On Computationally Efficient Approaches to Agent-based Group Decision-making

Minyi Li

Submitted in fulfillment of the requirements of the degree of Doctor of Philosophy

2012
Abstract

Group decision-making, in which a collective decision needs to be derived from individual agents’ preferences has been an active area of research. In group decision-making, autonomous agents need some procedures or mechanisms that can collect preference information from them and make a decision; or some protocols that enable them to be involved in the decision process, to interact with each other and to jointly make a collective decision. Examples include preference aggregation mechanisms, negotiation protocols, voting procedures, and auctions.

This thesis studies the problem of agent-based group decision-making in various problem settings, proposing several efficient approaches to support multiple agents in reaching efficient and fair collective decisions over multiple issues. The research work in this thesis can be divided into two parts, addressing the agent-based group decision-making problem in circumstances in which there is, respectively, complete and incomplete information available.

The first part of this thesis focuses on preference aggregation in combinatorial domains, provided all agents’ preferences. Structured preferences Conditional Preference Network (CP-nets) and its variant Trade-off enhanced CP-nets (TCP-nets) are used as formal models for representing the agents’ preferences in group decision-making. Before going into the problem of making a group decision from a collection of CP-nets or TCP-nets, the problem of individual preference reasoning is studied and a computationally efficient heuristic algorithm for dominance testing in CP-nets, which forms the basis for many group decision-making mechanisms, is introduced.

Subsequently, we consider the group decision-making problem with multiple agents’ CP-nets and TCP-nets. As the space of possible outcomes in combinatorial domains has a size exponential in the number of variables, computational complexity in group decision-making will be one of the issues that need to be addressed. Two computationally efficient approaches are proposed in order to support agent-based group decision-making in combinatorial domains with different decision criteria: (i) an approach that converts the qualitative preferences derived from CP-nets or TCP-nets into numerical penalty scoring functions, and then searches for an optimal collective decision based on the individual penalty scores; and (ii) an approach that focuses on combinatorial vote and computes the winners (i.e., winning alternatives) from a collection of CP-nets.
according to Majority voting rule.

The second part of this thesis is concerned with the disadvantage of disclosing preference information in group decision-making and considers the incomplete information setting. Two negotiation approaches are presented in order to support agent-based group decision-making in, respectively, combinatorial domains and in domains with multiple continuous issues. Firstly, a negotiation protocol in combinatorial domains is proposed, which enables agents to distributively make decisions and leads them to Pareto-optimal agreements under incomplete information setting. Then for the domains with multiple continuous issues, a cooperative mediated negotiation approach is proposed to support the agents searching for joint utility gains and reaching an efficient and fair agreement.

For every approach introduced in this thesis, experiments are carried out and the results are presented and analysed in order to provide insights into the performance of these approaches.

Thesis Supervisors: Quoc Bao Vo and Ryszard Kowalczyk
Acknowledgements

First and foremost, I sincerely express my deepest gratitude to my coordinating supervisor, Quoc Bao Vo. Bao has been a great advisor. Over the course of countless research meetings and discussions, he has taught me how to do high quality research. He has also taught me to think of the big picture and present my work accordingly. In addition, Bao’s advice has extended beyond research to teaching, career planning, and even personal issues; and he has always been willing to take extra time out of his busy schedule when important or unusual issues came up. Meanwhile, he worked behind the scenes to make sure that I did not have to worry about funding or administrative problems, and could focus on research. Finally, and perhaps most importantly, Bao has been endlessly motivating and encouraging. Without his consistent support and encouragement, I would not have been able to complete this manuscript. In fact, to me, Bao is not only a supervisor, but also an important friend of mine.

I thank Swinburne University of Technology and the Faculty of Information and Communication Technologies for offering me a full Research Scholarship throughout my doctoral program. I also thank the Research Committee of the Faculty of Information and Communication Technologies for research publication funding support and for providing me with financial support to attend academic conferences.

I would like to take this opportunity to thank my associate supervisor Professor Ryszard Kowalczyk, for his helpful comments, continuous encouragement and financial supports to attend academic conferences. I would also like to thank Professor Edmund Durfee and Professor Yan Zhang, for being my thesis viewer and providing me with helpful and valuable comments on this thesis. Moreover, I am indebted to many other researchers in Artificial Intelligence, Multi-agent systems and Computational Social Choice with whom I have had valuable discussions. While the following list is undoubtedly incomplete (due to the anonymity of the review process), special thanks go out to Lin Padgham, Jérôme Lang, Vincent Conitzer, Lirong Xia, Dongmo Zhang, Sebastian Sardina, John Thangarajah, Toby Walsh, Francesca Rossi and Ulle Endriss.

I would also like to thank Geoffrey Vincent, a professional editor, for correcting my English language usage and grammatical errors.

Back at Swinburne, my thanks also go to staff members, research students and research
assistants at previous CITR/CS3 and current SUCCESS for their help, suggestions, friendship and encouragement, in particular, Professor Yun Yang, Professor Chengfei Liu, Robert Merkel, Gillian Foster, Sharon Raj, Vicki Redrup, Xiao Liu, Dong Yuan, Miao Du, Jianxin Li, Rui Zhou, Qiang He, Feifei Chen, Wei Dong, Jing Gao, Daihai Cao and Wenhao Li.

I also want to thank my close friend Guoan Zheng for supporting me and providing me with a lot of helpful information during my candidature.

I am deeply grateful to my parents Liying Guo and Zhiwen Li for raising me up, teaching me to be a good person, and supporting me to study abroad. Finally, I would like to express my greatest thanks to my husband Zhi Li, for his understanding, patience, support and love.
Declaration

This is to certify that this thesis contains no material which has been accepted for the award of any other degree or diploma and that to the best of my knowledge this thesis contains no material previously published or written by another person except where due reference is made in the text of the thesis.

Minyi Li
List of Publications


Contents

Abstract iii

Acknowledgements

Declaration v

List of Publications vi

1 Introduction 1
   1.1 Motivation ................................................. 1
   1.2 A broad overview ........................................... 3
   1.3 Research questions ........................................ 10
   1.4 Contributions ............................................ 11
      1.4.1 Preference aggregation in combinatorial domains .... 12
      1.4.2 Multi-issue Negotiation ............................... 15
   1.5 Chapter Outlines ........................................ 19

2 Background 24
   2.1 The basic components ................................. 24
   2.2 The domain .............................................. 25
   2.3 Preference ............................................. 26
      2.3.1 Preference relation ................................. 27
      2.3.2 Utility ........................................... 28
      2.3.3 Graphical representations CP-net and TCP-net .... 29
      2.3.4 Individual dominance testing in CP-nets and TCP-nets 37
   2.4 Typical goals in group decision-making .................. 39
      2.4.1 Pareto-optimality ................................. 39
      2.4.2 Fairness ........................................... 41
   2.5 Relevant Literature ................................. 43
      2.5.1 Preference aggregation and voting in combinatorial domains 43
      2.5.2 Multi-issue negotiation ............................. 47

1 Preference aggregation in combinatorial domains 50

3 Penalty score-based heuristic for dominance testing in CP-nets 51
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>An example of utility function</td>
<td>28</td>
</tr>
<tr>
<td>2.2</td>
<td>An example of CP-net</td>
<td>32</td>
</tr>
<tr>
<td>2.3</td>
<td>Preference ordering induced by the CP-net in Figure 2.2</td>
<td>33</td>
</tr>
<tr>
<td>2.4</td>
<td>An example of TCP-net $N$</td>
<td>35</td>
</tr>
<tr>
<td>2.5</td>
<td>Dependency graph and its w-directed graphs of $N$</td>
<td>35</td>
</tr>
<tr>
<td>3.1</td>
<td>An example CP-net and its variable importance weight</td>
<td>56</td>
</tr>
<tr>
<td>3.2</td>
<td>Improving search tree for query $N \models abcd &gt; abcd$ using Least-variable flipping rule</td>
<td>61</td>
</tr>
<tr>
<td>3.3</td>
<td>Improving search tree</td>
<td>64</td>
</tr>
<tr>
<td>3.4</td>
<td>Avg. number of visited nodes with binary-value tree-structured CP-nets</td>
<td>76</td>
</tr>
<tr>
<td>3.5</td>
<td>Multi-valued tree-structured CP-nets</td>
<td>76</td>
</tr>
<tr>
<td>3.6</td>
<td>Avg. number of visited nodes with binary-valued, directed-path singly connected CP-nets</td>
<td>77</td>
</tr>
<tr>
<td>3.7</td>
<td>Multi-valued, directed-path singly connected CP-nets</td>
<td>77</td>
</tr>
<tr>
<td>3.8</td>
<td>Binary-value arbitrary acyclic CP-nets</td>
<td>78</td>
</tr>
<tr>
<td>3.9</td>
<td>Multi-valued arbitrary acyclic CP-nets</td>
<td>79</td>
</tr>
<tr>
<td>4.1</td>
<td>An example of TCP-net</td>
<td>82</td>
</tr>
<tr>
<td>4.2</td>
<td>Dependency graph and arc values of $N$</td>
<td>84</td>
</tr>
<tr>
<td>4.3</td>
<td>The TCP-nets 3 agents</td>
<td>92</td>
</tr>
<tr>
<td>4.4</td>
<td>An example of HeTCP procedure</td>
<td>97</td>
</tr>
<tr>
<td>4.5</td>
<td>Average execution time comparison with binary-valued CP-nets</td>
<td>106</td>
</tr>
<tr>
<td>4.6</td>
<td>Average execution time comparison with multi-valued CP-nets</td>
<td>107</td>
</tr>
<tr>
<td>5.1</td>
<td>Illustration for Proposition 5.1</td>
<td>121</td>
</tr>
<tr>
<td>5.2</td>
<td>Illustration for Propositions 5.2 and 5.3</td>
<td>122</td>
</tr>
<tr>
<td>5.3</td>
<td>Illustration for Proposition 5.2</td>
<td>123</td>
</tr>
<tr>
<td>5.4</td>
<td>CP-nets of the agents</td>
<td>133</td>
</tr>
<tr>
<td>5.5</td>
<td>Average execution time for computing the set of possible winners with binary CP-nets(Log scale plot)</td>
<td>145</td>
</tr>
<tr>
<td>5.6</td>
<td>Average time comparison for computing the set of wLCWs with binary CP-nets(Log scale plot)</td>
<td>145</td>
</tr>
<tr>
<td>5.7</td>
<td>Average execution time for computing the set of wLCWs with binary CP-nets(Log scale plot)</td>
<td>146</td>
</tr>
<tr>
<td>5.8</td>
<td>Percentage of cases when there are no wLCWs with binary CP-nets (%)</td>
<td>146</td>
</tr>
</tbody>
</table>
List of Figures

5.9 Percentage of cases when there exists a unique wLCW with binary CP-nets (%) .......................... 147
5.10 Average execution time for computing the set of possible winners with multi-value CP-nets (Log scale plot) .................................................. 147
5.11 Average time comparison for computing the set of wLCWs with multi-valued CP-nets(Log scale plot) ................................................................. 148
5.12 Average execution time for computing the set of wLCWs with Multi-valued CP-net (Log scale plot) ................................................................. 148
5.13 Percentage of cases when there are no wLCWs with multi-valued CP-nets (%) ................................................................. 149
5.14 Percentage of cases when there exists a unique wLCW with multi-valued CP-nets (%) .................. 149
6.1 Two agents’ CP-nets .................................................. 154
6.2 Negotiation tree .................................................. 162
6.3 The fringes of the agents in the negotiation .................................................. 163
6.4 The preference orderings of two agents .................................................. 163
6.5 Average execution time with binary-valued poly-tree structured CP-nets (Log scale plot) ............ 173
6.6 Average execution time with multi-valued poly-tree structured CP-nets (Log scale plot) ............ 173
6.7 Average execution time with binary-valued arbitrary acyclic CP-nets (Log scale plot) ............... 174
6.8 Average execution time with multi-valued arbitrary acyclic CP-nets (Log scale plot) ............... 174
7.1 Example of indifference curve .................................................. 179
7.2 An example of utility gradient .................................................. 181
7.3 Jointly improving directions set .................................................. 185
7.4 The iterative negotiation procedure .................................................. 194
7.5 Negotiation over fishing rights: indifference curves at the initial agreement .................................................. 197
7.6 Average ESW .................................................. 206
7.7 Average DR .................................................. 206
7.8 $10^{th}$-$90^{th}$ Percentile box of DR .................................................. 207
7.9 Average distance from egalitarian solution point .................................................. 207
7.10 $10^{th}$-$90^{th}$ Percentile box of the distance from egalitarian solution .................................................. 208
7.11 Average number of iterations .................................................. 208
List of Tables

3.1 Average execution time with Tree-structured binary-valued CP-nets (Set 1) .......................................................... 73
3.2 Average execution time with Tree-structured multi-valued CP-nets (Set 2) .......................................................... 73
3.3 Average execution time with directed-path singly connected binary-valued CP-nets (Set 3) ........................................... 74
3.4 Average execution time with directed-path singly connected binary-valued CP-nets (Set 4) ........................................... 74
3.5 Average execution time with arbitrary acyclic binary-valued CP-nets (Set 5) .......................................................... 74
3.6 Average execution time with arbitrary acyclic multi-valued CP-nets (Set 6) .......................................................... 75
4.1 Outcome space and average number of visited Nodes with binary TCP-nets ............................................................ 104
4.2 Average outcome space and average number of visited Nodes with multi-valued TCP-nets ........................................ 105
5.1 Average number of wLCWs when they exist with binary CP-nets (%) .......................................................... 143
5.2 Average outcome space and average number of wLCWs when they exist with multi-value CP-nets ................................ 144
6.1 Negotiations with binary-valued poly-tree structured CP-nets .......................................................... 169
   Continued with Table 6.1 .......................................................... 170
6.3 Negotiations with binary-valued arbitrary acyclic CP-nets .......................................................... 171
6.4 Negotiations with multi-valued poly-tree structured CP-nets .......................................................... 172
6.5 Negotiations with multi-valued arbitrary acyclic CP-nets .......................................................... 172
7.1 The results of fishing right negotiation in bilateral setting .......................................................... 198
7.2 The results of fishing right negotiation in multilateral setting .......................................................... 199
7.3 Range of parameter values of the parties’ utility functions .......................................................... 200
7.4 The results of water policy negotiation .......................................................... 201
Chapter 1

Introduction

1.1 Motivation

Multi-agent systems are computational approaches that are being more and more widely used for solving real world, dynamic and open system problems [67]. Many important scenarios are conceptualised as a collection of autonomous agents that collaborate to choose one or some outcomes (or solutions) as a collective decision. For instance, the routing of telephone calls and data packets in wireless networks has been controlled and decided by a group of autonomous agents. Multi-agent techniques are now being developed for a range of applications in power engineering industries, including diagnostics [46], market simulation [141], and network control [49, 83]. Research has also been done on how agents systems can react to, and control, automotive and airplane traffic in real time [105].

As agent systems become increasingly interactive and collaborative, more and more problems come to the forefront that are fundamentally about selecting good outcomes in the face of the conflicting preferences of different agents. Examples include in large scale web service systems, in which individual agents are required to cooperate and decide on how to allocate various tasks and resources among themselves, in order to provide a composition of service [36, 47, 59, 127, 129]; the customer agents and service provider agents may also need to negotiate and reach service level agreements (SLAs) [45, 113, 151]. Moreover, autonomous agents may have the potential to aid, or potentially act on behalf of, humans in some important real world group decision-making process. For instance, the governments of different countries or nations may
need to negotiate and reach an agreement over multiple issues on political issues or natural resources sharing policies (e.g. water usage policy [6, 138], fishing rights [58], climate change policy [119]), or on the form that an international treaty will take, etc. Another example might be an institute that wants to elect an administrative steering committee, composed of a president, a vice-president and a treasurer [17]. The motivation of the research in agent-based group decision-making is driven firstly by the applications of agent technologies in solving real world group decision-making problems.

From another perspective, in order to make a group decision effectively, there is a need for some procedures or mechanisms that can collect the preference information from the agents and make a decision; or some protocols that enable the agents to be involved in the decision-making process, to interact with each other and to jointly make the collective decision. There are two great challenges in designing a good collective decision-making mechanism or protocol, which further motivates the research in this field. Firstly, a good protocol or mechanism should, provided with either all of the agents’ preference information or only a part of it, choose an outcome that is considered to be socially good for the group of agents. Thus, the studies on the desirable properties of the final chosen outcome by the group decision-making mechanisms play a significant role. Secondly, many real world decision-making problems involved complex attribute domains, e.g., when the alternative space has a combinatorial structure. In such complex settings, deciding on a good collective decision, even with all of the agents’ preference information in hand, may require the solution of a computationally hard optimization problem. This kind of computationally nontrivial problem must be solved in order to support agent-based group decision-making in real world applications. Therefore, there is a need to investigate practical approaches for solving complex group decision-making problems on multi-attribute domains.

To this end, the aim of this thesis is to investigate techniques to support agent-based group decision-making in various problem settings. The research work in this thesis can be divided into two parts, addressing the agent-based group decision-making problem under circumstances in which there is, respectively, complete information and incomplete information. In order to lead multiple agents to efficient and fair collective decisions over multiple issues, several computationally efficient approaches are proposed in this thesis, including: i) a preference aggregation mechanism in combinatorial

---

1The space of possible outcomes is a Cartesian product of finite value domains for each one of a set of variables.
domains, based on a numerical transformation of the agents’ preference information; 
ii) a CSP-based (SAT-based) approach to aggregate multiple agents’ preferences in 
combinatorial domains according to a majority rule; iii) a distributive protocol for ne-
gotiation in combinatorial domains with general preferences under circumstances in 
which there is only incomplete information; iv) a mediated negotiation approach for 
group decision-making over multiple continuous issues under incomplete information.

1.2 A broad overview

In general, agent-based group decision-making (a.k.a., agent-based collective decision-
making) is a type of multi-party decision-making process through which multiple 
agents act collectively, consider and evaluate possible alternatives based on their pref-
ferences, and select from among the possible alternatives a solution or a set of solutions. 
It is an interdisciplinary field of study at the interface of group decision-making theory, 
and computer science (particularly artificial intelligence (AI) and computing theory), 
promoting an exchange of ideas in both directions [40, 112, 133].

Group decision-making theory is concerned with the design and analysis of meth-
ods for group decision-making. Most existing work in group decision-making theory 
has focused on establishing theoretical results regarding the existence of procedures 
or solutions meeting certain requirements, but computational issues have rarely been 
considered [40]. Agent-based group decision-making is then concerned with the ap-
plication of techniques (in particular, intelligent agent technologies) developed in com-
puter science and artificial intelligence, such as algorithm design, search strategies and 
complexity analysis, to the study of group decision-making mechanisms, such as nego-
tiation procedures, voting protocols, fair division algorithms and preferences aggrega-
tion mechanisms. For example, the applications of AI and computing theory to group 
decision-making include, the formal specification and verification of group decision-
making procedures (for instance, fair division algorithms), and the development of 
compact representation of preferences in combinatorial domains.

On the other hand, the study of group decision-making theory is also very relevant 
to multi-agent systems [133]. In many real world applications of multi-agent systems, 
we need to represent and reason about the simultaneous preferences of multiple agents, 
and make a collective decision [88, 125]. For instance, the case for managing societies 
of autonomous software agents requires the study of negotiation and voting procedures
in group decision-making theory. Other examples include tasks and resource allocations in multi-agent systems, and negotiation and cooperation between multiple service provider agents in order to provide a composition of service. Importing concepts from group decision-making theory into multi-agent systems facilitates interaction, cooperation or negotiation within agent societies, and enables the investigations of the properties of the final chosen outcome(s).

Existing studies on agent-based group decision-making can be divided into two basic scenarios, according to different information settings. A large body of work from social choice theory has been considered a complete information setting, in which the agents are required to submit their preferences over the possible alternatives such that a collective decision can be made to best convey the preference of the group \[9, 12, 114, 121\]. For instance, most voting protocols and preferences aggregation mechanisms that collect the preference information from the agents and compute an optimal outcome or a winning alternative according to a given decision rule. Some other work has been concerned with the disadvantage of revealing preference information in group decision-making and, thus, considers an incomplete information setting, i.e., the agents’ preferences are not common knowledge \[57, 66, 122, 127\]. For instance, using a negotiation approach to make a collective decision, the individual agents may not have information on the opponents’ preferences and they may be reluctant to completely reveal their own preferences during the process of negotiation.

Furthermore, based on the format of the agents’ preferences in group decision-making, research in this field is composed of two main classes of problems well studied in the areas of game theory, welfare economics and social choice theory, and artificial intelligence \[92\]. Research in the first class of problems assumes that individual agents’ preferences are represented quantitatively by *utility functions*, which form the basis for the collective decision. Existing research uses negotiation approaches to enable agents to interact with each other and jointly make a collective decision \[55, 57, 66, 78, 122\], or investigates collective decision functions to select an outcome that satisfies a set of axioms or conditions based on the agents’ utility values (e.g. \[2, 9, 79, 109\]). The second class of problems typically expressed preferences in an ordinal way by means of *preference relations* over the set of possible alternatives. Then, the agents negotiate with each other based on their preference relations, or these preferences are aggregated in order to identify (or elect) an acceptable common alternative or a collective preference relation in an automated way (without negotiation). For instance, a *social choice function* or a *voting rule* maps a collection of individual preference relations into a
single selected candidate [21, 33, 72, 95, 120, 121, 148, 150], or a social welfare function maps a collection of individual preference relations into a collective preference relation (e.g. [73, 88, 125]).

Finally, from the process point of view, there are different protocols that allow an agreement to be reached. For instance, the single-stage decision-making process, e.g., one-stage voting and most aggregation mechanisms in combinatorial domains; and multi-stage decision-making process, e.g., sequential voting and most negotiation protocols for addressing group decision-making problems.

**Research areas in agent-based group decision-making**

The following section gives a (non-exhaustive) list of research areas falling within the field of agent-based group decision-making [40]. The classification is based on the nature of the group decision-making problem dealt with.

**Preference aggregation.** Many problems require the aggregation of preferences of different individuals (or agents). The origin problem of preference aggregation is how to transform diverse individual level preferences into common collective preferences, which can be used to make collective choices [38, 40, 76]. Aggregating preferences means mapping a collection of agents’ preferences into a collective preference relation. For example, when planning a wedding, we must combine the preferences of the bride, the groom and possibly their families. However, sometimes we are only concerned with determining a socially preferred alternative, or a subset of socially preferred alternatives, rather than a full collective preference relation. For instance, we may only want to know the commonly accepted arrangement for a wedding party, rather than knowing the explicit collective preference ordering of all people involved over all the possible alternatives.

Scientific fields that traditionally deal with preference aggregation, such as social choice theory, game theory, computational mechanism design and political science, have highlighted a number of impossibility theorems [9, 70]. For instance, Arrow’s Impossibility Theorem (Arrow [9]), a seminal result in the field, shows that there can exist no preference aggregation mechanism that would simultaneously satisfy a small number of natural requirements (e.g., the aggregation function should not be dictatorial).
Moreover, over recent years, preference aggregation has attracted a dramatically increased degree of attention from computer scientists coming from various fields such as artificial intelligence and mechanism design. Computer science extends existing research by introducing computational and communication efficiency, decentralised mechanism execution (e.g., distributed protocols that allow autonomous entities to aggregate their conflicting preferences), privacy and new applications such as scheduling, file-sharing or knowledge transfer. In general, the topic of preference aggregation on its own is less specific than voting (see below), which mostly deal with some sort of preference aggregation, but in a much more specific context.

**Voting**. Another natural way in which agents can make a joint decision when they have possibly conflicting preferences over a set of alternatives is voting [32, 149]. Voting is a method for a group such as a committee or an electorate to make a decision, often following discussions, debates or election campaigns. It is an area closely related to preference aggregation. Mathematically, a *voting rule* or *voting correspondence* is defined as a mapping from a set of expressions of individual preferences (called a voting profile), to a winning alternative or a winning set of alternatives. Different voting systems may use different types of vote. For instance, consider an election for a conference chair with the options Alice, Bob, Charlie, Dan and Emily. Plurality voting system uses a single vote, in which the voter selects his or her most preferred candidate. For example, a voter might vote for Alice as her most preferred outcome. Approval voting uses a multiple vote, i.e., the voter can vote for any subset of the alternatives. Some other voting systems use a ranked vote (for instance, preferential voting systems), by means of which each agent (voter) initially expresses her preferences on a set of alternatives (called candidates); the individuals’ preferences are then aggregated in order to identify (or elect) an acceptable common alternative in an automated way (without negotiation). For example, an agent might vote for Bob in first place, then Emily, then Alice, then Daniel and finally Charlie.

An important concept in voting is the Condorcet winner. A Condorcet winner is a candidate who, when compared with every other candidate, is preferred by more voters. Informally, the Condorcet winner is the one who would win a pairwise election against any of the other candidates. A Condorcet winner does not always exist for a given preference profile, which is known as Condorcet voting paradox. Obviously, when there exists a Condorcet winner, then it is unique. A decision rule satisfies the Condorcet criterion (called Condorcet-consistent rule)
if it chooses the Condorcet winner when it exists. Any method conforming to the Condorcet criterion is known as a Condorcet method.

Vote problems have been investigated by researchers in social choice theory and voting theory (for instance, see in Arrow et al. [13] that a whole panorama of voting rules have been proposed). The main research problems in this area include the properties of various families of voting rules (e.g., Majority rule, Borda rule, Plurality rule), winner determination problems, and algorithm design for computing winning alternative according to a specific voting rule.

**Negotiation (a.k.a., bargaining).** Negotiation (bargaining) is one of the most common approaches used to make decisions and manage disputes in groups. A negotiation situation is characterised by a set of agents (individuals) who have a common interest in discussion and reaching an agreement, but who have conflicting interests concerning the possible alternative solutions or the particular way in which a solution is reached. It is a complex task in the course of the performance of which a balance between competition and collaboration needs to be managed.

Negotiation has been well studied in the fields of Economics and Artificial Intelligence. Most research in this field to date has focussed on the competitive aspects. However, work by Dispute Resolution theorists in the social sciences has also focussed substantially on how to achieve negotiated agreements that are of a high value to all parties (e.g., see [68, 96]). The earlier stage of research work in this area focuses mainly on the negotiation problem over a single issue, such as the price of a good to be negotiated. Single-issue negotiation is a “Win-Lose” situation, i.e. the gain of one agent always creates a loss for the other agent. Over recent years, the negotiation problem over multiple issues has aroused increasing interest and received more and more attentions. Multi-issue negotiation is a negotiation that involves multiple issues that need to be negotiated simultaneously. For instance, commanders and sailors usually have to negotiate multiple issues, e.g., payment rate, projected rotation date, length of service, training and many other matters. Multi-issue negotiation has been widely used to address multi-agent group decision-making problems because of its advantage of possible “Win-Win” solutions for both sides. “Win-Win” solutions means that in a multi-issue negotiation, because agents may have different preferences on the issues, each of the sides may achieve a more satisfactory agreement on the issues that are most important to them by trading off some on those that are not so important [91].
The aforementioned two group decision-making areas, namely preference aggregation and voting, have mainly been studied in the complete information setting, i.e., the agents are required to submit their preference information in order to make a collective decision. However, in negotiation the disclosure of an agent’s preference may be exploited by the opponents to the disadvantage of that agent. Thus, most negotiations in real life take place under incomplete information about the preferences of the negotiating parties [123, 131]. One of the most important goals of the research in this area is to investigate negotiation protocols or procedures that can lead rational agents to Pareto-efficient agreements under incomplete information setting. Pareto-efficiency means that the outcome (or solution) ensures that no other outcomes (or solution) would make some agents better off without making some other agents worse off. Another desirable property of a negotiated outcome is fairness, which has been well studied in game theory and social choice theory. Existing metrics for measuring the fairness of a negotiated outcome include the Nash bargaining solution (Nash [109]), the Kalai-Smorodinsky bargaining solution (Kalai and Smorodinsky [79]), Utilitarian social welfare (Binmore [18]) and Egalitarian social welfare (Binmore [18]).

**Fair division.** Fair division is a mathematical theory based on an idealization of a real life problem. The real life problem is the one of dividing goods or resources between a set of self-interested agents that has an entitlement to them, in such a way that all recipients believe that they have received a fair amount. The central tenet of fair division is that such a division should be performed by the players themselves, possibly using a mediator but certainly not an arbiter, as only the players really know how they value the goods. The research in this area aims to achieve a good allocation that satisfies different desirable properties, such as Pareto-efficiency. However, a division through which one agent gets everything is efficient by this definition, so this definition on its own does not guarantee a fair share. Another well studied property is Envy-freeness (Brams and Taylor [34]), in the sense that the division guarantees that no agent will prefer another agent’s share more than their own. Other properties include simple fairness, exactness, equity, and a range of others.

**Judgement aggregation and belief merging.** In judgment aggregation, individual agents are required to vote for or against a certain decision (the conclusion) while providing reasons for their choice. The reasons and the conclusion are logically connected propositions and are given in the decision problem. The problem
is how a collective judgment on logically interconnected propositions can be defined from individual judgments on the same propositions. The researchers in social choice and political theories have claimed that judgment aggregation raises serious concerns [117]. For example, consider a set of premises and a conclusion in which the latter is logically equivalent to the former. Such aggregation problems occur in many different collective decision-making bodies (especially committees and expert panels). The challenge in this area lies in the fact that a natural aggregation procedure, like the majority rule, turns out to generate possibly inconsistent collective outcomes, which indicates that the reasons do not support the selected conclusion. Judgment aggregation is distinct from the more familiar problem of preference aggregation. Yet, just as preference aggregation is impeded by a paradox (Condorcet paradox of cyclical majority preferences), judgment aggregation is also impeded by a paradox: the "discursive dilemma" or "doctrinal paradox". Even worse, it can be shown that any judgment aggregation procedure that satisfies some desirable properties is condemned to produce occasionally irrational outcomes [117]. Consequently, the existing literatures on judgement aggregation are concerned with the unpleasant occurrence of irrational collective outcomes, see e.g., [48, 116, 117].

Belief merging is another closely related problem that formally investigates how to aggregate a finite number of individual belief bases into a collective one [117]. It defines a class of operators that produce collective beliefs from individual (and possibly conflicting) belief bases. Some of the merging operators proposed in the literature were inspired by some of the voting procedures that have been well studied in social choice theory. The theory of belief revision and merging has recently been applied to judgement aggregation, in which the individual’s decision does not inevitably come as a preference ordering, but has a propositional form, e.g., [117].

In addition to the investigation on the nature of the group decision-making problem, another concern in the field is the technical issues that need to be addressed in different group decision-making settings. These include the computational complexity of aggregation and voting rules, computational aspects of strategy-proofness and manipulation, compact preference representation and reasoning, incompleteness and incomparability in preference aggregation.
1.3 Research questions

The central problem addressed in this thesis is the design of efficient group decision-making mechanisms and approaches to support multiple agents reaching efficient and fair collective decisions on multi-attribute domains. The research work of this thesis is conducted according to the following research questions in agent-based group decision-making.

Environment

- In what domains do the agents need to make a decision? What are the types of issues involved, discrete or continuous? Are the issues interrelated or independent of each other?
- Given different problem domains, what preference representation models or languages are used to represent the agents’ preferences in group decision-making?
- Would the group decision-making process take place in a complete information setting, within which all the agents’ preferences are provided; or in an incomplete information setting, within which the agents’ preferences are not common knowledge?

Decision process

- According to what rule will the collective decision be made? (e.g., majority, minimax, egalitarian, etc.)
- Is the decision-making process a single-stage method that directly computes a collective decision from a collection of multiple agents’ preferences, or is it a multi-stage process that requires the agents to participate and jointly make a decision?

Properties of the chosen outcome

- Does the final chosen outcome satisfy the fundamental property Pareto-efficiency in group decision-making?
- How to measure fairness of the final chosen outcome? What are the fairness metrics with different preference representations in group decision-making?
1.4 Contributions

Computational issue

- Is the mechanism able to handle the cases in which the number of possible outcomes from which to choose is very large?

- What is the computational complexity of the mechanisms? What are the advantages of using the mechanism with regard to the computational aspect?

1.4 Contributions

The research work in this thesis can be divided into two parts based on different information settings, addressing the agent-based group decision-making problem under complete information setting and under incomplete information setting, respectively. The first part of this thesis focuses on preference aggregation in combinatorial domains, provided every agent’s preference. Then, in the second part of this thesis, negotiation approaches for group decision-making in combinatorial domains, and in the domains with multiple continuous issues, respectively, are investigated, with incomplete or limited information on the agents’ preferences. This thesis contains the following five main contributions (results).

Contribution 1: A penalty score-based heuristic for individual dominance testing in Conditional Preference Networks (CP-nets).

Contribution 2: A penalty score-based approach to preference aggregation with CP-nets and Trade-off enhanced CP-nets (TCP-nets).

Contribution 3: A majority-rule-based approach to preference aggregation with CP-nets.

Contribution 4: A protocol for negotiation in combinatorial domains with general preferences.

Contribution 5: A mediation approach to utility-based negotiation over multiple continuous issues.

Based on different problem settings investigated in the thesis, the work presented in this thesis is in close relation to the areas of preference aggregation in combinatorial...
domains (providing the agents’ preferences), and multi-issue negotiation (in the incomplete information setting). In the following pages, we give a more detailed introduction to these two fields, along with a more detailed discussion of the aforementioned five contributions of this thesis.

### 1.4.1 Preference aggregation in combinatorial domains

In many real life preference aggregation problems, the space of alternatives has a combinatorial structure: there are multiple issues (or attributes), each taking values from its respective domain, and an alternative is characterised by a combination of values that the issues take. Such domains are called combinatorial domains. Take the election of a committee (rather than a single candidate) as an example, wherein the number of possible committees is exponential in the number of seats to be filled. The decisions are not independent. As another example, consider a situation where the citizens of a country vote to directly determine the content of a government plan, composed of multiple sub-plans for several interrelated issues, such as transportation, environment, and health. Clearly, an agent’s preference for one issue generally depends on the decision taken on the other issues. For instance, suppose that the members of an association have to elect a steering committee, composed of a president, a vice-president and a treasurer [17]. Perhaps, the voters may not like the president and the treasurer to be close friends (or enemies). Hence we cannot decide on each position separately.

#### Preference representation and reasoning issues

Probably the first question arises in this field is how to express and reason about preferences in combinatorial domains. Because the space of possible alternatives in such domains has a size exponential in the number of variables, it is not reasonable to ask the agents to explicitly rank all alternatives or evaluate them on a utility scale. The literature on preference elicitation, representation and reasoning in combinatorial domains has been growing rapidly. Several logical or graphical compact representation languages have been developed, which aim to represent preference structures, the size of which would be prohibitive if represented explicitly, in as little space as possible. For example, CP-nets (Conditional Preference Networks) (Boutilier et al. [24]) is one of the most extensively studied languages. It provides a compact representation of preference ordering in terms of natural preference statements under a *ceteris paribus*
1.4. Contributions

(all else being equal) interpretation. Unfortunately, reasoning about the preference ordering (dominance relations) expressed by most of those compact preference representation languages is far from easy [24, 71]. For instance, with the exception of special cases such as CP-nets with tree or poly-tree structured conditional dependencies between variables, dominance testing has been shown to be NP-hard in general CP-nets [71].

Contribution 1: A penalty score-based heuristic for individual dominance testing in CP-nets. In the first part of this thesis, structured preferences CP-nets (Conditional Preference Networks), and its variant TCP-nets (Trade-off enhanced CP-nets), are used as a formal model for representing the agents’ preferences in group decision-making. Before going into the problem of making a group decision from a collection of CP-nets or TCP-nets, the problem of individual preference reasoning is revisited in Chapter 3. A computationally efficient heuristic algorithm for dominance testing in CP-nets, which forms the foundation for many group decision-making mechanisms discussed in the later chapters, is introduced. The proposed approach is based on a computationally efficient numerical transformation of CP-nets, i.e., compiling an individual CP-net into a penalty scoring function. Then the penalty scoring function is used as a heuristic of an algorithm, called DT*, for dominance testing in arbitrary acyclic multi-valued CP-nets.

Computational issues

The commonly studied methods for preference aggregation and classical results from social choice theory may not always be applicable when the number of alternatives from which to choose is large, especially when it involves a combinatorial structure. For example, a research group plan to order several personal computers (PCs) and the group members need to decide on a standard group PC configuration. Consider for example that the group members have to agree on a common PC configuration to be composed of processor, hard disk, RAM and operating system, with a choice of six possible options for each. This makes $6^4 = 1296$ candidates.

One straightforward way in which to aggregate preferences in combinatorial domains is issue-by-issue (a.k.a. seat-by-seat) voting. This requires that the agents explicitly express their preferences over each issue separately, and each issue is then decided by applying local (issue-wise) voting rules independently. This would not be a problem if
the four decision variables to choose were independent from each other. That is, if the preference of every agent over any issue is independent of the values taken by the other issues. In this case, the group decision-making problem over a set of 1296 candidates would come down to four independent problems over a set of six candidates each, and any standard voting rule could be applied without difficulty.

However, matters become more complicated when the decisions are not independent, because, perhaps, the preferred operating systems may depend on the given processor type. If an agent has non-separable preferences, it is not clear how he or she should vote in such an issue-by-issue election. For instance, “I prefer to choose the WinXP operating system rather than Linux if an Intel processor is given.”, etc. Hence, we cannot decide on the issues separately. Indeed, it has been shown that separating issues for voting in such a context can lead to very undesirable results. As soon as variables are not preferentially independent, it is very likely that deciding on the issue separately will lead to suboptimal choices [33]. The study of ”multiple election paradoxes” by Brams et al. [33] shows that decomposing a vote problem with \( m \) variables into a set of \( p \) smaller problems, each one bearing on a single variable leads to suboptimal choices. Brams et al. [33] give real-life examples of such paradoxes, including simultaneous referenda on related issues. They further argue that the only way of avoiding the paradox would entail ”deciding for combinations of variable values”.

There has recently been interest in computational properties of preference aggregation and voting in combinatorial domains [44, 92, 125]. All of the published results of studies show that when the set of alternatives has a combinatorial structure, aggregation or voting is a computationally difficult problem. Therefore, there is a need for the investigation of efficient mechanisms to support group decision-making in such domains. Moreover, since in combinatorial domains preferences are often described in a compact representation language, the existing literature focuses on operating directly on this language, generating neither the individual nor the aggregated preference relations explicitly.

In the first part of this thesis, with CP-nets and TCP-nets as the preference representation models, two computationally efficient approaches to group decision-making in combinatorial domains are proposed, based on different decision criteria.

**Contribution 2: A penalty score-based approach to preference aggregation with CP-nets and TCP-nets.** The first approach introduced (see Chapter 4) is based on a numerical transformation (the penalty scoring functions) of individual CP-
nets discussed in Chapter 3. Chapter 4 goes one step further by incorporating the relative importance information among pairs of variables. It introduces a penalty scoring function as a numerical approximation, not only for acyclic CP-nets, but also for conditionally acyclic TCP-nets. After the individual penalty scores are computed, a collective penalty scoring function is further defined as the basis for group decision-making with CP-nets and TCP-nets in combinatorial domains. In order to address the collective outcome selection problem in the exponential outcome space in combinatorial domains, a heuristic algorithm is also presented. Instead of having to conduct an exhaustive search in the huge alternative space, the proposed algorithm only needs to visit a small subset of outcomes and thus is computationally efficient.

Contribution 3: A majority-rule-based approach to preference aggregation with CP-nets. Another approach introduced in Chapter 5 considers the majority decision rule and represents the agents’ preferences by CP-nets. An efficient CSP-based (SAT-based) approach is introduced to compute the set of possible winners from a collection of CP-nets. The collective decision problem is first reduced to a CSP (Constraint Satisfaction Problem) or a SAT (Boolean Satisfiability Problem) for cardinality constraints. As such, the models of the corresponding CSP or SAT is a set of locally winning alternatives. A locally winning alternative is an alternative that assigns to each variable a non-dominated value given the values assigned to other variables by that alternative. Then, the set of possible winners is the subset of locally winning alternatives, after the filtering out of those that are majority-dominated by some alternative.

1.4.2 Multi-issue Negotiation

Concerning the disadvantage of disclosing preference information in group decision-making, another area of focus in this thesis is multi-issue negotiation. Multi-issue negotiation is one of the most effective techniques that enables self-interested agents to resolve conflicts and reach mutually beneficial agreements [47, 91, 122, 123, 140]. It is increasingly being used in different domains including agent-based trading systems (e.g., [3, 4, 86, 106]), policy negotiations (e.g., [5, 6, 119]), resource allocation, and service level agreement negotiations (e.g., [20, 42, 127]). In real world scenarios, situations in which multiple issues are involved in a negotiation simultaneously are common. Examples include, the price, quality attributes and delivery time in a supply
contract; or the transferability of water rights, degrees of environmental protection and new infrastructure development in a water usage policy; or the response time, levels of security and traceability in a service level agreement.

A multi-issue negotiation is a negotiation that involves multiple issues, all of which need to be negotiated simultaneously. In many situations, the negotiation agents are able to make trade-offs and search for possible joint gains. This means that they may increase their utilities by lowering their requirements on some negotiation issues that are not so important to them while demanding more on other more important issues, thus leading to an agreement that is mutually better [57]. Usually, multi-issue negotiation is characterised by the situations in which two or even more parties recognize not only the differences in their interest over multiple issues, but also the existing value of cooperation between them. Thus, they want to seek a compromise agreement [122]. For instance, consider a situation involving the selling of a laptop computer. A seller agent prefers a higher margin on the laptop price and is willing to provide an extended warranty, technical support and free delivery service which are of a lower cost for her. A buyer agent is also interested in such a package because she values highly the extended warranty and prefers free delivery. Furthermore, the after-sales service is very important to her. Such situations, in which all the parties are in a more favourable position, are normally called “win-win” solutions.

In a multi-issue negotiation, “win-win” solution means that because the agents may have different preferences on the issues, both sides may achieve a more advantages agreement on the issues that are most important to them by making trade-offs on those that they consider to be not so important. Multi-issue negotiation is widely used in solving many real-world decision-making problems. For instance, automated multi-issue negotiation is an important and valuable mechanism in the Navy detailing system, in order to realise efficient, distributed and “Win-Win” matching [90, 98]. In the following section, we list some challenges met in most multi-issue negotiation problems.

**Challenges**

In multi-issue negotiation, the agents should intuitively have a common interest in cooperating and searching for possible win-win solutions. However, empirical evidence suggests that self-interested agents often fail to reach consensus or end up with inefficient results [58, 62, 96, 122, 139, 140]. In the following section, we analyse the main
1.4. Contributions

challenges and issues that we need to cope with in multi-issue negotiation.

- **Complex preference.** In a multi-issue negotiation, the preference of an agent over multiple issues is complex and the outcome space is $m$-dimensional ($m > 1$) rather than a single-dimension line as in a single-issue negotiation. This makes the negotiation strategy in multi-issue negotiations complex: if an agent plans to concede, he or she needs to first decide on the direction of concession in a $m$-dimensional outcome space. Moreover, the decision on the concession direction also depends on the opponents’ preferences, because conceding on the issues that are more important to the opponents can make the offer more acceptable.

- **Incomplete information.** Increasing efficiency and fairness in multi-issue negotiation requires agents to share preference information. However, the disclosure of an agent’s preference to their opponents puts it at a disadvantage in a negotiation. Therefore, in most real world applications, agents’ preferences are not common knowledge. This means that agents do not have complete information about their opponents’ preferences, and they might not want to completely reveal their preferences to their opponents during the process of negotiation. Under incomplete information, the burden of computation and reasoning for the negotiation strategy becomes even heavier, and it is thus difficult to reach efficient and fair outcomes.

- **Computational complexity.** In multi-issue negotiation, the complexity of the negotiation process is mainly determined by the complexity of the agents’ preference languages or utility functions. For instance, the case that the agent’s utility function is a weighted sum of utility evaluation functions of each issue is tractable. This model of issue variables influences overall utility independent of other issues. Thus, it allows for the use of efficient negotiation strategies. However, in some domains, issue dependencies influence the overall utility of a bid. In such cases there is no efficient method that an agent can use to negotiate multiple issues, even if an agent tries to guess the opponents’ preferences [82].

The advantages and challenges of multi-issue negotiations have been recognised for a long time. In particular, Raiffa [122] describes multi-issue negotiation in the following terms “It’s no longer true that if one party gets more, the other necessarily has to get less: they both can get more.” However, the frequent failure of the negotiators to achieve efficient agreements in practice is also well studied in [122]. Other studies include, for instance, the discussion by Lax and Sebenius on the Negotiator’s Dilemma
in deciding whether to pursue a cooperative or a competitive strategy at a particular time during negotiation [96]. Fatima et al. [62, 63] point out that self-interested agents would like to reach an agreement that is as favourable to them as possible, whereas the final decision is jointly made and needs to be agreed to by all the agents. These studies and results indicate that the problems met by negotiation agents are not only to choose cooperative or competitive strategies, but also to consider how much they could gain individually if they cooperated and through which form of cooperation they could gain more, or at least receive a fair deal. Multi-issue negotiation, therefore, requires techniques that deal with rational agents with fairness and lead them to mutually beneficial agreements.

In the first part of this thesis, we mainly study the problem of group decision-making where the agents’ preferences are assumed given, which may not be reasonable in some real world circumstances. In the second part of this thesis, we subsequently relax this assumption and consider the case where complete information is not available. Two negotiation techniques are investigated in Chapter 6 and Chapter 7, respectively, for the problem of making group decisions over multiple issues under incomplete information setting. While Chapter 6 introduces a distributed negotiation protocol that only works in discrete issue domains, the method discussed in Chapter 7 further considers continuous domains and uses a mediation approach.

**Contribution 4: A protocol for negotiation in combinatorial domains with general preferences.** In Chapter 4 and Chapter 5, the agents’ preferences are assumed given in order to compute the joint collective decision. However, in some circumstances, asking agents to submit their preferences fully is not feasible. Thus, a natural question may arise: how can we choose collective decisions when the agents are not willing to reveal their preferences completely? Subsequently, we continue the problem of group decision-making in combinatorial domains in Chapter 6, while relax the assumption of complete information. In Chapter 6, we propose a negotiation protocol in combinatorial domains, which allows the agents to distributively make decisions and leads them to Pareto-optimal agreements under incomplete information setting. It allows the agents to visit only a small subspace of the whole outcome space, and thus is computationally efficient. Moreover, the proposed protocol differs from most of the existing research in the field of utility-based negotiation, because it not only can handle quantitative preferences, but also can work with purely qualitative preference models. The proposed protocol is sufficiently general that it is applicable to most prefer-
1.5. Chapter Outlines

What follows is a chapter-by-chapter outline of the thesis. Chapter 2 provides necessary background knowledge on group decision-making. The related work is also provided in Chapter 2 containing several results most relevant to the work presented in this thesis. The main contribution of this thesis will be presented in two technical parts, taking into account different information settings (complete and incomplete information) in group decision-making. We elaborate below on the structure of this partition and the results given therein.
Part I: Preference aggregation in combinatorial domains.

The first part of the thesis mainly investigates techniques and mechanisms for preference aggregation over combinatorial domains, proving complete information about the agents’ preferences. The theory of CP-net (Conditional Preference Network) and its variant, TCP-net (Trade-offs-enhanced Conditional Preference Network), are used as the formal models for representing and reasoning with the individual agents’ preferences. Three main results are presented, including a heuristic approach for individual preference reasoning and two approaches for preference aggregation in combinatorial domains.

Chapter 3: Penalty score-based heuristic for dominance testing in CP-nets. Before going into the problem of aggregating multiple agents’ CP-nets, a fundamental problem of individual preference reasoning in CP-nets, namely dominance testing, is investigated in this chapter. The existing literature formulates dominance queries as a search of the improving flipping sequence from one outcome to another, which provides a proof of the dominance relation in all rankings satisfying the given CP-net. However, it is generally a difficult problem even for binary-valued, acyclic CP-nets, and tractable search algorithms exist only for specific problem classes. Hence, there is a need for efficient algorithms and techniques to guide the search process for dominance testing in more general problem settings. To this end, this chapter first presents an efficient method that compiles an individual CP-net into a penalty scoring function as a numerical approximation for the CP-net. As such, the penalty scoring function preserves the strict preference ordering induced by the original CP-net. Then, it further utilises the penalty scoring function as a heuristic of an algorithm, called $D^T*$, to answer dominance testing queries in arbitrary acyclic multi-valued CP-nets. The proposed approach guides the search process for dominance testing efficiently, and allows significant reduction of search effort without impacting the completeness of the search process.

Part of this work has been published in *Proceedings of the Tenth International Conference on Autonomous Agents and Multi-agent Systems - Volume 1*, AAMAS 2011, pages 353–360 (see [102]).

Chapter 4: Penalty score-based preference aggregation with CP-nets and TCP-nets. The preference information provided by a CP-net or a TCP-net on its own is purely qualitative. However, many group decision-making methods require numerical measures of degrees of desirability of alternative outcomes. Chapter 4 extends the penalty scoring function for CP-nets introduced in Chapter 3, and
introduces a penalty scoring function as a numerical approximation, not only for acyclic CP-nets, but also for conditionally acyclic TCP-nets. After the individual penalty scores are computed, a collective penalty scoring function is further defined in this chapter, in order to represent the preference of a group of agents. Finally, the problem of computing an optimal collective outcome in the exponential outcome space is also addressed by means of a proposed efficient heuristic algorithm, called HeTCP. HeTCP efficiently searches for an optimal collective decision according to a given aggregation rule, instead of exhaustively listing all possible alternatives.

Part of this work has been published in *Proceedings of the Tenth International Conference on Autonomous Agents and Multi-agent Systems - Volume 2, AAMAS 2011*, pages 1073–1074 (see [103]) and *Proceedings of the 19th European Conference on Artificial Intelligence, ECAI 2010*, pages 375 – 380 (see [101]).

**Chapter 5: Majority-rule-based preference aggregation with CP-nets.** Chapter 5 studies another preference aggregation scenario, in which majority rule is considered to be the decision criterion in group decision-making, and it is assumed that the agents’ preferences are represented by CP-nets. An efficient CSP-based (SAT-based) approach, called MajCP (Majority-rule-based collective decision-making with CP-nets), is presented, in order to compute the possible majority winning alternatives from a collection of CP-nets. The proposed approach allows the agents to have different preferential independence structures, and is able to aggregate preferences even when the agents’ CP-nets are cyclic. With multiple agents’ CP-nets as input, it first reduces the problem to a CSP (Constraint Satisfaction Problem) or SAT (Boolean Satisfiability Problem) for cardinality constraints. As such, the models of the corresponding CSP or SAT are a set of locally winning alternatives. Finally the set of possible winners is the subset of the locally winning alternatives, after filtering out those that are majority-dominated by some alternative. The proposed approach substantially reduces the search space and is computationally efficient.

Part of this work has been published in *Proceedings of the Tenth International Conference on Autonomous Agents and Multi-agent Systems - Volume 1, AAMAS 2011*, pages 659–666 (see [104]) and *Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning, KR2010*, pages 578–580 (see [100]).

**Part II: Multi-issue negotiation with incomplete information.**

The second part of this thesis studies the negotiation problems over multiple issues
under incomplete information setting, i.e., the agents’ preferences are not common knowledge. It contains two other contributions of this thesis, addressing the problem of negotiation in combinatorial domains and negotiation over multiple continuous issues, respectively.

**Chapter 6: Negotiation in combinatorial domains with general preferences.** The techniques that were introduced in the first part of this thesis work only with CP-nets and TCP-nets. The mechanisms are based on the assumption that complete information of every agent’s preference is available, and the decision-making is achieved through centralised computation from a collection of agents’ preferences. In Chapter 6, a scenario where the agents’ preferences are not common knowledge is considered. A protocol for negotiation in combinatorial domains is proposed, which enables the agents to distributively make decisions and leads them to Pareto-optimal agreements. The proposed protocol works even when the agents do not have prior knowledge of the opponent’s preferences and they are not willing to reveal their preferences during the process of negotiation. Moreover, it allows the agents to visit only a small subspace of the whole outcome space, and is thus computationally efficient. Last but not least, the proposed protocol is sufficiently general that it is applicable to most preference representation models in combinatorial domains.

Part of this work has been published in *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, - Volume 2, UAI 2011, pages 436 – 444 (see [103]) and *Proceedings of the 19th European Conference on Artificial Intelligence*, ECAI 2010, pages 375 – 380, (see [101]).

**Chapter 7: Utility-based mediated negotiation over multiple continuous issues.**

Chapter 7 further studies the negotiation problem over the domains with multiple continuous issues. A classical representation model, utility, is used to represent the agents’ preference and a status quo (pre-negotiated outcome or initial agreement) is assumed to be given. In order to increase the efficiency from the status quo, Chapter 7 employs a trusted third party, the mediator, to support the negotiating agents in searching for possible joint gains, while protecting them from unnecessary disclosure of information to their opponents. At each stage of negotiation, the mediator chooses a fair compromise direction that improves all the negotiating agents’ utilities. Subsequently, it determines a new mutually preferred tentative agreement in this compromise direction for the negotiating agents to take into consideration. This process continues until an efficient outcome is achieved. The aim of this approach is to search for the mutually preferred out-
come, which minimises the difference between the agents’ utility gains from the status quo, leading to fair agreements.

Part of this work has been published in Proceedings of the Eighth International Conference on Autonomous Agents and Multi-agent Systems - Volume 2, AAMAS 2009, pages 1049–1056 (see [99]).

Finally, Chapter 8 summarises the thesis with a number of concluding remarks, and then outlines some future research issues.
Chapter 2

Background

The aim of this chapter is to introduce the formal mathematical notations and to provide the necessary background knowledge that will serve us throughout this thesis. The relevant literature is also provided containing results most relevant to the work presented in this thesis.

2.1 The basic components

An agent-based group decision-making scenario consists of:

- Agents (*aka.* decision makers): in a group decision-making problem, there will be a set of decision-makers (or agents) who are involved in the discussion of a group decision. The participating agents are denoted by $j, (j = 1, ..., n)$.

- Attributes (*aka.* issues, variables, criterions): Attributes are the issues that the agents are making decisions over, such as price, length of warranty, delivery time, and delivery fee in a supply contract; or departure date, departure time, airline company, hotel in a group travel plan, etc.. Let $V = \{X_1, X_2, \ldots, X_m\}$ be a set of attributes, each attribute $X$ can take a value $x$ such as ‘$1500’ or ‘3 years’ from its corresponding interval $D(X)$ ($x \in D(X)$). The attributes can be discrete (the attributes that can take a finite set of values), or continuous (the attributes that can take values in some interval).
2.2. The domain

- Outcomes: an outcome (aka. alternative, solution) o is represented by a complete assignment, i.e., a combination of values of all variables in the corresponding domain V: o = (x₁, x₂...xₘ) and o ∈ O, e.g., o = ([price =]$2500; [warranty = ]3 years; [delivery = ]10 days; [delivery fee = ]$10).

The set of all possible outcomes is denoted by O. We may alternatively call the outcome space or alternative space throughout this thesis. The elements of this set are mutually exclusive. That means, choice of one implies the rejection of the others. For example, O can represent the set of candidates in an election and the agents needs to choose for whom to vote; Or it can be a set of water usage policy solutions, and different countries or departments must negotiate with each other and finally agree on one to fulfil.

- Preference: each agent (or decision maker) will have its own preference over the outcomes. The preference indicates the ranking (or order, precedence) of possible outcomes based on relative satisfaction they could provide for an agent. Preference can be modelled by preference relations, utility or other representation languages.

2.2 The domain

The research work in this thesis are focus on the group decision-making problems in combinatorial domains, however, we also further investigate the domains with multiple continuous issues in Chapter 7.

Combinatorial domains

Many real world problem involves a combinatorial domain, i.e., the set of possible alternatives has a combinatorial structure. For instance, negotiation over multiple indivisible goods or an election of a committee. A combinatorial domain is a Cartesian product of finite value domains for each one of a set of variables. A possible alternative in such a domain is a tuple of values (a combination of variable values). In combinatorial domains, the number of possible alternatives is exponential in the number of variables. For instance, in a negotiation over multiple indivisible goods, the number of bundles an agent may obtain is exponential in the number of goods; or in the election
of a committee, the number of possible committees is exponential in the number of seats to be filled.

Formally, let $V = \{X_1, \ldots, X_m\}$ be a set of $m$ attributes in a combinatorial domain. For each $X \in V$, $D(X)$ is the domain of $X$. A variable $X$ is binary if $D(X) = \{x, \bar{x}\}$. An alternative (or outcome) is uniquely identified by the combination of its attribute values, i.e., a complete assignment of all variables in $V$. Hence, there are $D(X_1) \times \cdots \times D(X_m)$ possible alternatives (outcomes), denoted by $O$. Elements of $O$ are denoted by $o, o', o''$ etc. and represented by concatenating the values of the variables. For example, if $V = \{A, B, C\}$, $D(A) = \{a, \bar{a}\}$, $D(B) = \{b, \bar{b}\}$ and $D(C) = \{c, \bar{c}\}$, then the assignment $a\bar{b}c$ assigns $a$ to variable $A$, $\bar{b}$ to $B$ and $c$ to $C$. If $X = \{X_{p_1}, \ldots, X_{p_l}\} \subseteq V$, with $p_1 < \cdots < p_l$ then $D(X)$ denotes $D(X_{p_1}) \times \cdots \times D(X_{p_l})$ and $x (x \in D(X))$ denotes an assignment of variable values to $X$. If $X = V$, $x$ is a complete assignment (corresponds to a possible outcome); otherwise $x$ is called a partial assignment. Note that a complete assignment corresponds to an outcome $o \in O$. If $x$ and $y$ are assignments to disjoint sets $X$ and $Y$, respectively ($X \cap Y = \emptyset$), we denote the combination of $x$ and $y$ by $xy$. If $X \cup Y = V$, we call $xy$ a completion of assignment $x$. We denote by $\text{Comp}(x)$ the set of completions of $x$. For any assignment $x \in D(X)$, we denote by $x[X]$ the value $x \in D(X)$ ($X \in X$) assigned to variable $X$ by that assignment; and $x[W]$ denotes the assignment of variable values $w \in D(W)$ assigned to the set of variables $W \subseteq X$ by that assignment. We allow concatenations of partial assignments. For instance, let $V = \{A, B, C, D, E, F\}$, $X = \{A, B, C\}$, $Y = \{D, E\}$, $x \in D(X)$, $y \in D(Y)$, $x = a\bar{b}c$ and $y = \bar{d}e$, then $xy\bar{f}$ denotes the alternative $a\bar{b}c\bar{d}e\bar{f}$.

### 2.3 Preference

In group decision-making, the participating agents need to collaborate in the decision-making process, and a consensus amongst all participating agents needs to be finally reached. In fact, most problem scenarios are over-constrained and the agents would not be able to solve the conflicts in their society if every agent insists that all of his requirements need to be strictly met. Therefore, many problems are more naturally described via preferences rather than hard statements. Moreover, different solutions (or outcomes) to a solvable problem have different desirability and different participating agents usually have different or even conflicting preferences over the set of possible
alternatives or outcomes. The preference of an agent refers to the set of assumptions related to ordering some alternatives, based on the degree of happiness, satisfaction, gratification, enjoyment, or utility they provide for that agent [11].

2.3. Preference

2.3.1 Preference relation

The standard way to model an agent’s preference in a group decision-making problem is with his preference relation, also called a binary relation [80]. The preference relation on the set of possible outcomes \( O \) of an agent indicates the ranking (or order, precedence) of possible outcomes (or alternatives, solutions), representing the relative merits of any two outcomes for that agent with respect to some criteria [80]. For instance, let \( o, o' \) be two possible outcomes \( o, o' \in O \), a relation "is preferred to" (or "is better than"), denoted by \( R \), can be defined on the set of candidates, and interpreted as "candidate \( o \) is preferred to \( o' \)" whenever we write \( oR o' \). Through out this thesis, we use the following notation to denote strict and weak preferences. Let \( \succsim \) is a preference relation on \( O \) such that \( o \succsim o' \) if and only if \( o \) is at least as preferable as \( o' \) (or, \( o \) is weakly preferred to \( o' \)). And \( o \) is strictly preferred to \( o' \), denoted by \( o \succ o' \), if and only if \( o \succsim o' \) but \( o' \nsucc o \). Notice the following logical implications that \( o \succ o' \Leftrightarrow o \succsim o' \land o' \nsucc o \).

When \( o \succsim o' \) and \( o' \succsim o \), we say that the agent is indifferent between these two outcomes \( o \) and \( o' \), denoted by \( o \sim o' \). Moreover, in some situations that the agent can not determine the preference relation between \( o \) and \( o' \) \((o \nnsuccsim o' \land o' \nsuccsim o)\), we say \( o \) and \( o' \) are incomparable for that agent, denoted by \( o \nsim o' \).

Consequently, suppose we present an agent with two alternatives \( o \) and \( o' \), and ask him to rank them according to some criterion. There are four possible answers we can get:

1. \( o \) is better than \( o' \) \((o \succ o')\);  
2. \( o' \) is better than \( o \) \((o' \succ o)\);  
3. \( o \) and \( o' \) are indifferent \((o \sim o')\);  
4. \( o \) and \( o' \) are incomparable, \( o \nnsucc o' \land o' \nnsucc o \land o \sim o' \).  

The fourth relation is also logically equivalent to \( o \nnsucc o' \land o' \nnsucc o \).


2.3.2 Utility

While the preference relation is usually a conventional foundation to examine the behaviour of an agent, it is often convenient to represent preference with a utility function and reason indirectly about the preference using the utility [134]. Utility (cardinal utility) is a quantitative measure of the agent’s relative satisfaction with a particular outcome or solution. It is often used to represent an agent’s preference in the domains with continuous issues. A utility function of an agent is a function mapping from possible alternatives to the real numbers (\( \mathbb{R} \)), such that the preference relation over the alternative space is preserved. More formally [80],

\[
\text{Definition 2.1} \quad \text{A function } u : O \rightarrow \mathbb{R} \text{ is a utility function representing an individual agent's preference relation } \preceq \text{ if the following holds for all } o, o' \in O: \quad u(o) \geq u(o') \iff o \succeq o'.
\]

Utility function is often modelled based on the agent’s value function and some group-specific regulations or factors, such as participation fees, transaction costs, etc.

Figure 2.1 shows an example of a seller agent’s utility function with a single issues price. For instance, the utility of the seller agent is increasing as the selling price increases. When the product can be sold at 200 dollars, his utility is 0.6. And when he sells the product at 400 dollars, he gets a higher utility 0.8.
2.3. Preference

Notice that there are many utility functions that can represent the same preference relation, because preferences on their own are ordinal. That is, preference relations only specify the ranking of the alternatives, but not how far apart they are from each other. However, a utility function assigns numerical values to various alternatives, the magnitude of any differences in the utility values between two alternatives is cardinal. Given this measure, one may speak meaningfully of increasing or decreasing utility, and explain to what degree an outcome is better than another one based on the difference between the utility values of those two outcomes. This is because the magnitude of any differences in the utility between two alternatives represents a significant quantity difference between the degree of satisfaction provided by these two alternatives.

Utility provides a quantitative measurement to capture the users’ desires over different outcomes, and thus makes it easy to compare between outcomes. Given two alternatives \( o \) and \( o' \), if \( u(o) > u(o') \) then we can say \( o \succ o' \); else if \( u(o') > u(o) \) then we can say \( o' \succ o \); finally, if \( u(o') = u(o) \) then we can say \( o' \sim o \) (there does not exist incomparability when we represent preferences by means of utility). Moreover, instead of examining conditions under which preference relations produce maximal elements for a set of alternatives, utility also provides a convenient way for answering outcome optimization queries. It is easier to specify the numerical representation and then apply standard optimization techniques to find the maximum. Then, the "best" alternative outcomes from the set \( O \) are precisely the outcomes that have the maximum utility.

2.3.3 Graphical representations CP-net and TCP-net

In classical decision theory and decision analysis, the agents’ preferences are represented mathematically by utility functions. Utility function is a powerful form of knowledge representation and outcome comparison is relatively easy by means of comparing the values of utility. Nonetheless, utility is a device to represent preferences rather than something from which preferences come. Moreover, in many situations, the utility-based preference elicitation is complicated and typical users may not be able to provide much more than qualitative rankings of outcomes [26]. Last but not least, conditional preferences are easier to represent in a qualitative way rather than by quantitative preference [24]. Therefore, when a utility function is unavailable or unnecessary, one can resort to other, more qualitative forms of preference representation.

\[^1\text{Cardinal properties are those that are only preserved under strictly increasing transformations.}\]
Researchers in social choice theory have extensively studied the properties of voting rules and aggregation functions for group decision-making problems, up to an important detail: candidates are supposed to be ranked explicitly. That means, each individual voter is supposed to give out an explicit ranking over all possible candidates, based on the assumption that they are not too numerous. Unfortunately, many real world group decision-making problems involve a complex combinatorial structure, where the size of the alternative space grows exponentially with the set of variables and becomes quickly very large. This makes explicit representations and straightforward elicitation and optimisation no longer reasonable. Consider for example that a group of agents have to agree on a common PC configuration setting to be composed of a processor, a CPU, a screen and a RAM, with a choice of 6 items for each. This makes $6^4 = 1296$ candidates.

Still, this would not be a problem if each of the four items to be decided on were independent from the other ones. In this case, the decision over a set of 64 candidates would come down to four independent decision problems over sets of 6 candidates each, and any standard voting rule or aggregation mechanism could be applied without difficulty. However, imagine that the decisions are not independent, because, perhaps, the preferred operation systems may depend on the given processor type. For instance, “I prefer to choose WinXP operation system rather than Linux if an Intel processor is given.” Hence, we cannot decide on the issues separately. Because the issues in such domains are interdependent and the preference structure of each agent cannot be reasonably expressed by listing all candidates, what is needed is a compact preference representation language.

The problems of eliciting, representing and reasoning with qualitative preferences over multi-attribute domains arise in many fields such as planning, design, and collective decision-making [71, 88, 93, 125]. Several compact, succinct qualitative preference representation models have been developed in AI community in recent years. They escape the combinatorial blow up of explicit representations by exploiting structural properties such as conditional preferential independence, and thus make it easier to encode human preferences and support the decision-making systems in real world applications. The following notions of preferential independence and relative importance are from the work of Boutilier and colleagues [24, 30].
2.3. Preference

Preferential independency

Let $X$ and $Y$ be nonempty sets that partition $V$ and $\succ$ a preference relation over $D(V)$. We say $X$ is preferentially independent of $Y$ if and only if, for all $x, x' \in D(X), y, y' \in D(Y)$:

$$xy \succ x'y \iff xy' \succ x'y'$$

If this relation holds, we say that $x$ is preferred to $x'$ ceteris paribus (all else being equal).

Let $X, Y, \text{ and } Z$ be nonempty sets that partition $V$ and $\succ$ a preference relation over $D(V)$. We say $X$ is conditionally preferentially independent of $Y$ given $z \in D(Z)$ if and only if, for all $x, x' \in D(X), y, y' \in D(Y)$:

$$xyz \succ x'yz \iff xy'z \succ x'y'z$$

If this relation holds, we say that $x$ is preferred to $x'$ given $z$ ceteris paribus. If $X$ is conditionally preferentially independent of $Y$ for all $z \in D(Z)$, then $X$ is conditionally preferentially independent of $Y$ given the set of variables $Z$.

Relative importance

Let $X$ and $Y$ be a pair of variables from $V$ that are mutually preferentially independent given $W = V - \{X, Y\}$. $X$ is more important than $Y$, denoted by $X \succ Y$, if for every assignment $w \in D(W)$ and for every $x, x' \in D(X), y, y' \in D(Y)$, such that $x \succ x'$ given $w$, we have that:

$$xyw \succ x'y'w$$

Let $X$ and $Y$ be a pair of variables from $V$, and $Z \subseteq V - \{X, Y\}$ and let $W = V - \{X, Y\} - Z$, $X$ is more important than $Y$ ($X \succ Y$) given $z \in D(Z)$ (also written as $X \succ_z Y$) iff, for all $w \in D(W)$ and for every $x, x' \in D(X), y, y' \in D(Y)$, such that $x \succ x'$ given $zw$, we have that:

$$xyzw \succ x'y'zw$$

Finally, if for some $z \in D(Z)$, either $X \succ_z Y$, or $Y \succ_z X$, then the relative importance
Chapter 2. Background

Among those compact representation languages developed recently, the formalism CP-net (Conditional Preference Network) is one of the most extensively studied languages for representing and reasoning with preferences in combinatorial domains. Most group decision-making mechanisms proposed in this thesis consider CP-net and its variant TCP-net (Trade-offs-enhanced CP-net) as the formal models to represent agents’ preferences. We hereby provide some necessary background knowledge of the preference representation models CP-net and TCP-net.

2.3.3.1 CP-net

A CP-net $\mathcal{N}$ [24] over a set of domain attribute $V = \{X_1, \ldots, X_m\}$ is an annotated directed graph $G$, in which nodes stand for the problem attributes. Each node $X$ is annotated with a conditional preference table $CPT(X)$, which associates a total order $\succ^{X|u}$ with each instantiation $u$ of $X$’s parents $Pa(X)$, i.e. $u \in D(Pa(X))$. For instance, let $V = \{A, B, C\}$, all three being binary, and assume that the preference of a given agent over all possible outcomes can be defined by a CP-net whose structural part is the directed acyclic graph with three directed arcs: $(\overline{C}, \overline{A})$, $(\overline{C}, \overline{B})$, and $(\overline{A}, \overline{B})$. This means that the agent’s preference over the values of $C$ is unconditional, preference over the values of $A$ (resp. $B$) is fully determined given the value of $C$ (resp. the values of $C$ and $A$). The preference statements contained in the conditional preference tables are written with the usual notation, that is, $\overline{a}c : \overline{b} \succ b$ means that when $A = \overline{a}$ and $C = c$, then $B = \overline{b}$ is preferred to $B = b$ (see for example Figure 2.2).

![Figure 2.2: An example of CP-net](image)
2.3. Preference

Figure 2.3: Preference ordering induced by the CP-net in Figure 2.2

Note that the preference relation induced from a CP-net is generally not complete (a partial order) [24]. That means, over the alternative space there may be more than one total preference ranking that satisfy a CP-net. In most literature, the graph $G$ is assumed to be acyclic, i.e., the relation graph does not contain cycle. Under this assumption, the induced preference relation of a CP-net is a strict (asymmetric) partial order over the possible alternatives (see for example, the preference ordering over the alternative space in Figure 2.3 induced by the CP-net in Figure 2.2).

Notice that CP-nets are not fully expressive, because some preference relations are not expressible by CP-nets [41, 51]. On the positive side, preferences expressed as CP-nets are easy to elicit, provided that the relation graph is known and possesses a small enough node in-degree. Moreover, the space needed to specify a CP-net is exactly the cumulated size of its tables, whereas the explicit representation of the preference relation is always exponentially large. Finally, outcome optimization is computationally easy, provided that the network is acyclic.

2.3.3.2 TCP-net

CP-nets support reasoning with qualitative conditional preference statements, which are relatively natural for users to express. Nonetheless, the orderings derived from the variable dependency and preference statements of a CP-net alone are relatively weak. To understand this, consider a simple CP-net consisting of two preferentially independent Boolean variables $A$ and $B$ with values $a, \overline{a}$ and $b, \overline{b}$, respectively. Because $A$ and $B$ are preferentially independent, we can specify a preference order over $A$ values, say $a \succ \overline{a}$, independently of the value of $B$. Similarly, our preference over $B$ values, say $b \succ \overline{b}$, is independent of the value of $A$. From the preference information provided by this simple CP-net, we can deduce that $ab$ is the optimal outcome and $\overline{ab}$ is the least preferred outcome. However, there is no clear evidence about the preference relation between the alternatives $\overline{ab}$ and $\overline{a}b$. It is a common case when we have independent variables: we can rank each one given a fixed value of the other, but often, we cannot
compare outcomes in which both values are different [31].

One possible solution to address some (though not necessarily all) such comparisons is to provide further information about relative importance relation between variables. In the previous example, if we have $A$ is more important than $B$, such that obtaining a better value for $A$ is more important (preferred to) obtaining a better value for $B$. In that case, we can know that $a b > a b$, and thus resulting in a total order over the set of all possible alternatives: $a b > a b > a b$.

TCP-nets (Trade-offs-enhanced CP-nets) [30], maintain the spirit of CP-nets, i.e., using only very intuitive qualitative information, by capturing information about both conditional preferential independence and conditional relative importance. As such, the formalism TCP-nets provides a richer framework to better model tradeoffs between variables that the users would like to make and allows stronger conclusions to be drawn [30].

A TCP-net $N$ [30] is a preference-representation structure, which extends the CP-net by further incorporating the relative importance relation between variables. The same as that in a CP-net, the nodes of a TCP-net are the domain variables. There are three sets of arcs between variables: $cp$, $i$ and $ci$. The first set of arcs $cp$ comes from the original CP-nets model. It captures direct preferential dependencies. $cp$ denotes a set of directed $cp$-arcs ($cp$ stands for conditional preference). A $cp$-arc $(X, Y)$ is in $N$ iff the preferences over the values of $Y$ depend on the actual value of $X$. Similarly, we called $X$ is a parent variable of $Y$. Each variable $Y$ is then annotated with a conditional preference table $CPT(Y)$, which associates a total order $\succ^Y u$ with each instantiation $u$ of $Y$’s parents $Pa(Y)$, i.e. $u \in D(Pa(Y))$. $i$ is a set of directed $i$-arcs (where $i$ stands for importance). An $i$-arc $(X, Y)$ is in $N$ iff $X$ is unconditionally more important than $Y$, i.e., $X \succ Y$. $ci$ is a set of undirected $ci$-arcs (where $ci$ stands for conditional importance). A $ci$-arc $(X, Y)$ is in $N$ iff we have $RI(X, Y|Z)$ for some $Z \subseteq V - \{X, Y\}$ and $Z$ is called the selector set of $(X, Y)$. We denote the selector set of a $ci$-arc $\gamma = (X, Y)$ by $S(\gamma)$ and the union of the selector set in a TCP-net $N$ by $S(N)$. Each $ci$-arc $\gamma = (X, Y)$ is associated with a conditional importance table $CIT(\gamma)$ from every instantiation of $s \in D(S(\gamma))$ to the orders over the set $\{X, Y\}$. A TCP-net in which the sets $i$ and $ci$ (and therefore, the conditional importance tables) are empty, is also a CP-net. Figure 2.4 shows an example of TCP-net, whose structural part is the graph with a $cp$-arc $(A, C)$, an $i$-arc $(C, D)$ and a $ci$-arc $(B, C)$. This TCP-net demonstrate that agent 1’s preference over the values of $A$ is unconditional, preference over the values of $C$ is fully determined given the values of $A$. The pref-
2.3. Preference

Figure 2.4: An example of TCP-net $\mathcal{N}$

![Diagram](image1)

Figure 2.5: Dependency graph and its $w$-directed graphs of $\mathcal{N}$

(a) Dependency graph $\mathcal{N}^*$  
(b) $a$-directed graph  
(c) $\bar{a}$-directed graph

Preference statements contained in the CPTs are written with the usual notation. That is, $a : c \succ c$ means that when $A = a$ then $C = c$ is preferred to $C = \bar{c}$. Moreover, variable $C$ is unconditionally more important than $D$, while the importance relation between $B$ and $C$ is fully determined given the values of $A$. The relative importance statements contained in the CITs are written with the following notation: $a : C \triangleright B$ means that when $A = a$ then $C$ is more important than $B$ ($C \triangleright B$), i.e., $C \triangleright_a B$.

Similar to CP-net, a TCP-net induces a partial ordering over the alternative space. There may be more than one total preference rankings satisfy a TCP-net. The work on group decision-making with TCP-nets in this thesis only focuses on a class of conditionally acyclic TCP-nets. Conditionally acyclic TCP-nets are guaranteed to induce a consistent preference ordering over the alternative space (i.e., no cycles in the induced preference ordering over the alternative space). As to examine the preference structure induced by a TCP-net, we first recall the following notion of the dependency graph of a TCP-net introduced by Brafman et al. [30]:
**Definition 2.2 (Dependency graph)** The dependency graph $N^*$ of a TCP-net $N$ contains all the nodes and arcs of $N$, and for every ci-arc $\gamma = (X, Y)$ in $N$ and every variable $Z \in S(\gamma)$, $N^*$ contains a directed arc $\langle Z, \bar{X} \rangle$ (resp. $\langle Z, \bar{Y} \rangle$), if there is no arc between $Z$ and $X$ (resp. $Z$ and $Y$) in $N$. Throughout this thesis, we call such a directed arc from a variable $Z$ in the selection set to the variable $X$ (resp. $Y$) that has importance relation conditioned on $Z$ as a sci-arc. Moreover, we denote sci as the set of sci-arcs in $N^*$.

Figure 2.5(a) shows an example of dependency graph of the TCP-net in Figure 2.4. It contains all the nodes and arcs of $N$ in Figure 2.4. In addition, as there is a ci-arc $\gamma = (B, C)$ in $N$ and $S(\gamma) = \{A\}$, there is a directed sci-arc $\langle A, \bar{B} \rangle$ in $N^*$. Note that there is no sci-arc $\langle A, \bar{C} \rangle$ in $N^*$ because there already exists an cp-arc $\langle A, \bar{C} \rangle$ in $N$. Recall that $S(N)$ denotes the union of the selector sets in TCP-net $N$. Once we assign a value to the union selector set $S(N)$ (denoted by $w \in D(S(N))$), we are, in essence, orienting all the conditional importance edges in the TCP-net $N$. That is, once given an assignment to the set of variables $S(N)$, there is a unique direction in every ci-arc in $N$. More generally, once all selector sets are assigned, we transform both $N$ and $N^*$. The following definition of the $w$-directed graphs of a TCP-net is introduced by Brafman et al. [30]:

**Definition 2.3** For a TCP-net $N$, let $N^*$ denote the dependency graph of $N$. Given an assignment $w$ to the set of variables $S(N)$, the $w$-directed graph of $N^*$ contains all the nodes and directed arcs of $N^*$ (including cp-arcs, i-arcs and sci-arcs). In addition, for every undirected ci-arc $\gamma = (X, Y)$ in $N^*$, there is a directed edge $\langle X, \bar{Y} \rangle$ if the CIT for $(X, Y)$ specifies that $X \triangleright Y$ given $z$ (i.e., $X \triangleright_a Y$), where $z$ is the projection of $w$ to the set of variables $S(\gamma)$.

For instance, Figure 2.5(b) and Figure 2.5(c) present the two $w$-directed graph of the TCP-net in Figure 2.4 with the dependency graph in Figure 2.5(a). In this example TCP-net $N$, $S(N) = \{A\}$, because there is only one undirected ci-arc $(B, C)$. For the assignment $w = a$ (resp. $w = \bar{a}$), in Figure 2.5(b) (in Figure 2.5(c)), there is directed edge $\langle C, \bar{B} \rangle$ (resp. $\langle B, \bar{C} \rangle$), because $C \triangleright_a B$ (resp. $B \triangleright_a C$).

Using Definitions 2.2 and 2.3, the class of conditionally acyclic TCP-nets is defined by Brafman et al. [30]:

**Definition 2.4** A TCP-net $N$ is conditionally acyclic if for every possible assignment
w to the set of variables in \( S(N) \), the induced \( w \)-directed graph of its dependency graph \( N^* \) is acyclic.

2.3.4 Individual dominance testing in CP-nets and TCP-nets

Given any preference representation formalism, one of the most fundamental queries is whether some outcome \( o \) dominates (i.e., is strictly preferred to) some other outcome \( o' \), called Dominance Testing. As discussed in [24, 130], such dominance testing in CP-nets is required whenever we wish to generate more than one non-dominated solutions to a set of hard constrains. Moreover, it is also a most important query in group decision-making, e.g., as to know whether an outcome \( o \) Pareto-dominates another outcome \( o' \), each agent need to conduct dominance testing according to its own preference network.

CP-nets and TCP-nets provide a compact representation of preference ordering in terms of natural preference statements under a ceteris paribus (all else being equal) interpretation. Given a CP-net or TCP-net, comparisons between two outcomes that differ in the value of a single variable are easy: we only need to check the CPT of that variable and determine which outcome assigns it to a more preferred value. The better (improved) outcome can be considered as a product of a single improving flip in the value of a variable \( X \) from the worse outcome. An improving flip changes (flips) the value of a single variable in an outcome to a better value and obtains a better (improved) outcome.

Ceteris paribus semantics of a CP-net or a TCP-net induces a graph, known as induced preference graph [24, 26]. For any pair of outcomes \( o \) and \( o' \), \( o \) is said to dominate another outcome \( o' \) with respect to a CP-net or TCP-net \( \mathcal{N} \) (written as \( \mathcal{N} \models o \succ o' \)) if there exists a directed path, also called a sequence of improving flips from \( o' \) to \( o \) in the induced preference graph of \( \mathcal{N} \) [24, 30]. It consists of successively improving outcomes in the graph from \( o' \) to \( o \). Otherwise, we say \( \mathcal{N} \) does not entail \( o \succ o' \) (\( \mathcal{N} \not\models o \succ o' \)). Improving flips and the sequence of improving flips are defined analogously. Obviously, any sequence of improving flips is the inverse of the sequence of worsening flips, and vice versa. The following definition of improving flipping sequence in CP-net and TCP-net are from the work of Boutilier et al. [24] and Brafman et al. [30], respectively.
Definition 2.5 (Improving flipping sequence in CP-nets) A sequence of outcomes \( o' = \sigma_1, \sigma_2, \ldots, \sigma_{\ell-1}, \sigma_{\ell} = o \) such that
\[
o' = \sigma_1 \succ \sigma_2 \succ \cdots \succ \sigma_{\ell-1} \succ \sigma_{\ell} = o
\]
is an improving flipping sequence with respect to a CP-net \( N \) if and only if, \( \forall 1 \leq i \leq \ell \), outcome \( \sigma_i \) is different from the outcome \( \sigma_{i+1} \) in the value of exactly one variable \( X \), and \( \sigma_{i+1}[X] \succ \sigma_i[X] \) given the parent context \( u \) of \( X \) assigned by \( \sigma_i \) and \( \sigma_{i+1} \).

By incorporating the relative importance information between variables, the improving flipping sequence in TCP-nets is defined as follows.

Definition 2.6 (Improving flipping sequence in TCP-nets) A sequence of outcomes \( o' = \sigma_1, \sigma_2, \ldots, \sigma_{\ell-1}, \sigma_{\ell} = o \) such that
\[
o' = \sigma_1 \succ \sigma_2 \succ \cdots \succ \sigma_{\ell-1} \succ \sigma_{\ell} = o
\]
is an improving flipping sequence with respect to a TCP-net \( N \) if and only if, \( \forall 1 \leq i \leq \ell \), \( \sigma_i \) to \( \sigma_{i+1} \) is either:

**a CP-flip:** outcome \( \sigma_i \) is different from the outcome \( \sigma_{i+1} \) in the value of exactly one variable \( X \), and \( \sigma_{i+1}[X] \succ \sigma_i[X] \) given the parent context \( u \) of \( X \) assigned by \( \sigma_i \) and \( \sigma_{i+1} \); or

**an I-flip:** outcome \( \sigma_i \) is different from the outcome \( \sigma_{i+1} \) in the value of exactly two variables \( X \) and \( Y \),
\[
\sigma_i[X] \succ \sigma_{i+1}[X] \text{ and } \sigma_i[Y] \prec \sigma_{i+1}[Y] \text{ given the (identical) values of } Pa(X) \text{ and } Pa(Y) \text{ in } \sigma_i \text{ and } \sigma_{i+1}, \text{ and } X \triangleright Y \text{ given } R \emptyset\emptyset(X, Y|Z) \text{ and the (identical) values of } Z \text{ in } \sigma_i \text{ and } \sigma_{i+1}.
\]

Clearly, each value flip in such a flipping sequence is sanctioned by the CP-net or TCP-net \( N \), and the CP-flips in TCP-net are exactly the flips allowed in CP-nets. The preference ordering over the alternative space induced by a CP-net or TCP-net is strict partial order, i.e., it is generally not complete. Two outcomes \( o \) and \( o' \) may also be incomparable with respect to \( N \), denoted by \( N \models o \not\succ o' \), i.e., there is no improving
flipping sequence from $o$ to $o'$ neither from $o'$ to $o$. When the structure of the CP-net (resp. TCP-net) is acyclic (resp. conditionally acyclic), i.e. does not contain any dependency cycles, the dominance relations induced by the CP-net (resp. TCP-net) are irreflexive and transitive. That is, two outcomes $o$ and $o'$ can stand in one of the following three possible relations with respect to $N$:

1. $N \models o \succ o'$, denote that $o$ is strictly preferred to $o'$ with respect to $N$;
2. $N \models o' \succ o$, denote that $o'$ is strictly preferred to $o$ with respect to $N$; or
3. $N \models o \nprecedes o'$, denote that $o$ and $o'$ are incomparable with respect to $N$.

The third case means that the given CP-net or TCP-net $N$ does not contain enough information on whether either outcome is preferred to the other, i.e., $N \not\models o \succ o'$ and $N \not\models o' \succ o$. With cyclic preference networks, it may happen that $N \models o' \succ o$ and $N \models o \succ o'$. In such a situation, we say that the preference relation induced by $N$ is not consistent.

Recent work in the literature have studied the computational complexity of testing dominance relations in CP-nets, e.g. [71]. Unfortunately, the results show that reasoning about the preference ordering (dominance relation) expressed by a CP-net or a TCP-net is far from easy [24, 30, 71]. With the exception of special cases such as CP-nets (or TCP-nets) with tree or polytree structured conditional dependencies, dominance testing has been shown to be NP-hard in general CP-nets [71].

## 2.4 Typical goals in group decision-making

### 2.4.1 Pareto-optimality

The most basic and common objective of a group decision-making mechanism is to lead rational agents to optimal (or efficient) outcomes [53, 77, 122, 131, 132]. By optimal (or efficient) solution, we refer to a solution that is Pareto-optimal (or Pareto-efficient). Pareto-optimality (or Pareto-efficiency) is a concept in economics with applications in social sciences. We will use Pareto-optimality and Pareto-efficiency alternatively throughout this thesis.
Give a pair of outcomes \( o \) and \( o' \), we say that \( o \) Pareto-dominates \( o' \) (resp. \( o \) weakly Pareto-dominates \( o' \)) if and only if \( o \) is at least as preferred as \( o' \) for all the agents and is strictly to \( o' \) for at least one agent (resp. \( o \) is strictly prefer to \( o' \) for all the agents).

Let \( o \succeq_j o' \) (resp. \( o \succ_j o' \)) denotes an agent \( j \)'s preference over \( o \) and \( o' \) that \( o \) is at least as preferred as \( o' \) (resp. \( o \) is strictly preferred to \( o' \)) for agent \( j \), we have the following formal definitions for strong and weak Pareto-dominance [14, 34].

**Definition 2.7 (Strong Pareto-dominance)** For any pair of outcomes \( o \) and \( o' \) (\( o, o' \in O \) and \( o \neq o' \)), \( o \) strongly Pareto-dominates \( o' \) if and only if all agents prefers \( o \) to \( o' \). Mathematically, \( \forall j \in \{1, \ldots, n\}, \ o \succ_j o' \).

**Definition 2.8 (Weak Pareto-dominance)** For any pair of outcomes \( o \) and \( o' \) (\( o, o' \in O \) and \( o \neq o' \)), \( o \) weakly Pareto-dominates \( o' \) if and only if \( o \) is at least as preferred as \( o' \) for all the agents, and strictly preferred to \( o' \) for at least one agent. Mathematically, \( \forall j \in \{1, \ldots, n\}, \ o \succeq_j o' \) and \( \exists k \in \{1, \ldots, n\}, \ o \succ_k o' \).

Moreover, an outcome is Pareto-optimal (resp. weakly Pareto-optimal), if there is no other outcomes weakly Pareto-dominates (resp. strongly Pareto-dominates) that outcome. We have the formal definition as follows.

**Definition 2.9 (Weak Pareto-optimality)** An outcome \( o \) is Weakly Pareto-optimal (WPO) if there exists no other outcome \( o' \in O \) and \( o \neq o' \), such that \( o' \) strongly Pareto-dominates \( o \).

**Definition 2.10 (Pareto-optimality (Pareto-efficient))** An outcome \( o \) is Pareto-optimal (PO) if there does not exist another outcome \( o' \in O \) and \( o \neq o' \), such that \( o' \) weakly Pareto-dominates \( o \).

When an outcome is weak Pareto-optimal (WPO), there are no possible alternative outcomes whose realization would cause every individual agent to gain. Weak Pareto-optimality is “weaker” than Pareto-optimality (PO) in the sense that the conditions for WPO status are “weaker” than those for PO status. Pareto-optimality implies weak Pareto-optimality: any outcome that can be considered as a PO outcome will be also qualified as a WPO outcome. However, the reverse does not hold: a WPO outcome
is not necessarily PO. Throughout this thesis, we use the terms Pareto-optimal and Pareto-efficient interchangeably.

A collective decision (an outcome) that is not Pareto-efficient implies that a certain change in the assignment of attribute values may result in some individual agents being made “better off” with no individual agent being made “worse off”. Hence, it is commonly accepted that outcomes that are not Pareto-efficient are to be avoided, and therefore Pareto-efficiency is an important criterion for evaluating a collective decision for a group of agents.

Given a set of alternative outcomes and a way of valuing them, the Pareto-frontier (also called Pareto-set or Pareto-front) is the set of alternatives that are Pareto-efficient. The Pareto-frontier is particularly useful in group decision-making: by restricting attention to the set of outcomes that are Pareto-efficient, the agents can make tradeoffs within this set, rather than considering the entire outcome space.

### Pareto-optimality in the context of CP-nets and TCP-nets

Rossi et. al [125] provide the definitions of Pareto-dominance and Pareto-optimality in the context of CP-nets. In this thesis, we make an assumption that indifference is not allowed in CP-nets and TCP-nets. Given a pair of outcomes \( o \) and \( o' \), \( o \) Pareto-dominates \( o' \) (written \( o \succ_p o' \)) iff all the agents’ preference networks entail that \( o \) is better than \( o' \). Accordingly, the formal definition of Pareto-optimality in the context of CP-nets and TCP-nets is as follows.

**Definition 2.11 (Pareto-optimality)** An outcome \( o \) is Pareto-optimal if and only if there exists no other outcome \( o' \) such that every agent’s CP-net or TCP-net entails \( o' \) is preferred to \( o \). Mathematically, \( o \) is Pareto-optimal if \( \forall o' \in O \text{ and } o' \neq o, \text{ s.t. } \forall j \in \{1, \ldots, n\}, N_j \models o' \succ o. \)

### 2.4.2 Fairness

Pareto-efficiency is the most fundamental criterion of efficiency: an outcome should be such that there is no other outcomes that would be better for some agents without being

---

\(^2\)Notice that the notion of indifference in the values of variables in CP-nets as well as the notion of indifference flip have been defined in Rossi et al. [125]
worse for any of the others. However, Pareto-efficiency does not necessarily result in a socially desirable distribution of welfares: it makes no statement about equality, or the overall well-being of a society [7, 14]. In a group decision-making system, fairness is another important goal in a group decision-making system. With the utility or other quantitative forms of preference representation of the agents’ payoffs, fairness is axiomatized in the literature (See, for example [34, 79, 108, 109, 152]). For instance, following are the most widely studied social welfare metrics for measuring the fairness of an outcome.

**Utilitarian**: The utilitarian social welfare of an outcome is the sum of all the individual agents’ payoffs. The Utilitarian solutions are the outcomes in which the sum of all the individual agents’ payoffs is maximized.

**Egalitarian** (aka. *Maximin*): The Egalitarian social welfare of an outcome is defined as the minimum payoff among the participating agents under that outcome. The Egalitarian solutions are the outcomes in which the minimum payoff among the participating agents is maximized.

**Nash**: The Nash social welfare is defined as the product of the agents’ payoffs. The Nash solutions are the outcomes in which the product of the agents’ payoffs is maximized.

Asking for maximal utilitarian social welfare is a very strong requirement. Utilitarian social welfare mainly considers average payoffs in the agent society, however, it does not reflect clearly about the individual agent’s well-being. Instead, egalitarian social welfare, analyse the well-being of the society from the individual point of view, considering the payoff of the worst-off agent in the whole society. Nash social welfare, on the other hand, can be considered as a compromise between utilitarianism and egalitarianism, taking into account the society as a whole, as well as the individual payoffs.

Another fairness criteria is *envy-freeness*, which has been well studied in the field of resource allocation and fair division (e.g., see [1, 34, 39, 108]). An allocation is envy-free if no agent would rather prefer to obtain the bundle held by one of the others.
2.5 Relevant Literature

Fairness in the context of CP-nets and TCP-nets

The preference induced from CP-nets and TCP-nets is purely qualitative, i.e., a CP-net or TCP-net induces a preference ranking (generally a partial order) over the possible alternative space. One possible way to evaluate fairness of a group decision-making mechanism with a collection of CP-nets or TCP-nets is to quantify the well-beings (also known as pay-offs) of the agents of the chosen outcome. As such, the aforementioned criterion of fairness, e.g., egalitarian social welfare, can be directly applied. Another alternative is to analyse the fairness properties of the aggregation mechanism itself rather than the outcome chosen by that mechanism. For instance, whether the mechanism satisfies Non-dictatorship, Independence of irrelevant alternatives (IIA) and Unanimity. Unfortunately, Arrow’s impossibility theorem [10] points out that, when there are three or more possible alternatives (options), no aggregation or voting system can convert the ranked preferences of individuals into a community-wide ranking while also meeting all the three criteria: Non-dictatorship, IIA and Unanimity.

2.5 Relevant Literature

This section serves as a broad overview of the existing approaches in relations to the research problems tackled in this thesis. The work related to the contributions of this thesis will be discussed in that chapters that followed.

2.5.1 Preference aggregation and voting in combinatorial domains

The research work in this field is mainly conducted in combinatorial domains. It focuses on the design and analysis of methods for collective decision-making not only from the prospective of social choice theory, e.g., the properties of collective decision-making mechanisms or voting rules and the existence (or otherwise) of procedures meeting certain requirements, etc., but also from the computational aspect, e.g., complexity analysis, the appropriate algorithmic techniques and the compact representation of preferences in combinatorial domains.

In combinatorial domains the set of possible alternatives is a Cartesian product of finite value domains for each one of a set of variables, and thus known methods for group
decision-making and classical results from social choice theory may not always be applicable [41].

Firstly, for such combinatorial problems, the mere representation of the preferences of individuals over different alternatives becomes a non-trivial problem. For instance, in classical voting theory, voters are supposed to submit their preferences as linear orders over the set of alternatives, and then a voting rule is applied to select one alternative as the winner. If the set of alternatives have a combinatorial structure, then the number of alternatives is exponentially large, so it is unrealistic to ask voters to specify their preferences as (explicit) linear orders. Just imagine that having 10 binary variables, each involved agent would need to present his preference over 1024 alternatives. The researchers in artificial intelligence have made an important contribution to the development for preference representation and elicitation tools.

Several representation languages aiming at enabling a succinct representation have been proposed, without having to enumerate a prohibitive number of possible outcomes. Many of these preference representation languages in combinatorial domains are graphical: preferences are expressed locally (on small subsets of variables). The common feature of these languages is that they allow for a concise representation of the preference structure, while preserving a good readability that similar to the way users express their preferences in natural languages e.g., [23, 24, 29]. For instance, CP-nets [24], which are tailored for representing preference relations on the domain of each variable conditioned by the values of the variables it depends on; TCP-nets [29, 31], which enrich CP-nets by allowing the expression of relative importance statements between single variables. Another example, conditional preference theories (CP theories) [142, 143], further extend CP-nets and TCP-nets. They are more general as they allow conditional preference statements on the values of a variable, together with a set of variables that are allowed to vary when interpreting the preference statement. The language considered in [145] is even more general: the preference statements do not compare single values of variables but tuples of values of different variables. Taking inspiration from previous work on graphical languages for preference representation, specifically CP-nets, Sylvain et al. [28] introduce conditional importance networks (CI-nets), to express monotonic preferences between sets of goods. In their model, the importance statements can bear on arbitrary sets of variables, and not only on singletons.

Some other preference representation languages are based on propositional logic (or possibly a fragment of it). For instance, prioritized goals, distance-based goals, weighted
goals, bidding languages for combinatorial auctions. Languages based on propositional logic have been proposed recently for some multi-agent problems, such as for combinatorial auctions [27, 111, 128] and automated negotiation [146]. Finally, some languages are domain-specific (for instance, they may be tailored for expressing bids in auctions), while others are not.

We do not give out an exhaustive list of languages for modelling preferences in combinatorial domains here. Nonetheless, the readers are referred to Lang [92] for a detailed survey of compact preference representations languages in combinatorial domains.

Secondly, as there are exponentially many alternatives, even if the agents’ compact preferences are given, direct election or search for a good collective decision from the entire set of possible outcomes becomes impractical. For instance, voting theory typically assumes that every voter would submit his preferences as linear order over the set of alternatives, and then a voting rule is applied to select one or some alternatives as the winner. However, as there are exponential number of alternatives, it is not practical to apply traditional voting rules in a straightforward way. Imagine that having 10 binary variables, for some pair wise decision rules, a group decision may involved \( \binom{2^{10}}{2} = 523776 \) pair wise comparisons. The easiest way to cope with this problem is to decompose an election into a set of independent elections, each of which bears on a single variable. This only works with the assumption that the preferences of the voters are separable. That means, the voters’ preferences over each variable are independent from the values of other variables. However, when the variables are not independent it soon becomes impractical for the following two reasons. Firstly, in this case, a voter cannot specify preferences over a single variable without knowing the values of the other variables. Secondly, as soon as voters have preferential dependencies between variables, it is generally a bad idea to decompose a vote problem. Decomposing a vote problem can give rise to “multiple election paradoxes”. Such paradoxes have been studied by a number of authors, see for example, [16, 33, 87].

The problem of preference aggregation or voting in combinatorial domains has been studied in the literature, e.g., [95, 125]. Several computational efficient approaches for preference aggregation and winner determination problem of voting in combinatorial domains have been developed in the field of computational social choice. Most of them directly operate on the compact preference representations, rather than reason about the complete individual or collective preference relations over the alternative space. For instance, Gonzales et al. [72] address the preference aggregation problem in combinatorial domain, where the agents’ preferences are represented by generalized
additive decomposable (GAI) utility functions. Gonzales et al. consider several criteria to define the notion of compromise solution (maxmin, minmaxregret, weighted Tchebycheff distance). For each of them, they further propose a fast procedure for the exact determination of the optimal compromise solution in the product set, based on a ranking algorithm enumerating solutions according to the sum of the agents’ individual utility values until a boundary condition is reached. Lafage and Lang [88] propose two logical approaches, namely weighted logics and distance-based logics, for representing preferences in a group decision-making context. After the individual agents’ preferences are computed independently, these logical preferences are aggregated into a normalized collective decision function that best conveys the preferences of the group of the agents.

Some existing work has considered the preference aggregation scenario where the agents’ preferences are represented by CP-nets. In particular, Rossi et al. [125] introduce mCP-nets as an extension of the CP-net (conditional preference network) formalism to model and handle the qualitative and conditional preferences of multiple agents. Rossi et al. give a number of different semantics for reasoning with CP-nets. The semantics are all based on the idea of agent voting. Purrington and Durfee [120, 121] apply the CP-net preference representation to the problem of negotiating optimal joint outcomes. They present a plausible strategy to assess outcomes based on their relative positions in the agents’ partial orders induced by the agents’ CP-nets.

Recent work has also started to investigate using CP-nets to represent preferences in voting contexts with multiple issues. For instance, instead of assuming the issues are separable, Lang and Xia [95] investigates sequential voting with CP-nets in combinatorial domains. In their work, it is assumed that there is an order over issues such that every voter’s preferences for “later” issues depends only on the decisions made on “earlier” issues, then the voters’ CP-nets are acyclic, and a natural approach is to apply issue-wise voting rules sequentially [95]. Such sequential voting process enjoys low communication and computational cost when each of the local voting rules is easy to compute. However, the assumption of a common acyclic structure among the voters is rather restrictive and demanding. Recent extensions of sequential voting rules with CP-nets include order-independent sequential voting rules [150], as well as a framework for voting when preferences are modelled by general (that is, not necessarily acyclic) CP-nets [148].

In this thesis, two computationally efficient approaches for preference aggregation are investigated in order to support agent-based group decision-making in combinatorial
domains. Both of them consider CP-nets (and its variants) as the representation models of the agents’ preferences in combinatorial domains, while do not require any common structure between different agents. The first approach is introduced in Chapter 4, which considers acyclic preference dependencies. It is based on an efficient numerical transformation of individual CP-nets (or TCP-nets). The second approach introduced in Chapter 5 then further considers general preference structures, where the agents’ CP-nets can be cyclic. In Chapter 5, we consider a majority decision rule. Extending the work of Xia et al. [148], a computationally efficient approach to compute the possible winning alternatives is proposed, by first eliminating many alternatives efficiently, and then determining the possible winners among the remaining alternatives.

2.5.2 Multi-issue negotiation

The research on multi-issue negotiation has been conducted in the fields of economics, in particular game theory, and artificial intelligence [91, 97].

Multi-issue negotiation in economics can be divided into two branches: non-cooperative and cooperative game theory. Non-cooperative negotiation considers negotiation as a fully specified game and focuses on analysing the strategy equilibrium of the agents. It refers to the negotiation protocol that the players follow during the negotiation process, for example, see [8, 37, 64, 126]. These protocols have been applied mainly to evaluate single-issue negotiations, for example, negotiate over the price of a product. In many real life situations, however, negotiations involve more than two players, and may involve more than one issue. The problem with multiple issues is so complex that rigorous modelling and analysis with Non-cooperative game theory turns out to be intractable. Thus, the non-cooperative research literature mostly addresses the challenge by decomposing the problem into issue-by-issue negotiations.

Instead of analysing the negotiation process, the research in cooperative game theory aims at finding an outcome satisfying a set of axioms or conditions when given some possible outcomes. For instance, the Nash bargaining solution and the Egalitarian solution. Researchers in this field propose methods on how to find out efficient and fair solutions by assuming the agents are cooperate and can solve multi-criteria-decision-making (MCDM) problems [19, 69, 79, 109, 110]. However, the research in this field is mainly conducted under complete information setting, i.e., given that the agent’s utility functions or preferences are known, which is too strong an assumption that is
not able to be maintained in the real world applications.

The research work in AI field on the other hand, emphasizes on designing appropriate models with automated and tractable negotiation mechanisms, such as negotiation framework, trading-off mechanism and heuristic methods. Most of the existing work has been dealing with the utility-based negotiation problems, where the agents’ preferences are mathematically represented by utility functions. For instance, Fatima et al. [61, 62, 65] propose an agenda-based framework for multi-issue negotiation under time constraints in an incomplete information setting. In this framework, agents can propose either combined or single offer on the remaining issues and make decisions on issues independently, faced with a combined offer. Nonetheless, this framework is based on an assumption that the utility functions of the agents are linear additive.

There are different ways that the agents can make tradeoffs to reach a “Win-Win” solution in a negotiation. Faratin [60] propose the use of similarity criteria based on fuzzy rules to make agents trade off. The heuristic employed in this approach is to select an offer that is most similar to the opponent’s last offer by a function based on fuzzy similarity. By the experiment analysis, Faratin et al. argue that the method can help agents to squeeze out more favourable agreements and reach Pareto-optimal solution or very close. Sycara [135, 136, 137] uses a case-based reasoning approach where the agents make offers based on similarity of the multi-issue negotiation context (including issues, opponents, and environment) to previous negotiations. Sycara provides a case study of labour management negotiations, where this approach is validated as realistic by domain experts.

Some other works about multi-issue negotiation are mediation-based (e.g., [56, 58, 74, 78, 89]). For instance, Ehtamo et al. [58] present a gradient search method for making trade-offs, while also creating joint utility gains for the negotiating agents until they finally reach the Pareto-frontier. Ehtamo et al. examine how the compromise directions should be chosen and conclude that the simple 50-50 split between the two negotiators’ gradient directions turns out to be a suitable and fair compromise direction. Heiskanen et al. [74] proposes a constraint proposal method for computing Pareto-optimal solutions for multi-party negotiations. They employ a neutral coordinator to assist the agents in searching for Pareto-optimal solutions. The agents are not required to reveal their utility functions yet have to indicate their most preferred points on different sets of linear constraints. Their method can be used to generate either one Pareto-optimal solution dominating the status quo solution of the negotiation or an approximation to the Pareto-frontier.
Another mediation-based negotiation model is given by Lai et al. [89]. In their approach, the mediator conducts a Pareto-efficient enhancement for a proposal in each negotiation period. In each period, agents are required to propose a preferred solution on a base line. Then based on this proposal, the mediator applies a computationally efficient query process to find a point that approaches Pareto-efficiency and mutually better for both agents. However, their approach does not address the fairness issue between the utility gains of two negotiating agents. The running time of their algorithm is reasonable in two-attribute case; however, it grows rapidly as the number of attributes increases.

Concerning with the disadvantage of disclosing preference information, this thesis further considers using negotiation approaches to support group decision-making without requiring the agents to give out the complete preference information. Two negotiation approaches are proposed, addressing the negotiation problem in combinatorial domains and in the domains with multiple continuous issues, respectively. We first continue with the group decision-making problem in combinatorial domains in Chapter 6, where an efficient distributive protocol for negotiation with general preference is introduced. Then in Chapter 7, the problem of group decision-making over multiple continuous issues is further explored. In Chapter 7, we improve the work by Ehtamo et al. [58], and propose a mediation-based negotiation approach to support multiple agents reaching efficient and fair agreements.
Part I

Preference aggregation in combinatorial domains
Chapter 3

Penalty score-based heuristic for dominance testing in CP-nets

Chapter 3 revisits an important problem of preference reasoning with CP-nets, namely dominance testing. Dominance testing is generally a difficult problem even for binary-valued, acyclic CP-nets, and tractable search algorithms exist only for specific problem classes. Hence, there is a need for efficient algorithms and techniques for dominance testing in more general problem settings. In this chapter, an heuristic algorithm, called DT*, for dominance testing in arbitrary acyclic multi-valued CP-nets is introduced. The proposed approach guides the search process efficiently and allows significant reduction of search efforts without impacting the completeness of the search process. Experimental results that demonstrate the computational efficiency and feasibility of the proposed approach are also presented in this chapter.

3.1 Introduction

The problems of eliciting, representing and reasoning with qualitative preferences over multi-attribute domains arise in many fields such as planning, design, and collective decision making [71, 88, 125]. As the number of alternative outcomes of such domains is exponentially large in the number of attributes, it is impractical to express preferences explicitly by giving out the ordering over the entire alternative space. CP-nets
provide a compact representation of preference ordering in terms of natural preference statements under a ceteris paribus (all else being equal) interpretation. Ceteris paribus semantics induce a graph, known as an induced preference graph [24, 26]. As such, an outcome $o$ is said to dominate another outcome $o'$ if there exists a directed path, called a sequence of improving flips, consisting of successively improving outcomes in the graph from $o'$ to $o$ [24, 25]. Unfortunately, as discussed in Chapter 2, reasoning about the preference ordering (dominance relation) expressed by a CP-net is far from easy [24, 71]. With the exception of special cases, such as CP-nets with tree or poly-tree structured conditional dependencies between variables, dominance testing has been shown to be PSPACE-complete in general CP-nets [24, 71]. The complexity of dominance testing for CP-nets whose dependency graph is acyclic (but not necessarily singly-connected) is still an open problem. Existing results can only show that dominance testing in acyclic CP-nets is NP-hard and belongs to PSPACE [50, 71].

Some general pruning rules have been studied in [24] to reduce the search effort, but they may not be able to guide the search efficiently when the structure of the CP-net is complex or the number of variables is large. Some other work tries to approximate the Conditional Preference with lower computational costs (see e.g., [51, 144]). Another work published by Santhanam et al. [130] explores an approach to dominance testing with acyclic CP-nets via Model Checking. However, their approach applies mainly to binary-valued preference networks. The complexity and feasibility of their approach to dominance testing in multi-valued preference networks is still an open question. Hence, there is a need for efficient algorithms and techniques to deal with dominance testing in more general problem settings.

To this end, this chapter revisits the problem of dominance testing. It focuses on a wide class of CP-nets, i.e., arbitrary acyclic multi-valued CP-nets. An efficient heuristic algorithm, called $\text{DT}^*$, is introduced in this chapter to search for an improving flipping sequence from the worse outcome to the better outcome of the given dominance query.\footnote{The proposed heuristic will be described in the context of improving flipping sequences, but it can be applied to worsening searches according to the same principle.}

We first present an efficient approach that compiles an individual CP-net into a numerical penalty scoring function. As such, the penalty scoring function preserves the strict preference ordering induced by the given CP-nets. Then, this penalty scoring function is further used as a heuristic of an algorithm, called $\text{DT}^*$, to search for improving flipping sequence in dominance testing in arbitrary acyclic multi-valued CP-nets. We theoretically prove that our proposed $\text{DT}^*$ algorithm guarantees completeness of the
search. DT* algorithm does not reduce the worst-cases complexity; however, it efficiently guides the search process and significantly reduces the number of nodes visited during the search. Thus, the running time of dominance testing using the proposed approach, on average, is reduced. Furthermore, when no flipping sequences are possible, it returns the quick failure to the dominance query without having to exhaustively search through all possible branches. In order to test the running time and feasibility of the proposed approach, extensive experiments with different problem settings (i.e., different CP-net structures and different domain sizes) have been carried out.

The remainder of this chapter is organised as follows. We review some existing approximations to CP-nets and introduce a penalty scoring function for approximating the preference induced by CP-nets in Section 3.2. Then in Section 3.3, we discuss some existing pruning techniques in dominance testing and present the technical details of the proposed heuristic approach. We subsequently present the experimental results in Section 3.4, and finally summarises this chapter in Section 3.5.

### 3.2 Penalty scoring functions for CP-nets

#### 3.2.1 Existing approximations of CP-nets

As dominance testing is generally a difficult problem for CP-nets, several approximations of CP-nets have been introduced in the literature, i.e., [23, 51, 52, 107, 142, 143]. In particular, Wilson develops a formalism that maintains important properties of CP-nets in [143] and further extends this formalism to express TCP-nets in [142]. There are simple sufficient conditions for consistency in a given CP-net or TCP-net, and under these conditions, it is straightforward to find a total order on outcomes which extends the conditional preference order. This approximation is purely qualitative, but many decision-making scenarios require quantitative measurements on the degree of desirability of alternative outcomes. Dubois et al. [52] propose a numerical approximation for acyclic CP-nets based on possibilistic logic, which, however, requires complex possibilistic bases merging. The work Boutilier et al. [23] based on UCP-net can be used as a quantitative approximation of acyclic CP-nets. However, generating UCP-nets is exponential in the size of CP-net node’s Markov family \(^2\), and thus in the CP-net node out-degree [51]. Similarly, McGeachie and Doyle [107] use direct graph theor-

\(^2\)Markov family of a node X contains X, its parents and children, and the parents of its children.
etic methods to construct numeric utility functions for ceteris paribus preferences. But such direct constructions can also prove costly because small sets of ceteris paribus preference rules can specify very large graphs. Another work by Domshlak et al. [51] provides a numerical approximation for acyclic CP-nets using weighted soft constraints. Their approach is computationally efficient, i.e., both compiling an acyclic CP-nets into a corresponding weighted SCSP (soft constraint satisfaction problem) and comparing between a pair of outcomes are polynomial in the size of the network.

3.2.2 The proposed penalty scoring function

The penalty scoring function introduced here, which would be used as an important component of the proposed heuristic for dominance testing in CP-nets introduced later in the chapter, is based on the work of Domshlak et al. [51].

For a variable $X$, let $|D(X)|$ be the domain size of $X$ and thus there are $|D(X)|$ degrees of penalties of $X$, denoted by $d_1, \ldots, d_{|D(X)|}$. Without loss of generality, we assume the degree of penalties of a variable $X$ range between 0 and $|D(X)| - 1$; that is, $d_1 = 0, \ldots, d_{|D(X)|} = |D(X)| - 1$. For instance, in a binary-valued CP-net each variable will have only two degrees of penalties, i.e., $d_1 = 0$ and $d_2 = 1$. For a variable $X$, consider a preference ordering over the value of $X$ given an instantiation of $X$’s parents, let the rank of the most preferred value of $X$ be 0 and the rank of the least preferred valued of $X$ be $|D(X)| - 1$, given an outcome $o$, the degree of penalty of a variable $X$ in $o$ is then the rank of the value $o[X]$ in the preference ordering over $X$ given the parent context $u = o[Pa(X)]$. We denote by $d^o_X$ ($d^o_X \in \{d_1, \ldots, d_{|D(X)|}\}$) the degree of penalty of $X$ with respect to $o$. For instance, consider a variable $X$ such that $D(X) = \{x, x', x''\}$. Assume that, under a parent context $u = o[Pa(X)]$ assigned by an outcome $o$, $x \succ x' \succ x''$. If $o[X] = x$, then $d^o_X = d_1 = 0$; if $o[X] = x'$, then $d^o_X = d_2 = 1$; if $o[X] = x''$, then $d^o_X = d_3 = 2$.

CP-net imposes a rich structure to allow variables to have different degrees of importance: variables “higher-up” in the structure of the network are considered to be more important than the lower level variables [24–26]. Thus, it is more important to obtain a preferred value for a variable than any of its descendents. We now analyse the importance weight of a variable in a CP-net. Given an acyclic CP-net $N$ and consider an improving flip from an outcome $o$ to another outcome $o'$ that flips the value of a single variable $X$, changing the value of $O$ may also affect the preference status of $X$’s chil-
3.2. Penalty scoring functions for CP-nets

Algorithm 3.1: \texttt{assgWeightCP} ($\mathcal{N}$)

Thus, the resulting changes from $o$ to $o'$ includes: (i) the degree of penalty of $X$ decreases from $d^o_X$ to $d^o_0$ ($d^o_X > d^o_0$); and (ii) the degrees of penalty of $X$’s children may change, which in the worst case, may result in the degree of penalty of each child $Y$ increasing from $d^o_Y = 0$ to $d^o_0 = |D(Y)| - 1$. Consequently, in order to preserve the preference ordering induced by the given CP-net, the importance weight of a variable in that CP-net must be larger than the sum of the maximum penalties of its children \footnote{This does not necessarily mean that the importance of a parent is greater than the sum of all of its descendants induced by the original CP-nets. The importance weight assignment introduced here is just an approximation that preserves the strict preference ordering induced by the original CP-nets.}. We now provide the formal definition of the variable importance weight in an acyclic CP-net.

**Definition 3.1 (Importance Weight)** Given an acyclic CP-net $\mathcal{N}$ over a set of variables $V$. For each variable $X \in V$, let $\text{Ch}(X)$ denote the set of children of $X$ in $\mathcal{N}$, the importance weight of variable $X$, denoted by $w(X)$, is recursively defined by:

$$w(X) = q_X + \sum_{Y \in \text{Ch}(X)} w(Y) \cdot (|D(Y)| - 1)$$

(3.1)

where $q_X$ can be any positive real numbers ($q_X > 0$).

Algorithm 3.1 provides a simple implementation to compute importance weights. It takes linear time in the size of the network. Following a reverse topological ordering, it first assigns the importance weights to the variables that have no descendents (line 3–4) and then iteratively assigns the importance weights to the upper level variables according to Equation (3.1). Note that there are several ways to assign importance weights to the variables and the way we use here is different from [51]. This chapter...
consider the tight lower bound of the importance weight assignment, i.e. the sum of maximum penalties of the variable children \( \sum_{Y \in \text{Ch}(X)} w(Y) \cdot (|D(Y)| - 1) \). We would show later in Theorem 3.1 that \( q_X > 0 \) is sufficient to guarantee that the weight assignment is order-preserving.

**Example.** Consider an agent’s CP-net over a set of 5 variables \( V = \{ A, B, C, D, E \} \) in Figure 3.1(a). In this example, since all variables are binary, i.e. \( \forall X \in V, |D(X)| = 2 \) and we assume that \( \forall X \in V, q_X = 1 \). We can assign the importance weight to each variable in a reverse topological ordering of variables: \( w_D = q_D = 1 \); \( w_B = q_B + w_D \cdot (2 - 1) = 2 \); \( w_C = q_C + w_D \cdot (2 - 1) = 2 \); \( w_A = q_A + (w_B \cdot (2 - 1) + w_C \cdot (2 - 1)) = 5 \); \( w_E = q_E + w_B \cdot (2 - 1) = 3 \). The importance weight of each variable in this CP-net is attached on top of the variables respectively in Figure 3.1(b).

Given an acyclic CP-net \( N \) and an outcome \( o \), the penalty score of a variable \( X \) in \( o \) is the degree of penalty of \( X \) in \( o \) multiplied by the importance weight of \( X \). The penalty score of \( o \) is then defined by the sum of penalties of the domain variables. We define the following penalty scoring function for an acyclic CP-net based on the work by Domshlak et al. [51].

**Definition 3.2 (Penalty scoring function)** Given an acyclic CP-net \( N \) over a set of variables \( V \) and an outcome \( o \). The penalty scoring function \( \text{pen} \), mapping from an outcome \( o \in O \) to \([0, +\infty] \), is defined as follows:

\[
\forall o \in O, \text{pen} (o) = \sum_{X \in V} w(X) \cdot d_X^o
\]  (3.2)
EXAMPLE. Consider our running example in Figure 3.1(a) with the variable importance weight assignment in Figure 3.1(b). Consider the outcome \( o = \tilde{a} \tilde{b} \tilde{c} \tilde{d} \tilde{e} \). As the agent unconditionally prefers \( A = a \) to \( A = \tilde{a} \) (resp. \( E = e \) to \( E = \tilde{e} \)), \( d_A^o = 1 \) (resp. \( d_E^o = 1 \)). On the other hand, \( b \succ \tilde{b} \) (resp. \( \tilde{c} \succ c \), \( \tilde{d} \succ d \)) given the parent context \( A = \tilde{a} \) and \( E = \tilde{e} \) (resp. \( A = \tilde{a} \), and \( B = \tilde{b} \) and \( C = \tilde{c} \)) and thus \( d_B^o = 1 \) (resp. \( d_C^o = 0 \), \( d_D^o = 0 \)). Consequently, the penalty score of outcome \( \tilde{a} \tilde{b} \tilde{c} \tilde{d} \tilde{e} \) is:
\[
pen(o) = w_A \cdot 1 + w_B \cdot 1 + w_C \cdot 0 + w_D \cdot 0 + w_E \cdot 1 = 5 \cdot 1 + 2 \cdot 1 + 2 \cdot 0 + 1 \cdot 0 + 3 \cdot 1 = 10.
\]
In order to compute the penalty score of an outcome, we simply need to sweep through the network from top to bottom (i.e., from ancestors to descendants), and to check the degree of penalty of the currently considered variable given its parent context. And finally we compute the penalty score of the outcome based on Equation (3.2). Consequently, the penalty score computation for a particular outcome takes \textit{polynomial time} in the size of the network. We now prove that the way we assign penalties over alternative outcomes preserves the strict preference ordering induced by the original CP-net.

\textbf{Theorem 3.1} Given an acyclic CP-net \( \mathcal{N} \), we have:
\[
\forall o, o' \in O, \text{ if } \mathcal{N} \models o \succ o' \text{ then } pen(o') > pen(o)
\]

\textbf{Proof.} \( \mathcal{N} \models o \succ o' \) if and only if there exists a sequence of improving flips from \( o' \) to \( o \), denoted by \( Seq(o', o) = \sigma_1 (= o'), \sigma_2, \ldots, \sigma_{t-1}, \sigma_t, (= o) \), with respect to the conditional preference tables in \( \mathcal{N} \). Each improving flip from \( \sigma_p \) to \( \sigma_{p+1} \) in \( Seq(o', o) \) that improves the value of a single variable \( X \), \( pen(\sigma_p) - pen(\sigma_{p+1}) = w(X) \cdot (d_X^{\sigma_p} - d_X^{\sigma_{p+1}}) + \delta \), where \( \delta \geq - \sum_{Y \in CH(X)} w(Y) \cdot (|D(Y)| - 1) \) and \( (d_X^{\sigma_p} - d_X^{\sigma_{p+1}}) \geq 1 \). Thus, \( pen(\sigma_p) - pen(\sigma_{p+1}) \geq w(X) - \sum_{Y \in CH(X)} w(Y) \cdot (|D(Y)| - 1) = w(X) - (w(X) - q_X) = q_X > 0 \). Consequently, with each improving flip from \( \sigma_p \) to \( \sigma_{p+1} \), \( pen(\sigma_p) > pen(\sigma_{p+1}) \). Following from the transitivity: \( pen(\sigma_1 (= o')) > pen(\sigma_2) > \cdots > pen(\sigma_{t-1}) > pen(\sigma_t) (= o) \) and thus \( pen(o') > pen(o) \). \( \Box \)
Corollary 3.1 Given an acyclic CP-net $N$, $\forall o, o' \in O$,

- if $\text{pen}(o') > \text{pen}(o)$ then $N \models o \succ o'$ or $N \models o \succsim o'$
- if $\text{pen}(o') = \text{pen}(o)$ then $N \models o \succsim o'$

3.3 Penalty score-based heuristic in Dominance Testing

In this section, we address a fundamental query Dominance Testing in CP-nets. Given a CP-net $N$, dominance testing is to test whether some outcome $o$ dominates (i.e., is preferred to) some other outcome $o'$ according to the preference induced by $N$. Particularly, we focuses on addressing the dominance testing problem in acyclic CP-nets, i.e. does not contain any dependency cycles among the variables. In such case the dominance relation is irreflexive and transitive. Two outcomes $o$ and $o'$ can stand in one of three possible relations with respect to $N$: either $N \models o \succ o'$ ($o$ is strictly preferred to $o'$); or $N \models o' \succ o$ ($o'$ is strictly preferred to $o$); or $N \models o \succsim o'$ ($o$ and $o'$ are incomparable: $N \not\models o \succ o'$ and $N \not\models o' \succ o$). The third case means that the given CP-net $N$ does not contain enough information to prove that either outcome is preferred to the other.

Given an acyclic CP-net, comparisons between two outcomes that differ in the value of a single variable are easy: we only need to check the CPT of that variable and determine which outcome assigns it to a more preferred value. The better (improved) outcome can be considered as a product of a single improving flip in the value of a variable $X$ from the worse outcome. Recall that for any pair of outcomes that differ on more than one variables, an outcome $o$ is said to dominate another outcome $o'$ with respect to an acyclic CP-net $N$ ($N \models o \succ o'$) if there exists a sequence of improving flips from $o'$ to $o$, denoted by $o' = \sigma_1, \sigma_2, \ldots, \sigma_{\ell-1}, \sigma_{\ell} = o$, where $\forall 1 \leq i \leq \ell$, outcome $\sigma_i$ is different from the outcome $\sigma_{i+1}$ in the value of exactly one variable $X$, and $\sigma_{i+1}[X] \succ \sigma_i[X]$ given the parent context $u$ of $X$ assigned by $\sigma_i$ and $\sigma_{i+1}$. Otherwise, $N \not\models o \succ o'$.

Consequently, given an acyclic CP-net $N$, a query $N \models o \succ o'$ can be treated as a search for an improving flipping sequence from the less preferred outcome $o'$ to the more preferred outcome $o$. The search process can be implemented as an improving search tree rooted at $o'$, $T(o')$, which has been introduced by Boutilier et al. [24].
3.3. Penalty score-based heuristic in Dominance Testing

The children of every node\(^4\) \(\sigma\) in \(T(\sigma')\) are those outcomes that can be reached by a single improving flip from \(\sigma\). Consequently, every rooted path in \(T(\sigma')\) corresponds to some improving flipping sequence from the outcome \(\sigma'\) with respect to \(\mathcal{N}\). Taking different directions in \(T(\sigma')\) leads to different improving sequences; however, taking a different direction during the tree traversal may also lead to a dead end, i.e., reach the optimal outcome of \(\mathcal{N}\) without visiting the target outcome \(o\) of the query. Notice that the improving search tree [24] is different from the so-called “preordered search tree” introduced by Wilson [144] for an upper approximation of conditional preferences.

Recent works have studied the computational complexity of testing dominance relations in CP-nets, e.g. [24, 50, 71]. Though some special tractable cases of answering dominance queries do exist [24], the general problem is known to be NP-hard even for acyclic CP-nets [50], and PSPACE-complete for cyclic CP-nets [71].

3.3.1 Some General Search Techniques

Since the hardness of dominance testing in CP-nets, several search techniques for dominance queries have been studied by Boutilier et al. [24] in order to reduce the search effort. In dominance testing, it is important that an algorithm for finding possible improving flipping sequence guarantees completeness. Given a dominance query of deciding whether \(\mathcal{N} \models o \succ \sigma'\), we say an algorithm is \emph{complete} if and only if it guarantees to find out at least one improving flipping sequence if one exists.

**Suffix Fixing.** Let \(X_1 > \cdots > X_m\) be an topological ordering consistent with the CP-net \(\mathcal{N}\), then \(\forall X_p \ (p \leq m)\) and \(\forall X_q \in Pa(X_p)\), we have \(q < p\). An \(r\)th \((r \geq 1)\) suffix of an outcome \(o\) is the subset of the outcome values \(o[X_1] o[X_{r+1}] \cdots o[X_m]\). The \(r\)th suffix of outcomes \(o\) and \(o'\) match iff \(\forall r \leq j \leq n, \ o[X_j] = o'[X_j]\). For a query \(\mathcal{N} \models o \succ \sigma'\), suffix fixing rules out the exploration of any possible flipping sequences that destroy the suffix of the currently considered outcome that matches the target outcome \(o\). It prunes the subtree that improves the value of a variable within the matching suffix. For instance, consider the CP-net \(\mathcal{N}\) in Figure 3.1(a) and the query \(\mathcal{N} \models abcd \succ \tilde{a}bcd\tilde{e}\). Let \(o = abcd\) and \(o' = \tilde{a}bcd\tilde{e}\), if pruned using suffix fixing and consider the variable ordering \(A > E > B > C > D\), the 2\(^{nd}\) suffix \(cd\) of \(o\) and \(o'\) matches. Thus, the values of \(C\) and \(D\) will never be improved in the improving search tree \(T(o')\), although

\(^4\)A node in the improving search tree is also an outcome.
given the assignment \( o'[A] = \bar{a} \) (resp. \( o'[B] = b \) and \( o'[C] = c \), \( \bar{d} \succ c \) (resp. \( d \succ d \)). As shown in [24], any complete search algorithm for the improving search tree remains complete if pruning using suffix fixing is used.

**Least-variable flipping.** For every node \( \sigma \) in the improving search tree, least-variable flipping rule restricts flips to the variables that are *least-improvable*. Formally, a variable \( X \) is least-improvable in an outcome \( \sigma \) with respect to \( N \) if there is some value \( x \in D(X) \) such that \( x \succ_{\sigma} \) \( \sigma[X] \) (where \( \sigma = \sigma[Pa(X)] \) is the parent context assigned by \( \sigma \)), and no descendent of \( X \) in \( \sigma \) has this property. For a query \( N \models o \succ o' \), least-variable flipping rule restricts attention to those variables that are not part of any matching suffix with the target outcome \( o \) and requires that the only neighbours of a node \( \sigma \) can be expanded in the improving search tree \( T'(o') \) are those in which some least improvable variable with respect to \( \sigma \) is improved.

However, least-variable flipping rule is only complete for a restricted class of CP-nets [24], i.e. tree-structured CP-nets and binary-valued, directed-path singly connected CP-nets. For multiply-connected networks, and networks with multi-valued variables, it does not guarantee completeness. That means, least-variable flipping may fail to find any improving sequence from \( o' \) to \( o \) although there does exist at least one. In such case, it does not provide a correct answer to the given query. For instance, consider the CP-net \( N \) in Figure 3.1(a) and the query \( N \models abcde \succ \bar{a}b\bar{c}\bar{d}e \). Starting with the root node \( \bar{a}b\bar{c}d\bar{e} \), the only least improvable variable that can be flipped is \( B \). Unfortunately, flipping \( B \) to value \( b \) leads to outcome \( \bar{a}b\bar{c}d\bar{e} \), from which the target outcome \( abcde \) is unreachable. All branches in the improving search tree grow towards the optimal outcome \( abcde \) without going through the target outcome \( abcde \) of the query. Figure 3.2 shows the complete improving search tree \( T(abcde) \) using least-variable flipping. However, there in fact exists a sequence of improving flips from \( \bar{a}b\bar{c}d\bar{e} \) to \( abcde \): \( \bar{a}b\bar{c}d\bar{e}, \bar{a}b\bar{c}\bar{d}e, \bar{a}b\bar{c}d\bar{e}, abcde, \bar{a}b\bar{c}d\bar{e}, abcde, abcd\bar{e}, abcde \).

When the number of variables is large or the structure of the CP-net is complex, suffix fixing may not be able to guide the search efficiently while least-variable flipping rule does not guarantee completeness for general acyclic CP-nets. To this end, in this chapter we present another efficient heuristic approach to dominance testing. The proposed approach efficiently guides the search and significantly prunes the improving search tree without impacting the completeness of the search process.

---

5This example has also been discussed in Example 7 in [24].
3.3. Penalty score-based heuristic in Dominance Testing

3.3.2 The DT* heuristic algorithm

This section presents the proposed heuristic approach, called DT*, to dominance testing in arbitrary acyclic multi-valued CP-nets. The penalty scoring function mentioned in Section 3.2 provides an order-preserving numerical approximation for a given CP-net. In this section, the penalty scoring function is used as an heuristic in the search process for improving flipping sequence in dominance testing. The proposed heuristic algorithm for dominance testing has a number of desirable properties:

- it often returns the quick failure for the dominance query if no flipping sequence is possible;

- it often quickly shows that back-tracking is needed when there is no possible flipping sequence to the target outcome following the currently considered path; and,

- it efficiently guides the search direction without compromising completeness of the search process.

Figure 3.2: Improving search tree for query \( N = \{abcde \succ \overline{abcde}\} \) using Least-variable flipping rule
Chapter 3. Penalty score-based heuristic for dominance testing in CP-nets

Given an acyclic CP-net $N$ and a pair of outcomes $o$ and $o'$, for the query $N \models o \succ o'$, we build the improving search tree $T(o')$ and search for an improving flipping sequence to the target outcome $o$ as discussed in [24]. However, instead of blind or exhaustive searching, we here develop an efficient heuristic to guide the search process and investigate a pruning technique to reduce the search effort. The evaluation function $f$ for the proposed heuristic search strategy as follows:

**Definition 3.3 (Evaluation function)** Given an acyclic CP-net $N$ and the query $N \models o \succ o'$ ($o, o' \in O$). Using the variable importance weight assignment in Definition 3.1 and set $\forall X \in V, q_X = 1$. The evaluation function $f$, mapping from a node (i.e., an outcome) $\sigma$ in the improving search tree $T(o')$ to $[0, +\infty]$, is defined by:

$$f(\sigma) = \text{pen} (\sigma) - HD(\sigma, o) - \text{pen} (o)$$  \hspace{1cm} (3.3)\]

where $HD(\sigma, o)$ the hamming distance between the current considered outcome $\sigma$ and the target outcome $o$. Both in binary-valued and multi-valued CP-nets, the hamming distance is defined by the number of variables that the two outcomes differ from each other.

The proposed heuristic algorithm $\mathcal{D}T^*$ (see Algorithm 3.2) is adapted from the $A^*$ heuristic search algorithm with $f(\sigma)$ being the evaluation function. It maintains a priority queue of nodes to be expanded, known as the fringe. On the one hand, the lower $f$ value for a node $\sigma$, the higher its priority is. On the other hand, we only consider the outcomes that the $f$ value is non-negative. That means, an outcome $\sigma$ will be added into the fringe only if $f(\sigma) > 0$. In essence, an outcome with a negative $f$ value means that there is no possible improving flipping sequence from that outcome to the target outcome $o$ (see Lemma 3.2). Before adding the original node $o'$ into the fringe, the $f$ value of $o'$ will be computed and the algorithm will return $\text{False}$ if $f(o') < 0$ (line 1–2). In this case, the query fails (i.e., return $\text{False}$, $N \not\models o \succ o'$) even before building the root node of the improving search tree. Otherwise, $o'$ will be added into the fringe as the root node of the improving search tree $T(o')$ (line 4). At each iteration of $\mathcal{D}T^*$, the first node $\sigma$, i.e. the node with the lowest $f$ value, is removed from the fringe and being expanded (line 4). The children of a node in $T(o')$ are those outcomes that can be reached by a single improving flip from that node.

Our proposed algorithm applies suffix fixing rules, restricting attention to those variables in $\sigma$ that are not part of any matching suffix with the target outcome $o$ (line 11).
3.3. Penalty score-based heuristic in Dominance Testing

**Input:** a dominance query (an acyclic CP-net \( \mathcal{N} \); a pair of outcomes \( o \) and \( o' \); and determining whether \( \mathcal{N} \models o > o' \))

**Output:** True: \( \mathcal{N} \models o > o' \); False: \( \mathcal{N} \not\models o > o' \)

```plaintext
if \( f(o') < 0 \) then
    return False;
else
    fringe \( \leftarrow \) INSERT(MAKE-NODE(o'), fringe);
    while fringe \( \neq \emptyset \) do
        \( \sigma \leftarrow \) REMOVE-FIRST(fringe);
        if GOAL-TEST(\( \sigma = o \)) then
            return True;
        else
            For each \( X \in \mathcal{N} \) do
                if IMPROVABLE(\( \sigma, X \)) \&\& \( X \notin \) ANY-MATCHING-SUFFIX(\( \sigma, o \)) then
                    \( \sigma' \leftarrow \) SINGLE-FLIP(\( \sigma, X \));
                    if NOT-REPEATED(\( \sigma' \)) \&\& \( f(\sigma') \geq 0 \) then
                        INSERT-ASC(MAKE-NODE(\( \sigma' \), fringe))
                end
            end
        end
    end
else
    return False
end
```

**Algorithm 3.2:** \( DT^*(\mathcal{N} \models o > o') \)

Moreover, it requires that a child \( \sigma' \) of a node be added into the fringe if and only if: (i) \( \sigma' \) has not been traversed before; and (ii) \( f(\sigma') \geq 0 \) (line 13). For the current node \( \sigma \) under consideration, we add each child \( \sigma' \) of \( \sigma \) that meets the above requirements into the fringe in ascendant order of the \( f \) values of the nodes in the fringe (line 14). \( DT^* \) continues until: the currently considered node for expansion equals to the target outcome \( o \), then it ends and returns True (\( \mathcal{N} \models o > o' \)) (line 7–8); or the fringe is empty, it returns False (\( \mathcal{N} \not\models o > o' \)) (line 20). Such a improving search tree is shown in Figure 3.3 for the dominance queries whether \( \mathcal{N} \models abcd \succ \overline{a}b\overline{c}d\overline{e} \) with the CP-net \( \mathcal{N} \) in Figure 3.1(a).
Chapter 3. Penalty score-based heuristic for dominance testing in CP-nets

Figure 3.3: Improving search tree

Completeness of DT*

In order to prove the completeness of our proposed heuristic algorithm, we first proof the follow lemmas.

**Lemma 3.1** Given an acyclic CP-net $\mathcal{N}$ over a set of variables $V$, assume that $\forall X \in V; q_X = 1$. Let $o, o'$ be any pair of outcomes that $\mathcal{N} \models o \succ o'$; $\mathbf{IS}$ the set of all possible improving flipping sequence from $o'$ to $o$ with respect to the CPTs in $\mathcal{N}$; $\text{Seq}(o', o) \in \mathbf{IS}$ be an improving flipping sequence from $o'$ to $o$; $|\text{Seq}(o', o)|$ be the length of $\text{Seq}(o', o)$ and thus the number of improving flips from $o'$ in this sequence is $|\text{Seq}(o', o)| - 1$, then,

$$HD(o', o) \leq |\text{Seq}(o', o)| - 1 \leq \text{pen}(o') - \text{pen}(o)$$
3.3. Penalty score-based heuristic in Dominance Testing

PROOF. As each improving flip flips the value of a single variable, if $\mathcal{N} \models o \succ o'$, there must be at least $HD(o', o)$ flips that flips the value of each variable $X$ that $o'$ and $o$ differ, from $o'[X]$ to $o[X]$. Thus, $\forall Seq(o', o) \in IS$, $|Seq(o', o)| - 1 \geq HD(o', o)$. On the other hand, any improving flip from $\sigma_i$ to $\sigma_{i+1}$ in $Seq(o', o)$ that flips the value of a single variable $X$, $pen(\sigma_i) - pen(\sigma_{i+1}) \geq q_X$ (see the proof of Theorem 3.1). As $\forall X \in V, q_X = 1$, $pen(\sigma_i) - pen(\sigma_{i+1}) \geq 1$. Assume $\sigma$ is an outcome in $Seq(o', o)$ that is improved from $o'$ by $t$ flips, $pen(o') - pen(\sigma) \geq t$ and $pen(o') - t \geq pen(\sigma)$. Thus, $pen(o') - pen(o) = t \geq pen(\sigma) - pen(o)$. If $t > pen(o') - pen(o)$, then $pen(o') - pen(o) - t < 0$ and $pen(\sigma) - pen(o) < 0$. According to Corollary 3.1, $\mathcal{N} \models \sigma \succ o$ or $\mathcal{N} \models \sigma \bowtie o$ ($\mathcal{N} \not\models o \succ \sigma$), contradicting the fact that $\sigma$ is in the improving sequence $Seq(o', o)$. Hence, the number of improving flips from $o'$ in $Seq(o', o)$ can not be greater than $pen(o') - pen(o)$, $|Seq(o, o')| - 1 \leq pen(o') - pen(o)$. □

Lemma 3.2 Given an acyclic CP-net $\mathcal{N}$ and a query $\mathcal{N} \models o \succ o'(o, o' \in O), \forall \sigma \in O$, if $f(\sigma) < 0$, then $\sigma$ would not be part of any possible improving flipping sequence from $o'$ to $o$.

PROOF. During the execution of $DT^*$ algorithm, for any outcome $\sigma$ (including $o'$), $f(\sigma) = pen(\sigma) - HD(\sigma, o) - pen(o)$. Assume that there exist an improving flipping sequence $Seq(\sigma, o) = \sigma_1(= \sigma), \sigma_2, \ldots, \sigma_{\ell-1}, \sigma_\ell(= o)$ from $\sigma$ to the target outcome $o$. Based on Lemma 3.1, we know that there must be at least $HD(\sigma, o)$ flips improved from $\sigma$. For any improving flip from $\sigma_i$ to $\sigma_{i+1}$, $pen(\sigma_i) - pen(\sigma_{i+1}) \geq 1$. Consequently, for any outcome $\sigma'$ that is improved from $\sigma$ by $HD(\sigma, o)$ flips, $pen(\sigma) - pen(\sigma') \geq HD(\sigma, o)$ and thus $pen(\sigma) - HD(\sigma, o) \geq pen(\sigma')$. Hence, $pen(\sigma) - HD(\sigma, o) - pen(o) \geq pen(\sigma') - pen(o)$. Because $f(\sigma) < 0$, $pen(\sigma') - pen(o) \leq pen(\sigma) - HD(\sigma, o) - pen(o) < 0$. Consequently, $pen(\sigma') - pen(o) < 0$ and $\mathcal{N} \not\models o \succ \sigma'$, $\sigma'$ will not be part of any possible improving flipping sequence to $o$, contradicting the fact that there exist an improving flipping sequence $Seq(\sigma, o)$ from $\sigma$ to the target outcome $o$. □

We now prove the completeness of our proposed heuristic algorithm $DT^*$.

Theorem 3.2 $DT^*$ is complete for any arbitrary acyclic CP-nets.
Chapter 3. Penalty score-based heuristic for dominance testing in CP-nets

PROOF. $\mathbb{D}_T^*$ traverses the tree searching all neighbours; it follows lowest evaluated value path and keeps a sorted priority queue of alternate path segments along the way. If at any point the path being followed has a higher evaluated value than other encountered path segments, the higher evaluated value path is kept in the fringe and the process is continued at the lower value sub-path. This continues until the currently considered node for expansion is the target outcome or the fringe is empty.

During the execution of $\mathbb{D}_T^*$ algorithm, there are three kinds of nodes will be pruned: (i) the outcomes that have been explored previously; (ii) the outcomes that improve the value of the variable that is part of some matching suffix with the target outcome; and (iii) the outcomes with negative $f$ values. Obviously, checking repeated nodes does not affect the completeness of the algorithm. Also, as shown in [24], any complete search algorithm for the improving search tree remains complete if pruning using suffix fixing rule is used. Furthermore, we have already proved in Lemma 3.2 that an outcome $\sigma$ with $f(\sigma) < 0$ will not be part of any possible improving sequence from $\sigma'$ to $\sigma$, so pruning the third kind of nodes also does not affect the completeness of the algorithm. Consequently, $\mathbb{D}_T^*$ is complete for any acyclic CP-nets. □

An illustration

We now demonstrate the execution of $\mathbb{D}_T^*$ algorithm with the CP-net in our running example (Figure 3.1(a)) and consider the query $\mathcal{N} \models abcde \Rightarrow \tilde{abc\tilde{d}}$. Let $o = abcde$ and $o' = \tilde{abc\tilde{d}}$, we first consider the $f$ value of the less preferred outcome $o'$ of the query. As $f(o') = pen(o') - HD(o', o) - pen(o) = 10 - 5 - 3 = 2 > 0$, we build the improving search tree $T(o')$ with $o'$ being the root node and add $o'$ into the fringe.

In the $1^{\text{st}}$ iteration of $\mathbb{D}_T^*$, $\sigma = \tilde{abc\tilde{d}}$ is removed from the fringe to be expanded. There are three improvable variable from $\sigma$: $A$, $B$ and $E$. Hence, there are three children nodes: $\tilde{ab\tilde{c}d\tilde{e}}$, $\tilde{ab\tilde{c}d\tilde{e}}$ and $\tilde{a\tilde{b}\tilde{c}d\tilde{e}}$. The $f$ value of these three children nodes are computed accordingly: $f(\tilde{abc\tilde{d}}) = 0$, $f(\tilde{ab\tilde{c}d\tilde{e}}) = 1$ and $f(\tilde{a\tilde{b}\tilde{c}d\tilde{e}}) = 0$. As none of the $f$ value of these three children nodes is negative, all of them are added into the fringe according to the ascendant order of the $f$ value.

In the $2^{\text{nd}}$ iteration, the first outcome $\sigma = \tilde{abc\tilde{d}}$ with the lowest $f$ value is removed from the fringe (Assume that the nodes with the same $f$ value will be traversed in the order from left to right). There are three possible children nodes of $\sigma$: $\tilde{a\tilde{b}\tilde{c}d\tilde{e}}$, $\tilde{ab\tilde{c}d\tilde{e}}$ and $\tilde{a\tilde{b}\tilde{c}d\tilde{e}}$. As $f(\tilde{abc\tilde{d}}) = 5 - 3 - 3 = -1 < 0$; $f(\tilde{ab\tilde{c}d\tilde{e}}) = 6 - 3 - 3 = 0$; and $f(\tilde{a\tilde{b}\tilde{c}d\tilde{e}}) = 2 - 3 - 3 = -4 < 0$. There is only one outcome $\tilde{ab\tilde{c}d\tilde{e}}$ will be added into
3.4. Experiments

In the 3rd iteration, we continue with the outcome $\sigma = \overline{a}bcde$. There are three possible outcomes can be reached by a single flip from $\sigma$: $a\overline{b}c\overline{d}e$, $\overline{a}bcd\overline{e}$ and $\overline{a}bcd\overline{e}$. We compute the $f$ value of these three outcomes: $f(a\overline{b}c\overline{d}e) = 3 - 2 - 3 = -2 < 0$; $f(\overline{a}bcd\overline{e}) = 5 - 2 - 3 = 0$; and $f(\overline{a}bcd\overline{e}) = 1 - 2 - 3 = -4 < 0$. Only one outcome $abcd\overline{e}$ can be added into the fringe.

Similarly, in the 4th iteration, we explore the outcome $\sigma = \overline{a}bcd\overline{e}$ and add only one outcome $abcd\overline{e}$ into the fringe.

In the 5th iteration, we explore the outcome $\sigma = abcd\overline{e}$. In essence, there are two variables can be improved from $\sigma$: $D$ and $E$. However, as $D$ is in the 3rd matching suffix with the target outcome $o$ (using the topological order $A > E > B > C > D$), we only consider flipping the value of $E$. And this step produces the target outcome $o$, which will be explored in the last iteration and the algorithm ends by returning True to this query.

Note that as we have discussed in Section 3.3.1, an algorithm based on Least-variable flipping rule is incomplete in this case.

3.4 Experiments

Extensive computational tests have been carried out. The objectives of the computer experiments are to study experimentally the running time of Algorithm $\mathcal{DT}^*$ for answering dominance testing in CP-nets. All table and graphic outputs are included in Section 3.6 and Section 3.7 for ease of reading.

Random CP-net generator

The random CP-net generator generates an acyclic CP-net in random (with different CP-net structures). Two sets of data have to be generated for an agent’s CP-net, the preference network topology (i.e., dependency graph) and the conditional preference table of each variables.
Network topology

In this chapter, we restrict the preference network to be acyclic, i.e., there is no cycle in the dependency graph. There are three main structures of CP-nets we consider in these experiments, namely, tree-structured CP-nets, directed-path singly connected CP-nets and arbitrary acyclic CP-nets. To generate the network topology, the required input is:

- the number of variables $m$; and
- the maximum node in-degree $d$.

Firstly, a topological ordering $\mathcal{O} = X_{O_1} > \cdots > X_{O_m}$ over the set of domain variables $V = \{X_1, \ldots, X_m\}$ is randomly generated. For different structure CP-nets, we conduct the following steps as to generate the preference network topology.

**Tree-structured CP-nets** The first variable $X_{O_1}$ is considered to be the root of the tree, i.e., $Pa(X_{O_1}) = \emptyset$. Following the topological ordering $\mathcal{O}$, for every variable $X_{O_i}$ ($i \geq 1$), randomly choose at most one variable from $\{X_{O_1}, \ldots, X_{O_{i-1}}\}$ as its parent.

**Directed-path singly connected CP-nets** For the first variable $X_{O_1}$, $Pa(X_{O_1}) = \emptyset$. Then following the topological ordering $\mathcal{O}$, for every variable $X_{O_i}$ ($i > 1$), generate a random integer number $k$ such that $0 \leq k \leq \max\{i-1, d\}$ ($d$ is the maximum node in-degree). If $k = 0$, then $Pa(X_{O_i}) = \emptyset$; Otherwise, randomly pick up a set of no more than $k$ variables from $\{X_{O_1}, \ldots, X_{O_{i-1}}\}$, denoted by $U = \{X_{p_1}, \ldots, X_{p_k}\}$, as the parent variables of $X_{O_i}$ such that the intersection of the ancestors of any two variables in $U$ is $\emptyset$.

**Arbitrary acyclic CP-nets** For the first variable $X_{O_1}$, $Pa(X_{O_1}) = \emptyset$. Then following the topological ordering $\mathcal{O}$, for every variable $X_{O_i}$ ($i > 1$), generate a random integer number $k$ such that $0 \leq k \leq \max\{i-1, d\}$ ($d$ is the maximum node in-degree). If $k = 0$, then $Pa(X_{O_i}) = \emptyset$; Otherwise, randomly pick up a set of $k$ variables from $\{X_{O_1}, \ldots, X_{O_{i-1}}\}$, denoted by $U = \{X_{p_1}, \ldots, X_{p_k}\}$, as the parent variables of $X_{O_i}$.
3.4. Experiments

Conditional preference tables

To generate the conditional preference tables (CPTs), the required input from the users is the set of domain sizes, each of which is for one variable in $V$. For each variable $X \in V$, let $|D(X)|$ be the domain size of $X$ (the set of values that $X$ can take from is $\{0, \ldots, |D(X)| - 1\}$); $Pa(X) = \{X_{p_1}, \ldots, X_{p_k}\}$ be the set of parent variables of $X$. If $X$ is a root variable, i.e., $Pa(X) = \emptyset$, then randomly generate a preference ordering over the set of domain values ($\{0, \ldots, |D(X)| - 1\}$) of $X$. Otherwise, there will be $|D(X_{p_1})| \times \cdots \times |D(X_{p_k})|$ rows in the CPT of $X$. Randomly generate $|D(X_{p_1})| \times \cdots \times |D(X_{p_k})|$ preference orderings over the domain values of $X$ such that the set of preference orderings satisfy the following condition. For every parent variable $X_{p_i}$, for a least one possible assignment $u$ to the other parent variables $u \in D(X_{p_1}) \times \cdots D(X_{p_{i-1}}) \times D(X_{p_{i+1}}) \times \cdots \times D(X_{p_k})$, at least two different values of $X_{p_i}$ will result in two distinct preference orderings over the domain value of $X$.

Experiment Design

Basically the experiments can be divided into six sets in two dimensions: $i)$ different structures, including tree-structured CP-nets, directed path singly connected CP-nets and arbitrary acyclic CP-nets; $ii)$ different domain sizes including those with binary-valued CP-nets, and those on multi-valued CP-nets. The results of experiments show the feasibility of our approach to dominance testing with respect to $i)$ the average number of visited nodes during the search process; $ii)$ the number of variables $s_{var}$ and the domain size $s_{dz}$ that can be efficiently handled in practice; and $iii)$ the structure of CP-nets. We compare the performance of the proposed $DT^*$ algorithm with $i)$ a standard depth-first search algorithm that applies suffix fixing during the search, called $DF$; and $ii)$ an algorithm using Least-variable flipping rule, called $LVF$.

We generate random preference networks by varying the number of variables, the structure of the network and the preference of the variables. For directed-path singly connected CP-nets and arbitrary acyclic CP-nets, we restrict the maximum in-degree of each node in the generated CP-nets to 10. That means, we allow $2^{10} = 1024$ rows maximum in the CPT table of a variable. For multi-valued CP-nets, we restrict the maximum domain size $s_{dz}$ to 5. We conduct the following six sets of experiments. At each set of experiments, we generate 1000 CP-nets randomly and using each resulting preference network, we evaluate 5 dominance queries by picking distinct pairs of
outcomes at random.

Experiment Results

Set 1: binary-valued tree-structured CP-nets. We vary the number of variables $s_{\text{var}}$ from 2 to 30 and only generate tree-structured dependences. From Figure 3.4 we can observe that on average, the numbers of visited nodes by both $\text{DT}^\star$ and $\text{LVF}$ algorithms are much less than $\text{DF}$ algorithm. Note that for binary-valued tree-structured CP-nets, $\text{LVF}$ (Least-variable flipping rule) is guaranteed to be complete and backtrack-free. However, on average, $\text{DT}^\star$ is more efficient than the $\text{LVF}$ algorithm for dominance testing in tree-structured CP-nets. The average execution time of $\text{DT}^\star$ approach with 30 variables is less than 0.03 seconds. It offers more than four orders of magnitude improvement in performance over the $\text{DF}$ algorithm. The average execution time of these approaches with binary-valued tree-structured CP-nets could be found in Table 3.1.

Set 2: multi-valued tree-structured CP-nets. We vary $s_{\text{var}}$ from 2 to 15. The results of multi-valued tree-structured CP-nets (see Figure 3.5(a)) is similar to the set of experiments with binary-valued tree-structured CP-nets. However, $\text{LVF}$ algorithm does not guarantee completeness in multi-valued CP-nets. Figure 3.5(b) shows the percentage of cases in which the $\text{LVF}$ algorithm is incomplete, i.e., it gives an incorrect answer to the query. In general, the percentage of incompleteness of $\text{LVF}$ algorithm is increasing as the number of variables increases. When there are 15 variables, in more than 28% cases that $\text{LVF}$ algorithm fails to find the improving flipping sequence for the given query although there does exist at least one. On the other hand, according to the experiment data, $\text{DT}^\star$ completes the search process in about 5 seconds on average in the cases of 15 variables. The average execution time of these approaches with multi-valued tree-structured CP-nets could be found in Table 3.2.

Set 3: binary-valued, directed-path singly connected CP-nets. In this set of experiments, the number of variables $s_{\text{var}}$ is from 2 to 25. Note that $\text{LVF}$ algorithm guarantees completeness in binary-valued, directed-path singly connected CP-nets while it may require back-tracking during the search. The average number of visited nodes in this set of experiments is shown in Figure 3.6. Both $\text{LVF}$

---

6 The readers can also refer to [24] Page 161, TreeDT algorithm for binary-valued, tree-structured CP-nets
3.4. Experiments

and DT* algorithms are much more efficient than the DF algorithm. When there are 25 variables, the average execution time of DT* is about 5.7 seconds, which is more than two orders of magnitude less than the DF algorithm. The average execution time of these approaches with binary-valued, directed-path singly connected CP-nets could be found in Table 3.3.

**Set 4: multi-valued, directed-path singly connected CP-nets.** We vary $s_{var}$ from 2 to 12. Figure 3.7(a) shows that the average number of visited nodes of both LVF and DT* algorithms are much less than DF algorithm. Although the result shows that when the number of variables is large, the LVF algorithm may visit less nodes than DT* algorithm, the percentage of incompleteness of LVF is on the other hand, increasing as the number of variables increases (see Figure 3.7(b)). When there are 12 variables, this percentage is more than 25%. According to the experimental data, with 12 variables and each variable with the maximum domain size of 5, the average execution time of DT* approach is still less than 10 seconds. The average execution time of these approaches with multi-valued, directed-path singly connected CP-nets could be found in Table 3.4.

**Set 5: binary-valued arbitrary acyclic CP-nets.** We vary $s_{var}$ from 2 to 20. Similar to the results presented in Set 4, when the number of variables is large (more than 15) the average number of visited nodes of DT* algorithm is more than that of LVF algorithm (see Figure 3.8(a)). However, for binary-valued CP-nets in general, LVF does not guarantee completeness and the percentage of cases that the LVF algorithm returns incorrect answers is increasing as the number of variable increases (Figure 3.8(b)). When there are 20 variables, this percentage is more than 20%. While on average, DT* algorithm returns a correct answer to the given query in about 20 seconds. The average execution time of these approaches with binary-valued, arbitrary acyclic CP-nets could be found in Table 3.5.

**Set 6: multi-valued arbitrary acyclic CP-nets.** In the last set of experiments, we vary $s_{var}$ from 2 to 10. The results with arbitrary acyclic CP-nets in multi-valued setting is similar to that in binary-valued setting (see Figure 3.9(a) and Figure 3.9(b)). When there are 10 variables, the percentage of incomplete cases the LVF algorithm is more than 20%; on the other hand, DT* guarantees to return a correct answer in about 9 seconds on average. The average execution time of these approaches with multi-valued, arbitrary acyclic CP-nets could be found in Table 3.6.
In summary, the experiment results show that on average, the proposed DT* algorithm is much more efficient than the DF algorithm. It is as relatively efficient as LVF algorithm while guaranteeing completeness of the search process. From the experiment, it can be clearly seen that the proposed DT* algorithm allows dominance queries for CP-nets that are quite large and complex to be answered in reasonable time.

3.5 Summary

In Chapter 3 the problem of dominance testing in CP-nets was studied. Firstly, a numerical approximation to CP-nets, called penalty scoring function was proposed. Further, the penalty scoring function in dominance testing was applied, presenting an efficient heuristic algorithm DT* for dominance testing with arbitrary acyclic CP-nets. The proposed approach significantly reduces search effort without any impact on completeness. Extensive experimental results have been presented to demonstrate the computational efficiency of the proposed approach. They show that the proposed approach allows dominance queries for CP-nets that are quite large and complex to be answered within a reasonable time.

Nonetheless, the present work is only applicable to acyclic CP-nets. The investigation of techniques to deal with dominance testing with cyclic preferences need to be explored further.
## 3.6 Table Output

### Table 3.1: Average execution time with Tree-structured binary-valued CP-nets (Set 1)

<table>
<thead>
<tr>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.00056</td>
<td>0.00084</td>
<td>0.00083</td>
<td>17</td>
<td>0.3055</td>
<td>0.14979</td>
<td>0.01135</td>
</tr>
<tr>
<td>3</td>
<td>0.00084</td>
<td>0.00111</td>
<td>0.00104</td>
<td>18</td>
<td>0.3918</td>
<td>0.2102</td>
<td>0.01466</td>
</tr>
<tr>
<td>4</td>
<td>0.00123</td>
<td>0.00156</td>
<td>0.00128</td>
<td>19</td>
<td>0.8255</td>
<td>0.29864</td>
<td>0.01408</td>
</tr>
<tr>
<td>5</td>
<td>0.00187</td>
<td>0.00225</td>
<td>0.00174</td>
<td>20</td>
<td>0.8441</td>
<td>0.39323</td>
<td>0.01602</td>
</tr>
<tr>
<td>6</td>
<td>0.002865</td>
<td>0.00349</td>
<td>0.00222</td>
<td>21</td>
<td>1.7078</td>
<td>0.57713</td>
<td>0.01648</td>
</tr>
<tr>
<td>7</td>
<td>0.00427</td>
<td>0.00528</td>
<td>0.00261</td>
<td>22</td>
<td>3.0800</td>
<td>0.80667</td>
<td>0.01797</td>
</tr>
<tr>
<td>8</td>
<td>0.00639</td>
<td>0.00695</td>
<td>0.00338</td>
<td>23</td>
<td>4.5526</td>
<td>1.0221</td>
<td>0.01805</td>
</tr>
<tr>
<td>9</td>
<td>0.01008</td>
<td>0.00979</td>
<td>0.00413</td>
<td>24</td>
<td>9.9563</td>
<td>1.2837</td>
<td>0.01816</td>
</tr>
<tr>
<td>10</td>
<td>0.0152</td>
<td>0.01471</td>
<td>0.00485</td>
<td>25</td>
<td>13.474</td>
<td>1.8437</td>
<td>0.01928</td>
</tr>
<tr>
<td>11</td>
<td>0.0230</td>
<td>0.02062</td>
<td>0.00555</td>
<td>26</td>
<td>20.064</td>
<td>2.4413</td>
<td>0.02079</td>
</tr>
<tr>
<td>12</td>
<td>0.0372</td>
<td>0.02830</td>
<td>0.00572</td>
<td>27</td>
<td>34.399</td>
<td>3.4294</td>
<td>0.02115</td>
</tr>
<tr>
<td>13</td>
<td>0.0615</td>
<td>0.04026</td>
<td>0.00721</td>
<td>28</td>
<td>50.168</td>
<td>5.0714</td>
<td>0.02299</td>
</tr>
<tr>
<td>14</td>
<td>0.0914</td>
<td>0.05665</td>
<td>0.00819</td>
<td>29</td>
<td>132.03</td>
<td>7.3138</td>
<td>0.02551</td>
</tr>
<tr>
<td>15</td>
<td>0.1338</td>
<td>0.08099</td>
<td>0.00835</td>
<td>30</td>
<td>279.25</td>
<td>10.366</td>
<td>0.02944</td>
</tr>
<tr>
<td>16</td>
<td>0.2268</td>
<td>0.11747</td>
<td>0.00865</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

### Table 3.2: Average execution time with Tree-structured multi-valued CP-nets (Set 2)

<table>
<thead>
<tr>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.00153</td>
<td>0.00193</td>
<td>0.00179</td>
<td>11</td>
<td>23.638</td>
<td>0.8690</td>
<td>0.34923</td>
</tr>
<tr>
<td>3</td>
<td>0.00804</td>
<td>0.00383</td>
<td>0.00331</td>
<td>12</td>
<td>123.43</td>
<td>2.0844</td>
<td>0.80864</td>
</tr>
<tr>
<td>4</td>
<td>0.01429</td>
<td>0.00751</td>
<td>0.00438</td>
<td>13</td>
<td>303.39</td>
<td>3.4488</td>
<td>1.1439</td>
</tr>
<tr>
<td>5</td>
<td>0.03985</td>
<td>0.01398</td>
<td>0.00575</td>
<td>14</td>
<td>587.09</td>
<td>6.8618</td>
<td>2.9946</td>
</tr>
<tr>
<td>6</td>
<td>0.10854</td>
<td>0.02679</td>
<td>0.00911</td>
<td>15</td>
<td>1018.1</td>
<td>14.632</td>
<td>4.894</td>
</tr>
<tr>
<td>7</td>
<td>0.39873</td>
<td>0.05774</td>
<td>0.01660</td>
<td>16</td>
<td>NA</td>
<td>25.530</td>
<td>6.807</td>
</tr>
<tr>
<td>8</td>
<td>1.1569</td>
<td>0.10454</td>
<td>0.03285</td>
<td>17</td>
<td>NA</td>
<td>51.488</td>
<td>12.156</td>
</tr>
<tr>
<td>9</td>
<td>2.1986</td>
<td>0.18849</td>
<td>0.06572</td>
<td>18</td>
<td>NA</td>
<td>125.933</td>
<td>36.112</td>
</tr>
<tr>
<td>10</td>
<td>7.2323</td>
<td>0.5429</td>
<td>0.11814</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>


Table 3.3: Average execution time with directed-path singly connected binary-valued CP-nets (Set 3)

<table>
<thead>
<tr>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.000793</td>
<td>0.000699</td>
<td>0.000606</td>
<td>14</td>
<td>0.389</td>
<td>0.0456</td>
<td>0.0226</td>
</tr>
<tr>
<td></td>
<td>0.000793</td>
<td>0.000699</td>
<td>0.000606</td>
<td>15</td>
<td>0.684</td>
<td>0.0728</td>
<td>0.0356</td>
</tr>
<tr>
<td></td>
<td>0.000793</td>
<td>0.000699</td>
<td>0.000606</td>
<td>16</td>
<td>1.39</td>
<td>0.0972</td>
<td>0.0471</td>
</tr>
<tr>
<td></td>
<td>0.000793</td>
<td>0.000699</td>
<td>0.000606</td>
<td>17</td>
<td>1.9</td>
<td>0.129</td>
<td>0.0825</td>
</tr>
<tr>
<td></td>
<td>0.00426</td>
<td>0.00321</td>
<td>0.00211</td>
<td>18</td>
<td>3.29</td>
<td>0.171</td>
<td>0.128</td>
</tr>
<tr>
<td></td>
<td>0.00759</td>
<td>0.00464</td>
<td>0.00293</td>
<td>19</td>
<td>8.08</td>
<td>0.228</td>
<td>0.213</td>
</tr>
<tr>
<td></td>
<td>0.0134</td>
<td>0.0066</td>
<td>0.00348</td>
<td>20</td>
<td>11.1</td>
<td>0.297</td>
<td>0.423</td>
</tr>
<tr>
<td></td>
<td>0.00224</td>
<td>0.00951</td>
<td>0.00499</td>
<td>21</td>
<td>22.5</td>
<td>0.398</td>
<td>0.601</td>
</tr>
<tr>
<td></td>
<td>0.0432</td>
<td>0.013</td>
<td>0.00658</td>
<td>22</td>
<td>39.1</td>
<td>0.511</td>
<td>0.809</td>
</tr>
<tr>
<td></td>
<td>0.0626</td>
<td>0.0183</td>
<td>0.00816</td>
<td>23</td>
<td>58.6</td>
<td>0.669</td>
<td>1.5</td>
</tr>
<tr>
<td></td>
<td>0.14</td>
<td>0.025</td>
<td>0.0117</td>
<td>24</td>
<td>131.</td>
<td>0.884</td>
<td>2.66</td>
</tr>
<tr>
<td></td>
<td>0.273</td>
<td>0.0343</td>
<td>0.0147</td>
<td>25</td>
<td>632.</td>
<td>1.18</td>
<td>5.68</td>
</tr>
</tbody>
</table>

Table 3.4: Average execution time with directed-path singly connected binary-valued CP-nets (Set 4)

<table>
<thead>
<tr>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.00165</td>
<td>0.00185</td>
<td>0.00133</td>
<td>9</td>
<td>15.3</td>
<td>0.169</td>
<td>1.2</td>
</tr>
<tr>
<td></td>
<td>0.0042</td>
<td>0.00325</td>
<td>0.00215</td>
<td>10</td>
<td>93.1</td>
<td>0.322</td>
<td>2.00</td>
</tr>
<tr>
<td></td>
<td>0.0184</td>
<td>0.00649</td>
<td>0.00385</td>
<td>11</td>
<td>425.</td>
<td>0.484</td>
<td>4.3</td>
</tr>
<tr>
<td></td>
<td>0.00273</td>
<td>0.00217</td>
<td>0.00162</td>
<td>12</td>
<td>1900.</td>
<td>0.97</td>
<td>9.077</td>
</tr>
<tr>
<td></td>
<td>0.116</td>
<td>0.0272</td>
<td>0.0172</td>
<td>13</td>
<td>NA</td>
<td>1.993</td>
<td>22.996</td>
</tr>
<tr>
<td></td>
<td>0.547</td>
<td>0.0501</td>
<td>0.0525</td>
<td>14</td>
<td>NA</td>
<td>2.628</td>
<td>42.363</td>
</tr>
<tr>
<td></td>
<td>2.24</td>
<td>0.0951</td>
<td>0.276</td>
<td>15</td>
<td>NA</td>
<td>4.378</td>
<td>104.33</td>
</tr>
</tbody>
</table>

Table 3.5: Average execution time with arbitrary acyclic binary-valued CP-nets (Set 5)

<table>
<thead>
<tr>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.000571</td>
<td>0.000703</td>
<td>0.000725</td>
<td>12</td>
<td>0.26</td>
<td>0.0401</td>
<td>0.0335</td>
</tr>
<tr>
<td></td>
<td>0.000887</td>
<td>0.00101</td>
<td>0.000953</td>
<td>13</td>
<td>0.599</td>
<td>0.0534</td>
<td>0.0496</td>
</tr>
<tr>
<td></td>
<td>0.00144</td>
<td>0.00154</td>
<td>0.00132</td>
<td>14</td>
<td>1.47</td>
<td>0.0778</td>
<td>0.0836</td>
</tr>
<tr>
<td></td>
<td>0.00251</td>
<td>0.00227</td>
<td>0.00179</td>
<td>15</td>
<td>2.21</td>
<td>0.109</td>
<td>0.241</td>
</tr>
<tr>
<td></td>
<td>0.00468</td>
<td>0.00345</td>
<td>0.00239</td>
<td>16</td>
<td>5.49</td>
<td>0.15</td>
<td>0.537</td>
</tr>
<tr>
<td></td>
<td>0.0086</td>
<td>0.00526</td>
<td>0.00334</td>
<td>17</td>
<td>18.</td>
<td>0.207</td>
<td>1.45</td>
</tr>
<tr>
<td></td>
<td>0.0164</td>
<td>0.00783</td>
<td>0.00475</td>
<td>18</td>
<td>40.1</td>
<td>0.273</td>
<td>2.49</td>
</tr>
<tr>
<td></td>
<td>0.0342</td>
<td>0.0115</td>
<td>0.00704</td>
<td>19</td>
<td>79.9</td>
<td>0.359</td>
<td>6.91</td>
</tr>
<tr>
<td></td>
<td>0.0639</td>
<td>0.0194</td>
<td>0.0131</td>
<td>20</td>
<td>182.</td>
<td>0.49</td>
<td>19.1</td>
</tr>
<tr>
<td></td>
<td>0.132</td>
<td>0.0277</td>
<td>0.0202</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
### Table 3.6: Average execution time with arbitrary acyclic multi-valued CP-nets (Set 6)

<table>
<thead>
<tr>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
<th>Variable</th>
<th>DF</th>
<th>LVF</th>
<th>DT*</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.00159</td>
<td>0.00191</td>
<td>0.00133</td>
<td>8</td>
<td>8.95</td>
<td>0.081</td>
<td>0.276</td>
</tr>
<tr>
<td>3</td>
<td>0.0047</td>
<td>0.00331</td>
<td>0.00215</td>
<td>9</td>
<td>57.5</td>
<td>0.152</td>
<td>1.29</td>
</tr>
<tr>
<td>4</td>
<td>0.021</td>
<td>0.00687</td>
<td>0.00385</td>
<td>10</td>
<td>311</td>
<td>0.297</td>
<td>8.17</td>
</tr>
<tr>
<td>5</td>
<td>0.0829</td>
<td>0.0135</td>
<td>0.00751</td>
<td>11</td>
<td>NA</td>
<td>0.410</td>
<td>12.133</td>
</tr>
<tr>
<td>6</td>
<td>0.419</td>
<td>0.0246</td>
<td>0.0172</td>
<td>12</td>
<td>NA</td>
<td>0.916</td>
<td>26.557</td>
</tr>
<tr>
<td>7</td>
<td>1.97</td>
<td>0.0464</td>
<td>0.0525</td>
<td>13</td>
<td>NA</td>
<td>1.9333</td>
<td>36.880</td>
</tr>
</tbody>
</table>
3.7 Graphic Output

Figure 3.4: Avg. number of visited nodes with binary-value tree-structured CP-nets

Figure 3.5: Multi-valued tree-structured CP-nets
3.7. Graphic Output

Figure 3.6: Avg. number of visited nodes with binary-valued, directed-path singly connected CP-nets

![Graph showing Avg. number of visited nodes with binary-valued, directed-path singly connected CP-nets.](image)

(a) Avg. number of visited nodes

(b) LVF Incompleteness

Figure 3.7: Multi-valued, directed-path singly connected CP-nets

![Graph showing Multi-valued, directed-path singly connected CP-nets.](image)
Chapter 3. Penalty score-based heuristic for dominance testing in CP-nets

Figure 3.8: Binary-value arbitrary acyclic CP-nets

(a) Avg. number of visited nodes

(b) LVF Incompleteness

Percentage

Variables

Nodes

Variables

5 10 15 20

0 5 10 15 20 25 30

5 10 15 20

Figure 3.8: Binary-value arbitrary acyclic CP-nets
3.7. Graphic Output

Figure 3.9: Multi-valued arbitrary acyclic CP-nets
In Chapter 4, the problem of collective decision-making in combinatorial domains in which the agents’ preferences are represented by CP-nets and TCP-nets is studied. The qualitative and structural features of CP-nets and TCP-nets enable us to easily encode human preferences. However, many group decision-making methods require numerical measures of degrees of desirability of alternative outcomes. The last chapter introduces a penalty scoring function for acyclic CP-nets. In Chapter 4, we go one step further by incorporating the relative importance information among pairs of variables. A penalty scoring function is presented in Chapter 4 as a numerical approximation for both acyclic CP-nets and conditionally acyclic TCP-nets. After the individual penalty scores are computed, a collective penalty scoring function is further defined to aggregate multiple agents’ preferences. As the outcome space is exponentially large in combinatorial domains, an heuristic algorithm for computing an optimal collective decision based on penalty scores is also presented.
4.1 Introduction

The preference information provided by a CP-net or a TCP-net on its own is purely qualitative. However, many group decision-making methods require numerical measures of degrees of desirability of alternative outcomes. Moreover, quantitative comparison is typically less computationally intensive than qualitative comparison, which in CP-nets (or TCP-nets) involves dominance testing between different outcomes. Recent work shows that dominance testing for an arbitrary CP-net (and, thus, TCP-net) is NP-hard [71]. Collective decision-making with multiple agents’ CP-nets (or TCP-nets) may further require dominance testing on each pair of outcomes on each individual CP-net (or TCP-net). For example, having ten binary variables, each involved agent would need to compare \(2^{10} \choose 2 = 523,776\) pairs of outcomes. This problem is likely to be even harder than NP or coNP problems.

To permit a natural way of preference elicitation while providing quantitative comparisons, this chapter first extends the penalty scoring function for CP-nets introduced in Chapter 3, presenting a penalty scoring function as a numerical approximation not only for acyclic CP-nets, but also for conditionally acyclic TCP-nets. The proposed approach preserves all the strict preference ordering induced by the original CP-net and TCP-net, and provides a numerical measurement of desirability of alternative outcomes. Moreover, it provides an easy way of making preferential comparisons, i.e., both the function construction and the penalty score computation with an outcome is polynomial in the size of the network.

Given that the individual preferences have been elicited and represented as CP-nets or TCP-nets, the problem of preference aggregation will be addressed in the following two steps: i) the penalty score of a particular outcome of each agent is computed, based on her individual penalty scoring function (independent of the other agents); ii) then the individual penalty scores are aggregated into a normalised collective penalty scoring function. The collective penalty scoring function models the preferences of a group of agents, and forms the basis in group decision-making. As the outcome space is exponentially large in combinatorial domains, this chapter also addresses the collective decision-making problem by presenting an efficient heuristic algorithm, called HeTCP, to search for an optimal collective decision according to the given aggregation rule.

The remainder of this chapter is organised as follows. We present the proposed pen-
Chapter 4. Penalty score-based preference aggregation with CP-nets and TCP-nets

4.2 Penalty scoring functions for TCP-nets

Chapter 3 introduces a penalty scoring function for approximating the preference relations induced by acyclic CP-nets. This section goes one step further by incorporating the relative importance information among pairs of variables and introduces a penalty scoring function as a numerical approximation not only for acyclic CP-nets, but also for conditionally acyclic TCP-nets (trade-offs-enhanced CP-nets). In broad terms, given a conditionally acyclic TCP-net, we generate a penalty scoring function representing that TCP-net in the following steps. First, we assign an importance weight to each variable based on the structure of the given TCP-net. Next, a penalty scoring function is defined based on penalty analysis.

Given a TCP-net $\mathcal{N}$, we denote $\mathcal{N}^*$ as its dependency graph (see Definition 2.3). For a variable $X$ in $\mathcal{N}^*$, we call the variables $Y$ s.t. $\langle X, Y \rangle \in \text{cp} \cup \text{sci}$ as the dependants of $X$. Based on the dependency structure of a conditionally acyclic TCP-net, we define the following notion of strict topological ordering over variables.
4.2. Penalty scoring functions for TCP-nets

**Definition 4.1 (Strict topological ordering)** Given a dependency graph $N^*$ of a conditionally acyclic TCP-net $N$, a strict topological ordering of variables for $N^*$ is an ordering over the domain variables $X_1 > \cdots > X_m$, such that for all $\{\overline{X_i}, \overline{X_j}\} \in cp \cup i \cup sci$, we have $X_i > X_j$, and for all $X_k$ that $(X_i, X_k) \in ci$, we have $X_k > X_j$.

**Example.** Consider an agent’s TCP-net $N$ in Figure 4.1, the dependency graph $N^*$ of this TCP-net is shown in Figure 4.2(a). It contains all the nodes and arcs of $N$ in Figure 4.1. In addition, as there is a ci-arc $\gamma = (B, C)$ in $N$ and $S(\gamma) = \{A\}$, there is a directed sci-arc $\langle A, \overline{C} \rangle$ in $N^*$. Note that there is no sci-arc $\langle A, \overline{C} \rangle$ in $N^*$ because there already exists an cp-arc $\langle A, \overline{B} \rangle$ in $N$. Two possible strict topological orders of $N^*$ is $A > B > C > D$ or $A > C > B > D$. Note that $B$ must be ordered before $D$ even though there is no directed path between them in the dependency graph, because $\langle C, \overline{D} \rangle \in i$ and $(B, C) \in ci$. For the TCP-nets without any ci-arc, there is no additional directed arc in the dependency graph.

Now we assign preferences over the alternative outcome space using penalty scores. For a variable $X$, let $|D(X)|$ be the domain size of $X$ and thus there are $|D(X)|$ degrees of penalties. Without loss of generality, we assume the degree of penalties of a variable $X$ range between 0 and $|D(X)| - 1$, that is, $d_1 = 0, \ldots, d_{|D(X)|} = |D(X)| - 1$. For the TCP-nets in Figure 4.1, since all variables are binary, there are only two degrees of penalties, i.e., $d_1 = 0$ and $d_2 = 1$. For a variable $X$, consider a preference ordering over the value of $X$ given an instantiation of $X$’s parents, let the rank of the most preferred value of $X$ be 0 and the rank of the least preferred value of $X$ be $|D(X)| - 1$. Consequently, given an outcome $o$, the degree of penalty of a variable $X$ in $o$ is the rank of the value $o[X]$ in the preference ordering over $X$ given the parent context $u = o[Pa(X)]$. We denote by $d_X^o (d_X^o \in \{d_1, \ldots, d_{|D(X)|}\})$ the degree of penalty of $X$ with respect to $o$. For instance, consider a variable $X$ such that $D(X) = \{x, x', x''\}$. Assume that, under a parent context $u = o[Pa(X)]$ assigned by an outcome $o$, $x > x' > x''$. Hence, if $o[X] = x$, then $d_X^o = d_1 = 0$; if $o[X] = x'$, then $d_X^o = d_2 = 1$; if $o[X] = x''$, then $d_X^o = d_3 = 2$.

We now analyse the *importance weights* of variables in a TCP-net. TCP-nets impose a rich structure to allow variables to have different degrees of importance: variables “higher-up” in the structure of the network are considered to be more important than the lower level variables [25, 30]. Moreover, as the importance relations among pairs of variables can be context-dependent (i.e., conditional relative importance), the importance weight of a variable can also be context-dependent. Consequently, we first
assign the value to each arc in the dependency graph of the given TCP-net, then, we analyse the importance weight of a variable $X$ in a particular outcome $o$ by considering (i) the values of the directed $cp$-, $i$- and $sci$-arcs $\langle \overrightarrow{X,Y} \rangle$ that originate at $X$; and (ii) the values of the $ci$-arcs $\gamma = (X,Y) \in ci$ s.t. $X \triangleright Y$ given $z = o[S(\gamma)]$. Moreover, as the importance weight of a variable is context-dependent, when we assign the value to an arc $\gamma$, we consider the upper bound weight of the variable that $\gamma$ points to. The upper bound weight of a variable $X$ is computed under the assumption that for all $ci$-arc $\langle X,Y \rangle \in ci$, $X$ is contextually more important than $Y$. Furthermore, we allow users to define an additional importance weight $q_X$ for each variable $X$ ($q_X > 0$), which specifies how important $X$ itself is to agent $i$, without considering the influences of $X$ on its dependants. We now provide the formal definition of upper bound variable weight and arc value as follows:

**Definition 4.2 (Upper bound weight and arc value)** Given a dependency graph $N^*$,
of an acyclic TCP-net $N$. For each variable $X$ in $N^*$, let $cp[X]$, $i[X]$ and $sci[X]$ denote the set of directed $cp$-, $i$- and $sci$-arcs that originate at $X$, respectively; and $ci[X]$ the set of $ci$-arcs that connect $X$, the upper bound variable weight and arc value in $N^*$ is recursively defined as follows:

The upper bound weight $w^{ub}(X)$ of a variable $X$ is the importance weight of $X$ given the assumption under which for all $\gamma = (X, Y) \in ci[X]$, $X \triangleright Y$:

$$w^{ub}(X) = \max \left( \max_{\gamma \in i[X]} v(\gamma), \max_{\gamma \in ci[X]} v^{X \triangleright Y}(\gamma) \right) + \sum_{\gamma \in cp[X] \cup sci[X]} v(\gamma) + q_X$$

(4.1)

where

- $q_X (q_X > 0)$ is the user-defined weight for $X$.
- for any directed arc $\gamma = (X, Y) \in cp[X] \cup sci[X]$:

$$v(\gamma) = w^{ub}(Y) \cdot (|D(Y)| - 1)$$

- for any directed arc $\gamma = (X, Y) \in i[X]$:

$$v(\gamma) = w^{ub}(Y) \cdot |D(Y)|$$

- for any undirected arc $\gamma = (X, Y) \in ci[X]$, $v^{X \triangleright Y}(\gamma)$ denotes the value of $\gamma$ given the condition under which $X$ is more important than $Y$.

$$v^{X \triangleright Y}(\gamma) = w^{ub|-\gamma}(Y) \cdot |D(Y)|$$

where $w^{ub|-\gamma}(Y) (\gamma \in ci[Y])$ is the importance weight of $Y$ given the assumption under which for all $\gamma' = (Y, Z) \in ci[Y] \setminus \{\gamma\}$, $Y \triangleright Z$; that is,

$$w^{ub|-\gamma}(Y) = \max \left( \max_{\gamma' \in i[Y]} v(\gamma'), \max_{\gamma' \in ci[Y] \setminus \{\gamma\}} v^{Y \triangleright Z}(\gamma') \right) + \sum_{\gamma' \in cp[Y] \cup sci[Y]} v(\gamma') + q_Y$$

(4.2)

\footnote{Every ci-arc $\gamma = (X, Y)$ has two values: $v^{X \triangleright Y}(\gamma)$ and $v^{Y \triangleright X}(\gamma)$. $v^{X \triangleright Y}(\gamma)$ (resp. $v^{Y \triangleright X}(\gamma)$) denotes the value of $\gamma$ given the condition under which $X$ is more important than $Y$ (resp. $Y$ is more important than $X$). For example, consider the ci-arc $\gamma = (B, C)$ in Figure 4.2(b), $v^{B \triangleright C}(\gamma)$ is the value of $\gamma$ when $B \triangleright C$; and $v^{C \triangleright B}(\gamma)$ is the value of $\gamma$ when $C \triangleright B$.}
Chapter 4. Penalty score-based preference aggregation with CP-nets and TCP-nets

Input: $N^*$, a dependency graph of a conditionally acyclic TCP-net
Order variables of $N^*$ in a reverse strict topological ordering;

For each $X \in N^*$ do
  For each $\gamma = \langle X, Y \rangle \in cp[X] \cup sci[X]$ do
    if $v(\gamma) = Null$ then
      $v(\gamma) \leftarrow w^{ub}(Y) \cdot (|D(Y)| - 1)$;
    end
  end
  For each $\gamma = \langle X, Y \rangle \in i[X]$ do
    if $v(\gamma) = Null$ then
      $v(\gamma) \leftarrow w^{ub}(Y) \cdot |D(Y)|$;
    end
  end
  For each $\gamma = (X, Y) \in ci[X]$ do
    if $v_{X\rightarrow Y}(\gamma) = Null$ then
      $v_{X\rightarrow Y}(\gamma) \leftarrow assgci(\gamma, X, Y)$;
    end
  end
  assign $w^{ub}(X)$ according to Equation (4.1);
end

Algorithm 4.1: assgTCP($N^*$)

Given a conditionally acyclic TCP-net, assigning the arc values and compute the upper bound weights of the variables in the dependency graph take polynomial time in the size of the network (see Algorithm 4.1). Algorithm 4.1 considers the variables of $N^*$ in a reverse strict topological order (Algorithm 4.1; line 1). For each variable $X$, we first assign the value to the directed arcs (i.e., cp-, i- and sci-arcs) that originate at $X$ (Algorithm 4.1; line 3–12). Then we assign the value to the undirected arcs (i.e., ci-arcs) that connect $X$ (Algorithm 4.1; line 13–17). Finally, we assign the upper bound weight of $X$ (Algorithm 4.1; line 18).

Assigning value to a ci-arc is recursive (see Algorithm 4.2). In order to compute the value of a ci-arc $\gamma = (X, Y)$ under the condition of $X \rightarrow Y$, i.e., $assg-ci((X, Y), X \rightarrow Y)$, we first assign the value to the directed arcs (i.e., cp-, i- and sci-arcs) that originate at $Y$ (Algorithm 4.2; line 2–10). Then the computation is reduced to calculating the value of each ci-arc $\gamma' = (Y, Z) \in ci[Y] \setminus \{\gamma\}$ that connects $Y$ under the condition of $Y \rightarrow Z$ (Algorithm 4.2; line 12–16). Eventually it reaches the step of computing a ci-arc $\gamma = (U, V)$ such that $\gamma$ is the only ci-arc connecting $V$ (i.e., $ci[V] \setminus \{\gamma\} = \emptyset$), it then calculates $w^{ub}\gamma(V)$, and computes the arc value $v^{U\rightarrow V}(\gamma)$ and returns $v^{U\rightarrow V}(\gamma)$ to the
### 4.2. Penalty scoring functions for TCP-nets

**Algorithm 4.2: assg-ci** \( ((X,Y), X \triangleright Y) \)

```plaintext
Input: \((X,Y)\), a ci-arc; \(X \triangleright Y\), a condition for this ci-arc
1. \(\gamma \leftarrow (X,Y)\;
2. For each \(\gamma' = (Y,Z) \in \text{cp}[Y] \cup \text{sci}[Y]\) do
   3. if \(v(\gamma') = \text{Null}\) then
      4. \(v(\gamma') \leftarrow w_{ub}(Z) \cdot (|D(Z)| - 1)\);
   5. end
   6. end
7. For each \(\gamma' = (Y,Z) \in i[Y]\) do
   8. if \(v(\gamma') = \text{Null}\) then
      9. \(v(\gamma') \leftarrow w_{ub}(Z) \cdot |D(Z)|\);
   10. end
11. end
12. For each \(\gamma' = (Y,Z) \in \text{ci}[Y] \setminus \{\gamma\}\) do
13. if \(v^{Y\triangleright Z} (\gamma') = \text{Null}\) then
14. \(v^{Y\triangleright Z} (\gamma') \leftarrow \text{assg-ci}(\gamma', Y \triangleright Z)\);
15. end
16. end
17. assign \(w^{ub \downarrow \gamma} (Y)\) according to Equation (4.2);
18. \(v^{X \triangleright Y} (\gamma) \leftarrow w^{ub \downarrow \gamma} (Y) \cdot |D(Y)|\);
19. return \(v^{X \triangleright Y} (\gamma)\)
```

**Example.** Consider the dependency graph \(\mathcal{N}^{*}\) (Figure 4.2(a)) of agent 1 in our running example. In a simple setting: \(\forall X \in V\), \(q_X = 1\) and \(\forall X \in V\), \(|D(X)| = 2\). We follow a reverse strict topological ordering \(DCBA\). For variable \(D\), as \(\text{cp}[D] = i[D] = \text{ci}[D] = \text{sci}[D] = 0\), \(w_{ub}(D) = q_D = 1\). We then analyse the upper bound weight of variable \(C\), \(\text{cp}[C] = \text{sci}[C] = 0\), \(i[C] = \{\langle C, D \rangle\}\) and \(\text{ci}[C] = \{(B,C)\}\).\n
\[
v(C, D) = w_{ub}(D) \cdot |D(D)| = 2.\]

Let \(\gamma = (B, C)\), it then calls Algorithm 4.2 to assign the value of \(\gamma\) for variable \(C\) \((v^{C \triangleright B}(\gamma))\), i.e., \(\text{assg-ci}(\gamma, C \triangleright B)\). As \(B\) has only one ci-arc \(\gamma\) and \(\text{cp}[B] = i[B] = \text{sci}[B] = 0\), \(w^{ub \downarrow \gamma} (B) = q_2 = 1\), and thus \(v^{C \triangleright B}(\gamma) = w^{ub \downarrow \gamma} (B) \cdot |D(B)| = 2.\) \(w_{ub}(C) = \max \left( v(C, \hat{D}), v^{C \triangleright B}(\gamma) \right) + q_C = \max (2, 2) + 1 = 3.\) Then for variable \(B\): \(v^{B \triangleright C}(\gamma) = w^{ub \downarrow \gamma} (C) \cdot |D(C)| = \left( \max \left( v(C, \hat{D}) \right) + q_C \right) \ast 2 = (2+1) \ast 2 = 6 \) and \(w_{ub}(B) = \max \left( v^{B \triangleright C}(\gamma) \right) + q_B = 7.\) Eventually for variable \(A\), \(\text{cp}[A] = \{v(\langle A, C \rangle)\}; \text{sci}[A] = \{v(\langle A, B \rangle)\};\) and \(\text{ci}[A] = i[A] = 0. v(\langle A, B \rangle) = w_{ub}(B) \cdot (|D(B)| - 1) = 7 \) and \(v(\langle A, C \rangle) = w_{ub}(C) \cdot (|D(C)| - 1) = 3.\) We have \(w_{ub}(A) = \left( v(\langle A, B \rangle) + v(\langle A, C \rangle) \right) + q_A = \)
7 + 3 + 1 = 11.

**Definition 4.3 (Context-dependent importance weight)** Given a dependency graph \( N^* \) of a conditionally acyclic TCP-net \( N \) and consider an outcome \( o \). For each variable \( X \), let \( ci^*[X] \) denotes the subset of \( ci[X] \) such that \( \forall \gamma = (X,Y) \in ci^*[X]\), \( X > Y \) given \( o|S(\gamma) \), the importance weight of a variable \( X \) in \( o \), denoted by \( w^o(X) \), is defined by:

\[
 w^o(X) = \max \left( \max_{\gamma \in [X]} v(\gamma), \max_{\gamma \in ci^*[X]} v^{X>Y}(\gamma) \right) + \sum_{\gamma \in cp[X] \cup sci[X]} v(\gamma) + q_X \quad (4.3)
\]

In essence, only the importance weight of the variables with at least one \( ci \)-arc (e.g. \( B \) and \( C \) in Figure 4.1) may vary from different outcomes for an agent. The variables without \( ci \)-arc (e.g. \( A \) and \( D \) in Figure 4.1) have fixed importance weight that equal to their upper bound weight.

**Example.** Consider the preference of agent 1 in our running example (see Figure 4.1 and Figure 4.2) and the outcome \( o = \bar{abcd} \). The importance weight of each variable in \( o \) is as follows. As \( ci[A] = \emptyset \), \( w^o(A) = w^{ub}(A) = 21 \). According to the CIT of \( \gamma = (B,C), C > B \) given \( A = a \) (see Figure 4.1). Thus, \( ci^*[B] = \emptyset \) and \( ci^*[C] = \gamma \). \( w^o(B) = q_B = 1 \) and \( w^o(C) = \max \left( v((C,D)), v^{C>B}(\gamma) \right) + q_C = 2 + 1 = 3 \). Finally, \( ci[D] = \emptyset \) and thus, \( w^o(D) = w^{ub}(D) = 1 \).

Given a TCP-net \( N \) and an outcome \( o \), the penalty of a variable \( X \) in \( o \) is the degree of penalty of \( X \) in \( o \), i.e. \( d^o_X \), multiplied by the importance weight of \( X \) in \( o \), i.e. \( w^o(X) \). Then we can analyse the penalty score of an outcome by considering the penalties of variables in that outcome.

As to avoid the problem of scale sensitivity, we normalize the penalty score of an outcome into the interval \([0,1]\) and define the following penalty scoring function.

**Definition 4.4 (Individual penalty scoring function)** Given a conditionally acyclic TCP-net \( N \), let \( \hat{o} \) be the worst outcome of \( N \); \(^2\) total(\( \hat{o} \)) = \( \sum_{X \in V} w^\hat{o}(X) \cdot d^\hat{o}_X \) is
4.2. Penalty scoring functions for TCP-nets

the sum of the penalties of the variables in \( \tilde{o} \), \(^3\) the individual penalty scoring function of \( \mathcal{N} \), denoted by pen, mapping from \( O \) to \([0, 1]\) is defined by:

\[
\forall o \in O, \quad \text{pen} (o) = \frac{\sum_{X \in V} w^o(X) \cdot d^o_X}{\text{total} (\tilde{o})} \tag{4.4}
\]

Constructing the individual penalty scoring function, in essence, only requires assigning the arc values in the dependency graph. As mentioned before, this can be done in \textit{polynomial} time in the size of the network. The computation of the penalty score for a particular outcome \( o \) on the other hand, also takes \textit{polynomial time} in the size of the network. Informally, we simply need to sweep through the network following a strict topological order of variables, compute the context-dependent importance weight and check the degree of penalty of each variable \( X \) in \( o \), and finally calculate the penalty score based on Equation (4.4).

\textbf{Example.} In our running example, suppose that we need to calculate the penalty score of the outcome \( o = abcd \) with respect to agent 1’s TCP-net (Figure 4.1 and Figure 4.2). The worst outcome of \( \mathcal{N} \) is \( \tilde{o} = \bar{a}bcd \). For the ci-arc \( \gamma = (B, C) \), \( B \triangleright C \) given \( \tilde{o} [A] = \bar{a} \). Thus, \( w^\tilde{o} (B) = \max \left( v^{B \triangleright C} (\gamma) \right) + q_B = 7 \) and \( w^\tilde{o} (C) = \max \left( v^{C \triangleright D} (\gamma) \right) + q_C = 3 \). As all variables are binary, \( \forall X \in V, \quad d^\tilde{o}_X = 1 \) and \( \text{total} (\tilde{o}) = w^\tilde{o} (A) \cdot 1 + w^\tilde{o} (B) \cdot 1 + w^\tilde{o} (C) \cdot 1 + w^\tilde{o} (D) \cdot 1 = 11 + 7 + 3 + 1 = 22 \).

On the other hand, for the outcome \( o = \bar{a}bcd \), \( A \) (resp. \( C \)) is assigned a preferred value by \( o \) w.r.t. \( \text{CPT}(A) \) (resp. \( \text{CPT}(C) \)), hence \( d^o_A = 0 \) (resp. \( d^o_C = 0 \)). \( B \) (resp. \( D \)) is assigned a less preferred value by \( o \) w.r.t. \( \text{CPT}(B) \) (resp. \( \text{CPT}(D) \)), hence \( d^o_B = 1 \) (resp. \( d^o_D = 1 \)). As \( C \triangleright B \) given \( o [A] = a, w^o (B) = 1 \) and \( w^o (C) = 3 \). Consequently, \( \text{pen} (o) = \frac{w^o (A) \cdot 1 + w^o (B) \cdot 1 + w^o (C) \cdot 1 + w^o (D) \cdot 1}{\text{total} (\tilde{o})} = \frac{11 + 1 + 3 + 1}{22} = \frac{2}{22} \).

The penalty scoring function \( \text{pen} \) of a conditionally acyclic TCP-net induces a weak total order, i.e., a transitive, reflexive and total relation, over the entire outcome space. We now prove that this order preserves all strict preferences induced by the original TCP-net.

\textbf{Theorem 4.1} \textit{The penalty scoring function} \( \text{pen} \) \textit{of a conditionally acyclic TCP-net} \( \mathcal{N} \)

\(^3\)Note that as \( \tilde{o} \) is the worst outcome with respect to \( \mathcal{N}, \forall X \in V, \quad d^\tilde{o}_X = d_{|D(X)|} = |D(X)| - 1.\)
Chapter 4. Penalty score-based preference aggregation with CP-nets and TCP-nets

preserves all the strict preferences induced by \( \mathcal{N} \), i.e.

\[
\forall o, o' \in O, \text{ if } \mathcal{N} \models o \succ o' \text{ then } \text{pen}(o') > \text{pen}(o)
\]

PROOF. \( \forall o, o' \in O, \mathcal{N} \models o \succ o' \) iff there exists a sequence of improving flips from \( o' \) to \( o \) with respect to \( \mathcal{N} \), denoted by \( \text{Seq}(o', o) = \sigma_1 (= o'), \sigma_2, \ldots, \sigma_{t-1}, \sigma_t (= o) \). Each improving flip from \( \sigma_p \) to \( \sigma_{p+1} \) in \( \text{Seq}(o', o) \) is either a CP-flip or an I-flip. If it is a CP-flip that flips the value of a variable \( X \), the resulting changes from \( \sigma_p \) to \( \sigma_{p+1} \) may include: (i) the degree of penalty of \( X \) decreases from \( d^p_X \) to \( d^{p+1}_X \) \( (d^p_X > d^{p+1}_X) \); (ii) the degrees of penalty of \( X \)'s children changes, which in the worst case, results in the degree of penalty of each children \( Y \) of \( X \) increasing from \( d^p_Y = d \) to \( d^{p+1}_Y = d_{|D(Y)|} \); and (iii) the importance weights of the variables \( Y \) and \( Z \), where \( \gamma = (Y, Z) \in \mathcal{ci} \) and \( X \in \mathcal{S} (\gamma) \). If it is an I-flip that flips the values of two variables \( X \) and \( Y \), \( X \triangleright Y \) given the condition by \( \sigma_p \) (and \( \sigma_{p+1} \)), it may result in additional changes: (iv) the degree of penalty of \( Y \) increases, which in the worst case, results in the degree of penalty of \( Y \) increases from \( d^p_Y = d \) to \( d^{p+1}_Y = d_{|D(Y)|} \); (v) the degrees of penalty of \( Y \)'s children changes, which in the worst case, results in the degree of penalty of each children \( Z \) of \( Y \) increasing from \( d^p_Z = d \) to \( d^{p+1}_Z = d_{|D(Z)|} \); and (vi) the importance weights of the variables \( Z \) and \( W \), where \( \gamma = (Z, W) \in \mathcal{ci} \) and \( Y \in \mathcal{S} (\gamma) \). As \( d^p_X - d^{p+1}_X \geq 1 \), \( \delta^p(X) = \delta^{p+1}(X) \) and \( \delta^p(Y) = \delta^{p+1}(Y) \), \( \delta^p(X) \cdot \delta^{p+1}_X \geq \delta^p(X) \).

According to Definition 4.2, for each arc \( \gamma = \langle X, Y \rangle \in \text{cp} \cup \text{sci} \) (resp. \( \gamma = \langle X, Y \rangle \in \text{ci} \) and \( \gamma = \langle X, Y \rangle \in \text{ci} \)), the arc value \( v(\gamma) = w^{ab}(Y) \cdot (|D(Y)| - 1) \) (resp. \( v(\gamma) = w^{ab}(Y) \cdot |D(Y)| - 1 \) + \( w^{ab}(Y) \)) and \( v^{X \triangleright Y}(\gamma) = w^{ab}-\gamma(Y) \cdot |D(Y)| = w^{ab}-\gamma(Y) \cdot (|D(Y)| - 1) + w^{ab}-\gamma(Y) \). the maximum penalties of variable \( Y \) is \( w^{ab}(Y) \cdot (|D(Y)| - 1) \) (resp. \( w^{ab}(Y) \cdot (|D(Y)| - 1) \) and \( w^{ab}-\gamma(Y) \cdot (|D(Y)| - 1) \)). Consequently, the value \( v(\gamma) \) (resp. \( v(\gamma) \) and \( v^{X \triangleright Y}(\gamma) \)) of each arc \( \gamma = \langle X, Y \rangle \in \text{cp} \cup \text{sci} \) (resp. \( \gamma = \langle X, Y \rangle \in \text{ci} \) and \( \gamma = \langle X, Y \rangle \in \text{ci} \)) is guaranteed to be larger than the maximum penalties of variable \( Y \) (resp. the maximum penalties of variable \( Y \) and the sum of penalties of \( Y \)'s dependants). As the weight of a variable is recursively defined according to Equation (4.3), the weight of variable \( X \) is guaranteed to be larger than (i) the sum of the maximum penalties of all \( X \)'s dependants; (ii) the maximum penalties of any less important variable \( Y \); and (iii) the sum of the maximum penalties of any less important variable \( Y \)'s dependants. Consequently, \( \text{pen}(\sigma_p) > \text{pen}(\sigma_{p+1}) \). Following
from the transitivity: \( \text{pen}(\sigma_1(=o')) > \text{pen}(\sigma_2) > \cdots > \text{pen}(\sigma_{t-1}) > \text{pen}(\sigma_t(=o)) \)
and thus \( \text{pen}(o') > \text{pen}(o) \). \( \square \)

**Corollary 4.1** Given the penalty scoring function \( \text{pen} \) of a conditionally acyclic TCP-net \( \mathcal{N} \):

\[ \forall o, o' \in O, \text{ if } \text{pen}(o') > \text{pen}(o) \text{ then } \mathcal{N} \models o > o' \text{ or } \mathcal{N} \models o \preceq o' \; \]

\[ \forall o, o' \in O, \text{ if } \text{pen}(o') = \text{pen}(o) \text{ then } \mathcal{N} \models o \preceq o' \]

### 4.3 The collective penalty scoring function

In this section, after the individual penalty scores are computed independently, these penalty scores are aggregated into a normalized collective penalty scoring function that best conveys the preferences of the group of agents. The method developed here is based on the work of Lafage and Lang [88], which builds a collective disutility function for group decision-making using logical preference representations.

**Definition 4.5 (Collective penalty scoring function)** Given a set of acyclic CP-nets or conditionally acyclic TCP-nets \( \mathcal{N} = \{N_1, \ldots, N_n\} \), the collective penalty scoring function \( P \) mapping from \( O \) to \([0, +\infty]\) is defined by:

\[ \forall o \in O, \; P(o) = \Diamond \{ \text{pen}_i(o) \mid i = 1, \ldots, n \} \tag{4.5} \]

where \( \Diamond \) is a function that satisfies non-decreasingness for each of its argument and commutativity.

As discussed in [88], the most natural choices for \( \Diamond \) are \( \text{sum} \) and \( \text{max} \). \( \text{sum} \) is a utilitarian aggregation operator, stating that the collective penalty score of an outcome is the sum of the penalty scores of all the agents in the group. On the other hand, \( \text{max} \) states that the maximum penalty score among all the agents should be considered. Thus, the \( \text{max} \) aggregation operator corresponds to the egalitarian social welfare.

**Example.** Consider the three agents’ TCP-nets in Figure 4.3. For the outcome \( o = abcd \), \( \text{pen}_1(o) = \frac{2}{22} \); \( \text{pen}_2(o) = \frac{4}{7} \) and \( \text{pen}_3(o) = \frac{6}{17} \). Thus, when \( \Diamond = \text{sum} \), \( P(o) = \Diamond \{ \frac{2}{22}, \frac{4}{7}, \frac{6}{17} \} = \frac{179}{114} \); or when \( \Diamond = \text{max} \), \( P(o) = \Diamond \{ \frac{2}{22}, \frac{4}{7}, \frac{6}{17} \} = \frac{4}{7} \).
Chapter 4. Penalty score-based preference aggregation with CP-nets and TCP-nets

Definition 4.6 (◊-optimality) An outcome $o^*$ is ◊-optimal iff $\forall o' \in O$ and $o \neq o'$, $P(o^*) \leq P(o')$.

It is important to know the relation between ◊-optimal and Pareto-optimal.

Theorem 4.2 A ◊-optimal outcome $o^*$ is also Pareto-optimal.
PROOF. Assume that an $\Diamond$-optimal outcome $o^*$ is not Pareto-optimal, then there must exist another outcome $o'$ such that $o'$ Pareto-dominates $o$. As CP-nets or TCP-nets do not model indifference, if $o'$ Pareto-dominates $o^*$ then for each $\mathcal{N}_i \in \mathbb{N}$: $\mathcal{N}_i | o' > o^*$ and $\text{pen}_i (o^*) > \text{pen}_i (o')$ according to Theorem 3.1 and Theorem 4.1. As $\Diamond$ satisfies non-decreasingness for each of its argument, $P(o^*) > P(o')$, contradicting the fact that $o^*$ is $\Diamond$-optimal. □

Nonetheless, a Pareto-optimal outcome is not necessarily $\Diamond$-optimal. For instance, let $\Diamond = \max$ and assume there are only two possible outcomes $o_1, o_2$ and two agents $A_1$ and $A_2$, for $A_1$: $\text{pen}_1 (o_1) = \frac{1}{2}$ and $\text{pen}_1 (o_2) = 0$; and for $A_2$: $\text{pen}_2 (o_1) = 0$ and $\text{pen}_2 (o_2) = \frac{1}{3}$. Hence, $P(o_1) = \diamond \{\frac{1}{2}, 0\} = \frac{1}{2}$ and $P(o_2) = \diamond \{0, \frac{1}{3}\} = \frac{1}{3}$. In such case both $o_1$ and $o_2$ are Pareto-optimal, however, only $o_2$ is $\Diamond$-optimal according to $\max$.

4.4 The proposed penalty score-based heuristic for group decision-making

Using CP-nets or TCP-nets representation, the outcome space is exponentially large in the number of domain variables, and thus it is impractical to search through the entire outcome space for an optimal collective decision. In this section, we present an efficient heuristic algorithm for group decision-making with acyclic CP-nets or TCP-nets, called HeTCP, which guarantees optimality for the given aggregation rule.

We conceptualize the assignment of the variable values as a tree, known as the search tree. For a group decision-making problem with $n$ agents and a set of $m$ variables $V$, let $k$ be the maximum size of the variable domain: $\forall X \in V, |D(X)| \leq k$, the search tree is then a $k$-ary tree, with depth $m+1$ and the root being at level 0. Assume that the variables are assigned values level by level according to a random order $\sigma = X_{\sigma_1} > \cdots > X_{\sigma_m}$, i.e., level $\ell$ assigns the values to the variable $X_{\sigma_\ell}$. The root node represents an empty assignment and each level represents the assignment to the next variable $X$ in the chosen order. Note that HeTCP does not depend on the order of variables being considered along the path. If a node from the upper level is being expanded with the value of a variable $X$ at the current level, it has $|D(X)|$ branches and each branch assigns a different value to $X$. Each leaf node at level $m$ represents an alternative outcome and the path to reach that leaf node from the root specifies the complete assignment to the set of domain variables according to that outcome. Similar
to other heuristic search algorithms [115], HeTCP creates a search tree by iteratively selecting a node that appears to be most likely to lead towards the optimal collective decision, i.e. the outcome with the minimum collective penalty score.

**Definition 4.7 (Best possible alternative)** At each node $\eta$ of the search tree, each agent $i$ has a best possible alternative on that node, denoted by $\text{BPA}_i(\eta)$, which is the optimistic outcome that agent $i$ can obtain with the values assigned to the variables along the path from the root node to the current node being fixed.

$$\text{BPA}_i(\eta) = \text{OPT} - \text{ALTERNATIVE}(\text{assg}, N_i)$$

where $\text{assg} = \text{PATH} - \text{ASSIGNMENT}(\eta)$, is the path assignment from the root node to $\eta$. The best possible alternative of the root node for an agent $i$ corresponds to the optimal outcome for agent $i$ in the entire outcome space, i.e. all variables are assigned the preferred values according to $N_i$.

Computing the best possible alternative of a node for an agent is similar to the individual outcome optimization problem. We simply need to sweep through the preference network from top to bottom, assigning the most preferred value to each remaining variable $X$ (i.e., the variable that has not been assigned a value along the path) with respect to the parent context. Note that for acyclic CP-net or conditionally acyclic TCP-net, the best possible alternative of a node for an agent is unique, that means, it is better than any other completions of the path assignment of the node for that agent. Intuitively, the best possible alternative of an agent $i$ at a node $\eta$, has the minimum penalty score among the completions of the path assignment of $\eta$ with respect to $N_i$:

$$\text{BPA}_i(\eta) = \arg\min_{o \in \text{Comp}(\text{assg})} \text{pen}_i(o), \text{ where } \text{assg} = \text{PATH} - \text{ASSIGNMENT}(\eta).$$

**Example.** Consider the TCP-nets in Figure 4.3 and the node $\eta_1$ with the path assignment $a$. The best possible alternative for agent 1: $\text{BPA}_1(\eta_1) = abcd$ (see Figure 4.3(a)); agent 2: $\text{BPA}_2(\eta_1) = abc$ (see Figure 4.3(b)); and agent 3: $\text{BPA}_3(\eta_1) = ab\bar{c}d$ (see Figure 4.3(c)).

Now we define an evaluation function $f$ for the heuristic search strategy as follows:

**Definition 4.8 (Collective evaluation function)** The collective evaluation function $f$,
4.4. The proposed penalty score-based heuristic for group decision-making

Input: \( N \), a set of acyclic TCP-nets \( \{ N_1, \ldots, N_n \} \)
Output: \( \text{assg} \), a complete assignment

1. Randomly choose an order over variables \( \sigma = X_{\sigma_1} > \cdots > X_{\sigma_m} \);
2. \( \text{fringe} \leftarrow \text{INSERT} (\text{MAKE-NODE} (\text{True}), \text{fringe}) \);
3. \( \text{while} \ \text{fringe} \neq \emptyset \ \text{do} \)
   4. \( \eta \leftarrow \text{REMOVE-FIRST} (\text{fringe}) \);
   5. \( \text{assg} \leftarrow \text{PATH-ASSIGNMENT} (\eta) \);
   6. \( \text{if} \ \text{LEAF-NODE} (\eta) \ \text{then return} \ \text{assg} \);
   7. \( \text{else} \)
   8. \( X \leftarrow \text{NEXT-VARIABLE} (\sigma, \text{assg}) \);
   9. \( \text{for each} \ x \in D (X) \ \text{do} \)
      10. \( \text{INSERT} (\text{MAKE-NODE} (\text{assg} \land x), \text{fringe}) \);
   11. \( \text{end} \)
   12. \( \text{SORT-FRINGE-ASC} (\text{fringe}) \);
   13. \( \text{end} \)
4. \( \text{end} \)

Algorithm 4.3: \( \text{HeTCP}(N) \)

mapping from a node \( \eta \) to \( [0, +\infty] \), is defined by:

\[
f(\eta) = \diamond \{ \text{pen}_i (\text{BPA}_i (\eta)) \mid i \in \{1, \ldots, n\}\}
\]

where \( \diamond \) is an aggregation operator, e.g. \( \text{sum}, \text{max} \).

Starting with the root node with an empty assignment (logically expressed by \text{True}) (line 2), \( \text{HeTCP} \) maintains a priority queue of nodes to be expanded, known as the \( \text{fringe} \). The lower \( f(\eta) \) for a node \( \eta \), the higher its priority is. At each iteration of \( \text{HeTCP} \), the first node \( \eta \), i.e. the node with the lowest \( f \) value, is removed from the \( \text{fringe} \) and being expanded (line 4). Let \( \text{assg} \) be the path assignment of the current node \( \eta \) (line 5) and \( X \) the variable that is being considered at the current iteration (line 8), for each \( x \in D (X) \), a children node expanded from \( \eta \) with the path assignment \( \text{assg} \land x \) will be created and added into the \( \text{fringe} \) (line 8–11). The \( f \) value of these children nodes are computed accordingly, and the \( \text{fringe} \) are sorted in ascending order of the \( f \) values of the nodes in the \( \text{fringe} \) (line 12). \( \text{HeTCP} \) continues until the current chosen node for expansion is a leaf node, i.e. the path assignment of the node forms a complete assignment to the set of domain variables \( V \), it then returns the path assignment of the current node as the final collective decision (line 6). Such a search tree is shown in Figure 4.4(a) for the group decision-making problem from the set of 3 TCP-nets in Figure 4.3.
Lemma 4.1 The heuristic evaluation function $f$ never overestimates the collective penalty scores of the leaf nodes.

PROOF. Travel along the path of the tree, both the individual and the collective penalty score is non-decreasing. At each node of the search tree, the heuristic evaluation function $f$ is aggregated from a set of optimistic (i.e. the possible minimum) penalty scores of the agents. Thus, it never overestimates the collective penalty scores of the leaf nodes. □

Theorem 4.3 The outcome returned by HeTCP algorithm is $\Diamond$-optimal.

PROOF. HeTCP traverses the tree searching all neighbours; it follows lowest evaluated value path and keeps a sorted priority queue of alternate path segments along the way. If at any point the path being followed has a higher evaluated value than other encountered path segments, the higher evaluated value path is kept in the fringe and the process is continued at the lower value sub-path. This continues until the current node chosen for expansion is a leaf node. As we already proved that the evaluation function $f$ never overestimates the collective penalty scores of the leaf nodes (see Lemma 4.1), the first leaf node chosen for expansion is thus guaranteed to be optimal. □

An illustration

Now, we demonstrate the execution of HeTCP approach with our running example in Figure 4.3. As all variables are binary, the search tree is a binary tree. Let $q_X^j$ denotes the additional weight of variable $X$ assigned by agent $j$. We consider a simple setting in which $\forall j \in \{1, \ldots, 3\}, X \in V. q_X^j = 1$. The dependency graph and importance weights of the variables are shown on the right of each TCP-net in Figure 4.3. We construct the search tree in the order of $ABCD$ and consider the max aggregation rule. The search tree is depicted in Figure 4.4(a) associated with an illustration of the ongoing changes occurs in the fringe (see Figure 4.4(b)).

Start from the root node $\eta_0$: $f(\eta_0) = \Diamond \{0, 0, 0\} = 0$, the $1^{st}$ iteration removes $\eta_0$ from the fringe and expands it with the path assignments $a$ and $\bar{a}$ to the first variable $A$ respectively. Two children nodes $\eta_1$ and $\eta_2$ are created and added to the
4.4. The proposed penalty score-based heuristic for group decision-making

(a) The search tree

(b) The fringe

Figure 4.4: An example of HeTCP procedure

fringe respectively. With the path assignment $a$ assigned to variable $A$: for agent 1, $BPA_1(a) = abcd$ and $pen_1(abcd) = 0$; for agent 2, $BPA_2(a) = ab$ and $pen_2(abcd) = \frac{4}{7}$; and for agent 3, $BPA_3(a) = abcd$ and $pen_3(abcd) = \frac{3}{12}$. Consequently, $f(\eta_1) = \max\{0, \frac{4}{7}, \frac{3}{12}\} = \frac{4}{7}$. Similarly, $f(\eta_2) = \max\{\frac{11}{22}, 0, 0\} = \frac{11}{22}$. Hence, the fringe is then sorted as $\eta_2 \eta_1$ in ascending order of the $f$ values. At the 2nd iteration, we remove the first node $\eta_2$ from the fringe to expand and create two children nodes $\eta_3$ and $\eta_4$. We add each child of $\eta_2$ into the fringe and sort the fringe
as $\eta_1, \eta_3, \eta_4$ in ascending order of the $f$ values ($f(\eta_1) = \frac{4}{7}$, $f(\eta_3) = \max\{\frac{11}{12}, 0, \frac{9}{12}\} = \frac{9}{12}$ and $f(\eta_4) = \max\{\frac{18}{22}, 0, \frac{1}{7}\} = \frac{18}{22}$). Now the node has minimum $f$ value appears to be $\eta_1$ and we need to go back track to expand $\eta_1$ in the next iteration. The algorithm then continues until we finally expand $\eta_{10}$ (the child of $\eta_7$) as $\eta_{10}$ is a leaf node, HeTCP then returns the path assignment $abcd$ of $\eta_{10}$ as the final collective decision.

### 4.5 Experiments

We have tested HeTCP on a large number of scenarios varying the agents’ preferences and the number of variables. The objectives of the computer experiments are to study experimentally the average running time and average visited nodes of Algorithm HeTCP for computing the optimal collective decision of a given decision rule. All table and graphic output is included in Section 4.7 and Section 4.8 for ease of reading. For the random CP-net generator, the reader is referred to the experiment section 3.4 in Chapter 3. We hereby introduce a random TCP-net generator.

#### Random TCP-net generator

The random TCP-net generator generates a random TCP-net. Two sets of data have to be generated for an agent’s TCP-net, the preference network topology (i.e., dependency graph) and the conditional preference of each variables.

#### Network topology

In this chapter, we restrict the preference network $N$ to be conditionally acyclic, i.e., for every possible assignment $w$ to $S(N)$, the resulting $w$-directed graph of the dependency graph is acyclic. To generate the network topology, the required input from the users is:

- the number of variables $m$; and
- the maximum node in-degree $d$.

A strict topological ordering $O = X_{O_1} > \cdots > X_{O_m}$ over the set of domain variables $V = \{X_1, \ldots, X_m\}$ is randomly generated. The first variable $X_{O_1}$ is considered to
be the root of the tree, i.e., $Pa(X_{O_1}) = \emptyset$. Then following the topological ordering $O$, for every variable $X_{O_i}$ ($i > 1$), we first choose the parent variable for the current considered variable $X_{O_i}$. Generate a random integer number $pa$ such that $0 \leq k \leq \max\{i - 1, d\}$ ($d$ is the maximum node in-degree). If $pa = 0$, then $Pa(X_{O_i}) = \emptyset$; Otherwise, randomly pick up a set of $k$ variables from $\{X_{O_1}, \ldots, X_{O_{i-1}}\}$, denoted by $U = \{X_{p_1}, \ldots, X_{p_k}\}$, as the parent variables of $X_{O_i}$.

We then choose the variables that are considered to be unconditionally more important than the current considered variable. Generate a random integer number $ui$ such that $0 \leq ui \leq \max\{i - 1, d - pa\}$ ($d$ is the maximum node in-degree and $pa$ is the number of parent variables of this variable). If $ui \leq 0$, then there is no variable unconditionally more important than the current considered variable $X_{O_i}$; Otherwise, randomly pick up a set of $ui$ variables from $\{X_{O_1}, \ldots, X_{O_{i-1}}\}$, denoted by $UI = \{X_{p_1}, \ldots, X_{p_{ui}}\}$, as the number of variables that are unconditionally more important than the current variable. Note that UI must also satisfies that none of the variable in $UI$ is an ancestor of $X_{O_i}$, in order to avoid redundant arcs in the TCP-net.

Finally, we choose the variable that have conditional relative importance relation with $X_{O_i}$. Generate a random integer number $ci$ such that $0 \leq ci \leq \max\{i - 1, d - pa - ui\}$ ($d$ is the maximum node in-degree, $pa$ is the number of parent variables of this variable, and $ui$ is the number of variables that are unconditionally more important than $X_{O_i}$). If $ci \leq 0$, then there is no variable have conditionally relative importance relation with $X_{O_i}$; Otherwise, randomly pick up a set of $ci$ variables from $\{X_{O_1}, \ldots, X_{O_{i-1}}\}$, denoted by $CI = \{X_{p_1}, \ldots, X_{p_{ci}}\}$, as the number of variables that have conditional relative importance relation with $X_{O_i}$. Note that $CI$ must also satisfies that none of the variable in $CI$ is an ancestor of $X_{O_i}$ or is more important than $X_{O_i}$. For each variable $X_{O_k} \in CI$, choose a set of selectors $S$ from $\{X_1, \ldots, X_{k-1}\}$ for the relative importance relation between $X_{O_k}$ and $X_{O_i}$. Note that $S$ must also satisfies that none of the variable in $S$ has any conditional relative importance relation with $X_{O_k}$ and $X_{O_i}$.

### Conditional preference tables

To generate the conditional preference tables (CPTs), for each variable $X \in V$, let $D(X)$ be the domain size of $X$, $Pa(X) = \{X_{p_1}, \ldots, X_{p_{pa}}\}$ be the set of parent variables of $X$. If $X$ is a root variable, i.e., $Pa(X) = \emptyset$, then randomly generate a preference ordering over the domain values ($\{0, \ldots, D(X)\}$) of $X$. Otherwise, there will be $|D(X_{p_1})| \times \cdots \times |D(X_{p_{pa}})|$ rows in the CPT of $X$. Randomly gener-
ate $|D(X_{p_1})| \times \cdots \times |D(X_{p_m})|$ preference orderings over the domain values of $X$ such that the set of preference orderings satisfies the following condition. For every parent variable $X_{p_k}$, for at least one possible assignment $u$ to the other parent variables $u \in D(X_{p_1}) \times \cdots \times D(X_{p_{k-1}}) \times D(X_{p_{k+1}}) \times \cdots \times D(X_{p_m})$, at least two different values of $X_{p_k}$ will result in two distinct preference orderings over the domain value of $X$.

**Conditional importance tables**

To generate the conditional importance tables (CPTs), for each conditional importance relation $(X; Y)$, let $S$ be the selector set of the importance relation between $X$ and $Y$, then there will be $|D(X_{p_1})| \times \cdots \times |D(X_{p_{|S|}})|$ rows in the CIT of the relation $(X; Y)$. Randomly generate $|D(X_{p_m})| \times \cdots \times |D(X_{p_{|S|}})|$ importance orderings over $X$ and $Y$ such that the set of importance relations satisfies the following condition. For every variable $X_{p_k}$ in the selector set $S$, for at least one possible assignment $s$ to the other parent variables $s \in D(X_{p_1}) \times \cdots \times D(X_{p_{k-1}}) \times D(X_{p_{k+1}}) \times \cdots \times D(X_{p_{|S|}})$, at least two different values of $X_{p_k}$ will result in two distinct importance orderings between $X$ and $Y$.

**Experiment design**

In these experiments, we tested the performance of the proposed algorithm in both binary-valued and multi-valued TCP-nets. The numbers of agents are 5 and 15. For binary TCP-nets, the domain size is 2 for each of the variable; for multi-valued TCP-nets, we randomly generate a set of $m$ integer ranged between 2 to 5, each of which is a domain size for a variable in $V$. For each number of agents and each number of variables, we generate 1,000 random examples of the agents’ TCP-nets. Specifically, we consider $\Diamond = \max$ as the aggregation operator, i.e., the optimal social outcome in this context is an egalitarian solution.

We compare the performance of the proposed algorithm to two other algorithms, namely Brute-force algorithm and Graph-Search algorithm, for computing the optimal social outcome. Brute-force algorithm enumerates the entire outcome space, computes the collective score of each outcome and searches for an optimal collective decision. Graph-Search algorithm (see Purrington and Durfee [121]), traverses the space of outcome classes, starting from the most satisfying, in order to find an optimal social outcome. In Graph-Search algorithm, each agent maintains a set of accepted
candidates and a list of remaining candidates that grouped into equivalence classes. At each step of the algorithm, the agents that are most satisfied with their best remaining outcomes expand their accepted candidate sets to include that class of outcomes. The algorithm terminates when the intersection between the sets of accepted candidate outcomes is non-empty.

**Experiment Results**

**Test Results for binary-valued TCP-nets**

The log-scale plot in Figure 4.5 shows the average execute time of the Brute-force algorithm, Graph-Search algorithm and the proposed HeTCP algorithm in the case of 5 agents and 15 agents with binary-valued TCP-nets. It demonstrate that the Brute-force algorithm and Graph-Search algorithm perform in a similar fashion. The average execution time of the Brute-force and Graph-Search algorithms increase exponentially as the number of variables increases. However, HeTCP is much more efficient than both Brute-force and Graph-Search algorithms. In general, for large numbers of variables, it offers several orders of magnitude improvement in performance over the Brute-force and Graph-Search algorithms. For instance, when there are 13 variables (resp. 11 variables), the average execution time of HeTCP is reduced by more than two orders of magnitude as compared to the Brute-force and Graph-Search algorithms in 5 agents (resp. 15 agents) case.

We further test 100 cases for 12 variables and 15 agents with binary-valued TCP-nets, according to the experiment data, the execution time of the Brute-force algorithm is on average more than 400 seconds. On the other hand, on average, the proposed HeTCP approach can produce the optimal outcome in less than 5 seconds. According to the experiment data, even with 25 variables, the proposed HeTCP approach can produce the optimal outcome in less than 3 seconds (resp. 100 seconds) when there are 5 agents (resp. 15 agents). Table 4.1 further shows the average number of visited nodes of HeTCP approach in the experiments with binary TCP-nets. The average number of visited nodes of HeTCP approach is increasing as the number of variables increases. However, compared with the exponentially increasing outcome space, HeTCP approach significantly reduces the search effort. When the number of variable is large.
Chapter 4. Penalty score-based preference aggregation with CP-nets and TCP-nets

(e.g., 25), on average, the search effort is reduced by more than 5 orders of magnitude in 5 agents and 4 orders of magnitude in the case of 15 agents.

Test Results for multi-valued TCP-nets

The test results regarding the average execution time with multi-valued TCP-nets networks with 5 and 15 agents are shown in Figure 4.6(a) and 4.6(b) respectively. Similar to the results with binary TCP-nets, the average execution time of the Brute-force and Graph-Search algorithms increase exponentially as the number of variables increases. HeTCP algorithm is much more efficient than both Brute-force and Graph-Search algorithms. When there are 7 variables, the average execution time of HeTCP is reduced by more than two orders of magnitude (resp. an order of magnitude) as compared with the Brute-force and Graph-Search algorithms in the case of 5 agents (resp. 15 agents). According to the experiment data, even with 25 variables and 5 agents (resp. 20 variables and 15 agents), the proposed HeTCP approach can produce the optimal outcome in about 100 seconds (less than 200 seconds).

Table 4.2 further shows the average outcome space and the average number of visited nodes of HeTCP approach in the experiments with multi-valued TCP-nets. The average generated outcome space is increasing exponentially as the number of variables increases. However, along with the exponentially increasing outcome space, HeTCP approach significantly reduce the search effort. When the number of variable is large (e.g., 20), on average, the search effort is reduced by more than 8 orders of magnitude in the case of 5 agents and 7 orders of magnitude in the case of 15 agents.

4.6 Summary

In Chapter 4, the problem of group decision-making with CP-nets and TCP-nets was studied. The contribution of this work is twofold. Firstly, based on the previous work of the penalty scoring function for acyclic CP-nets introduced in Chapter 3, Chapter 4 has gone one step further by incorporating the relative importance relation among pairs of variables. It introduced an individual penalty scoring function as a numerical approximation, not only for acyclic CP-nets, but also for conditionally acyclic TCP-nets. We have also shown that both the function construction and the penalty score computation are computationally efficient. To the best of our knowledge, this work provides
the first numerical approximation for TCP-nets. Secondly, given the exponentially increasing outcome space in combinatorial domains, the problem of collective decision-making problem with CP-nets and TCP-nets has also been addressed in this chapter. A collective penalty scoring function was introduced, which forms the basis of group decision-making. Then, a heuristic algorithm $\text{HeTCP}$ was presented to search for an optimal collective decision in the large outcome space. Experimental results have also been presented to demonstrate the computational efficiency of the proposed approach.
### 4.7 Table Output

Table 4.1: Outcome space and average number of visited Nodes with binary TCP-nets

<table>
<thead>
<tr>
<th>Variable</th>
<th>Outcome Space</th>
<th>Agents</th>
<th>Variable</th>
<th>Outcome Space</th>
<th>Agents</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>4.398</td>
<td>15</td>
<td>2.15 × 10⁹</td>
<td>86.6</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>5.727</td>
<td>7.721</td>
<td>4.29 × 10⁹</td>
<td>93.45</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>7.026</td>
<td>10.55</td>
<td>8.59 × 10⁹</td>
<td>97.7</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>8.378</td>
<td>14.05</td>
<td>1.72 × 10¹⁰</td>
<td>105.</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>9.632</td>
<td>18.32</td>
<td>3.44 × 10¹⁰</td>
<td>104.45</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>10.85</td>
<td>22.5</td>
<td>6.87 × 10¹⁰</td>
<td>111.13</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>12.27</td>
<td>26.87</td>
<td>1.37 × 10¹¹</td>
<td>130.4</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>13.66</td>
<td>33.46</td>
<td>2.75 × 10¹¹</td>
<td>183.8</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>15.07</td>
<td>37.98</td>
<td>5.5 × 10¹¹</td>
<td>156.8</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>16.76</td>
<td>43.07</td>
<td>1.1 × 10¹²</td>
<td>174.1</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>18.85</td>
<td>52.59</td>
<td>2.2 × 10¹²</td>
<td>200.6</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>20.51</td>
<td>57.1</td>
<td>4.4 × 10¹²</td>
<td>224.3</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>22.03</td>
<td>62.58</td>
<td>8.8 × 10¹²</td>
<td>269.6</td>
</tr>
<tr>
<td>15</td>
<td>32768</td>
<td>24.09</td>
<td>78.29</td>
<td>1.76 × 10¹³</td>
<td>230.9</td>
</tr>
<tr>
<td>16</td>
<td>65536</td>
<td>25.96</td>
<td>96.48</td>
<td>3.52 × 10¹³</td>
<td>318.</td>
</tr>
<tr>
<td>17</td>
<td>131072</td>
<td>29.08</td>
<td>117.9</td>
<td>7.04 × 10¹³</td>
<td>296.4</td>
</tr>
<tr>
<td>18</td>
<td>262144</td>
<td>30.43</td>
<td>120.3</td>
<td>1.41 × 10¹⁴</td>
<td>288.2</td>
</tr>
<tr>
<td>19</td>
<td>524288</td>
<td>34.65</td>
<td>175.7</td>
<td>2.81 × 10¹⁴</td>
<td>324.6</td>
</tr>
<tr>
<td>20</td>
<td>1.05 × 10⁹</td>
<td>36.84</td>
<td>199.7</td>
<td>5.63 × 10¹⁴</td>
<td>358.5</td>
</tr>
<tr>
<td>21</td>
<td>2.1 × 10⁶</td>
<td>36.4</td>
<td>209.6</td>
<td>1.13 × 10¹⁵</td>
<td>416.5</td>
</tr>
<tr>
<td>22</td>
<td>4.19 × 10⁹</td>
<td>42.09</td>
<td>248.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>8.39 × 10⁹</td>
<td>42.97</td>
<td>314.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>1.68 × 10⁷</td>
<td>54.15</td>
<td>396.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>3.36 × 10⁷</td>
<td>57.41</td>
<td>420.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>6.71 × 10⁷</td>
<td>52.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>1.34 × 10⁸</td>
<td>62.6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>2.68 × 10⁸</td>
<td>70.71</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>5.37 × 10⁸</td>
<td>85.65</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>1.07 × 10⁹</td>
<td>84.39</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4.2: Average outcome space and average number of visited Nodes with multi-valued TCP-nets

<table>
<thead>
<tr>
<th>Variable</th>
<th>Agents</th>
<th>5 agents</th>
<th>15 agents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Outcome Space</td>
<td>Visited Nodes</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>12.5926</td>
<td>5.7407</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>44.75</td>
<td>8.3796</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>156.509</td>
<td>9.9537</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>462.287</td>
<td>10.778</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>1839.14</td>
<td>10.759</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>6358.15</td>
<td>14.037</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>19530.</td>
<td>15.481</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>72157.7</td>
<td>17.463</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>33559.3</td>
<td>21.12</td>
</tr>
<tr>
<td>1.</td>
<td></td>
<td>1. \times 10^9</td>
<td>27.8</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>3.42 \times 10^6</td>
<td>26.4</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>1.07 \times 10^7</td>
<td>30.43</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>4.36 \times 10^7</td>
<td>41.69</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>1.39 \times 10^8</td>
<td>66.51</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>5.65 \times 10^8</td>
<td>82.4</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>2.16 \times 10^9</td>
<td>76.33</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>6.1 \times 10^9</td>
<td>75.18</td>
</tr>
<tr>
<td>19</td>
<td></td>
<td>1.92 \times 10^{10}</td>
<td>189.1</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>8.3 \times 10^{10}</td>
<td>251.3</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>2.5 \times 10^{11}</td>
<td>342.18</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>8.14 \times 10^{11}</td>
<td>456.43</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>2.64 \times 10^{12}</td>
<td>572.13</td>
</tr>
<tr>
<td>24</td>
<td></td>
<td>2.21 \times 10^{13}</td>
<td>1013.1</td>
</tr>
<tr>
<td>25</td>
<td></td>
<td>5.28 \times 10^{13}</td>
<td>1060.</td>
</tr>
</tbody>
</table>
Chapter 4. Penalty score-based preference aggregation with CP-nets and TCP-nets

4.8 Graphic Output

Figure 4.5: Average execution time comparison with binary-valued CP-nets

(a) 5 agents; 2-25 variables; binary TCP-nets (log-scale plot)

(b) 15 agents; 2-25 variables; binary TCP-nets (log-scale plot)
4.8. Graphic Output

Figure 4.6: Average execution time comparison with multi-valued CP-nets

(a) 5 agents; 2-20 variables; multi-valued TCP-nets (log-scale plot)

(b) 15 agents; 2-20 variables; multi-valued TCP-nets (log-scale plot)
Chapter 5

Majority-rule-based preference aggregation with CP-nets

In Chapter 5, the problem of collective decision-making with CP-nets in another problem setting is studied. A scenario of aggregating preferences based on majority rule is considered in this chapter. Due to the exponential outcome space in combinatorial domains, it is impractical to reason about the individual full rankings over the alternative space and apply majority rule directly. Most existing work either does not consider computational requirements, or depends on a strong assumption that the agents have acyclic CP-nets that are compatible with a common order on the variables. To this end, in Chapter 5 we introduce an efficient CSP-based approach, called \textit{MajCP} (Majority-rule-based collective decision-making with CP-nets), to compute the possible majority winning alternatives giving a set of (possibly cyclic) CP-nets.

5.1 Introduction

In this chapter, we consider the majority-rule-based preference aggregation problem in combinatorial domains, in which the agents’ preferences are represented by (possibly cyclic) CP-nets. In classical voting theory, voters are supposed to submit their preferences as linear orders over the set of all possible alternatives. Subsequently, a voting rule is applied to select one winning alternative or a winning set of alternatives.
5.1. Introduction

However, in combinatorial domains, the number of possible alternatives is exponential in the number of variables, and the attributes are inter-dependent. Thus, it is impractical to reason about the individual full rankings over the alternative space and apply majority rule directly. Moreover, as discussed in Chapters 2 and 3, the complexity of deciding whether one outcome dominates another (dominance testing) for an arbitrary CP-net is NP-hard. Furthermore, computing the majority winning alternatives with multiple agents’ CP-nets may require dominance testing on each pair of alternatives on each individual CP-net. For example, having ten binary variables, each involved agent would need to compare \( \binom{2^{10}}{2} = 523776 \) pairs of alternatives. This problem is likely to be even more difficult than NP or coNP problems. Last but not least, on the positive side, preferences expressed as CP-nets are easy to elicit, provided that the relation graph is known and possesses a sufficiently small node in-degree. However, it must also be admitted that CP-nets are not fully expressive. Some preference relations are not expressible by CP-nets [41, 51] and the preference induced by a CP-net is incomplete in the general case [24]. A large body of work has studied the problems of voting and winner determination with incomplete preference. In particular, a situation in which the voter’s preference orderings over the alternative space are not fully known when the voting rule applies is considered, e.g., see [84, 94, 118, 147]. Similarly, by using CP-nets, preferences have been only partially elicited in general. In some cases we may conclude that the elicited preferences known so far, although incomplete, are sufficiently informative that a necessary winner can be determined. If this is not the case, an interesting problem may be computing a set of candidates who might possibly win the vote after the preferences have become fully known, thus giving the agents an opportunity to focus on these possibly winning candidates (entitled ‘possible winners’) and forget about the others.

In this chapter, given that the individual preferences have been elicited and represented as CP-nets, the problem of majority-rule-based preference aggregation will be addressed. Computing aggregation rules from a collection of CP-nets has been studied in the literature, e.g.,[95, 125]. In particular, Lang and Xia [95] consider decomposition with voting rules. They assume that the agents’ preferences can be represented with acyclic CP-nets being compatible with a common order on the variables. However, such an assumption is unlikely to be applicable in most real world situations [148]. Xia et al. [150] partially address this shortcoming by introducing an order-independent sequential composition of voting rules. In their framework, the profile is still required to be compatible with some order on the variables, but this order is not specified in the definition of the rule. Nevertheless, the domain restriction by this order-independent
sequential composition of voting rules is still severe; there must be some (unspecified) directed acyclic graph with which the profile is compatible. Xia et al. [148] generalise the earlier, more restrictive method, proposing an aggregation methodology that does not require any relationship among the agents’ CP-net structures. Their approach focuses on a set of locally winning alternatives, called local Condorcet winners. A local Condorcet winner is an alternative that majority-dominates any other alternatives which differ in one single variable with that alternative. However, the performance of their algorithm for computing the local Condorcet winners also depends on the consistency among the structures of the agents’ CP-nets. Moreover, we will show later in this chapter that a local Condorcet winner is in essence, not guaranteed to be a possible winner for the given collection of CP-nets.

To this end, this chapter addresses the above drawbacks, proposing an efficient CSP-based approach, called MajCP (Majority-rule-based collective decision-making with CP-nets), to compute the set of possible winners (the candidates who may become a Condorcet winner or weak Condorcet winners when the total preferences of the agents can be provided). With multiple agents’ CP-nets as input, it first reduces the problem to a CSP (Constraint Satisfaction Problem) in multi-valued CP-nets or SAT (Boolean Satisfiability Problem) in binary CP-nets. As such, we first obtain a set of weak Local Condorcet Winners (wLCWs) 1, by computing the models of the corresponding CSP or SAT. We subsequently prove that the set of possible winners is a subset of wLCWs. Consequently, the set of possible winners could be obtained from the set of wLCWs, by filtering out those majority-dominated wLCWs.

The proposed approach allows the agents to have different preferential independence structures, and enables us to aggregate preferences even when the agents’ CP-nets are cyclic. Moreover, the proposed approach reduces the search space and is computationally efficient. Theoretical results are provided, regarding the complexity of the transformation from the original problem to the corresponding CSP (or SAT), and the complexity of computing the set of wLCWs. Experimental results are also presented in this chapter, in order to provide an illustration of the computational efficiency of the proposed approach in practice.

This chapter is organised as follows. Section 5.2 provides some background information about majority-rule-based preference aggregation. Then, in Section 5.3 we study a local composition of majority rule and analyse its incompatibility with the original

1The notion of weak Local Condorcet winner is defined such that the framework is general enough to cope also with the case when there are even number of agents.
majority preferences by using several counter-examples. Subsequently, Section 5.4 proceeds to present the approach that is proposed to compute the possible winners, and Section 5.6 discusses the experimental results. Finally, Section 5.7 summarises the chapter.

5.2 Preliminaries

Let \( V = \{X_1, \ldots, X_m\} \) be a set of \( m \) domain variables, a preference order \( \mathcal{R} \) on \( D(V) \) is a reflexive, transitive and antisymmetric relation on \( D(V) \) (recall that \( \mathcal{R} \) is antisymmetric if and only if for all \( o, o' \in D(V) \), \( \mathcal{R}(o, o') \) and \( \mathcal{R}(o', o) \) implies \( o = o' \)). \( \mathcal{R}(o, o') \) is also denoted by \( o \succeq \mathcal{R} o' \). \( \mathcal{R} \) denotes the strict relation induced from \( \mathcal{R} \), defined by \( o \succ \mathcal{R} o' \) if and only if \( o \succeq \mathcal{R} o' \) and not \( o' \succeq \mathcal{R} o \). An order \( \mathcal{R} \) is complete (linear order) if and only if either \( o \succeq \mathcal{R} o' \) or \( o' \succeq \mathcal{R} o \) holds for all \( o, o' \in D(V) \). Let \( \mathcal{R}, \mathcal{R}' \) be two orders on \( V \), \( \mathcal{R}' \) extends \( \mathcal{R} \) if and only if \( \mathcal{R} \subseteq \mathcal{R}' \). An order \( \mathcal{R}' \) extends another order \( \mathcal{R} \) means, if \( o \succ \mathcal{R} o' \) then \( o \succ \mathcal{R}' o' \) (\( o \succ \mathcal{R} o' \) implies \( o \succ \mathcal{R}' o' \)). Lastly, a complete order \( \mathcal{R}' \) is a complete extension of \( \mathcal{R} \) if and only if \( \mathcal{R}' \) extends \( \mathcal{R} \). We denote by \( Ext(\mathcal{R}) \) the set of all complete extensions of \( \mathcal{R} \).

Let an order \( \mathcal{R}_j \) on \( D(V) \) denote the individual preference profile of an agent \( j \); and the collection of orders \( \mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_n\} \) be the (collective) preference profile, then \( \mathcal{R} \) is said to be complete if and only if \( \mathcal{R}_j \) is complete for each \( j \in \{1, \ldots, n\} \). Otherwise, \( \mathcal{R} \) is incomplete. A complete profile \( \mathcal{R}' = \{\mathcal{R}'_1, \ldots, \mathcal{R}'_n\} \) is a complete extensions of \( \mathcal{R} \) if and only if each complete order \( \mathcal{R}'_j \) extends \( \mathcal{R}_j \) (\( \mathcal{R}_j \in \mathcal{R} \)); and \( Ext(\mathcal{R}) \) denotes the set of all complete extensions of \( \mathcal{R} \). The notion of complete extension is generalized from individual to collective preference profiles:

\[
Ext(\mathcal{R}) = Ext(\mathcal{R}_1) \times \cdots \times Ext(\mathcal{R}_n)
\]

**Majority rule**

Given a set of alternatives, we need to aggregate multiple agents’ preferences and finally decide on one. In classical social choice theory, majority rule is one of the most well known aggregation rule for collective decision-making. It is a binary decision rule that selects one of two alternatives, based on which has more than half of the votes. Consider a group PC configuration problem as a motivating example. Suppose Alice
prefers the Intel processor to the AMD processor. On the other hand, Jack prefers the AMD processor to the Intel one. Finally, Bob is the same as Alice and prefers Intel to AMD. What should be the final decision when both of the options are Pareto-optimal? In this example the Intel processor may probably be a “best” choice according to majority rule as it is the preferred choice for both Alice and Bob, whilst AMD is the preferred option only for Jack. Given two alternatives \( o \) and \( o' \), we say \( o \) majority dominates (is majority preferred to) \( o' \) if there is a majority of agents prefer \( o \) to \( o' \).

Given that the preference profile is complete, for two alternatives \( o \) and \( o' \), we have \( o \) majority dominates \( o' \) (written as \( o \succ_{maj} o' \)) if and only if there is a majority of agents prefers \( o \) to \( o' \); and \( o \) majority ties with \( o' \) (written as \( o \sim o' \)) if and only if the number of agents who prefer \( o \) to \( o' \) is equal to the number of agents who prefer \( o' \) to \( o \). The following definitions of the Condorcet winner are adapted from the standard social choice literature with complete preference profiles [9]:

**Definition 5.1 (Condorcet Winner)** An alternative \( o \) is a Condorcet winner if and only if it majority-dominates every other alternative in a pair wise matchup: \( \forall o' \in O \) and \( o' \neq o \), \( o \succ_{maj} o' \).

**Definition 5.2 (Weak Condorcet Winner)** An alternative \( o \) is a weak Condorcet winner if and only if it majority-dominates or is incomparable to every other alternative in a pair wise matchup: \( \forall o' \in O \) and \( o' \neq o \), either \( o \succ_{maj} o' \) or \( o \sim o' \) holds.

When the Condorcet winner exists, it is unique. A Condorcet winner is also a weak Condorcet winner, while the reverse does not hold: a weak Condorcet winner is not necessarily a Condorcet winner. A weak Condorcet winner only exists in the case when there are even number of agents.

In majority-rule based group decision-making, however, it is possible for a paradox to form, in which the collective preferences can be cyclic (i.e. not transitive), even if the preferences of individual agents are not. For instance, it is possible that there are three alternatives \( o \), \( o' \), and \( o'' \), such that a majority prefers \( o \) to \( o' \), another majority prefers \( o' \) to \( o'' \), and yet another majority prefers \( o'' \) to \( o \). Consider the case that there are three agents (agent 1, agent 2 and agent 3) with the following preferences respectively. Agent 1: \( o > o' > o'' \); agent 2: \( o' > o'' > o \); and agent 3: \( o'' > o > o' \). In such case, a majority of agents (agent 1 and agent 3) prefer \( o \) to \( o' \), another majority of agents (agent 1 and agent 2) prefer \( o' \) to \( o'' \), and yet another majority of agents (agent 2 and
agent 3) prefer \( o'' \) to \( o \). Consequently, in this case, the requirement of majority rule then provides no clear Condorcet winner. Consequently, the set of majority winning alternatives can be empty. On the other hand, it is also possible that there are more than one weak Condorcet winners when the number of agents is even. Note that the set of weak Condorcet winners majority tie with each other (with complete preference profiles).

In the context of preference aggregation from a collection of agents’ CP-nets \( N = \{N_1, \ldots, N_n\} \), each \( N_j \) compactly represents an agent’s preference over the alternative space \( D(V) \). Let an order \( R_j \) on \( D(V) \) denote the individual preference profile of an agent \( j \) induced by his CP-net \( N_j \); and the collection of orders \( R = \{R_1, \ldots, R_n\} \) be the (collective) preference profile. As each preference ordering \( R_j \) induced by the CP-net \( N_j \) is generally incomplete, \( R \) is also generally incomplete (a profile of partial orders). In order to cope with this new setting where the profile is generally incomplete, Konczak and Lang [84] extend the notions of Condorcet winners. In particular, we will call, 

i) a necessary Condorcet winner, a candidate which is a Condorcet winner for all complete extensions of \( R \); and

ii) a possible winner, a candidate which is a Condorcet winner for at least one complete extensions of \( R \). Note that these notions are defined under the assumption that there are odd number of agents. Finally notice that in this work, we operate directly on the agents’ CP-nets. Thus, no reasoning on the individual or collective preference ordering over the alternative space is required.

Given two alternatives \( o \) and \( o' \) and \( o' \neq o \), let \( S_o, S_{o'}, S_\neq \) be the sets of agents whose CP-nets induce, respectively, that \( o \succ o' \), \( o \prec o' \), and \( o \sim o' \) (incomparable). We assume in this thesis that the agents’ CP-nets do not model indifference relations. Consequently, the preference order over the alternative space induced by an individual agent’s CP-net is irreflexive, transitive and antisymmetric. We first define the following semantics of necessary majority relation in the context of CP-nets, from which the necessary and possible winners can be clearly redefined. Moreover, the semantics and notions defined in the following are general enough to deal with both the cases when there are odd number of agents and when there are even number of agents.

**Definition 5.3 (Necessary majority semantics)** Given two alternatives \( o \) and \( o' \),

**Domination.** We say \( o \) necessarily majority dominates \( o' \), written as \( o \succmaj o' \) if and only if there is a majority of agents, whose CP-nets induce that \( o \succ o' \).
Mathematically,
\[ o \succ \succ_{\text{maj}} o' \iff |S_o| > |S_{o'}| + |S_\infty| \]

If \( o \succ \succ_{\text{maj}} o' \), then \( o \) is majority preferred to \( o' \) in all complete extensions of \( \mathcal{R} \).

Tie. We say \( o \) necessarily ties with \( o' \), written as \( o \sim \succ_{\text{maj}} o' \) (equally, \( o' \sim \succ_{\text{maj}} o \)), if and only if \( o \) and \( o' \) are comparable for every agent and the number of agents whose CP-nets induce \( o \succ o' \) is equal to the number of agents whose CP-nets induce \( o' \succ o \). Mathematically,
\[ o \sim \succ_{\text{maj}} o' \iff |S_\infty| = 0 \& \& |S_o| = |S_{o'}| \]

Clearly, a tie will happen only when there are even number of agents.

Incomparability. And finally, if \( o \) does not dominate \( o' \), \( o' \) does not dominate \( o \) and they do not tie with each other, then we say \( o \) and \( o' \) are incomparable according to the necessary majority semantics, written as \( o \prec_{\text{maj}} o' \). Mathematically,
\[ o \prec_{\text{maj}} o' \iff o \succ_{\text{maj}} o' \& \& o' \succ_{\text{maj}} o \& \& o \sim_{\text{maj}} o' \]

Lemma 5.1 For the properties of these binary relations based on necessary majority semantics, we have:

- \( \succ_{\text{maj}} \) is irreflexive: \( \forall o \in O, \neg o \succ_{\text{maj}} o \).
- \( \succ_{\text{maj}} \) is antisymmetric: \( \forall o, o' \in O, o \succ_{\text{maj}} o' \Rightarrow o' \not\succ_{\text{maj}} o \).
- \( \sim_{\text{maj}} \) (resp. \( \prec_{\text{maj}} \)) is symmetric: \( \forall o, o' \in O, o \sim_{\text{maj}} o' \Rightarrow o' \sim_{\text{maj}} o \) (resp. \( \forall o, o' \in O, o \prec_{\text{maj}} o' \Rightarrow o' \prec_{\text{maj}} o \)).
- \( \succ_{\text{maj}} \) (resp. \( \sim_{\text{maj}} \)) is intransitive: \( \forall o, o', o'' \in O, o \succ_{\text{maj}} o' \& \& o' \succ_{\text{maj}} o'' \Rightarrow o \succ_{\text{maj}} o'' \) (resp. \( \forall o, o', o'' \in O, o \sim_{\text{maj}} o' \& \& o' \sim_{\text{maj}} o'' \Rightarrow o \sim_{\text{maj}} o'' \)).

Proof. Clearly, the relation \( \succ_{\text{maj}} \) is irreflexive and \( \sim_{\text{maj}} \) (resp. \( \prec_{\text{maj}} \)) is symmetric. In the following, we prove the antisymmetry of \( \succ_{\text{maj}} \), and the intransitivity of \( \succ_{\text{maj}} \) and \( \sim_{\text{maj}} \).

Firstly, \( \succ_{\text{maj}} \) is antisymmetric because, \( \forall o, o' \in O, \) if \( o \succ_{\text{maj}} o' \), then \( |S_o| > |S_{o'}| + |S_\infty| \), and so \( |S_{o'}| < |S_o| - |S_\infty| \leq |S_o| + |S_\infty| \) (\( |S_\infty| \geq 0 \)). Consequently, \( |S_{o'}| < |S_o| + |S_\infty| \), and \( o' \not\succ_{\text{maj}} o \).
To prove $\succmaj$ is intransitive, consider the case that there are three agents and three outcomes $o$, $o'$ and $o''$, the preferences induced by the agents CP-nets are as follows.

Agent 1: $o \succ o' \succ o''$; agent 2: $o'' \succ o \succ o'$; agent 3: $o'' \succ o' \succ o$. Then we have $o \succmaj o'$ and $o' \succmaj o''$, however, $o \not\succmaj o''$ because comparing between $o$ and $o''$: $|S_o| \not> |S_{o'}| + |S_{o''}|$. On the contrast, in this case, we have $|S_{o''}| > |S_o| + |S_{o''}|$ and $o'' \succmaj o$, a paradox occurs.

Similarly, to prove $\simmaj$ is intransitive, consider the case that there are four agents and given three outcomes $o$, $o'$ and $o''$, the preferences induced by the agents CP-nets are as follows, respectively: agent 1: $o \succ o' \succ o''$; agent 2: $o'' \succ o \succ o'$ and agent 4: $o' \succ o'' \succ o$. Then we have $o \simmaj o'$ and $o' \simmaj o''$, however, $o'' \not\succmaj o$ because comparing between $o$ and $o''$: $|S_{o''}| > |S_o| + |S_{o''}|$, $o'' \succmaj o$ and so $o \simmaj o''$.

The necessary majority semantics can be considered as the lower approximation of the (partially known) majority relation on $D(V)$. This is because the strict preferences induced by a CP-net satisfy all the complete preference rankings extend that CP-net. Consequently, given two alternative $o$ and $o'$, the number $|S_o|$ and $|S_{o'}|$ will not be change when the agents can further provide complete orderings extend their own CP-nets. Thus, $o \succmaj o'$ (resp. $o \simmaj o'$), essentially means $o$ majority dominates $o'$ (resp. $o$ ties with $o'$) in all complete extensions of the partial preference profile $R$ induced by the set of agents’ CP-nets $N$. It is clear that when the preference profiles induced by the collection of CP-nets are complete, i.e., each CP-net induces a complete preference ordering over $D(V)$, these notions of necessary majority preference relations coincide with the standard notion of majority preference relations with complete preference.

Based on the necessary majority semantics, we can mathematically redefine the necessary winner discussed in [84]. Moreover, the notion of necessary weak Condorcet winner is further defined, in order to deal with the case when there are even number of agents.

**Definition 5.4 (Necessary Winner)** An alternative $o$ is a necessary Condorcet winner if and only if it necessarily majority-dominates every other alternative in a pair wise matchup: $\forall o' \in O$ and $o' \neq o$, $o \succmaj o'$ holds. An alternative $o$ is a necessary weak Condorcet winner if and only if it necessarily majority-dominates or ties with every other alternative in a pair wise matchup: $\forall o' \in O$ and $o' \neq o$, either $o \succmaj o'$ or
Also based on the necessary majority semantics, we have the following lemma that mathematically reformulates the notion of possible winner.

**Lemma 5.2 (Possible Winner)** An alternative $o$ is a possible winner if and only if $\forall o' \in O$ and $o' \neq o$, $o' \not\succ_{maj} o$ holds.

**Proof.** On the one hand, a possible winner $o$ is a weak Condorcet winner for at least one complete extensions of $R$, that means there exists at least one completion extension of $R$, denoted by $R'$, such that $\forall o' \in O$ and $o' \neq o$, either $o \succ_{maj} o'$ or $o \sim_{maj} o'$ holds, i.e., $o' \not\succ_{maj} o$ in $R'$. According to Definition 5.3, clearly, $o' \succ_{maj} o$ does not hold, because $o' \succ_{maj} o$ requires that $o \succ_{maj} o$ holds for all complete extensions of $R$. Consequently, if $o$ is a possible winner then $\forall o' \in O$ and $o' \neq o$, $o' \not\succ_{maj} o$ holds.

On the other hand, if $\forall o' \in O$ and $o' \neq o$, $o' \not\succ_{maj} o$, then $o$ and $o'$ can stand in the following three situations: $o \succ_{maj} o'$, or $o \sim_{maj} o'$, or $o \not\succ_{maj} o'$. The relation of $\succ_{maj}$ and $\sim_{maj}$ are necessary relations that imply $\succ_{maj}$ and $\sim_{maj}$ for all complete extensions of $R$ respectively. So if $o \succ_{maj} o'$ (resp. $o \sim_{maj} o'$), then $o \succ_{maj} o'$ (resp. $o \sim_{maj} o'$) for all complete extensions of $R$. If $o \not\succ_{maj} o'$, as $Ext(R) = Ext(R_1) \times \cdots \times Ext(R_m)$, there must exist at least one extension $R'$, such that all the agents in $S_o$ favour $o$ over $o'$. Thus, there exists at least one complete extensions $R' \in Ext(R)$, such that $\forall o' \in O$ and $o \neq o'$, $o \succ_{maj} o'$ or $o \sim_{maj} o'$ holds. Then $o$ is a weak Condorcet winner for $R'$. Thus, $o$ is a possible winner.

When a necessary Condorcet winner exists, it is unique. A necessary Condorcet winner is also a necessary weak Condorcet winner, while the reverse does not hold: a necessary weak Condorcet winner is not guaranteed to be a necessary Condorcet winner. Moreover, a (weak) necessary Condorcet winner must also be a possible winner. When the (weak) necessary Condorcet winner(s) exists, it is the only possible winner(s).
5.2. Preliminaries

**Remark.**

- We have considered defining a reflexive domination \( o \succ^\text{maj} o' \), hoping that this relation can naturally derive the irreflexive relation \( \succ^\text{maj} \) and the ties \( \sim^\text{maj} \):
  
  \[
  \begin{align*}
  &- o \succ^\text{maj} o' \text{ iff } |S_o| \geq |S_{o'}| + |S_\infty|; \\
  &- o \succ^\text{maj} o' \text{ iff } o \succ^\text{maj} o' \text{ and } o' \not\succ^\text{maj} o; \\
  &- o \sim^\text{maj} o' \text{ iff } o \succ^\text{maj} o' \text{ and } o' \succ^\text{maj} o; \\
  &- o \preceq^\text{maj} o' \text{ iff } o \not\succ^\text{maj} o' \text{ and } o' \not\succ^\text{maj} o.
  \end{align*}
\]

However, in some cases, it is incorrect to induce \( \succ^\text{maj} \) from \( \succ^\text{maj} \). For instance, given two outcomes \( o \) and \( o' \) and three agents, \( |S_o| = 2 \), \( |S_{o'}| = 1 \) and \( |S_\infty| = 1 \). Then, \( o \succ^\text{maj} o' \) and \( o' \not\succ^\text{maj} o \). However, it is not correct to say that \( o \succ^\text{maj} o' \), because \( |S_o| \not\succ |S_{o'}| + |S_\infty| \). Consequently, we choose to directly define the necessary dominant relation \( \succ^\text{maj} \) and the tie \( \sim^\text{maj} \).

- We have also considered using another potential semantics \( \succ^+_\text{maj} \) to define the possible domination relations between outcomes. In particular, given two alternatives \( o \) and \( o' \) and \( o \neq o' \), we may say \( o \succ^+_\text{maj} o' \) if \( |S_o| + |S_\infty| > |S_{o'}| \). That means, there is a majority of agents’ whose CP-nets do not induce \( o' \succ o \). As such, there exists at least one complete extension of \( R \) in which \( o \succ^+_\text{maj} o' \). Then, a possible winner \( o \) can be defined as \( \forall o' \in O \) and \( o' \neq o \), \( o \succ^+_\text{maj} o' \).

Unfortunately, such a relation is not *antisymmetric*: it may be the case that both \( o \succ^+_\text{maj} o' \) and \( o' \succ^+_\text{maj} o \) hold with \( o \neq o' \). For a simple example of three agents \( \{1, 2, 3\} \), that respectively, \( N_1 \models o \succ o' \), \( N_2 \models o' \succ o \) and \( N_3 \models o \preceq o' \). Then, \( |S_o| = |S_{o'}| = |S_\infty| = 1 \). As \( |S_o| + |S_\infty| > |S_{o'}| \) and \( |S_{o'}| + |S_\infty| > |S_o| \), both \( o \succ^+_\text{maj} o' \) and \( o' \succ^+_\text{maj} o \) hold. Moreover, according to the semantics, a winner can be dominated by some alternatives as it is possible that both directions of dominance coexist. Hence, there is not enough evidence or reason why it should be considered as a winner.

Consequently, for the sake of simplicity and clarity, in the rest of this chapter, we only consider the necessary majority semantics \( \succ^\text{maj} \).

- Notice that the possible winner here is equivalent to the concept “weak Condorcet winner” in our paper [104], where we actually rewrite the traditional definition of weak Condorcet winner such that it is general enough to cope with incomplete preference profiles.
5.3 H-composition of majority rules

Rossi et al. [125] study the computational complexity of a brute-force algorithm for aggregating preference with CP-nets based on majority rule. Suppose that we are making a decision over a set of \( m \) variables \( V = \{X_1, \ldots, X_m\} \) from a set of \( n \) agents’ CP-nets \( N = \{N_1, \ldots, N_n\} \). The high computational costs come from two aspects:

- The combinatorial structure of the domain. As discussed in [125], in a combinatorial domain with \( m \) variables, in order to test whether an alternative is a majority winner we need to compare the given alternative with all other alternatives (2\(^m\)) in all CP-nets (\( n \)). Moreover, in order to find the set of majority winners, we further need to compare all alternatives (2\(^m\)) to all other alternatives (2\(^m\)) in all CP-nets(\( n \)) in order to find the set of majority winners.

- The computational cost of individual dominance testing. Give a pair of outcomes \( o \) and \( o' \), deciding whether \( o \) is preferred to \( o' \) according to an agent’s preference is known to be NP-hard [71].

Consequently, it is important to avoid pair-wise comparison between outcomes as much as possible.

Instead of applying voting rules directly over the alternative space, Xia et al. [148] propose a *Hypercube-wise composition (H-composition)* of local voting rules. A H-composition of local rules is defined as the following two steps. First, the set of all possible alternatives are represented as a directed graph, and alternatives that differ on only one variable are neighbours on this directed graph as discussed by Domshlak and Brafman [50]. Then an induced graph is generated by applying local rules to each pair of neighbours. In the second step, a choice set is selected based on the induced graph as the set of winners. Although constructing the directed graph could not avoid listing all possible alternatives, the comparisons between neighbours based on local rules are straightforward and computationally easy. Basically, in order to decide the direction of an edge between a pair of neighbours that differs only in the value of a single variable \( X \), each agent just needs to refer to its CPT of \( X \) and see which value of \( X \) is more preferred between this pair of neighbours.

According to the representation by Xia et al. [148], we apply majority rule between each pair of neighbours and obtain the following majority induced graph. Note that
when the domain variables are all binary (binary CP-nets), the majority induced graph is also called a majority hypercube, which coincides with the latest related literature [43].

**Definition 5.5 (Majority induced graph)** Given a collection of CP-nets $\mathcal{N} = \{\mathcal{N}_1, \ldots, \mathcal{N}_n\}$, the majority induced graph, denoted by $\mathcal{H} = (O, E)$, is defined by the following edges between alternatives. For each variable $X$, any two alternatives $o, o' \in O$ that differ only on the value of $X$, let there be a directed edge $o \rightarrow o'$ if a majority of agents prefer $o$ to $o'$; there be a directed edge $o' \rightarrow o$ if a majority of agents prefer $o'$ to $o$. If $o$ ties with $o'$, $\mathcal{H}$ does not contain any edge between $o$ and $o'$.

For any two alternative $o, o' \in O$ that differ only on the value of $X$, $o[X] = x$ and $o'[X] = x'$, let $W = V - \{X\}$ and $w = o[W] (= o'[W])$. Whether or not there is a directed edge $o \rightarrow o'$ (resp. $o' \rightarrow o$) can be computed directly from the conditional preference table $CPT_j(X)$ of each agent $j$'s CP-net $\mathcal{N}_j$. Because for each agent $j$, $\mathcal{N}_j \models o \succ o'$ (resp. $\mathcal{N}_j \models o' \succ o$) if and only if $x \succ^{X|W}_{\mathcal{N}_j} x'$ (resp. $x' \succ^{X|W}_{\mathcal{N}_j} x$). Note that for an individual agent, neighbours are always comparable. However, a pair of neighbours $o$ and $o'$ can tie with each other, when the number of agents is even and the number of agents who prefer $o$ to $o'$ is equal to the number of agents who prefer $o'$ to $o$.

The dominance relations in $\mathcal{H}$ are then induced by the directed paths between alternatives [148]:

**Definition 5.6 (Dominance)** Given a collection of CP-nets $\mathcal{N} = \{\mathcal{N}_1, \ldots, \mathcal{N}_n\}$, let $\mathcal{H} = (O, E)$ be the majority induced graph. For any $o, o' \in O$, we say that $o$ dominates $o'$ in $\mathcal{H}$, denoted by $o \succ_{\mathcal{H}} o'$ if and only if: i) there is a directed path from $o$ to $o'$, and ii) there is no directed path from $o'$ to $o$.

According to Xia et al. [148], the transitive closure $\succeq_{\mathcal{H}}$ of $E$ specifies the minimum preorder such that if there is a directed path from $o$ to $o'$ in $\mathcal{H}$ then $o \succeq_{\mathcal{H}} o'$. $\succ_{\mathcal{H}}$ is the strict order induced by $\succeq_{\mathcal{H}}$: $o \succ_{\mathcal{H}} o'$ if and only if $o \succeq_{\mathcal{H}} o' \land o' \not\succeq_{\mathcal{H}} o$. Based on the induced graph, a choice set function is then defined, which always chooses the following alternatives as the winners.

**Definition 5.7** Let $\mathcal{H} = (O, E)$ be the majority induced graph for a collection of CP-nets $\mathcal{N} = \{\mathcal{N}_1, \ldots, \mathcal{N}_n\}$. We say,
• an alternative is a global Condorcet winner (GCW), if it dominates all other alternatives in \( \mathcal{H} \);

• an alternative is a local Condorcet winner (LCW), if it dominates all its neighbours in \( \mathcal{H} \);

• an alternative is a weak local Condorcet winner (wLCW), if it dominates or ties with all its neighbours in \( \mathcal{H} \).

When the global Condorcet winner (GCW) exists, it is unique. A GCW is also a local Condorcet winner (LCW), while the reverse does not hold: a LCW is not necessarily a GCW. When GCW exists, it is then the only one LCW. Similarly, a LCW is also a weak local Condorcet winner (wLCW), while a wLCW is not necessarily a LCW. wLCW exists only in the case where the number of agents is even. When the CP-nets are all binary, the wLCWs, LCW and GCW are also called hypercube Condorcet winners in [43]. Finally observe that there can be more than one wLCWs or LCWs and it can be the case that there exist both LCWs and wLCWs.

We denote by \( PW(\mathcal{N}) \) the set of possible winners for the set of agents’ CP-nets \( \mathcal{N} \). We denote by \( GCW(\mathcal{H}), LCW(\mathcal{H}), wLCW(\mathcal{H}) \), respectively, the global Condorcet winner, the set of local Condorcet winners, and the set of weak local Condorcet winners in the majority induced graph \( \mathcal{H} \). We emphasize here that none of the \( GCW(\mathcal{H}), LCW(\mathcal{H}) \) and \( wLCW(\mathcal{H}) \) is identical to the possible winners \( PW(\mathcal{R}) \). In particular, we will prove later that \( PW(\mathcal{R}) \) is a subset of \( wLCW(\mathcal{H}) \) \( (PW(\mathcal{R}) \subseteq wLCW(\mathcal{H})) \).

In the following, we first analyse the relation between the preferences derived from a majority induced graph \( \mathcal{H} \) and the necessary majority dominance among the agents.

**Proposition 5.1** Necessary majority-domination \( \succ_{maj} \) does not follow from the domination \( \succ_{\mathcal{H}} \) derived from the majority induced graph \( \mathcal{H} \).

**Proof.** To prove this proposition, we need to prove that given a collection of CP-nets \( \mathcal{N} = \{\mathcal{N}_1, \ldots, \mathcal{N}_n\} \), the majority induced graph \( \mathcal{H} = (O, E) \) and a pair of alternatives \( o, o' \in O \) and \( o \neq o' \), it may be the case that \( o \succ_{\mathcal{H}} o' \) but \( o \not\succ_{maj} o' \). Consider an example of 3 agents making decision over 2 binary domain variables. The agents’ CP-nets, their partial order over the alternative space and the majority hypercube (as all variables are binary) are depicted in Figure 5.1. According to the majority hypercube...
5.3. \( H \)-composition of majority rules

(a) \( \mathcal{N}_1 \)

(b) \( \mathcal{N}_2 \)

(c) \( \mathcal{N}_3 \)

(d) Majority hypercube

Figure 5.1: Illustration for Proposition 5.1

(see Figure 5.1(d)), there is a directed path from outcome \( ab \) to \( \bar{a}b \) and no directed path from \( \bar{a}b \) to \( ab \), i.e. \( ab \succ_\mathcal{H} \bar{a}b \). However, for both \( \mathcal{N}_1 \) and \( \mathcal{N}_2 \), these two alternatives are incomparable (see Figure 5.1(a) and Figure 5.1(b)). Thus, \( \bar{a}b \) and \( ab \) are incomparable according to the necessary majority semantics, i.e. \( \bar{a}b \not\succ_{\text{maj}} ab \). Consequently, in this example, \( ab \succ_\mathcal{H} \bar{a}b \) but \( ab \not\succ_{\text{maj}} \bar{a}b \). \( \square \)

**Proposition 5.2** Domination \( \succ_\mathcal{H} \) derived from the majority induced graph does not preserve the strict necessary majority preference relation \( \succ_{\text{maj}} \).

**Proof.** To prove this proposition, we need to prove that given a collection of CP-nets \( \mathbf{N} = \{ \mathcal{N}_1, \ldots, \mathcal{N}_n \} \), the majority induced graph \( \mathcal{H} = (O, E) \) and a pair of alternatives \( o, o' \in O \) and \( o \neq o' \), it may be the case that \( o \succ_{\text{maj}} o' \) but \( o \not\succ_\mathcal{H} o' \). Consider the agents’ CP-nets, their partial order over the alternative space and the corresponding majority hypercube depicted in Figure 5.2. According to the majority hypercube Figure. 5.2(d), there is no directed path from alternative \( \bar{a}bc \) to alternative \( abc \), i.e. \( \bar{a}bc \not\succ_{\mathcal{H}} abc \). However, both agent 1 and agent 2 preferred \( \bar{a}bc \) to \( abc \) (see Figure 5.2(a) and Figure 5.2(b)). Hence, \( \bar{a}bc \succ_{\text{maj}} abc \). Consequently, in this example, \( \bar{a}bc \succ_{\text{maj}} abc \) but \( \bar{a}bc \not\succ_{\mathcal{H}} abc \). \( \square \)

As \( \succ_\mathcal{H} \) does not preserve the necessary majority preference \( \succ_{\text{maj}} \), a (weak) local Condorcet winner that dominates (or ties) with all its neighbors may still be necessarily majority-dominated by some alternative. Thus, it is not guaranteed to be a possible winner.
Corollary 5.1 A (weak) local Condorcet winner is not guaranteed to be a possible winner.

Consider the example in Figure 5.2. Alternative abc is a local Condorcet winner as it dominates all its neighbours (abś, ābc and ābś) in the majority hypercube (see Figure 5.2(d)). However, ābś $\succ maj$ abc because both $N_1$ (Figure 5.2(a)) and $N_2$ (Figure 5.2(b)) preferred ābś to abc, i.e., $|S_{ābś}| > |S_{abc}| + |S_{x}|$. Thus, abc is not a possible winner.

Now we are interested in whether or not the set of (weak) local Condorcet winners is guaranteed to be a non-dominated set according to the lower approximation $\succ maj$, i.e. whether an alternative in $wLCW(H)$ (or $LCW(H)$) can not be necessarily majority dominated by any other alternative not in $wLCW(H)$ (or $LCW(H)$). Unfortunately,
5.3. H-composition of majority rules

the following proposition gives a negative answer to this question. It further shows that even when a weak local Condorcet winner (wLCW) is not dominated by any other wLCWs, it can still be dominated by some other alternative.

**Proposition 5.3** A (weak) local Condorcet winner \( o (o \in w\text{LCW}(\mathcal{H})) \) can be necessarily majority-dominated by an alternative \( o' \) that \( o' \notin w\text{LCW}(\mathcal{H}) \), even though for all \( o'' \in w\text{LCW}(\mathcal{H}) \), \( o'' \not\succ maj o \).

**Proof.** Consider the example in Figure 5.2, there are only two LCWs \( abc \) and \( \bar{a}bc \) and \( abc \not\succ maj \bar{a}bc \): they are incomparable for both \( N_2 \) (see Figure 5.2(b)) and \( N_3 \) (see Figure 5.2(c)). However, as we mentioned before, \( \bar{a}bc \not\succ maj abc \), and \( \bar{a}bc \) is not a LCW or wLCW, \( \bar{a}bc \notin w\text{LCW}(\mathcal{H}) \). □
Finally, we are interested in the following question: whether a global Condorcet winner that dominates every other alternative in the majority induced graph, is guaranteed to be a possible winner for the collection of CP-nets. We subsequently have the following proposition, from which the above question could be clearly answered.

**Proposition 5.4** Domination $\succ_{\mathcal{H}}$ derived from the majority induced graph may conflict with necessary majority domination $\succ_{\text{maj}}$.

**Proof.** To prove this proposition, we need to prove that given a collection of CP-nets $\mathbf{N} = \{N_1, \ldots, N_n\}$, the majority induced graph $\mathcal{H} = (O, E)$ and a pair of alternatives $o, o' \in O$ and $o \neq o'$, it may be the case that $o \succ_{\mathcal{H}} o'$ but $o' \succ_{\text{maj}} o$. Consider the agents’ CP-nets, their preference ordering over the alternative space and the corresponding majority hypercube in Figure 5.3. According to the majority hypercube Figure. 5.3(d), there is a directed path from $abc$ to $\bar{a}\bar{b}\bar{c}$ and no directed path from $\bar{a}\bar{b}\bar{c}$ to $abc$, thus $abc \succ_{\mathcal{H}} \bar{a}\bar{b}\bar{c}$. However, agent 1 and 2 prefer $\bar{a}\bar{b}\bar{c}$ to $abc$ (see Figure 5.3(a) and Figure 5.3(b)) and so $|S_{\bar{a}\bar{b}\bar{c}}| > |S_{abc}| + |S_{\infty}|$: $\bar{a}\bar{b}\bar{c} \succ_{\text{maj}} abc$. Consequently, in this example, $abc \succ_{\mathcal{H}} \bar{a}\bar{b}\bar{c}$ but $\bar{a}\bar{b}\bar{c} \succ_{\text{maj}} abc$. □

**Corollary 5.2** A global Condorcet winner is not guaranteed to be a possible winner.

**Proof.** Consider the previous example used for Proposition 5.4 (See Figure 5.3). In this example, $abc$ is a unique global Condorcet winner in $\mathcal{H}$: there is a directed path from $abc$ to every other alternative and no incoming edges to $abc$ (see Figure 5.3(d)). However, as we mentioned before, this global Condorcet winner $abc$ is majority-dominated by $\bar{a}\bar{b}\bar{c}$ ($\bar{a}\bar{b}\bar{c} \succ_{\text{maj}} abc$). Consequently, $abc$ is a GCW, but not a possible winner. □

From the above, it become clear that the majority induced graph may not always represent the majority preferences properly. In particular, a winner in the majority induced graph, i.e., a GCW, LCW or wLCW winner is not necessarily a possible winner. However, we observe that a possible winner must be at least a weak local Condorcet winner.

**Theorem 5.1** Let $\mathcal{H} = (O, E)$ be the majority induced graph for a collection of CP-nets $\mathbf{N} = \{N_1, \ldots, N_n\}$. Then a possible winner is also a weak local Condorcet winner in $\mathcal{H}$: $\text{PW}(\mathbf{N}) \subseteq \text{wLCW}(\mathcal{H})$. 

124
5.3. \textit{H-composition of majority rules}

**PROOF.** Suppose a possible winner \( o \) is not a weak local Condorcet winner (wLCW), then there exist at least one neighbour \( o' \) in \( \mathcal{H} \) such that \( o' \rightarrow o \in E \). That means, there is a majority of agents prefers \( o' \) to \( o \) (\( o' \succ_{\text{maj}} o \)), contradicting the fact that \( o \) is a possible winner. Thus, a possible winner must also be a wLCW. \( \square \)

A possible winner must be a wLCW, while the reverse does not hold: a wLCW is not necessarily a possible winner (See Corollary 5.1). Nonetheless, this relation provides us a more efficient way to compute the possible winners among a large alternative space: first compute the set of wLCWs, and then compute the set of possible winners by filtering out those that are necessarily majority-dominated by some alternative.

**Remark.**

- Here the notion of majority induced graph is identical to the definition in [148] when the number of agents is odd, and differ only in the presence of ties between neighbours when the number of agents is even. According to [148], when a pair of neighbours \( o \) and \( o' \) tie with each other, \( \mathcal{H} \) contains directed edges both from \( o \) to \( o' \) and from \( o' \) to \( o \). However, their definition may exclude the possible winner when the number of agents is even. For instance, if a possible winner necessarily ties with one of its neighbours, then it is not considered to be a wLCW (nor a GCW or LCW) according to their definition.

- We define the notions of local Condorcet winner, weak local Condorcet winner, and global Condorcet winner. However, we do not have a definition for a weak global Condorcet winner. This is because there won’t exist more than one global Condorcet winners. Consequently, it would not be the case that more than one global Condorcet winners tie with each other. If \( o \) is a global Condorcet winner, then \( o \) must dominate any other outcome \( o' \) in the directed graph, i.e., there must exist a directed path from \( o \) to every other outcome \( o' \).

In fact, the definition of local tie is clear, as if a pair of neighbours \( o \) and \( o' \) ties with each other, then the number of agents who prefer \( o \) over \( o' \) is equal to the number of agents who prefer \( o' \) to \( o \). However, there is no reasonable way to define a global tie, because the comparisons between outcomes that are not neighbours are not direct. By contrast, such a comparison is based on the transitive closure (directed path) of the graph; however, majority rule does not comply with transitivity.
5.4 The MajCP approach

This section presents the proposed CSP-based approach, MajCP (Majority-rule-based collective decision-making with CP-nets), for computing the possible winners from a collection of CP-nets. The proposed approach includes the following two steps. First, it computes the set of wLCWs (weak local Condorcet winners) via a reduction to an extended CSP (Constraint satisfaction problems) \(^2\) for cardinality constraints. Then, in the second step, the set of possible winners can be obtained by filtering out those that are necessarily majority-dominated by some alternative.

Assume \(n\) agents \(A = \{1, \ldots, n\}\) are making decisions over a set of \(m\) variables \(V = \{X_1, \ldots, X_m\}\). Let \(q = n/2 + 1\) be the minimum number that satisfies the majority criterion. The preference of each agent \(j\) is captured by a (possibly cyclic) multi-valued CP-net \(N_j\) and let \(N = \{N_1, \ldots, N_n\}\). We reduce the problem of computing the set of weak local Condorcet winners (wLCWs) to a corresponding CSP (or SAT) problem by generating a set of optimality constraints that a wLCW must satisfy according to majority rule. The variables in our reduction consist of the variables in the agents’ CP-nets. This section discusses the optimality constraints for each variable in the CP-nets. It firstly analysis the optimality constraints for binary-valued CP-nets, then it extends to multi-valued CP-nets. For each variable \(X\), let \(D(X) = \{x_1, \ldots, x_k\}\) (\(D(X) = \{x, \bar{x}\}\) if \(X\) is binary) be the domain of \(X\), each agent \(j\) has a conditional preference table \(CPT_j(X)\) stating the conditional preference on the values of variable \(X\) with each instantiation of \(X\)’s parents \(Pa_j(X)\). For any pair of values \(x, x'\) of a variable \(X\), we separate these condition entry in \(CPT_j(X)\) into the following two categories.

- The set of parent context in which agent \(j\) prefers \(x\) to \(x'\):
  \[
  U_{x > x'}^j = \{ u \in D(Pa_j(X)) \mid x >_{N_j}^x x' \} 
  \]

- The set of parent context in which agent \(j\) prefers \(x'\) to \(x\):
  \[
  U_{x' > x}^j = \{ u \in D(Pa_j(X)) \mid x' >_{N_j}^x x \} 
  \]

Let \(P_{x > x'}^j = \bigvee_{u \in U_{x > x'}^j} u\) (resp. \(P_{x' > x}^j = \bigvee_{u \in U_{x' > x}^j} u\)), i.e., the disjunction of the condition part of the entry whose conclusion is \(x > x'\) (resp. \(x' > x\)) in the \(CPT_j(X)\) of agent.

\(^2\)When all variables are binary, it is a SAT (Boolean satisfiability problem).
5.4. The MajorityCP approach

Thus, \( x \succ^*_N j \ x' \) (resp. \( x' \succ^*_N j \ x \)). Moreover, for any setting \( w = D(W) \) (\( W = V - \{X\} \)) that satisfies \( P^j_{x \succ x} \) (resp. \( P^j_{x' \succ x} \)), then \( xw \succ_N j x'w \) (resp. \( x'w \succ_N j xw \)). Note that if agent \( j \) has unconditional preference over a variable \( X \), \( Pa_j(X) = \emptyset \) and \( x \succ^*_N j x' \) (resp. \( x' \succ^*_N j x \)), that means the condition \( P^j_{x \succ x} \) (resp. \( P^j_{x' \succ x} \)) is always True and \( P^j_{x \succ x} \) (resp. \( P^j_{x' \succ x} \)) is always False. For each individual agent \( j \), if \( N_j \) is complete\(^3\), \( U_{x \succ x} \) and \( U_{x' \succ x} \) are complementary. Thus, \( P^j_{x \succ x} = -P^j_{x' \succ x} \) (resp. \( P^j_{x' \succ x} = -P^j_{x \succ x} \)). However, we emphasize that in this work we do not require the agents’ CP-nets to be complete.

Given a directed graph \( H = (O,E) \), for any two alternatives \( o, o' \in O \) that differ only on the value of \( X \): \( o[X] = x \) and \( o'[X] = x' \). There is a directed edge \( o \rightarrow o' \) (resp. \( o' \rightarrow o \)) in \( H \) if and only if, for the setting \( w = o[W] = o'[W] \) and \( W = V - \{X\} \), there exists a set of at least \( q \) CP-nets (recall that \( q = n/2+1 \)), denoted by \( N^* (N^* \subseteq N^*) \), \( \forall N_j \in N^* \) has the following conditional (unconditional) preference \( x \succ^*_N j x' \) (resp. \( x' \succ^*_N j x \)), i.e., \( w \) satisfies \( \bigwedge_{N_j \in N^*} P^j_{x \succ x} \) (resp. \( \bigwedge_{N_j \in N^*} P^j_{x' \succ x} \)). Furthermore, there will be a set of \( \binom{n}{q} \) distinct \( q \)-subsets of agents that satisfies this majority requirement, denoted by \( S \). Consequently, if the setting \( w \) satisfies \( \bigvee_{N^* \in S} \left( \bigwedge_{N_j \in N^*} P^j_{x \succ x} \right) \) (resp. \( \bigvee_{N^* \in S} \left( \bigwedge_{N_j \in N^*} P^j_{x' \succ x} \right) \)), then there is a directed edge \( o \rightarrow o' \) (resp. \( o' \rightarrow o \)).

Thus, \( o \succ_H o' \) (resp. \( o' \succ_H o \)). For the purpose of explanation, we reason directly with cardinality formulas, which have been widely explored in CSPs and SAT (cardinality constraints), see e.g., [15] and [75]. For each variable \( X \), let \( F_{x \succ x'} \) and \( F_{x' \succ x} \) be the following cardinality formula respectively:

\[
F_{x \succ x'} = [\geq q] \left( P^1_{x \succ x'}, \ldots, P^n_{x \succ x'} \right) \tag{5.1}
\]

\[
F_{x' \succ x} = [\geq q] \left( P^1_{x' \succ x}, \ldots, P^n_{x' \succ x} \right) \tag{5.2}
\]

Such that \( F_{x \succ x'} \) (resp. \( F_{x' \succ x} \)) is True when at least \( q \) formulas among \( P^1_{x \succ x'}, \ldots, P^n_{x \succ x'} \) (resp. \( P^1_{x' \succ x}, \ldots, P^n_{x' \succ x} \)) are True. Note that the cardinality formula \( F_{x \succ x'} \) (resp. \( F_{x' \succ x} \)) is logically equivalent to the classical propositional formula \( \bigvee_{N^* \in S} \left( \bigwedge_{N_j \in N^*} P^j_{x \succ x} \right) \) (resp. \( \bigvee_{N^* \in S} \left( \bigwedge_{N_j \in N^*} P^j_{x' \succ x} \right) \)).

\(^3\)A CP-net is complete if, for each variable \( X \), the conditional preference table of \( X \) contains a total ordering over the values of \( X \) for each instantiation of \( X \)’s parents.
Consequently, given an directed graph $H = (O, E)$, let $o, \ o' \in O$ be two alternatives that differ only on the value of a variable $X$, $o [X] = x$ and $o' [X] = x'$. Let $w = o [W] (= o' [W])$ and $W = V - \{X\}$. If the setting $w$ satisfies $F_{x > x'}$ (resp. $F_{x' > x}$), then there is an directed edge $o \rightarrow o'$ (resp. $o' \rightarrow o$) in $H$.

### 5.4.1 Optimality constraints for binary-valued CP-nets

In binary-valued CP-net, each variable $X$ is binary, i.e., $D(X) = \{x, \bar{x}\}$. An alternative $o^*$ is a weak local Conдорctet winner, also called hypercube Conдорctet winner, iff there is on incoming edges from any neighbours. Consequently, for each variable $X$, if $o^*$ satisfies $F_{x > \bar{x}}$ (resp. $F_{\bar{x} > x}$), then $o^*[X] = x$ (resp. $o^*[X] = \bar{x}$). Otherwise, there will be an incoming edge from another outcome $o'$ that $o'$ assigns a different value to variable $X$ and there is a majority of agents prefers $o'$ to $o^*$. Hence, a weak local Conдорctet winners must satisfy the following optimality constraints for each variable $X$.

**Definition 5.8 (Optimality constraints)** Given a collection of CP-nets $N = \{N_1, \ldots, N_n\}$, for each variable $X$, the majority-optimality constraint to the value of $X$ is:

$$\varphi_X = (F_{x > \bar{x}} \Rightarrow X = x) \land (F_{\bar{x} > x} \Rightarrow X = \bar{x})$$

(5.3)

Note that if there is an odd number of agents and the agents’ CP-nets are complete, $F_{x > \bar{x}} = \neg F_{\bar{x} > x}$ and the above constraint can be simplified to:

$$\varphi_X = (F_{x > \bar{x}} \iff X = x)$$

### 5.4.2 Optimality constraints for multi-valued CP-nets

In multi-valued CP-nets, a variable $X$ may have more than two values. Let $D(X) = \{x_1, \ldots, x_k\}$ be the value domain of a variable $X$. We have discussed the optimality constraints for binary-valued CP-nets in the previous section. We note that for each variable $X$, if $F_{x > \bar{x}}$ (resp. $F_{\bar{x} > x}$) satisfied, then $X = x$ (resp. $X = \bar{x}$) is the dominant
5.4. The MajCP approach

value for variable $X$, i.e., there are a majority of agents prefers $x$ to $\bar{x}$ (resp. there are a majority of agents prefers $\bar{x}$ to $x$).

In multi-valued setting, we analyse the problem from another perspective: under what circumstances can a value of a variable be in a weak local Condorcet winner. Given a majority induced graph $\mathcal{H} = (O, E)$, for the value $x \in D(X)$, let $o \in O$ be an alternative that $o[X] = x$, $o' \in O$ be any neighbour of $o$ that differ only on the value of $X$, $o'[X] = x'$ ($x' \in D(X)$ and $x' \neq x$), there is an directed edge $o \rightarrow o'$ (resp. $o' \rightarrow o$) in $\mathcal{H}$ if and only if, for the setting $w = o[\mathcal{W}] (= o'[\mathcal{W}])$ and $\mathcal{W} = \mathcal{V} - \{X\}$, $w$ satisfies $F_{x \succ x'}$ (resp. $F_{x' \succ x}$).

Consequently, for a variable $X$, a value $x$ can be in a weak local Condorcet winner $o^*$ only if it is not majority-dominated by another value of $X$, i.e., $x$ is one of the non-dominated values under the parent context given by that weak local Condorcet winner $o^*$. As we know that if $F_{x' \succ x}$ is satisfied, then there is a majority of agents prefer $x'$ to $x$. Consequently, if $X = x$ is a non-dominated value of $X$, then for any other value $x'$ of $X$, $F_{x' \succ x}$ must not be satisfied. Formally, we have:

$$ (X = x) \Rightarrow \bigwedge_{x' \in D(X) \& \& x' \neq x} (\neg F_{x' \succ x}) \quad (5.4) $$

Consequently, the weak local Condorcet winners must satisfy the following optimality constraints for each variable $X$.

**Definition 5.9 (Optimality constraints)** Given a collection of CP-nets $\mathcal{N} = \{N_1, \ldots, N_n\}$, for each variable $X$, the majority-optimality constraint $\varphi_X$ to the value of $X$ is:

$$ \varphi_X = \bigwedge_{x \in D(X)} \left( (X = x) \Rightarrow \bigwedge_{x' \in D(X) \& \& x' \neq x} (\neg F_{x' \succ x}) \right) \quad (5.5) $$

Notice that the constraint (Definition 5.9) is identical to the constraint (Definition 5.8) when the variable is binary. This is because $(X = x) \Rightarrow \neg F_{\bar{x} \succ x}$ (resp. $(X = \bar{x}) \Rightarrow \neg F_{x \succ \bar{x}}$) is logically equivalent to $F_{\bar{x} \succ x} \Rightarrow X = \bar{x}$ (resp. $F_{x \succ \bar{x}} \Rightarrow X = x$).
5.4.3 The corresponding CSP

Finally, the corresponding CSP (or SAT for binary-valued CP-nets) $\varphi$ is the conjunction of all $\varphi_X$ (one for each variable $X$) (line 26):

$$
\varphi = \bigwedge_{X \in V} \varphi_X
$$

(5.6)
5.4. The MajCP approach

**Input:** \( N \), a set of CP-nets of the agents.

**Output:** \( \varphi \), the corresponding CSP

\[
q \leftarrow (n + 1)/2 \text{ where } n \text{ is the total number of agents;}
\]

\[
\varphi \leftarrow \text{True}.
\]

**For each** \( X \in V \) **do**

**For each** \( x \in D(X) \) **do**

\[
\text{con} \leftarrow \text{True};
\]

**For each** \( x' \in D(X) \) **do**

\[
\text{if } x' \neq x \text{ then}
\]

\[
\text{list}' \leftarrow \emptyset;
\]

**For each** \( N_j \in N \) **do**

\[
\text{if } Pa_j(X) = \emptyset \text{ then}
\]

\[
\text{if } x >^X_{N_j} x' \text{ then}
\]

\[
P_j^{x > x} \leftarrow \text{False};
\]

\[
\text{else}
\]

\[
P_j^{x > x} \leftarrow \text{True};
\]

\[
\text{end}
\]

\[
\text{else}
\]

\[
P_j^{x > x} \leftarrow \text{False};
\]

**For each** \( cp\text{-statement} \in CPT_j(X) \) **do**

\[
\text{if } u \in U_{x'}^{j} \text{ then}
\]

\[
P_j^{x > x} \leftarrow P_j^{x > x} \lor u
\]

\[
\text{end}
\]

\[
\text{end}
\]

\[
\text{add } P_j^{x > x} \text{ to list'};
\]

\[
F^{x > x} \leftarrow [\geq q] \text{list'};
\]

**end**

\[
\text{constraint} \leftarrow \text{constraint} \land \lnot F^{x > x};
\]

**end**

\[
\varphi_X \leftarrow \varphi_X \land ((X = x) \rightarrow \text{constraint});
\]

**end**

\[
\varphi \leftarrow \varphi \land \varphi_X
\]

**end**

**return** \( \varphi \);

\[\text{Algorithm 5.2: ConvertToCSP}\]

**Theorem 5.2** Let \( \mathcal{H} = (O, E) \) be the majority induced graph for a collection of CP-nets \( N = \{N_1, \ldots, N_n\} \). An alternative \( o \) is a weak local Condorcet winner if and only if it satisfies the above CSP (or SAT) \( \varphi \).
Chapter 5. Majority-rule-based preference aggregation with CP-nets

Input: \( N \), a set of CP-nets of the agents; \( \varphi \), the corresponding CSP or SAT

Output: \( CW \), a set of (weak) Condorcet winners

\[
\begin{align*}
&\varphi \leftarrow \text{ConvertToCSP}(N); \quad /\!*\!\varphi \leftarrow \text{ConvertToSAT}(N) \text{ for binary CP-nets } */
&\text{graphWinners} \leftarrow \text{the models of } \varphi;
&\text{CW} \leftarrow \text{optimalityCheck}(\text{graphWinners}, N);
&\text{return CW;}
\end{align*}
\]

Algorithm 5.3: \textit{MajCP}

\begin{proof}
(Soundness) Let \( o \) be an alternative that satisfies \( \varphi \). For any neighbour \( o' \) of \( o \) that differs on the value of a single variable \( X \in V \), as \( o \) satisfies \( \varphi_X \), then there is no directed edge \( o' \rightarrow o \), i.e., either there is a directed edge \( o \rightarrow o' \), or there is no edge between \( o \) and \( o' \). According to Definition 5.7, \( o \) is a weak local Condorcet winner.

(Completeness) Assume first that there is at least one weak local Condorcet winner \( o \), and suppose that \( o \) does not satisfy \( \varphi \). Then there exists at least one optimality constraint \( \varphi_X \) that \( o \) does not satisfy. Assume that \( o[X] = x \). An implication is unsatisfied only when the hypothesis is \textit{True} and the conclusion is \textit{False}. According to Definition (5.9) and because \( o[X] = x \), we know that \( o \) does not satisfy \((X = x) \Rightarrow \bigwedge_{x' \in D(X)}(\neg F_{x' > x})\). That means, there exists at least one value \( x' \) of variable \( X \), such that \( F_{x' > x} \) is true given the values assigned to other variables by \( o \) (\( w = o[W] \) and \( W = V - \{X\} \)). Consequently, there must exist a neighbour \( o' \) of \( o \) that differ on the value of \( X \) and \( o'[X] = x' \), there is a majority of agents prefers \( o' \) to \( o \), and thus there is a directed edge \( o' \rightarrow o \) in the majority induced graph \( H(o' \succ_H o) \), contradicting the fact that \( o \) is a weak local Condorcet winner. Hence, the above CSP \( \varphi \) must be satisfied by all the weak local Condorcet winners. \( \Box \)

As such, we reduce the problem of computing (weak) local Condorcet winners (wLCWs) to an extended CSP or SAT problem with cardinality constraint (see Algorithm 5.1 and Algorithm 5.2 for converting the majority-rule-based preference aggregation problem into a corresponding SAT with binary CP-nets and CSP with multi-valued CP-nets, respectively). Then the set of wLCWs can be obtained by computing the models of the corresponding SAT (line 2, Algorithm 5.3) or CSP (line 2, Algorithm 5.3).

In the second step, we test the majority optimality of each weak local Condorcet winner (i.e. the models of the corresponding CSP or SAT) by comparing each wLCW to all other alternatives and filtering out those that are majority-dominated by some
5.4. The MajCP approach

Figure 5.4: CP-nets of the agents

(a) $N_1$

(b) $N_2$

(c) $N_3$

(d) $N_4$

alternative (line 3, Algorithm 5.3).

An illustration

This section demonstrates the execution of the proposed approach by an example with binary-valued CP-nets. Assume four agents $A = \{1, 2, 3, 4\}$ are making decision over a set of four Boolean variables $V = \{A, B, C, D\}$. Consider the agents’ CP-nets depicted in Figure 5.4. We first generate a set of majority-optimality constraints that a weak local Condorcet winner must satisfy. For variable $A$, we refer to each agent $j$’s conditional preference table $CPT_j(A)$:

$N_1$: $U_{a \rightarrow a}^1 = \{bd, \bar{b}\}$ and $U_{\bar{a} \rightarrow a}^1 = \{b\bar{d}, \bar{b}\}$, thus $P_{a \rightarrow a}^1 = bd \lor \bar{b}d$ and $P_{\bar{a} \rightarrow a}^1 = b\bar{d} \lor \bar{b}d$;

$N_2$: the preference over variable $A$ is unconditional, $a \succ_{N_2} \bar{a}$, thus $P_{a \rightarrow a}^2 = True$ and $P_{\bar{a} \rightarrow a}^2 = False$;
$N_3$: the preference over variable $A$ is unconditional, $a \succ^A_{N_3} \tilde{a}$, thus $P^3_{a\succ\tilde{a}} = True$ and $P^3_{a\succeq\tilde{a}} = False$;

$N_4$: $U^4_{a\succ\tilde{a}} = \{bd\}$ and $U^4_{a\succeq\tilde{a}} = \{bd, \tilde{b}d, \tilde{b}d\}$, thus $P^4_{a\succ\tilde{a}} = bd$ and $P^4_{a\succeq\tilde{a}} = bd \vee \tilde{b}d \vee \tilde{b}d$.

Consequently,

$F_{a\succ\tilde{a}} = [\geq 3](P^1_{a\succ\tilde{a}}, P^2_{a\succ\tilde{a}}, P^3_{a\succ\tilde{a}}, P^4_{a\succ\tilde{a}}) = [\geq 3](bd \vee \tilde{b}d, True, True, bd) = b \vee \tilde{d}

F_{a\succeq\tilde{a}} = [\geq 3](P^1_{a\succeq\tilde{a}}, P^2_{a\succeq\tilde{a}}, P^3_{a\succeq\tilde{a}}, P^4_{a\succeq\tilde{a}}) = [\geq 3](bd \vee \tilde{b}d, False, False, bd \vee \tilde{b}d

= False

Hence, the winning alternative must satisfy the following optimality constraint for variable $A$:

$\varphi_A = (b \vee \tilde{d} \Rightarrow a) \land (False \Rightarrow \tilde{a})$

An implication is unsatisfied only when the hypothesis is $True$ and the conclusion is $False$, thus $False \Rightarrow \tilde{a}$ is always $True$ and $\varphi_A$ is logically equivalent to:

$\varphi_A = b \vee \tilde{d} \Rightarrow a$

Similarly, we obtained the following optimality constraints (simplified form of the cardinality constraints) for variable $B$, $C$ and $D$:

$B$: $\varphi_B = (\tilde{a}c \Rightarrow b) \land (a\tilde{c} \vee a\tilde{d} \vee \tilde{c}d \Rightarrow \tilde{b})$

$C$: $\varphi_C = (\tilde{a}b \Rightarrow c) \land (a\tilde{b} \Rightarrow \tilde{c})$

$D$: $\varphi_D = \tilde{d}$

Consequently, we obtain the following SAT:

$\varphi = \varphi_A \land \varphi_B \land \varphi_C \land \varphi_D = (b \vee \tilde{d} \Rightarrow a) \land (\tilde{a}c \Rightarrow b) \land (a\tilde{c} \vee a\tilde{d} \vee \tilde{c}d \Rightarrow \tilde{b}) \land (\tilde{a}b \Rightarrow c) \land (a\tilde{b} \Rightarrow \tilde{c}) \land \tilde{d}$

The above SAT has only one satisfied assignment $abc\tilde{d}$. After checking the majority optimality of $abc\tilde{d}$, it is also a weak Condorcet winner in this example.

### 5.5 Complexity analysis

This section studies the computational issues of the proposed approach.

**Theorem 5.3 (complexity)**: Given a collection of $n$ CP-nets $N = \{N_1, \ldots, N_n\}$, if $\forall N_j \in N$, the node in-degree is bounded by a constant, then transforming the problem of computing weak local Condorcet winners into a corresponding extended CSP (or SAT) problem for cardinality constraints is polynomial.
5.5. Complexity analysis

PROOF. Assume there are $m$ variables and the number of parents of a node in the dependency graph of each agent is bounded by a constant $d$. Let $k$ be the maximum domain size of the variables in the agents’ CP-nets, in order to transform the problem of computing (weak) local Condorcet winners (wLCWs) into the corresponding CSP problem $\varphi$, we need to generate a majority-optimality constraint $\varphi_X$ for each variable $X$ in $V$. For each variable $X$ and each value $x$ of $X$ (there are less than $k$ different values of $X$), we need to check each $N_j$’s conditional preference table $CPT_j(X)$ and compare with other $k - 1$ values of $X$. The number of cp-statements in $CPT_j(X)$ is exponential in the number of parents of $X$ in the dependency graph of a $N_j$. Since we assume that node in-degree is bounded by a constant $d$, the exponential is still a constant (i.e. $k^d$) and the number of variables included in the condition entry of every cp-statement is also bounded by $d$. Thus, the running time of translation is $O(m \cdot n \cdot k \cdot (k - 1) \cdot k^d \cdot d)$. □

Theorem 5.4 (complexity) Given a collection of $n$ CP-nets $N = \{N_1, \ldots, N_n\}$, if $\forall N_j \in N$, the node in-degree is bounded by a constant, then i) determining whether an alternative is a weak local Condorcet winner is polynomial; and, ii) determining whether there exists a weak local Condorcet winner is NP-complete.

PROOF. Based on Theorem 5.2, to check whether an alternative $o$ is a local Condorcet winner we just need to check whether $o$ is a model of the corresponding extended CSP problem $\varphi$, that is, whether $o$ satisfies the optimality constraint $\varphi_X$ of each variable $X$ in $V$. Let $k$ be the maximum domain size of the variables in the agents’ CP-nets. As $\varphi_X$ is the conjunction of at most $k$ constraints, each for a possible value $x$ of $X$. For each value $x$, we will need to check every other value $x'$ of $X$ that if $\neg F_{x' \succ x}$ is True or not. Checking the truth value of $F_{x' \succ x}$ can be done by counting the element in the list of $F_{x' \succ x}$ that is evaluated to True: if there are less than $(n + 1)/2$ formulas are evaluated to True then $F_{x' \succ x}$ is evaluated to False and $\neg F_{x' \succ x}$ is True, i.e., $o$ satisfies the optimality constraint for variable $X$. Suppose there are $m$ variables and node in-degree is bounded by a constant $d$. Then there are $n$ formulas listed in the cardinality constraints and each formula is a disjunction of at most $k^d$ conjunctions of at most $d$ literals. Consequently, the running time of checking whether an alternative is a model of $\varphi$ is thus $O(m \cdot n \cdot (k - 1) \cdot k^d \cdot d)$.

As we already show that testing whether an alternative is a local Condorcet winner (i.e. verify whether an alternative is a model of $\varphi$) is polynomial, the problem of
determining whether there exist at least one local Condorcet winner is in NP. To show hardness, we reduce 3-SAT to our problem: given a 3-CNF formula \( F \), for each clause \((a \lor b \lor c) \in F\), we construct the optimality constraint: \( \geq 2 [\bar{a}, \bar{b}] \Rightarrow c \). There exists at least one alternative that satisfies this set of optimality constraints iff the original formula is satisfiable. Any satisfying assignment of the original 3-CNF formula, at lease one of \( a, b \) and \( c \) is true. If \( a \) or \( b \) are True, then the condition of the optimality constraint is not satisfied and thus the optimality constraint is satisfied. If \( a \) and \( b \) are both False, then \( c \) is True, which satisfies the optimality constraint as this is the preferred value of a majority of agents. Hence, any model of the original 3-CNF formula is an optimal assignment of the set of optimality constraints. The argument reverses: any optimal assignment is also a model.

We emphasise here that the above complexity is regarding wLCWs rather than the possible winners. As we show in Corollary 5.1 and Theorem 5.1, a wLCW is not necessarily a possible winner but a possible winner must be a wLCW. To test whether a wLCW is a possible winner, we still need to compare that wLCW to every other alternatives. Consequently, in order to find out the set of possible winners, this checking is required even when there exists only one wLCW. Conitzer et al. [43] investigate the maximum number of wLCWs and the average number of wLCWs. Their results show that with \( m \) variables, the maximum number of wLCWs is \( 2^m \), however, the average number of wLCWs is 1. Consequently, in the worst case scenario, in order to compute a set of possible winners, we still need to compare \( 2^m \) outcome to other \( 2^m - 1 \) candidates (the same as the brute-force algorithm). However, on average, it is expected that we only need to test the majority optimality of only one outcome, as the expected number of wLCWs is one.

### 5.6 Experiments

The objectives of the computer experiments are to study experimentally the running time of Algorithm \texttt{MajCP} for computing the possible Winners and the weak local Condorcet winners (wLCWs). All table and graphic outputs are included in Section 5.8 and Section 5.9 for ease of reading.
5.6. Experiments

Experiment Design

Regarding the running time of computing the possible winners, we compare the performance of the proposed MajCP approach to a Brute-force algorithm, which runs a direct election over the alternative space. The purpose of this set of experiments is to investigate how much time can be saved after the reduction of the search space (i.e., only testing the majority optimality of the wLCWs), compared to searching in the entire alternative space.

Regarding the running time of computing the set of wLCWs, we are interested in comparing the proposed (CSP-based) SAT-based MajCP approach to an existing algorithm discussed in Xia et al. [148], we call it Partition method in the rest of this section. From a technique point of view, the principle of Partition method is to first partition the domain variables into a sequence of sets that are compatible with every agent’s CP-net structure, and then construct the collective induced graph following the ordering of the sets and with the only possible partial assignment to the upper set variables. This method works well when the structures of the agents’ CP-nets are similar, while when it is not the case, not much reduction can be made.

In these experiments, we tested a large number of scenarios varying the agents’ preferences, the number of variables and the number of agents. The CP-nets are acyclic, i.e., there is no cycle in the dependency graph. Regarding the CP-net generator, the reader is referred to the experiment section (Section 3.4) in Chapter 3.

Basically, the experiments can be divided into two parts: those with binary-valued CP-nets, and those with multi-valued CP-nets. We first conduct the experiment with binary CP-nets with the domain size 2 for every variable, and we vary the number of agents from 2 to 17. The number of parents of a variable in the agents’ CP-nets is bounded by 6 (maximum node-in degree in the dependency graph is 6). For computing the set of possible winners, we vary the number of variables from 2 to 15 and for each number of variable, we randomly generate 500 examples of agents’ CP-nets. We test the average running time for computing the set of possible winners with Brute-force and MajCP methods, respectively. After that, we further run 2000 random experiments with 2 to 30 variables to computer the set of wLCWs. We compare the running time of Partition and MajCP methods, respectively, and we further provide a detailed analysis of the distributions of the number of wLCWs under different problem setting.

We subsequently extend the aforementioned experiments to multi-valued CP-nets scen-
ario. The number of agents are from 3 to 9. Similarly, we first conduct 500 runs of experiments to compare the running time of Brute-force and MajCP methods to compute the set of possible winners with 2 to 10, then we conduct 2000 runs to compare the running time of Partition and MajCP methods to compute the set of wLCWs with 2 to 12 variables.

The computer tests are not comprehensive, but they provide insight into the performance of the proposed approach. Finally notice that when we filter out from those weak local Condorcet winners the alternatives that are necessarily dominated by some other alternative, we use the individual dominance testing techniques introduced in Chapter 3 for both binary-valued and multi-valued CP-nets.

**Experiment Results**

**Test Results for binary-valued CP-nets**

Regarding the running time comparison of the Brute-force algorithm and the proposed MajCP approach for computing the set of possible winners, the log-scale plot in Figure 5.5 shows the average execution time of the Brute-force algorithm and the proposed MajCP approach in the case of 5 agents and 17 agents, respectively. It demonstrates that the proposed MajCP approach is much more efficient than the Brute-force algorithm. In general, for large numbers of variables, it offers several orders of magnitude improvement in performance over the Brute-force algorithm both for 5 agents and 17 agents. For instance, when there are 8 variables, the execution time of MajCP is reduced by more than two orders of magnitude as compared to Brute-force algorithm in both the cases of 5 agents and 17 agents.

We further test the execution time of Brute-force algorithm in several cases with 10 variables, 5 and 17 agents, respectively. The results show that which shows that the execution time of the Brute-force algorithm is on average more than 5000 seconds (resp. 10000 seconds). On the other hand, the proposed MajCP approach can produce the majority winners in about 3 seconds in the case of 5 agents and 10 seconds in the case of 17 agents. According to the experimental data, when the number of variable is large (e.g., 15 variables) and with a large number of agents (17 agents), on average, the MajCP approach can still produce the results in a reasonable time of less than 360 seconds.
Remind that the whole process of computing the possible winner includes two steps: we firstly compute a set of weak local Condorcet winners (wLCWs), also called hyper-cube Condorcet winners when all the variables are all binary, by computing the models of the corresponding CSP (or SAT). Then, we need to test the majority-optimality of each wLCW and filter out those that are majority-dominated by some other outcome.

Regarding the computation of wLCWs (the first step) with binary CP-nets, Figure 5.6 compares the running time of Partition and MajCP algorithms with 5 and 17 agents, and Figure 5.7 further shows the running time of MajCP with four different number of agents 5, 9, 13 and 17. Note that when there exist wLCWs (1 or more), the proposed MajCP approach still need to test the majority-optimality of the wLCWs by comparing each wLCW to all other alternatives. However, when there are no wLCWs, the proposed approach can return the result quickly by only solving the corresponding SAT problem. From Figure 5.7 we can observe that compared to Partition algorithm, MajCP can compute the set of wLCWs in much less time. When the number of variables is larger than 10, it offers more than three orders of magnitude time reduction compared to the Partition methods. Even with large number of variables and agents, MajCP approach can compute the set of wLCWs in a reasonable time. For instance, according to the experimental data, with 17 agents and 30 variables, the proposed approach returns the set of wLCWs within 10 seconds. Consequently, if there does not exist any wLCWs (and thus possible winners), the proposed MajCP can return the failure quickly.

Moreover, we provide a detailed analysis for the number of wLCWs in these experiments. Figure 5.8 and Figure 5.9 show the probability when there exists no wLCWs and when there exists a unique wLCW for the given agents’ preferences in those experiments respectively. It can be clearly seen that as the number of variables increases, the probability that there exists no wLCWs is increasing while the probability that there exists a unique wLCW is decreasing. Table 5.1 further counts the average number of wLCWs in these experiments as compared with the exponential outcome space. Note that Table 5.1 only counts the cases where there exist at least one wLCWs. This is because we are most curious about when there exists wLCWs, on average how many alternatives we need to test the majority optimality in practice. Conitzer et al. [43] ran similar simulations to analyse the average number of hypercube Condorcet winners with binary CP-nets. However, the average number of hypercube Condorcet winners they count is for general cases, including also the cases where there exists no hypercube Condorcet winner. From the aforementioned experiments, we can observe that,
the proposed approach significantly cuts the search space by only testing the majority optimality of the set of wLCWs. Even when the number of variable is 30, on average, there are less than 2 wLCWs that needed to be tested.

Conitzer et al. [43] further consider the input of arbitrary consistent CP-nets whose variables are all binary. They go further in studying the properties of local Condorcet winners and analyse the worst-case and expected numbers of Hypercube Condorcet Winners under various assumptions. Notice that in [43], they ran similar simulations to analyse the probability that there exists at least one hypercube Condorcet winners and the average number of hypercube Condorcet winners. The results here is slightly different from [43] because the simulation here are only on acyclic CP-nets and each variable has no more than 6 parents. Notice that the simulation here also contains the part of filtering out those majority-dominated wLCWs, which involves computational complex dominance testing in CP-nets [24].

**Test Results for multi-valued CP-nets**

With multi-valued CP-nets, the average running time comparison of **Brute-force** and **MajCP** algorithm for computing possible winners are shown in Figure 5.10. Similar to the case with binary CP-nets, the proposed **MajCP** approach is much more efficient than the **Brute-force** algorithm in the case with multi-valued CP-nets. In general, for large numbers of variables, it offers several orders of magnitude improvement in performance over the **Brute-force** algorithm both for 3 agents and 9 agents. In the case of 9 agents, when the number of variables is larger than 6, using **Brute-force** method is too computationally costly and **Brute-force** method is not applicable. It can be clearly seen that with multi-valued CP-nets, even when the number of variable is large (e.g., 10 variables) and with a large number of agents (9 agents), on average, the **MajCP** approach can still compute the set of possible winners in a reasonable time of less than 1300 seconds.

Regarding computing the set of wLCWs, Figure 5.6 compares the average running time of **Partition** and **MajCP** algorithms in the cases with 3 and 9 agents, respectively. Figure 5.7 further shows the running time with four different number of agents: 3, 5, 7 and 9. The number of variables in these experiments is from 2 to 12. We can observed that the proposed **MajCP** is more efficient that the **Partition** algorithm. Even with large number of variables and agents, **MajCP** approach can compute the set of wLCWs in a very short time. For instance, with 9 agents and 12 variables, the
proposed approach returns the set of wLCWs within 100 seconds. Consequently, if there does not exist any wLCWs (and thus possible winners), the proposed \texttt{MajCP} can return the failure quickly. We also provide a detailed analysis for the number of wLCWs in the case of multi-valued CP-nets in these experiments. Figure 5.13 and Figure 5.14 shows the probability when there does not exist wLCW and when there exists a unique wLCW for the given agents’ preferences in those experiments respectively. It can be clearly seen in that the results with multi-valued CP-nets are similar to the case with binary CP-nets. As the number of variables increases, the probability that there exists no wLCWs is increasing while the probability that there exists a unique wLCW is decreasing. Table 5.2 further counts the average number of wLCWs in these experiments as compared with the average outcome space in multi-valued CP-nets. Also notice that Table 5.2 only counts the cases where there exist at least one wLCWs. We can observe that, the outcome space is increasing exponentially as the number of variable increases. However, the proposed approach significantly cuts the search space by only testing the majority optimality of the set of wLCWs. Even when the number of variable is 12, on average, there are more than $5 \times 10^5$ possible outcomes, there are only about one wLCW needed to be tested.

5.7 Summary

In Chapter 5, the problem of majority-rule-based group decision-making in combinatorial domains with CP-nets was studied. We first provided an introductory background to this topic, reviewed the related literature on computing aggregation rules from a collection of CP-nets, and then revisited an existing framework for computing a set of locally winning alternatives. This reveals some restrictions of the existing framework, and motivates the investigation into the creation of a more efficient framework for the solution of this problem. We then proceed to introduce an efficient approach to compute the set of possible winners for a collection of CP-nets. Unlike previous work, in which the agents’ preferences are required to satisfy some restrictive conditions on the dependence graph (such as the existence of a common acyclic graph to all the agents), the proposed approach allows the agents to have different preferential independence structures and also allows the CP-nets to be cyclic.

It first computes a set of locally winning alternatives, called weak local Condorcet winners (wLCWs), by reducing the problem to a CSP (Constraint satisfaction problem)
or SAT (Boolean Satisfiability Problem) for cardinality constraints. The complexity of this reduction has been shown to be polynomial. After the reduction, the set of possible winners is a subset of wLCWs, identified after the filtering out of those that are majority-dominated by some alternative).

The experiments presented in this chapter show that applying majority rules directly on the set of all possible alternatives is prohibitively time-consuming, while the proposed approach reduces the size of the search space and is computationally efficient.

Future research can extend the proposed approach to compute the winners of other aggregation rules. Another extension would be to investigate techniques to aggregate preferences that are represented by more powerful variants such as TCP-nets and UCP-nets.
### Table 5.1: Average number of wLCWs when they exist with binary CP-nets (%)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Outcome Space</th>
<th>5</th>
<th>9</th>
<th>13</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>1.054</td>
<td>1.054</td>
<td>1.057</td>
<td>1.061</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>1.094</td>
<td>1.113</td>
<td>1.108</td>
<td>1.114</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>1.14</td>
<td>1.177</td>
<td>1.162</td>
<td>1.166</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>1.17</td>
<td>1.212</td>
<td>1.228</td>
<td>1.227</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>1.23</td>
<td>1.257</td>
<td>1.279</td>
<td>1.278</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>1.265</td>
<td>1.286</td>
<td>1.309</td>
<td>1.33</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>1.29</td>
<td>1.327</td>
<td>1.347</td>
<td>1.343</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>1.315</td>
<td>1.319</td>
<td>1.365</td>
<td>1.369</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>1.304</td>
<td>1.375</td>
<td>1.396</td>
<td>1.377</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>1.276</td>
<td>1.401</td>
<td>1.376</td>
<td>1.421</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>1.324</td>
<td>1.418</td>
<td>1.406</td>
<td>1.411</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>1.399</td>
<td>1.408</td>
<td>1.499</td>
<td>1.492</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>1.375</td>
<td>1.393</td>
<td>1.466</td>
<td>1.439</td>
</tr>
<tr>
<td>15</td>
<td>32768</td>
<td>1.379</td>
<td>1.484</td>
<td>1.454</td>
<td>1.476</td>
</tr>
<tr>
<td>16</td>
<td>65536</td>
<td>1.39</td>
<td>1.484</td>
<td>1.491</td>
<td>1.494</td>
</tr>
<tr>
<td>17</td>
<td>131072</td>
<td>1.421</td>
<td>1.522</td>
<td>1.479</td>
<td>1.541</td>
</tr>
<tr>
<td>18</td>
<td>262144</td>
<td>1.416</td>
<td>1.458</td>
<td>1.549</td>
<td>1.519</td>
</tr>
<tr>
<td>19</td>
<td>524288</td>
<td>1.436</td>
<td>1.549</td>
<td>1.546</td>
<td>1.445</td>
</tr>
<tr>
<td>20</td>
<td>1048576</td>
<td>1.466</td>
<td>1.499</td>
<td>1.536</td>
<td>1.514</td>
</tr>
<tr>
<td>21</td>
<td>2097152</td>
<td>1.482</td>
<td>1.55</td>
<td>1.576</td>
<td>1.631</td>
</tr>
<tr>
<td>22</td>
<td>4194304</td>
<td>1.513</td>
<td>1.521</td>
<td>1.613</td>
<td>1.573</td>
</tr>
<tr>
<td>23</td>
<td>8388608</td>
<td>1.528</td>
<td>1.584</td>
<td>1.614</td>
<td>1.507</td>
</tr>
<tr>
<td>24</td>
<td>16777216</td>
<td>1.522</td>
<td>1.611</td>
<td>1.671</td>
<td>1.598</td>
</tr>
<tr>
<td>25</td>
<td>33554432</td>
<td>1.517</td>
<td>1.628</td>
<td>1.562</td>
<td>1.526</td>
</tr>
<tr>
<td>26</td>
<td>67108864</td>
<td>1.521</td>
<td>1.574</td>
<td>1.634</td>
<td>1.628</td>
</tr>
<tr>
<td>27</td>
<td>134217728</td>
<td>1.504</td>
<td>1.621</td>
<td>1.614</td>
<td>1.721</td>
</tr>
<tr>
<td>28</td>
<td>268435456</td>
<td>1.597</td>
<td>1.61</td>
<td>1.569</td>
<td>1.626</td>
</tr>
<tr>
<td>29</td>
<td>536870912</td>
<td>1.536</td>
<td>1.539</td>
<td>1.635</td>
<td>1.623</td>
</tr>
<tr>
<td>30</td>
<td>1073741824</td>
<td>1.545</td>
<td>1.643</td>
<td>1.613</td>
<td>1.654</td>
</tr>
</tbody>
</table>
Table 5.2: Average outcome space and average number of wLCWs when they exist with multi-value CP-nets

<table>
<thead>
<tr>
<th>Var.</th>
<th>Agents</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>wLCWs</td>
<td>os</td>
<td>wLCWs</td>
<td>os</td>
<td>wLCWs</td>
<td>os</td>
<td>wLCWs</td>
</tr>
<tr>
<td>2</td>
<td>1.014</td>
<td>8.9227</td>
<td>1.023</td>
<td>9.3693</td>
<td>1.022</td>
<td>9.043</td>
</tr>
<tr>
<td>3</td>
<td>1.046</td>
<td>26.3011</td>
<td>1.035</td>
<td>27.166</td>
<td>1.051</td>
<td>26.9015</td>
</tr>
<tr>
<td>4</td>
<td>1.042</td>
<td>80.7666</td>
<td>1.074</td>
<td>82.0844</td>
<td>1.099</td>
<td>81.9861</td>
</tr>
<tr>
<td>5</td>
<td>1.078</td>
<td>243.04</td>
<td>1.118</td>
<td>242.621</td>
<td>1.09</td>
<td>252.072</td>
</tr>
<tr>
<td>6</td>
<td>1.092</td>
<td>692.652</td>
<td>1.072</td>
<td>699.418</td>
<td>1.114</td>
<td>683.397</td>
</tr>
<tr>
<td>7</td>
<td>1.074</td>
<td>2350.51</td>
<td>1.092</td>
<td>2265.21</td>
<td>1.124</td>
<td>2127.59</td>
</tr>
<tr>
<td>8</td>
<td>1.076</td>
<td>6616.58</td>
<td>1.125</td>
<td>6313.34</td>
<td>1.139</td>
<td>6135.16</td>
</tr>
<tr>
<td>9</td>
<td>1.107</td>
<td>20801.3</td>
<td>1.117</td>
<td>20450.5</td>
<td>1.152</td>
<td>20736.2</td>
</tr>
<tr>
<td>10</td>
<td>1.112</td>
<td>56723.2</td>
<td>1.116</td>
<td>58746.5</td>
<td>1.182</td>
<td>65454.6</td>
</tr>
<tr>
<td>11</td>
<td>1.083</td>
<td>177822.</td>
<td>1.148</td>
<td>161532.</td>
<td>1.121</td>
<td>174114.</td>
</tr>
<tr>
<td>12</td>
<td>1.094</td>
<td>534938.</td>
<td>1.134</td>
<td>548862.</td>
<td>1.152</td>
<td>464208.</td>
</tr>
</tbody>
</table>
5.9 Graphic Output

Figure 5.5: Average execution time for computing the set of possible winners with binary CP-nets (Log scale plot)

Figure 5.6: Average time comparison for computing the set of wLCWs with binary CP-nets (Log scale plot)
Chapter 5. Majority-rule-based preference aggregation with CP-nets

Figure 5.7: Average execution time for computing the set of wLCWs with binary CP-nets (Log scale plot)

Figure 5.8: Percentage of cases when there are no wLCWs with binary CP-nets (%)
5.9. Graphic Output

Figure 5.9: Percentage of cases when there exists a unique wLCW with binary CP-nets (%)

Figure 5.10: Average execution time for computing the set of possible winners with multi-value CP-nets (Log scale plot)
Figure 5.11: Average time comparison for computing the set of wLCWs with multi-valued CP-nets (Log scale plot)

Figure 5.12: Average execution time for computing the set of wLCWs with Multi-valued CP-net (Log scale plot)
5.9. Graphic Output

Figure 5.13: Percentage of cases when there are no wLCWs with multi-valued CP-nets (%)

Figure 5.14: Percentage of cases when there exists a unique wLCW with multi-valued CP-nets (%)

149
Part II

Multi-issue negotiation with incomplete information
Chapter 6

Negotiation in combinatorial domains with general preferences

In the previous two chapters, we assume the agents’ preferences are given in order to compute a desirable joint outcomes. Since this assumption is not always reasonable in real world, Chapter 6 further relaxes this assumption by investigating negotiation protocol for group decision-making in combinatorial domains under incomplete information setting. When the agents’ preferences for the possible alternatives are not common knowledge, it is difficult to reach optimal agreements in bilateral or multi-lateral negotiations. Self-interested agents often end up negotiating inefficient agreements in such situations. In this chapter, a protocol for negotiation in combinatorial domains is presented in order to lead rational agents to reach optimal agreements under incomplete information setting. The proposed protocol enables the negotiating agents to identify efficient solutions using a distributed search that visits only a small subspace of the whole outcome space. Moreover, the proposed protocol is sufficiently general that it is applicable to most preference representation models in combinatorial domains, including quantitative models like utility and qualitative models like CP-nets and TCP-nets.
6.1 Introduction

Multi-issue negotiation is one of the most preferred approaches for resolving conflicts in agent society [85]. It is increasingly being used in different multi-agent domains, including trading systems, resource allocation, service level agreement negotiations and a variety of others [85, 127]. When multiple issues, such as price range, model type and the length of warranty in a car trading market, are simultaneously involved in negotiations, the agents with divergent preferences can cooperate to reach agreements that are beneficial for all of them. However, when the preferences of the participating agents are not common knowledge, they often fail to explore win-win possibilities and end up with inefficient results. Therefore, there is a need for negotiation protocols that can lead rational agents to optimal agreements.

This chapter focuses on the negotiation problems in which the space of alternatives has a combinatorial structure, such as, negotiations over multiple indivisible goods or resources (where the number of bundles an agent may obtain is exponential in the number of goods or resources). When the negotiating agents know about one another’s preferences, they can reach an efficient agreement using distributed protocols like one-step monotonic concession protocol or monotonic concession protocol [124], through which each agent searches the entire space of possible agreements.

Similar scenarios of multi-attribute decision-making with complete information have also been studied in the field of collective decision-making in combinatorial domains, i.e., voting theory and preference aggregation (for instance, [95],[114], [148]), which determines either one, some or all of the optimal alternatives from a given collection of the agents’ preferences according to a given preference aggregation rule. However, most negotiations in real life take place under incomplete information in which the agents do not have complete knowledge of the preferences of their opponents. Some protocols for negotiation over multiple indivisible resources in incomplete information scenarios have been proposed by Brams and Taylor [35]. These protocols can produce optimal agreements only for negotiating over multiple uncorrelated resources, i.e., situations in which the utility of possessing two resources is the sum of the utilities of possessing each individual resource. The scenarios they considered are similar to the negotiation problem in combinatorial domains in which the attributes are all independent. However, real-life negotiations typically involve interdependent attributes and the decision-making process tends to become much more complex. For example, a research group plans to order several PCs and the group members need to decide on a
standard group PC configuration. The decisions are not independent, because, perhaps, the preferred operating systems may depend on the given processor type. For instance, one member of the group may state that, “I prefer to choose WinXP operating system rather than Linux if an Intel processor is given.” Hence, the members cannot decide on the issues separately. However, in those situations in which there are multiple interdependent or correlated issues, these existing protocols can produce very inefficient agreements in negotiation[127].

The objective of this chapter is to design an efficient protocol for agent-based negotiation in combinatorial domains, which can lead participating agents to Pareto-optimal agreements. We consider a completely uncertain negotiation scenario in which participating agents do not have any knowledge of the preferences of the other agents; and the agents do not want to reveal their preferences for the possible alternatives during the process of negotiation. We propose a two-phase negotiation protocol POANCD (Protocol to Reach Optimal Agreement in Negotiation on Combinatorial Domains). The first phase of POANCD involves an iterative negotiation process to generate a set of initial agreements that are close to optimal. The second phase further enhances the initial agreements to be Pareto-optimal by searching for possible mutually beneficial agreements.

The proposed protocol makes a contribution distinct from those made by other existing voting protocols or aggregation mechanisms, in the sense that it works in situations in which complete information is unavailable and in a distributed manner. Moreover, the proposed protocol differs from most of the existing research in the field of utility-based negotiation. Not only can it deal with quantitative preferences, it can also work with purely qualitative preference models. It is general enough to allow for various types of preferences and representation models in combinatorial domains. The preferences can be cardinal (e.g., utilities) or ordinal preferences (preference relations). In addition, the representation models can be based on conditional preferences, for instance CP-nets [24] and its variants, e.g., TCP-nets; or it can be based on propositional logic (or possibly a fragment thereof), such as prioritised goals, distance-based goals, weighted goals or bidding languages for combinatorial auctions. Another advantage of the proposed protocol is that each agent only needs to consider a small subset of alternatives instead of the entire outcome space. It requires significantly fewer outcome comparisons, compared to exhaustive searches in most instances of negotiation.

For clarity of presentation, we attempt to describe the proposed negotiation protocol with acyclic CP-nets, i.e., the relation graph does not contain cycles. However, the
proposed protocol is independent of the preference representation models used by the agents. Hence, the proposed approach can be used to handle various preferences models and languages, given the corresponding techniques for answering dominance queries\(^1\) and outcome optimisation queries\(^2\). Notice that the negotiation process is elicitation-free: the agents are never asked to report their preferences. Therefore, for instance, if the agents have CP-nets as their preferences, they are not going to reveal their CP-nets and the structures of their CP-nets do not play any role.

### 6.2 The POANCD negotiation protocol

This section presents the proposed protocol: Protocol to reach Optimal agreement in Negotiation on Combinatorial Domains (POANCD).

---

\(^1\)A dominance query, given two alternatives \(o\) and \(o'\), asks whether \(o\) is preferred to \(o'\) with respect to the given agent’s preferences.

\(^2\)An outcome optimization query determines the set of non-dominated outcomes among the feasible outcome space with respect to the given agent’s preferences.
6.2. The POANCD negotiation protocol

6.2.1 The framework

Negotiation tree

For a negotiation problem over \( m \) attributes \( V = \{X_1, \ldots, X_m\} \), we conceptualize the assignment of the attribute values as a tree, known as the negotiation tree. Let \( k \) be the maximum size of the attribute domain: \( \forall X \in V, |D(X)| \leq k \), the negotiation tree is then a \( k \)-ary tree. The depth of the negotiation tree is \( m \) with the root being at depth 0. We assume that the set of attributes are ordered in some way \( O = X_{O_1} > \cdots > X_{O_m} \), e.g., a random order chosen by a non-bias natural device. The root node represents an empty assignment; each path to a node at depth \( \ell \) specifies a unique value assignment \( assg \in D(X_{O_1}) \times \cdots \times D(X_{O_\ell}) \) to the set of attributes \( \{X_{O_1}, \ldots, X_{O_\ell}\} \) in that order. Each node at depth \( m \) represents one possible alternative (outcome) and the path to reach that node from the root specifies the complete assignment to the set of domain attributes according to that alternative. Such a negotiation tree is shown in Figure 6.2 for a bilateral negotiation scenario over a set of three attributes \( V = \{A, B, C\} \), where the agents preferences are presented by the CP-nets depicted in Figure 6.1.

A negotiation tree is created iteratively by the negotiating agents in a distributed manner and under incomplete information setting. In each iteration, the only information that each agent obtains is the nodes in the negotiation tree which are currently available for him to make a proposal on. There is no prior information about the preferences of the opponents. Moreover, the proposals made by an agent during negotiation are invisible for its opponents. The procedure starts with a root with an empty assignment. The negotiation tree is then created in a top-down process, where in each iteration of negotiation, each agent can only choose one of the existing leaf nodes in the negotiation tree to make a proposal on. Note that the negotiating agents may make proposals on different nodes in an iteration of negotiation. Note also that each agent is not allowed to make proposals on the same node more than once during the entire negotiation process. Once a leaf node \( \eta \) at depth \( \ell (\ell < m) \) is agreed by all the participating agents (i.e., every agent has at some point made a proposal on that node during negotiation), the subtree of \( \eta \) will be expanded with every possible value assigns to the next attribute \( X_{O_{\ell+1}} \); and these children nodes will be explored by the agents and be available for them to make proposals on in the next iteration. We formally describe the following definitions of open nodes and agreement nodes in a negotiation tree.

**Definition 6.1 (Open node)** A node \( \eta \) in the negotiation tree is marked as open if and
only if it is a leaf node and it is agreed by all the negotiation agents, i.e., every agent has ever made a proposal on this node during negotiation (not necessarily in the current iteration).

Note that once a node $\eta$ at depth $\ell$ ($\ell < m$) is marked as open in the current iteration, it will be expanded with every possible value assigned to the next attribute $X_{O_{\ell+1}}$. Thus, in the next iteration, $\eta$ is not an open node any more, because it will no longer be a leaf node.

**Definition 6.2 (Agreement node)** A node $\eta$ in the negotiation tree is an agreement node if and only if it is an open node at depth $m$.

An initial agreement is reached if there is at least one agreement node in the negotiation tree. The path to reach an agreement node from the root specifies the complete assignment to the set of domain attributes that the agents have agreed on during negotiation.

**Best possible agreement (BPA)**

At each node $\eta$ of the negotiation tree, each agent $i$ has a best possible agreement on that node, denoted by $BPA_i(\eta)$, which is the optimistic outcome that agent $i$ can obtain with the values assigned to the attributes along the path from the root to $\eta$ being fixed. Let $assg = \text{PATH}(\eta)$ be the value assignment specified by the path from the root to $\eta$, then $BPA_i(\eta)$ is the best outcome among the completions of $assg(\text{Comp}(assg))$ for agent $i$. Moreover, the best possible agreement (BPA) of the root node for an agent corresponds to the optimal (best) alternative of that agent in the entire outcome space, i.e. each attribute is assigned a most preferred value according to that agent’s preference.

In the context of acyclic CP-nets, computing the best possible agreement of a node for an agent is similar to the individual outcome optimization with constraints (i.e., the values assigned to the attributes along the path from the root node to the current node being fixed) [24]. We simply need to sweep through the network from ancestors to descendants, assigning the most preferred value to each remaining attribute $X$ (i.e., the attribute that has not been assigned a value along the path) respecting to the parent context. A similar technique has also been discussed in Chapter 4.
EXAMPLE. Consider the agents’ CP-nets in Figure 6.1 and assume a path assignment for a node \( \eta \) is \( a \). According to agent 1’s CP-net in Figure 6.1(a), we consider an order over attributes from ancestors to descendants: \( O = C > A > B \). We first assign \( c \) to \( C \), because \( c \succ \bar{c} \). The next variable to be considered is \( B \), because the value of \( A \) has already been specified by the path to \( \eta \). Then we assign \( b \) to \( B \), because \( b \succ \bar{b} \) given \( A = a \) and \( C = c \). Consequently, \( BPA_1(\eta) = abc \). Similarly, for agent 2, \( BPA_2(\eta) = \bar{a} \bar{b} \bar{c} \).

6.2.2 The process of negotiation

We now present an example of a bilateral negotiation scenario using POANCD in combinatorial domains. POANCD is defined in two phases. The first phase of POANCD consists of the distributed formation of a negotiation tree by the participating agents. After the first phase, the agents will be left with a few candidates, i.e., initial agreements. In the second phase, the agents will act cooperatively to achieve Pareto-optimal agreement by exploring possible mutually beneficial alternatives.

First phase of POANCD:

This phase involves the distributed generation of the negotiation tree by the negotiating agents.

Step 1: A random device chooses an order over the domain attributes, e.g., \( O = X_{O_1} > \cdots > X_{O_m} \), such that the negotiation tree is created following that order. Initially, a root node and all its possible \( |D(X_{O_1})| \) children nodes (each branch assigns a distinct value to the attribute \( X_{O_1} \)) are created in the negotiation tree.

Step 2: Each negotiating agent makes a proposal on an existing leaf node in the negotiation tree.

After each agent makes a proposal, let \( Q \) denotes the set of nodes marked as open in the current iteration. Note that there would be at most two nodes marked as open in each iteration (\( |Q| \leq 2 \)), because we are considering a bilateral negotiation and in each iteration, each agent can only make a proposal on one node.\(^3\)

\(^3\)In a multilateral negotiation, there may be more than two nodes marked as open in an iteration of negotiation.
Chapter 6. Negotiation in combinatorial domains with general preferences

- If there exist at least one agreement nodes in the negotiation tree, collect the set of open nodes $Q$ and go to Step 3.
- Otherwise, for each $\eta \in Q$, let $\ell$ denotes the depth of $\eta$, and thus the next attribute to be considered in the subtree of $\eta$ is $X_{\ell+1}$. Expand $\eta$ with all possible $|D(X_{\ell+1})|$ children nodes in the negotiation tree; and go back to Step 2.

Remark. The negotiation process takes place under incomplete information setting and in a distributive environment. In each iteration of negotiation, each agent $i$ only knows a set of nodes (options), denoted by $\Omega_i$, on which he can make proposals, i.e., the set of leaf nodes in the negotiation tree that he has not made a proposal on during the previously rounds. For each node $\eta$ in $\Omega_i$, an agent $i$ will have a best possible agreement (BPA) of $\eta (BPA_i(\eta))$, which indicates the optimistic outcome that agent $i$ can obtain following the subtree of $\eta$. An agent can always choose to go backward as long as the BPA of the current node is less preferred than that of another node. Thus, a rational agent will always try to get the most preferred alternative among the possible options. Consequently, we consider the following strategy in negotiation: chooses a node $\eta \in \Omega_i$ whose BPA is the best among the BPAs of the nodes in $\Omega_i$. That means, there does not exist another node $\eta'$ in $\Omega_i (\eta' \neq \eta)$, such that $BPA_i(\eta') \succ_i BPA_i(\eta)$.

As an implementation, each agent can maintain a priority queue of feasible leaf nodes to make proposals on, known as the fringe. The nodes in an agent’s fringe is sorted according to the preference ordering over the BPAs of these nodes for that agent. The more preferred the $BPA_i(\eta)$ is, the higher priority the node $\eta$ is in the fringe of agent $i$. In each iteration of negotiation, for each agent $i$, the first node is removed from the fringe and agent $i$ will make a proposal in the negotiation tree on that node. If there are new nodes created in the negotiation tree (i.e., the children nodes of the open nodes created in the current iteration), each agent will add the new nodes into its fringe according to its own preference ordering on the BPAs.

Figure 6.3 shows an example of two participating agents’ fringes in a bilateral negotiation corresponding to the negotiation scenario depicted in Figure 6.2. Note that with acyclic CP-nets, the BPA is always unique at each node of the negotiation tree. With other types of preference when there may exist more than one best possible agreements, the BPA can be defined as a set of most preferred outcomes, which will be all added into the fringe. The fringe will be sorted
and in the next iteration the agent will choose the first one to make an offer on. The ordering of the nodes in the *fringe* preserves the preference ordering of the agent, i.e., if \( o > o' \) then \( o \) is given a higher priority than \( o' \) in the *fringe*. However, it is also possible that \( o \) is in front of \( o' \) in the *fringe* but they are incomparable for that agent. Finally notice that even if the BPA of a node is unique, the *fringe* can also contain incomparable outcomes, i.e., the BPA of different nodes in the tree might be incomparable. However, despite which preferred node the agent would consider in the next iteration, the protocol guarantees the same property.

**Step 3:** We refer to the set of agreement nodes as \( A \) and the set of agreements (complete assignments) corresponding to the agreement nodes in \( A \) as \( I \): \( I = \{ o^* \mid o^* = \text{PATH}(\eta^*) \text{ and } \eta^* \in A \} \), where \( \text{PATH}(\eta^*) \) denotes the value assignment to the set of domain attributes specified by the path from the root to \( \eta^* \). Note that an agreement node is also an open node, thus \( A \subseteq Q \).

- If \( Q = A \), the first phase ends and proceed to the second phase.
- Otherwise, the set \( Q \) must contain one agreement node at depth \( m \), denoted by \( \eta^* \); and one open node at depth \( \ell \) \( (\ell < m) \), denoted by \( \eta' \). That means, in the last iteration, an agent \( i \) makes a proposal on \( \eta^* \), which has been agreed by the other agent \( j \) \( (i \neq j) \) during a previous iteration of negotiation; and the other agent \( j \) makes a proposal on \( \eta' \), which has been agreed by agent \( i \) in a previous iteration of negotiation. Let \( o^* \) be the complete assignment (alternative) specified by the path from the root to the agreement node \( \eta^* \). Even though \( \eta' \) is not an agreement node, there exists a potential agreement under the subtree of \( \eta' \), because both agents have agreed on \( \eta' \). Moreover, this potential agreement can not be strictly preferred to \( o^* \) for agent \( j \) (otherwise, he would have made a proposal on \( \eta' \) before he makes a proposal \( \eta^* \)), but may be more preferred than \( o^* \) to agent \( i \). However, since there already exists an agreement \( o^* \), agent \( j \) will not make further concession in the subtree of \( \eta' \). Consequently, in order to be fair, we ask agent \( j \), who proposes \( \eta' \) in the last iteration, to give out the BPA of \( \eta' \) \( (BPA_{j}(\eta')) \). Then agent \( i \) can either choose to include \( BPA_{j}(\eta') \) in the initial agreement set: \( I = I \cup \{ BPA_{j}(\eta') \} \); or stick with the current set \( I \). Note that agent \( i \) will choose to include \( BPA_{j}(\eta') \) only if \( BPA_{j}(\eta') \succ_i o^* \). The first phase ends and we proceed to the second phase with a set of initial agreements \( I \).
Second phase of POANCD (Enhancement):

This phase is also called the *enhancement phase*, in which the participating agents will act cooperatively to explore possible mutually beneficial agreements and decide on the final agreement. We first introduce the following notations:

1. let \( \Omega_i \) denote the set of leaf nodes in the negotiation tree that agent \( i \) has not yet made a proposal on during the first phase of negotiation, i.e., the remaining nodes in agent \( i \)’s *fringe*;

2. For each initial agreement \( o^* \in I \) and each negotiating agent \( i \), let:
   - \( \Gamma_i(o^*) \) denote the set of nodes in \( \Omega_i \) (\( \Gamma_i(o^*) \in \Omega_i \)) whose BPAs are indifferent or incomparable with \( o^* \) for agent \( i \):\[ \Gamma_i(o^*) = \{ \eta \in \Omega_i \mid BPA_i(\eta) \sim_i o^* \text{ or } BPA_i(\eta) \succ_i o^* \}; \] and
   - \( \Theta_i(o^*) \) denote the corresponding set of agent \( i \)’s BPAs of the nodes in \( \Gamma_i(o^*) \):\[ \Theta_i(o^*) = \{ o \mid o = BPA_i(\eta) \text{ and } \eta \in \Gamma_i(o^*) \}. \]

Notice that for any node \( \eta \in \Omega_i \), \( BPA_i(\eta) \) can not be strictly preferred to \( o^* \) for agent \( i \); otherwise, agent \( i \) would have made a proposal at \( \eta \) before he makes a proposal on the corresponding agreement node of \( o^* \).

In this phase, for each initial agreement \( o^* \), let us define a set \( \Phi(o^*) \), which is the set of Pareto-optimal alternatives that can possibly replace \( o^* \) in \( I \). Originally, \( \Phi(o^*) = \emptyset \).

Each agent \( i \) gives out \( \Theta_i(o^*) \), then the other agent (agent \( j \)) can either choose one of the alternatives in \( \Theta_i(o^*) \) to put in \( \Phi(o^*) \) or stick with \( o^* \). Note that agent \( j \) will choose an alternative \( o' \) from \( \Theta_i(o^*) \) to put in \( \Phi(o^*) \) only if \( o' \succ_j o^* \); and the alternative \( o' \) that agent \( j \) chooses would be the best alternative among \( \Theta_i(o^*) \) for agent \( j \). If \( \Phi(o^*) \neq \emptyset \), then \( o^* \) in \( I \) will be replaced by the outcomes in \( \Phi(o^*) \).

Finally, after the second phase, the set \( I \) contains the set of final agreements. If only one element remains in \( I \), it will be selected as the final agreement. Otherwise, any one of them will be chosen randomly as the final agreement.

**An illustration**

Now, we demonstrate the execution of POANCD with an example. Assume two agents are negotiating over a set of three binary-valued attributes \( V = \{A, B, C\} \). The ne-
6.2. The POANCD negotiation protocol

gotiating agents’ preferences are depicted in Figure 6.1. As all attributes are binary, 
the negotiation tree is a binary tree. In the first phase, firstly, an ordering over the 
attributes is randomly generated, e.g., we consider the ordering $\mathcal{O} = A > B > C$ 
following which the negotiation tree will be created. Each node $\eta$ in the negotiation 
tree associates with a proposal table, in which the first row displays the proposals that 
the agents make on that node: the left (resp. right) column depicts the proposal that 
agent 1 (resp. agent 2) makes; each proposal is marked with a number that depicts the 
number of the current iteration, i.e., a proposal marked by "(p)" (resp. "<p>"), is the 
proposal that agent 1 (resp. agent 2) makes in the $p^{th}$ iteration. For explanation pur-
pose, we also attach the best possible agreements (BPA) of both agents at each node 
in the second row of the table: the left (resp. right) column depicts the BPA of that 
node of agent 1 (resp. agent 2). However, it is important to note that the information 
including the proposal that an agent makes and the BPA of a node of that agent is its 
private information and invisible for its opponent.

Figure 6.2 shows the formation of the negotiation tree in the first phase of POANCD 
and Figure 6.3 provides an illustration of the ongoing changes occurs in each agent’s 
fringe.\textsuperscript{4} For the purpose of explanation, we also provide both participating agents’ 
preference orderings over the outcome space in Figure 6.4, which are induced from 
the corresponding agents’ CP-nets in Figure 6.1. However, note that the agents do 
not need to reason about the preference relations over the entire outcome space during 
negotiation. They only need to answer a few dominance queries when adding new 
nodes into their fringes.

Initially, a root node is created in the negotiation tree. Since the first attribute to be 
considered is $A$ and $D[A] = \{a, \bar{a}\}$, two children nodes $\eta_1, \eta_2$ from the branch $a, \bar{a}$ 
are created in the negotiation tree. Each agent will create a fringe and add $\eta_1$ and $\eta_2$ 
into its fringe according to their preference orderings on the BPAs of $\eta_1$ and $\eta_2$. For 
instance, $BPA_1(\eta_1) = abc$, $BPA_1(\eta_2) = \bar{a}bc$, because $abc \succ_1 \bar{a}bc$ (see the preference 
ordering of agent 1 in Figure 6.4(a)), the order in agent 1’s fringe is $\eta_1\eta_2$. Similarly, 
the order in agent 2’s fringe is $\eta_1\eta_2$.

In the 1\textsuperscript{st} iteration, each agent will make a proposal on one of the leaf nodes in the 
negotiation tree. For instance, the first node $\eta_1$ in agent 1’s fringe is pop out and 
agent 1 makes a proposal (1) at $\eta_1$ in the negotiation tree. Similarly, agent 2 also 
makes a proposal $<1>$ on node $\eta_1$. $\eta_1$ is marked as open in this iteration and $Q = \{\eta_1\}$. 
Since the next attribute to be considered in the subtree of $\eta_1$ is $B$ and $D(B) = \{b, \bar{b}\}$.

\textsuperscript{4}The nodes depicts in red colour are the new nodes created in that iteration.
Chapter 6. Negotiation in combinatorial domains with general preferences

Two children nodes $\eta_3, \eta_4$ of $\eta_2$ from the branches $b, \bar{b}$ respectively are created in the negotiation tree. Each agent adds these two nodes into its own fringe according to its preference. For instance, $BPA_1(\eta_2) = \bar{a}bc, BPA_1(\eta_3) = abc, BPA_1(\eta_4) = \bar{a}bc$, and because $abc \succ_1 \bar{a}bc \succ_1 \bar{a}bc$, the fringe of agent 1 is $\eta_3\eta_4$ (Figure 6.3(a)); similarly the fringe of agent 2 is $\eta_4\eta_2\eta_3$ (Figure 6.3(b)).

In the $2^{nd}$ iteration, agent 1 makes a proposal $<2>$ on node $\eta_4$ and agent 2 makes a proposal on node $\eta_2$. In this iteration, there is no open node ($Q = \emptyset$); agent 1’s fringe is $\eta_3\eta_4$ and agent 2’s fringe is $\eta_2\eta_3$.

In the $3^{rd}$ iteration, agent 1 continues to make a proposal $<3>$ on node $\eta_4$ (i.e., the first node in its fringe) and agent 2 makes a proposal $<3>$ on node $\eta_2$. There is one node marked as open in the current negotiation tree: $Q = \{\eta_4\}$, thus we created two children nodes $\eta_5, \eta_6$ of $\eta_4$ from the branches $c$ and $\bar{c}$ to the next attribute $C$ respectively. Both agents add these two nodes into their fringes for consideration according to their preferences over the BPAs of the nodes: agent 1’s fringe becomes $\eta_3\eta_4\eta_2\eta_5\eta_6$ and agent 2’s fringe becomes $\eta_2\eta_3\eta_4\eta_5\eta_6$.

In the $4^{th}$ iteration, agent 1 makes a proposal $<4>$ on node $\eta_5$ and agent 2 makes a proposal $<4>$ on node $\eta_6$. No node is marked as open in this iteration of negotiation; agent 1’s fringe is $\eta_3\eta_5\eta_6$ and agent 2’s fringe is $\eta_2\eta_3\eta_5\eta_6$.

Finally, in the last iteration (the $5^{th}$ iteration), agent 1 makes a proposal $<5>$ on node $\eta_2$ and agent 2 makes a proposal $<5>$ on node $\eta_5$. As such, in this iteration, there are two nodes marked as open: $Q = \{\eta_2, \eta_5\}$. Moreover, since $\eta_5$ is an agreement node (i.e., it
6.2. The POANCD negotiation protocol

<table>
<thead>
<tr>
<th>Initial</th>
<th>(\eta_1)</th>
<th>(\eta_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st iteration</td>
<td>(\eta_3)</td>
<td>(\eta_2)</td>
</tr>
<tr>
<td>2nd iteration</td>
<td>(\eta_4)</td>
<td>(\eta_2)</td>
</tr>
<tr>
<td>3rd iteration</td>
<td>(\eta_5)</td>
<td>(\eta_2)</td>
</tr>
<tr>
<td>4th iteration</td>
<td>(\eta_2)</td>
<td>(\eta_6)</td>
</tr>
<tr>
<td>5th iteration</td>
<td>(\eta_6)</td>
<td></td>
</tr>
</tbody>
</table>

(a) Agent 1

<table>
<thead>
<tr>
<th>Initial</th>
<th>(\eta_1)</th>
<th>(\eta_2)</th>
<th>(\eta_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st iteration</td>
<td>(\eta_4)</td>
<td>(\eta_2)</td>
<td>(\eta_3)</td>
</tr>
<tr>
<td>2nd iteration</td>
<td>(\eta_2)</td>
<td>(\eta_3)</td>
<td></td>
</tr>
<tr>
<td>3rd iteration</td>
<td>(\eta_6)</td>
<td>(\eta_5)</td>
<td>(\eta_3)</td>
</tr>
<tr>
<td>4th iteration</td>
<td>(\eta_5)</td>
<td>(\eta_3)</td>
<td></td>
</tr>
<tr>
<td>5th iteration</td>
<td>(\eta_3)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) Agent 2

Figure 6.3: The fringes of the agents in the negotiation

\[
abc \succ a\overline{b}c \succ \overline{a}b\overline{c} \succ \overline{a}b\overline{c} \succ ab\overline{c} \succ ab\overline{c}
\]

(a) Agent 1

\[
\overline{a}b\overline{c} \succ \overline{a}b\overline{c} \succ \overline{a}b\overline{c} \succ ab\overline{c} \succ ab\overline{c} \succ ab\overline{c} \succ ab\overline{c}
\]

(b) Agent 2

Figure 6.4: The preference orderings of two agents

is at depth 3), Step 2 ends and we proceed to Step 3 of the first phase.

As \(A = \{\eta_5\}\) and \(A \neq Q\), \(Q\) contains one agreement node \(\eta_5\) (the path from the root to \(\eta_5\) specifies a complete assignment \(\overline{a}bc\)); and one open node \(\eta_2\) (the path from the root to \(\eta_2\) specifies a partial assignment \(\overline{a}\)). Since agent 1 is the proposer of node \(\eta_2\) in the last iteration, it gives out its BPA of \(\eta_2\): \(\text{BPA}_1(\eta_2) = \overline{a}bc\). For agent 2, as \(\overline{a}b\overline{c} \succ \overline{a}bc\) (see the preference ordering of agent 2 in Figure 6.4(b)), agent 2 will include \(\overline{a}bc\) in the set of initial agreements. Consequently, the first phase ends and we proceed to the second phase of POANCD with the set of initial agreements \(I = \{\overline{a}bc, \overline{a}bc\}\).

Originally, \(F = \{\overline{a}bc, \overline{a}bc\}\). For the initial agreement \(\overline{a}bc\), originally \(\Phi(\overline{a}bc) = \emptyset\). For agent 1, there is only one leaf node \(\eta_6\) that he has not yet made a proposal on (\(\Omega_1 = \{\eta_6\}\)) and \(\text{BPA}_1(\eta_6) = ab\overline{c}\). Since \(abc \succ ab\overline{c}\), there is no node in \(\Omega_1\) whose BPA is indifference or incomparable with \(ab\overline{c}\) for agent 1, \(\Gamma_1(ab\overline{c}) = \emptyset\) and thus \(\Theta_1(ab\overline{c}) = \emptyset\). Similarly, for agent 2 whose CP-net induced a strict total preference ordering over the
outcome space (see Figure 6.4(b)), the BPAs of the leaf nodes that he has not yet made proposals on are less preferred than the current agreement $a\overline{bc}$. Hence, $\Gamma_2(ab\overline{c}) = \emptyset$ and $\Theta_2(ab\overline{c}) = \emptyset$. Consequently, $\Phi(ab\overline{c}) = \emptyset$ and $ab\overline{c}$ will not be replaced. Similarly, for another initial agreement $\overline{a}bc$, $\Phi(\overline{a}bc) = \emptyset$ and $\overline{a}bc$ will not be replaced. Consequently, both initial agreements $abc$ and $\overline{a}bc$ are Pareto-optimal and we obtain the set of final agreements $F = \{a\overline{bc}, \overline{a}bc\}$. As $F$ contains more than one element, we randomly select one of them as the final agreement and the negotiation process ends.

### 6.3 Formal properties of POANCD

In this section, we present the formal proof of Pareto-optimality of the proposed protocol as follows. Recall that an outcome $o$ is Weakly Pareto-optimal (WPO) if there exists no other outcome $o'$ such that all agents strictly prefers $o'$ to $o$. An outcome $o$ is Pareto-optimal (PO) if there exists no other outcome $o'$ such that $o'$ is at least as preferred as $o$ or incomparable with $o$ for all agents, and strictly preferred to $o$ for at least one agent. Pareto-optimality (PO) implies weak Pareto-optimality (WPO). That is, when an alternative is PO, it is also WPO. However, the reverse does not hold: a WPO alternative is not necessarily PO.

**Theorem 6.1** The agreements reached by POANCD is Pareto-optimal.

In order to proof this theorem, we first need to proof the following proposition.

**Proposition 6.1** The initial agreements reached in the first phase of POANCD is weakly Pareto-optimal.

Proof of Proposition 6.1 Assume first that an initial agreement $o^*$ on node $\eta^*$ that the agents reach in the first phase of POANCD is not weakly Pareto-optimal, then there exists another alternative $o'$, such that for any agent $i$: $o' \succ_i o^*$. We assume $\eta'$ is a leaf node whose path assignment is identical to $o'$. For any agent $i$, $BPA_i(\eta') \succeq_i o'$ and thus $BPA_i(\eta') \succ_i o^*$. Then both agents will make the proposals on $\eta'$ or the nodes in the subtree of $\eta'$ before they make proposals on $\eta^*$; and $\eta^*$ will not be an agreement node, contradicting the face that $o^*$ is an initial agreements from the first phase of negotiation. \(\Box\)
6.4 Experiments

Proof of Theorem 6.1 From Proposition 6.1 we know that the initial agreements the agents reach in the first phase of negotiation is weakly Pareto-optimal. In the second phase, the agents are acting cooperatively to reach Pareto-optimal agreements by replacing each inefficient initial agreement \( o^* \) with a set of Pareto-optimal alternatives \( \Phi(o^*) \), such that every alternative \( o' \in \Phi(o^*) \) is indifferent or incomparable with \( o^* \) for one agent \( i \), and is more preferred than \( o^* \) for the other agent \( j \). Moreover, after an agent \( i \) gives out the set \( \Theta_i(o^*) \), if the other agent \( j \) would like to replace \( o^* \) in \( I \), the alternative he chooses from \( \Theta_i(o^*) \) will be the best one among \( \Theta_i(o^*) \). Consequently, the final agreement reached by \textsc{POANCD} is Pareto-optimal. □

6.4 Experiments

Limited experiments have been carried out in the scenario of bilateral negotiation, i.e., two agents. The objectives of the computer experiments are to give some insight about the feasibility and computational efficiency of the proposed \textsc{POANCD} protocol to bilateral negotiation in combinatorial domains with respect to (i) the number of attributes \( s_{\text{attr}} \) and domain size \( s_{ds} \) that can be efficiently handled in practice; (ii) the corresponding outcome space \( s_{\text{os}} \), the average number of different outcomes (alternatives) \( s_{\text{out}} \) that each agent needs to consider during the entire negotiation process and the average number of nodes created in the negotiation tree \( s_{tn} \); (iii) the average number of dominance queries (outcome comparisons) \( s_{dq} \) that each agent needs to answer during the entire negotiation process; (iv) the average number of iterations \( s_{iter} \) that the first phase of the negotiation process involves; and (v) the average execution time \( s_{time} \) of the entire negotiation process. All table and graphic outputs are included in Section 6.6 and Section 6.7 for ease of reading.

Experiment Design

In these experiments, we use CP-nets to represent the agents’ preferences in negotiation. We first consider a simple CP-net structure, i.e., poly-tree structured CP-nets; and we restrict the maximum in-degree of each node in the generated CP-nets to 5. Then we further test the performance of the proposed approach with less constrained CP-nets, i.e., arbitrary acyclic CP-nets. Basically the experiments can be divided into two parts: those with binary-valued CP-nets, and those on multi-valued CP-nets. We
first conduct the experiment with binary CP-nets with the domain size 2 for every variable and then extend the experiment for multi-valued CP-nets with the maximum domain size be 5. For each number of variables, we generate random examples of the agents’ CP-nets in both the cases of binary and multi-valued CP-nets. The preference networks are generated randomly by varying the number of attributes, the structure of the network and the preferences for the attributes. For the random CP-net generator, the reader is referred to Section 3.4 of Chapter 3.

In the negotiation process, we use the individual outcome optimization technique in acyclic CP-nets introduced by Boutilier et al. [24] and the heuristic approach to answer dominance query in CP-nets introduced in Chapter 3 of this thesis.

**Experiment Results**

**Tests Results for binary-valued CP-nets**

We first conduct the experiments for binary-valued CP-nets, in which the number of attributes $s_{\text{attr}}$ is varying from 2 to 100 in the case of poly-tree structured CP-nets and from 2 to 30 in the case of arbitrary acyclic CP-nets. For each number of attributes we run 1000 rounds of experiments by randomly generating two negotiating agents’ preferences.

Table 6.1 shows the experimental result in binary-valued poly-tree structured CP-nets. The average number of outcomes $s_{\text{out}}$, the average number of dominance queries $s_{\text{dq}}$, the average number of nodes created in the negotiation tree $s_{\text{tn}}$, the average number of iterations $s_{\text{iter}}$ and the average execution time $s_{\text{time}}$ are increasing as the number of attributes increases. However, we can observe that the proposed negotiation protocol POANCD can efficiently handle large number of attributes in negotiation with binary CP-nets.

Figure 6.5 provides a vision of the increase of the running time of POANCD protocol as the number of variable increases. Even with 100 variables, the negotiation process finishes in about 30 seconds. Compared to the huge outcome space, by using the proposed protocol POANCD, the average number of alternatives $s_{\text{out}}$ that each agent needs to consider is significantly reduced (comparing the second and third column of Table 6.1). When the number of attributes is large (e.g., 100), on average, the number
of dominance queries \( s_{dq} \) that each agent needs to answer during the entire negotiation process is only about 400 and the first phase of negotiation finishes in less than 170 rounds. Moreover, according to the experiment data, when the number of attributes is 100, on average, the entire negotiation process ends in about 30 seconds.

We further test the performance of POANCD protocol with arbitrary acyclic CP-nets. Table 6.3 and Figure 6.7 display these experimental results. It can be seen that even when the structure of the CP-nets is less constrained, the proposed POANCD protocol can still efficiently handle large number of variables. According to the experimental data, although with 30 attributes (in which case the maximum domain size is \( 2^{30} \)), on average, each agent only needs to consider less than 50 alternatives and answer less than 150 dominance queries during the whole negotiation process; the first phase of POANCD finishes in about 50 iterations; and the negotiation process ends in about 70 seconds.

**Tests Results for multi-valued CP-nets**

We extend the experiment for multi-value CP-nets (see Table 6.4). For multi-valued CP-nets, we restrict the maximum domain size to 5. We vary the number of attributes \( s_{attr} \) from 2 to 25 in poly-tree structured CP-nets and from 2 to 15 in arbitrary acyclic CP-nets. For each number of attributes we run 500 rounds of experiments.

Similar to the scenario with binary-valued poly-tree structured CP-nets, \( s_{out}, s_{dq}, s_{iter} \) and \( s_{time} \) are increasing as the number of attributes increases in multi-valued arbitrary acyclic CP-nets. However, although with 25 attributes (in which case the maximum domain size is \( 5^{25} \)), on average, each agent only needs to consider less than 100 alternatives and answer less than 300 dominance queries during the whole negotiation process; the first phase of POANCD finishes in about 50 iterations. Figure 6.6 provides a vision of the increase of the running time of POANCD protocol as the number of variable increases. Moreover, according to the experimental data, on average the entire negotiation process ends in about 300 seconds with 25 attributes. Note that in multi-value CP-nets, answering dominance queries is much more complex than that in binary-valued CP-nets.

Regarding the results with arbitrary acyclic CP-nets, Table 6.5 and Figure 6.8 display these experimental results. It can be seen that even when the structure of the CP-nets is less constrained, the proposed POANCD protocol can still efficiently handle large
number of variables. According to the experimental data, although with 15 attributes (in which case the maximum domain size is $5^{15}$), on average, each agent only needs to consider less than 100 alternatives and answer about 500 dominance queries during the whole negotiation process; the first phase of POANCD finishes in about 50 iterations; and the negotiation process ends within 1000 seconds.

From the experiment, we can conclude that the proposed POANCD protocol is computationally efficient. It allows preferences structures that are quite large and complex to be executed in reasonable time.

### 6.5 Summary

In Chapter 6, the problem of negotiation in combinatorial domains was studied. An efficient distributed negotiation protocol, POANCD, is proposed for negotiation in combinatorial domains when the agents do not know one another’s preferences and they do not want to reveal their preferences for the possible alternatives during the process of negotiation. It has been theoretically shown that POANCD leads rational agents to Pareto-optimal agreements. Experimental results have also been presented to show the significant reduction in search efforts and the number of dominance queries each participating agent needs to answer by using the proposed protocol.

A major advantage of POANCD is its extensibility to multilateral negotiation. In this chapter, POANCD is presented for bilateral negotiation, but extension to multilateral negotiation can be done with minor modifications.

However, the negotiation scenarios with cyclic or inconsistent preferences need to be explored further, because there may be more than one best possible agreement (BPA) of a node in a negotiation tree. At the current stage, POANCD is presented for bilateral negotiation, we are planning to extend POANCD for multi-agent negotiation. Furthermore, the fairness issue of the negotiated outcome using the proposed protocol is also an important aspect of future research.
## 6.6 Table Output

Table 6.1: Negotiations with binary-valued poly-tree structured CP-nets

<table>
<thead>
<tr>
<th>$s_{attr}$</th>
<th>$s_{os}$</th>
<th>$s_{out}$</th>
<th>$s_{tn}$</th>
<th>$s_{dg}$</th>
<th>$s_{iter}$</th>
<th>$s_{time}$ (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>2.8012</td>
<td>5.0248</td>
<td>5.3738</td>
<td>3.3799</td>
<td>0.009921</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>6.1712</td>
<td>10.648</td>
<td>15.477</td>
<td>7.1184</td>
<td>0.041856</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>9.4667</td>
<td>16.164</td>
<td>25.734</td>
<td>10.697</td>
<td>0.10318</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>12.843</td>
<td>21.748</td>
<td>36.111</td>
<td>14.271</td>
<td>0.19559</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>15.791</td>
<td>26.987</td>
<td>44.843</td>
<td>17.57</td>
<td>0.3019</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>18.873</td>
<td>32.241</td>
<td>53.385</td>
<td>20.826</td>
<td>0.4387</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>21.882</td>
<td>37.555</td>
<td>61.628</td>
<td>24.24</td>
<td>0.57818</td>
</tr>
<tr>
<td>16</td>
<td>65536</td>
<td>24.711</td>
<td>42.768</td>
<td>69.44</td>
<td>27.447</td>
<td>0.72933</td>
</tr>
<tr>
<td>18</td>
<td>$2^{18}$</td>
<td>28.038</td>
<td>48.292</td>
<td>78.219</td>
<td>30.981</td>
<td>0.92436</td>
</tr>
<tr>
<td>20</td>
<td>$2^{20}$</td>
<td>31.109</td>
<td>53.58</td>
<td>86.127</td>
<td>34.287</td>
<td>1.0903</td>
</tr>
<tr>
<td>22</td>
<td>$2^{22}$</td>
<td>33.86</td>
<td>58.931</td>
<td>93.958</td>
<td>37.58</td>
<td>1.3691</td>
</tr>
<tr>
<td>24</td>
<td>$2^{24}$</td>
<td>36.932</td>
<td>64.201</td>
<td>101.96</td>
<td>40.925</td>
<td>1.5377</td>
</tr>
<tr>
<td>26</td>
<td>$2^{26}$</td>
<td>40.316</td>
<td>69.586</td>
<td>110.48</td>
<td>44.317</td>
<td>1.8327</td>
</tr>
<tr>
<td>28</td>
<td>$2^{28}$</td>
<td>43.276</td>
<td>74.995</td>
<td>118.38</td>
<td>47.737</td>
<td>2.2912</td>
</tr>
<tr>
<td>30</td>
<td>$2^{30}$</td>
<td>46.203</td>
<td>80.213</td>
<td>126.42</td>
<td>50.943</td>
<td>2.5922</td>
</tr>
<tr>
<td>32</td>
<td>$2^{32}$</td>
<td>49.057</td>
<td>85.461</td>
<td>134.54</td>
<td>54.262</td>
<td>2.8783</td>
</tr>
<tr>
<td>34</td>
<td>$2^{34}$</td>
<td>52.265</td>
<td>90.958</td>
<td>142.59</td>
<td>57.713</td>
<td>3.1143</td>
</tr>
<tr>
<td>36</td>
<td>$2^{36}$</td>
<td>55.373</td>
<td>96.323</td>
<td>150.24</td>
<td>61.095</td>
<td>3.2389</td>
</tr>
<tr>
<td>38</td>
<td>$2^{38}$</td>
<td>58.478</td>
<td>101.34</td>
<td>158.09</td>
<td>64.178</td>
<td>3.8087</td>
</tr>
<tr>
<td>40</td>
<td>$2^{40}$</td>
<td>61.779</td>
<td>106.57</td>
<td>165.81</td>
<td>67.465</td>
<td>3.8234</td>
</tr>
<tr>
<td>42</td>
<td>$2^{42}$</td>
<td>65.303</td>
<td>112.14</td>
<td>175.01</td>
<td>70.899</td>
<td>4.9553</td>
</tr>
<tr>
<td>44</td>
<td>$2^{44}$</td>
<td>67.988</td>
<td>117.35</td>
<td>181.46</td>
<td>74.193</td>
<td>5.5119</td>
</tr>
<tr>
<td>46</td>
<td>$2^{46}$</td>
<td>70.743</td>
<td>122.77</td>
<td>190.23</td>
<td>77.623</td>
<td>5.0844</td>
</tr>
<tr>
<td>48</td>
<td>$2^{48}$</td>
<td>74.789</td>
<td>128.15</td>
<td>199.53</td>
<td>81.015</td>
<td>5.1842</td>
</tr>
<tr>
<td>50</td>
<td>$2^{50}$</td>
<td>77.523</td>
<td>133.49</td>
<td>207.22</td>
<td>84.335</td>
<td>5.6333</td>
</tr>
<tr>
<td>52</td>
<td>$2^{52}$</td>
<td>79.745</td>
<td>138.7</td>
<td>214.11</td>
<td>87.528</td>
<td>6.7984</td>
</tr>
<tr>
<td>54</td>
<td>$2^{54}$</td>
<td>82.14</td>
<td>144.21</td>
<td>221.93</td>
<td>91.04</td>
<td>7.467</td>
</tr>
<tr>
<td>56</td>
<td>$2^{56}$</td>
<td>86.05</td>
<td>149.49</td>
<td>230.71</td>
<td>94.349</td>
<td>8.3339</td>
</tr>
<tr>
<td>58</td>
<td>$2^{58}$</td>
<td>88.965</td>
<td>155.06</td>
<td>238.98</td>
<td>97.896</td>
<td>8.3945</td>
</tr>
<tr>
<td>60</td>
<td>$2^{60}$</td>
<td>93.37</td>
<td>159.88</td>
<td>247.29</td>
<td>100.78</td>
<td>8.5507</td>
</tr>
<tr>
<td>62</td>
<td>$2^{62}$</td>
<td>95.733</td>
<td>165.38</td>
<td>255.1</td>
<td>104.23</td>
<td>9.5266</td>
</tr>
<tr>
<td>64</td>
<td>$2^{64}$</td>
<td>98.995</td>
<td>170.66</td>
<td>263.23</td>
<td>107.51</td>
<td>10.886</td>
</tr>
<tr>
<td>66</td>
<td>$2^{66}$</td>
<td>102.59</td>
<td>176.14</td>
<td>272.19</td>
<td>111.06</td>
<td>10.959</td>
</tr>
<tr>
<td>68</td>
<td>$2^{68}$</td>
<td>105.97</td>
<td>181.49</td>
<td>280.38</td>
<td>114.38</td>
<td>11.379</td>
</tr>
<tr>
<td>70</td>
<td>$2^{70}$</td>
<td>108.45</td>
<td>186.76</td>
<td>287.93</td>
<td>117.62</td>
<td>13.386</td>
</tr>
<tr>
<td>72</td>
<td>$2^{72}$</td>
<td>111.37</td>
<td>192.19</td>
<td>295.85</td>
<td>121.11</td>
<td>15.947</td>
</tr>
<tr>
<td>74</td>
<td>$2^{74}$</td>
<td>114.53</td>
<td>197.8</td>
<td>304.68</td>
<td>124.58</td>
<td>16.474</td>
</tr>
</tbody>
</table>
Table: Continued with Table 6.1

<table>
<thead>
<tr>
<th>$s_{attr}$</th>
<th>$s_{os}$</th>
<th>$s_{out}$</th>
<th>$s_{inf}$</th>
<th>$s_{dq}$</th>
<th>$s_{iter}$</th>
<th>$s_{time}$ (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>$2^{16}$</td>
<td>117.4</td>
<td>202.98</td>
<td>312.2</td>
<td>127.79</td>
<td>17.139</td>
</tr>
<tr>
<td>78</td>
<td>$2^{18}$</td>
<td>120.67</td>
<td>208.16</td>
<td>320.48</td>
<td>131.07</td>
<td>18.757</td>
</tr>
<tr>
<td>80</td>
<td>$2^{20}$</td>
<td>124.47</td>
<td>213.46</td>
<td>328.98</td>
<td>134.37</td>
<td>19.35</td>
</tr>
<tr>
<td>82</td>
<td>$2^{22}$</td>
<td>126.4</td>
<td>218.71</td>
<td>335.71</td>
<td>137.6</td>
<td>20.209</td>
</tr>
<tr>
<td>84</td>
<td>$2^{24}$</td>
<td>129.5</td>
<td>224.05</td>
<td>343.86</td>
<td>140.99</td>
<td>20.343</td>
</tr>
<tr>
<td>86</td>
<td>$2^{26}$</td>
<td>134.17</td>
<td>229.51</td>
<td>353.91</td>
<td>144.48</td>
<td>20.564</td>
</tr>
<tr>
<td>88</td>
<td>$2^{28}$</td>
<td>136.93</td>
<td>234.72</td>
<td>361.86</td>
<td>147.67</td>
<td>22.214</td>
</tr>
<tr>
<td>90</td>
<td>$2^{30}$</td>
<td>141.75</td>
<td>240.25</td>
<td>371.67</td>
<td>151.25</td>
<td>25.969</td>
</tr>
<tr>
<td>92</td>
<td>$2^{32}$</td>
<td>144.09</td>
<td>245.4</td>
<td>378.38</td>
<td>154.39</td>
<td>28.592</td>
</tr>
<tr>
<td>94</td>
<td>$2^{34}$</td>
<td>144.81</td>
<td>250.67</td>
<td>384.02</td>
<td>157.57</td>
<td>28.181</td>
</tr>
<tr>
<td>96</td>
<td>$2^{36}$</td>
<td>147.18</td>
<td>256.19</td>
<td>391.85</td>
<td>161.1</td>
<td>28.893</td>
</tr>
<tr>
<td>98</td>
<td>$2^{38}$</td>
<td>150.93</td>
<td>261.43</td>
<td>400.42</td>
<td>164.4</td>
<td>29.502</td>
</tr>
<tr>
<td>100</td>
<td>$2^{40}$</td>
<td>155.3</td>
<td>266.98</td>
<td>409.97</td>
<td>167.91</td>
<td>30.39</td>
</tr>
</tbody>
</table>
Table 6.3: Negotiations with binary-valued arbitrary acyclic CP-nets

<table>
<thead>
<tr>
<th>$s_{atrr}$</th>
<th>$s_{os}$</th>
<th>$s_{out}$</th>
<th>$s_{tn}$</th>
<th>$s_{dg}$</th>
<th>$s_{iter}$</th>
<th>$s_{time}$ (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>2.7619</td>
<td>4.9365</td>
<td>5.2183</td>
<td>3.2381</td>
<td>0.0081845</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>4.5238</td>
<td>7.746</td>
<td>10.02</td>
<td>5.1508</td>
<td>0.019345</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>6.1905</td>
<td>10.492</td>
<td>15.353</td>
<td>6.9286</td>
<td>0.037202</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>8.0754</td>
<td>14.</td>
<td>21.897</td>
<td>9.3254</td>
<td>0.06932</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>9.5319</td>
<td>16.466</td>
<td>26.775</td>
<td>10.973</td>
<td>0.12545</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>11.356</td>
<td>19.796</td>
<td>33.455</td>
<td>13.123</td>
<td>0.20387</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>13.737</td>
<td>24.067</td>
<td>43.35</td>
<td>16.125</td>
<td>0.31952</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>15.087</td>
<td>26.256</td>
<td>45.821</td>
<td>17.465</td>
<td>0.42664</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>16.332</td>
<td>28.323</td>
<td>49.566</td>
<td>18.704</td>
<td>0.5598</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>17.805</td>
<td>30.623</td>
<td>52.955</td>
<td>20.186</td>
<td>0.65082</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>19.587</td>
<td>34.515</td>
<td>59.97</td>
<td>22.731</td>
<td>0.81512</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>21.554</td>
<td>36.731</td>
<td>65.422</td>
<td>24.186</td>
<td>1.0743</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>22.497</td>
<td>38.659</td>
<td>66.802</td>
<td>25.317</td>
<td>1.2987</td>
</tr>
<tr>
<td>15</td>
<td>32768</td>
<td>24.632</td>
<td>41.916</td>
<td>73.814</td>
<td>27.234</td>
<td>1.705</td>
</tr>
<tr>
<td>16</td>
<td>65536</td>
<td>25.784</td>
<td>44.383</td>
<td>76.955</td>
<td>29.03</td>
<td>2.2333</td>
</tr>
<tr>
<td>17</td>
<td>$2^{17}$</td>
<td>26.964</td>
<td>46.228</td>
<td>79.772</td>
<td>29.94</td>
<td>2.785</td>
</tr>
<tr>
<td>18</td>
<td>$2^{18}$</td>
<td>28.036</td>
<td>50.168</td>
<td>85.772</td>
<td>32.587</td>
<td>3.3679</td>
</tr>
<tr>
<td>19</td>
<td>$2^{19}$</td>
<td>29.443</td>
<td>52.347</td>
<td>89.183</td>
<td>34.066</td>
<td>4.0926</td>
</tr>
<tr>
<td>20</td>
<td>$2^{20}$</td>
<td>31.281</td>
<td>55.377</td>
<td>93.53</td>
<td>35.934</td>
<td>5.7391</td>
</tr>
<tr>
<td>21</td>
<td>$2^{21}$</td>
<td>32.31</td>
<td>58.729</td>
<td>104.04</td>
<td>38.224</td>
<td>8.0917</td>
</tr>
<tr>
<td>22</td>
<td>$2^{22}$</td>
<td>33.723</td>
<td>60.087</td>
<td>108.12</td>
<td>39.717</td>
<td>10.836</td>
</tr>
<tr>
<td>23</td>
<td>$2^{23}$</td>
<td>36.212</td>
<td>63.746</td>
<td>109.91</td>
<td>42.384</td>
<td>13.86</td>
</tr>
<tr>
<td>24</td>
<td>$2^{24}$</td>
<td>36.388</td>
<td>65.913</td>
<td>113.55</td>
<td>45.58</td>
<td>18.96</td>
</tr>
<tr>
<td>25</td>
<td>$2^{25}$</td>
<td>38.966</td>
<td>69.812</td>
<td>116.27</td>
<td>46.163</td>
<td>23.378</td>
</tr>
<tr>
<td>26</td>
<td>$2^{26}$</td>
<td>41.176</td>
<td>71.014</td>
<td>118.98</td>
<td>47.783</td>
<td>29.323</td>
</tr>
<tr>
<td>27</td>
<td>$2^{27}$</td>
<td>40.882</td>
<td>73.484</td>
<td>120.16</td>
<td>49.12</td>
<td>40.782</td>
</tr>
<tr>
<td>28</td>
<td>$2^{28}$</td>
<td>43.695</td>
<td>76.672</td>
<td>129.1</td>
<td>49.489</td>
<td>49.4</td>
</tr>
<tr>
<td>29</td>
<td>$2^{29}$</td>
<td>44.176</td>
<td>78.684</td>
<td>130.2</td>
<td>50.654</td>
<td>58.902</td>
</tr>
<tr>
<td>30</td>
<td>$2^{30}$</td>
<td>46.231</td>
<td>81.601</td>
<td>134.36</td>
<td>52.443</td>
<td>70.164</td>
</tr>
</tbody>
</table>
Chapter 6. Negotiation in combinatorial domains with general preferences

Table 6.4: Negotiations with multi-valued poly-tree structured CP-nets

<table>
<thead>
<tr>
<th>$s_{attr}$</th>
<th>$s_{os}$</th>
<th>$s_{out}$</th>
<th>$s_{tn}$</th>
<th>$s_{dq}$</th>
<th>$s_{iter}$</th>
<th>$s_{time}$ (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>12.47939262</td>
<td>7.3807</td>
<td>8.4035</td>
<td>17.403</td>
<td>4.1627</td>
<td>0.048991</td>
</tr>
<tr>
<td>3</td>
<td>42.94360087</td>
<td>11.559</td>
<td>13.753</td>
<td>33.428</td>
<td>6.6725</td>
<td>0.15958</td>
</tr>
<tr>
<td>4</td>
<td>154.5791757</td>
<td>16.228</td>
<td>19.599</td>
<td>52.625</td>
<td>9.6746</td>
<td>0.39894</td>
</tr>
<tr>
<td>5</td>
<td>558.2581345</td>
<td>20.493</td>
<td>25.247</td>
<td>76.774</td>
<td>12.108</td>
<td>1.1010</td>
</tr>
<tr>
<td>6</td>
<td>1947.969631</td>
<td>24.181</td>
<td>30.089</td>
<td>83.935</td>
<td>14.753</td>
<td>1.3252</td>
</tr>
<tr>
<td>7</td>
<td>6581.193059</td>
<td>28.053</td>
<td>35.432</td>
<td>112.04</td>
<td>17.124</td>
<td>2.2241</td>
</tr>
<tr>
<td>8</td>
<td>23463.33623</td>
<td>32.048</td>
<td>40.202</td>
<td>115.02</td>
<td>19.349</td>
<td>3.7402</td>
</tr>
<tr>
<td>9</td>
<td>72767.44902</td>
<td>33.941</td>
<td>42.718</td>
<td>113.9</td>
<td>20.607</td>
<td>5.6372</td>
</tr>
<tr>
<td>10</td>
<td>286879.6941</td>
<td>37.722</td>
<td>47.705</td>
<td>126.15</td>
<td>22.761</td>
<td>7.0969</td>
</tr>
<tr>
<td>11</td>
<td>1.02351833 × 10^6</td>
<td>41.33</td>
<td>53.141</td>
<td>142.29</td>
<td>25.51</td>
<td>8.1478</td>
</tr>
<tr>
<td>12</td>
<td>3.210566417 × 10^6</td>
<td>43.291</td>
<td>55.983</td>
<td>143.87</td>
<td>26.704</td>
<td>8.3398</td>
</tr>
<tr>
<td>13</td>
<td>1.18469194 × 10^7</td>
<td>47.503</td>
<td>60.946</td>
<td>155.75</td>
<td>29.07</td>
<td>9.8992</td>
</tr>
<tr>
<td>14</td>
<td>4.448715022 × 10^9</td>
<td>51.153</td>
<td>65.114</td>
<td>163.52</td>
<td>30.751</td>
<td>16.745</td>
</tr>
<tr>
<td>15</td>
<td>1.469719257 × 10^9</td>
<td>55.265</td>
<td>70.6</td>
<td>181.65</td>
<td>33.538</td>
<td>29.942</td>
</tr>
<tr>
<td>16</td>
<td>4.549522199 × 10^8</td>
<td>57.087</td>
<td>73.527</td>
<td>176.9</td>
<td>34.756</td>
<td>34.772</td>
</tr>
<tr>
<td>17</td>
<td>2.100779134 × 10^9</td>
<td>61.001</td>
<td>78.58</td>
<td>186.07</td>
<td>37.066</td>
<td>54.217</td>
</tr>
<tr>
<td>18</td>
<td>6.939483369 × 10^9</td>
<td>65.282</td>
<td>83.58</td>
<td>199.54</td>
<td>39.378</td>
<td>64.226</td>
</tr>
<tr>
<td>19</td>
<td>2.354505507 × 10^10</td>
<td>67.986</td>
<td>88.753</td>
<td>225.7</td>
<td>41.817</td>
<td>49.585</td>
</tr>
<tr>
<td>20</td>
<td>8.214259723 × 10^10</td>
<td>71.377</td>
<td>92.221</td>
<td>221.47</td>
<td>43.317</td>
<td>87.323</td>
</tr>
<tr>
<td>21</td>
<td>1.960313278 × 10^11</td>
<td>73.698</td>
<td>95.524</td>
<td>224.7</td>
<td>44.609</td>
<td>86.618</td>
</tr>
<tr>
<td>22</td>
<td>1.021646598 × 10^11</td>
<td>76.977</td>
<td>99.43</td>
<td>230.31</td>
<td>46.638</td>
<td>179.87</td>
</tr>
<tr>
<td>23</td>
<td>3.264095784 × 10^12</td>
<td>79.583</td>
<td>103.68</td>
<td>237.1</td>
<td>48.557</td>
<td>71.738</td>
</tr>
<tr>
<td>24</td>
<td>1.092536175 × 10^13</td>
<td>81.664</td>
<td>107.51</td>
<td>242.2</td>
<td>50.026</td>
<td>104.63</td>
</tr>
<tr>
<td>25</td>
<td>3.789763682 × 10^13</td>
<td>86.913</td>
<td>113.37</td>
<td>269.6</td>
<td>52.967</td>
<td>310.79</td>
</tr>
</tbody>
</table>

Table 6.5: Negotiations with multi-valued arbitrary acyclic CP-nets

<table>
<thead>
<tr>
<th>$s_{attr}$</th>
<th>$s_{os}$</th>
<th>$s_{out}$</th>
<th>$s_{tn}$</th>
<th>$s_{dq}$</th>
<th>$s_{iter}$</th>
<th>$s_{time}$ (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>12.2852349</td>
<td>7.3993</td>
<td>8.4446</td>
<td>17.384</td>
<td>4.1965</td>
<td>0.04116</td>
</tr>
<tr>
<td>3</td>
<td>42.11577181</td>
<td>12.091</td>
<td>14.438</td>
<td>37.357</td>
<td>7.2686</td>
<td>0.16847</td>
</tr>
<tr>
<td>4</td>
<td>146.488255</td>
<td>16.571</td>
<td>20.272</td>
<td>58.596</td>
<td>10.007</td>
<td>0.43931</td>
</tr>
<tr>
<td>5</td>
<td>519.0956376</td>
<td>22.887</td>
<td>28.208</td>
<td>97.227</td>
<td>13.933</td>
<td>1.6605</td>
</tr>
<tr>
<td>6</td>
<td>1831.486532</td>
<td>30.843</td>
<td>38.679</td>
<td>136.45</td>
<td>19.325</td>
<td>5.2835</td>
</tr>
<tr>
<td>7</td>
<td>6476.507589</td>
<td>36.069</td>
<td>45.55</td>
<td>185.56</td>
<td>22.825</td>
<td>11.646</td>
</tr>
<tr>
<td>8</td>
<td>22644.80944</td>
<td>40.345</td>
<td>53.702</td>
<td>230.15</td>
<td>26.752</td>
<td>22.598</td>
</tr>
<tr>
<td>9</td>
<td>62970.05556</td>
<td>43.847</td>
<td>62.028</td>
<td>282.97</td>
<td>28.028</td>
<td>56.132</td>
</tr>
<tr>
<td>10</td>
<td>457786.</td>
<td>48.153</td>
<td>69.75</td>
<td>322.79</td>
<td>30.25</td>
<td>84.411</td>
</tr>
<tr>
<td>11</td>
<td>1.234767778 × 10^6</td>
<td>50.75</td>
<td>72.5</td>
<td>352.67</td>
<td>33.806</td>
<td>120.16</td>
</tr>
<tr>
<td>12</td>
<td>2.289017143 × 10^6</td>
<td>58.443</td>
<td>76.343</td>
<td>380.6</td>
<td>37.257</td>
<td>185.06</td>
</tr>
<tr>
<td>13</td>
<td>1.969958286 × 10^7</td>
<td>65.386</td>
<td>83.3</td>
<td>402.9</td>
<td>42.057</td>
<td>309.5</td>
</tr>
<tr>
<td>14</td>
<td>3.206740655 × 10^7</td>
<td>72.273</td>
<td>91.727</td>
<td>471.7</td>
<td>45.182</td>
<td>409.04</td>
</tr>
<tr>
<td>15</td>
<td>1.934737127 × 10^8</td>
<td>81.455</td>
<td>107.15</td>
<td>521.7</td>
<td>52.818</td>
<td>909.04</td>
</tr>
</tbody>
</table>
6.7 Graphic Output

Figure 6.5: Average execution time with binary-valued poly-tree structured CP-nets (Log scale plot)

Figure 6.6: Average execution time with multi-valued poly-tree structured CP-nets (Log scale plot)
Chapter 6. Negotiation in combinatorial domains with general preferences

Figure 6.7: Average execution time with binary-valued arbitrary acyclic CP-nets (Log scale plot)

Figure 6.8: Average execution time with multi-valued arbitrary acyclic CP-nets (Log scale plot)
Chapter 7

Utility-based mediated negotiation over multiple continuous issues

Up to this point, this thesis so far has been dealing with the group decision-making problem only in discrete issue domains. Since in many real world decision-making problems, some issues involved are continuous, one may easily question that how we can make the joint decision in such setting. In order to address this issue, in Chapter 7, we subsequently investigate the problem of negotiation over multiple continuous issues. In this chapter, utility is used as the formal model to represent agents’ preferences and a mediated negotiation approach is considered for making a final collective decision. We propose a cooperative mediated negotiation framework to support the agents in reaching an efficient and fair agreement in multi-party multi-issue negotiation under incomplete information. The proposed approach improves the agents’ utility values from the status quo, by searching for the mutually preferred outcomes that minimise the difference between the agents’ utility gains, leading to fair agreements. Two case studies are presented in order to illustrate the capacity of the proposed approach in real world negotiation scenarios. Experimental evaluations of the overall performance of the proposed approach are also provided.
7.1 Introduction

This chapter focuses on the utility-based negotiation problem over continuous issues and investigates the technique that can achieve more mutually beneficial agreements in negotiation under incomplete information setting. In real world scenarios, the situation commonly arises in which multiple issues are simultaneously involved in a negotiation. Typical examples include the price, quality attributes and delivery time in a supply contract; or the transferability of water rights, degrees of environmental protection and new infrastructure development in a water usage policy; or the response time, levels of security and traceability in a service level agreement.

In many situations the negotiation agents are able to make tradeoffs, i.e. they may increase their utility values by lowering their requirements on some negotiation issues that are not so important for them, while demanding more on other more important issues, thus leading to an agreement that is more mutually beneficial than would be the case in a negotiation about a single issue [57]. For instance, consider a situation involving the selling of a laptop computer. A seller agent prefers a higher margin on the laptop price and is willing to provide an extended warranty, technical support and free delivery service which are of a lower cost for her. A buyer agent is also interested in such a package because she values highly the extended warranty and prefers free delivery. Furthermore, the after-sales service is very important to her. Such situations in which all the parties are in a more favourable position, are normally called “win-win” situations.

A typical objective of win-win negotiations is to find a Pareto-efficient or Pareto-optimal outcome [109]. The simplest way in which to increase the efficiency of the outcome can be for the agents to share their preference information during a search for possible joint gains in the negotiation process. However, such information when disclosed may be exploited by opponents to the disadvantage of the disclosing agent [123, 131]. Therefore, most negotiations in real life take place under incomplete information in which there is no common knowledge about the preferences of the negotiating parties. However, empirical evidence suggests that in such an incomplete information setting, the negotiating agents often fail to elicit possible joint gains and end up with inefficient results [96].

In order to increase the efficiency of a negotiated outcome, we employ a trusted third party, the mediator, to support the negotiating agents in searching for possible joint
gains, while protecting them from unnecessary disclosure of information to their opponents.

Fairness is an important property [22, 34]; it is desirable that the resulting agreement satisfies a certain fairness criterion. In mediation, the initial tentative agreement is important, because it defines the beginning of the enhancement process and affects the fairness of the final agreement to be reached through mediation. If the initial agreement is unfair, it could be very difficult to improve fairness. Therefore, if the initial agreement is not fair, the final outcome is not guaranteed to be fair. Ehtamo et al. [56] present several methods to choose the initial tentative agreement (called the reference point in their paper). Vo and Padgham [139] present a procedure to discover the fair initial tentative agreement by maximising the utility values of the agents, meanwhile reducing the difference between the agents’ valuations of each attribute in the initial tentative agreement.

This chapter focuses on the enhancement process and we assume that a fair initial agreement is given. Subsequently, the proposed approach fairly improves the initial agreement and finally achieves Pareto-optimality. Moreover, the mediator does not know any utility value of the agents during the whole negotiation process. Consequently, the notion of fairness we investigate in this chapter is based on the utility gains of the agents.

Given the initial agreement \( o \), which can be viewed as the status quo (i.e., a natural reference point of the negotiation, or an agreement of pre-negotiation), the utility values of the agents on \( o \) can be considered as the status quo utility (i.e., the utility obtained if one decides not to negotiate with other players). Consequently, the fairness of the agreed outcome can be measured through the egalitarian social welfare, the minimum utility gain of the negotiating agents [79]. An outcome is an egalitarian solution if it maximises the egalitarian social welfare among all possible outcomes.

In this chapter, the mediator does not profit in any way from the outcome of the negotiation, nor favour some agents over others in the negotiations. The mediator fairly improves the utility values of the negotiation agents from the status quo, by iteratively creating a series of fair joint gains for the agents, until an efficient outcome is achieved. From a technical point of view, our framework employs the method of improving directions. At each stage of a negotiation process, the mediator chooses a fair compromise direction that improves the utility values of all of the negotiating agents and determines a new, mutually preferred tentative agreement along this direction for the negotiating
agents to take into consideration. This process continues, until an efficient outcome is achieved. For the direction-choosing problem, we first introduce a simple approach, called Equal Directional Derivative (EDD), for bilateral negotiations.

However, since an EDD direction does not necessarily exist for negotiations with more than two agents, we then present a non-trivial generalisation of the EDD approach for multilateral negotiations. The proposed approach involves the solution to a mathematical programming problem, called DMP (Deviation Minimisation Problem). The objective of our proposed approach is to find more efficient outcomes, which improve the utility values of all negotiating agents while minimising the difference between the agents’ utility gains, leading to fair agreements. It should be emphasized that we do not assume any utility functions to be explicitly known by the mediator. We only assume that the information needed to execute the method, e.g. the utility gradients at the tentative agreements, is available. As such, the proposed approach strongly supports automated negotiation in real life, where, for instance, it may be impractical to elicit the complete preference of an agent, or the preference of an agent may be changing dynamically during the process of negotiation.

We apply the proposed approach to two real world negotiation problems; negotiation over fishing rights in the Grand Banks area with logarithmic utility functions, and negotiation between water users in California. These two case studies illustrate the capacity of the proposed approach. We also experimentally evaluate the overall performance of the proposed approach with quadratic utility functions. The experimental results demonstrate that the proposed approach not only guarantees Pareto-efficiency, but also produces outcomes that are close to the fair egalitarian solutions, in both bilateral and multilateral negotiations.

In the rest of this chapter, these ideas will be presented in detail. Firstly, in Section 7.2, background knowledge on decision and negotiation theory is provided, and some related work is discussed in Section 7.3. Then, in Section 7.4, we present the technical details of the proposed approach. We subsequently present two case studies of real world negotiation problems, and then provide an overall performance evaluation of the proposed approach compared with other existing methods in Section 7.5. Finally, in Section 7.6, we present the concluding remarks.
7.2 Background and related work

In this chapter, we consider utility function as a formal model to represent an individual agent’s preference. Moreover, we will assume in this chapter that each agent $i$’s preference can be described by a differentiable strictly concave utility function $u_i : O \rightarrow \mathbb{R}$, and the feasible outcome set $O$ is closed and convex. Thus, an agent $i$ prefers an outcome $o$ to another outcome $o'$ if $u_i(o) > u_i(o')$, and is indifferent between $o$ and $o'$ if $u_i(o) = u_i(o')$. Without loss of generality, we normalize the utility value of an agent as a real number in the interval $[0, 1]$.

Indifference Curve

In utility-based multi-issue negotiation, an individual agent may need to make concessions as to achieve an agreement among all the involved agents. When we analysis an agent’s behaviour or strategy to achieve higher individual utility (or satisfaction) in group decision-making, it is important to know the definition of indifference curves.

In the $m$-dimensional outcome space, for a particular outcome point $o$, each agent has an indifference curve go though that point. It is formed by a set of alternatives that has the same utility as the current point. That is, utility is a constant along an indifference curve. For instance, if we have only two issues price and delivery time, Figure 7.1(a) shows an example of an indifference curve of a seller agent. For instance, it is indif-
ference for him if he sells the product at $400 but need to delivery the product with in 10 working days and if he charge a much lower price, say $200, but he can deliver in more than 20 working days. Notice that if there are more than two issues, it will become an indifference surface. A graph of indifference curves for an individual agent associated with different utility levels is called an indifference map (see for example Figure 7.1(b)). Points yielding different utility levels are each associated with distinct indifference curves and is like a contour line on a topographical map. Each point on the curve represents the same elevation.

Indifference curves with higher utility are preferred to the ones with lower utility level. Indifference curves are downward sloping. If the satisfaction of one attribute is reduced, then an agent must require more of the other attribute to compensate for the loss. At each point on an indifference curve, there is a tangent line that is tangent to that indifference curve. Its slope is the derivative and represents the rate at which agents are willing to substitute one attribute for the other. Finally notice that indifference curves do not cross (intersect with each other), since this would imply a contradiction.

Utility gradient

Given a particular outcome point o and an agent’s utility function $u$, the utility gradient, denoted by $\nabla u(o)$ of that agent at $o$ represents the most preferred direction in which that agent would like to move to make immediate gains from $o$. That is, the utility gradient points in the direction that gives the agent the most/maximum utility gains. Its magnitude ($\|\nabla u_i(o)\|$), i.e., the norm of the gradient vector, represents the greatest rate of increase of the agent’s utility. Geometrically, the utility gradient $\nabla u(o)$ is perpendicular to the tangent line $TL$, which is tangent to the indifference curve $IC$ that go through the outcome point $o$ (see Figure 7.2).

7.3 Related research

Some earlier research works have been dealing with the joint gains seeking problems using mediation approaches [56, 58, 89, 122]. They mainly deal with bilateral negotiations. In particular, Raiffa [122] develops an approach for making moves and finding joint gains in bilateral negotiations. Each step of improvement increases one party’s utility but keeps another unchanged. However, their techniques mainly focus on the
7.3. Related research

situations when negotiation parties have complete information about each other’s preference. Ehtamo et al. [58] present a mediation framework for making trade-offs based on the method of improving directions in bilateral negotiation under incomplete information. Given any interior point in the continuous outcome space, their method generates a jointly improving direction to move along, which is the direction that bisects two agents’ gradient directions. However, as it only considers the gradient directions of the agents and ignores their gradient magnitudes, the approach leaves the fairness issue between the agents’ utility gains largely unanswered.

Another mediation-based negotiation model with incomplete information is given by Lai et al. [89]. In their approach, the mediator conducts a Pareto-efficient enhancement for a proposal in each negotiation period. In each period, agents are required to propose a preferred solution on a base line. Then based on this proposal, the mediator applies a computationally efficient query process to find a point that approaches Pareto-efficiency and mutually better for both agents. However, their approach does not address the fairness issue between the utility gains of two negotiating agents. Their algorithm for finding the Pareto-efficient point (referred to as $\epsilon$-Satisfying approach in the following sections) is of high efficiency in two-attribute cases, however, the computational complexity grows rapidly as the number of attributes increases. Furthermore, it may attempt to compute values that do not necessarily exist. For an example, the reader is referred to the case discussed in the experiment in Section 4.1. In such situations, the Pareto-frontier search direction cannot be determined and the algorithm is incapable of proceeding.
Multilateral negotiation is even more complex and challenging. Ehtamo et al. [57] extend their previous work [58] and study the joint gains seeking problems in multi-party setting. Ehtamo et al. emphasize that the direction-choosing problem in multilateral negotiation is a non-trivial generalization of the bisecting approach in bilateral negotiation. Because the bisecting direction does not necessarily exist and there is no straightforward way to calculate a direction with some specific desirable properties in multi-agent cases. Thus, Ehtamo et al. take a mathematical programming approach to search for a compromise direction that approximates the characteristic of the bisecting direction. However, their proposed approach also ignores the negotiating agents’ gradient magnitudes, and thus, it shares the same limitation as the bisecting approach in [58].

The proposed approached is motivated by [57, 58] and the contribution of this chapter is two-fold. First, we address the fairness issues between the negotiating agents both in bilateral and multilateral negotiation. The proposed mediation approach not only guarantees Pareto-efficiency, but also provides fair agreements that minimize the difference between two negotiating agents’ utility gains. Second, since the problem of finding the exact Pareto-frontier is NP-hard [54], we present an approximate algorithm for approaching the Pareto-frontier. As such, the mediation process converges to a ‘final solution’ in a reasonable number of iterations; and the final solution is guaranteed to be either Pareto-efficient, or sufficiently close to the Pareto-frontier.

7.4 The proposed mediation framework

We assume a set of $n$ agents $N = \{1, \ldots, n\}$ who are acting cooperatively to solve a problem involving $m$ continuous attributes $X = \{X_1, \ldots, X_m\}$. The agents are honest and each agent $i$’s preference over the set of attributes $X$ can be described by a differentiable strictly concave utility function $u_i : O \to \mathbb{R}$, and the feasible outcome set $O$ is closed and convex.\footnote{The rationale of this assumption on the properties of the agents’ utility functions and the outcome space will be discussed in Section 7.4.1} We focus on the enhancement process and assume that the agents are currently at an initial tentative agreement (i.e. the status quo) $o = (x_1, x_2, \ldots, x_m)$. Finally, we assume that a non-biased mediator is available. From a technical point of view, our framework is based on the method of improving directions. By taking the negotiating agents’ utility values as its objectives the mediator itself faces a multi-
The proposed mediation framework

The issues faced by the mediator are: (i) the mediator requires access to the gradient information about the negotiating agents’ utility functions at the tentative agreement, and (ii) it needs to make efficient and fair tradeoffs between different agents’ utility values. In each iteration of negotiation, the mediator mainly performs the following two tasks:

**Task 1:** choosing an efficient and fair compromise direction;

**Task 2:** determining a new tentative agreement.

The first task only requires the agents to submit their utility gradients of the current tentative outcome point to the mediator, they do not have to reveal their utility functions completely. Based on the agents’ gradient information, the mediator chooses a compromise direction that fairly improves all the negotiating agents’ utility values. After the compromise direction is determined, the mediator then starts to search for a feasible, jointly preferred outcome along this direction as a new tentative agreement. In Task 2, each agent only needs to tell whether it would like to move to the new tentative agreement proposed by the mediator or not. Technical details of performing these two tasks are presented in Section 7.4.2 and Section 7.4.3 respectively.

### 7.4.1 Method of improving direction: mathematical description and analysis

Before going into technical details, we first present the mathematical description and analysis of the proposed mediation technique as follows.

**Feasible direction**

Given a tentative agreement \( o = (x_1, ..., x_m) (o \in O) \), a vector \( \tilde{d} (||\tilde{d}|| = 1) \) is called a feasible direction at point \( o \), if there exists \( \delta > 0 \) such that \( o + \lambda \cdot \tilde{d} \in O \) for all \( \lambda \in (0, \delta) \). The set containing all feasible directions at point \( o \) is denoted by \( F(o) \) and

---

2At a particular point in the outcome space, there are multiple utility functions that have the same gradient. Therefore, giving out the gradient does not mean to reveal the agent’s utility function completely. Moreover, the proposed approach does not require the agents to reveal their gradients for every possible outcome point.
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

called the set of feasible directions. Formally,

$$F(o) = \{ \tilde{d} \mid \exists \delta > 0 \text{ s.t. } \forall \lambda \in (0, \delta), \ o + \lambda \cdot \tilde{d} \in O \& \& \|\tilde{d}\| = 1 \} \quad (7.1)$$

Improving direction

Given a tentative agreement \( o = (x_1, ..., x_m) \ (o \in O) \), a vector \( \tilde{d} \) is called an \textit{improving direction} for an agent \( i \) at point \( o \) if agent \( i \)'s utility increases in the direction of \( \tilde{d} \). The set of \textit{improving directions} for agent \( i \) at point \( o \) is thus defined by:

$$D_i(o) = \{ \tilde{d} \mid \exists \delta > 0, \ \text{s.t. } \forall \lambda \in (0, \delta), \ u_i \left( o + \lambda \cdot \tilde{d} \right) - u_i(o) > 0 \& \& \|\tilde{d}\| = 1 \} \quad (7.2)$$

Given a direction \( \tilde{d} \), when the moving distance \( \lambda \rightarrow 0 \), we obtain the following \textit{marginal utility gains} of an agent \( i \) in the direction of \( \tilde{d} \): \( \lim_{\lambda \rightarrow 0} \frac{u_i(o + \lambda \cdot \tilde{d}) - u_i(o)}{\lambda} \). It represents the instantaneous rate of increase of \( u_i \), moving through point \( o \), in the direction of \( \tilde{d} \). Following the differentiability of \( u_i \), then \( \lim_{\lambda \rightarrow 0} \frac{u_i(o + \lambda \cdot \tilde{d}) - u_i(o)}{\lambda} = \nabla u_i(o) \cdot \tilde{d} \) and thus:

$$D_i(o) = \{ \tilde{d} \mid \nabla u_i(o) \cdot \tilde{d} > 0 \& \& \|\tilde{d}\| = 1 \} \quad (7.3)$$

Jointly improving direction

Accordingly, given a tentative agreement \( o = (x_1, ..., x_m) \ (o \in O) \), a vector \( \tilde{d} \) is called a \textit{jointly improving direction} if it is improving the utility values of all agents. The set of jointly improving directions is thus defined by the intersection of all the agents’ improving direction sets:

$$D(o) = \bigcap_{i=1}^{n} D_i(o) \quad (7.4)$$

Based on Equation (7.3) and (7.4), the utility gradient of an agent \( i \) at point \( o \) suffices to determine \( D_i(o) \), and thus \( D(o) \) is known if all the negotiating agents’ gradients at \( o \) are known.
7.4. The proposed mediation framework

Figure 7.3: Jointly improving directions set

Pareto-efficiency

Given an outcome $o \in O$, we say $o$ is Pareto-efficient if and only if there exist no other point $o' \in O$, $o' \neq o$, such that $\forall i \in \{1, \ldots, n\}$, $u_i(o') \geq u_i(o)$ and the inequality is strict for at least one agent. Based on the classical assumption of the convexity of the outcome space and the strict concavity of the agents’ utility function, $D(o)$ is associated with the efficiency of an outcome in the following way according to the work by Ehtamo et al. [57].

Theorem 7.1 [57] An outcome $o^* \in O$ is Pareto-efficient if and only if $D(o^*) \cap F(o^*) = \emptyset$.

The above theorem is applicable in both bilateral and multilateral cases. Note that the assumption of the convexity of the outcome space and the strict concavity of the agents’ utility function is to rule out possibilities of local optima agreements. Under this assumption, Pareto-efficiency is guaranteed for the negotiated outcome. Without this assumption, the proposed approach still works to fairly improve all the agents’ utility values. However, it is possible to converge to a locally optimal agreement and global Pareto-efficiency is not guaranteed.

Computing whether $D(o) \cap F(o) = \emptyset$ at an outcome point $o \in O$ is generally hard. A more straightforward way to test efficiency for the interior points $O^{ln}$ in bilateral negotiation is studied in Kersten and Noronha [81]. Gregory and Sunil first present the local properties of the opposite relationship between two negotiating agents. Given an outcome point $o \in O^{ln}$, let $\phi(o) = \arccos\left(\frac{\nabla u_1(o) \cdot \nabla u_2(o)}{\|\nabla u_1(o)\| \cdot \|\nabla u_2(o)\|}\right)$ be the angle
between two negotiating agents’ gradients at point o [81]:

- The agents are in local weak opposition if and only if their utility gradients \( \nabla u_1(o), \nabla u_2(o) \) form an acute or right angle at point \( o \): \( 0 \leq \phi(o) \leq \frac{\pi}{2} \) (See Figure 7.3(a)). In a special case when \( \phi(o) = 0 \), the two gradients point in the same direction \( \nabla u_1(o) = k\nabla u_2(o), k \in \mathbb{R}^+ \).

- The agents are in local strong opposition if and only if their utility gradients \( \nabla u_1(o), \nabla u_2(o) \) form an obtuse angle at point \( o \): \( \frac{\pi}{2} < \phi(o) < \pi \) (See Figure 7.3(b)).

- The agents are in local strict opposition at a point \( o \) if and only if the angle between their utility gradients is equal to \( \pi \): \( \phi(o) = \pi \) (See Figure 7.3(c)). In such case, the two gradients are in the opposite direction \( \nabla u_1(o) = -k\nabla u_2(o), k \in \mathbb{R}^+ \).

Then the opposite relationship between two negotiating agents can be used for testing outcome efficiency according to the work by Kersten and Noronha [81]:

**Theorem 7.2 [81]** In bilateral negotiation, an interior outcome point \( o^* \in O^{ln} \) is Pareto-efficient if and only if the agents are in local strict opposition at \( o^* \): \( \phi(o^*) = \pi \).

Using the geometric interpretation of the gradients, these efficient (local strict opposition) points may also be called tangential points, i.e., points where the two indifference curves (or indifference surfaces) tangent to each other.

Now, the mediator’s choice of a compromise direction (Task 1) can be described at a high level by a function \( T : O \rightarrow \mathbb{R}^m \) such that \( T(o) \in D(o) \cap F(o) \) and \( T(o) = \text{Null} \) whenever \( o \) is Pareto-efficient (i.e. \( D(o) \cap F(o) = \emptyset \)). We will discuss how to choose a fair compromise direction and analyse its fairness properties in both bilateral and multi-lateral cases in Section 7.4.2.

**Choice of new tentative agreement**

Once a compromise direction \( \tilde{d} = T(o) \) has been chosen, the mediator begins to search for a new feasible, jointly preferred tentative agreement along \( \tilde{d} \) which improves the
7.4. The proposed mediation framework

utility values of all agents from \( o \) (Task 2). Let \( M(o, \vec{d}), M : \mathcal{O} \times \mathbb{R}^n \rightarrow \mathcal{O} \) be the function of choosing a new tentative agreement. For each agent \( i \), let \( \delta_i (\delta_i \geq 0) \) be the largest solution that \( \forall \lambda \in [0, \delta_i], u_i(o + \lambda \vec{d}) - u_i(o) \geq 0 \). Because the agents utility functions are assumed to be strictly concave, we have \( u_i(o + \lambda \vec{d}) - u_i(o) > 0 \) for all \( \lambda \in (0, \delta_i) \). We assume that the agents are honest and acting cooperatively. As such, an agent \( i \) is willing to move from a point \( o \) to another point \( o + \lambda \vec{d} \) if and only if \( u_i(o + \lambda \vec{d}) - u_i(o) > 0 \). Let \( \vec{d} = \min \{\delta_1, \ldots, \delta_n\} \) and \( \lambda \) denotes the moving distance from \( o \) along direction \( \vec{d} \), thus, an outcome \( o + \lambda \vec{d} \) is jointly preferred to \( o \) for all agents if \( 0 < \lambda < \vec{d} \). We then can define \( M(o, \vec{d}) = o + \lambda \vec{d} \) such that \( 0 < \lambda < \vec{d} \).

Convergence of Pareto-efficiency

After a new tentative agreement is determined, the process continues as long as joint gains can be realized then, eventually, any move would make at least one agent worse off, that means, a Pareto-efficient agreement is achieved. Denoting subsequent tentative agreements by \( o_k, k = 1, 2, \ldots \), the negotiation process described above can be defined by means of \( T \) and \( M \) as: \( o_{k+1} = M(o_k, T(o_k)), k = 1, 2, \ldots \). Note that the above procedure can result in two possible situations: either a Pareto-efficient agreement \( o^* \) s.t. \( D(o^*) \cap F(o^*) = \emptyset \) is achieved in a finite number of iterations; or an infinite sequence is generated that approaches the Pareto-frontier. While the problem of finding the exact Pareto-frontier is NP-hard [54], in Section 7.4.3 we will introduce an approximation algorithm to approach the Pareto-frontier in which the users can directly determine how close the final agreement is to the Pareto-frontier.

7.4.2 Choosing a fair compromise direction

7.4.2.1 Bilateral negotiation

The compromise direction-choosing problem in bilateral negotiation has been studied in the literature. In particular, Ehtamo et al. [58] present a bisecting approach to choose the compromise direction in bilateral negotiation. Their approach is defined only on the set of interior points \( \mathcal{O}^{In} \) and bisects the angle between two negotiators’ gradient directions: \( T^{BS}(o) = \frac{\nabla u_1(o)}{\|\nabla u_1(o)\|} + \frac{\nabla u_2(o)}{\|\nabla u_2(o)\|} \). Moving along the bisecting direction, the marginal gains of two negotiating agents are always divided at the ratio of their gradient magnitudes: \( \frac{\nabla u_1(o) \cdot T^{BS}(o)}{\nabla u_2(o) \cdot T^{BS}(o)} = \frac{\|\nabla u_1(o)\|}{\|\nabla u_2(o)\|} \). As the agents’ gradi-
ent magnitudes can be very different, using the bisecting approach can result in unfair outcomes.

As the agents’ utility functions are not common knowledge, i.e., the utility value of a particular outcome of an agent is its private knowledge and not known by the mediator nor the opponents, we consider the fairness in the context of utility gains in the enhancement process. Motivated by the work by Ehtamo et al. [58], we address the fairness issue by introducing an EDD (Equal Directional Derivative) approach, aiming at minimizing the difference between the agents’ utility gains. Our approach chooses a compromise direction $T^E$ from the set of jointly improving and feasible directions, which has equal directional derivative on two negotiating agents utility values, providing equal shares of the marginal gains between the negotiating agents:

$$
\nabla u_1(o) \cdot T^E = \nabla u_2(o) \cdot T^E \quad \text{s.t.} \quad T^E \in D(o) \cap F(o) \quad (7.5)
$$

We show that for any interior points $o (\in O^{ln})$ such that $\phi(o) \neq \pi$ (i.e. $o$ is inefficient) and $\phi(o) \neq 0$ (i.e. two agents’ gradients point in different directions), it is always feasible to find a direction that produces the equal marginal gains on two agents’ utility values.

**Theorem 7.3** In bilateral negotiation, for any interior point $o (\in O^{ln})$ such that $\phi(o) \neq 0$ and $\phi(o) \neq \pi$, there exists a compromise direction $T^E \in D(o) \cap F(o)$ such that:

$$
\nabla u_1(o) \cdot T^E = \nabla u_2(o) \cdot T^E
$$

**Proof.** Let $\alpha$ be a plane (or hyperplane) defined by the two gradient vectors $\nabla u_1(o)$ and $\nabla u_2(o)$, $TL_1$ and $TL_2$ be the tangent lines$^3$ that intersect two agents’ indifference curves$^4$ at the current point $o$ (See Figure 7.3). Accordingly, when $\phi(o) \neq \pi$ and $\phi(o) \neq 0$ the set of jointly improving directions $D(o)$ is the set of directions between $TL_1$ and $TL_2$ (See the shaded area in Figure 7.3(a) and Figure 7.3(b)). For any jointly improving direction $d \in F(o) \cap D(o)$, it forms two acute angles with the gradient vectors $\nabla u_1(o)$ and $\nabla u_2(o)$, denoted by $\phi_1$ and $\phi_2$ respectively; and two angles with the tangent lines $TL_1$ and $TL_2$, denoted by $\tilde{\phi}_1$ and $\tilde{\phi}_2$ respectively.

By definition, $\nabla u_1(o)$ is perpendicular to $TL_1$ and thus $\phi_1 = \frac{\pi}{2} - \tilde{\phi}_1$; and, $\nabla u_2(o)$

---

$^3$For a m-dimensional case ($m > 2$), these would become tangent hyperplanes.

$^4$For a m-dimensional case ($m > 2$), these would become indifference surfaces.
is perpendicular to $TL_2$ and thus $\phi_2 = \frac{\pi}{2} - \tilde{\phi}_2$. Let $\theta \ (0 < \theta < \pi)$ be the angle between these two tangent lines, then $\tilde{\phi}_2 = \theta - \tilde{\phi}_1$. Let $f$ be the function of $\tilde{\phi}_1$ and $f(\tilde{\phi}_1) = \frac{\sin(\tilde{\phi}_1)}{\sin(\theta - \tilde{\phi}_1)}$, then $f$ is a continuous function over $\tilde{\phi}_1 \in (0, \theta)$ and the range of this function is $(0, +\infty)$. Based on the Intermediate Value Theorem, there must exist $\tilde{\phi}_1^* \in (0, \theta)$, such that $0 < f(\tilde{\phi}_1^*) = \frac{\sin(\tilde{\phi}_1^*)}{\sin(\theta - \tilde{\phi}_1^*)} = \frac{\|\nabla u_2(o)\|}{\|\nabla u_1(o)\|}$. Because $\frac{\sin(\tilde{\phi}_1^*)}{\sin(\theta - \tilde{\phi}_1^*)} = \frac{\sin(\tilde{\phi}_1^*)}{\cos(\tilde{\phi}_2^*)} = \frac{\|\nabla u_2(o)\|}{\|\nabla u_1(o)\|}$, and thus $\|\nabla u_1(o)\| \cdot \cos(\tilde{\phi}_1^*) = \|\nabla u_2(o)\| \cdot \cos(\tilde{\phi}_2^*)$. Let $\tilde{d}^*$ be the unit vector in plane $\alpha$ that forms two acute angles $\tilde{\phi}_1^*, \tilde{\phi}_2^*$ with two gradient vectors, and two acute angles $\tilde{\phi}_1^*, \tilde{\phi}_2^*$ with two tangent lines respectively. As $\nabla u_1(o) \cdot \tilde{d}^* = \|\nabla u_1(o)\| \cdot \cos(\tilde{\phi}_1^*)$, it also satisfies $\nabla u_1(o) \cdot \tilde{d}^* = \nabla u_2(o) \cdot \tilde{d}^*$ and the proof is completed. $\square$

Let $o \ (o \in O^{ln})$ be a tentative agreement such that $\phi(o) \neq \pi$ and $\phi(o) \neq 0$, as $T^E$ is in the plane $\alpha$ defined by the two gradient vectors, based on Coplanar Vector Theorem, $T^E$ could be considered as the sum of two vectors which are in the same or opposite direction of two gradients $\nabla u_1(o)$ and $\nabla u_2(o)$ respectively,

$$T^E = k_1 \cdot \frac{\nabla u_1(o)}{\|\nabla u_1(o)\|} + k_2 \cdot \frac{\nabla u_2(o)}{\|\nabla u_2(o)\|} \quad (k_1, k_2 \in \mathbb{R}) \quad (7.6)$$

Substituting $T^E$ in Equation (7.5) with Equation (7.6), we obtain:

$$k_1 = \frac{\|\nabla u_2(o)\| - \|\nabla u_1(o)\| \cdot \cos(\phi(o))}{\|\nabla u_1(o)\| - \|\nabla u_2(o)\| \cdot \cos(\phi(o))} \quad (7.7)$$

Let $k_1 = t \cdot \|\nabla u_2(o)\| - \|\nabla u_1(o)\| \cdot \cos(\phi(o))$, and $k_2 = t \cdot \|\nabla u_1(o)\| - \|\nabla u_2(o)\| \cdot \cos(\phi(o))$ ($t \in \mathbb{R}$). Then we can define the following collinear vector $\tilde{T}$ of $T^E$,

$$T^E = t \cdot \tilde{T} \ (t \in \mathbb{R} \text{ and } \|T^E\| = 1):$$

$$\tilde{T} = \frac{\|\nabla u_2(o)\| - \|\nabla u_1(o)\| \cdot \cos(\phi(o))}{\|\nabla u_1(o)\|} \cdot \frac{\nabla u_1(o)}{\|\nabla u_1(o)\|} + \frac{\|\nabla u_1(o)\| - \|\nabla u_2(o)\| \cdot \cos(\phi(o))}{\|\nabla u_2(o)\|} \cdot \frac{\nabla u_2(o)}{\|\nabla u_2(o)\|} \quad (7.8)$$

Now we can define the direction-choosing function for any given interior point in bilateral negotiation $T^E : O^{ln} \to \mathbb{R}^m$. 189
• if $\phi(o) = \pi$ (i.e. $o$ is Pareto-efficient),

$$T^E(o) = \text{Null}$$

• else if $\phi(o) = 0$ (i.e. two negotiating agents’ gradients point in the same direction at $o$),

$$T^E(o) = \frac{\nabla u_1(o)}{||\nabla u_1(o)||} = \frac{\nabla u_2(o)}{||\nabla u_2(o)||}$$

• else,

  – If $\bar{T} \cdot \nabla u_i(o) > 0$, $i = 1, 2$:

$$T^E(o) = \frac{\bar{T}}{||T||}$$

  – else,

$$T^E(o) = -\frac{\bar{T}}{||T||}$$

The direction-choosing function $T^E(o)$ is necessary feasible for any interior point in the outcome space and $T^E(o) = \text{Null}$ if and only if $o$ is Pareto-efficient. Consequently, the process continues as long as jointly improving directions can be realized until eventually, any move would make at least the other agent worse off. In other words, a Pareto-efficient agreement has been achieved. We notice that only the gradient information $\nabla u_1(o)$ and $\nabla u_2(o)$ at point $o$ are needed for the direction-choosing function $T^E(o)$. In bilateral negotiation, the proposed EDD approach not only guarantees Pareto-efficiency, but also minimizes the differences between two agents’ utility gains. We will show in the experiment section that our proposed EDD approach produces fairer outcomes that are closer to the egalitarian solutions compared to other existing methods.

### 7.4.2.2 Multilateral negotiation

In multilateral cases, however, the direction-choosing problem is much more complex and the direct generalizations of the bisecting and EDD approaches no longer work. Ehtamo et al. [57] study the direction-choosing problem in multilateral negotiation and present a mathematical programming approach, in which the compromise directions are chosen from the set of feasible and jointly improving directions. They mainly consider the following two heuristic optimization problems:
7.4. The proposed mediation framework

- **PMP** (Product Maximization Problem): to find a direction $T^{PMP}(o)$ that maximizes the product of its projections to the agents’ gradient directions.

$$
\max_{\vec{d}} \prod_{i=1}^{n} \frac{\nabla u_i(o)}{\|\nabla u_i(o)\|} \cdot \vec{d} \quad \text{s.t.} \quad \vec{d} \in D(o) \cap F(o)
$$

The first constraint defines $\vec{d}$ to be feasible and improving the utility values of all negotiating agents. The last constraint requires $\vec{d}$ to be in the unit sphere. The explanation of the constraints also holds in the following sections.

- **MMP** (Minimum Maximization Problem): to find a direction $T^{MMP}(o)$ that maximizes the minimum of its projections to the agents’ gradient directions.

$$
\max_{\vec{d}, z} z \quad \text{s.t.} \quad \frac{\nabla u_i(o)}{\|\nabla u_i(o)\|} \cdot \vec{d} \geq z; \quad i = 1, \ldots, n, \quad \vec{d} \in D(o) \cap F(o)
$$

As is shown in [57], PMP and MMP work in a similar fashion. However, both of them ignore the gradient magnitudes of the agents, and thus they share the same limitation as bisecting approach in bilateral negotiation.

In this chapter, we extend the EDD bilateral negotiation approach, introducing a novel approach that aims at assigning joint gains fairly among all the negotiating agents. The proposed approach involves the solution to the mathematical programming problem, called DMP (Deviation Minimization Problem), for the direction-choosing problem in multilateral negotiation. We first define the marginal gain ratio of an agent $i$ at a tentative agreement $o$, in the direction of $\vec{d}$, denoted by $MR_{i, \vec{d}}$:

$$
MR_{i, \vec{d}} = \frac{\nabla u_i(o) \cdot \vec{d}}{\sum_{j=1}^{n} \nabla u_j(o) \cdot \vec{d}}
$$

(7.9)

The marginal gain ratio, in essence, captures the marginal gain share of agent $i$ in the sum of marginal gains produced by $\vec{d}$ for all the agents, and $\sum_{i=1}^{n} MR_{i, \vec{d}} = 1$. Then we consider the following deviation minimization problem, aiming at reducing the difference between the marginal gain shares among the negotiating agents.

**DMP** (Deviation Minimization Problem): to find a direction $T^{DMP}(o)$ that minimizes
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

the deviation of the agents’ marginal gain ratios.

$$\min_d \sqrt{\frac{1}{n} \sum_{i=1}^{n} (MR_{i,d} - \frac{1}{n})^2} \quad \text{s.t. } d \in D(o) \cap F(o) \quad (7.10)$$

where $n$ denotes the number of agents and $\frac{1}{n}$ is the mean marginal gain ratio.

The objective of DMP, in essence, is to assign marginal gains equally among the agents at each period of negotiation. When there exists a direction $d \in D(o) \cap F(o)$ in which each agent shares $\frac{1}{n}$ of the sum of marginal gains: $\forall i, j \in \{1, \ldots, n\}, MR_{i,d} = MR_{j,d} = \frac{1}{n}$, it will be chosen as the compromise direction at the current point $o$ according to the principle of DMP approach. Consequently, **DMP approach is consistent with EDD approach in bilateral negotiation.**

The reason why we choose to minimize the deviation of marginal gain ratios rather than simply minimize the deviation of marginal gains of the agents is that, there may exist a direction $\vec{d}$, in which $\nabla u_i(o) \cdot \vec{d} = 0$ for all $i \in \{1, \ldots, n\}$. Such a direction would not be applicable for the agents to improve their utility values, although it could produce the smallest deviation of the agents’ marginal gains. By using the marginal gain ratio, we avoid such kind of extreme cases. Finally we notice that to determine $T_{DMP}^D(o)$ only the agents’ utility gradients have to be known at $o$.

### 7.4.3 Approaching the Pareto-frontier

In each iteration of negotiation, after the compromise direction has been chosen, the mediator searches for a new feasible, jointly preferred tentative agreement. Ehtamo et al. [57] consider the problem of determining a new tentative agreement as the maximization of the decision-makers’ utility values. In their proposed procedure, each agent is required to announce a point along the chosen compromise direction that maximizes its utility, and the mediator chooses the one among them that is closest to the current agreement. However, their approach may result in unfair outcomes and there is no stopping point when the procedure results in an infinite sequence of iterations.

In this chapter, we claim that smaller movements along the compromise direction in each iteration of negotiation results in fairer outcomes. Nevertheless, the computational complexity should also be considered, because more iterations and data transfers
7.4. The proposed mediation framework

between the mediator and the negotiating agents may be needed as \( \lambda \) reduces. Therefore, we consider \( \lambda \) as a parameter that is pre-defined based on an overall consideration of various system goals (e.g. computational complexity, the accuracy of fairness, etc.). We also provide a careful comparison of the overall performance between different length of moving distance \( \lambda \), the degree of fairness it achieves and the corresponding number of iterations during the negotiation process in the experiment section.

Moreover, when the tentative outcome point is very close to the Pareto-frontier, using a pre-defined moving distance \( \lambda \) may not guarantee that each individual agent can increase its utility, particularly when \( \lambda > \bar{\delta} \) (see Section 7.4.1, \( \bar{\delta} = \min \{ \delta_1, \ldots, \delta_2 \} \)). Consequently, we present a binary search process for approaching Pareto-frontier by iteratively decreasing \( \lambda \) with \( \lambda = \lambda / 2 \).

**Definition 7.1** Given a point \( o \) in the \( m \)-dimensional space, we call the set of possible alternatives

\[
\sigma_\tau(o) = \{ o' \mid |o' - o| \leq \tau, \ o, o' \in O \}
\]

as the \( \tau \)-space\(^5\) of point \( o \).

**Definition 7.2** A point \( o \) is a \( \tau \)-satisfying Pareto-efficient solution if one of the following two properties is satisfied:

- there does not exist any point that is mutually preferred to \( o \) for all the negotiating agents, i.e., there does not exist \( o' \in O \) and \( o' \neq o \), s.t. for all agent \( i \in \{1, \ldots, n\} \), \( u_i(o') \geq u_i(o) \) and there exist an agent \( k \in \{1, \ldots, n\} \), \( u_k(o') > u_k(o) \);

- all the points that are mutually preferred to \( o \) for all the negotiating agents, if existing, are located in the \( \tau \)-space of point \( o \), i.e., if there exists \( o' \in O \), \( o' \neq o \), \( u_i(o') \geq u_i(o) \) for all agent \( i \in \{1, \ldots, n\} \) and \( u_k(o') > u_k(o) \) for at least one agent \( k \in \{1, \ldots, n\} \), then \( o' \in \sigma_\tau(o) \).

**Lemma 7.1** When \( \tau \rightarrow 0 \), a \( \tau \)-satisfying Pareto-efficient solution is also Pareto-efficient.

\(^5\)Within a two (or three) dimensional space, the \( \tau \)-space is a circle (or a sphere) centred at \( o \) and with radius \( \tau \).
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

PROOF. According to Definition 7.1, when $\tau \to 0$, then $\sigma_\tau(o) \to \emptyset$. And according to Definition 7.2, if $\sigma_\tau(o) \to \emptyset$, there is no point that is mutually preferred to $o$ for all the negotiating agents, thus $o$ is Pareto-efficient. □

Consequently, $\tau$ can be used as a pre-defined accuracy in approaching the Pareto-frontier. That means, we can directly determine how close the final agreement is to the exact Pareto-frontier by specifying the value of $\tau$. As such, we provide a clear stopping point for the negotiation process, i.e., the process ends if the current tentative agreement under consideration is a $\tau$-satisfying Pareto-efficient solution.

Now we are describing an iteration to reach the next tentative agreement $o_{k+1}$ from the current tentative outcome point $o_k$. The mediator first collects the agents’ utility gradients $\nabla u_i(o_k)$, $i = 1, \ldots, n$ at the current tentative outcome point $o_k$:

- If $D(o_k) \cap F(o_k) = \emptyset$ (or $\phi(o_k) = \pi$ in bilateral negotiation), according to Theorem 7.1 (or Theorem 7.2 in bilateral negotiation), $o_k$ is Pareto-efficient and the negotiation process ends.

- Otherwise, the mediator chooses the compromise direction $\vec{d} = T^{DMP}(o_k)$ (or $\vec{d} = T^{E}(o_k)$ in bilateral negotiation) and asks the agents to consider the possible new tentative agreement $o_{k+1} = o_k + \lambda \cdot \vec{d}$.
7.5. Experiments

- If all the negotiating agents are willing to move, then $o_{k+1}$ is chosen to be the new tentative agreement.

- Otherwise, the mediator begins a binary search process to find out the possible mutually preferred alternative along $d$ by iteratively decreasing $\lambda$ with $\lambda = \lambda/2$. This binary search process continues, until either it finds out a point such that all agents are willing to go; or if $\lambda \leq \tau$, then the current point $o_k$ is a $\tau$-satisfying Pareto-efficient outcome and the negotiation process ends.

Given an initial tentative outcome point $o_0$, the method described above allows a finite sequence of tentative outcome points $o_1, o_2, \ldots$ to be iteratively generated. The iterative procedure stops whenever a Pareto-efficient or a $\tau$-satisfying Pareto-efficient outcome is achieved. Figure 7.4 shows this iterative procedure.

### 7.5 Experiments

This section evaluates the performance of the proposed EDD and DMP approaches in the context of the egalitarian social welfare compared with other existing methods. In bilateral negotiation, we compare the proposed EDD approach with the bisecting approach introduced in [58], and the $\epsilon$-Satisfying approach proposed by Lai et al. [89]. In multilateral negotiations, we compare the performance of the proposed DMP approach with the PMP and MMP approaches introduced by Ehtamo et al. [57]. We first study two real world negotiation problems with different type of utility functions to demonstrate the capacity of the proposed approach. Then we will provide an overall performance comparison of these approaches with quadratic utility functions. In these experiments, we use numerical optimization techniques in Mathematica 7.0 to solve the direction-choosing function in multilateral negotiation.  

In this chapter, we only focus on the enhancement process and an agent’s preference (utility function) is neither known by the mediator nor its opponents. Consequently, the notion of fairness here is defined in the context of the agents’ utility gains during the enhancement process. Recall that the agents’ utility functions are normalized into

---

6In these experiments, we use the numerical minimization function "NMinimize" and maximization function "NMaximize" in Mathematica 7.0, with 5 effective digits of accuracy and 5 effective digits of precision.
interval $[0, 1]$. Let $G_i (i \in \{1, \ldots, n\})$ denotes the utility gain of an agent $i$ from the initial outcome $o_0$ to the final agreement $o^*$, thus $G_i = u_i(o^*) - u_i(o_0) (i \in \{1, \ldots n\})$. We consider the following measurements in those experiments:

- **Egalitarian social welfare (ESW):** for a final agreement $o^* \in O$, the egalitarian social welfare, $ESW(o^*)$, is defined as the minimum utility gain among the agents: $ESW(o^*) = \min_{i \in \{1, \ldots, n\}} G_i$. An outcome $o^*$ is called an egalitarian solution if $ESW(o^*)$ is maximized among all possible outcomes (there is no other outcome $o'$ such that $ESW(o') > ESW(o^*)$). Note that the definition of egalitarian social welfare is different from the traditional meaning in social welfare [79]. The definition of $ESW$ here is defined in the context of the agents’ utility gains.

- **Deviation of the agents’ utility gain ratios (DR):** an efficient outcome is relatively fair if it produces a fair joint gain division among the negotiating agents. In multi-agent negotiation, we define the following definition, deviation of the agents’ utility gain ratios (called $DR$ in the following sections), to measure the difference between agents’ utility gains:

\[
DR = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left( \frac{G_i}{\sum_{j=1}^{n} G_j} - \frac{1}{n} \right)^2}
\]

where $n$ denotes the number of agents; $\frac{G_i}{\sum_{j=1}^{n} G_j}$ calculates the utility gain ratio of an agent $i$ and $\frac{1}{n}$ is the mean utility gain ratio. $DR$ provides a measure of the difference between the agents’ utility gains, the smaller the $DR$ is, the fairer the outcome is, or vice versa. Particularly when $DR = 0$, each agent shares $\frac{1}{n}$ of the joint gains. For the purpose of explanation, the value of $DR$ is expressed as a percentage in the following sections.

### 7.5.1 Application 1: negotiation over fishing rights

In this section, we continue with the fishing game mentioned in [58] and [57]. Assume first that two countries that harvest the same stock of fish are willing to make an agreement about the next year’s fishing rights. These two countries affect each other at time
7.5. Experiments

(a) Indifference curves of two agents

(b) $\epsilon$-Satisfying approach

Figure 7.5: Negotiation over fishing rights: indifference curves at the initial agreement

$t$ through the size of the remaining fish population $Q_t \ (x_1 + x_2 \leq Q_t)$ and they wish to maximize their overall discounted utility. Suppose that, if uninterrupted, the fish population would grow according to $Q_{t+1} = Q_t^\alpha \ (0 < \alpha < 1)$. In accordance with [58], let $o = (x_1, x_2)$ be a possible alternative, $o \in O$ and $O = \{ (x_1, x_2) \mid x_1, x_2 \geq 0 \text{ and } x_1 + x_2 \leq Q_t \}$, the utility functions of two countries are logarithmic functions as follows:

$$u_i(o) = \ln x_i + \beta_i \ln (1/2)(Q - x_1 - x_2)^\alpha$$

where $\alpha$ is the growth parameter of the fish population, $\beta_i$ is the discount factor of the country $i$ and $Q$ is the initial stock of fish. In the experiments we use the same setting as in [58], i.e. $Q = 1.250; \alpha = 0.2852; \beta_1 = 0.9; \beta_2 = 0.4$; the initial tentative agreement $o = (0.6, 0.6)$. A sample contour plot of two negotiating agents’ indifference curves at the initial agreement is depicted in Figure 7.5(a). We consider a pre-defined accuracy $\tau = 0.0001$ (we use the same accuracy for $\epsilon$-Satisfying approach, i.e. $\epsilon = 0.0001$) and moving distance $\lambda = 0.01$. Notice that the $\epsilon$-Satisfying approach is not based on the method of improving directions and does not consider moving distance.

Table 7.1 shows the results of bisecting and the proposed EDD approaches in bilateral negotiation. Both for bisecting and EDD approaches, the negotiation process finishes in 14 iterations in this example. Compared to bisecting approach, EDD approach results in a greater egalitarian social welfare ($ESW = 0.092$) than the bisecting approach ($ESW = 0.0636$). It also produces a fairer joint gain division with a much smaller deviation of the agents’ utility gain ratios ($DR = 2.70\%$) than that of bisecting approach.
Table 7.1: The results of fishing right negotiation in bilateral setting

<table>
<thead>
<tr>
<th>Approach</th>
<th>Iterations</th>
<th>ESW</th>
<th>DR (%)</th>
<th>Agreement</th>
<th>DisEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bisecting</td>
<td>14</td>
<td>0.0636</td>
<td>18.24%</td>
<td>(0.506961,0.558305)</td>
<td>0.027</td>
</tr>
<tr>
<td>ε-Satisfying</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>EDD</td>
<td>14</td>
<td>0.0920</td>
<td>2.70%</td>
<td>(0.486452,0.58133)</td>
<td>0.004</td>
</tr>
<tr>
<td>Egalitarian</td>
<td>-</td>
<td>0.0977</td>
<td>0</td>
<td>(0.489039, 0.578446)</td>
<td>-</td>
</tr>
</tbody>
</table>

Note: \( \lambda = 0.01, \tau = 0.0001; \) DisEP: Distance from the Egalitarian point.

\((DR = 18.24\%).\)

Moreover, we compute the egalitarian solution with the agents’ utility functions (see the bottom row of Table 7.1). We can see that although under incomplete information setting, the results of EDD approach are very close to that of the egalitarian solution. Column 6 of Table 7.1 further illustrates the Euclidean distances between the egalitarian solution point and the final agreements achieved by different approaches, respectively.

Note that ε-Satisfying approach is not applicable in this example. Because ε-Satisfying approach acts on the premise that given the initial agreement \( o = (x_1, x_2) \) and the indifference curve of each agent that go through this point, there is a unique point located on each agent’s indifference curve when setting one of the attribute (e.g. \( X_2 \)) equal to \( X_2 = x_2 + \epsilon \) and \( X_2 = x_2 - \epsilon \) respectively. Otherwise, the Pareto-frontier search direction could not be determined and the procedure is incapable of proceeding. In this example, given the initial agreement \((0.6, 0.6)\), either with \( x_2 = 0.6 + \epsilon \) or \( x_2 = 0.6 - \epsilon \), there are two points \((x^i_1)\) on agent 1’s indifference curve \( IC_1 \) (See the magnified graph Figure 7.5(b) with interval \( x_1 \in [0.3, 0.7] \) and \( x_2 \in [0.4, 0.7] \)). Hence, it is unable to determine the search direction for the Pareto-frontier and the algorithm is not applicable in this example.

Now assume that the number of countries increases to 3. Then the total harvest cannot exceed the stock size \( Q (x_1 + x_2 + x_3 \leq Q) \), and the catches must be nonnegative \((x_1, x_2, x_3 \geq 0)\). The utility functions of the countries are as follows:

\[
    u_i(o) = \ln x_i + \beta_i \ln(1/2)(Q - x_i - x_2 - x_3)^\alpha, \quad i = 1, 2, 3.
\]

In accordance with [57], we use the following parameter values: \( Q = 2, \alpha = 0.2852, \)
7.5. Experiments

Table 7.2: The results of fishing right negotiation in multilateral setting

<table>
<thead>
<tr>
<th>Approach</th>
<th>Iterations</th>
<th>ESW</th>
<th>DR (%)</th>
<th>Agreement</th>
<th>DisEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMP</td>
<td>11</td>
<td>0.0559</td>
<td>7.79%</td>
<td>(0.3960, 0.5265, 0.7623)</td>
<td>0.017</td>
</tr>
<tr>
<td>MMP</td>
<td>15</td>
<td>0.0558</td>
<td>7.85%</td>
<td>(0.3960, 0.5264, 0.7621)</td>
<td>0.017</td>
</tr>
<tr>
<td>DMP</td>
<td>12</td>
<td>0.0714</td>
<td>0.54%</td>
<td>(0.3849, 0.5240, 0.7768)</td>
<td>0.001</td>
</tr>
<tr>
<td>Egalitarian</td>
<td></td>
<td>0.0732</td>
<td>0</td>
<td>(0.3856, 0.5242, 0.7760)</td>
<td>-</td>
</tr>
</tbody>
</table>

Note: $\lambda = 0.01$, $\tau = 0.0001$; DisEP: Distance from the Egalitarian point.

$(\beta_1, \beta_2, \beta_3) = (0.9, 0.7, 0.5)$ and the initial agreement $o = (0.46058, 0.59217, 0.82904)$. Table 7.2 shows the results of PMP, MMP and the proposed DMP approaches with the pre-defined $\lambda = 0.01$ and $\tau = 0.0001$. In general, the negotiation process ends in similar numbers of iterations using PMP, MMP and DMP approaches. Compared to PMP and MMP approaches, DMP approach results in a greater egalitarian social welfare ($ESW = 0.714$), and produces a fairer joint gain division with a much smaller deviation of the agents’ utility gain ratios ($DR = 0.54\%$) in multilateral negotiation. Consistent with EDD approach, DMP approach also produce the results that are very close to the egalitarian solution (see the bottom of Table 7.2).

7.5.2 Application 2: negotiation between water users

In this section, we take the California water policy negotiations as another motivating example to demonstrate the capacity of the proposed mediated negotiation approach. The original simulation is developed by Adams et al. [6] as a simulation of a non-cooperative bargaining framework. We consider the same setting as presented in [6], i.e., the same utility function and data interval (see Table 7.3). However, in this section, we simulate a cooperative mediation framework for searching possible joint gains for multiple agents.

The formal game consists of three parties: agricultural water users (A), urban water users (U) and environmentalists (E); and three issues: the degree of new infrastructure development ($X_1$), the degree of transferability of water rights ($X_2$), and the degree of environmental protection ($X_3$). Each issue is normalized to a unit interval $[0, 1]$ and an possible outcome $o$ is represented by a combination of attribute values $(x_1, x_2, x_3)$. According to [6], each party $i$ has a most preferred outcome $o_i^*$, called his ideal point in the outcome space; and then the utility value of an outcome $o = (x_1, x_2, x_3)$ of
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

A negotiation party $i$ is a decreasing function of the distance between $o$ and his most preferred outcome $o_i^*$. This utility is a constant elasticity of substitution function of the following form:

$$u_i(o) = \left( \sum_{k=1}^{3} \gamma_{i,k} \left[ \theta_i - (x_k - x_{i,k})^{2} \right]^{(1-\rho_i)} / \xi_i \right)^{1/\rho_i}$$

where $x_k \ (k \in \{1, 2, 3\})$ represents the value assigned to the $k^{th}$ issue by $o$; the parameter $x_{i,k}^*$ is the value assigned to the $k^{th}$ attribute by $o_i^*$, i.e., party $i$'s most preferred setting (i.e., the ideal point) for the $k^{th}$ issue; and $\gamma_{i,k}$ reflects the relative weight (i.e., importance) that party $i$ attaches to this variable. Moreover, $\xi_i$ is the substitutability coefficient that determines the curvature of parties’ indifference surfaces and $\rho_i$ is a risk aversion factor. Finally, $\theta_i$ is to ensure that the term inside the square bracket is always positive. For a more detailed description of the utility functions, the reader is referred to Section 4 in [6].

Table 7.3: Range of parameter values of the parties’ utility functions

<table>
<thead>
<tr>
<th>Variable</th>
<th>Agricultural users (A)</th>
<th>Urban users (U)</th>
<th>Environmentalists (E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{i,1}$</td>
<td>0.90 1.00</td>
<td>0.90 1.00</td>
<td>0.00 0.10</td>
</tr>
<tr>
<td>$x_{i,2}$</td>
<td>0.25 0.35</td>
<td>0.90 1.00</td>
<td>0.50 0.60</td>
</tr>
<tr>
<td>$x_{i,3}$</td>
<td>0.00 0.10</td>
<td>0.00 1.00</td>
<td>0.90 1.00</td>
</tr>
<tr>
<td>$\gamma_{i,1}$</td>
<td>0.90 1.00</td>
<td>0.90 1.00</td>
<td>0.75 0.85</td>
</tr>
<tr>
<td>$\gamma_{i,2}$</td>
<td>0.25 0.35</td>
<td>0.90 1.00</td>
<td>0.50 0.60</td>
</tr>
<tr>
<td>$\gamma_{i,3}$</td>
<td>0.75 0.85</td>
<td>0.25 0.35</td>
<td>0.90 1.00</td>
</tr>
<tr>
<td>$\xi_i$</td>
<td>-6.00 -1.00</td>
<td>-6.00 -1.00</td>
<td>-6.00 -1.00</td>
</tr>
</tbody>
</table>

Note: $i$ represents the party, $i \in \{A, U, E\}$.

We conduct 20 simulations in this application and use the following data interval in Table 7.3 provided by [6]. In the simulation, for every party $i$, $\rho_i = 0.5$, $\theta_i = 1$; and the initial agreement for each policy variable is randomly generated from the interval $[0, 1]$. Other parameters of the parties’ utility functions are randomly selected from pre-specifies intervals depicted in Table 7.3. Agricultural groups strongly support new infrastructure development and urban water users strongly prefer high degree of water transferability. The environmentalist groups prefer high level of environmental protection, while agricultural and urban interests prefer low levels. Thus the parties ideal points in this dimension ($x_{i,k}^*$) are constrained to be randomly drawn form the intervals $[0.90, 1.00]$ and $[0.0, 0.1]$ for the environmental party, and the urban and agricultural
7.5. Experiments

parties respectively. Moreover, the relative importance weight $\gamma_i$ that agricultural interests attach to this policy issues is higher than the weight that urban water users attach to the issues. Thus, the parameter $\gamma_i$ of the agriculture and urban parties are constrained to be randomly drawn from the intervals $[0.75, 0.85]$ and $[0.25, 0.35]$, respectively. For a more detailed explanation of the data interval, the reader is referred to Section 4 in [6].

A report of the statistical results of the simulations is presented in Table 7.4. With different type of utility functions, the results in water policy negotiation scenario are similar to the results of fishing rights negotiation with logarithmic utility functions in multilateral setting. On average, the negotiation process finished in similar number of iterations by PMP, MMP and DMP approaches. The proposed DMP approach produces greater $ESW$ and smaller $DR$ than PMP and MMP approaches. Moreover, in Table 7.4 (column 5) shows the average distances from the outcomes point obtained by different approaches to the Egalitarian point. On average, the final agreement point achieved by DMP approach is closer to the Egalitarian point, compared to PMP and MMP approaches.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Avg. iterations</th>
<th>Avg. ESW</th>
<th>Avg. DR (%)</th>
<th>Avg. DisEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMP</td>
<td>42</td>
<td>0.043</td>
<td>12.12%</td>
<td>0.123</td>
</tr>
<tr>
<td>MMP</td>
<td>41.5</td>
<td>0.047</td>
<td>11.44%</td>
<td>0.095</td>
</tr>
<tr>
<td>DMP</td>
<td>54.2</td>
<td>0.057</td>
<td>0.91%</td>
<td>0.008</td>
</tr>
<tr>
<td>Egalitarian</td>
<td>-</td>
<td>0.058</td>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

Note: $\lambda = 0.01$, $\tau = 0.0001$; DisEP: Distance from the Egalitarian point.

### 7.5.3 Overall performance with quadratic utility functions

In this section, we analyse the overall performance of the proposed EDD and DMP approaches with quadratic utility functions. We have tested the proposed EDD and DMP approaches on 5000 experiments with random preference combinations and random initial tentative agreements, respectively. All graphic outputs are included in Section 7.7 for ease of reading.

We provide a careful analysis on the impact of different pre-defined moving distances $\lambda (\lambda \in \{0.01, 0.05, 0.1, 0.5, 1\})$ on the scale of fairness and the corresponding computational complexity (measured by the total number of iterations during the negotiation
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

process). Moreover, we also compute the egalitarian solution with the agents’ utility functions in each experiment such that a comparison between the results obtained under complete information and incomplete information setting is provided. In those experiments, $\tau$ is still 0.0001, the value of each attribute is normalized to $[0, 1]$. Assume first that two agents who are negotiating over two issues $x_1$ and $x_2$. The preferences of agents 1 and agent 2 is characterized by the following quadratic utility functions:

$$u_1(x_1, x_2) = 1 - \sum_{j=1}^{2} w_{1,j} x_j^2$$

$$u_2(x_1, x_2) = 1 - \sum_{j=1}^{2} w_{2,j} (1 - x_j)^2$$

where $w_{i,j}$ is the weight that agent $i$ puts on the attribute $j$, $\sum_{j=1}^{2} w_{i,j} = 1$, $i \in \{1, 2\}$.

Figure 7.6(a) show the average $ESW$ (egalitarian social welfare) with different pre-defined moving distance $\lambda$ in bilateral negotiations. Averagely, compared to the results of the bisecting approach and the $\epsilon$-Satisfying approach, $EDD$ produces a larger $ESW$. From Figure 7.7(a) we can also see that the proposed $EDD$ approach results in fairer outcomes with a much smaller average $DR$ than that of bisecting approach and $\epsilon$-Satisfying approach.

Furthermore, Figure 7.8(a) shows the distribution of $DR$ of different approaches with different $\lambda$ in bilateral negotiations. The maximum, mean, median and minimum cases are marked respectively. The lower bound of the bar containers in Figure 7.8(a) is the 10th percentile of $DR$ and the upper bound is the 90th percentile. Thus, each of these bar containers contains 80% data of $DR$ of each approach with each $\lambda$ in those experiments respectively. And the shorter the bar is, the more centralized the data is. We can observe that the $DR$ of $EDD$ approach are much more concentrated than bisecting approach and $\epsilon$-Satisfying approach, especially when $\lambda = 0.01$, the $DR$ data of $EDD$ approach is centred largely close to zero. Finally notice that in more than 20% of those experiments, $\epsilon$-Satisfying could not be used and in only 71.16% of the cases where it is applicable, it actually reaches final optimal outcomes.

We provide a analysis on the impact of different pre-defined moving distances $\lambda$ ($\lambda \in \{0.01, 0.05, 0.1, 0.5, 1\}$) on the scale of fairness and the corresponding computational complexity (measured by the total number of iterations during the negotiation pro-
7.5. Experiments

Applying different $\lambda$, the results of \texttt{EDD} and bisecting approaches varies in a similar regularity in bilateral negotiation. Generally, the outcomes are less fair: the average $ESW$ is decreasing (Figure 7.6(a)), while $DR$ is increasing (Figure 7.7(a)), and the distribution of $DR$ is less concentrated (Figure 7.8(a)) as $\lambda$ increases. Comparing those approaches with the egalitarian solutions (See the axes gridlines in Figure 7.6(a) and Figure 7.7(a)), we can observe that when $\lambda$ is small, the results of \texttt{EDD} is very close to the egalitarian solutions. Figure 7.9(a) and Figure 7.10(a) further numerically demonstrate the average distance and the distance distribution between the final agreements of those approaches and the egalitarian solutions. In the distribution bar chart Figure 7.10(a), the maximum, mean, median and minimum cases are marked respectively. The lower bound of the bar containers is the 10th percentile of $DR$ and the upper bound is the 90th percentile. Thus, each of these bar containers contains 80% data of the distance between the egalitarian solutions and final agreements of each approach with each $\lambda$ in those experiments respectively. And the shorter the bar is, the more centralized the data is. When $\lambda$ is small, the proposed \texttt{EDD} approach converge to the agreements that are also geometrically much closer to the egalitarian solutions compared to other existing methods.

We then extend the above experiment to multilateral setting. In multilateral cases, we assume there are three agents negotiating over three attributes $x_1, x_2$ and $x_3$. The utility functions of agents are defined by the following quadratic utility functions:

\[
\begin{align*}
    u_1(x_1, x_2, x_3) &= 1 - (w_{1,1}x_1^2 + w_{1,2}x_2^2 + w_{1,3}x_3^2) \\
    u_2(x_1, x_2, x_3) &= 1 - [w_{2,1}(1 - x_1)^2 + w_{2,2}x_2^2 + w_{2,3}(1 - x_3)^2] \\
    u_3(x_1, x_2, x_3) &= 1 - [w_{3,1}x_1^2 + w_{3,2}(1 - x_2)^2 + w_{3,3}(1 - x_3)^2]
\end{align*}
\]

where $w_{i,j}$ is the weight that agent $i$ puts on the attribute $j$, $\sum_{j=1}^{3} w_{i,j} = 1$, $i \in \{1, 2, 3\}$.

For multilateral setting, we obtain a consistent result of the proposed \texttt{DMP} approach with \texttt{EDD} approach in bilateral negotiation. Compared to the \texttt{PMP} and \texttt{MMP} approaches, the proposed \texttt{DMP} approach results in averagely larger $ESW$, much smaller $DR$ and more centralized $DR$ distribution than \texttt{PMP} and \texttt{MMP} approaches (see Figure 7.6(b), Figure 7.7(b) and Figure 7.8(b)). Similarly, in the experiments of multilateral negotiation, when the moving distance $\lambda$ increases, the results of \texttt{PMP}, \texttt{MMP} and the proposed

---

\textsuperscript{7}We recall that \textit{$\epsilon$-Satisfying} approach is not based on the method of improving directions and does not consider different moving distances.
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

DMP approaches are less fair (see Figure 7.6(b), Figure 7.7(b) and Figure 7.8(b)). Comparing with the egalitarian solution, we can also see that when $\lambda$ is small, the results of the proposed DMP approach are very close to that of the egalitarian solutions. Figures 7.9(b) and Figure 7.10(b) also numerically demonstrates the average distance and the distance distribution between the egalitarian solutions and the final agreements achieved by different approaches in multilateral negotiation.

The proposed EDD and DMP approaches guarantees Pareto-efficiency, and in each iteration of negotiation it tries to assign joint gains fairly among the agents. Thus, they improve the utility gain of the worst-off agent, leading to fairer agreements. Regarding the computational complexity, Figure 7.11(a) and Figure 7.11(b) calculate the average number of iterations with different $\lambda$ in bilateral and multilateral negotiation using different approaches respectively. The number of iterations of those approaches are decreasing in a very similar fashion as $\lambda$ increases. Both in bilateral and multilateral negotiation, the average number of iterations of EDD, Bisecting, PMP, MMP and DMP approaches is approximately 35 when $\lambda = 0.01$.

7.6 Summary

In Chapter 7, the problem of negotiation over multiple continuous issues using a mediation approach was studied. Some important aspects of this topic were first discussed, including the preference representation (utility function), the basic negotiation model and other relevant topics, and a survey of the literature on the joint seeking problems in negotiation was presented. The survey reveals that there are a number of methods to address the joint gain seeking problems in utility-based negotiation; however, most of them do not address the fairness issues between the agents. This motivates the establishment of a fair framework for this problem.

A mediation framework for iteratively creating joint gains in multilateral negotiations, based on the method of improving directions, was then proposed. In the proposed framework, the mediator searches for an efficient and fair compromise direction in which the agents can move along in each iteration of negotiation, until a Pareto-efficient outcome is achieved. A simple approach to the direction-choosing problem in bilateral negotiations, entitled Equal Directional Derivative (EDD) has been presented in this chapter.
A non-trivial generalisation of this approach has been further provided to deal with multilateral negotiations involving the solution to a mathematical programming problem, called DMP Deviation Minimisation Problem (DMP). The proposed approach aims to improve the negotiating agents’ utility values from the status quo, while minimising the difference between the agents’ utility gains, leading to fair agreements. In this chapter, we have presented two case studies to illustrate the capacity of the proposed approach in real world negotiation scenarios. We have also evaluated experimentally the overall performance of our proposed approach. The experimental results demonstrate that our proposed approach not only guarantees Pareto-efficiency, but also produces fairer agreements that are closer to egalitarian solutions.

Nonetheless, the proposed approach is based on the assumption that the agents are honest. Thus, investigating the strategy of non-truthful agents is an important topic for future research on this topic.
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

7.7 Graphic Output

Figure 7.6: Average ESW

Figure 7.7: Average DR
7.7. Graphic Output

(a) Bilateral negotiation

(b) Multilateral negotiation

Figure 7.8: 10th-90th Percentile box of DR

Figure 7.9: Average distance from egalitarian solution point
Chapter 7. Utility-based mediated negotiation over multiple continuous issues

(a) Bilateral negotiation. The maximum distance between the egalitarian solution point and the \( \epsilon \)-Satisfying solution point goes further to about 1.2 and is not included in this graph.

(b) Multilateral negotiation

Figure 7.10: 10\(^{th}\)-90\(^{th}\) Percentile box of the distance from egalitarian solution

Figure 7.11: Average number of iterations
Chapter 8

Conclusion

In this final chapter, a brief summary is presented on the thesis work and findings. Subsequently, some suggested future research directions are discussed.

8.1 Summary of this thesis

This thesis contributes to the field of agent-based group decision-making, presenting several approaches for group decision-making that can lead agents to efficient and fair outcomes in different scenarios and with different decision criteria. The research work in this thesis could be divided into two parts, namely, preference aggregation in combinatorial domains and multi-issue negotiation. These two parts address the group decision-making problem in settings in which there is, respectively, complete and incomplete information.

The first part of this thesis focused on combinatorial domains and investigated techniques for group decision-making, providing that the agents’ preferences are known (by the preference aggregator). In this part, graphical models CP-nets and TCP-nets are used as the representation languages of individual agents’ preferences. Two different mechanisms for group decision-making with CP-nets and TCP-nets under different decision criteria were presented.

Before going into the setting of making a collective decision from a collection of CP-nets, Chapter 3 first revisited the problem of dominance testing in CP-nets, and invest-
Chapter 8. Conclusion

igated a numerical approximation, i.e., a penalty scoring function, for arbitrary acyclic CP-nets. This penalty scoring function was further used as a heuristic of an algorithm to answer dominance queries in CP-nets.

Based on the penalty scoring function for CP-nets, Chapter 4 further incorporated the relative importance information among pairs of variables, and presented a more general penalty scoring function for both acyclic CP-nets and conditionally acyclic TCP-nets. After that, we continued to propose the first approach for group decision-making with CP-nets and TCP-nets based on the penalty scoring function. Given that the individual penalty scores are computed, we further defined a collective penalty function in Chapter 4, which can convey the preference of the group. As there are exponentially many outcomes in combinatorial domains, a heuristic algorithm, by which to compute the optimal outcome according to a given decision rule, was also presented in Chapter 4.

The second mechanism for group decision-making in combinatorial domains was presented in Chapter 5, in which the agents’ preferences are represented by (possibly cyclic) CP-nets and we considered majority rule to be the decision rule. The chapter started with several theoretical results on the relationship of different notions of winners, namely, weak local Condorcet winner (wLCW) and possible winner. Subsequently, we presented a polynomial-time reduction that converts the problem of finding all wLCWs to a corresponding CSP problem with cardinality constraints. As such, the possible winners could be obtained by filtering out the dominated outcomes from the set of wLCWs. The computational efficiency of this approach was shown both theoretically and experimentally. Different from the technique introduced in Chapter 4, the approach introduced in this chapter is based only on the qualitative information from the CP-nets. Hence, no numerical transformation or conversion is required from the original CP-nets. Moreover, this approach enables us to aggregate preferences, even when the agents’ CP-nets are cyclic.

The two mechanisms for agent-based group decision-making introduced in the first part of this thesis contain a number of assumptions: (i) it is assumed that the agents’ preferences are represented by structured preferences CP-nets and TCP-nets (Chapter 4 also requires that the preference networks are acyclic); and (ii) it is assumed that each agent will fully submit their preference networks in order to make a collective decision. To relax the assumption of knowing the preference of every agent, the second part of this thesis contributes to the problem of agent-based group decision-making under incomplete information. The second part of this thesis mainly investigated multi-issue
negotiation approaches by means of which the agents could jointly make a collective
decision.

Chapter 6 relaxes the restriction on the language for representing the agents prefer-
ences. A proposal was advanced in Chapter 6 for an efficient protocol for negotiation in
combinatorial domains, which is independent of the preference representation models.
The proposed protocol is sufficiently general that it is applicable to most preference
representation models in combinatorial domains, including quantitative models such
as utility and qualitative models such as CP-nets and TCP-nets. Another advantage
of the proposed protocol is that it enables the negotiating agents to efficiently identify
solutions along the Pareto-frontier in combinatorial domains. It uses a distributed
search that visits only a small subspace of the whole outcome space. Thus, it is com-
putationally efficient and does not require the agents to fully reveal their preference
information.

Subsequently, the domain with continuous issues was further explored in Chapter 7.
A mediated negotiation approach for group decision-making over multiple continuous
issues is proposed. Based on the assumptions that each agent has a utility function
over the decision issues, and that the agents are originally at a pre-negotiated outcome
point (the status quo, initial tentative agreement), the proposed approach employs an
unbiased third party, the mediator, to encourage the agents to act cooperatively and
to support them in reaching mutually better agreements through negotiation. The ne-
gotiation process takes place under an incomplete information setting, i.e., it is not
assumed that any utility functions are explicitly known by the mediator. We only as-
sume that the information needed to execute the method, e.g. the utility gradients at the
tentative agreements, is available. As such, the proposed approach strongly supports
automated negotiation in real life, in which, for instance, it may be impractical to elicit
the complete preference of an agent, or the preference of an agent may be changing
dynamically during the process of negotiation. Finally, in each iteration of negotiation,
the proposed approach tries to assign joint gains fairly among the agents. Thus, it im-
proves the utility gain of the least fortunate agent, leading to fairer agreements. From
a technical point of view, it can be considered as an approximation to the egalitarian
solution with only incomplete information.

To summarise, this thesis is designed to address the agent-based collective decision-
making problem, in the case of having complete information and under incomplete
information setting, respectively. It focused mainly on combinatorial domains and
with structured preferences CP-nets and TCP-nets. A technique was also proposed for
group decision-making that can deal with general preferences, and further investigation of the group decision-making problem over continuous issues with utility functions was conducted.

8.2 Limitations of and extensions to the current work

This thesis proposed several computationally efficient group decision-making mechanisms that, given the problem specification (e.g., the issues and their value domain, the decision rule, etc.), lead rational agents to Pareto-optimal and fair outcomes. However, the research in multi-agent group decision-making problems is far from being mature. Further work is required to develop a richer framework that provides an adequate set of alternative solutions to a wider set of problems. In particular, each mechanism introduced in this thesis has some limitations that need to be addressed, and some extensions are required in future research.

8.2.1 Extensions to penalty score-based techniques for dominance testing and group decision-making

The heuristic algorithm for individual dominance testing introduced in Chapter 3 only works for acyclic CP-nets. We expect that the extension to cope with dominance testing in conditionally acyclic TCP-nets is straightforward, based on the penalty scoring function discussed in Chapter 6. However, these penalty scoring functions are only applicable to acyclic preference networks, i.e., acyclic CP-nets or conditionally acyclic TCP-nets. The extension to deal with cyclic preferences is expected to be much more complex and needs to be investigated further.

One possible way to define penalty scoring functions for cyclic preference networks is to use a fixed-point definition, which may be identical to the proposed penalty scoring functions in the cases in which the preference network is acyclic. Moreover, we have a preliminary plan to extend the penalty score-based group decision-making techniques discussed in Chapter 4. We attempt to translate penalties into a generic constraint-optimisation framework (rather than designing a specific heuristic algorithm). Such a method is believed to not only benefit from the associated efficient algorithms, but also to be able to encode more general problems (e.g., collective decision-making problems
8.2. Limitations of and extensions to the current work

with hard constraints). Nonetheless, an investigation into the possibility and efficiency of such a transformation is a pre-requisite for the extension to these approaches.

8.2.2 Extensions to the majority-rule-based preference aggregation mechanism

In Chapter 5, an efficient algorithm to compute the possible majority winners in group decision-making with CP-nets was presented. The approach can be directly applied to TCP-nets by considering only the preference information of local variables when computing the set of weak local Condorcet winners (wLCWs). This means that the relative importance relation will only be taken into account in the second step, where we filter out the dominated alternatives in the set of wLCWs. However, we believe that incorporating the relative importance information in the process of computing wLCWs could further narrow the search space and improve the computational efficiency of the proposed approach. Hence, we are planning to investigate this issue further. Moreover, computing the possible winners for other decision rules (e.g., Lex, Max) is also an interesting future research issue regarding this work.

8.2.3 Further investigation on the fairness properties of the protocol for negotiation in combinatorial domains

Regarding the negotiation protocol introduced in Chapter 6, there is a need to investigate the fairness properties of the proposed protocol. The fact that the final outcome reached by means of the proposed protocol is (a set of) Pareto-optimal outcomes is only a weak guarantee. To be sure, one could have obtained such a set by asking every agent to give its optimal outcome. These outcomes are obviously Pareto-optimal, and the amount of communication or interaction between agents would have been much smaller. What the proposed protocol results in, in essence, is much better than that, because it seems that, in general, the obtained outcomes will be close to optimal for (ordinal) egalitarian social welfare. Consequently, it would be interesting to see a theoretical result on the fairness properties (if possible) or, at least, an experimental analysis completed by measuring the average egalitarian social welfare based on possible ways of measurement of individual agents’ payoffs.
One possible measurement of the agents’ payoffs is to use the individual ranking of the final outcome. As such, the egalitarian social welfare can be measured by the rank of the least satisfied agent. Another possibility is to consider the penalty scores introduced in Chapters 3 and 6 as the payoff measurement of the agents. Consequently, the egalitarian social welfare can be defined by the maximum penalty of the final outcome among all agents.

8.2.4 Extensions to the mediated negotiation approach

There are also several limitations to the mediated negotiation approach introduced in Chapter 7. Firstly, the negotiation is performed in a centralised fashion and is based on the assumption that the agents are honest, meaning that the agents truthfully submit their utility gradient information to the mediator in each iteration of negotiation. This makes the approach less suitable for real world applications. Consequently, one future research direction that might be taken when following up this work would be to investigate the strategy of non-truthful agents. Moreover, the mediator may become a computational and communications bottleneck when the problem is complex and involves a large number of variables. The scalability of the approach needs to be investigated further, for instance, the issue of time complexity (the time that the negotiation takes may actually influence the usefulness of the results of the negotiation).

The proposed approach is still restrictive, because it is based on an assumption that the agents’ utility functions are differentiable and strictly concave, and the outcome space is closed and convex. This assumption excludes the possibility of reaching local optima and thus guarantees the Pareto-optimality of the final agreement. However, it limits the applicability of the proposed approach for real world problems. Consequently, extensions to deal with more general problem settings, e.g., when the agents’ utility functions are non-monotonic, are another focus of future research.

Finally, notice that this work focuses only on addressing the joint gain seeking problems from an initial agreement point. Nonetheless, further investigation into how to choose a fair initial tentative agreement is needed, because it defines the beginning of the enhancement process and affects the fairness of the final agreement to be reached through mediation. If the initial agreement is unfair, it could be very difficult to improve general fairness.
8.2. Limitations of and extensions to the current work

8.2.5 Extensions to mix type of issues

The group decision-making methods proposed in this thesis either deal with combinatorial domains, in which the decision issues can take values from a finite set of discrete values; or over multiple continuous issues. However, there is still a lack of mechanisms that can deal with a more general problem setting with mixed types of issues. Hence, we are also planning to investigate in our future research the case in which there exists both continuous issues and discrete issues.

8.2.6 Extensions to the experiments

The corresponding computer experiments for each of the group decision-making mechanisms presented in this thesis provide valuable insights into the problem for the first time. However, since the research is still at a very early stage, many interesting tests were not performed and it is desirable that they be conducted in future research. Specifically, tests on a real world group decision-making support system with actual preference datasets would be of great interest, because they can provide researchers with an idea of how the framework and algorithms will perform in a realistic problem setting.

In most of the computer experiments we’ve carried out thus far, the scale of the problem is limited because of the running time limitation (limited number of variables, limited domain size and limited number of agents). More tests on larger problem settings and complicated preferences situations are desirable. Note that a more efficient implementation of the individual preference reasoning and group decision-making algorithm is a pre-requisite for extended tests.
Bibliography


226


