega_1, \omega_2, \omega_7\}$, $\{\omega_3, \omega_4, \omega_6\}$ *and* $\{\omega_5\}$ *satisfying the respective sets of constraints* $\{c_0, c_1, c_3\}$, $\{c_2, c_4, c_6\}$ *and* $\{c_5\}$. *At the end of the second iteration, the only remaining constraint is* c_5 *enforcing* $\omega_5 = \bar{a}b\bar{c}$ *down to a fourth level. If we consider the path graph on Figure 4.4, the minimum specificity principle ranks configurations within a* 3 *layer set, namely* $E = \{\{\omega_0\}, \{\omega_1, \omega_3, \omega_7\}, \{\omega_2, \omega_4, \omega_5, \omega_6\}\}$.

Using the pessimistic reasoning approach, preference specifications translate to the same constraints previously announced where the possibilistic measure Π *is replaced by the guaranteed possibility measure* Δ. *The induced well-ordered partition of specifications in Figure 4.5 E is composed of* 4 *layers:* $E_0 = \{\omega_5\}$, $E_1 = \{\omega_3, \omega_4, \omega_6\}$, $E_2 = \{\omega_1, \omega_2, \omega_7\}$ *and* $E_3 = \{\omega_0\}$. *After the first iteration, configurations in subset* E_0 *satisfy constraints* c_0, c_2 *and* c_5. *After the second iteration, the partition set satisfies all remaining constraints except of* c_3 *which adds two subset layers in E namely* E_2, *that makes* c_3 *true, and* E_3. *If we consider network in Figure 4.4, the default algorithm yields the well-ordered partition* $E = \{\{\omega_5\}, \{\omega_2, \omega_4, \omega_6\}, \{\omega_0, \omega_1, \omega_3, \omega_7\}\}$ *which is composed of only* 3 *layers.*

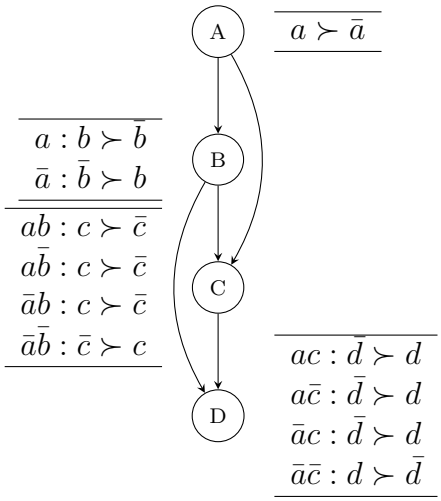
In fact, adding edges from a parent node to a grand-children node has not always the effect of increasing the number of layers. However, it is possible to have more than layers by adding edges from parents to grand-children in appropriate places in a path graph. However, given a fixed number N of decision variables, the size of the order obtained from the completely connected DAG structure corresponds to the maximum number of levels that can be obtained for any structure defined on N variables. See the following Example 4.5 that describes orderings induced from networks that describe preferences over 4 decision variables.

Example 4.5 *Given* 4 *decision variables A, B, C and D the set of configurations* Ω *is the same as defined in Example 4.2. Let us consider networks in Figure 4.6. Table 4.3 give orderings induced from these networks. The network structures in Figures 4.1 and 4.6(a) both hold one arc from a grand-parent to a child node and both induce an ordering composed of* 4 *layers. Adding an arc from a grand-grand-parent to a child node does not have an impact on the number of layers of the induced ordering (see results relative to Figure 4.6(b) for example). Figure 4.6(c) combine dependencies between grand-parent node generations described by the mentioned networks, i.e.,* $A \rightarrow C$ *and* $B \rightarrow D$. *The number of the induced well-ordered partition remains unchanged with* 4 *levels. Moreover, combining arcs between grand-parents and children, and between grand-grand-parents and children does not effect the number of the partition layers (see orderings of network in Figure 4.6(e)). However in the completely connected DAG*

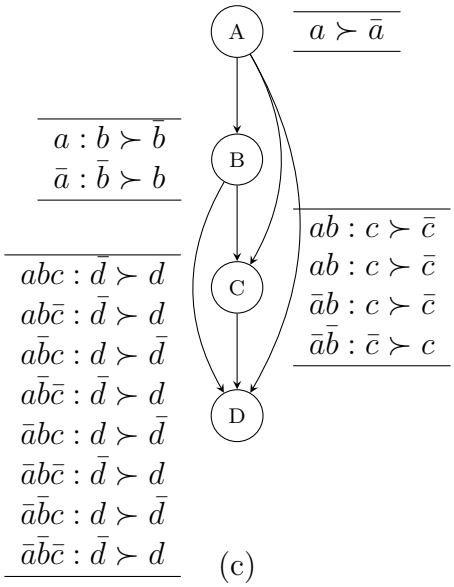
structure in Figure 4.6(c), the induced default ordering is defined on 5 *layers, unlike other networks defined on* 4 *layers.*



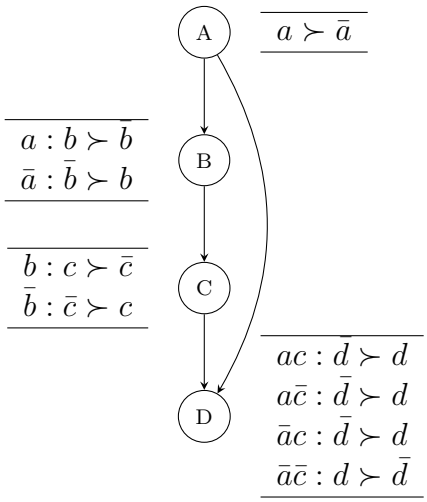
(a)



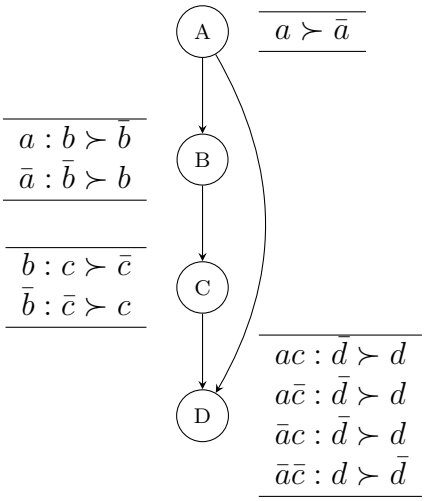
(b)



(c)



(d)



(e)

Figure 4.6: Examples of conditional preference networks

Figure 4.6(a):

Levels	**Minimum specificity ordering**	**Maximum specificity ordering**
1	$\{\omega_0\}$	$\{\omega_0, \omega_1\}$
2	$\{\omega_1, \omega_3, \omega_4, \omega_5, \omega_{15}\}$	$\{\omega_2, \omega_3, \omega_4, \omega_5, \omega_8, \omega_{12}, \omega_{14}, \omega_{15}\}$
3	$\{\omega_2, \omega_6, \omega_7, \omega_8, \omega_9, \omega_{12}, \omega_{13}, \omega_{14}\}$	$\{\omega_6, \omega_7, \omega_9, \omega_{11}, \omega_{13}\}$
4	$\{\omega_{10}, \omega_{11}\}$	$\{\omega_{10}\}$

Figure 4.6(b):

Levels	**Minimum specificity ordering**	**Maximum specificity ordering**
1	$\{\omega_1\}$	$\{\omega_0, \omega_1\}$
2	$\{\omega_0, \omega_3, \omega_4, \omega_{15}\}$	$\{\omega_2, \omega_3, \omega_4, \omega_5, \omega_7, \omega_9, \omega_{12}, \omega_{14}, \omega_{15}\}$
3	$\{\omega_2, \omega_5, \omega_6, \omega_7, \omega_8, \omega_9, \omega_{12}, \omega_{13}, \omega_{14}\}$	$\{\omega_6, \omega_8, \omega_{11}, \omega_{13}\}$
4	$\{\omega_{10}, \omega_{11}\}$	$\{\omega_{10}\}$

Figure 4.6(c):

Levels	Minimum and maximum specificity orderings
1	$\{\omega_1\}$
2	$\{\omega_0, \omega_3, \omega_4, \omega_{15}\}$
3	$\{\omega_2, \omega_5, \omega_7, \omega_8, \omega_{12}, \omega_{14}\}$
4	$\{\omega_6, \omega_9, \omega_{11}, \omega_{13}\}$
5	$\{\omega_{10}\}$

Figure 4.6(d):

Levels	Minimum specificity ordering	Maximum specificity ordering
1	$\{\omega_1\}$	$\{\omega_3, \omega_7\}$
2	$\{\omega_0, \omega_3, \omega_4, \omega_{15}\}$	$\{\omega_0, \omega_1, \omega_2, \omega_5, \omega_6, \omega_9, \omega_{13}, \omega_{14}, \omega_{15}\}$
3	$\{\omega_2, \omega_4, \omega_5, \omega_6, \omega_9, \omega_{10}, \omega_{11}, \omega_{13}, \omega_{15}\}$	$\{\omega_4, \omega_8, \omega_{10}, \omega_{12}\}$
4	$\{\omega_8, \omega_{12}\}$	$\{\omega_{11}\}$

Figure 4.6(e):

Levels	Minimum specificity ordering	Maximum specificity ordering
1	$\{\omega_0\}$	$\{\omega_0, \omega_1, \omega_4\}$
2	$\{\omega_1, \omega_2, \omega_4, \omega_5\omega_{14}\}$	$\{\omega_2, \omega_3, \omega_5, \omega_6, \omega_9, \omega_{13}, \omega_{14}, \omega_{15}\}$
3	$\{\omega_3, \omega_6, \omega_7, \omega_9, \omega_{13}, \omega_{15}\}$	$\{\omega_7, \omega_8, \omega_{10}, \omega_{12}\}$
4	$\{\omega_8, \omega_{10}, \omega_{11}, \omega_{12}\}$	$\{\omega_{11}\}$

Table 4.3: well-ordered partitions induced from networks in Figure 4.6

Whatever the topology of the graph and whatever the specificity principle, if the network does not hold edges from the grandparents nodes to children nodes, the number of elements forming the well-ordered partition remains constant and equal to 3. The following propositions confirm this claim for topologies of Figure 4.7 and Figure 4.8, respectively.

Given any conditional preference network with one parent node and $N-1$ children, the well-ordered partition of configurations output by the minimum specificity principle based algorithm has exactly 3 layers.

Proposition 4.5 *Let* $\mathcal{C}_\Pi = \{c_i : \Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q}_i), p_i : q_i \succ \bar{q}_i \mid i = 1, \ldots, k\}$ *be the set of possibilistic default constraints. Let* E_Π *be the well-ordered partition of*

solutions obtained using the minimum specificity principle on constraints of $\mathcal{C}_\Pi$*. Then, any conditional preference network with one parent node and* $N-1$ *children results into a partition* E_Π *of exactly* 3 *layers.*

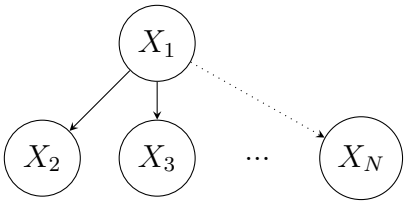


Figure 4.7: A graph with one parent and N children

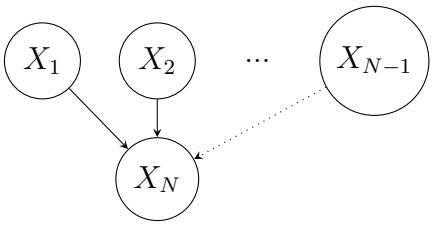


Figure 4.8: A graph with N parents and one child

Proof 4.5 *Let us consider the graph* $\mathcal{G}$ *of Figure 4.7 with one parent and* $N-1$ *children node. The root has a preference statement* $x_1 \succ \bar{x}_1$*. For* $i = 2 \cdots, N$*, each child node bears conditional constraints of the form* $x_1x_i \succ x_1\bar{x}_i$ *and* $\bar{x}_1\bar{x}_i \succ \bar{x}_1x_i$*. The un-dominated set is the complement of propositions on the right of constraints, namely* $E_0 = x_1 \bigwedge_{i=2}^{N} x_i$*. Constraints* $x_1 \succ \bar{x}_1$ *and* $x_1x_i \succ x_1\bar{x}_i$ *are satisfied by this solution and are then deleted. The second level set* E_1 *contains models of* $\overline{\bar{x}_1 \wedge \bigvee_{i=2}^{N} x_i} = x_1 \vee \bigwedge_{i=2}^{N} \bar{x}_i$*. Note that all the left-hand side propositions* $\bar{x}_1\bar{x}_i$ *of the remaining constraints are consistent with* $x_1 \vee \bigwedge_{i=2}^{N} \bar{x}_i$*. Hence again 3 levels are obtained.*

Given any conditional preference network with one parent node and $N-1$ children, the well-ordered partition of configurations output by the maximum specificity principle based algorithm has exactly 3 layers.

Proposition 4.6 *Let* $\mathcal{C}_\Delta = \{c_i : \Delta(p_i \wedge q_i) > \Delta(p_i \wedge \bar{q}_i), p_i : q_i \succ \bar{q}_i \mid i = 1, \ldots, k\}$ *be the set of possibilistic default constraints. Let* E_Δ *be the well-ordered partition of solutions obtained using the minimum specificity principle on constraints of* $\mathcal{C}_\Pi$*. Then, any conditional preference network with one parent node and* $N-1$ *children results into a partition* E_Δ *of exactly* 3 *layers.*

Proof 4.6 *Considering the same graph* $\mathcal{G}$ *in Figure 4.7, the root has a preference statement* $x_1 \succ \bar{x}_1$*, while each child node carries conditional constraints of the form*

$x_1 x_i \succ x_1 \bar{x_i}$ *and* $\bar{x_1}\bar{x_i} \succ \bar{x_1} x_i$*, for* $i = 2 \cdots, N$*. The rejected solution is unique and is defined by the complement of propositions on the left of constraints, namely* $E_0 = \bar{x_1} \bigwedge_{i=2}^{N} x_i$*. Constraints* $x_1 \succ \bar{x_1}$ *and* $\bar{x_1}\bar{x_i} \succ x_1 \bar{x_i}$ *are satisfied by this solution and are then deleted. The second level set* E_1 *contains that never appear on the left side of the remaining constraints. It is thus composed of models of* $\overline{x_1 \wedge \bigvee_{i=2}^{N} x_i} = \bar{x_1} \vee \bigwedge_{i=2}^{N} \bar{x_i}$*. This formula is consistent with the left-hand side propositions* $x_1 \bar{x_i}$ *of the remaining constraints. Hence again 3 levels are obtained with* $E_0 = \bar{x_1} \bigwedge_{i=2}^{N} x_i$*,* $E_1 = \bar{x_1} \vee \bar{x_i}$ *and* $E_2 = x_1 \bar{x_i}$*.*

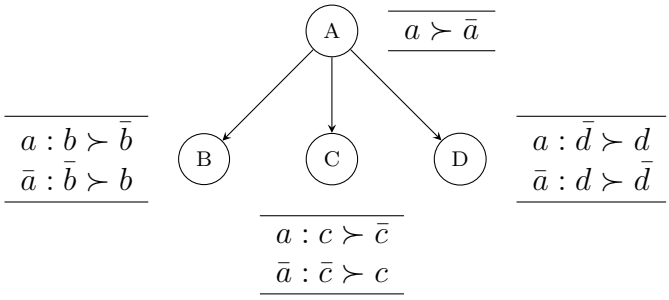


Figure 4.9: A preference network with one parent and 3 children

Example 4.6 *The graph in Figure 4.9 depicts a preference network composed of one parent node A with its 3 children B, C and D. The set of possible configurations is* $\Omega = \{\omega_0 = abcd, \omega_1 = abc\bar{d}, \omega_2 = ab\bar{c}d, \omega_3 = ab\bar{c}\bar{d}, \omega_4 = a\bar{b}cd, \omega_5 = a\bar{b}c\bar{d}, \omega_6 = a\bar{b}\bar{c}d, \omega_7 = a\bar{b}\bar{c}\bar{d}, \omega_8 = \bar{a}bcd, \omega_9 = \bar{a}bc\bar{d}, \omega_{10} = \bar{a}b\bar{c}d, \omega_{11} = \bar{a}b\bar{c}\bar{d}, \omega_{12} = \bar{a}\bar{b}cd, \omega_{13} = \bar{a}\bar{b}c\bar{d}, \omega_{14} = \bar{a}\bar{b}\bar{c}d, \omega_{15} = \bar{a}\bar{b}\bar{c}\bar{d}\}$*. Preference specifications are translated into the following constraints* $\mathcal{C}_\Pi = \{c_0 = \Pi(a) > \Pi(\bar{a}), c_1 = \Pi(ab) > \Pi(a\bar{b}), c_2 = \Pi(\bar{a}\bar{b}) > \Pi(\bar{a}b), c_3 = \Pi(ac) > \Pi(a\bar{c}), c_4 = \Pi(\bar{a}\bar{c}) > \Pi(\bar{a}c), c_5 = \Pi(a\bar{d}) > \Pi(ad), c_6 = \Pi(\bar{a}d) > \Pi(\bar{a}\bar{d})\}$*. The well-ordered partition induced from the minimum specificity postulate is* $E = \{\{\omega_1\}, \{\omega_0, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6, \omega_7, \omega_{14}\}, \{\omega_8, \omega_9, \omega_{10}, \omega_{11}, \omega_{12}, \omega_{13}, \omega_{15}\}\}$*. It indeed involves 3 layer sets. The first partition satisfies constraints* c_0, c_1, c_3 *and* c_5*. The remaining constraints are verified by the second partition set which leads to put configurations* $\overline{a\bar{b} \vee a\bar{c} \vee ad}$ *in the last layer.*

Under the maximum specificity principle configurations on both hand-sides of constraints remain the same while Δ *takes the place of* Π*. The well-ordered partition induced from the pessimistic reasoning is* $E = \{\{\omega_9\}, \{\omega_6, \omega_8, \omega_{10}, \omega_{11}, \omega_{12}, \omega_{13}, \omega_{14}, \omega_{15}\}, \{\omega_0, \omega_2, \omega_3, \omega_4, \omega_5, \omega_7\}\}$*. It is indeed composed of* 3 *levels. Configurations of the first layer validate constraints* c_0 *,*c_2*,* c_4 *and* c_6*. Those in the second layer verify the remaining ones. In addition of confirming Propositions 4.5 and 4.6, this result also confirms*

that, for such structure, the graph size does not impact the number of the partition layers.

Given any conditional preference network with $N-1$ independent parent nodes and one child variable, the well-ordered partition of configurations output by the minimum specificity principle based algorithm has exactly 3 levels.

Proposition 4.7 *Let $\mathcal{C}_\Pi = \{c_i : \Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q_i}), p_i : q_i \succ \bar{q_i} \mid i = 1, \ldots, k\}$ be the set of possibilistic default constraints. Let E_Π be the well-ordered partition of solutions obtained using the minimum specificity principle on constraints of $\mathcal{C}_\Pi$. Then, any conditional preference network with $N-1$ independent parent nodes and one child variable results into a partition E_Π of exactly 3 layers.*

Proof 4.7 *Assume the graph $\mathcal{G}$ of Figure 4.8. In the same vein as Propositions 4.3 and 4.5, parent nodes bear constraints $x_i \succ \bar{x_i}$ for $i = 1, \cdots, N-1$. Denote by u the disjunction of parents configurations such that x_N is preferred to $\bar{x}_N$, where it is supposed that u satisfies $\wedge_{i=1}^{N-1} x_i$ and $\bar{u}$ satisfies $\wedge_{i=1}^{N-1} \bar{x}_i$. The remaining conditional constraints at step 2 reduce to $ux_N \succ u\bar{x}_N$ and $\bar{u}\bar{x}_N \succ \bar{u}x_N$. Obviously we get 3 levels again.*

Given any conditional preference network with $N-1$ independent parent nodes and one child variable, the well-ordered partition of configurations output by the maximum specificity principle based algorithm has exactly 3 levels.

Proposition 4.8 *Let $\mathcal{C}_\Delta = \{c_i : \Delta(p_i \wedge q_i) > \Delta(p_i \wedge \bar{q_i}), p_i : q_i \succ \bar{q_i} \mid i = 1, \ldots, k\}$ be the set of possibilistic default constraints. Let E_Π be the well-ordered partition of solutions obtained using the minimum specificity principle on constraints of $\mathcal{C}_\Delta$. Then, any conditional preference network with $N-1$ independent parent nodes and one child variable results into a partition E_Δ of exactly 3 layers.*

Proof 4.8 *In a like manner as for Proposition 4.7, graphs sharing the same structure as in Figure 4.8 bear constraints of the form $x_i \succ \bar{x_i}$ for root nodes and $ux_N \succ u\bar{x}_N$ and $\bar{u}\bar{x}_N \succ \bar{u}x_N$ for the child node. The first level is composed of models of $\wedge_{i=1}^{N-1} \bar{x_i} \wedge u\bar{x_N} \vee \bar{u}x_N$. This formula only leaves the constraint $ux_N \succ u\bar{x}_N$ unsatisfied which yields two partition levels namely $E_1 = \bar{u} \vee \bar{x_N}$ and $E_2 = ux_N$.*

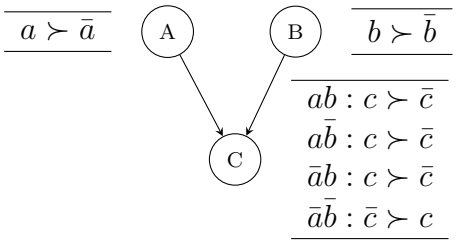


Figure 4.10: A preference network with 2 parents and one child

Example 4.7 *Unlike network in Figure 4.9, Figure 4.10 depicts a graph with several root nodes and only one child. Possible configurations* Ω *is the same as in Example 4.4. Preference constraints are* $\mathcal{C}_\Pi = \{c_0 = \Pi(a) > \Pi(\bar{a}), c_1 = \Pi(b) > \Pi(\bar{b}), c_2 = \Pi(abc) > \Pi(ab\bar{c}), c_3 = \Pi(a\bar{b}c) > \Pi(a\bar{b}\bar{c}), c_4 = \Pi(\bar{a}bc) > \Pi(\bar{a}b\bar{c}), c_5 = \Pi(\bar{a}\bar{b}\bar{c}) > \Pi(\bar{a}\bar{b}c)\}$*. The first layer of the well-ordered partition induced from the minimum specificity postulate is composed of the unique optimal configuration:* ω_0*. This makes constraints* c_0, c_2, c_4 *and* c_6 *verified. The second partition layer is composed of configurations* $\{\omega_1, \omega_2, \omega_4, \omega_7\}$ *satisfying therefore all the remaining constraints, namely* c_1, c_3 *and* c_5*. The last unranked configurations in* Ω *compose the final layer. The well-ordered partition is thus divided in* 3 *levels of subsets.*

Adopting a pessimistic point of view, specifications are translated into constraints using the measure Δ*. Solutions of both sides constraints remain unchanged. The first partition level is composed of the unique worst configuration namely* ω_6 *which verifies constraints* c_0*,* c_1 *and* c_5*. The second layer is holds configurations* $\{\omega_1, \omega_3, \omega_5, \omega_7\}$*. The remaining constraints are satisfied and the last partition set contains* $\{\omega_0, \omega_2, \omega_4\}$*.*

The last result considers a more general structure (see Figure 4.11) we call *quasi-linear* and subsumes the preceding results.

Consider a conditional preference network $\mathcal{G} = \{\mathcal{X}, E\}$, where the set $\mathcal{X}$ of variables is partitioned in $\mathcal{X}_1, \cdots, \mathcal{X}_N$. Suppose $\forall j \in [1, m]$, each variable $X \in \mathcal{X}_i$ has its parents only at the previous level $i-1$, i.e., $U_X \subseteq \mathcal{X}_{i-1} \forall X \in \mathcal{X}_i$. The minimum specificity principle results in a well-ordered 3-partition of solutions.

Proposition 4.9 *Let* $\mathcal{C}_\Pi = \{c_i : \Pi(p_i \wedge q_i) > \Pi(p_i \wedge \bar{q}_i), p_i : q_i \succ \bar{q}_i \mid i = 1, \ldots, k\}$ *be the set of possibilistic default constraints. Let* E_Π *be the well-ordered partition of solutions obtained using the minimum specificity principle on constraints of* $\mathcal{C}_\Pi$*. Consider a conditional preference network* $\mathcal{G} = \{\mathcal{X}, E\}$*, where the set* $\mathcal{X}$ *of variables is partitioned in* $\mathcal{X}_1, \cdots, \mathcal{X}_N$*. Suppose* $\forall j \in [1, m]$*, each variable* $X \in \mathcal{X}_i$ *has its parents only at*

the previous level $i-1$, i.e., $U_X \subseteq \mathcal{X}_{i-1} \forall X \in \mathcal{X}_i$. Then, any conditional preference network $\mathcal{G}$ results into a partition E_Π of exactly 3 layers.

Proof 4.9 *$\forall i = 2,\cdots,N$ all nodes $X_i \in \mathcal{X}_i$ are associated to the conditional constraints $u_i x_i \succ u_i \bar{x}_i$ and $\overline{u_i x_i} \succ \overline{u_i} x_i$, where u_i is the disjunction of configurations of U_{X_i} such that x_i is preferred to $\bar{x}_i$, plus $x_1 \succ \overline{x_1}$ for nodes $X_1 \in \mathcal{X}_1$. Assuming $\bigwedge_{X_{i-1} \in U_{X_i}} x_i \models u_i$ and $\bigwedge_{X_{i-1} \in U_{X_i}} \bar{x}_i \models \bar{u}_i$, the non-dominated set E_0 reduces to $(\bigwedge_{X_1 \in \mathcal{X}_1} x_1) \wedge \bigwedge_{i=2}^{N} \bigwedge_{X_i \in \mathcal{X}_i} [(\bar{u}_i \vee x_i) \wedge (u_i \vee \bar{x}_i)] = \bigwedge_{X \in \mathcal{X}} x$. After deleting the satisfied constraints, the remaining ones are $\forall X_i \in \mathcal{X}_i$, $\bar{u}_i \bar{x}_i \succ \bar{u}_i x_i$, $\forall i = 2,\cdots,N$. The undominated set $E_1 \cup E_0$ forms the models of $\bigwedge_{i=2}^{N} \bigwedge_{X_i \in \mathcal{X}_i} (u_i \vee \bar{x}_i)$. We can easily check that $\bar{u}_i \bar{x}_i$ is consistent with E_1 since they share $\bar{x}_i$, $\forall i = 2,\cdots,N$ and $\forall X_i \in \mathcal{X}_i$. By consequence the third element of the well-ordered partition E_2 equals $\bigvee_{i=2}^{N} \bigvee_{j=1}^{m} \bigvee_{X_{ij} \in \mathcal{X}_i} \bar{u}_i \wedge x_i$.*

Consider a conditional preference network $\mathcal{G} = \{\mathcal{X}, E\}$, where the set $\mathcal{X}$ of variables is partitioned in $\mathcal{X}_1,\cdots,\mathcal{X}_N$. Suppose $\forall j \in [1,m]$, each variable $X \in \mathcal{X}_i$ has its parents only at the previous level $i-1$, i.e., $U_X \subseteq \mathcal{X}_{i-1} \forall X \in \mathcal{X}_i$. The maximum specificity principle results in a well-ordered 3-partition of solutions.

Proposition 4.10 *Let $\mathcal{C}_\Delta = \{c_i : \Delta(p_i \wedge q_i) > \Delta(p_i \wedge \bar{q}_i), p_i : q_i \succ \bar{q}_i \mid i = 1,\ldots,k\}$ be the set of possibilistic default constraints. Let E_Δ be the well-ordered partition of solutions obtained using the minimum specificity principle on constraints of $\mathcal{C}_\Delta$. Consider a conditional preference network $\mathcal{G} = \{\mathcal{X}, E\}$, where the set $\mathcal{X}$ of variables is partitioned in $\mathcal{X}_1,\cdots,\mathcal{X}_N$. Suppose $\forall j \in [1,m]$, each variable $X \in \mathcal{X}_i$ has its parents only at the previous level $i-1$, i.e., $U_X \subseteq \mathcal{X}_{i-1} \forall X \in \mathcal{X}_i$. Then, any conditional preference network $\mathcal{G}$ results into a partition E_Δ of exactly 3 layers.*

Proof 4.10 *$\forall i = 2,\cdots,N$ all nodes $X_i \in \mathcal{X}_i$ are associated to the conditional constraints $u_i x_i \succ u_i \bar{x}_i$ and $\overline{u_i x_i} \succ \overline{u_i} x_i$, where u_i is the disjunction of configurations of U_{X_i} such that x_i is preferred to $\bar{x}_i$, plus $x_1 \succ \overline{x_1}$ for nodes $X_1 \in \mathcal{X}_1$. Assuming $\bigwedge_{X_{i-1} \in U_{X_i}} x_i \models u_i$ and $\bigwedge_{X_{i-1} \in U_{X_i}} \bar{x}_i \models \bar{u}_i$, the dominated set E_0 reduces to $(\bigwedge_{X_1 \in \mathcal{X}_1} x_1) \wedge \bigwedge_{i=2}^{N} \bigwedge_{X_i \in \mathcal{X}_i} [(\bar{u}_i \vee \bar{x}_i) \wedge (u_i \vee x_i)] = \bigwedge_{X \in \mathcal{X}} \bar{x}$. After deleting the satisfied constraints, the remaining ones are $\forall X_i \in \mathcal{X}_i$, $u_i x_i \succ u_i \bar{x}_i$, $\forall i = 2,\cdots,N$. The dominated set $E_1 \cup E_0$ forms the models of $\bigwedge_{i=2}^{N} \bigwedge_{X_i \in \mathcal{X}_i} (\bar{u}_i \vee \bar{x}_i)$. We can easily check that $\bar{u}_i x_i$ is consistent with E_1 since they share x_i, $\forall i = 2,\cdots,N$ and $\forall X_i \in \mathcal{X}_i$. By consequence the third element of the well-ordered partition E_2 equals $\bigvee_{i=2}^{N} \bigvee_{j=1}^{m} \bigvee_{X_{ij} \in \mathcal{X}_i} u_i \wedge x_i$.*

Thus, up to very quite special graph structures, the default-like rules approach leads to a complete pre-order with only 3 levels which is not very discriminating. In

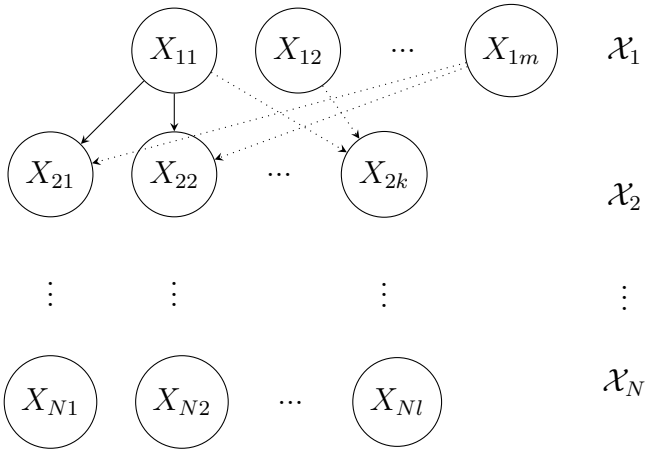


Figure 4.11: A quasi-linear DAG

the next section, we explain how to repair another problem of this approach which is to conflict with the Pareto ordering.

4.5 Improving possibilistic default rules-based orderings

The ordering generated by means of the default rules using some specificity principles not only leads to a sparsely discriminant ranking but can also lead to contradictions with the Pareto order. This only take place when symbolic weights of variables in the context of parents are equal (unique symbol per node). In fact, at some step of the ranking algorithm, some configurations might no longer appear in any of the remaining constraints, although they are assigned in a set in the partition, the highest or the lowest possible one, depending on whether we adopt the minimum or maximum specificity principle. [Ben Amor et al., 2021a].

In order to overcome this problem, we propose Algorithm 4.3 which is a new version of *Benferhat et. al* algorithm 4.1 that considers the same sets of inputs but checks at each iteration if there exists configurations that do not appear in any remaining constraint. These configurations are assigned to a set E'_{Π} (respectively E'_{Δ}) and are considered just not better than previously ranked configurations. $Config(C_{\Pi})$ (respectively $Config(C_{\Delta})$) returns all configurations in C_{Π} (respectively C_{Δ}). This improved version results in a partial order on configurations that could be in full compliance with the Pareto order. We illustrate this new algorithm in the Example 4.8.

Algorithm 4.3: Ordering Ω using an optimistic approach (Improved version)

Input: Ω , $\mathcal{C}_\Pi$
Output: E_Π, E'_Π

```
1  i = 0
2  while C_Π ≠ ∅ do
3  |  foreach ω ∈ Ω do
4  |  |  if ω ∉ Config(C_Π) then
5  |  |  |  E'_Π[i] = E'_Π[i] ∪ {ω};
6  |  |  |  Ω = Ω \ {ω}
7  |  end
8  |  foreach ω ∈ Ω do
9  |  |  foreach c ∈ C_Π do
10 |  |  |  if  ω ∉ RC(c) then
11 |  |  |  |  E_Π[i] = E_Π[i] ∪ {ω}
12 |  |  |  |  Ω = Ω \ {ω}
13 |  |  end
14 |  end
15 end
16 foreach ω ∈ E_Π[j] do
17 |  foreach c ∈ C_Π do
18 |  |  if  ω ∈ LC(c) then
19 |  |  |  C_Π = C_Π \ {c}
20 |  end
21 |  i = i + 1
22 end
23 if Ω ≠ ∅ then
24 |  foreach  ω ∈ Ω do
25 |  |  E_Π[i] = E_Π[i] ∪ {ω}
26 |  end
27 Return E_Π, E'_Π
```

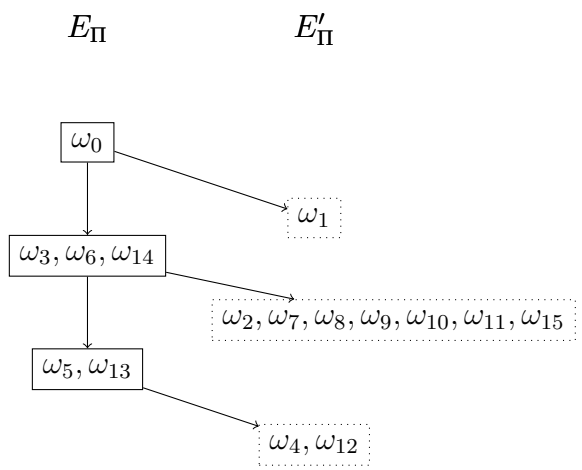


Figure 4.12: Improved optimistic ordering of π-pref net in Figure 4.1

Example 4.8 *Let us consider pursue with π-pref net in Figure 4.1. Under the optimistic or pessimistic approach, preference statements are translated into default rules and then to the following constraints:*

$c_0 : a \succ \bar{a}$ $\quad L(c_0) = \{\omega_0, \ldots, \omega_7\}, R(c_0) = \{\omega_8, \ldots, \omega_{15}\}$

$c_1 : a : b \succ \bar{b}$ $\quad L(c_1) = \{\omega_0, \ldots, \omega_3\}, R(c_1) = \{\omega_4, \ldots, \omega_7\}$

$c_2 : \bar{a} : \bar{b} \succ b$ $\quad L(c_2) = \{\omega_{12}, \ldots, \omega_{15}\}, R(c_0) = \{\omega_8, \ldots, \omega_{11}\}$

$c_3 : b : c \succ \bar{c}$ $\quad L(c_3) = \{\omega_0, \omega_1, \omega_8, \omega_9\}, R(c_3) = \{\omega_2, \omega_3, \omega_{10}, \omega_{11}\}$

$c_4 : \bar{b} : \bar{c} \succ c$ $\quad L(c_4) = \{\omega_6, \omega_7, \omega_{14}, \omega_{15}\}, R(c_4) = \{\omega_4, \omega_5, \omega_{12}, \omega_{13}\}$

$c_5 : bc : d \succ \bar{d}$ $\quad L(c_5) = \{\omega_0, \omega_8\}, R(c_5) = \{\omega_1, \omega_9\}$

$c_6 : b\bar{c} : \bar{d} \succ d$ $\quad L(c_6) = \{\omega_3, \omega_{11}\}, R(c_6) = \{\omega_2, \omega_{10}\}$

$c_7 : \bar{b}c : \bar{d} \succ d$ $\quad L(c_0) = \{\omega_5, \omega_{13}\}, R(c_7) = \{\omega_4, \omega_{12}\}$

$c_8 : \bar{b}\bar{c} : d \succ \bar{d}$ $\quad L(c_8) = \{\omega_6, \omega_{14}\}, R(c_8) = \{\omega_7, \omega_{15}\}.$

Following steps of Algorithm 4.3, the first partition set is $E_\Pi[0] = \{\omega_0\}$. Constraints c_0, c_1, c_3 and c_5 are deleted. For the second iteration, $E_\Pi[1] = \{\omega_0, \omega_3, \omega_6, \omega_{14}\}$. The remaining constraints c_2, c_4, c_6, c_7 and c_8 impose no restriction on ω_1. Therefore this configuration is assigned to $E'_\Pi[1]$. After the second iteration of the Algorithm 4.3, constraints c_2, c_4,

Using an optimistic or pessimistic mindset, the previous example shows the possibility of obtaining a well-ordered partition on configurations that is in full agreement with the Pareto strategy. In this case, the new partitioning procedure succeeds in repairing all the contradictions induced by the ordering obtained from the algorithms 4.1 and 4.2. However, in some cases, the algorithm may fail to repair some strong Pareto violations as shown by Example 4.9.

E_Δ E'_Δ

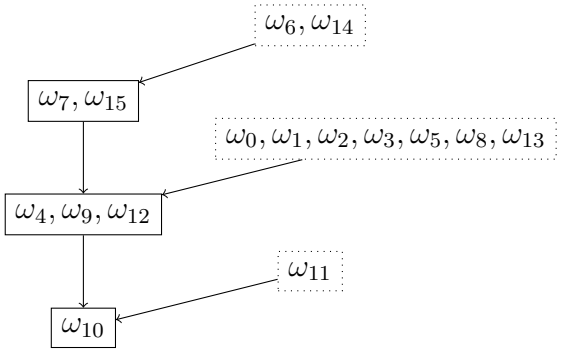
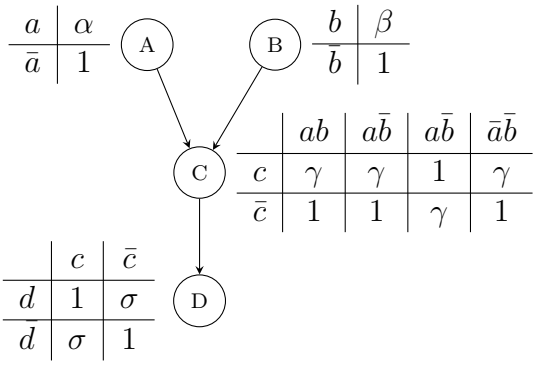


Figure 4.13: Improved pessimistic ordering of π-pref net in Figure 4.1



a	α
$\bar{a}$	1

b	β
$\bar{b}$	1

	ab	$a\bar{b}$	$\bar{a}b$	$\bar{a}\bar{b}$
c	γ	γ	1	γ
$\bar{c}$	1	1	γ	1

	c	$\bar{c}$
d	1	σ
$\bar{d}$	σ	1

Figure 4.14: Example of a π-pref net with one symbolic degree per variable

Example 4.9 *Figure 4.14 depicts an example of a π-pref net that encodes the following set of preference specifications: $\bar{a} \succ a$, $\bar{b} \succ b$, $ab : \bar{c} \succ c$, $a\bar{b} : \bar{c} \succ c$, $a\bar{b} : c \succ \bar{c}$, $\bar{a}\bar{b} : \bar{c} \succ c$, $c : d \succ \bar{d}$ and $\bar{c} : \bar{d} \succ d$. Quality vectors of configurations are given in Table 4.4. Preference specifications written under defaults are translated into the following constraints:*

$c_0 : \bar{a} \succ a$ $\quad L(c_0) = \{\omega_8, \ldots \omega_{15}\}, R(c_0) = \{\omega_0, \ldots \omega_7\}$

$c_1 : \bar{b} \succ b$ $\quad L(c_1) = \{\omega_4, \ldots \omega_7, \omega_{12}, \ldots \omega_{15}\}, R(c_1) = \{\omega_0, \ldots \omega_3, \omega_8, \ldots \omega_{11}\}$

$c_2 : ab : \bar{c} \succ c$ $\quad L(c_2) = \{\omega_2, \omega_3\}, R(c_0) = \{\omega_0, \omega_1\}$

$c_3 : ab : \bar{c} \succ c$ $\quad L(c_3) = \{\omega_6, \omega_7\}, R(c_3) = \{\omega_4, \omega_5\}$

$c_4 : \bar{a}b : c \succ \bar{c}$ $\quad L(c_4) = \{\omega_8, \omega_9\}, R(c_4) = \{\omega_{10}, \omega_{11}\}$

$c_5 : \bar{a}\bar{b} : \bar{c} \succ c$ $\quad L(c_5) = \{\omega_{14}, \omega_{15}\}, R(c_5) = \{\omega_{12}, \omega_{13}\}$

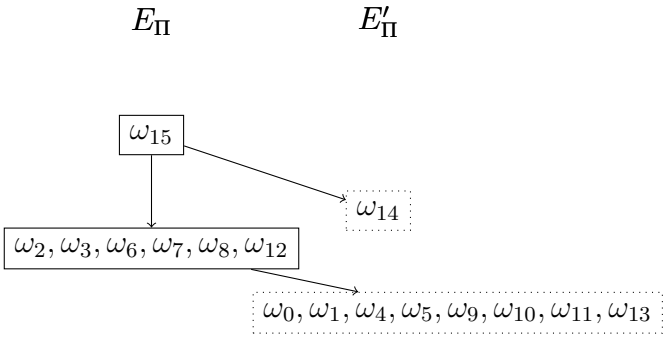
$c_6 : c : d \succ \bar{d}$ $\quad L(c_6) = \{\omega_0, \omega_4, \omega_8, \omega_{12}\}, R(c_6) = \{\omega_1, \omega_5, \omega_9, \omega_{13}\}$

$c_7 : \bar{c} : \bar{d} \succ d$ $\quad L(c_8) = \{\omega_3, \omega_7, \omega_{11}, \omega_{15}\}, R(c_8) = \{\omega_2, \omega_6, \omega_{10}, \omega_{14}\}.$

The optimistic ordering E_Π induced from Algorithm 4.1 is:

E_{Π_0}	$\{\omega_{15}\}$
E_{Π_1}	$\{\omega_2, \omega_3, \omega_6, \omega_7, \omega_8, \omega_{12}, \omega_{14}\}$
E_{Π_2}	$\{\omega_0, \omega_1, \omega_4, \omega_5, \omega_9, \omega_{10}, \omega_{11}, \omega_{13}\}$

The optimistic ordering induced from the improved Algorithm 4.3 is represented by the following figure:



The default and Pareto strategies contradict each other on the preference relation between configurations ω_2 and ω_9. Indeed, the Pareto order stipulates that $\omega_9 \succ_{Pareto} \omega_2$ since $\vec{\omega_9} = (1, \beta, 1, \sigma)$ and $\vec{\omega_2} = (\alpha, \beta, 1, \sigma)$, while both the classical and improved version of the partitioning procedure state that $\omega_2 \succ_{Opt} \omega_9$.

This improved procedure showed the possibility of obtaining a partial pre-order that generally does not contradict the Pareto order (proofs on the topic represent forthcoming studies). In the next section, we show that there exist sets of conditional preferences that can be only handled by a default-like approach [Ben Amor et al., 2019].

4.6 From default preference rules to conditional preference networks

While conditional preference graphs can be turned into default preference bases, we consider the reverse transformation, i.e., whether from any preference rule base, a network of conditional constraints can be generated. We show that this is generally not the case. Preference networks lead to very specific default preference statements. Contexts are always conjunctions of literals, which makes it possible the construction of corresponding conditional data tables. But general preference statements admit more general forms of contexts. Moreover preferences in networks are local in the sense that they deal with values of single variables only. Finally, information in a preference base can be insufficient to build a conditional preference graph as shown by counter-example 4.10.

Ω		$\vec{\omega}$
ω_0	$abcd$	$(\alpha, \beta, \gamma, 1)$
ω_1	$abc\bar{d}$	$(\alpha, \beta, \gamma, \sigma)$
ω_2	$ab\bar{c}d$	$(\alpha, \beta, 1, \sigma)$
ω_3	$ab\bar{c}\bar{d}$	$(\alpha, \beta, 1, 1)$
ω_4	$a\bar{b}cd$	$(\alpha, 1, \gamma, 1)$
ω_5	$a\bar{b}c\bar{d}$	$(\alpha, 1, \gamma, \sigma)$
ω_6	$a\bar{b}\bar{c}d$	$(\alpha, 1, 1, \sigma)$
ω_7	$a\bar{b}\bar{c}\bar{d}$	$(\alpha, 1, 1, 1)$
ω_8	$\bar{a}bcd$	$(1, \beta, 1, 1)$
ω_9	$\bar{a}bc\bar{d}$	$(1, \beta, 1, \sigma)$
ω_{10}	$\bar{a}b\bar{c}d$	$(1, \beta, \gamma, \sigma)$
ω_{11}	$\bar{a}b\bar{c}\bar{d}$	$(1, \beta, \gamma, 1)$
ω_{12}	$\bar{a}\bar{b}cd$	$(1, 1, \gamma, 1)$
ω_{13}	$\bar{a}\bar{b}c\bar{d}$	$(1, 1, \gamma, \sigma)$
ω_{14}	$\bar{a}\bar{b}\bar{c}d$	$(1, 1, 1, \sigma)$
ω_{15}	$\bar{a}\bar{b}\bar{c}\bar{d}$	$(1, 1, 1, 1)$

Table 4.4: Vectors associated with configurations of the π-pref net of Figure 4.14

Example 4.10 *Let us consider the counterpart of the well-known "penguin" example in non-monotonic reasoning [Benferhat et al., 1992][Kraus et al., 1990]. Let c, r and s now stand for "Chicken (C)", "Red wine (R)" and "Spicy plate (S)". Preference rules are {"With chicken, I prefer red wine", "If spicy, I prefer white wine" and "If spicy, I prefer chicken "}, where "white wine" is the negation of "red wine". It corresponds to constraints* $cr > c\bar{r}$, $s\bar{r} > sr$, $sc > s\bar{c}$ *using the minimum specificity principle.*

It is well-known that in this example, we get a well-ordered 3-partition with $E_0 = \bar{s} \wedge (\bar{c} \vee r)$, $E_1 = c \wedge \bar{r}$ *and* $E_2 = s \wedge (r \vee \bar{c})$ *[Benferhat et al., 1992][Ben Amor et al., 2019].*

The rules indicate that values of C and R depend on S and R depend on C, hence the graph of Figure 4.15. However some information is missing to get a full preference graph:

- *The absolute preference between s and* $\bar{s}$ *on node S (represented by* $s?\bar{s}$*).*
- *The preference for chicken or not when the plate is not spicy is not given (represented by* $\bar{s} : c?\bar{c}$*).*
- *The preference about wine when the dish is not chicken nor spicy (represented by* $\bar{s}\bar{c} : f?\bar{f}$*).*

- *The preference about wine when the chicken is spicy. From the given rules, this is a conflicting case represented by a double question mark in* $sc : r??\bar{r}$*. In fact,* S *and* C *act as independent parents of* R*, which causes the conflict.*

It is forbidden in a preference graph for a variable to have several parent groups. The conflict between S *and* C *is solved when applying minimum specificity ranking to the default rules (we conclude that* $\bar{s} > s$*, that* $\bar{s}c > \bar{s}\bar{c}$*,* $sc\bar{r} > scr$ *and no preference between* $\bar{s}\bar{c}r$ *and* $\bar{s}\bar{c}\bar{r}$*.*

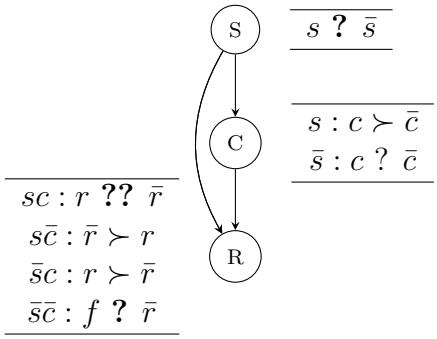


Figure 4.15: Partial network from preference rules

Clearly there is a gap between general default rule-like preferences and conditional preference networks. However, once we have computed the complete pre-order associated to the well-ordered partition of the configurations obtained by means of the minimum specificity principle for instance, we can obviously generate a conditional preference network from it (since this would solves the question marks in the previous example, for instance).

4.7 Experimental study

In this section, we propose to conduct an experimental study to support the propositions made in 4.4, to evaluate the expressiveness of the partitioning algorithms of Sections 4.3.1, 4.3.2 and 4.5, to finally compare their results with those of the Pareto ordering.

4.7.1 Experimental protocol

The first step of the experimental protocol is to generate a collection of conditional preference networks. To this end, we use the generation tool *GenCPnet* developed

by Thomas E. Allen et al. [Allen et al., 2016]. This generator produces uniformly random connected acyclic networks with specified set of constraints. It allows to vary different parameters, namely:

- the number of nodes n that we vary from $n = 3$ to $n = 7$;
- the maximum bound on the number of in-going edges also-called *in-degree* c that we vary from 1 to $n - 1$;
- the size of variable's domain d that we fix to $d = 2$.

Enumerating the number of acyclic DAGs (not necessarily connected) on n nodes with labels in $[1, \ldots, n]$ have been studied in [Robinson, 1977]. This number is calculated by the recurrence expressed by the following Equation 4.5:

$$a_n = \sum_{k=1}^{n} (-1)^{k+1} C_n^k 2^{k(n-k)} a_{n-k}. \tag{4.5}$$

For instance, for $n = 2$ we can generate $a_2 = 3$ different DAGs, for $n = 3$ we can generate $a_3 = 25$ different DAGs and for $n = 4$ we can generate $a_4 = 543$ different DAGs. Even with small numbers of nodes, we can see that the number of possible graphs increases exponentially as n grows, to reach 1.138.779.256 graphs for $n = 7$. This number grows even more if we consider all possible combinations of preference specifications.

The experiments are divided into two parts. We first start by computing the optimistic and pessimistic orderings to confirm propositions made in Section 4.4. To do so, we generated a first benchmark of 1000 instances of preference networks only composed of quasi-linear DAGs: 50 instances with $n = 3$, 200 instances with $n = 4$ and 250 instances for each of $n = 5$, $n = 6$ and $n = 7$. The purpose of the first experiment part is to confirm the proposition that for quasi-linear networks (see Figure 4.11), the number of partition layers of the optimistic and pessimistic orderings equals 3.

The second part of the experiments is first dedicated to study the behaviour of the size of default partitions with regards to the graphical structures. We then study the behaviour of the percentage of the Pareto and default orderings contradictions with regards to the number of nodes, the existence or not of links between grand-parents and child nodes and the maximum in-degree of graphs.

For this purpose, we generated a second benchmark with a total of 3600 networks, where for each (n, d) with $n = [3, \ldots, 7]$, we vary the value of c starting from $c = 2$. For $n = 3$, we generated 100 conditional preference networks, 500 networks for $n = 4$

and 1000 networks for each of $n = 5$, $n = 6$ and $n = 7$. Descriptions about this data set are given in Tables 4.5 and 4.6. This benchmark is composed of a quarter of quasi-linear graphs and 3/4 of graphs with a maximum in-degree equal to 2. If we divide the data set into subsets according to the size of the graphs, the Table 4.5 gives the percentage of graphs that contain grand-parent - children links compared to those that do not. For instance, for $n = 4$, 69.8% of graphs contain at least one arc going from a node to a grand-son, while 30.2% contain none. The Table 4.6 describes the benchmark in terms of the maximum in-degree. For example, the set of networks with $n = 3$ are composed of 43% of graphs with $c = 1$ and 57% of graphs with $c = 2$.

Nb nodes \ **Graph structure**	**Other**	**Quasi-linear**
3	36	64
4	69.8	30.2
5	82.5	17.5
6	92	8
7	95	5
Total	75	25

Table 4.5: The composition of the second benchmark in percentage

Max in-degree \ **Nb nodes**	3	4	5	6	7
1	43	22.2	14.9	7.5	5
2	57	41.4	25.2	12.5	12.5
3	0	36.4	30	20	17.5
4	0	0	29.9	25	25
5	0	0	0	35	15
6	0	0	0	0	25

Table 4.6: Percentage of graphs given their maximum in-degree for a fixed number of nodes in the second benchmark

For both benchmarks, the set of preference statements of each network is translated into default rules and then to default constraints as explained in 4.3.1 and 4.3.2. Therefore, for the sets of constraints, we apply Algorithms 4.1 and 4.2 to find respectively the optimistic and pessimistic orderings on the set of configurations. In a first step, the behaviour of the partition sizes is studied. In a second step, the orderings are compared with the Pareto ordering. Finally, the improved partitioning procedure explained in 4.5 is performed to confirm the accordance of its induced ordering with the Pareto order. For all these experiments, we consider the Pareto order assuming one symbolic degree per variable.

The goals of the second experiment part are summarized in the following:

(i) check if there exists a correlation between the number of layers of the default well-ordered partition and the following parameters:

- the number of nodes;
- the existence or not of grand-children links;
- the graph's maximum in-degree;

(ii) (Part 2) compare the expressive power of the Pareto ordering compared with the default orderings;

(iii) check if there exists a correlation between the percentage of strict dominance relations induced from the default and Pareto orderings and the following parameters:

- the number of nodes;
- the graph's maximum in-degree;

(iv) check if there exists a correlation between the percentage of strong violations between the Pareto and the default orderings given one of the following parameters:

- the number of nodes;
- the existence or not of grand-children links;
- the graph's maximum in-degree;

(v) confirm that, for most networks, the ordering induced from the improved default partitioning procedure can totally or partially repair contradictions with the Pareto order.

These experiments are conducted on an Intel Core $i7-7700HQ$ processor and 20Go workstation. The software tool and functions used for the experiments are programmed in JavaScript language.

4.7.2 Experimental results

In this subsection, we give the results of our experiments. The conclusions in this section are drawn from the experiments conducted on the data sets described above. The first subsection gives results on the size of the ordering induced by the partitioning Algorithm 4.1 (and 4.2). The second subsection describes results on the expressiveness

of the Pareto and default orders. The third subsection gives results on the improved partitioning procedure detailed in 4.5. Finally, the last subsection summarizes the results of all experiments.

Results on the well-ordered partition size

The first part of the experiment carried out on the first benchmark had confirmed the propositions of Section 4.4. Indeed, we found that the optimistic and pessimistic orderings of all the networks of the first benchmark are composed of exactly 3 levels. Moreover, no contradictions with the Pareto ordering were detected, which would suggest that these violations are perhaps related to the in-degree of graphs and/or to the presence of links between parent and grand-children. Besides, based on the given experiment results, the assumption that there is any correlation between the strong violations between the default and Pareto orderings and the number of nodes in the graph can be discarded for quasi-linear graphs (still needs to be proven in future research).

We now consider general graphs with grand-parents - children relations (75% of the second benchmark set) and study the behaviour of the default ordering partition size. Each set of networks with the same number of nodes results in partitions of different sizes. For each set, the Table 4.7 gives the percentage of graphs according to the size of their induced well-ordered partitions. Mind that for all of our experimental results, we note that for all preference networks, the sizes of the well-ordered partitions of the optimistic and pessimistic orderings are equal, meaning that all of the results described in this subsection are valid for both approaches. From the results described in 4.7, we can confirm that the sizes of the partitions increase with the number of nodes that compose the graphs. For instance, for $n = 4$, we have partitions with a number of layers going from 3 to 5, while for $n = 5$ we can in addition have partitions with 6 layers. Generally and based on these results, for networks of size N, default partition sizes can go up to $N + 1$ levels. However, note that for networks with $n = 3$, all induced partitions are of size 4. By comparing with the results of the first part of the experiment, we can confirm that there is a link between the size of the partition and the arcs between the generations of nodes. For future research, it is interesting to study the impact of the depth of arcs between nodes and grand-child generations on the number of levels in the partition.

Nb layers \ Nb nodes	3	4	5	6	7
2	0	0	0	0	0
3	0	6.3	3.3	2.1	1.6
4	100	69.9	43.9	21.1	20.1
5	0	23.8	37.8	27.9	24.2
6	0	0	15	32.4	21.7
7	0	0	0	16.5	21.5
8	0	0	0	0	10.9

Table 4.7: Variation of the percentage of graphs according to the size of their induced default partitions

The following experimental results aim to determine whether there is a correlation between the default partition size and the maximum degree of preference networks. For fixed graph sizes, we varied c and calculated the percentage of network instances with respect to different partition sizes. The results are given in Figure 4.16.

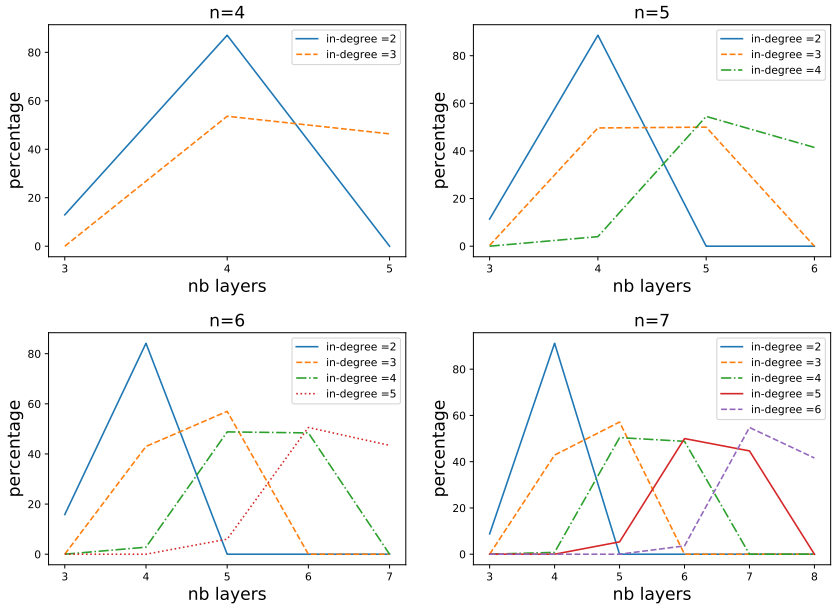


Figure 4.16: Percentage of networks by size of partitions for n from 4 to 7

The results show that generally there is a stron relation between the partition sizes and the network in-degrees. However, except for the graphs with $c = 2$, we can detect a certain tendency in the behavior of curves with higher in-degrees. In fact, for a fixed number of nodes n, we can notice that most of the partitions have sizes which vary from $c+1$ to $c+2$. It is the case of the curve of the figure relative to $n = 6$ where, for $c = 3$ the number of levels varies between 4 and 5, for $c = 4$ the number of levels varies between 5 and 6, for $c = 5$ the number of levels varies between 6 and 7. Nevertheless, this cannot be generalized for all partitions. Indeed, for instance for $n = 6$ and $c = 5$,

there are some partitions with a size equal to c.

In the same spirit, we want to determine if there is a correlation between the size of the default partition and the number of nodes in the preference networks. For fixed in-degrees, we varied the size of the networks and calculated the percentage of instances of these networks with respect to different partition sizes. Figure 4.17 describes the results of this experimentation.

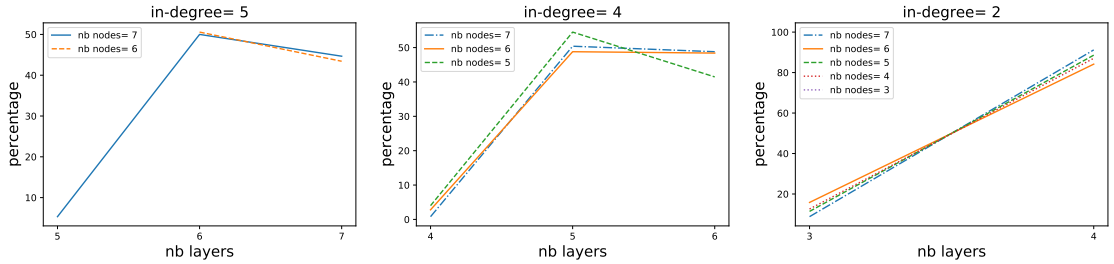


Figure 4.17: Percentage of networks by size of partitions for $c = 2$ $c = 4$ and $c = 5$

These results confirm the assertion that there is correlation between the number of partition layers and the size of the preference network. This claim only holds for the networks of the second benchmark, since in the first part of the experiments, we have seen that, whatever the size of the graph and for a fixed in-degree equal to 1, all networks lead to default orderings of size 3.

Results on Pareto and default orders

To evaluate the expressive powers of the Pareto and default orders, we computed for each subset of graphs having the same size, the percentage of strict comparisons out of total comparisons. Figure 4.18 represents results of this experiment. We notice that for the default ordering, the number of nodes has almost no effect on the percentage of comparisons. However, the Pareto order is strongly influenced by the network size, since the percentage decreases as the size increases, which is represented in the graph by a decreasing curve. We recall that the experiments in this subsection are performed on all instances of the second benchmark.

Figure 4.19 shows the percentages of strict dominance relations induced by the two approaches as a function of the in-degree of different graphs. The figure describes divergent curves. Indeed, as the number of in-degree increases, the percentage of default order comparisons increases and the percentage of Pareto order comparisons decreases. This shows that, contrary to the Pareto order on graph instances with a high number of nodes, the default approach manages to find strict dominance relations between a larger number of configuration pairs than the other approach.

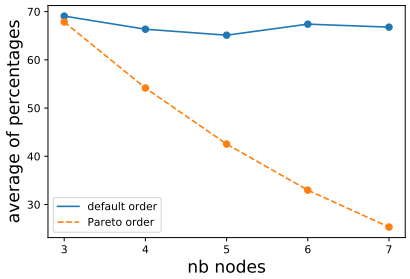


Figure 4.18: Average of the percentage of strict Pareto and default orders dominance relations as a function of the network size

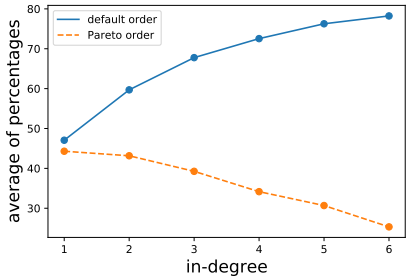


Figure 4.19: Average of the percentage of strict Pareto and default orders dominance relations as a function of the network in-degree

Of the total number of networks in the data set, 23.97% of the instances lead to default orderings that contradict the Pareto order. On this set of networks, we first try to find out if there is a relationship between the contradictions and the existence or not of arcs that connect the nodes to the grand-children. Indeed, experiment results have shown that, on the set of networks with Pareto contradictions, almost all instances (99.08%) with grand-parent - children links hold strong violations with the Pareto order.

We now move to explore the relationship between the contradictions and the size of the graphs. To do this, we fix the in-degrees of the graphs and observe the behavior of the percentage of networks that violate the Pareto order as a function of the number of nodes. Results in Figure 4.20 do not reveal a real tendency. For the curves with in-degrees equal to 3 and 4, we can presume the existence of a tendency described by their two respective curves. Indeed, we can see that the more the number of nodes increases the more the percentage increases. Nevertheless, this tendency is not respected by the curves of in-degrees 2 and 5. For example for in-degree= 5, when the number of nodes increases from 6 to 7 the percentage of contradictions decreases from about 45 to 0.8 percent. For in-degree= 2, no trend can be discerned by the respective curve. Note that for an in-degree equal to 6, we only have networks with a fixed number of nodes equal to 7, only 0.8% of these networks induce a default ordering that contradicts the

Pareto order.

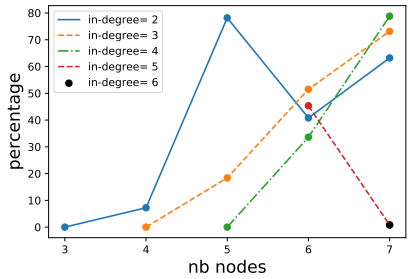


Figure 4.20: Percentage of networks with Pareto contradictions as a function of the graph size

We now fix the number of nodes and vary the in-degree of the graphs to study the relationship between the contradictions and the in-degrees. Figure 4.21 presents the percentage of networks with Pareto violations as a function of the in-degree of the graph. All curves have different inflection points showing the presence of a relation between the in-degree of the networks and the number of Pareto contradictions.

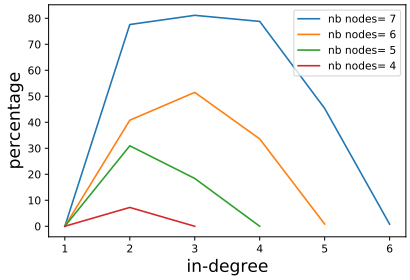


Figure 4.21: Percentage of networks with Pareto contradictions as a function of the graph in-degree

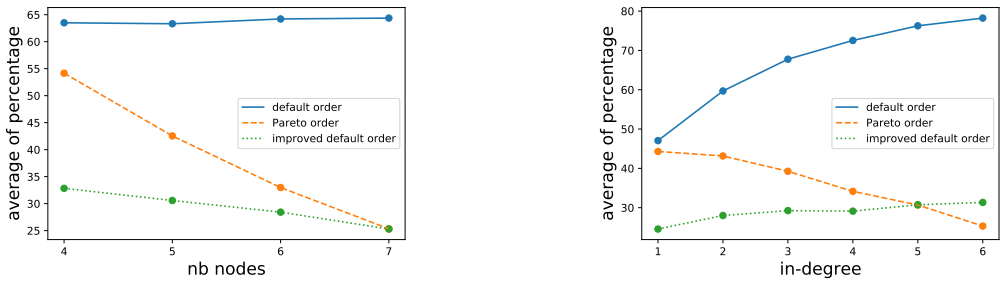
Results on improved partitioning procedure

Experiments on the improved partitioning procedure showed that for the majority of instances that lead to strong violations with the Pareto order (i.e., on 23.97% of the second benchmark), the new algorithm was able to fully or partially repair violations with the Pareto order. In our benchmark, 62.93% of orderings have been partially or totally improved. In some networks, no improvement is reported and the percentage of contradictions remains constant (see Example 4.9). No deterioration or increase in the percentage of violations was detected. Results of the experiment are reported in the Table 4.8.

Observation	Percentage
The order is totally repaired	19.71%
The order is partially repaired	43.22%
No repair	37.07%

Table 4.8: Experiment results of the improved partitioning procedure

Figures 4.22 (a) and (b) give the average of the percentage of strict comparisons induced from the Pareto, the default partitioning algorithm and its improved version in function of the network sizes and in-degrees respectively. In both figures, we notice that the classical partitioning procedure and the improved partitioning procedure behave in the same way: whatever the size of the graph considered, the percentage of strict comparisons is practically the same. We even notice that the number of nodes and the in-degree have an influence on the improvement. More precisely, for $n = 7$ or $c = 5$, the improved procedure allows to have a more discriminating order than the Pareto one. Nevertheless, this order is not necessarily free of contradictions with the Pareto strategy. The same behavior can be observed for all the graphs having contradictions with the Pareto order.



(a) (b)

Figure 4.22: Average of the percentage of strict order dominance relations as a function of (a) the network sizes and (b) the network in-degrees for all the benchmark

Discussion of the experiment results

Using the default partitioning procedure, quasi-linear DAGs, described without arcs between grand generations of nodes, lead to total pre-orders described on exactly 3 levels. From our experiments, these default partitions were found to be free of Pareto contradictions. We have seen that on other DAG structures, the partition size is at least equal to 4 which means that there is a relationship between the existence of links between nodes and grand-children and the number of layers in the partition. Moreover, no relationship between the default ordering size and the graph in-degrees have been found. However, experiments have shown that generally, the lower the in-degree of the

network, the larger the partition size. Deeper experiments to study the impact of the number of arcs between generations and the size of those links should be considered as research perspectives on the topic.

Concerning the percentage of strict dominance relations between pairs of configurations, the experiments showed that the default ordering procedure behaves in the same way and orders practically around 65% of the possible comparisons, whatever the size of the graph, which is not the case for the Pareto order. Indeed, the larger the number of nodes, the lower the percentage of comparisons induced by the Pareto order. Moreover, the two orders are sensitive to the variation of the in-degree and behave in divergent ways. In fact, the higher the in-degree, the more the percentage of default comparisons increases and the more the percentage of Pareto comparisons decreases.

The default partitioning procedure can lead to contradictions with the Pareto order. The networks that describe this discrepancy contain in most cases links between nodes and grand-children. The improved partitioning algorithm version that we propose generally allows to completely or partially reduce the number of these contradictions. However, experiments on our benchmark have shown that the order induced by the new procedure leads to a less discriminating order than the Pareto strategy for small DAG in-degrees and sizes but seems to overcome this drawback for larger numbers. More experiments on higher number of nodes ans higher network in-degrees should conducted for future perspectives.

4.8 Conclusion

In this chapter, we have represented a set of conditional preferences by means of a collection of default-like rules encoded in terms of inequality constraints between possibilistic set functions applied to sets of configurations. Then we have shown that using a minimum or maximum specificity principle we can obtain a complete pre-order on configurations. Still this pre-order have some drawbacks: (i) it has a limited discrimination power; (ii) it may partially conflict with the Pareto order. However, in the general case, it is possible to remedy to the problem, by adapting the partitioning algorithm proposed in [Benferhat et al., 2001a] that we apply to the constraints representing preferences, by only handling the exact information given by the constraints. Generally, this results in a partial order with no or fewer Pareto violations. However, we have shown that some sets of default rules cannot be represented by a π-pref net structure; this may be explained by the rigorous rules of representation imposed by

the DAG structure, restricting the types of possible dependencies between variables.

In the next chapter, we propose solutions in order to obtain a complete pre-orders without Pareto violations while being more discriminant than the ones obtained in this chapter.

Chapter 5

Modifying Configuration Orderings in Agreement with Pareto Dominance

5.1 Introduction

Conditional statements in a preference graph may be translated into default-like rules which may be represented by different inequality constraints involving the minimum or the maximum of satisfaction degrees for two mutually exclusive situations. In the previous chapter, we have seen how to translate these rules into a set of preference or satisfaction constraints on which an information principle from possibility theory is applied to entail a well-ordered partition of configurations. When the default rules and the Pareto order are considered, they lead to different ordering of the solutions and may contradict each other. In order to take advantage of both approaches and to remedy the mentioned discrepancy, we propose to study the repair of the default order by the Pareto order. Besides, since the default order gives a total pre-order on the configurations, we then seek to refine the Pareto order with the relations of the default rule.

The work reported in this chapter departs from the approaches dealt with in Chapter 4 by the particular role played by Pareto order for repairing or refining the complete pre-orders obtained in Chapter 4.

This chapter is divided in four main sections corresponding to three approaches that all lead to complete pre-orders. In the following Section 5.2, we correct the ordering obtained with the minimum or maximum specificity principles in order to satisfy the Pareto order. Section 5.3 then proposes to build a complete pre-order from

both optimistic and pessimistic rankings that appears to be the most discriminant compared to the other orderings discussed in this thesis work; however, this can be applied only when these rankings do not contradict Pareto order (which is generally the case in practice). The before last Section 5.4 starts with the Pareto ordering and refines it by using the minimum or maximum specificity orderings. Finally, an experimental study that supports our work is reported in Section 5.5. Section 5.4 rely on paper [Ben Amor et al., 2021a]. The work reported in all other sections has not been published yet.

The same running example is used in the three main sections.

Moreover, in all the procedures describes in this chapter, we refer to a Pareto order. This means that we compute it by comparing quality vectors. These vectors are made of symbolic weights which are assigned just as in the conditional preference tables of a π-pref net. Still when dealing with the optimistic ordering, the symbolic weights are of the form 1 and $\alpha, \beta, \ldots$, while for the pessimistic ordering we use the weights $\alpha, \beta, \ldots$, and 0 (in reference to the different encodings discussed in Chapter 3).

5.2 Repairing optimistic or pessimistic orderings with Pareto order

In some cases, default-based orderings may exhibit strong violations with the Pareto order. In this section, we propose to correct the optimistic and pessimistic orderings with the Pareto order with the purpose of repairing these conflicts.

Given a well-ordered partition E (optimistic E_Π or pessimistic E_Δ), how can we possibly correct it by the Pareto order? By definition, the partition E is a set of k ordered layers, thus the repair process should be performed in two steps; the first is an intra-layer refinement consisting of pairwise comparisons within all configurations of a given layer, and the second step is an inter-layer refinement consisting of pairwise comparisons within configurations in a given layer ℓ and those of the lower layers $\ell + 1, \ldots, k$.

In the intra-layer refinement, for each pair of configurations (ω, ω'), we check whether the Pareto ordering gives additional dominance information (i.e., $\omega \succ_{Pareto} \omega'$ or $\omega' \succ_{Pareto} \omega$). If so, an intermediate level (between l and $l + 1$) is created and the less preferred configuration is put into it.

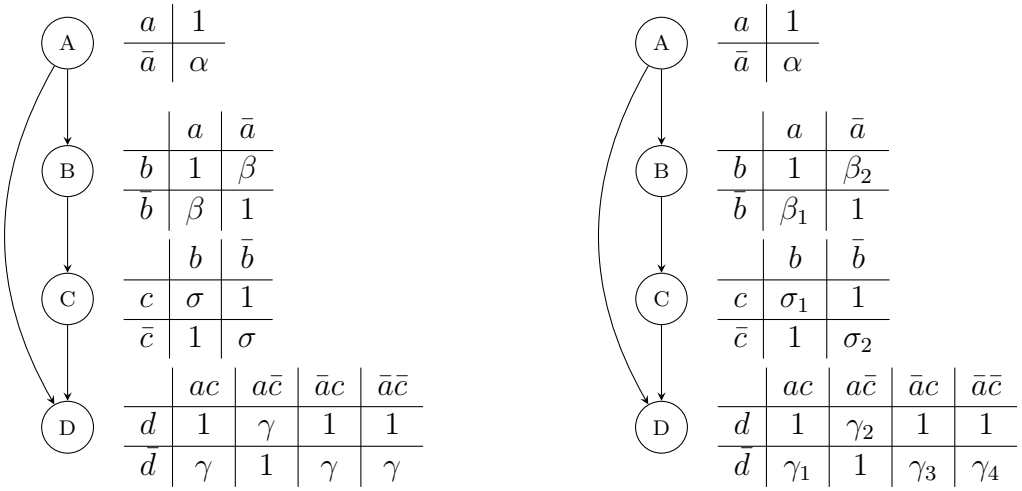
In the same vein, this principle is applied for the inter-layer refinement where

we check that, according to the Pareto order, any configuration ω in layer l of E is preferred to all configurations ω' at a lower level of E. If not the case, ω is moved down. If the Pareto ordering indicates an incomparability relation between ω and ω', then the dominance relation indicated by the default order is preserved.

Algorithm 5.1 outlines the explained process. Function $extend(E, i)$ permits to add an intermediary layer below the layer i. The function $Pref(M, \omega, \omega')$ gives the dominance relation between configurations ω_i and ω_j based on a given a matrix M that encodes an order.

Algorithm 5.1 has as input the Pareto incidence matrix M_{Pareto} and a well-ordered partition E such that its layers are sorted from best to worst ones. In the worst case, E is composed of not less than 3 layers (see propositions in Section 4.4) . To simplify the calculation of the algorithm's complexity, we consider that E contains a single layer holding all possible configurations. Thus, one layer is created for each configuration if the Pareto partial ordering is refined into a total order at the end. The time complexity of the refinement algorithm is $O(2^N)$.

In Example 5.1, we illustrate the application of Algorithm 5.1.



(a) (b)

Figure 5.1: Example of π-pref net (a) for equal symbolic degrees per variable and contexts (b) for different symbolic degrees per variable and context values

Example 5.1 *Let us consider π-pref nets in Figure 5.1 that encode the same set of preference specifications, namely $a \succ \bar{a}$, $a : b \succ \bar{b}$, $\bar{a} : \bar{b} \succ b$, $b : \bar{c} \succ c$, $\bar{b} : c \succ \bar{c}$, $ac : d \succ \bar{d}$, $a\bar{c} : \bar{d} \succ d$, $\bar{a}c : d \succ \bar{d}$, $\bar{a}\bar{c} : d \succ \bar{d}$. Vectors of weights associated with configurations are given in Table 5.2. The induced Pareto ordering on configurations is repre-*

Algorithm 5.1: Refining default ordering by Pareto ordering

Input: M_{Pareto}, E s.t. $E = E_{\Pi}$ or $E = E_{\Delta}$
Output: $E_{Refined}$

$E_{Refined} = E$
$i = 0$
while $i < len(E_{Refined})$ **do**
  `/* intra-layer refinement */`
  $extend(E_{Refined}, i)$
  foreach $\omega \in E_{Refined}[i]$ **do**
    foreach $\omega' \in E_{Refined}[i]$ *s.t.* $\omega' \neq \omega$ **do**
      `/* i+1 is the intermediary intra-layer of i */`
      $\succ_{Pareto} = Pref(M_{Pareto}, \omega, \omega')$
      if $\succ_{Pareto} \neq \omega' \bowtie \omega$ **then**
        if $\succ_{Pareto} = \omega' \succ \omega$ **then**
          $move(E_{Refined}, \omega, i, i+1)$
        else if $\succ_{Pareto} = \omega \succ \omega'$ **then**
          $move(E_{Refined}, \omega', i, i+1)$
    end
  end
  $clean(E_{Refined})$
  `/* inter-layer refinement */`
  foreach $\omega \in E_{Refined}[i]$ **do**
    $k = i + 1$
    $extend(E_{Refined}, k)$
    foreach $\omega' \in E_{Refined}[k]$ *s.t.* $\omega' \neq \omega$ **do**
      $\succ_{Pareto} = Pref(M_{Pareto}, \omega, \omega')$
      if $\succ_{Pareto} = \omega' \succ \omega$ **then**
        $move(E_{Refined}, \omega, i, k+1)$
    end
  end
  $clean(E_{Refined})$
  $i = i + 1$
end
Return $E_{Refined}$

sented by the Figure 5.3. The optimal configuration is $\omega_3 = ab\bar{c}\bar{d}$ *and the worst configuration is* $\omega_9 = \bar{a}bc\bar{d}$*. The induced optimistic ordering is* $E_\Pi = \{\{\omega_3\}, \{\omega_0, \omega_2, \omega_4, \omega_{12}\}, \{\omega_1, \omega_5, \omega_6, \omega_7, \omega_8, \omega_9, \omega_{10}, \omega_{13}, \omega_{14}\}, \{\omega_{11}, \omega_{15}\}\}$*. Note that the worst configuration* ω_9 *figures in the before last layer. We use the Algorithm 5.1 to repair this discrepancy. The repaired well-ordered partitions by the Pareto order considering one symbolic weight per preference statement and one weight per variable are respectively given in Table 5.1. Note that the refined orderings* E_Π *are composed of* 6 *layers instead of* 4*.*

$\boldsymbol{\Omega}$		$\vec{\boldsymbol{\omega}}$	$\vec{\boldsymbol{\omega}}$
$\omega_0 =$	$abcd$	$(1, 1, \sigma, 1)$	$(1, 1, \sigma_1, 1)$
$\omega_1 =$	$abc\bar{d}$	$(1, 1, \sigma, \gamma)$	$(1, 1, \sigma_1, \gamma_1)$
$\omega_2 =$	$ab\bar{c}d$	$(1, 1, 1, \gamma)$	$(1, 1, 1, \gamma_2)$
$\omega_3 =$	$ab\bar{c}\bar{d}$	$(1, 1, 1, 1)$	$(1, 1, 1, 1)$
$\omega_4 =$	$a\bar{b}cd$	$(1, \beta, 1, 1)$	$(1, \beta_1, 1, 1)$
$\omega_5 =$	$a\bar{b}c\bar{d}$	$(1, \beta, 1, \gamma)$	$(1, \beta_1, 1, \gamma_1)$
$\omega_6 =$	$a\bar{b}\bar{c}d$	$(1, \beta, \sigma, \gamma)$	$(1, \beta_1, \sigma_2, \gamma_2)$
$\omega_7 =$	$a\bar{b}\bar{c}\bar{d}$	$(1, \beta, \sigma, 1)$	$(1, \beta_1, \sigma_2, 1)$
$\omega_8 =$	$\bar{a}bcd$	$(\alpha, \beta, \sigma, 1)$	$(\alpha, \beta_2, \sigma_1, 1)$
$\omega_9 =$	$\bar{a}bc\bar{d}$	$(\alpha, \beta, \sigma, \gamma)$	$(\alpha, \beta_2, \sigma_1, \gamma_3)$
$\omega_{10} =$	$\bar{a}b\bar{c}d$	$(\alpha, \beta, 1, 1)$	$(\alpha, \beta_2, 1, 1)$
$\omega_{11} =$	$\bar{a}b\bar{c}\bar{d}$	$(\alpha, \beta, 1, \gamma)$	$(\alpha, \beta_2, 1, \gamma_4)$
$\omega_{12} =$	$\bar{a}\bar{b}cd$	$(\alpha, 1, 1, 1)$	$(\alpha, 1, 1, 1)$
$\omega_{13} =$	$\bar{a}\bar{b}c\bar{d}$	$(\alpha, 1, 1, \gamma)$	$(\alpha, 1, 1, \gamma_3)$
$\omega_{14} =$	$\bar{a}\bar{b}\bar{c}d$	$(\alpha, 1, \sigma, 1)$	$(\alpha, 1, \sigma_2, 1)$
$\omega_{15} =$	$\bar{a}\bar{b}\bar{c}\bar{d}$	$(\alpha, 1, \sigma, \gamma)$	$(\alpha, 1, \sigma_2, \gamma_4)$

Figure 5.2: Vectors of weights associated to configurations of π-pref net in Figure 5.1

Repairing optimistic ordering E_Π with Pareto	
Pareto with different symbolic weights in a node	**Pareto with equal symbolic weights in a node**
$\{\omega_3\}$	$\{\omega_3\}$
$\{\omega_0, \omega_2, \omega_4, \omega_{12}\}$	$\{\omega_0, \omega_2, \omega_4, \omega_{12}\}$
$\{\omega_1, \omega_5, \omega_7, \omega_{10}, \omega_{13}, \omega_{14}\}$	$\{\omega_1, \omega_5, \omega_7, \omega_{10}, \omega_{13}, \omega_{14}\}$
$\{\omega_6, \omega_8\}$	$\{\omega_6, \omega_8\}$
$\{\omega_9\}$	$\{\omega_{11}, \omega_{15}\}$
$\{\omega_{11}, \omega_{15}\}$	$\{\omega_9\}$

Table 5.1: Repairing optimistic ordering of π-pref net in Figure 5.1 based on Algorithm 5.1

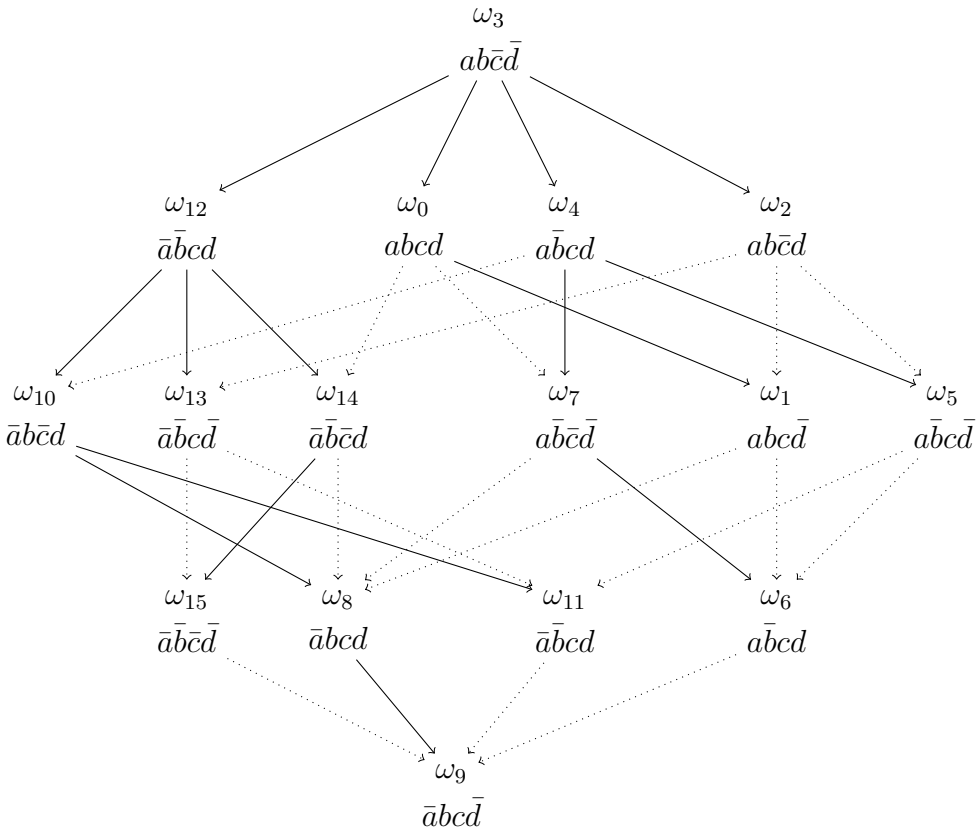


Figure 5.3: Pareto graph of π-pref nets in Figure 5.1 (dotted arrows represent additional comparisons given one symbol per variable)

5.3 Using optimistic and pessimistic approaches jointly

The minimum specificity algorithm outputs a well-ordered partition that clusters the worst configuration(s) with other more preferred ones all in the same set. This is due to the focus on the best models of formulas in the optimistic approach. In some sense, it does not take much in consideration the least preferred models. In order to refine results of the optimistic approach, we can also exploit preference statements based on the maximum specificity principle (pessimistic approach).

Preference graphs that induce a default ordering not consistent with the Pareto ordering are not taken into account, since experimentally, we could see that it is in this case that the optimistic and pessimistic orders may contradict one another.

On the graph structures considered in the Propositions from 4.3 to 4.10 of Chapter 4, it has been proved that the minimum and maximum specificity well-ordered partitions have three layers. In the pessimistic reasoning, the best solution is left on a par with the less preferred ones, while for the optimistic reasoning the worst configuration is left on a par with the more satisfactory ones. It is tempting to consider the

conjunction of the minimum and the maximum specificity rankings.

Algorithm 5.2 takes as input the set of possible solutions, the optimistic and the pessimistic orderings and outputs a well ordered partition, say E^*, consistent with both input orders. The first step consists on creating a refined partition composed of 2^N layers since, in the best case, well-ordered partitions yield a total order where each configuration is put in a different layer. The first and last layers of the output partition are associated with the best and worst configurations respectively. The second layer is then associated with the second layer of E_Π. Configurations of this layer are compared to those of the first layer of E_Δ. If they match, then there exists no possible refinement. Otherwise, configurations in the second layer $E^*[1]$ are put in a lower level since E_Δ expresses so This process is similarly considered to rank-order partition sets of E_Δ. Configurations in the penultimate layer of E_Δ are put in the last but one layer of E^*. These configurations are compared with the second layer of E^* and identical configurations found in the two layers are raised to a higher level in E^*. At the end of each iteration configurations in E^* are deleted from Ω. The procedure is repeated until all configurations are ranked. Function $clean(E)$ removes empty layers from E. In the worst case, the refinement algorithm yields a well ordered partition with a number of layer equal to the maximum between $|E_\Pi|$ and $|E_\Delta|$.

Mind that in case we start by considering E_Δ that we refine by E_Π thus building $E_{\Delta\Pi}$ we have to exchange instructions from line 12 to 24 with instructions from line 25 to 39. This does not affect the algorithm's output, and means that considering the pessimistic order instead of the optimistic order at first place have no impact on the result.

The Example 5.2 illustrates the procedure of Algorithm 5.2. As it can be seen, we obtain a more refined pre-order since we have now 5 layers instead of 3. This suggests the interest of this procedure.

Example 5.2 *Let us consider again the π-pref net in Figure 4.10. Details on results of applying both the minimum and the maximum specificity approaches are presented in Example 4.7. The optimistic well-ordered partition corresponds to* $E_\Pi = \{\{\omega_0\}, \{\omega_1, \omega_2, \omega_4, \omega_7\}, \{\omega_3, \omega_5, \omega_6\}\}$ *and the pessimistic one corresponds to* $E_\Delta = \{\{\omega_0, \omega_2, \omega_4\}, \{\omega_1, \omega_3, \omega_5, \omega_7\}, \{\omega_6\}\}$. *The refined partition E^* is associated with* $2^3 = 8$ *empty layers. The optimal and worst configurations are classified such that* $E^*[0] = \{\omega_0\}$ *and* $E^*[7] = \{\omega_6\}$. *The layer $E^*[1]$ is then composed of configurations in $E_\Pi[1]$ namely* $E^*[1] = \{\omega_1, \omega_2, \omega_4, \omega_7\}$ *from which are deleted* $E^*[1] \cap E_\Delta[0] = \{\omega_1, \omega_7\}$ *since they are considered by the pessimistic ordering as less satisfactory than ω_2 and ω_4. The latter are then associated with $E^*[2]$. At the end of the series of instructions*

Algorithm 5.2: Ordering Ω using both optimistic and pessimistic orders

Input: Ω , E_Π, E_Δ
Output: E^*

$len(E^*) = 2^N$
$E^*[0] = E_\Pi[0]$
$E^*[2^N - 1] = E_\Delta[len(E_\Delta - 1)]$
$i = 0$
$j = len(E_\Pi) - 2$
$index\_up = 1$
$index\_low = 2^N - 2$
$index\_opt = len(E_\Pi) - 2$
$index\_pess = 0$
while $\Omega \neq \emptyset$ **and** $i \leq len(E_\Pi)$ **and** $j \leq len(E_\Delta)$ **do**
    $E^*[index\_up] = E_\Pi[i]$
    $increment = 0$
    foreach $\omega \in E^*[index\_up]$ **do**
        if $\omega \notin E_\Delta[index\_pess]$ **and** $\omega \in \Omega$ **then**
            Add ω to $E^*[index\_up + 1]$
            $increment = 1$
            Delete ω from $E^*[index\_up]$
        Delete ω from Ω
    end
    $i = i + 1$
    $index\_up = index\_up + increment + 1$
    $index\_pess = index\_pess + 1$
    if $i \neq j$ **or** $index\_opt \neq index\_pess$ **then**
        $E^*[index\_low] = E_\Delta[j]$
        $increment = 0$
        foreach $\omega \in E^*[index\_low]$ **do**
            if $\omega \in E_\Pi[index\_opt]$ **and** $\omega \in \Omega$ **then**
                Add ω to $E^*[index\_low - 1]$
                $increment = -1$
                Delete ω from $E^*[index\_low]$
            Delete ω from Ω
        end
    $j = j - 1$
    $index\_low = index\_low + increment - 1$
    $index\_opt = index\_opt - 1$
end
$E^* = clean(E^*)$
Return E^*

(from line 12 to 21) the refined ordered partition is $E^* = \{\{\omega_0\}, \{\omega_2, \omega_4\}, \{\omega_1, \omega_7\}, \{\}, \{\}, \{\}, \{\}, \{\omega_6\}\}$ *and the remaining configurations in* Ω *are* $\{\omega_3, \omega_5\}$*. Moving to instructions from line 25 to 36, the layer before last is associated with* $E_\Pi[1] = \{\omega_1, \omega_3, \omega_5, \omega_7\}$ *from which are deleted* ω_1 *and* ω_7 *since they already have been ranked. No configurations are now left to be ranked. After deleting empty layers, the output refined ordering is* $E^* = \{\omega_0\}, \{\omega_2, \omega_4\}, \{\omega_1, \omega_7\}, \{\omega_3, \omega_5\}, \{\omega_6\}\}$ *composed of* 5 *layers while* E_Π *and* E_Δ *only contain 3 levels.*

5.4 Refining Pareto ordering with default rules

The Pareto order is an indisputable semantics for ordering preference solutions that coincides with the product chain rule order for symbolic π-pref nets with no additional constraints. Moreover, recent researches on the compatibility of CP-nets ordering with the Pareto ordering [Wilson et al., 2019] have proved that the CP-net ordering refines the Pareto ordering adding the *Ceteris Paribus* assumption to it.

In this paragraph, we propose to refine the Pareto ordering by chiefly considering it as the basic ordering to which we add information such as the one brought by the application of the minimum or maximum specificity postulates. Thus, we propose to refine the Pareto ordering with the optimistic and pessimistic orderings.

Each of the mentioned orderings are represented by a $2^N \times 2^N$ incidence matrix denoted M (M_{Pareto} for the Pareto ordering matrix, M_Π for the optimistic ordering matrix and M_Δ for the pessimistic one). We use the following encoding for describing dominance relations between two configurations:

- if $\omega_i \succ \omega_j$ then $M[i,j] = 1$ and $M[j,i] = 0$,
- if $\omega_i \simeq \omega_j$ then $M[i,j] = M[j,i] = 1$.

Algorithm 5.3 outlines the process for refining the Pareto ordering by a default ordering. The idea is to try resolving incomparabilities of the Pareto ordering by the minimum or maximum specificity principles.

Function $move(E, \omega, i, j)$ removes ω from $E[i]$ and place it in $E[j]$. Functions $Pref(M, \omega, \omega')$ and $clear(E)$ are the same as in Algorithm 5.3 and 5.2 respectively.

If two solutions are incomparable by the Pareto ordering but comparable with the default ordering, then preferences in M_{Pareto} are updated. The $Warshall$ function is then applied on M_{Pareto} to compute the transitive closure after each refinement, i.e.,

propagate modifications in M_{Pareto}. The time complexity of $Warshall$'s function is $O(2^{N^3})$ making the complexity of Algorithm 5.3 equal to $O(C^2_{2^N}.2^{N^3}) = O(2^N)$ since it performs $C^2_{2^N}$ comparisons with at most one call to $Warshall$'s function.

Algorithm 5.3: REFINING PARETO ORDERING BY DEFAULT ORDERING

Input: M_{Pareto}, Ω, $M = M_\Pi$ or $M = M_\Delta$
Output: $M_{RefinedPareto}$

```
1  M_RefinedPareto = M_Pareto
2  for i = 0 to 2^N − 2 do
3  |  for j = i + 1 to 2^N − 1 do
4  |  |  ω_i = Ω[i]
5  |  |  ω_j = Ω[j]
6  |  |  ≻_Pareto = Pref(M_Pareto, ω_i, ω_j)
7  |  |  ≻ = Pref(M, i, j)
8  |  |  if (≻_Pareto = −1 and ≻ = 1) or (≻_Pareto = −1 and ≻ = 2) then
9  |  |  |  M_RefinedPareto[i, j] = M[i, j]
10 |  |  |  M_RefinedPareto[j, i] = M[j, i]
11 |  |  |  M_RefinedPareto = warshall(M_RefinedPareto)
12 |  end
13 end
14 Return M_RefinedPareto
```

In the following Example 5.3, we illustrate the application of Algorithm 5.3, showing that indeed we substantially diminish the number of incomparabilities both with the minimum and the maximum specificity principle.

Example 5.3 *Let us consider π-pref net in Figure 5.1. Its induced Pareto ordering is represented by Figure 5.3. The Pareto ordering succeeds to rank order* 65 *leaving* 55 *incomparable pairs of configurations. The optimistic ordering is* $E_\Pi = \{\{\omega_3\}, \{\omega_0, \omega_2, \omega_4, \omega_{12}\}, \{\omega_1, \omega_5, \omega_6, \omega_7, \omega_8, \omega_9, \omega_{10}, \omega_{13}, \omega_{14}\}, \{\omega_{11}, \omega_{15}\}\}$. *The well ordered partition induced considering the pessimistic approach is* $E_\Delta = \{\{\omega_0, \omega_4\}, \{\omega_1, \omega_2, \omega_3, \omega_5, \omega_7, \omega_{10}, \omega_{12}, \omega_{13}, \omega_{14}\}, \{\omega_6, \omega_8, \omega_{11}, \omega_{15}\}, \{\omega_9\}\}$. *After performing Algorithm 5.3, the number of Pareto incomparabilities decreases to* 29 *when refining either with* E_Π *or* E_Δ. *Each of these default orderings strongly violate the Pareto order on* 2 *relations on pairs of configurations that is corrected in the final result. When considering one symbol for each preference statement, the number of Pareto incomparabilities increases to* 84/120. *When refining with the minimum specificity ordering this number reduces to* 38. *When refining with the maximum specificity ordering, Pareto incomparabilities diminishes to* 32.

The partial order obtained by Algorithm 5.3 can be turned into a complete preorder if we also exploit the ties of the optimistic (or the pessimistic) ordering(s).

5.5 Experimental study

For some networks, the Pareto and default orderings may contradict each other on the ranking of some pairs of configurations. In Section 5.3, we showed that for networks for which the default orderings do not contradict the Pareto ordering, when all the dominance relations of the optimistic and pessimistic orderings are considered jointly, we can entail a total preorder on configurations that is highly discriminating, with fewer equivalence relations between the pairs of configurations. The first purpose of our experiments consists on confirming this assertion. The second goal of our experimental study is to evaluate the ordering obtained by the Pareto ordering refinement procedure by the different default orderings (optimistic, pessimistic and both of their induced orderings jointly).

5.5.1 Experimental protocol

To conduct our experimental study, we use the same set of randomly generated conditional preference networks considered in the second part of the experiments in Section 4.7 of Chapter 4. We recall that we used the generation tool *GenCPnet* [Allen et al., 2016] to generate a set of preference network instances in which we varied the number of nodes n from $n = 3$ to $n = 7$ and the maximum in-degree c from 1 to $n - 1$. We consider only bivalent decision variables, which means that d is fixed at 2.

Experiments are divided in two parts. The goal of the first part is to confirm that the Algorithm 5.2 induces a total preorder not only free from Pareto contradictions but also highly refined.

The second part is conducted on the whole benchmark and shows results on the refinement of the Pareto ordering by well-ordered default partitions comparisons.

These experiments are also conducted using the same test server as experiments of Chapter 4. The server technical specifications are described with an Intel Core $i7 - 7700HQ$ processor and a 20Go workstation. We used the toolbox presented in chapter 7 for all experiments.

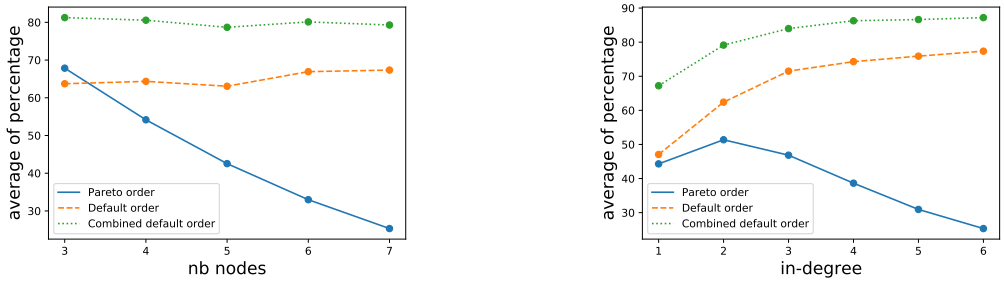
5.5.2 Experimental results

For the first part of the experiment, we first train the optimistic and pessimistic default network orderings on the benchmark described in section 4.7.1. We then consider only those networks whose default orderings do not contradict the Pareto order (73.03%) of the total number of instances). The two default partitions are then used to construct a more refined ordering that we call *combined default ordering*.

For each partitioning approach, we start by varying the size of the network and verify the behavior of strict dominance relations with respect to in-degree evolution. Figure 5.4 (a) shows the experiment results. We can see that, regardless of the size of the graph, the combined default order induces a highly discriminating order and succeeds in finding a strict ranking for almost 80% of the configuration pair comparisons.

In the same manner, for each partitioning approach, we vary the network size examine the behavior of strict dominance comparisons with respect to the in-degree evolution. Figure 5.4 (b) gives results of this experiment. The graphical plot shows that the higher the in-degree of the graph, the more discriminating the combined default order is and the less refined the Pareto order is.

This experiment confirms that the Algorithm 5.2 induces a total pre-order that is not only free of contradictions with the Pareto strategy but also highly refined.

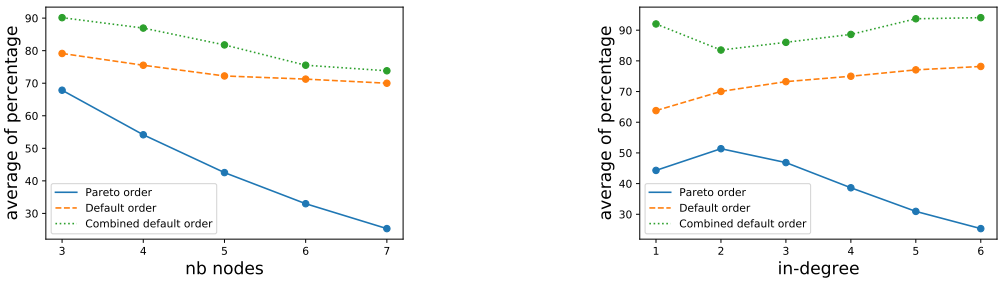


(a) (b)

Figure 5.4: Average of the percentage of strict order dominance relations as a function of (a) the network sizes and (b) the network in-degrees for networks with no Pareto contradictions

For the second part of the experiment, we consider the Pareto order and refine it by different ordering of the default partitioning procedures. For different network sizes, Figure 5.5(a) depicts the behavior of the percentage of strict dominance relations of the Pareto order refined by the optimistic partition (orange dashed curve) and refined by the combined default ordering (green dashed curve). For different network in-degrees, Figure 5.5(b) shows the relationship between the percentage of strict comparisons of

the Pareto order refined by the optimistic ordering (orange dashed curve) and then by the combined default ordering (green dashed curve).



(a) (b)

Figure 5.5: Average of the percentage of strict order dominance relations as a function of (a) the network sizes and (b) the network in-degrees for networks of all the benchmark

5.6 Conclusion

When comparing two potential solutions with respect to a set of conditional preferences, the Pareto partial order acknowledging that the set of preferences violated by one is a subset of the preferences violated by the other, is a natural basis for ordering solutions. In this chapter, we have shown that it is possible to enrich the Pareto ordering. To this end, we take advantage of two approaches, in terms of default-like constraints corresponding to the optimistic and to the pessimistic readings, which are been proved in Chapter 4 to be compatible with the π-pref net representation.

The constraints-based approach can lead to a complete pre-order once an optimistic or a pessimistic principle is applied, but at the price of some violations of the Pareto order. This chapter has proposed three algorithms for mending this situation, (i) one that leads to repair specificity-based orderings by the Pareto order, (ii) one that combines the optimistic and the pessimistic orderings, (iii) and a last one that refines the Pareto order by means of the optimistic or the pessimistic orderings.

The first and last algorithms can always be used, while the second one which may give more discriminant orderings is only applicable when the specificity-based orderings do not conflict with Pareto order.

In the next chapter, we investigate an approach briefly mentioned in Chapter 1, that is based on lexicographic preference trees (LP-trees for short). Its merits is to always lead to a total ordering. The LP-tree approach heavily relies on complete pre-orders stating the relative importance of the variables in the specifications of constraints.

This kind of information can be handled in the approaches handled in Chapters 1 to 5, in TCP-nets to a limited extent, and without restrictions in π-pref nets by the addition of constraints between symbolic weights.

Chapter 6

Comparing Possibilistic Preference Networks and LP-trees

6.1 Introduction

Preferences of a user can be represented by various graphical structures. The most used among them have been discussed in the first chapter. In order to express preference relations, the user may provide satisfaction degrees on values of conditional variables which amounts to construct the conditional preference table of a π-pref network. Alternatively, he may provide a strict total order on values of decision variables which may be attached to dependency constraints along with importance relations between variables. This latter set of information permits to construct an LP-tree.

As we shall see, there are different kind of LP-trees which are detailed in this chapter, where we also seek to compare π-pref nets and LP-trees as structures for representing preferences over Boolean variables.

The chapter is organised in five main sections. Section 6.2 reviews the different classes and extensions of LP-trees and provides their detailed definitions. Section 6.3 details the procedure to be applied to LP-trees in order to infer a (total) order on the set of solutions. Section 6.4 focuses on the comparison between LP-trees and π-pref nets. Section 6.5 discusses the transformation procedure for translating an LP-tree into a π-pref net, to finally end with concluding remarks.

This chapter develops the work published in [Ben Amor et al., 2022] and its French version in [Ben Amor et al., 2021b].

6.2 Classes of LP-trees

Remember from Chapter 1 that an LP-tree is a preference model defined both from an order relation between variables that expresses their relative importance, and from local preference relations on the domains of the variables. Both the importance relationship between variables and the local preferences can be conditioned on the values of other variables. For a better study of this model, arcs that represent this information will be differentiated in the graphical structure:

- **cp-arcs** depicted by unlabeled solid arrows reflect dependencies between variables;
- **i-arcs** represented by dotted arrows reveal importance relations between variables. They may be labeled by context(s) of more important variables.

An LP-tree can combine conditional and unconditional preferences and importance relations. The following example deals with the case of a general LP tree.

Example 6.1 *In Figure 6.1, cp − arcs from A to B and from B to C show that a preference dependency of B over A and from C over B. Preferences over variables A and D are not context-dependent. Priority between variables is not conditional in the left-hand branch from B to D ($B \triangleright C \triangleright D$), which is not the case for nodes B and D of the first layer ($A \triangleright_a B$, $A \triangleright_{\bar{a}} D$). Given $\bar{a}$, the preferences and priority relations of the variables B and C are not provided, resulting in an incomplete right branch in the tree.*

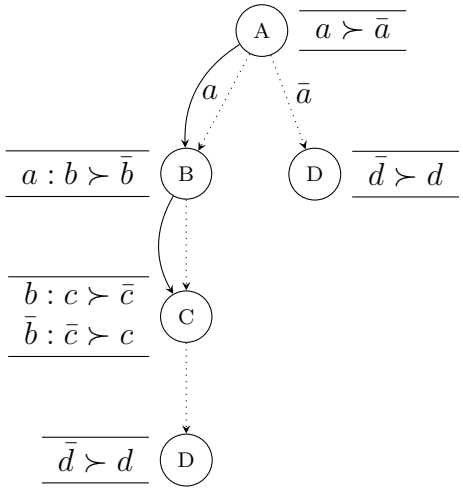


Figure 6.1: An example of a general LP-tree

There are several types of LP-trees [Booth et al., 2010] depending on (i) whether or not the preferences are conditional, (ii) whether or not the order of importance of the variables depends on the value of more important variables. Each of these options induces a separate LP-tree class. Next Section 6.2.1 regroups and details LP-trees obtained by imposing conditioning restrictions on importance relations, while Section 6.2.2 deals with LP-trees obtained by imposing conditioning on preferences.

This taxonomy of LP-trees into several classes has first been proposed in [Booth et al., 2010]. However, we can find in the literature [Bräuning and Hüllermeier, 2016] other classification principles for describing classes of LP-trees, by considering the conditioning on importance relationships as a categorization criterion.

The two types of conditioning can be combined as detailed in Section 6.2.3, while Section 6.2.4 provides an overview of the different classes of LP-trees.

Preferences may remain the same for whatever the branch in the tree. Section 6.2.5 is dedicated to the definition and study of LP-trees with such *fixed* preferences.

When some specifications are missing the LP-tree is considered incomplete. Preferences and importance relationships between variables can describe what is called a *complete* LP-tree. This property is defined in Section 6.2.6.

Also, an LP-tree may have nodes with multiple variables. If a node of the tree has up to k decision variables all grouped together, we speak of k *LP-trees*, which were first introduced in [Bräuning and Hüllermeier, 2016]. Section 6.2.7 is dedicated to this extension of LP-trees.

6.2.1 Conditioning on importance relations

The order of importance of variables can depend on the value of more important variables, an idea also introduced in [Booth et al., 2010] and investigated in [Bräuning and Hüllermeier, 2012]. Restraining priority relations to be conditional or not generates two classes of LP-trees characterized by: *unconditional* or *conditional importance relations*. A lexicographic order on variables exempt of any conditioning is graphically depicted by a linear path graph such that if $X_i \triangleright X_j$ then variables labeled with X_i and X_j are allied by an unlabeled $i - arc$. The order expressed on values of each variable can be conditioned by values of other variables. When all priority relations between variables do not depend from values of other nodes, we refer to the class of *UI* LP-trees for *Unconditional Importance* relations LP-trees. The importance relation between variables can depend on the value of the more important variables.

Each new dependency induces a new sub-branch. Each path from the root to a leaf defines an order of importance of the variables, for the context of the value(s) that determined the branch(es) defining this path. When an LP-tree encodes both types of importance relations, i.e., conditional and unconditional, we speak of *CI* LP-trees for *Conditional Importance* relations LP-trees (see Definition 6.1). This class of trees must hold at least one conditional importance relation.

Definition 6.1 (CI LP-tree) *A Conditional Importance relation LP-tree over decision variables $\mathcal{X}$ is defined by a tuple $\langle \mathbf{V}, \mathbf{E}, PT \rangle$ where:*

(i) $\mathbf{V}$ *correspond to the set of nodes. Each node is labeled with a decision variable in $\mathcal{X}$;*

(ii) $\mathbf{E}$ *is the set of arcs composed of*

- *$i-arc$ of the form $U_X \dashrightarrow X$ with $\{U_X, X\} \in \mathcal{X}$. An $i-arcs$ can either be labeled or not indicating a conditioned importance relation or not respectively;*
- *$cp-arc$ of the form $X_i \rightarrow X_j$ with $\{X_i, X_j\} \in \mathcal{X}$ and $X_i \in U_{X_j}$. A $cp-arc$ is unlabeled and indicate that preferences over a variable are conditioned by values of its parents.*

(iii) $PT = \{PT(V_1), \ldots, PT(V_M)\}$ *corresponds to the set of local tables where $PT(V_i)$ is a preference table that describes a total order on values of X_j such that X_j is the label of V_i. $PT(V_i)$ contains specifications either of the form $u_{V_i} : X_j \succ \neg X_j$, or of the form $X_j \succ \neg X_j$ where $u_{V_i} \in \underline{U_{V_i}}$.*

Each variable $X \in \mathcal{X}$ appears at most once in each branch of the tree. A CI LP-tree should respect the following statements:

1. *There must exist at least one conditional importance relation in the tree;*
2. *All nodes in the tree must be connected by an $i-arc$.*

Figure 6.2 depicts an example of a CI LP-tree where all importance relations are conditional. However, nothing prevents preferences to be all context-independent.

As for *UI* LP-trees, the formal definition is given below.

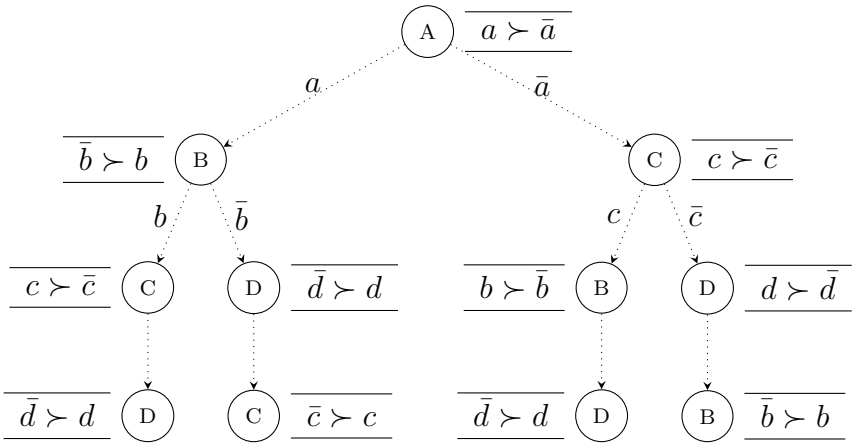


Figure 6.2: Example of a CI LP-tree

Definition 6.2 (UI LP-tree) *An Unconditional Importance relations LP-tree over decision variables $\mathcal{X}$ is a linear graph composed of the tuple $\langle \mathbf{V}, \mathbf{E}, PT \rangle$.*

A UI LP-tree should respect the following statements:

1. *Each variable $X_i \in \mathcal{X}$ appears at most once in each branch of the tree;*

2. *All importance relations are unconditional;*

3. *The set of arcs $\mathbf{E}$ contains both $i-arcs$ and $cp-arcs$. All $i-arcs$ are unlabeled of the form $U_X \dashrightarrow X$ expressing that U_X is more important than X regardless of the value of U_X;*

4. *Each local table $PT(V_i)$ describes a total order on values of X_j such that X_j is the label of V_i. A local table $PT(V_i)$ contains either conditional preferences of the form $u_{V_i} : X_j \succ \neg X_j$ with $u_{V_i} \in \underline{U_{V_i}}$, or unconditional preferences of the form $X_j \succ \neg X_j$.*

Figures 6.3 (a) and (b) are illustrative network examples defined on the same set of variables: $\mathcal{X} = \{A, B, C\}$. Both trees correspond to the class of UI LP-trees. In Figure 6.3 (a) all variables are independent, the network thus encodes the importance relation $A \triangleright B \triangleright C$ and preferences $a \succ \bar{a}$, $\bar{b} \succ b$ and $\bar{c} \succ c$. In this case, graphical modeling does not seem to be the most concise way to encode all the given information. Unlike this network, preferences in Figure 6.3 (b) are partially conditional since preferences over B depend from A.

(a) (b)

Figure 6.3: Examples of UI LP-trees

6.2.2 Conditioning on preferences

The preference statements on values of variables may or may not be conditioned by contexts expressed by the values of parent variables. This two situations give birth to two classes of trees respectively called: *unconditional* or *conditional preferences* LP-trees. When all the variables of the tree are context-free, i.e., independent, we speak of *UP* LP-trees for *Unconditional Preferences* LP-trees. When the local tables contain both conditional and unconditional preferences we deal with *CP* LP-trees for *Conditional Preferences* LP-trees. When a node X depends from U_X then the user must furnish a strict total order over assignments of X for each context value $u_X \in \underline{U_X}$. For both *CP* and *UP* LP-trees, the priority over variables can be unconditional or conditioned by assignments of more important nodes. However, in CP LP-tree there must exist at least one context-dependent preference, while in UI LP-trees all importance relations should be context-free. In the following we give formal definitions of these two LP-tree types.

Definition 6.3 (CP LP-tree) *A Conditional local preferences LP-tree over decision variables $\mathcal{X}$ is defined by a tuple $\langle \mathbf{V}, \mathbf{E}, PT \rangle$ where:*

(i) $\mathbf{V}$ *correspond to the set of nodes. Each node is labeled with a decision variable in $\mathcal{X}$;*

(ii) $\mathbf{E}$ *is the set of arcs composed of*

- *$i-arc$ of the form $U_X \dashrightarrow X$ with $\{U_X, X\} \in \mathcal{X}$. An $i-arcs$ can either be labeled or not indicating a conditioned importance relation or not respectively;*

- *cp − arc of the form* $X_i \rightarrow X_j$ *with* $\{X_i, X_j\} \in \mathcal{X}$ *and* $X_i \in U_{X_j}$. *A cp − arc is unlabeled and indicate that preferences over a variable are conditioned by values of its parents.*

(iii) $PT = \{PT(V_1), \ldots, PT(V_M)\}$ *corresponds to the set of local tables where* $PT(V_i)$ *is a preference table that describes a total order on values of* X_j *such that* X_j *is the label of* V_i. $PT(V_i)$ *contains specifications either of the form* $u_{V_i} : X_j \succ \neg X_j$, *or of the form* $X_j \succ \neg X_j$ *where* $u_{V_i} \in \underline{U_{V_i}}$.

Each variable $X \in \mathcal{X}$ *appears at most once in each branch of the tree. A CP LP-tree should respect the following statements:*

1. *There must exist at least one context-dependent preference relation;*

2. *All nodes in the tree must be connected by an* $i - arc$.

Definition 6.4 (UP LP-tree) *An Unconditional local preferences LP-tree over decision variables* $\mathcal{X}$ *is defined by the same structure as a CP LP-tree, i.e.,* $\langle \mathbf{V}, \mathbf{E}, PT \rangle$.

A UP LP-tree should respect the following statements:

1. *The set of arcs* $\mathbf{E}$ *only contains* $i-arcs$ *of the form* $U_X \dashrightarrow X$ *with* $\{U_X, X\} \in \mathbf{V}$. $i-arcs$ *can either be labeled by the value(s) of* U_X[1] *or not indicating a conditioned importance relation or not respectively;*

2. *Each local table* $PT(V_i)$ *describes a total order on values of* X_j *such that* X_j *is the label of* V_i, *and only contains specifications of the form* $X_j \succ \neg X_j$;

3. *All nodes* $V_i \in \mathbf{V}$ *labeled with the same variable* $X_j \in \mathcal{X}$ *have the same preference order over values in* $\underline{X_j}$.

Figure 6.4 is an illustrative example of the classe of CP LP-tree.

[1] An i-arc $U_X \dashrightarrow X$ labeled by all context values of the parent node encodes the same information as an unlabeled i-arc, which is that U_X is more important than X regardless of the value of U_X.

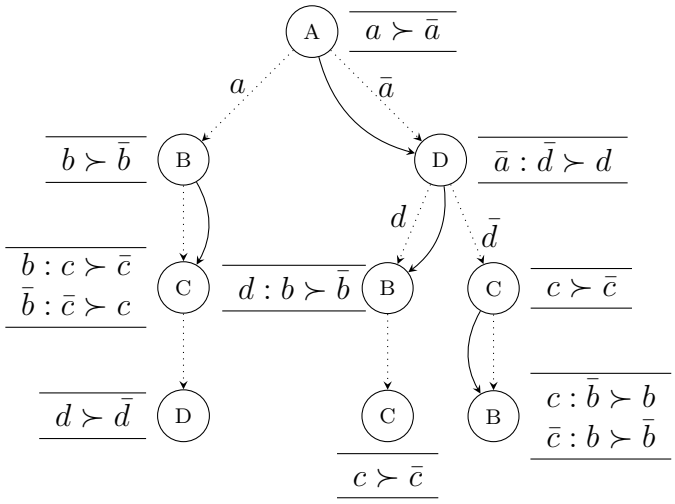


Figure 6.4: Example of a CP LP-tree

6.2.3 Other classes of LP-trees

The conditioning properties on local preferences and on the importance relation of an LP-tree can be combined, which yields four other classes of LP-trees, namely *UP-UI*, *CP-UI*, *UP-CI* LP-trees, *CP-CI*. For instance, Booth et al. define the class of *CP-UI* as

"(...) the class of all LP-trees with conditional preferences and an unconditional importance relation". [Booth et al., 2010]

No further and explicit definitions about these classes are provided in the literature. Based on network examples given in [Booth et al., 2010], *CP-UI* LP-trees do not permit to express conditioning on priority relations between variables. However, they can hold both conditional and unconditional preferences. Therefore, the class of UI LP-trees is the union of the classes CP-UI and UP-UI (see Figure 6.5).

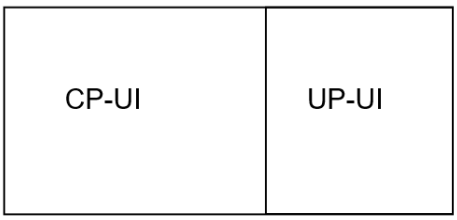


Figure 6.5: Class of UI LP-trees

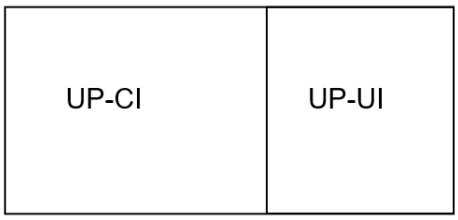


Figure 6.6: Class of UP LP-trees

In the same vein, the class of *UP-CI* would be defined by "the class of all LP-trees with unconditional preference and conditional importance relations". Such a

class should hold structures with both conditional and unconditional priority relations between variables. Moreover Bouth. et. al., privilege LP-trees where preferences over each decision variable are the same in all branches of the tree in addition to the fact that they have to be context-independent. Thus, the class of UP LP-trees is the union of the classes UP-UI and UP-CI (see Figure 6.6).

With the same reasoning mind, the class of *CP-CI* would be defined by "the class of all LP-trees with at least one conditional preference and at least one conditional importance relation". This means that preferences can either be context-dependent or not, and importance relations too. Therefore, if we focus on the conditional relations, the class of CP-CI corresponds to the intersection between the classes CP-UI and UP-CI.

The class of *UP-UI* defines the most restrictive LP-tree structure namely with unconditional preference and unconditional importance relation. The formal definition of this class is given below.

Definition 6.5 (UP-UI LP-tree) *An Unconditional local Preference and Unconditional Importance relation LP-tree over decision variables $\mathcal{X}$ is a linear graph defined by $\langle \mathbf{V}, \mathbf{E}, PT \rangle$.*

An UP-UI LP-tree should respect the following statements:

- *Each variable $X_i \in \mathcal{X}$ appears at most once in the tree;*
- $\mathbf{E}$ *only contains $i-arcs$;*
- *Each node V_i has one unlabeled outgoing $i-$arc of the form $U_X \dashrightarrow X$ expressing that U_X is more important than X regardless of the value of U_X;*
- *Each local table $PT(V_i)$ describes a total order on values of X_j such that X_j is the label of V_i. It only contains specifications of the form $X_j \succ \neg X_j$.*

If we consider unconditional relations, the class of UP-UI corresponds to the intersection between the classes CP-UI and UP-CI.

The following network illustrates an example of a UI LP-tree.

6.2.4 Discussion on other sub-classes

Given the LP-trees taxonomy discussed in previous sections, there exists no class that restricts preference relations as well as importance relations to be only conditional

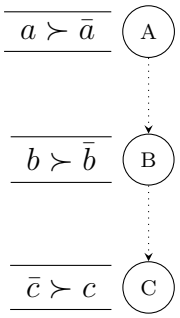


Figure 6.7: Example of an UP-UI LP-tree

everywhere. When these restrictions are combined, we may have additional classes of LP-trees that have never been defined and considered in the previous researches. In this section, we propose new classes of LP-trees defined in following:

- A **SCP** for *Strong Conditional Preferences* is an LP tree that contains only context-dependent preferences, whenever the dependency relationship is possible. Importance relations are free to be conditional or not (except for the most important variable), i.e., all nodes have at least one in-going $cp - arc$;
- A **SCI** for *Strong Conditional Importance relations* LP-trees contains only conditional importance relations. Preferences can be context-dependent or not. Root and intermediate node, except for parents of leaf nodes, have two outgoing labeled $i - arcs$. Parents of leaf nodes have one unlabled outgoing $i - arc$.

The combination of restrictions on conditioning on relationships can give rise to another class of LP-trees: **SCP-SCI** LP-trees that contain only conditional preferences and importance relations. The size of the graphical structures in this class grows exponentially as the number of decision variables increases. On a complexity scale, *SCP-SCI* trees would be the heaviest to learn and to elicit, while *UP-UI* would be the simplest ones and the less complicated to construct.

Figures 6.8, 6.9 and 6.10 represent examples of an SCP, SCI and SCP-SCI LP-tree, respectively.

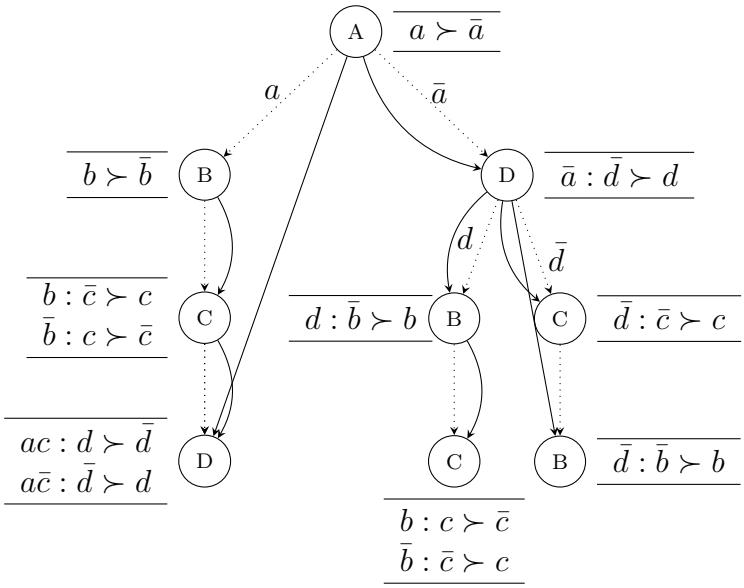


Figure 6.8: Example of an SCP LP-tree

6.2.5 Fixed preferences

The order of importance between variables can be independent or conditioned by some values. But whatever the case, preferences can be fixed, which means that they remain unchanged on all branches of the LP-tree forming the class of *FP* LP-trees [Booth et al., 2010]. Generally, local fixed preferences can be context-dependent.

In the following we give a formal definition of an *FP* LP-tree (see Figures in 6.11 for illustration).

Definition 6.6 (FP LP-tree) *A Fixed Preference LP-tree over decision variables $\mathcal{X}$ is defined by a pair $\langle \mathbf{V}, \mathbf{E}, PT \rangle$ where:*

(i) $\mathbf{E}$ *contains both $i-arcs$ and $cp-arcs$;*

(ii) $PT = \{PT(V_1), \ldots, PT(V_M)\}$ *is a set of local tables where V_i is labeled with $X \in \mathcal{X}$ and $PT(V_i)$ contains preference specifications of the form $X \succ \neg X$ (or $u_X : X \succ \neg X$ where $u_X \in \underline{U_X}$).*

Each variable $X_i \in \mathcal{X}$ appears at most once in each branch of the tree. A FP LP-tree should respect the following statement:

1. *Each variable X_i keeps the same dependency constraint and preference specifications in all branches on the tree.*

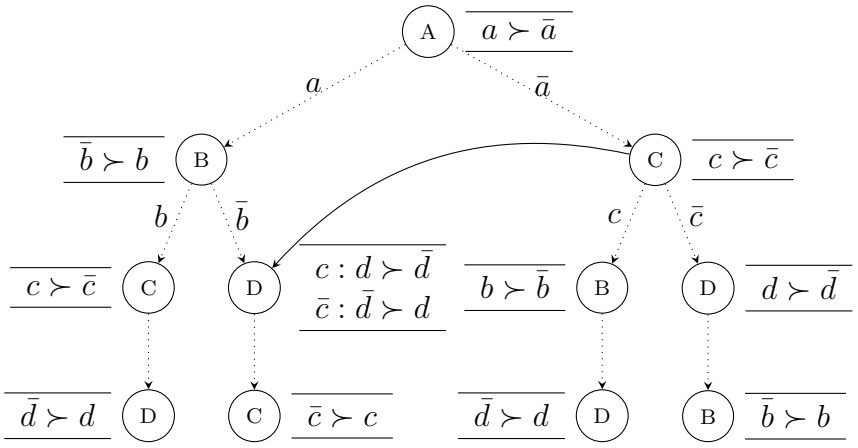


Figure 6.9: Example of an SCI LP-tree

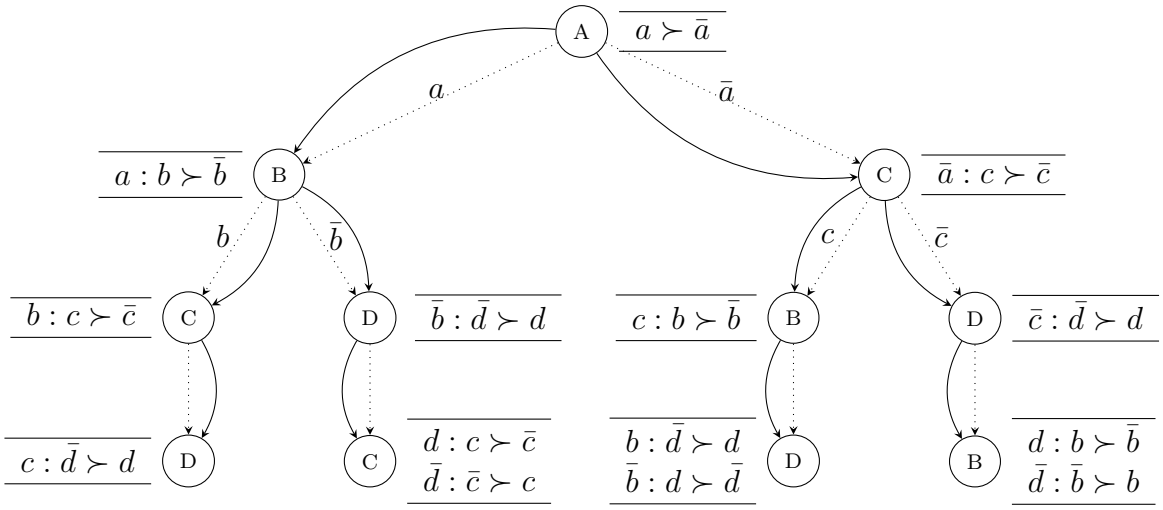


Figure 6.10: Example of an SCP-SCI LP-tree

Preferences can be fixed for both conditional and inconditional dependencies. Figure 6.11(a) gives an example of an FP LP-tree where all preferences are context-free. Figure 6.11(b) shows an example of an FP LP-tree with both conditional and unconditional preferences. Variables A, C and D are context-independent Preferences over values of their domains do not change in every branch of the tree. Preferences over the values of B depend from the value of C. This dependency relations is expressed in all branches of the tree and preferences over B remain fixed for every context in all the tree.

The fact that preferences are conditioned by values of one or many more important variables is not clearly allowed in the literature. However, this case may lead to unsatisfiable networks (see Section 6.3 for more details). When (i) the less important

variables depend from the more important ones and, (ii) if the lexicographic order over the dependent variables is the same in all branches of the tree, this troubling situation can be avoided.

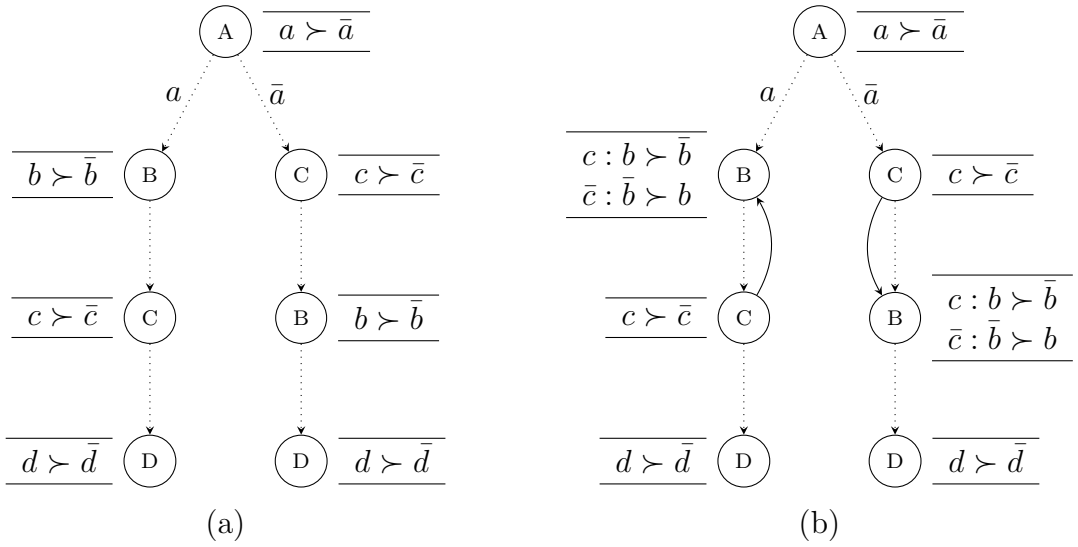


Figure 6.11: Example of an FP LP-tree (a) with unconditional, (b) conditional preferences

We can specify sub-classes of FP LP-trees, depending on whether or not the conditioning over importance relations and preferences is allowed. Table 6.1 summarizes these sub-classes. We use a $\sim$ to mention that the network contains both conditional and unconditional information. Symbols ✓ (respectively ✗) indicate that all relations in the tree are conditional (respectively inconditional). We use the exclamation mark to indicate a strong preference. The first and second columns specify whether or not the conditioning is performed on preferences and priority relations respectively. For instance the general class of FP trees contains both conditional an inconditional preferences as well as importance relations. It is represented by the first row of the Table 6.1. If we consider definitions given in 6.2.4 and depending on whether the conditioning covers all preference relations or not, we can define more sub-classes of FP LP-tree (see last 3 rows of Table 6.1). For example, the last row should be read as follows: When an LP-tree structure contains only conditional preferences with no conditioning on importance relations, then it belongs to the set of *FSCP-UI* LP-trees.

Note that *FP* LP-trees should not be confused with the class of *UP* LP-trees. Consider the variable X_i of a *FP* tree. In all branches of the graph the local preference relation over X_i is unique and equals $x_i \succ \bar{x}_i$. In contrast, if we consider a *UP* LP-tree, then specifications in X_i can take either $x_i \succ \bar{x}_i$ or $\bar{x}_i \succ x_i$. Actually, for a given number N of variables, the set of possible FP LP-trees is included in the set of possible UP LP-trees [Booth et al., 2010].

preference dependencies relations	**conditioning on importance relations**	**subclass**
∼	∼	FP
∼	✓	FP-CI
∼	✗	FP-UI
✗	∼	FUP
✗	✓	FUP-CI
✗	✗	FUP-UI
✓	∼	FCP
✓	✓	FCP-CI
✓	✗	FCP-UI
!	∼	FSCP
!	✓	FSCP-CI
!	✗	FSCP-UI

Table 6.1: New sub-classes of FP LP-trees

Most LP-trees preference learning algorithms proposed in the literature are greedy. Therefore, finding an upper-bound approximation of the number of possible FP LP-trees given a fixed number of variables seems interesting for future works.

Proposition 6.1 *Let $\mathcal{X}$ be composed of N preference variables. Considering unconditional dependency relations between variables, there exists $\prod_{i=1}^{N}(2 \times i)$ possible UI trees but only $N!$ FUP UI trees.*

Proof 6.1 *Since we consider that variables are independent, the preference over the parent node does not affect preference over its descendant. Thus, local preference tables of a UI tree are independent from the structure of the graph. While keeping in mind that by definition any UI LP tree is a path graph (see Definition 6.2), finding the number of possible FP trees given a number N of features comes down to find the set of linear graphs that can be constructed with N nodes, which corresponds to the number of permutations of N distinct objects i.e. $N!$.*

Given N, the idea is to find, for each path graph from $N!$ graphs, the number of UI trees that can be constructed. Let us fix the preference of the root node to $x \succ \bar{x}$ [2]. The next node X' holds a specification of the form $x' \succ \bar{x'}$ or $\bar{x'} \succ x'$. Thus we can construct two paths from node X to node X'. Consequently, the number of LP trees with N nodes equals $\prod_{i=1}^{N}(2 \times i)$.

For example let us consider $N = 2$ decision variables, namely A and B. There are two possible networks that can be constructed with either A being the root node

[2]The same reasoning is considered for the case $\bar{x} \succ x$

and B its children or inversely (see above most structures in Figure 6.12). If we fix the preference statement associated to A to be $a \succ \bar{a}$ then local table of node B can take two statements: $b \succ \bar{b}$ or $\bar{b} \succ b$, which leads us to two possible path graphs. If we switch the preference statement associated with A to be $\bar{a} \succ a$, we would have two other possible networks leading to $2^N = 4$ graphs (see left-most trees in Figure 6.12. Trees in the rectangle with solid line rectangle are FUP-UI LP-trees. If we remove the restriction of fixed preferences, each one of them leads to four UP-UI LP trees depicted in rectangles with dotted lines). This number is multiplied by the number of possible path graph structures with N nodes leading to a total of $2^N \times N! = 8$ LP trees.

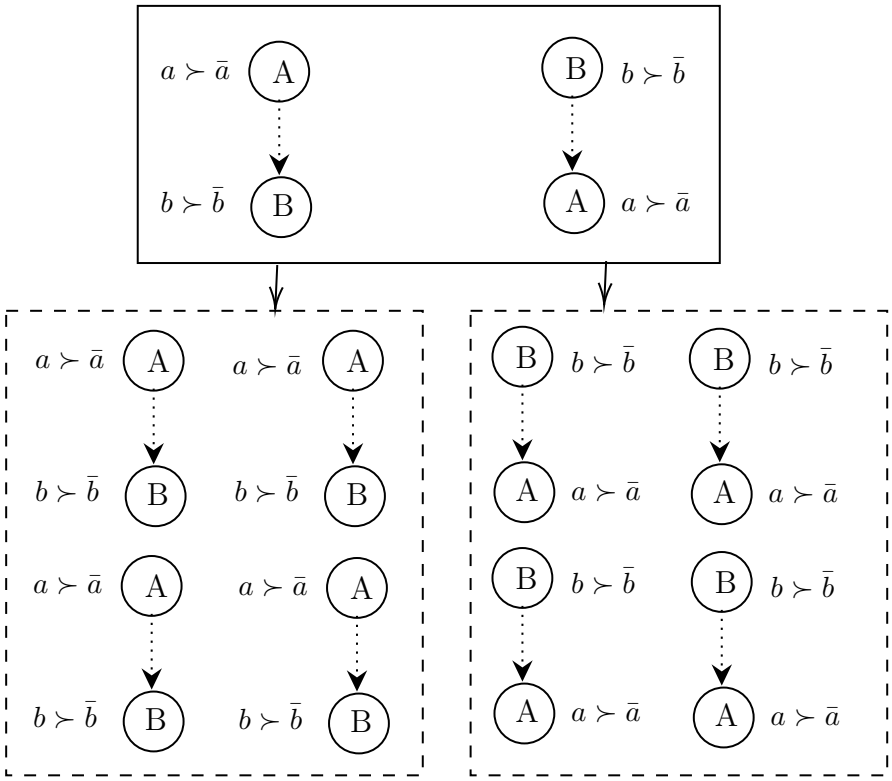


Figure 6.12: Sets of FUP-UI LP-trees (in solid line rectangle) and UP-UI LP-trees (in dotted line rectangles) given two decision variables

6.2.6 Completeness of an LP-tree

The notion of completeness of an LP-tree has been introduced in [Bräuning and Hüllermeier, 2016]. LP-trees which are said to be *complete*, require that a total importance order over variables is provided. Completeness covers only priority relationships since the order of preferences is complete by definition. A *complete* LP-tree should respect the following statements:

1. Each variable appears in each branch of the tree.

2. When preferences are conditional, all CPT should contain preference orders for all contexts of parents.

3. When preferences are not conditional all variables hold PTs that express a total order on its values.

4. When importance relations are conditional, the number of outgoing $i-arcs$ of all non-leaf nodes equals the size of the variable domain, each labeled with a node value. All leaves hold one unlabeled outgoing $i-arc$.

5. When importance relations are unconditional, all nodes have one unlabeled outgoing $i-arc$.

So far, except for the general LP-tree in Figure 6.1, all other network examples are complete. Figure 6.13 illustrates the example of incomplete LP-trees. In the rightmost branch of the tree, importance and preference information in the context of $\bar{c}$ are missing.

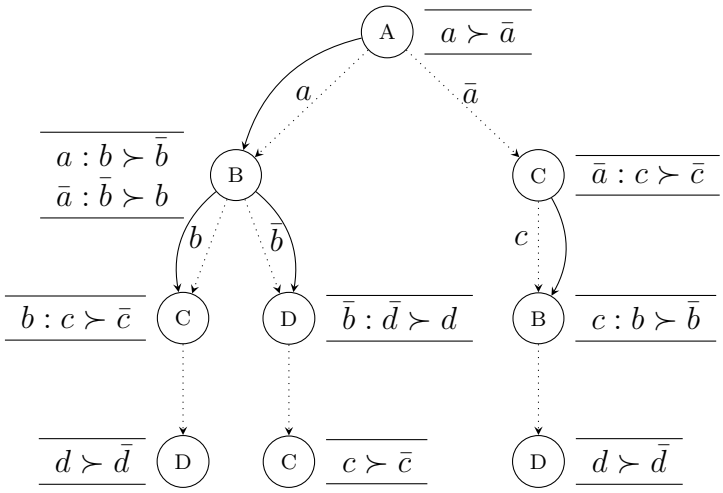


Figure 6.13: Example of an incomplete LP-tree

We mention that another class of trees called *partial lexicographic preference trees* (PLP-trees) has been introduced in [Liu and Truszczynski, 2015]. These networks allow some variables to be missing from some paths. They seem quite similar to incomplete LP-trees. This relaxation is of interest in a learning perspective
[Liu and Truszczynski, 2015]. we mention that PLP-trees describe total preorders (instead of total orders) on the set of configurations.

6.2.7 k LP-trees

In all previous defined classes, a variable is affected to a single node. However, a user may express importance equality for a set of features. A group of decision variables may have the same importance degree, they can thus be depicted by a single high dimensional node. An idea introduced in [Bräuning and Hüllermeier, 2016]. The expressiveness of LP-tree can thus be extended which gives rise to so called k *LP-trees* in which each node holds at most k grouped variables. Note that the fact of declaring two variables of equal importance does not authorise the separate expression of preferences concerning them. Preference statements in local tables attached to nodes with grouped variables define a total order over the Cartesian product of variable's domains. Any strict total order over a set $\mathcal{X}$ can be expressed by a k LP-tree, where $k = |\mathcal{X}|$.

As for previous classes, k LP-trees allow to make restrictions on conditioning over importance relation or on preference dependencies between variables. Figure 6.14 shows an example of 2 CP-UI LP-tree where nodes B and C share equal importance degree and do both depend from A.

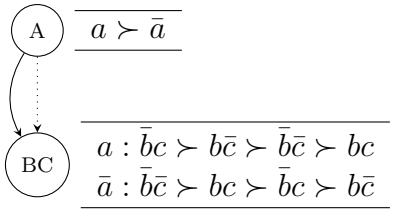


Figure 6.14: Example of a 2 CP-UI LP-tree

6.3 Order induced from an LP-tree

Unlike CP-nets and TCP-nets, finding the dominance relation $\succ_{Lex}$ between all pairs of solutions in $\mathcal{X}$ of an LP-tree can be ensured by a straightforward algorithm that consists of sweeping through $i-arcs$ of the tree and considering preferences associated to nodes. Algorithm 6.1 details the procedure of inferring an ordering given a complete UI LP-tree. Given a node $V \in \mathbf{V}$ labeled with X, we consider that x is always preferred to x'. Starting from the root node and according to its conditional table, configurations that model x are considered better than those that model x' s.t. $\{x, x'\} \in \underline{X}$. Formally, we can write $\forall \omega, \omega' \in \Omega$, the ordering $\succ_{Lex}$ specifies that if $\omega \models x$ and $\omega' \models x'$ then $\omega \succ_{Lex} \omega'$. The procedure is recursively repeated until reaching the leaf node.

Algorithm 6.1: LEXICOGRAPHIC ORDERING GIVEN AN UI LP-TREE $\mathcal{T}$

Input: $\mathcal{T}$, *end* `/* end equals the number of configurations */`
Output: $\succ_{Lex}$ `/* Ordered list of configurations from best to worst */`

1 **Algorithm:** function Ordering(i, *end*, $\succ_{Lex}$)
2 **begin**
3 **if** $i = |\mathbf{V}|$ **then**
4 $\succ_{Lex} = Reorder(res)$ `/* function Reorder keeps only the complete configurations */`
5 **return** $\succ_{Lex}$
6 **else**
7 $(pref, reject) = Pref\_Rejected\_Values(V)$ `/*` *pref* `(resp.` *reject*`) is the preferred (resp. rejected) value of variable in` V `*/`
8 **for** $j = 0$ *to* $end/2 - 1$ **do**
9 $Concatenate(res[j], pref)$
10 **end**
11 **for** $j = end/2$ *to* *end* **do**
12 $Concatenate(res[j], reject)$
13 **end**
14 $Add(\succ_{Lex}, res)$
15 $i = i + 1$
16 $tab1 = \succ_{Lex}.slice(0, end/2)$
17 $tab1 = \succ_{Lex}.slice(end/2, end)$
18 res = Ordering(i, $end/2$, $tab1$)
19 res = Ordering(i, $end/2$, $tab2$)
20 **end**
21 **end**

Example 6.2 *Let us consider the UI LP-tree in Figure 6.3. The root node labeled with A indicates that* $a \succ \bar{a}$*, the ordering array* $\succ_{Lex}$ *is thus composed of* 4 *cells containing a followed by* 4 *cells containing* $\bar{a}$*. The ordering array* $\succ_{Lex}$ *is spliced into two halves, namely tab*1 *composed of* 4 *cells containing a and tab*1 *composed of* 4 *cells containing* $\bar{a}$*. For each part, the procedure is repeated considering now the next important node B. Thus, tab*1 *equals now* $[ab, ab, a\bar{b}, a\bar{b}]$ *and tab*1 *equals now* $[\bar{a}b, \bar{a}b, \bar{a}\bar{b}, \bar{a}\bar{b}]$*. Same steps are considered for the last remaining node labeled with C to output the ordering* $\succ_{Lex}=$ $[ab\bar{c}, abc, a\bar{b}\bar{c}, a\bar{b}c, \bar{a}b\bar{c}, \bar{a}bc, \bar{a}\bar{b}\bar{c}, \bar{a}\bar{b}c]$ *such that configurations are organized from best to worst (configurations between brackets are rank-ordered according to the preference).*

In case of a conditional preference and importance relations, we have to consider each value of the variable X of a node $V_i \in \mathbf{V}$ separately. Starting by the root node, trace down the tree according to the preferred instantiation of each variable (in the context of parent value, if $\mathcal{T}$ is a conditioned preference LP-tree), which leads to getting the optimal alternative ω_{Opt}. Now consider the preference table $PT(L)$ corresponding to the least important variable $L \in \mathcal{X}$, where L labels the leaf node and has respective assignments $\{l, l'\}$ and preference relation $l \succ l'$ (or $u : l \succ l'$ s.t. $u \in \underline{U_L}$ for the conditional case). Since $\omega_{Opt} \models l$ (respectively ul), we can deduce that $\omega_{Opt} \succ \omega'$ s.t. $\omega' \models l'$ ((respectively ul'). The second step consists in flipping the value of $U(L)$ on its less preferred instantiation u' and verify $PT(L)$ given the new context u'. The process is repeated by subsequently considering variables from the least preferred ones until reaching the root node.

In the following example, we give the lexicographic induced ordering on configurations for all satisfiable LP-trees given in this chapter.

Example 6.3 *Let us consider LP-tree in Figure 6.1. Preference of the root node stipulates that* $a \succ \bar{a}$*, this means that all models of a should lexicographically dominate those of* $\bar{a}$*. When the variable A takes the value a, then the second most important variable is B. The contextual specification associated with this node is* $a : b \succ \bar{b}$*. This means that in configurations that satisfy a all configurations that satisfy b are preferred to those that satisfy* $\bar{b}$*. In the left-most branch the third most important variable is C. It holds preferences that depend from B. When a configuration satisfies b then* $c \succ \bar{c}$*. In the obtained order-ranking, configurations that satisfy c are preferred to those that satisfy* $\bar{c}$*. The least important node is D with the preference statement* $\bar{d} \succ d$ *which means that, whatever the value of the more important variables the configuration that verify* $\bar{d}$ *is preferred to d. Let us now move on to the right-most branch. The second most important variable given the value* $\bar{a}$ *of A is D. Configurations that model* $\bar{a}$ *and* $\bar{d}$ *are preferred than those that verify* $\bar{a}$ *and d. For ease of reading,*

we use the symbol $\succ$ *instead of* $\succ_{Lex}$ *throughout this example. The final ordering is:* $abc\bar{d} \succ abcd \succ ab\bar{c}\bar{d} \succ ab\bar{c}d \succ a\bar{b}\bar{c}\bar{d} \succ a\bar{b}\bar{c}d \succ a\bar{b}c\bar{d} \succ a\bar{b}cd \succ \omega \models \bar{a}\bar{d} \succ \omega' \models \bar{a}d$ *with* $\omega \neq \omega'$ *and* $\omega, \omega' \in \Omega$*. This order is a pre-order since the LP-tree is incomplete.*

We give the following lexicographic orderings for a set of LP-tree example:

- *Figure 6.2:* $a\bar{b}\bar{d}\bar{c} \succ a\bar{b}\bar{d}c \succ a\bar{b}d\bar{c} \succ a\bar{b}dc \succ abc\bar{d} \succ abcd \succ ab\bar{c}\bar{d} \succ ab\bar{c}d \succ \bar{a}cb\bar{d} \succ \bar{a}cbd \succ \bar{a}c\bar{b}\bar{d} \succ \bar{a}c\bar{b}d \succ \bar{a}\bar{c}d\bar{b} \succ \bar{a}\bar{c}db \succ \bar{a}\bar{c}\bar{d}\bar{b} \succ \bar{a}\bar{c}d\bar{b}$
- *Figure 6.3(a):* $a\bar{b}\bar{c} \succ a\bar{b}c \succ ab\bar{c} \succ abc \succ \bar{a}\bar{b}\bar{c} \succ \bar{a}\bar{b}c \succ \bar{a}b\bar{c} \succ \bar{a}bc$
- *Figure 6.3(b):* $ab\bar{c} \succ abc \succ a\bar{b}\bar{c} \succ a\bar{b}c \succ \bar{a}\bar{b}\bar{c} \succ \bar{a}\bar{b}c \succ \bar{a}b\bar{c} \succ \bar{a}bc$
- *Figure 6.4:* $abcd \succ abc\bar{d} \succ ab\bar{c}d \succ ab\bar{c}\bar{d} \succ a\bar{b}\bar{c}d \succ a\bar{b}\bar{c}\bar{d} \succ a\bar{b}cd \succ a\bar{b}c\bar{d} \succ \bar{a}d\bar{c}\bar{b} \succ \bar{a}d\bar{c}b \succ \bar{a}d\bar{c}\bar{b} \succ \bar{a}d\bar{c}\bar{b} \succ \bar{a}dbc \succ \bar{a}db\bar{c} \succ \bar{a}d\bar{b}c \succ \bar{a}d\bar{b}\bar{c}$
- *Figure 6.8:* $ab\bar{c}\bar{d} \succ ab\bar{c}d \succ abcd \succ abc\bar{d} \succ a\bar{b}cd \succ a\bar{b}c\bar{d} \succ a\bar{b}\bar{c}\bar{d} \succ a\bar{b}\bar{c}d \succ \bar{a}d\bar{c}\bar{b} \succ \bar{a}d\bar{c}b \succ \bar{a}d\bar{c}\bar{b} \succ \bar{a}d\bar{c}b \succ \bar{a}d\bar{b}\bar{c} \succ \bar{a}d\bar{b}c \succ \bar{a}dbc \succ \bar{a}db\bar{c}$
- *Figure 6.10:* $abc \succ abc \succ ab\bar{c} \succ ab\bar{c} \succ a\bar{b}\bar{d}\bar{c} \succ a\bar{b}\bar{d}c \succ a\bar{b}dc \succ a\bar{b}d\bar{c} \succ \bar{a}cb\bar{d} \succ \bar{a}cbd \succ \bar{a}c\bar{b}d \succ \bar{a}c\bar{b}\bar{d} \succ \bar{a}\bar{c}d\bar{b} \succ \bar{a}\bar{c}\bar{d}\bar{b} \succ \bar{a}\bar{c}db \succ \bar{a}\bar{c}d\bar{b}$
- *Figure 6.11(a):* $abcd \succ abc\bar{d} \succ ab\bar{c}d \succ ab\bar{c}\bar{d} \succ a\bar{b}cd \succ a\bar{b}c\bar{d} \succ a\bar{b}\bar{c}d \succ a\bar{b}\bar{c}\bar{d} \succ \bar{a}cbd \succ \bar{a}cb\bar{d} \succ \bar{a}c\bar{b}d \succ \bar{a}c\bar{b}\bar{d} \succ \bar{a}\bar{c}bd \succ \bar{a}\bar{c}b\bar{d} \succ \bar{a}\bar{c}\bar{b}d \succ \bar{a}\bar{c}\bar{b}\bar{d}$
- *Figure 6.13:* $abcd \succ abc\bar{d} \succ ab\bar{c}d \succ ab\bar{c}\bar{d} \succ a\bar{b}dc \succ a\bar{b}d\bar{c} \succ a\bar{b}dc \succ a\bar{b}d\bar{c} \succ \bar{a}cb \succ \bar{a}cb \succ \bar{a}c\bar{b} \succ \bar{a}c\bar{b} \succ \omega_i \models \bar{a}\bar{c}$ *with* $\omega_i \in \Omega$
- *Figure 6.14:* $a\bar{b}c \succ ab\bar{c} \succ a\bar{b}\bar{c} \succ abc \succ \bar{a}\bar{b}\bar{c} \succ \bar{a}bc \succ \bar{a}\bar{b}c \succ \bar{a}b\bar{c}$

When the LP-tree is complete, it represents a total linear ordering over $\mathcal{X}$ [Booth et al., 2010]. Given an ordering $\succ_{Lex}$, each configuration can be associated with a rank that can be calculated in polynomial time [Fargier and Mengin, 2021b]. Inversely, considering a given rank searching for the configuration which corresponds to it is also done in polynomial time [Lang et al., 2012]. Recall that, generally, CP-nets find their induced order in time exponential $O(N2^N)$, whereas LP-trees entail an ordering in time linear to the size of the tree $O(N)$ [Fargier et al., 2018] .

Conditioning by multiple variables

There is nothing to prevent having an LP-tree like the one in Figure 6.15(a) where the preferences of a node depend on several important variables (we have in fact $abc \succ ab\bar{c} \succ a\bar{b}c \succ a\bar{b}\bar{c} \succ \bar{a}bc \succ \bar{a}b\bar{c} \succ \bar{a}\bar{b}c \succ \bar{a}\bar{b}\bar{c}$). Even if it seems that no example can be found in the literature, conditioning could a *priori* also depend on less important variables. But this can lead to troubling situations as in Figure 6.15(b), where B is conditioned by both A more important and C less important. If we then seek to compare $ab\bar{c}\bar{d}$ and $a\bar{b}cd$, it seems that we are led to prefer the second configuration since $c \succ \bar{c}$ and no constraint concerning the more important variable B applies (yet the value of B in $ab\bar{c}\bar{d}$ is the preferred value in the context $a\bar{c}$ and the value of B in $a\bar{b}cd$ is not its preferred value in the ac context).

Moreover, the preference on D, admittedly less important, is violated by $a\bar{b}cd$ (and one could besides add many other less important preferences below D which would be violated in the same way!)

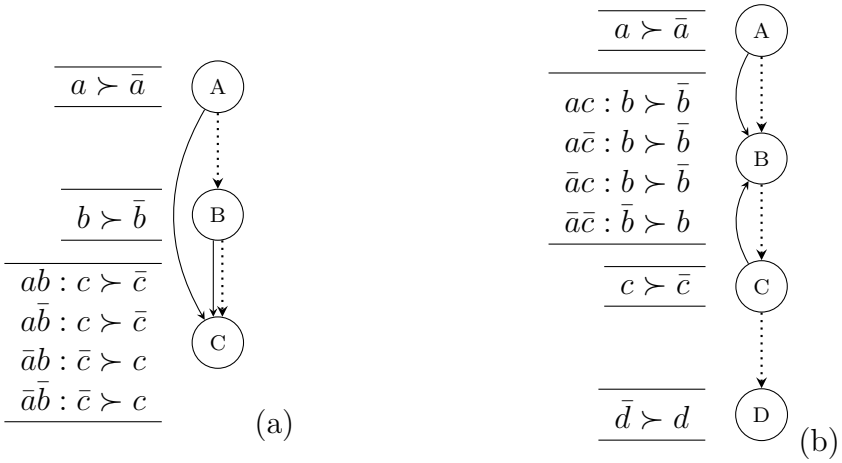


Figure 6.15: Multiple conditionings

Dominance query

To compare two configurations, it is necessary to know which variable(s) they differ on[3]. If this includes one or more variables involved in defining the branches, then the preference associated with the most important of these variables determines the order. This variable is called the *decisive* variable. If the difference in configurations involves only variables not included in the definition of the branches, then the path corresponding to their common "context" must be identified [Booth et al., 2010].

[3]In the case of a k LP tree with $k \geq 2$, we have to consider the node which contains the set of variables in which appears the variable(s) on which the two given configurations differ.

Example 6.4 *Let us consider the UI LP-tree in Figure 6.7, and the pair of configurations* abc, $a\bar{b}\bar{c}$. *The two configurations differ on the value of variables* B *and* C. *The lexicographic order expressed by the network states that* $A \triangleright B \triangleright C$. *The variable* B *constitutes thus the decisive variable. It stipulates that* $b \succ \bar{b}$ *which yields to conclude that* $abc \succ_{Lex} a\bar{b}\bar{c}$.

Lexicographic order and cardinality order

We have seen that CP-nets can violate cardinality order, while π-pref net agrees with it. It is interesting to examine the compatibility of the cardinality order with the ordering entailed from LP-trees. For this purpose, let us consider the simplest structure of UP-UI LP-trees illustrated by the Figure 6.7. The induced total order on configurations stipulates that $abc \succ_{Lex} \bar{a}\bar{b}\bar{c}$. However, the dominant configuration violates nodes B and C, while the dominated one violates only A. The two orders may thus lead contradictory dominance relations. This can be explained by the fact that the importance relation on variables has a dominant and discriminant power on the ranking of configurations.

6.4 Comparison π-pref-net vs LP-tree

In this section, we indicate several differences, more or less obvious, between π-pref-nets and LP-trees, after their respective presentations in the previous sections.

The π-pref network framework allows us to add importance relations between the preferences associated with nodes in the form of inequalities between symbolic weights, which allows us to refine the order of inclusion [Ben Amor et al., 2016b]. However, even with additional constraints on symbolic preference degrees, the order obtained from the chain rule remains partial and incomparabilities between configurations may persist.

Example 6.5 *Consider* π*-pref net in Figure 4.1(a). For example we can add* $\alpha > \beta > \sigma > \gamma$. *This expresses that it is less serious to violate the preference(s) on the variable* A *than those on the variable* B, *which are themselves of lesser priority than those on the variable* C, *and that the preferences on* D *are of greater priority than those on* C. *If we add this total order information between nodes to the* π*-pref net, we obtain an order which now makes it possible to make the comparison* $ab\bar{c}d \succ abc\bar{d}$ *(with respective preference degrees* σ *and* γ*), but which remains partial since for instance,*

one cannot compare $ab\bar{c}\bar{d}$ (with respective preference degree $\sigma\gamma$) either with $\bar{a}\bar{b}\bar{c}d$ (with respective preference degree α), or with $\bar{a}bcd$ (with respective preference degree $\alpha\beta$).

Therefore, the first main difference between LP-trees and π-pref nets is that complete LP-trees (where all variables are ordered) lead to a total ordering of configurations, while π-pref nets generally lead to only a partial order, even when adding constraints on inequalities between the symbolic weights associated with the variables, as shown in the above Example 6.5.

Besides, the π-pref nets allow the expression of the indifference between configurations, notably if we have conditional constraints such as $u : X \sim \neg X$ or if we have equalities between symbolic weights. With an LP-tree there are no tie configurations (except in the case of incomplete LP-trees, see the ordering of LP-tree in Figure 6.13 in the Example 6.3).

A more or less obvious difference between the two networks lies in their graphic structure which consists of a DAG in π-pref nets and of a directed tree in LP-trees. Moreover, for π-pref nets the structure of conditional preferences is only constrained by the structure of the DAG (more general than that of a tree).

Even if lexicographical orders, because of their simplicity, are very present in human cognition [Gigerenzer and Goldstein, 1996], the fact of always leading to a total order of the configurations (for the complete case) presents a forced character which can be all the less acceptable cognitively as there are many variables. In particular, the LP-tree idea is not compatible with the order provided by weighted averages of not very different weights. Indeed let us suppose that we have 4 Boolean variables A, B, C, D of respective weights α, β, γ, δ such that $\alpha+\beta+\gamma+\delta = 1$ with $\alpha \geq \beta \geq \gamma \geq \delta$; the lexicographic constraint $a \succ \bar{a}$ implies $\alpha \geq \beta+\gamma+\delta$, the lexicographic constraint $b \succ \bar{b}$ implies $\beta \geq \gamma+\delta$ the constraint $c \succ \bar{c}$ implies $\gamma \geq \delta$. We can therefore see that the weights must decrease rapidly. The networks are in comparison much more flexible.

6.5 From LP-trees to π-pref nets

In this section, we examine the question of building a π-pref net that recovers dominance relations induced from a complete LP-tree. To do so, the question is to know if it is possible to impose symbolic constraints in a π-pref net so that it is equivalent to an LP-tree. Let us first consider the simple case of a linear LP-tree as in Figure 6.16(a) before dealing with the general case illustrated by Figure 6.17.

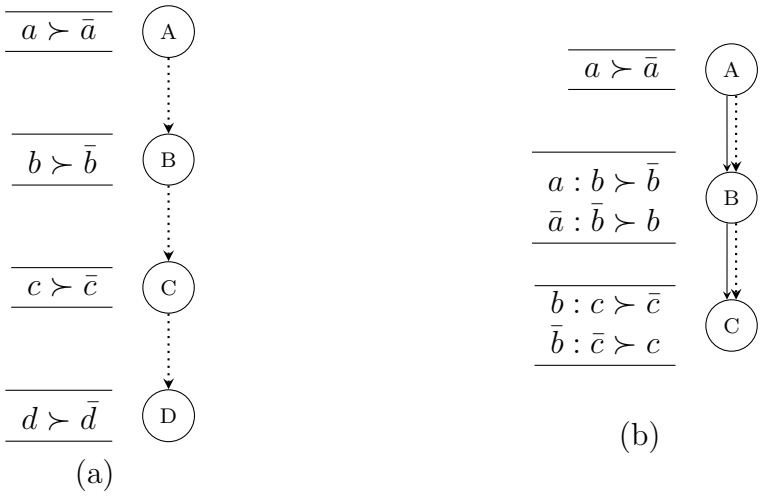


Figure 6.16: Two linear LP trees with (a) unconditional and (b) conditional preferences

For the LP-tree of Figure 6.16(a), each of the constraints $a \succ \bar{a}$, $b \succ \bar{b}$, $c \succ \bar{c}$, $d \succ \bar{d}$ is translated respectively by the possibility distributions $(1, \alpha)$, $(1, \beta)$, $(1, \gamma)$, $(1, \delta)$ on $\{a, \bar{a}\}$, $\{b, \bar{b}\}$, $\{c, \bar{c}\}$, $\{d, \bar{d}\}$ respectively. The order of importance of the variables induces the constraint $\alpha \leq \beta \leq \gamma \leq \delta$. But as can be seen by examining the table in Figure 6.19, this constraint is not sufficient to recover the lexicographical order. We must add the constraints $\alpha \leq \beta \times \gamma \times \delta$ and $\beta \leq \gamma \times \delta$ to recover it (the first constraint, for example, ensures that $a\bar{b}\bar{c}\bar{d} \succ \bar{a}bcd$). The syntax of these products shows that it is easy to generalise the constraints to be added for a linear LP-tree with any number of variables.

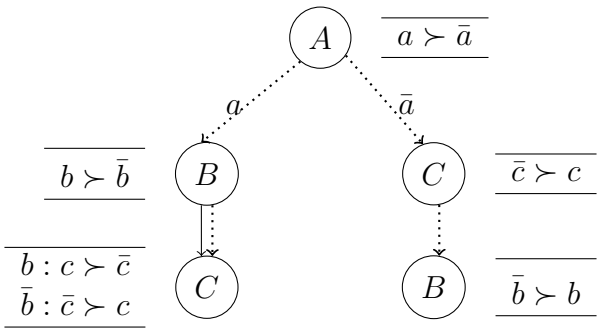


Figure 6.17: A general LP-tree

$\boldsymbol{\Omega}$	$\vec{\boldsymbol{\omega}}$
abc	$(1, 1, 1)$
$ab\bar{c}$	$(1, 1, \gamma_1)$
$a\bar{b}\bar{c}$	$(1, \beta_1, 1)$
$a\bar{b}c$	$(1, \beta_1, \gamma_2)$
$\bar{a}\bar{b}\bar{c}$	$(\alpha, 1, 1)$
$\bar{a}b\bar{c}$	$(\alpha, \beta_2, 1)$
$\bar{a}\bar{b}c$	$(\alpha, 1, \gamma_3)$
$\bar{a}bc$	$(\alpha, \beta_2\gamma_3)$

Figure 6.18: Vectors associated to configurations of π-pref net relative to LP-tree in Figure 6.17

Ω	$\vec{\omega}$
$abcd$	$(1,1,1,1)$
$abc\bar{d}$	$(1,1,1,\delta)$
$ab\bar{c}d$	$(1,1,\gamma,1)$
$ab\bar{c}\bar{d}$	$(1,1,\gamma,\delta)$
$a\bar{b}cd$	$(1,\beta,1,1)$
$a\bar{b}c\bar{d}$	$(1,\beta,1,\delta)$
$a\bar{b}\bar{c}d$	$(1,\beta,\gamma,1)$
$a\bar{b}\bar{c}\bar{d}$	$(1,\beta,\gamma,\delta)$
$\bar{a}bcd$	$(\alpha,1,1,1)$
$\bar{a}bc\bar{d}$	$(\alpha,1,1,\delta)$
$\bar{a}b\bar{c}d$	$(\alpha,1,\gamma,1)$
$\bar{a}b\bar{c}\bar{d}$	$(\alpha,1\gamma,\delta)$
$\bar{a}\bar{b}cd$	$(\alpha,\beta,1,1)$
$\bar{a}\bar{b}c\bar{d}$	$(\alpha,\beta,1,\delta)$
$\bar{a}\bar{b}\bar{c}d$	$(\alpha,\beta,\gamma,1)$
$\bar{a}\bar{b}\bar{c}\bar{d}$	$(\alpha,\beta,\gamma,\delta)$

Figure 6.19: Vectors associated to configurations for the π-pref net relative to LP-tree in Figure 6.16(a)

Let us now study a more generalized case. The LP-tree of Figure 6.17 which has for root A with preference $a \succ \bar{a}$ encoded by the distribution $(1, \alpha)$ on $\{a, \bar{a}\}$, has two branches. The one on the right (context $\bar{a}$) is coded with the two distributions $(\beta_2, 1)$, $(\gamma_3, 1)$ on $\{b, \bar{b}\}$ and on $\{c, \bar{c}\}$ respectively for the preferences $\bar{c} \succ c$ and $\bar{b} \succ b$. The constraint $\gamma_3 \leq \beta_2$ ensures the lexicographic order on this branch. As for the left branch (context a), we have on the one hand the distribution $(1, \beta_1)$ on $\{b, \bar{b}\}$ for $b \succ \bar{b}$, and on the other hand the distributions $(1, \gamma_1)$ on $\{c, \bar{c}\}$ if $B = b$ is true and $(\gamma_2, 1)$ on $\{c, \bar{c}\}$ if $B = \bar{b}$, with the importance constraints $\beta_1 \leq \min(\gamma_1, \gamma_2)$. This ensures the desired lexicographical order on this branch. The two branches correspond to the Table 6.18. To have $a\bar{b}c \succ \bar{a}\bar{b}\bar{c}$, we must add the constraint $\beta_1 \times \gamma_2 > \alpha$. As can be seen, the process of adding constraints, which is used to represent a linear LP tree, must be repeated on each branch in the case of a general LP tree.

The study of these two examples shows that it is necessary to add in a π-pref network not only constraints between symbolic weights to reflect the order of importance of nodes in a (sub)-branch, but also constraints involving products (with a number of terms that increases with the number of variables) to obtain a fully lexicographic order. This echoes the price to be paid for obtaining such an order under all circumstances.

Let us note finally that the 2-LP-tree of Figure 6.14 does not pose any particular difficulty and is easily expressed in the form of π-pref network with the possibility distributions $(1, \alpha)$ on $\{a, \bar{a}\}$, $(\gamma_1, \beta_1, 1, \sigma_1)$ on $bc, b\bar{c}, \bar{b}c, \bar{b}\bar{c}$ in context of a and $(\beta_2, \gamma_2, \sigma_2, 1)$ on $bc, b\bar{c}, \bar{b}c, \bar{b}\bar{c}$ in context of $\bar{a}$ with the constraints inequality $\alpha < \min(\beta_1, \beta_2, \sigma_1, \sigma_2, \gamma_1, \gamma_2)$, $\beta_1 > \sigma_1 > \gamma_1$ and $\beta_2 > \sigma_2 > \gamma_2$.

6.6 Conclusion

The introduction of LP-trees has been largely motivated by their use in preference learning [Liu et al., 2018] [Fargier et al., 2018]. However, if we consider that we can extrapolate a user's preferences from the observation of her/his preferences between a rather small number of configurations, making the assumption that we are trying to learn an LP-tree can constitute a rather important representation burden of having to provide importance relations, dependency relations and total order on variable domains!

LP-trees do not permit to express indifference which constitutes a biased representation and a restriction on the user preference specifications. From this point of view, the representation offered by π-pref nets is not biased (because it obeys at least the

Pareto order, which seems to be not very debatable, and can be modulated by adding constraints between the symbolic weights).

We have seen that many classes of LP-trees have not yet been well examined or even introduced in the literature. Moreover, the reverse transformation procedure for translating a π-pref net into a (potentially incomplete) LP-tree has not been investigated. However, translating a partial order into a total (pre-)order may seem arbitrary. Further studies on these of research topics need to be conducted.

Chapter 7

A Toolbox for Reasoning About Conditional Preferences

7.1 Introduction

In this chapter, we propose a possibilistic preference networks toolbox: Π**PNT**, developed in JavaScript language and designed to visualize and edit possibilistic preference networks. Π**PNT** proposes a number of features that allow to interrogate π-pref nets and compare induced ordering(s)[1] over the set of choices with respect to different other orderings such as default orderings, Pareto ordering and ceteris paribus ordering.

The implementation of Π**PNT** is based on the *CP-net visualizer*[2] toolbox dedicated to CP-nets [Shafran et al., 2016]. It contains revised code, additional functions and scripts that are produced as a byproduct of the research work in Chapters 3 to 5. The ergonomics of the window components and the graphical representation of a CP-net have been reused from the *CP-net visualizer* toolbox. Π**PNT** source code is published in the *GitHub* forge[3] .

Next section first describes how to create a π-pref net from a set of preference statements. It then explains how to construct network extensions of a π-pref net. Therefore, it details procedures of inferring orderings from the created and generated networks to then perform the dominance query on the set of solutions. Section 7.3 explains how preference specifications are translated into default rules used to entail orderings over the set of solutions. Section 7.4 shows how to compare these orderings.

[1] generated assuming all variants introduced in Chapter 3
[2] https://github.com/azsn/cp-net-visualizer
[3] https://github.com/SyrineIRIT/Possibilistic-Preference-Nets.git

Section 7.5 is dedicated to possible refinements between the Pareto and defaults orders. Finally, Section 7.6 is dedicated to conditional preference networks. It describes how to randomly generate these models using an existing library, which is an essential procedure that made our experiments possible.

The Π**PNT** toolbox proposes the following functionalities :

- Create a π-pref net and all of its extensions assuming different scales for preference degrees (see Section 3.4). This functionality includes:
  - Generate π-pref nets from a set of conditional preference statements;
  - Visualize the π-pref net.
- Compute the joint possibility distribution(s) using:
  - Equation 2.23 preference degrees;
  - Equation 3.5 when preferences are encoded by satisfaction degrees.
- Infer ordering based on:
  - the Pareto order for all π-pref nets extensions (see Section 3.4 of Chapter 3);
  - the pessimistic or optimistic approaches using Algorithms 4.1, 4.2 and 4.3;
  - the ceteris paribus assumption;
- Compare orderings of the default-like approach and the CP-net approach with the Pareto ordering
- Perform repairs and refinements on some generated orderings (see Sections 5.2 and 5.4 of Chapter 5)
- Perform dominance and optimization queries on both CP-nets and π-pref nets

The main menu of the toolbox is represented by the most-left column of Figures in 7.1.

7.2 Creating and querying π-pref nets

In this section, we explain how π-pref nets and their extensions are created, displayed and queried in the Π**PNT** toolbox.

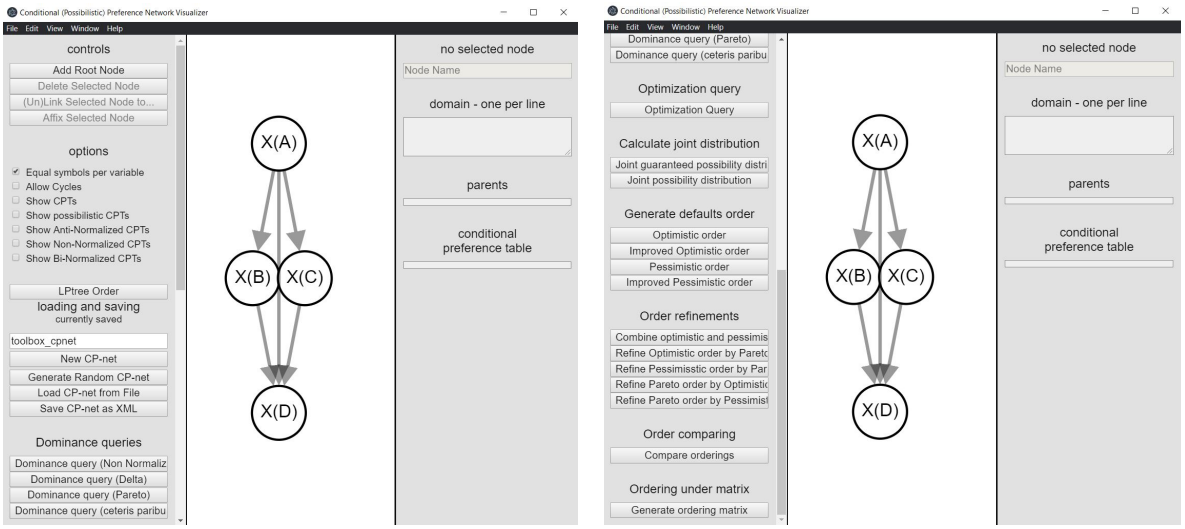


Figure 7.1: The toolbox main menu

7.2.1 Network definition

The π-pref net is defined by a structure called *Graph* which is described by a list of topologically sorted nodes, i.e., ancestors before descendants, called *Nodes* and a number of nodes denoted by N. Each node is associated with a local possibility distribution derived from statements in local tables CPTs associated to the nodes. Each node X_i is described by a set of attributes listed below. We provide the equivalent notation used in previous chapters between brackets:

- a unique name *Name* (X_i);
- an array *Domain* that saves the multiple values of the decision variable ($\underline{X_i}$);
- a set of parents *Parents* (U_{X_i});
- a set of children *Children* (Ch_{X_i});
- a bi-dimensional matrix $3 \times |U_{X_i}|$, named CPT (CPT(X_i) of Definition 1.7). Let $u \in \underline{U_{X_i}}$, each context u is allied with an array of size 3 where the first and last cells indicate the preferred and rejected values of X_i in $\underline{X_i}$, respectively. The intermediate cell indicates if the preference relation is strict (0) or loose (1);
- a bi-dimensional matrix $2 \times |U_{X_i}|$, named $PossibilisticCPT$ or πCPT for short (CPT(X_i) of Definition 3.2). For each context $u \in \underline{U_{X_i}}$ is associated an array of size 2 in reference to values of the node. The array contains degrees associated with the values of X_i.

This description allows to define nodes of the network and the dependency relations between them in addition to their associated local tables. We recall that we only consider binary decision variables and strict dominance relations between values.

Example 7.1 *Let us consider CP-net depicted in the toolbox window of Figure 7.2. First, the network structure is either uploaded from an* xml *file by clicking on the button "Load CP-net from file", randomly generated using the button "Generate random CP-net" or manually created by adding root nodes using the button "Add root node" and then dependencies between them. By clicking on each node (colored with red in figures), the user can first specify the variable name and domain in Domains in the right most column of the window. Links are then created by clicking on each node and selecting parents to add. Once the graphical structure is created, the user can now specify preferences for each context value. Conditional tables can be displayed by checking the checkbox "Show CPT's".*

Let the structure in Figure 7.1 define a conditional preference network that encodes preference specifications over 4 *decision variables* $Nodes = \{node(X(A)), node(X(B)),$ $node(X(C)),\ node(X(D))\}$. *Preferences over* $node(X(C))$ *and* $node(X(B))$ *depend from values of* $node(X(A))$. *Preferences over* $node(X(D))$ *depend from values of all other nodes. Preference specifications are:* $A \succ a$, $A : b \succ B$, $a : B \succ b$, $A : c \succ C$, $a : C \succ c$, $ABC : D \succ d$, $ABc : d \succ D$, $AbC : d \succ D$, $Abc : D \succ d$, $aBC : D \succ d$, $aBc : D \succ d$, $abC : d \succ D$, $abc : d \succ D$. *Figure 7.2 presents the displayed window of the toolbox that depicts the described conditional preference network. Attributes that describe each node are given by Columns* 1 *to* 5 *of Table 7.1.*

For each node, preference statements in CPT are considered to generate a possibilistic local distribution stored in the attribute πCPT.

Generating a π-pref net from a set of preference statements is explained in Section 3.2. Briefly, the preferred value is attributed a degree equal to 1 while its negation is allied with a positive symbolic degree strictly inferior to 1. The user can constraint symbols to be equal for all parents values or specific to each context. Algorithm 7.1 implements this procedure of creating a general normalized π-pref net (see Definition 3.2), which we call $\pi Graph$, from a statements of the corresponding CP-net structure called $Graph$. We define the algorithm's used functions in the following:

- function $ListOfSymbols$ returns a list of symbolic degrees;
- function $Pref\_Rejected\_Values$ takes as input a statement in the CPT table and returns the preferred *good* and rejected *bad* node values respectively;

- function *AddAttribute* adds the second parameter as an attribute to the object in the first parameter.

Nodes	**Name**	**Domain**	**Parents**	**Children**	**CPT**	**πCPT**
node(X(A))	X(A)	$[A, a]$	$[\emptyset]$	[node(X(B)), node(X(C)), node(X(D))]	$[A, 0, a]$	$[1, \alpha]$
node(X(B))	X(B)	$[B, b]$	[node(X(A))]	[node(X(D))]	$A : [b, 0, B]$ $a : [B, 0, b]$	$[\beta, 1]$ $[1, \beta]$
node(X(C))	X(C)	$[C, c]$	[node(X(A))]	[node(X(D))]	$A : [c, 0, C]$ $a : [C, 0, c]$	$A : [\gamma, 1]$ $a : [1, \gamma]$
node(X(D))	X(D)	$[D, d]$	[node(A), node(B), node(C)]	$[\emptyset]$	$ABC : [D, 0, d]$ $ABc : [d, 0, D]$ $AbC : [d, 0, D]$ $Abc : [D, 0, d]$ $aBC : [D, 0, d]$ $aBc : [D, 0, d]$ $abC : [d, 0, D]$ $abc : [d, 0, D]$	$ABC : [1, \sigma]$ $ABc : [\sigma, 1]$ $AbC : [\sigma, 1]$ $Abc : [1, \sigma]$ $aBC : [1, \sigma]$ $aBc : [1, \sigma]$ $abC : [\sigma, 1]$ $abc : [\sigma, 1]$

Table 7.1: Attributes describing the conditional preference network in Figure 7.2 (from column 1 to 5) and π-pref net in Figure 7.3 (all columns)

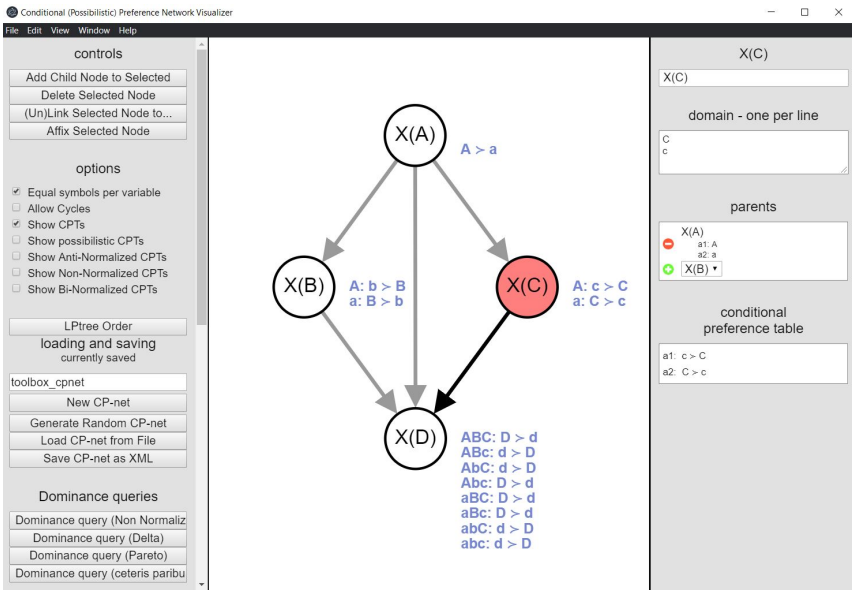


Figure 7.2: Example of a conditional preference network

Algorithm 7.1 starts by considering the root node. Function $Pref\_Rejected\_Values$ return the preferred *good* and rejected *bad* values of the node. The *good* value is associated with a degree 1 and the bad one with a symbolic degree from the list $ListOfSymbols$. A possibilistic local table πCPT is created in which preference degrees are added. Similarly, the algorithm goes through all remaining CPTs of

(non-root) nodes and, for each node, associates to the parameter *row* the preference specification of a given context value. The conditional preference *row* is then passed as parameter to the function $Pref\_Rejected\_Values$ and the same instructions as for root nodes are applied.

Note that, the user can choose to create a π-pref net that contains equal or different symbolic weight(s) per variable and context, the list $ListOfSymbols$ is thus adapted to take into consideration the choice of the user.

Algorithm 7.1: CREATE A GENERAL π-PREF NET

```
   Input: Graph                                              /* CP-net */
   Output: πGraph                                        /* π-pref net */
 1 begin
 2     symbols = ListOfSymbols()
 3     πGraph = Graph
 4     for i = 0 to Graph.N − 1 do
 5         j = 0
 6         pref_degree = [ ]
 7         if Graph.Nodes[i].Parents = ∅ then
 8             [good, bad] = Pref_Rejected_Values(Graph.Nodes[i].CPT)
 9             if Graph.Nodes[i].CPT[0] = good then
10                 pref_degree[0] = 1
11                 pref_degree[1] = symbols[i]
12             else if Graph.Nodes[i].CPT[0] = bad then
13                 pref_degree[0] = symbols[i]
14                 pref_degree[1] = 1
15         else
16             pref_degree[j] = [ ]
17             foreach row ∈ Graph.Nodes[i].CPT do
18                 [good, bad] = Pref_Rejected_Values(row)
19                 if Graph.Nodes[i].CPT[0] = good then
20                     pref_degree[j][0] = 1
21                     pref_degree[j][1] = symbols[i]
22                 else if Graph.Nodes[i].CPT[0] = bad then
23                     pref_degree[j][0] = symbols[i]
24                     pref_degree[j][1] = 1
25                 j = j + 1
26             end
27         Delete (πGraph.Nodes[i].CPT)
28         AddAttribute(πGraph.Nodes[i],PossibilisticCPT)
29         πGraph.Nodes[i].PossibilisticCPT = pref_degree
30     end
31     Return πGraph
32 end
```

Example 7.2 *Let us reconsider the network in Figure 7.2. By clicking on one of the four last checkboxes in the* option *panel in the left most column of the window, the toolbox creates and displays local tables associated to (extensions) of the π-pref net. For instance, the last column in Table 7.1 gives the added conditional tables to the CP-net structure in order to construct an normalized π-pref net: specifications of each node are translated into a possibility distribution. Besides, the user can choose to work with equal or different symbolic degrees per variable by checking the checkbox "Equal symbols per variable".*

Figures 7.3 and 7.4 represent induced π-pref nets given sets of preference statements of network in Figure 7.2, respectively with equal and different symbolic weights per variable and context.

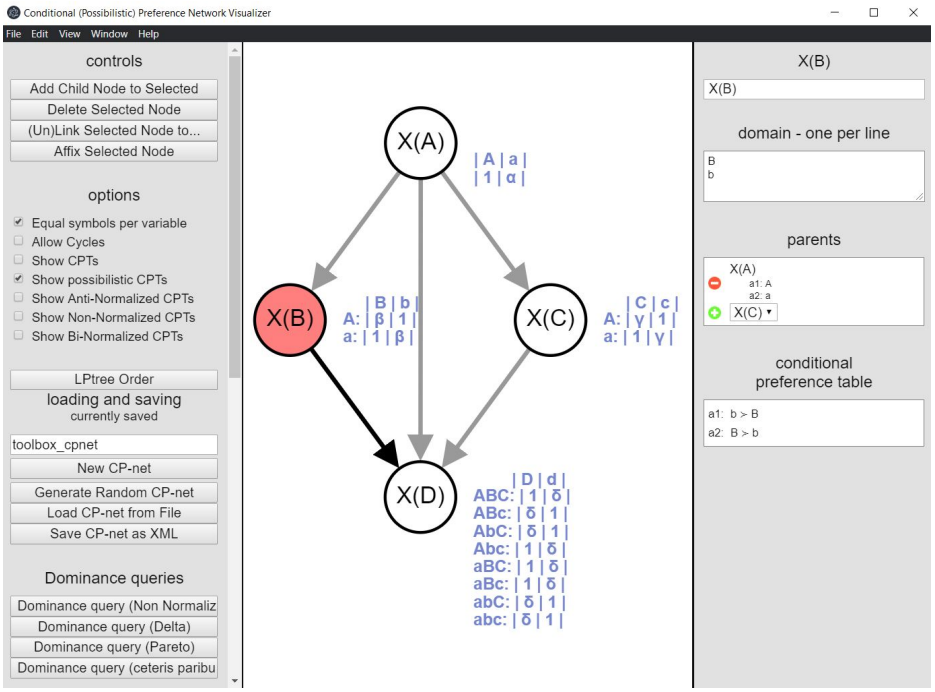


Figure 7.3: Example of a π-pref net with one symbolic weight per variable

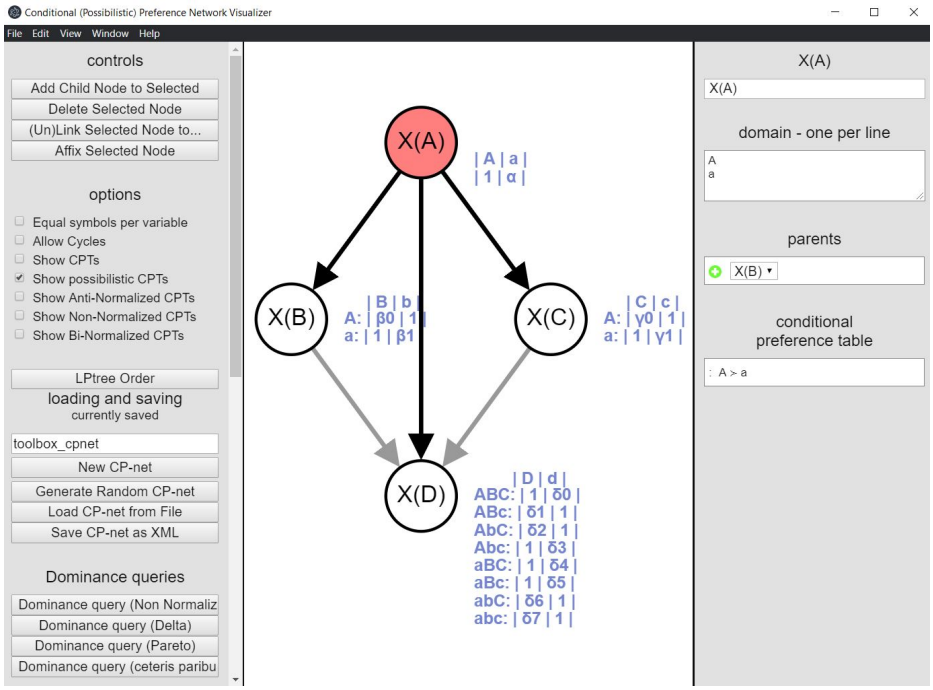


Figure 7.4: Example of a π-pref net with one symbolic weight per variable and context value

7.2.2 π-pref net extensions

In Section 3.4, we have proposed other encoding scales to express conditional preferences which gave rise to various π-pref net extensions. The example of the previous section was is an illustration of a general π-pref net or more precisely an normalized π-pref net (see Definition 3.2). The toolbox enables the user to generate from statements of a conditional preference network an anti-normalized π-pref net, a bi-normalized π-pref net or simply a preference network with a non-normalized distribution on conditional preferences (see Section 3.4). For each of these cases, the toolbox provides the user the freedom to choose working either with equal or different symbols for encoding preference degrees. Algorithm 7.1 is adapted and reused to generate preference networks with the different distribution scales.

Example 7.3 *Networks in Figures 7.5 (a), (b) and (c) correspond to extensions of π-pref net in Figure 7.4. Preferences of network in Figure 7.5(a) are encoded with satisfaction degrees using the guaranteed possibility measure. Those of network in Figure 7.5(b) are encoded with non normalized distributions which lay in the scale $(0, 1)$. In Figure 7.5(c) variable preferences are encoded by bi-normalized distribution such that the preferred value of the variable is associated with highest preference degree 1 and the rejected one is associated with a degree equal to 0.*

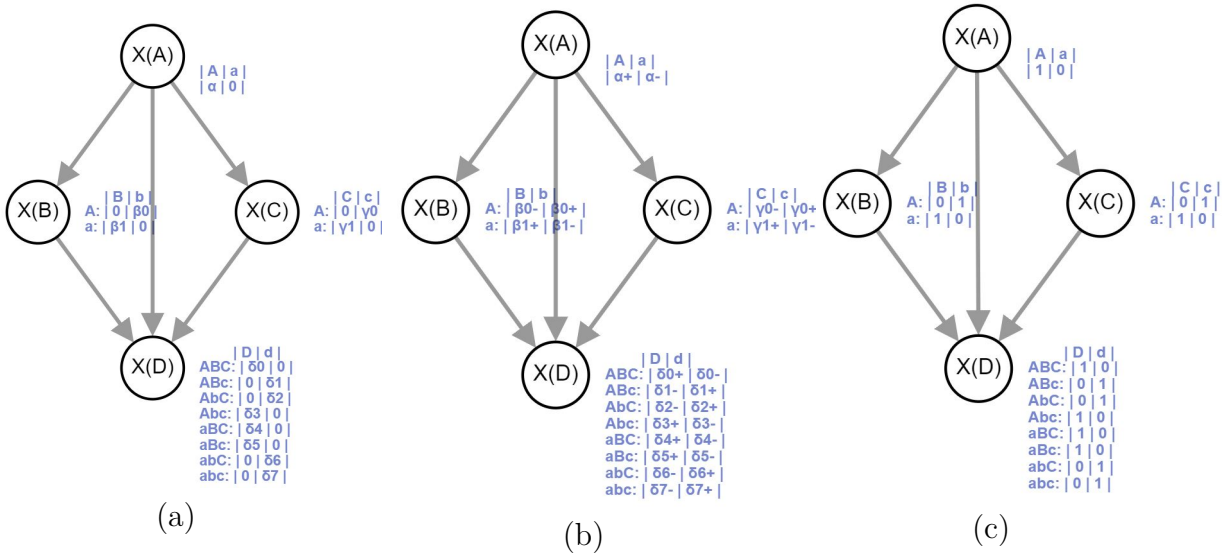


Figure 7.5: Examples of (a) an anti-normalized (guaranteed), (b) a non-normalized, (c) a bi-normalized π-pref net generated from network in Figure 7.4

7.2.3 Joint distribution over solutions

Depending on the distribution scale, the toolbox is able to calculate the joint distribution on configurations covered by a π-pref net. When preferences are encoded by possibilistic weights, the product chain rule in Equation 3.1 is used to infer an ordering over Ω. To do so, we have used Algorithm 7.2 which associates to each complete solution ω its preference weight called *degree*. The joint possibility distribution of a configuration is calculated based on its vector of weights. Algorithm 7.3 describes the procedure of constructing the vector $\vec{\omega}$ of a given solution ω. It takes as input the configuration itself ω and the π-pref net $\pi Graph$. The entered solution is then written in a Boolean form using the function $omegaUnderBooleans$. The algorithm then simply sweeps through all configuration values in ω and searches for their corresponding symbolic weights, to then output the vector of weights $\vec{\omega}$. Function $ProductChainRule$ simply multiplies degrees in $\vec{\omega}$ and returns the possibility degree associated with the input vector.

Algorithm 7.2: CALCULATE JOINT POSSIBILITY DISTRIBUTION

```
   Input: Ω
1  begin
2      foreach ω ∈ Ω do
3          ω⃗ = Construct_Vector(ω)
4          degree = ProductChainRule(ω⃗)
5          AddAttribute(ω, degree)
6      end
7  end
```

Algorithm 7.3: CONSTRUCT VECTOR OF WEIGHTS (CONSTRUCT_VECTOR)

Input: ω , $\pi Graph$
Output: $\vec{\omega}$

```
1  begin
2      ω⃗ = [ ]
3      ω_b = omegaUnderBooleans(ω)
4      for i = 0 to πGraph.N − 1 do
5          if πGraph.Nodes[i].Parents = ∅ then
6              degree = πGraph.Nodes[i].PossibilisticCPT[ω_b[i]]
7          else if Graph.Nodes[i].Parents ≠ ∅ then
8              c_ω = GetContext(ω, Graph.Nodes[i])
9              row_{c_ω} = findCPTContextRow(πGraph.Nodes[i].PossibilisticCPT, c_ω)
10             degree = πGraph.Nodes[i].PossibilisticCPT[row_{c_ω}][ω_b[i]]
11         ω⃗ = Concatenate(ω⃗, degree )
12     end
13     Return ω⃗
14 end
```

Example 7.4 *Let us consider the configuration $\omega_2 = ABcD$ and π-pref net in Figure 7.4.*

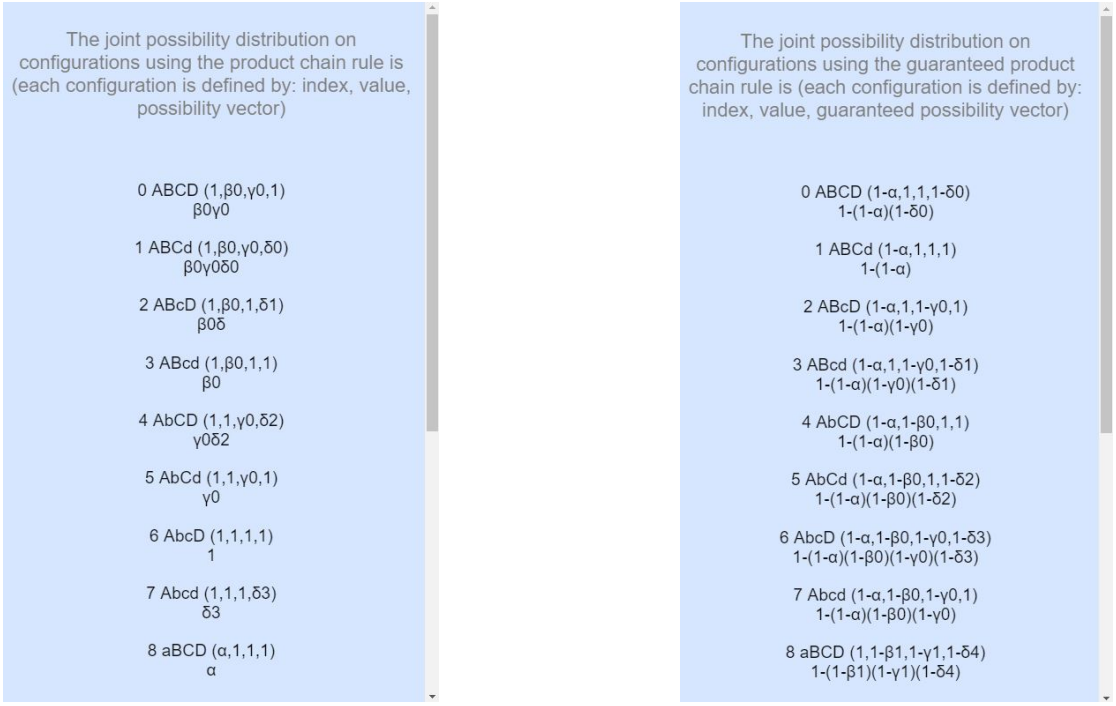
- *In the first iteration, function omegaUnderBooleans returns $\omega_{2_b} = 0010$. The possibilistic local table attached to node$(X(A))$ is [1,α]. The projection of ω_{2_b} on the possibilistic conditional table of node$(X(A))$ corresponds to the degree of ω_2[node$(X(A))$]. After this first iteration degree $= 1$ and vector $\vec{\omega_2} = [1]$.*

- *For the second iteration, the context $c_{\omega_2} = A$ and the possibility distribution of node$(X(B))$ in the context of A is $row_{c_{\omega_2}} = [\beta_0, 1]$. Since $\omega_{2_b}[1] = 0$, the preference weight added to $\vec{\omega_2}$ is degree $= \beta_0$ which yields $\vec{\omega_2} = [1, \beta_0]$.*

- *In a third iteration, we get $c_{\omega_2} = A$, $row_{c_{\omega_2}} = [\gamma_0, 1]$ and $\omega_{2_b}[2] = 1$. The vector of weights is thus equal to $\vec{\omega_2} = [1, \beta_0, 1]$. Finally, for the last vertex node$(X(D))$, the context is $c_{\omega_2} = ABc$ and the possibilistic conditional table is $row_{c_{\omega_2}} = [\delta_1, 1]$. The binary value of node$(X(D))$ is $\omega_{2_b}[3] = 0$. The returned output vector describing ω_2 is $\vec{\omega_2} = [1, \beta_0, 1, \delta_1]$.*

Figure 7.6(a) gives an example of the window displayed by the toolbox when calculating the joint possibility distribution inferred from π-pref net in Figure 7.4. Each configuration ω is described by its index in Ω, its complete instantiation and its vector $\vec{\omega}$. The joint preference degree is then displayed in a second line.

Algorithms for calculating the joint distribution on configurations when preferences



(a) (b)

Figure 7.6: Windows displaying the joint possibility distribution given (a) π-pref net in Figure 7.4 and (b) π-pref net in Figure 7.5(a)

are expressed in terms of satisfaction degrees are also implemented (see Figure 7.6(b) for illustration).

We have proposed and coded one of many solutions that permit to calculate the joint distribution on solutions.

7.2.4 Dominance query

As noted above, the product chain rule induces the exact same ordering as the Pareto strategy without additional constraints on symbolic weights [Ben Amor et al., 2015]. Thus, the toolbox implements the latter strategy (see Definition 3.4) in order to compare pairs of configurations. Algorithm 7.4 details the comparison procedure. It describes the function $Pref(M, \omega, \omega')$ used in Algorithm 5.1.

To find the dominance relation $\succ_{Pareto}$ between the entered solutions, the algorithm uses a result vector vec in the size of decision nodes. Configurations preference degrees are compared one by one. If $\pi(\omega[i]) > \pi(\omega'[i])$ then $vec[i] = \succ$. Conversely, if $\pi(\omega'[i]) > \pi(\omega[i])$ then $vec[i] = \prec$. When both degrees correspond to equal symbolic weights, then $vec[i] =$ '$\simeq$' otherwise they are considered incomparable and $vec[i] =$ '$\bowtie$'. If vec contains either both $\succ$ and $\prec$ relations or the incomparability relation $\bowtie$ then ω and

ω' can not be compared by the Pareto order. Otherwise, if *vec* contains the relation $\succ$ but not $\prec$ than $\omega \succ_{Pareto} \omega'$. By contrast, if *vec* contains the relation $\prec$ but not $\succ$ than $\omega' \succ_{Pareto} \omega$.

Similarly, the toolbox implements the Pareto order for comparing vectors of weights corresponding to all extensions of a π-pref net.

Algorithm 7.4: DOMINANCE QUERY BASED ON PARETO ORDER

Input: ω, ω', $\pi Graph$
Output: $\succ_{Pareto}$

```
1  begin
2      ω⃗ = Construct_Vector(ω)
3      ω⃗' = Construct_Vector(ω')
4      vec = [ ]
5      for i = 0 to πGraph.N − 1 do
6          if (ω⃗[i] = 1) and !(ω⃗'[i] = 1) then
7              vec[i] =≻
8              else if !(ω⃗[i] = 1) and (ω⃗'[i] = 1) then
9                  vec[i] =≺
10                 else if !(ω⃗[i] or ω⃗'[i] = 1) then
11                     if ω⃗[i] = ω⃗'[i] then
12                         vec[i] =≃
13                     else
14                         vec[i] =⋈
15     end
16     if vec.includes(⋈) or (vec.includes(≻) and vec.includes(≺)) then
17         ≻Pareto=⋈
18         else if vec.includes(≻) and !vec.includes(≺) then
19             ≻Pareto=≻
20             else if !vec.includes(≻) and vec.includes(≺) then
21                 ≻Pareto=≺
22     Return ≻Pareto
23 end
```

Example 7.5 *We pursue with the same π-pref net example depicted by Figure 7.4. The comparison vector vec of configurations $\omega_0 = ABCD$ with $\vec{\omega_0} = (1, \beta_0, \gamma_0, 1)$ and $\omega_{15} = abcd$ with $\vec{\omega_{15}} = (\alpha, \beta_1, \gamma_1, 1)$ is $vec = [\succ, \bowtie, \bowtie, \simeq]$. Algorithm 7.4 deduces an incomparability relation. For a second illustration, let us compare configurations ω_0 and ω_3. Their associated vectors of weights are respectively $\vec{\omega_0} = (1, \beta_0, \gamma_0, 1)$ and $\vec{\omega_3} = (1, \beta_0, 1, 1)$. The computed comparison vector is $vec = [\simeq, \simeq, \prec, \simeq]$. Thus, the algorithm deduces the dominance relation $\prec$ which means that $\omega_3 \succ_{Pareto} \omega_0$.*

7.3 Default reasoning on preferences

Preference statements of a user are translated into default constraints composed of two subsets that we call LC for the left-hand side of constraints and RC for the right-hand side (see Equations 4.1 and 4.4). Algorithm 7.5 scrolls through all CPT's of the *Graph* structure and, for each node, associates to the parameter row the preference specification of a given context value. The conditional preference row is then passed as parameter to the function $Pref\_Rejected\_Values$ which returns the preferred *good* and rejected *bad* node values respectively. Function $CartesianProduct$ takes as input a list of nodes and returns the set $Par\_Contexts$ of all their possible assignments. For each context $u \in Par\_Contexts$, configurations that verify both values *good* and u are assigned to $LC[j]$ such that j is an iterator over the number of constraints. Configurations that verify values *bad* and u are assigned to the other hand side of the constraint j namely $RC[j]$. This procedure describes how constraints used by Algorithms 4.1 and 4.2 are created.

Example 7.6 *Table 7.2 gives the output of Algorithm 7.5 computed on the conditional preference network in Figure 7.2. The toolbox log file of the set of default constraints is given in Figure 7.7. Each constraint is represented by a set of dominant configurations loaded in a first row, and a set of dominated configurations loaded in a second row. As case in point, let us consider specifications of node$(X(B))$. Function $CartesianProduct$ takes as parameter the parent list composed of node$(X(A))$ and returns its possible values, namely A and a.*

- *For the first context A, $row = [b, 0, B]$. Function $Pref\_Rejected\_Values$ outputs $good = b$ and $bad = B$. The constructed constraint is thus $LC[1] = \{\omega_4 = AbCD, \omega_5 = AbCd, \omega_6 = AbcD, \omega_7 = Abcd\}$ (all models of $A \wedge b$) and $RC[1] = \{\omega_0 = ABCD, \omega_1 = ABCd, \omega_2 = ABcD, \omega_3 = ABcd\}$ (all models of $A \wedge B$).*
- *For the second context a, $row = [B, 0, b]$. The constructed constraint is thus $LC[2] = \{\omega_8 = aBCD, \omega_9 = aBCd, \omega_{10} = aBcD, \omega_{11} = aBcd\}$ and $RC[2] = \{\omega_{12} = abCD, \omega_{13} = abCd, \omega_{14} = abcD, \omega_{15} = abcd\}$.*

Algorithm 7.5: GENERATE DEFAULT CONSTRAINTS

Input: Ω, $Graph$
Output: LC, RC

```
1  begin
2      k = 0
3      for i = 0 to Graph.N − 1 do
4          if Graph.Nodes[i].Parents = ∅ then
5              row = Graph.Nodes[i].CPT
6              [good, bad] = Pref_Rejected_Values(row)
7              foreach ω ∈ Ω do
8                  if ω[i] = good then
9                      LC[j].Add(ω)
10                 else if ω[i] = bad then
11                     RC[j].Add(ω)
12             end
13             j = j + 1
14         else
15             Par_Contexts = CartesianProduct(Graph.Nodes[i].Parents)  /* all
                parents values */
16             foreach u ∈ Par_Contexts do
17                 row = findCPTContextRow(Graph.Node[i].CPT, u)
18                 [good, bad] = Pref_Rejected_Values(Graph.Nodes[i].CPT[row])
19                 foreach ω ∈ Ω do
20                     if ω[i] = good and ω.includes(u) then
21                         LC[j].Add(ω)
22                     else if ω[i] = bad and ω.includes(u) then
23                         RC[j].Add(ω)
24                 end
25                 j = j + 1
26             end
27     end
28     Return LC, RC
29 end
```

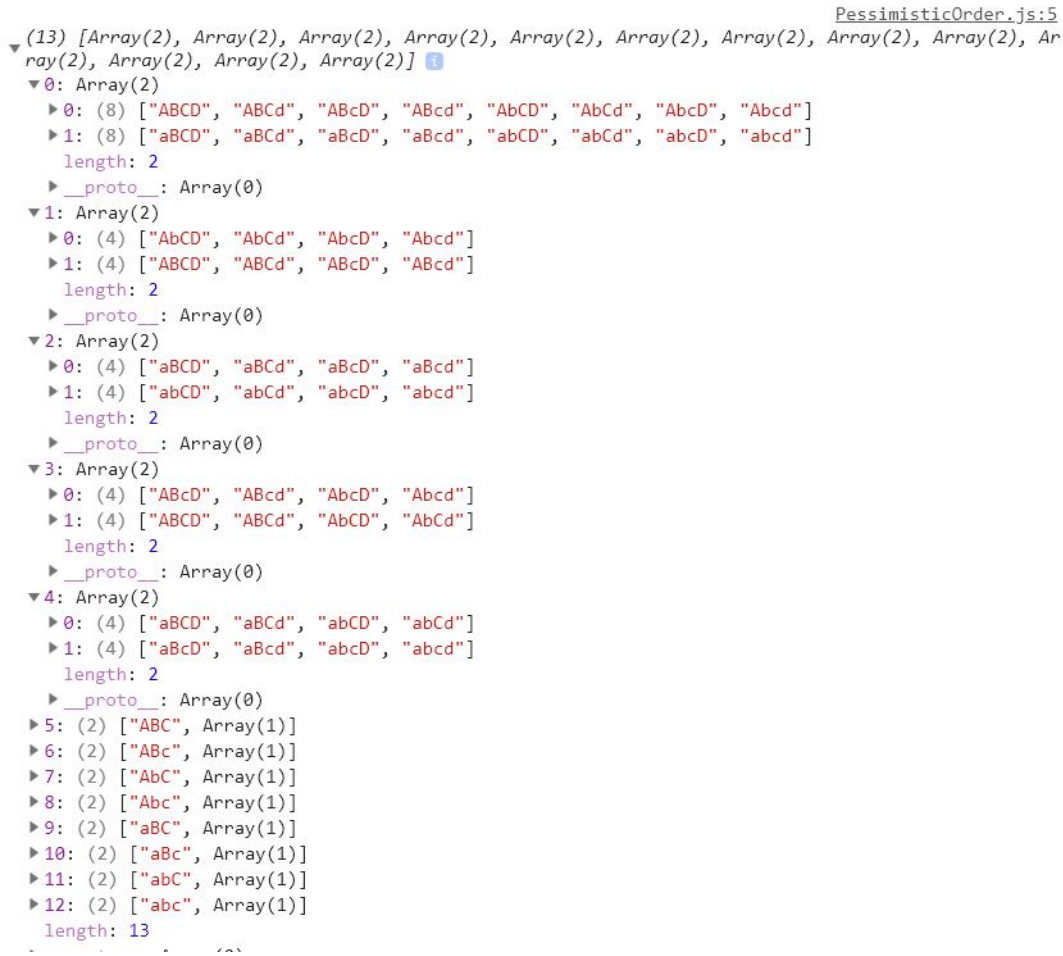


Figure 7.7: Log file of default constraints relative to preference statements of the conditional preference network in Figure 7.2

index j	$LC[j]$	$RC[j]$
0	$\{\omega_0, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6, \omega_7\}$	$\{\omega_8, \omega_9, \omega_{10}, \omega_{11}, \omega_{12}, \omega_{13}, \omega_{14}, \omega_{15}\}$
1	$\{\omega_4, \omega_5, \omega_6, \omega_7\}$	$\{\omega_0, \omega_1, \omega_2, \omega_3\}$
2	$\{\omega_8, \omega_9, \omega_{10}, \omega_{11}\}$	$\{\omega_{12}, \omega_{13}, \omega_{14}, \omega_{15}\}$
3	$\{\omega_2, \omega_3, \omega_6, \omega_7\}$	$\{\omega_0, \omega_1, \omega_4, \omega_5\}$
4	$\{\omega_8, \omega_9, \omega_{12}, \omega_{13}\}$	$\{\omega_{10}, \omega_{11}, \omega_{14}, \omega_{15}\}$
5	$\{\omega_0\}$	$\{\omega_1\}$
6	$\{\omega_3\}$	$\{\omega_2\}$
7	$\{\omega_5\}$	$\{\omega_4\}$
8	$\{\omega_6\}$	$\{\omega_7\}$
9	$\{\omega_8\}$	$\{\omega_9\}$
10	$\{\omega_{10}\}$	$\{\omega_{11}\}$
11	$\{\omega_{13}\}$	$\{\omega_{12}\}$
12	$\{\omega_{15}\}$	$\{\omega_{14}\}$

Table 7.2: Default constraints of preference statements of the conditional preference network in Figure 7.2

7.3.1 Specificity principles orders

Once Algorithm 7.5 is run and constraints are generated, Algorithm 7.6 permit to infer the default ordering on configurations based on the chosen reasoning attitude *approach*. It is basically composed of two blocks of instructions, the first one lays from line 8 to line 22 where for each right-hand side of constraint in RC, configurations that never appear are stored in the layer $E[i]$ and deleted from Ω. The second block goes from line 23 to line 34. It allows to delete constraints, both left and right side, that contain an element of $E[i]$. These blocks are repeatedly executed until no constraints are left in LC and RC or that all solutions have been ordered. This procedure is summarized by Algorithms 4.1 and 4.2.

Example 7.7 *Consider constraints sets LC and RC in Table 7.2. Let us use Algorithm 7.6 to deduce the pessimistic ordering on solutions:*

1. *in the first iteration, the configuration that never appear in any constraint in LC, i.e., for which the parameter value* $= false$ *is* ω_{14}*. The first partition layer* $E[0] = \{\omega_{14}\}$*. The configuration is then deleted from* Ω*. For the second block of the procedure (line* 23 *to* 33*),* ω_{14} *appears in* $RC[0]$*,* $RC[2]$*,* $RC[4]$ *and* $RC[12]$*. These subsets are thus removed from RC along with their complementary left-hand side constraints in LC;*

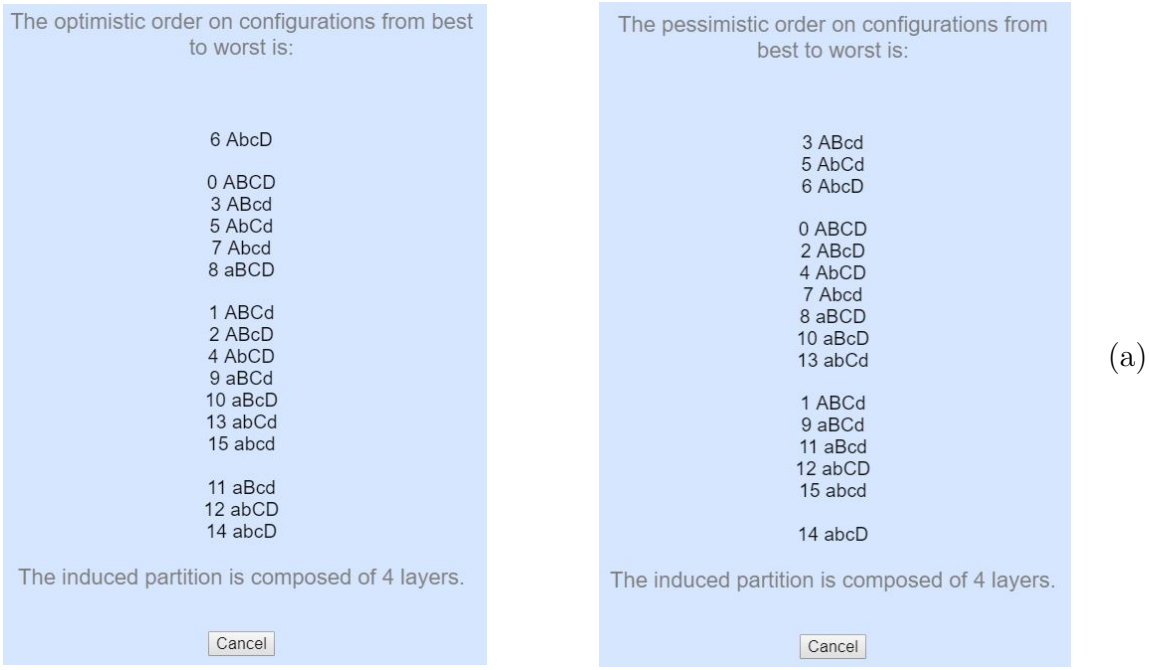
Algorithm 7.6: GENERATE WELL ORDERED PARTITION

```
   Input: LC, RC, approach
   Output: E
 1 begin
 2     if approach = 'optimistic' then
 3         side1 = RC
 4         side2 = LC
 5     else if approach = 'pessimistic' then
 6         side1 = LC
 7         side2 = RC
 8     i = 0
       /* construct partition layer                                    */
 9     while side1 ≠ ∅ or side2 ≠ ∅ or Ω ≠ ∅ do
10         foreach ω ∈ Ω do
11             verif = false
12             j = 0
13             while verif = false and j < len(side1) do
14                 c = side1[j]
15                 if c.includes(ω) then
16                     verif = true
17                 j = j + 1
18             end
19             if verif = false        /* ω never appears in any constraint */
20              then
21                 E[i].Add(ω)
22                 Ω.Delete(ω)
23         end
           /* remove satisfied constraints                             */
24         for i = 0 to len(side2) − 1 do
25             for j = 0 to len(E) − 1 do
26                 ω = E[j]
27                 c = side2[i]
28                 pos = find(c, ω)
29                 if pos ≠ −1                  /* ω appears in constraint c */
30                  then
31                     side1.Delete(c)
32                     side2.Delete(c)
33             end
34         end
35         i = i + 1
36     end
37     if approach = 'pessimistic' then
38         E.reverse()
39     Return E
40 end
```

2. *for the next iteration,* $E[1] = \{\omega_1, \omega_9, \omega_{11}, \omega_{12}, \omega_{15}\}$. *Satisfied constraints indexes are* 1, 3, 5, 9, 10 *and* 11*;*

3. *for the third iteration,* $E[1] = \{\omega_0, \omega_2, \omega_4, \omega_7, \omega_8, \omega_{10}, \omega_{13}\}$, *which leads to satisfy all remaining constraints and delete* $LC[6], RC[6], LC[7], RC[7], LC[8], RC[8]$. *Even-though there are no constraints left to verify, the power set* Ω *still contains un-ranked configurations. For each of them, the parameter* $verif = false$, *therefore, they are associated to the last partition layer:* $E[3] = \{\omega_3, \omega_5, \omega_6\}$.

The entire ordering is reversed so that configurations are classified from best to worst. Figures 7.8(a) and (b) represent results of Algorithm 7.6 displayed by the toolbox, using an optimistic and pessimistic strategy respectively.



(a)

(a) (b)

Figure 7.8: Optimistic (a) and pessimistic (b) orderings induced from constraints in Table 7.2

7.3.2 Improved specificity principles orders

When preferences are expressed under a chosen mind of reasoning (explained in Section 4.3), the ordering process described by Algorithm 7.6 is not sensitive to the coverage of solutions by default formulas. In Section 4.5, we proposed an enhanced version of the previous procedure that ranks only solutions encompassed by the available constraints

(see Algorithm 4.3). Algorithm 7.7 details the toolbox implemented procedure. Algorithm 7.7 expresses the exact same instructions as in Algorithm 4.3 but uses notations of the toolbox.

Example 7.8 *Let us look again at default constraints in Table 7.2 and adopt an optimistic mind.*

- *At the first iteration, all configurations are covered. The configuration that never appears in RC is* ω_6*. Thus, top layers are* $E[0] = \{\omega_6\}$ *and* $E'[0] = \{\emptyset\}$*. Constraints having indexes* 0, 1, 3 *and* 8 *are verified and removed from LC and RC.*
- *For the second iteration, the configuration* ω_7 *never appears in the remaining constraints. It is thus added to* $E'[1]$*. The second partition layer of* E *is* $E[1] = \{\omega_0, \omega_3, \omega_5, \omega_8\}$*. they satisfy constraints* 2, 4, 5, 6, 7 *and* 9*.*
- *In the next iteration, the remaining constraints are* 10, 11 *and* 12*. Configurations that have not yet been ranked and that do not appear as dominated in any of the left constraints are* ω_{10}*,* ω_{13} *and* ω_{15}*. All constraints are now verified and configurations that still in* Ω *are assigned to the last layer of* E'*.*

Results of computing Algorithm 7.7 on the conditional preference network in Figure 7.2 are given by Figures 7.9(a) and (b).

7.3.3 Combine specificity orderings

Algorithm 7.8 exactly reproduces Algorithm 5.2 that combines dominance relations induced by both optimistic and pessimistic orderings but uses the appropriate notation.

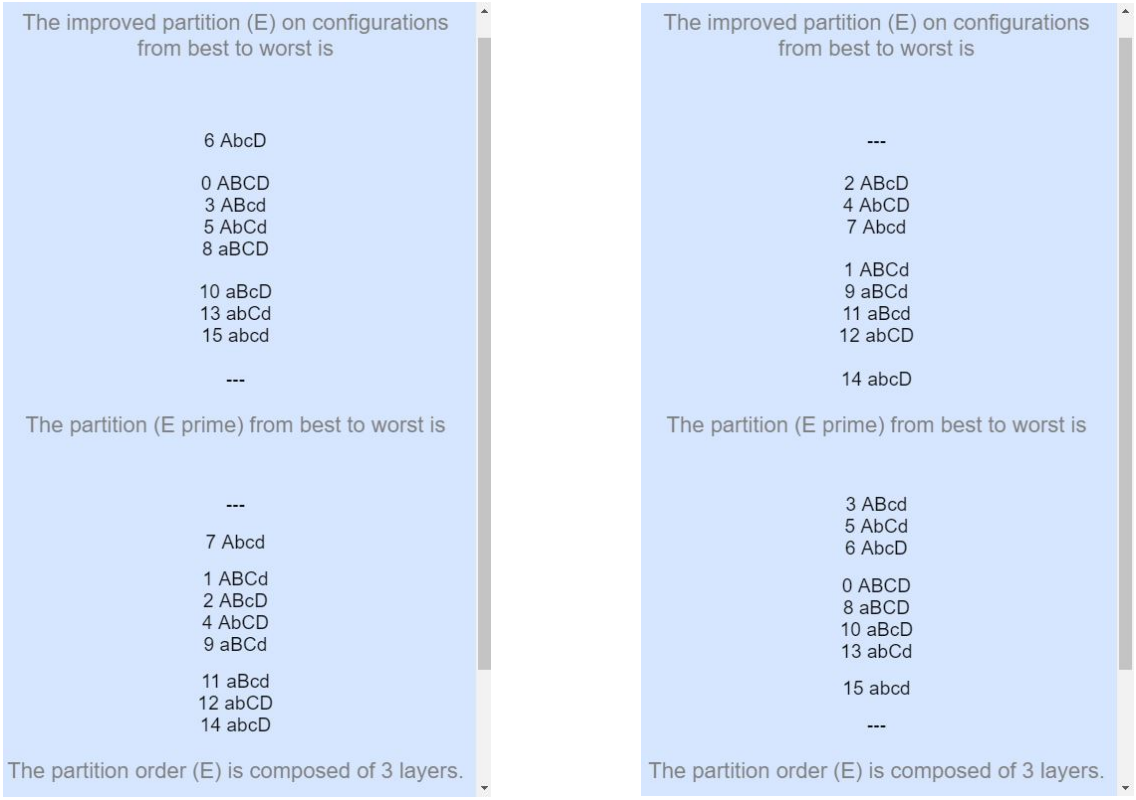
Algorithm 7.7: GENERATE IMPROVED WELL ORDERED PARTITION

```
   Input: LC, RC, approach
   Output: E, E'
 1 begin
 2     if approach = 'optimistic' then
 3         side1 = RC
 4         side2 = LC
 5     else if approach = 'pessimistic' then
 6         side1 = LC
 7         side2 = RC
 8     i = 0
 9     nb_constraints = len(LC) (or nb_constraints = len(RC))
10     while Ω ≠ ∅ or LC ≠ ∅ and RC ≠ ∅ do
           /* remove uncovered configurations                    */
11         foreach ω ∈ Ω do
12             uncovered = true
13             j = 0
14             while j < nb_constraints and uncovered = true do
15                 if LC[j].includes(ω) or RC[j].includes(ω) then
16                     uncovered = false
17                 j = j + 1
18             end
19             if uncovered = true then
20                 E'[i].Add(ω)
21                 Ω.Delete(ω)
22         end
           /* construct partition layer                          */
23         foreach ω ∈ Ω do
24             foreach c ∈ RC do
25                 if !c.includes(ω) then
26                     E[i].Add(ω)
27                     Ω.Delete(ω)
28             end
29         end
           /* remove satisfied constraints                       */
30         foreach c ∈ LC do
31             foreach ω ∈ E[i] do
32                 if c.includes(ω) then
33                     LC.Delete(c)
34                     RC.Delete(c)
35             end
36         end
37         i = i + 1
38     end
39     Return E, E'
40 end
```

Algorithm 7.8: ORDERING Ω USING BOTH OPTIMISTIC AND PESSIMISTIC ORDERS

Input: Ω , E_Π, E_Δ
Output: E

begin
  $len(E) = len(\Omega)$
  $E[0] = E_\Pi[0]$
  $E[len(\Omega) - 1] = E_\Delta[len(E_\Delta - 1)]$
  $i = 0$
  $j = len(E_\Pi) - 2$
  $index\_up = 1$
  $index\_low = len(\Omega) - 2$
  $index\_opt = len(E_\Pi) - 2$
  $index\_pess = 0$
  while $\Omega \neq \emptyset$ ***and*** $i \leq len(E_\Pi)$ ***and*** $j \leq len(E_\Delta)$
    foreach $\omega \in E_\Pi[i]$ **do**
      if $\omega \in \Omega$ **then** $E[index\_up].Add(\omega)$
    end
    $p = 0$
    foreach $\omega \in E[index\_up]$ **do**
      if $\omega \notin E_\Delta[index\_pess]$ **and** $\omega \in \Omega$ **then**
        $E[index\_up + 1].Add(\omega)$
        $p = 1$
        $E[index\_up].Delete(\omega)$
      $\Omega.Delete(\omega)$
    end
    $i = i + 1$
    $index\_up = index\_up + p + 1$
    $index\_pess = index\_pess + 1$
    if $i \neq j$ **or** $index\_opt \neq index\_pess$ **then**
      $E[index\_low] = E_\Delta[j]$
      $p = 0$
      foreach $\omega \in E[index\_low]$
        if $\omega \in E_\Pi[index\_opt]$ **and** $\omega \in \Omega$ **then**
          $E[index\_low - 1].Add(\omega)$
          $p = -1$
          $E[index\_low].Delete(\omega)$
        $\Omega.Delete(\omega)$
    $j = j - 1$
    $index\_low = index\_low + p - 1$
    $index\_opt = index\_opt - 1$
  $E = clean(E)$
  Return E
end



(a) (b)

Figure 7.9: Optimistic (a) and pessimistic (b) orderings induced from constraints in Table 7.2 using the improved partitioning algorithm version

Example 7.9 *Figure 7.10 shows the result of computing Algorithm 7.8 taking as input the optimistic and pessimistic orderings in Figures 7.8(a) and (b). At the beginning of the procedure, the output ordering E is initialized with $2^N = 16$ empty layers. The best configuration that composes the first layer of the optimistic ordering is assigned to the first layer of E. By analogy, we do the same with the worst configuration. Therefore, $E[0] = \{\omega_6\}$ and $E[15] = \{\omega_{14}\}$. Executing line 12 of the algorithm gives $E[1] = \{\omega_0, \omega_3, \omega_5, \omega_7, \omega_8\}$. By comparing $E[1]$ with $E_\Delta[0]$, we deduce that actually ω_3 and ω_5 are more satisfactory than the remaining configurations of $E[1]$, thus we only retain ω_3 and ω_5 in $E[1]$ and put the remaining solutions in a lower level, namely $E[2] = \{\omega_0, \omega_7, \omega_8\}$. Instructions from line 12 to 20 execute this sequence of steps. We now copy configurations in $E_\Delta[2]$ in $E[14]$. The optimistic order imposes to configurations ω_{11} and ω_{12} to be less preferred than all the remaining in $E_\Delta[2]$. Thus, $E[14] = \{\omega_{11}, \omega_{12}\}$ and $E[13] = \{\omega_1, \omega_9, \omega_{15}\}$. These instructions are executed by the block of code going from line 25 to 33. For the next iteration, we move to fill layer $E[2]$ with solutions in $E_\Pi[2]$. The pessimistic ordering agrees to rank order ω_2, ω_4, ω_{10} and ω_{13} which are the last remaining configurations in Ω. Hence, The output ordering of the algorithm 7.8 is $E = \{\{\omega_6\}, \{\omega_3, \omega_5\}, \{\omega_0, \omega_7, \omega_8\}, \{\omega_2, \omega_4, \omega_{10}, \omega_{13}\}, \{\omega_1, \omega_9, \omega_{15}\}, \{\omega_{11},$*

$\omega_{12}\}, \{\omega_{14}\}\}$.

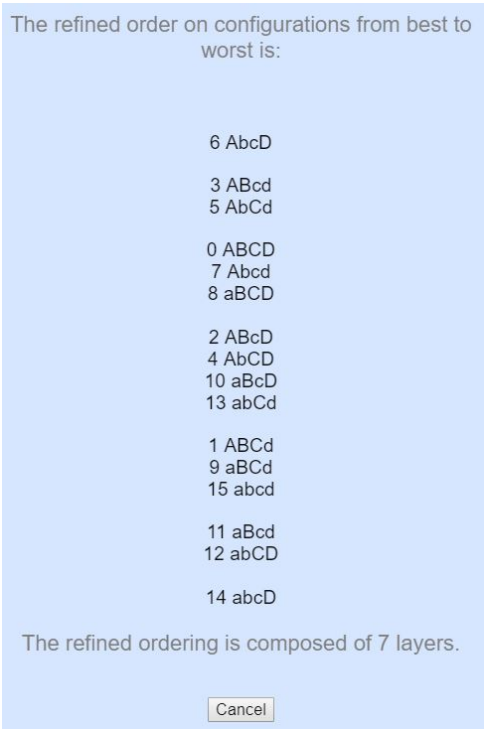


Figure 7.10: Ordering induced from combining the optimistic and pessimistic orderings of statements of in Figure 7.2

7.4 Comparing default orderings

The toolbox gives the user the opportunity to compare all previously mentioned orderings in this chapter. He may select 2 out of 7 order strategies, namely *Pareto*, *Ceteris paribus*, *Optimistic approach*, *Pessimistic approach*, *Improved optimistic approach*, *Improved pessimistic approach*, *Combined specificty approaches*, which makes a total of 21 pairs of order strategies. Figure 7.11 represents the displayed window for choosing orderings to examine.

An ordering can be represented by means of a square matrix M such that:

- if $\Omega[i] \succ \Omega[j]$ then $M[i][j] = 1$ and $M[j][i] = 0$,
- if $\Omega[i] \prec \Omega[j]$ then $M[i][j] = 0$ and $M[j][i] = 1$,
- if $\Omega[i] \simeq \Omega[j]$ then $M[i][j] = 1$ and $M[j][i] = 1$,
- if $\Omega[i] \bowtie \Omega[j]$ then $M[i][j] = 0$ and $M[j][i] = 0$.

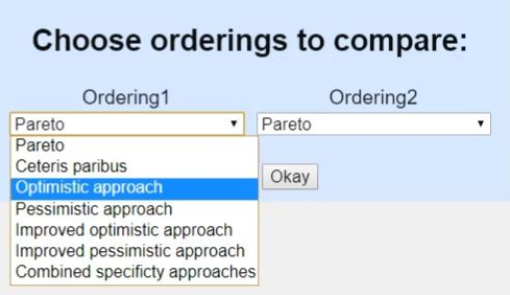


Figure 7.11: Window for comparing orderings

Algorithms 7.9, 7.10, 7.11 and 7.12 detail the procedure of constructing ordering matrices for respective orders: ceteris paribus, default orders, improved default orders and Pareto order. The user has the possibility to display any ordering matrix by the toolbox (see Figure 7.13 for an example).

Algorithm 7.9: GENERATE CETERIS PARIBUS ORDERING MATRIX

Input: Ω
Output: M

begin
  $M = InitializeMatrix()$
  for $i = 0$ **to** $len(\Omega) - 2$ **do**
    $\omega = \Omega[i]$
    $x = find(\Omega, \omega)$
    for $i = j + 1$ **to** $len(\Omega) - 1$ **do**
      $\omega' = \Omega[j]$
      $y = find(\Omega, \omega')$
      `/* function` $CPDominanceQuery$ `corresponds to the dominance query */`
      $cp\_dominance = CPDominanceQuery(\omega, \omega')$
      if $cp\_dominance =$ '$\succ$' **then**
        $M[x][y] = 1$
      else if $cp\_dominance =$ '$\prec$' **then**
        $M[y][x] = 1$
    end
  end
  Return M
end

Ones orderings are described by matrices, different metrics are calculated to compare them:

1. the number of strict dominance relations (*strict*)
2. the number of incomparability relations (*incomp*)
3. the number of equality relations (*equalities*)

Algorithm 7.10: GENERATE DEFAULT ORDERING MATRIX

```
   Input: E
   Output: M
 1 begin
 2     M = InitializeMatrix()
       /* equivalence relation between configurations of the same layer
          */
 3     for i = 0 to len(E) − 1 do
 4         for j = 0 to len(E[i]) − 1 do
 5             ω = E[i][j]
 6             x = find(Ω, ω)
 7             for k = 0 to len(E[i]) − 1 do
 8                 ω′ = E[i][k]
 9                 y = find(Ω, ω′) M[x][y] = 1
10                 M[y][x] = 1
11             end
12         end
13     end
       /* dominance relation between configurations of different layers
          */
14     for i = 0 to len(E) − 2 do
15         for j = i + 1 to len(E) − 1 do
16             foreach ω ∈ E[i] do
17                 foreach ω′ ∈ E[j] do
18                     x = find(Ω, ω)
19                     y = find(Ω, ω′) M[x][y] = 1
20                 end
21             end
22         end
23     end
24     Return M
25 end
```

Algorithm 7.11: GENERATE IMPROVED OPTIMISTIC ORDERING MATRIX

Input: E_Π, E_Δ
Output: M

```
1  begin
2      M = InitializeMatrix()
       /* equivalence relation between configurations of the same layer
          */
3      for i = 0 to len(E) − 1 do
4          for j = 0 to len(E[i]) − 1 do
5              ω = E[i][j]
6              for k = j to len(E[i]) − 1 do
7                  ω′ ∈ E[i][k]
8                  x = find(Ω, ω)
9                  y = find(Ω, ω′)
10                 M[x][y] = 1
11                 M[y][x] = 1
12             end
13         end
14     end
       /* dominance relation between configurations of different layers
          */
15     for i = 0 to len(E) − 2 do
16         for j = i + 1 to len(E) − 1 do
17             foreach ω ∈ E[i] do
18                 foreach ω′ ∈ E[j] do
19                     x = find(Ω, ω)
20                     y = find(Ω, ω′)
21                     M[x][y] = 1
22                 end
23             end
24         end
25     end
       /* dominance relation between configurations in E and those in E′
          */
26     for i = 0 to len(E) − 2 do
27         foreach ω ∈ E[i] do
28             foreach ω′ ∈ E[i + 1] do
29                 x = find(Ω, ω)
30                 y = find(Ω, ω′)
31                 M[x][y] = 1
32             end
33         end
34     end
35     Return M
36 end
```

Algorithm 7.12: GENERATE PARETO ORDERING MATRIX

Input: Ω
Output: M

```
1  begin
2      M = InitializeMatrix()
3      for i = 0 to len(Ω) − 2 do
4          ω = Ω[i]
5          x = find(Ω, ω)
6          ω⃗ = Construct_Vector(ω)
7          for j = i + 1 to len(Ω) − 1 do
8              ω′ = Ω[j]
9              y = find(Ω, ω′)
10             ω⃗′ = Construct_Vector(ω′)
               /* function ParetoDominanceQuery correspond to Algorithm
                  7.4                                                  */
11             pareto_dominance = ParetoDominanceQuery(ω⃗, ω⃗′)
12             if pareto_dominance = '≻' then
13                 M[x][y] = 1
14             else if pareto_dominance = '≺' then
15                 M[y][x] = 1
16         end
17     end
18     Return M
19 end
```

4. the number of resolved incomparability relations (*resolved_incomp*)

5. the number of resolved equalities (*resolved_eq*)

6. the number of strong violations (*contradiction*)

Computational details for calculating these metrics are described by Algorithm 7.13. An example of window result of the toolbox is given in Figure 7.12.

Algorithm 7.13: COMPARE ORDERINGS

Input: M, M'
Output: *metrics*

```
1  begin
       /* initialize all metrics to zero                                  */
2      metrics = Initiate(metrics)
3      for i = 0 to len(M) − 2 do
4          for j = i + 1 to len(M) − 1 do
5              M_dominance = Pref(M, Ω[i], Ω[j])
6              M'_dominance = Pref(M, Ω[i], Ω[j])
7              if (M_dominance = '≻' and M'_dominance = '≺') or
                (M'_dominance = '≻' and M_dominance = '≺') then
8                  contradiction = contradiction + 1
9              if M_dominance = '≻' or M_dominance = '≺' then
10                 M_strict = M_strict + 1
11             if M'_dominance = '≻' or M'_dominance = '≺' then
12                 M'_strict = M'_strict + 1
13             if M_dominance = '⋈' then
14                 M_incomp = M_incomp + 1
15                 if M'_dominance = '≻' or M'_dominance = '≺' then
16                     M'_resolved_incomp = M'_resolved_incomp + 1
17             if M'_dominance = '⋈' then
18                 M'_incomp = M'_incomp + 1
19                 if M_dominance = '≻' or M_dominance = '≺' then
20                     M_resolved_incomp = M_resolved_incomp + 1
21         end
22     end
23     Return metrics
24 end
```

7.5 Refinements and repairs between Pareto and default orderings

In Sections 5.4 and 5.2, we proposed algorithms for repairing the default ordering by the Pareto order and refining the Pareto ordering by a default ordering (see Sections

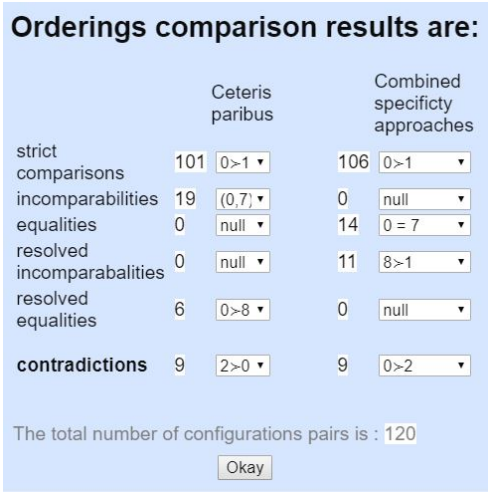


Figure 7.12: Metrics results for comparing the ceteris paribus ordering and the combined specificity approaches default ordering of the conditional preference network in Figure 7.2

5.2 and 5.4 respectively). The toolbox implements the described procedures which correspond to Algorithms 5.1 and 5.3 respectively, and enables the user to visualize the resulting ordering under the form of an incidence matrix. The user may also choose two configurations to compare based on the induced ordering. Figure 7.13 shows an example of an ordering displayed under the form of a matrix. Figures 7.14 and 7.15 show the displayed windows for performing the dominance query given a precise ordering.

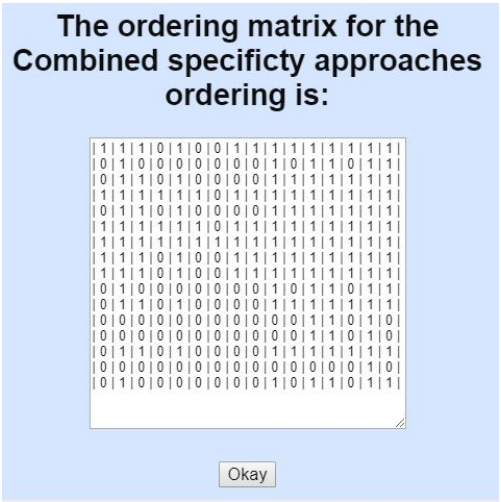


Figure 7.13: Incidence matrix of the combined specificity ordering in Figure 7.10

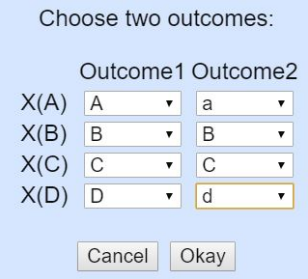


Figure 7.14: Window of dominance query

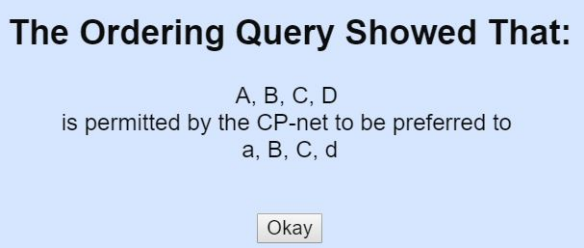


Figure 7.15: Window of an example of dominance query result

7.6 Generating π-pref nets from a conditional preference network

All of our previous experiments are conducted on a set of π-pref nets constructed from a set of randomly generated conditional preference networks. In this section, we briefly describe how these networks are generated using the *Gencpnet* library [Allen et al., 2016].

The original toolbox includes the *Gencpnet* library [Allen et al., 2016] as a network generator. It permits to generate multiple acyclic networks with respect to some specified parameters. Ones can vary the number of generated networks (m), the number of nodes (n), the maximum bound on in-degree (c), the domain size (d) and the probability of incompleteness (i). In our work, we focused on Boolean variable domain which means that the parameter d is fixed to 2. We only considered modifying parameters m and n. The generated networks are written and stored under XML format files that can afterword be uploaded by the *CP-net visualizer* and *π-pref net visualizer* toolboxes.

The *Gencpnet* is a free toolbox available in the Github forge (link). It is coded with $C++$ language and is designed to work only on GNU Linux system.

Example 7.10 *Let us consider the following command line:*

command line : ./gencpnet -n 3 -c 2 -d 2 -g 10 GeneratedExamples

It allows to generate 10 *random networks under the described specifications and store them in the folder "GeneratedExample". Parameters are fixed as follows:*

- *Number of generated CP-nets $g = 10$*

- *Number of nodes n = 3*
- *Bound on in-degree c = 2*
- *Domains size d = 2*
- *Probability of incompleteness i = 0*

The toolbox implements the optimization query described in 1.3.5, algorithm for the optimization query and gives a new implementation of the dominance query.

It has been proved that the ceteris paribus ordering is nothing less than a refinement of the Pareto ordering [Wilson et al., 2019]. From there, instead of searching for flipping sequences to perform the dominance query on a pair of configurations (see Section 12), we were interested in developing an algorithm that treats each variable value in configurations separately and returns a vector which is examined to deduce the dominance relation between them.

7.7 Conclusion

The $\pi-$**PNT** have been implemented to allow further researches on graphical structures of preferences. As a continuity of this work, further developments that implement other features should be considered. For instance, he should be granted the ability to instantiate symbolic weights with numerical degrees or to impose additional constraints on these weights so that, in a second time, the min or product-based chain rule distribution is compared with orderings of other approaches.

A particular and interesting case should be to develop a library that randomly generates π-pref nets using symbolic degrees, random constraints on them or numerically instantiated weights. The generated π-pref nets should therefore be saved and loaded under xml or csv files for future experiments. Preference structure generators should be parameterized to produce networks with specific categorized structures such as path graph, trees, multiply-connected-networks, etc.

Further developments on LP-trees would be eventually interesting. The toolbox should offer the possibility to create, visualize, question and generate LP-trees, which in a second step, would be translated into π-pref nets when possible.

Conclusion and Perspectives

A conditional preference statement is something which looks quite simple to represent and to exploit. When CP-nets have been proposed twenty years ago, we might have thought that it was a definitive approach to the problem of handling preferences stated in a concise manner.

The apparition of other settings such as cp-theories, LP-trees and π-pref nets have shown that other options were possible. The problem of answering dominance queries aiming at comparing complete configurations may receive different kinds of answers. Indeed, one may obtain partial orderings as well as complete pre-orderings. Then, this rises the question of what is the "good" answer.

On the one hand, it seems clear that the Pareto ordering that acknowledges the fact that a configuration that violates some preferences also violated by another configuration plus some other preferences is worst. Beyond that, the use of general principles like ceteris paribus, or for instance, the idea such that "a configuration is considered satisfactory unless preference statements say otherwise" seems natural and innocuous, but unfortunately this is not the case as recalled and explained in this thesis.

On the other hand, from a user point of view obtaining a complete pre-ordering or even a total ordering may seem more useful than providing a partial ordering only. However, if the complete pre-ordering has many ties or in other words a small number of layers, it is not very useful either. Generally speaking, qualitative approaches seem more in line with the qualitative specifications of conditional preference statements, even if we have seen that there may exist different ways of understanding and representing such statements.

This thesis has contributed to various advances in the discussion of the above issues. First, in the second part of Chapter 3, we have introduced several variants of π-pref nets suggesting that it may be beneficial to have a bipolar point of view and to use

both the extreme values 1 and 0 for stating full satisfaction and complete absence of satisfaction together with symbolic intermediary weights.

In Chapter 4, we have introduced the reading of conditional preference statements as default-like rule, together with the use of optimistic or pessimistic principles regarding ho to consider configurations which are not concerned by preference statements. We have shown that we obtain complete pre-orderings which might violate the Pareto order. Then, we have proposed a modified algorithm to cope with this problem, yielding a partial order. We have also shown that for a large class of graph structures the complete orderings obtained have only three layers, even if for very particular structures it is possible to have a number of layers that increases with the number of variables. Finally we have also shown that there exist systems of default rules representing preferences that cannot be represented by π-pref nets or CP-nets.

When the minimum or maximum specificity principle is used to rank-order configurations, contradictions with the Pareto order may take place. In Chapter 5, we have proposed to remedy this problem by considering three main approaches: (i) the optimistic or pessimistic ordering can be corrected by taking into consideration dominance relations induced from the Pareto order. This yields a slightly more refined ordering than the initial one; (ii) when no conflicts with the Pareto order are detected, one may take advantage of the joint use of the two specificity orderings to produce a more discriminant ordering; (iii) the Pareto partial ordering can be refined by specificity-based orderings to induce a complete pre-order that takes advantages of both principles which combines the advantages of the indisputable nature of Pareto order with an optimistic or pessimistic view.

The Chapter 6 is devoted to a detailed presentation of LP-trees and the algorithms for producing the total ordering of configurations, as we well as to their comparison with π-pref nets. It is explained that π-pref nets can encode lexicographic orderings (at the price of adding some constraints between symbolic weights). The proofs have yet to be provided. The converse transformation does not appear to be always feasible, since π-pref net encode only partial order in general (even if incomplete LP-trees provide a way to obtain partial order).

Chapter 7 presents an implemented toolbox which enables us to test the different approaches considered in the thesis on a variety of graph structures either directly specified or randomly generated.

The thesis have left a number of questions that remain unanswered, and which are topics for further researches in order to offer theoretical proofs about results of the

several proposed algorithms. The extension of the π-pref nets approaches with the bipolar view mentioned above (with symbolic weights intermediary between 0 and 1) seems also to be worth studying.

The representation of preferences held by groups of agents (possibly defined by sets of characteristic properties) and the analysis of possible conflicts of preferences between different groups is a topic of interest that can benefit from the works on the modeling of the preferences of a single agent [Ben Amor et al., 2018b] [Ben Amor et al., 2017b].

Learning the preferences of an agent, or of a group of agents, is an important issue. While there exist works on preference learning in terms of CP-nets or LP-trees, the possible use of π-pref nets, which offer a flexible and compact way of representing preferences, has not yet been considered in this perspective. The problem may be considered either by learning a π-pref net with numerical degrees (but such a network may provide more comparisons than we would reasonably expect from the qualitative specification of conditional preferences) by only identifying ordinal conditions between satisfaction degrees. These are open questions.

Bibliography

[Allen et al., 2016] Allen, T. E., Goldsmith, J., Justice, H. E., Mattei, N., and Raines, K. (2016). Generating cp-nets uniformly at random. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 872–878.

[Ben Amor et al., 2014] Ben Amor, N., Dubois, D., Gouider, H., and Prade, H. (2014). Possibilistic networks: A new setting for modeling preferences. In *Scalable Uncertainty Management - 8th International Conference, SUM 2014, Oxford, UK, September 15-17, 2014. Proceedings*, pages 1–7.

[Ben Amor et al., 2015] Ben Amor, N., Dubois, D., Gouider, H., and Prade, H. (2015). Possibilistic conditional preference networks. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty - 13th European Conference, ECSQARU 2015, Compiègne, France, July 15-17, 2015. Proceedings*, pages 36–46.

[Ben Amor et al., 2016a] Ben Amor, N., Dubois, D., Gouider, H., and Prade, H. (2016a). Graphical models for preference representation: An overview. In *Scalable Uncertainty Management - 10th International Conference, SUM 2016, Nice, France, September 21-23, 2016, Proceedings*, pages 96–111.

[Ben Amor et al., 2016b] Ben Amor, N., Dubois, D., Gouider, H., and Prade, H. (2016b). Preference modeling with possibilistic networks and symbolic weights: A theoretical study. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 1203–1211.

[Ben Amor et al., 2017a] Ben Amor, N., Dubois, D., Gouider, H., and Prade, H. (2017a). Expressivity of possibilistic preference networks with constraints. In *Scalable Uncertainty Management - 11th International Conference, SUM 2017, Granada, Spain, October 4-6, 2017, Proceedings*, pages 163–177.

[Ben Amor et al., 2017b] Ben Amor, N., Dubois, D., Gouider, H., and Prade, H. (2017b). Graphical representations of multiple agent preferences. In Benferhat, S., Tabia, K., and Ali, M., editors, *Advances in Artificial Intelligence: From Theory to Practice - 30th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2017, Arras, France, June 27-30, 2017, Proceedings, Part II*, volume 10351 of *Lecture Notes in Computer Science*, pages 142–153. Springer.

[Ben Amor et al., 2018a] Ben Amor, N., Dubois, D., Gouider, H., and Prade, H. (2018a). Possibilistic preference networks. *Inf. Sci.*, 460-461:401–415.

[Ben Amor et al., 2018b] Ben Amor, N., Dubois, D., Prade, H., and Saidi, S. (2018b). Representation of multiple agent preferences - A short survey. In *Scalable Uncertainty Management - 12th International Conference, SUM 2018, Milan, Italy, October 3-5, 2018, Proceedings*, pages 359–367.

[Ben Amor et al., 2019] Ben Amor, N., Dubois, D., Prade, H., and Saidi, S. (2019). Revisiting conditional preferences: From defaults to graphical representations. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 15th European Conference, ECSQARU 2019, Belgrade, Serbia, September 18-20, 2019, Proceedings*, pages 187–198.

[Ben Amor et al., 2021a] Ben Amor, N., Dubois, D., Prade, H., and Saidi, S. (2021a). Conditional preference networks - refining solution orderings beyond pareto dominance. In Fujita, H., Selamat, A., Lin, J. C., and Ali, M., editors, *Advances and Trends in Artificial Intelligence. Artificial Intelligence Practices - 34th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2021, Kuala Lumpur, Malaysia, July 26-29, 2021, Proceedings, Part I*, volume 12798 of *Lecture Notes in Computer Science*, pages 447–459. Springer.

[Ben Amor et al., 2021b] Ben Amor, N., Dubois, D., Prade, H., and Saidi, S. (2021b). Réseaux possibilistes de préférences et arbres de préférences lexicographiques – une comparaison. In Cépaduès, editor, *LFA 2021, Rencontres Francophones sur la Logique Floue et ses Applications, Paris, France, October 21-22, 2021*, pages 209–216.

[Ben Amor et al., 2022] Ben Amor, N., Dubois, D., Prade, H., and Saidi, S. (2022). Possibilistic preference networks and lexicographic preference trees – a comparison. In Springer, editor, *IPMU 2022, Information Processing and Management of Uncertainty in Knowledge-Based Systems, Milan, Italy, July 11-15, 2022*, pages 581–592.

[Benferhat et al., 1999a] Benferhat, S., Dubois, D., Garcia, L., and Prade, H. (1999a). Possibilistic logic bases and possibilistic graphs. In Laskey, K. B. and Prade, H., editors, *UAI '99: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, Stockholm, Sweden, July 30 - August 1, 1999*, pages 57–64. Morgan Kaufmann.

[Benferhat et al., 2002a] Benferhat, S., Dubois, D., Garcia, L., and Prade, H. (2002a). On the transformation between possibilistic logic bases and possibilistic causal networks. *Int. J. Approx. Reason.*, 29(2):135–173.

[Benferhat et al., 2000] Benferhat, S., Dubois, D., Kaci, S., and Prade, H. (2000). Encoding information fusion in possibilistic logic: A general framework for rational syntactic merging. In Horn, W., editor, *ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, August 20-25, 2000*, pages 3–7. IOS Press.

[Benferhat et al., 2001a] Benferhat, S., Dubois, D., Kaci, S., and Prade, H. (2001a). Bridging logical, comparative, and graphical possibilistic representation frameworks. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 6th European Conference, ECSQARU 2001, Toulouse, France, September 19-21, 2001, Proceedings*, pages 422–431.

[Benferhat et al., 2001b] Benferhat, S., Dubois, D., Kaci, S., and Prade, H. (2001b). Graphical readings of possibilistic logic bases. In Breese, J. S. and Koller, D., editors, *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, University of Washington, Seattle, Washington, USA, August 2-5, 2001*, pages 24–31. Morgan Kaufmann.

[Benferhat et al., 2002b] Benferhat, S., Dubois, D., Kaci, S., and Prade, H. (2002b). Bipolar possibilistic representations. In Darwiche, A. and Friedman, N., editors, *UAI '02, Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence, University of Alberta, Edmonton, Alberta, Canada, August 1-4, 2002*, pages 45–52. Morgan Kaufmann.

[Benferhat et al., 2002c] Benferhat, S., Dubois, D., Kaci, S., and Prade, H. (2002c). Possibilistic logic representation of preferences: relating prioritized goals and satisfaction levels expressions. In *Proceedings of the 15th Eureopean Conference on Artificial Intelligence, ECAI'2002, Lyon, France, July 2002*, pages 685–689.

[Benferhat et al., 2006] Benferhat, S., Dubois, D., Kaci, S., and Prade, H. (2006). Bipolar possibility theory in preference modeling: Representation, fusion and optimal solutions. *Inf. Fusion*, 7(1):135–150.

[Benferhat et al., 2008] Benferhat, S., Dubois, D., Kaci, S., and Prade, H. (2008). Modeling positive and negative information in possibility theory. *Int. J. Intell. Syst.*, 23(10):1094–1118.

[Benferhat et al., 1992] Benferhat, S., Dubois, D., and Prade, H. (1992). Representing default rules in possibilistic logic. In *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR'92). Cambridge, MA, USA, October 25-29, 1992*, pages 673–684.

[Benferhat et al., 1997] Benferhat, S., Dubois, D., and Prade, H. (1997). Nonmonotonic reasoning, conditional objects and possibility theory. *Artif. Intell.*, 92(1-2):259–276.

[Benferhat et al., 2001c] Benferhat, S., Dubois, D., and Prade, H. (2001c). Towards a possibilistic logic handling of preferences. *Appl. Intell.*, 14(3):303–317.

[Benferhat et al., 1999b] Benferhat, S., Dubois, D., Prade, H., and Williams, M. (1999b). A practical approach to fusing prioritized knowledge bases. In *Progress in Artificial Intelligence, 9th Portuguese Conference on Artificial Intelligence, EPIA '99, Évora, Portugal, September 21-24, 1999, Proceedings*, pages 223–236.

[Booth et al., 2010] Booth, R., Chevaleyre, Y., Lang, J., Mengin, J., and Sombattheera, C. (2010). Learning conditionally lexicographic preference relations. In *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, pages 269–274.

[Boutilier et al., 2001] Boutilier, C., Bacchus, F., and Brafman, R. I. (2001). Ucp-networks: A directed graphical representation of conditional utilities. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, University of Washington, Seattle, Washington, USA, August 2-5, 2001*, pages 56–64.

[Boutilier et al., 2004] Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., and Poole, D. (2004). Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *J. Artif. Intell. Res.*, 21:135–191.

[Boutilier et al., 1999] Boutilier, C., Brafman, R. I., Hoos, H. H., and Poole, D. (1999). Reasoning with conditional ceteris paribus preference statements. In *UAI '99: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, Stockholm, Sweden, July 30 - August 1, 1999*, pages 71–80.

[Brafman and Domshlak, 2002] Brafman, R. I. and Domshlak, C. (2002). Introducing variable importance tradeoffs into cp-nets. In *UAI '02, Proceedings of the 18th Con-*

ference in Uncertainty in Artificial Intelligence, University of Alberta, Edmonton, Alberta, Canada, August 1-4, 2002, pages 69–76.

[Brafman and Domshlak, 2011] Brafman, R. I. and Domshlak, C. (2011). Structure and complexity in planning with unary operators. *CoRR*, abs/1106.5256.

[Brafman et al., 2006] Brafman, R. I., Domshlak, C., and Shimony, S. E. (2006). On graphical modeling of preference and importance. *J. Artif. Intell. Res.*, 25:389–424.

[Brafman and Engel, 2009a] Brafman, R. I. and Engel, Y. (2009a). Directional decomposition of multiattribute utility functions. In *Algorithmic Decision Theory, First International Conference, ADT 2009, Venice, Italy, October 20-23, 2009. Proceedings*, pages 192–202.

[Brafman and Engel, 2009b] Brafman, R. I. and Engel, Y. (2009b). Directional decomposition of multiattribute utility functions. In Rossi, F. and Tsoukiàs, A., editors, *Algorithmic Decision Theory, First International Conference, ADT 2009, Venice, Italy, October 20-23, 2009. Proceedings*, volume 5783 of *Lecture Notes in Computer Science*, pages 192–202. Springer.

[Bräuning and Hüllermeier, 2012] Bräuning, M. and Hüllermeier, E. (2012). Learning conditional lexicographic preference trees. In *ECAI 2012 - 19th Workshop on Preference Learning of the European Conference on Artiticial Intelligence, Montpelier, France, 2012, Proceedings*.

[Bräuning and Hüllermeier, 2016] Bräuning, M. and Hüllermeier, E. (2016). Learning conditional lexicographic preference trees. *Archives of Data Science, Series A*, 1(1):41–55.

[Chevaleyre et al., 2010] Chevaleyre, Y., Koriche, F., Lang, J., Mengin, J., and Zanuttini, B. (2010). Learning ordinal preferences on multiattribute domains: The case of cp-nets. In Fürnkranz, J. and Hüllermeier, E., editors, *Preference Learning*, pages 273–296. Springer.

[Coste-Marquis et al., 2004] Coste-Marquis, S., Lang, J., Liberatore, P., and Marquis, P. (2004). Expressive power and succinctness of propositional languages for preference representation. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004), Whistler, Canada, June 2-5, 2004*, pages 203–212.

[Domshlak, 2008] Domshlak, C. (2008). A snapshot on reasoning with qualitative preference statements in ai. In *Preferences and Similarities*, pages 265–282.

[Domshlak and Brafman, 2002] Domshlak, C. and Brafman, R. I. (2002). Cp-nets: Reasoning and consistency testing. In *Proceedings of the Eights International Conference on Principles and Knowledge Representation and Reasoning (KR-02), Toulouse, France, April 22-25, 2002*, pages 121–132.

[Dubois et al., 1996] Dubois, D., Fargier, H., and Prade, H. (1996). Refinements of the maximin approach to decision-making in a fuzzy environment. *Fuzzy Sets Syst.*, 81:103–122.

[Dubois et al., 2015] Dubois, D., Hadjali, A., Prade, H., and Touazi, F. (2015). Erratum to: Database preference queries - a possibilistic logic approach with symbolic priorities. *Ann. Math. Artif. Intell.*, 73(3-4):359–363.

[Dubois et al., 2004] Dubois, D., Kaci, S., and Prade, H. (2004). Ordinal and absolute representations of positive information in possibilistic logic. In *10th International Workshop on Non-Monotonic Reasoning (NMR 2004), Whistler, Canada, June 6-8, 2004, Proceedings*, pages 140–146.

[Dubois et al., 2005] Dubois, D., Kaci, S., and Prade, H. (2005). Expressing preferences from generic rules and examples - A possibilistic approach without aggregation function. In Godo, L., editor, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 8th European Conference, ECSQARU 2005, Barcelona, Spain, July 6-8, 2005, Proceedings*, volume 3571 of *Lecture Notes in Computer Science*, pages 293–304. Springer.

[Dubois et al., 1987] Dubois, D., Lang, J., and Prade, H. (1987). Theorem proving under uncertainty - A possibility theory-based approach. In McDermott, J. P., editor, *Proceedings of the 10th International Joint Conference on Artificial Intelligence. Milan, Italy, August 23-28, 1987*, pages 984–986. Morgan Kaufmann.

[Dubois et al., 1994] Dubois, D., Lang, J., and Prade, H. (1994). Automated reasoning using possibilistic logic: Semantics, belief revision, and variable certainty weights. *IEEE Trans. Knowl. Data Eng.*, 6(1):64–71.

[Dubois and Prade, 1988] Dubois, D. and Prade, H. (1988). *Possibility Theory - An Approach to Computerized Processing of Uncertainty*. Springer.

[Dubois and Prade, 1990] Dubois, D. and Prade, H. (1990). The logical view of conditioning and its application to possibility and evidence theories. *Int. J. Approx. Reason.*, 4(1):23–46.

[Dubois and Prade, 1991] Dubois, D. and Prade, H. (1991). Conditional objects and non-monontonic reasoning. In *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). Cambridge, MA, USA, April 22-25, 1991*, pages 175–185.

[Dubois and Prade, 2012] Dubois, D. and Prade, H. (2012). Possibility theory and formal concept analysis: Characterizing independent sub-contexts. *Fuzzy Sets Syst.*, 196:4–16.

[Dubois and Prade, 2015] Dubois, D. and Prade, H. (2015). Possibility theory and its applications: Where do we stand? In *Springer Handbook of Computational Intelligence*, pages 31–60.

[Dubois and Prade, 2016] Dubois, D. and Prade, H. (2016). Qualitative and semi-quantitative modeling of uncertain knowledge - A discussion. In Beierle, C., Brewka, G., and Thimm, M., editors, *Computational Models of Rationality, Essays dedicated to Gabriele Kern-Isberner on the occasion of her 60th birthday*, pages 280–296. College Publications.

[Dubois et al., 2013] Dubois, D., Prade, H., and Touazi, F. (2013). Conditional preference nets and possibilistic logic. In van der Gaag, L. C., editor, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty - 12th European Conference, ECSQARU 2013, Utrecht, The Netherlands, July 8-10, 2013. Proceedings*, volume 7958 of *Lecture Notes in Computer Science*, pages 181–193. Springer.

[Eichhorn et al., 2016] Eichhorn, C., Fey, M., and Kern-Isberner, G. (2016). CP- and ocf-networks - a comparison. *Fuzzy Sets Syst.*, 298:109–127.

[Fargier et al., 2018] Fargier, H., Gimenez, P., and Mengin, J. (2018). Learning lexicographic preference trees from positive examples. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 2959–2966.

[Fargier and Mengin, 2021a] Fargier, H. and Mengin, J. (2021a). A knowledge compilation map for conditional preference statements-based languages. In *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, pages 492–500.

[Fargier and Mengin, 2021b] Fargier, H. and Mengin, J. (2021b). A knowledge compilation map for conditional preference statements-based languages. In Dignum, F.,

Lomuscio, A., Endriss, U., and Nowé, A., editors, *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, pages 492–500. ACM.

[Fürnkranz and Hüllermeier, 2010] Fürnkranz, J. and Hüllermeier, E. (2010). Preference learning and ranking by pairwise comparison. In Fürnkranz, J. and Hüllermeier, E., editors, *Preference Learning*, pages 65–82. Springer.

[Gigerenzer and Goldstein, 1996] Gigerenzer, G. and Goldstein, D. (1996). Goldstein, d.g.: Reasoning the fast and frugal way: models of bounded rationality. psychological review 103(4), 650. *Psychological Review*, 103:650–669.

[Gonzales and Perny, 2004] Gonzales, C. and Perny, P. (2004). GAI networks for utility elicitation. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004), Whistler, Canada, June 2-5, 2004*, pages 224–234.

[Gouider, 2017] Gouider, H. (2017). *Graphical Preference Representation under a Possibilistic Framework*. PhD thesis, Université de Tunis.

[Kaci, 2012] Kaci, S. (2012). Characterization of positive and negative information in comparative preference representation. In *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*, pages 450–455.

[Kaci and van der Torre, 2008] Kaci, S. and van der Torre, L. W. N. (2008). Reasoning with various kinds of preferences: logic, non-monotonicity, and algorithms. *Ann. Oper. Res.*, 163(1):89–114.

[Kraus et al., 1990] Kraus, S., Lehmann, D., and Magidor, M. (1990). Nonmonotonic reasoning, preferential models and cumulative logics. *Artif. Intell.*, 44(1-2):167–207.

[Lang et al., 2012] Lang, J., Mengin, J., and Xia, L. (2012). Aggregating conditionally lexicographic preferences on multi-issue domains. In *Principles and Practice of Constraint Programming - 18th International Conference, CP 2012, Québec City, QC, Canada, October 8-12, 2012. Proceedings*, pages 973–987.

[Liu and Truszczynski, 2015] Liu, X. and Truszczynski, M. (2015). Learning partial lexicographic preference trees over combinatorial domains. In Bonet, B. and Koenig, S., editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 1539–1545. AAAI Press.

[Liu et al., 2018] Liu, Z., Zhong, Z., Zhang, C., Yu, Y., and Liu, J. (2018). Learning cp-nets structure from preference data streams. *IEEE Access*, 6:56716–56726.

[Pearl, 1990] Pearl, J. (1990). System Z: A natural ordering of defaults with tractable applications to nonmonotonic reasoning. In *Proceedings of the 3rd Conference on Theoretical Aspects of Reasoning about Knowledge, Pacific Grove, CA, USA, March 1990*, pages 121–135.

[Ribeiro et al., 2018] Ribeiro, M. R., Barioni, M. C. N., de Amo, S., Roncancio, C., and Labbé, C. (2018). Incremental evaluation of continuous preference queries. *Inf. Sci.*, 453:127–153.

[Robinson, 1977] Robinson, R. W. (1977). Counting unlabeled acyclic digraphs. In Little, C. H. C., editor, *Combinatorial Mathematics V*, pages 28–43, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Shafran et al., 2016] Shafran, A., Saarinen, S., and Goldsmith, J. (2016). A tool to graphically edit cp-nets. In Schuurmans, D. and Wellman, M. P., editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 4387–4388. AAAI Press.

[Spohn, 1988] Spohn, W. (1988). Ordinal conditional functions: A dynamic theory of epistemic states. *Causation in Decision, Belief Change, and Statistics*, pages 105–134.

[Spohn, 2012] Spohn, W. (2012). *The Laws of Belief–Ranking Theory and its Philosophical Applications*. OUP Oxford.

[van Benthem et al., 2009] van Benthem, J., Girard, P., and Roy, O. (2009). Everything else being equal: A modal logic for *Ceteris Paribus* preferences. *J. Philos. Log.*, 38(1):83–125.

[Wang et al., 2018] Wang, H., Tao, Y., Yu, Q., Lin, X., and Hong, T. (2018). Incorporating both qualitative and quantitative preferences for service recommendation. *J. Parallel Distributed Comput.*, 114:46–69.

[Wilson, 2011] Wilson, N. (2011). Computational techniques for a simple theory of conditional preferences. *Artif. Intell.*, 175(7-8):1053–1091.

[Wilson et al., 2019] Wilson, N., Dubois, D., and Prade, H. (2019). Cp-nets, \pi -pref nets, and pareto dominance. In *Scalable Uncertainty Management - 13th International Conference, SUM 2019, Compiègne, France, December 16-18, 2019, Proceedings*, pages 169–183.

[Zadeh, 1978] Zadeh, L. A. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28.