


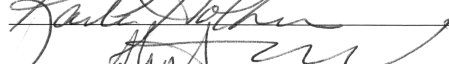




A DECISION-GUIDED GROUP PACKAGE RECOMMENDER BASED ON MULTI-
CRITERIA OPTIMIZATION AND VOTING

by

Hanan A. Mengash
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

	Dr. Alexander Brodsky, Dissertation Director
	Dr. Daniel Menasce, Committee Member
	Dr. Larry Kerschberg, Committee Member
	Dr. Karla Hoffman, Committee Member
	Dr. Stephen Nash, Senior Associate Dean
	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: April 13, 2016 Spring Semester 2016
George Mason University
Fairfax, VA

A Decision-Guided Group Package Recommender Based on Multi-Criteria Optimization
and Voting

A Dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy at George Mason University

by

Hanan A. Mengash
Master of Science
George Washington University, 1999
Bachelor of Computer Science
King Saud University, 1993

Director: Dr. Alexander Brodsky, Associate Professor
Department of Computer Science

Spring Semester 2016
George Mason University
Fairfax, VA



This work is licensed under a creative commons attribution-noderivs 3.0 unported license.

DEDICATION

To my loving father, who has been the greatest inspiration in my life. He believed that I could be who I am today more than I did. If not for him, this day would never have come.

To my sweet mother, who never stopped encouraging me. Her love, prayers, and encouragement gave me the energy and confidence to continue the hard work.

To my amazing husband, Ahmad, who has been my greatest supporter. He gave up years of his career to be next to me when I needed him most. He supported our whole family for many years and has always been there for our five children whenever their Mom was behind the computer screen.

To my five wonderful children, Bedour, Eman, Fahad, Moqbel, and Toleen, who never had enough time to tell Mom all the stories they wanted to tell. Their countless hugs gave me enormous energy to keep going.

To my sisters and brothers, who never stopped supporting me and believing in me. Their messages, calls, and love encouraged me along the way.

I dedicate this work with all my love.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor and mentor, Dr. Alexander Brodsky, for his endless support, advice, and patience throughout this process. My research would not have existed without his constant guidance and supervision. He believed in me and led me along the road to my goal. I am deeply grateful.

I am also deeply grateful to my committee members, Dr. Daniel Menasce, Dr. Larry Kerschberg, and Dr. Karla Hoffman, for serving on my committee and for generously giving me time and advice whenever I needed them.

I am also very grateful to Dr. David Schum, who served on my committee for three years until his health prevented him from continuing. Best wishes for his recovery.

I give special thanks to Princess Nora Bint Abdul Rahman University, which granted me a PhD scholarship, and to my country, which funded my education.

Most importantly, none of this would have been possible without the love, support, and patience of my family for all these years. I would like to express my sincere gratitude to my family.

TABLE OF CONTENTS

	Page
List of Tables	vii
List of Figures	viii
Abstract	ix
Chapter 1: Introduction	1
1.1 Group Package Recommenders: The Problem.....	1
1.2 Previous Research and Its Limitations	3
1.3 Research Challenges	6
1.4 Thesis Statement and Summary of Contributions.....	9
1.5 Organization of the Dissertation	13
Chapter 2: Related Work	14
2.1 Individual Recommender Systems.....	14
2.2 Package Recommender Systems.....	15
2.3 Group Recommender Systems	17
2.4 Overview of Voting Methods (Aggregation Strategies)	22
2.5 Social Factors in Group Recommender Systems	30
2.6 Multi-Criteria Recommender Systems.....	33
2.7 Summary of the Evaluation of Related Work	38
Chapter 3: Group Composite Alternatives Recommender (GCAR).....	41
3.1 Introduction	41
3.2 Overview of the Proposed GCAR Framework	43
3.3 Overview of Group Decision Methods Used in GCAR.....	46
3.4 Summary	59
Chapter 4: GCAR steps and evaluation under each voting method.....	60
4.1 Introduction	60
4.2 Eliciting User Utility Functions	61
4.3 Estimating the Group Utility Function.....	62

4.4	Optimization and Diversity Layering.....	66
4.5	Initial Experimental Evaluation	70
4.6	Summary	77
Chapter 5: Tailoring Group Package Recommendations to Large Heterogeneous Group		78
5.1	Introduction	78
5.2	GCAR with a Technique to Support Very Large Groups	79
5.3	Sampling the Entire Group and Eliciting User Utilities.....	81
5.4	Clustering Users' Utilities.....	81
5.5	Estimating Group Utility on the Basis of Subgroup Utilities.....	83
5.6	Experimental Evaluation of GCAR Scalability	86
5.7	Summary	91
Chapter 6: Case Study: Power Microgrid Operation and Investment Recommender (PMOIR)		93
6.1	Introduction	93
6.2	Renewable Energy Operation and Investment Group Recommender	95
6.3	Optimization Formalization	97
6.4	Power Component Modeling	105
Chapter 7: PMOIR Implementation and Experimental Study		117
7.1	Introduction	117
7.2	OPL Implementation	118
7.3	Experimental Study	127
7.4	Summary	132
Chapter 8: Conclusions and Future Work.....		133
8.1	Conclusions	133
8.2	Future Work	135
Appendix A: GMU Institutional Review Board (IRB) Approval Letter		137
Appendix B: Publications Related to this Dissertation.....		139
References.....		140

LIST OF TABLES

Table	Page
2.1 Overview of Voting Methods (Aggregation Strategies)	22
2.2 Example of Group Ratings Using Five Different Strategies.....	27
3.1 Example to Illustrate the Aggregation Strategies Used in GCAR	47
3.2 IRV Initial Votes	51
3.3 IRV Round 1	51
3.4 Hybrid Condorcet-IRV, Initial Votes.....	53
3.5 Hybrid Condorcet-IRV, Round 1	54
3.6 Hybrid Condorcet-IRV, Round 2.....	54
3.7 Hybrid Condorcet-IRV, Round 3.....	54
3.8 Categorization of Expertize Levels Based on Average Yearly Travel	55
5.1 Confidence Interval at Level 95% for the Accuracy Estimated Mean.....	91
7.1 Confidence Interval at Level 95% for the Estimated Mean of the Time Resolution (in Minutes)	132

LIST OF FIGURES

Figure	Page
3.1 Group Composite Alternatives Recommender (GCAR) Framework.....	44
3.2 Flowchart Illustrates the Hybrid Condorcet-IRV Method.....	53
3.3 Adjusted Group Utility	58
4.1 Diversity Layering	69
4.2 Average Recall and Precision for GCAR vs. Ideal (Average Strategy)	73
4.3 Average Recall and Precision for GCAR vs. Ideal (Least Misery Strategy).....	73
4.4 Average Recall and Precision for GCAR vs. Ideal (Average without Misery) ...	73
4.5 Average Recall and Precision for GCAR vs. Ideal (Structurally Adjusted Average) .	74
4.6 Average Recall and Precision for GCAR vs. Ideal (IRV Method).....	74
4.7 Average Recall and Precision for GCAR vs. Ideal (Condorcet-IRV Method)....	74
4.8 Confidence Interval at Level 95% for the Estimated Mean of the Percentage Differences Between GCAR Accuracy and Ideal.....	76
5.1 GCAR Framework with the Proposed Technique to Support Very Large Groups...	80
5.2 Alternatives Generation	87
5.3 User Utilities Generation	88
5.4 Average Recall vs. Rank (k) for the Proposed Framework	90
5.5 Average Precision vs. Rank (k) for the Proposed Framework.....	90
6.1 Example of a University Campus Microgrid.....	96
7.1 General Declarations in OPL.....	119
7.2 Common Component Cost Category Declarations in OPL	120
7.3 NPV Calculations in OPL.....	121
7.4 General Metrics Calculations in OPL	121
7.5 Common Operational and Investment Constraints in OPL	122
7.6 Utility Power Contract Parameters, Variables, and Metric Calculations in OPL ...	123
7.7 Utility Contract Operational and Investment Constrains in OPL	123
7.8 Backup Generator Parameters, Variables, and Metric Calculations in OPL	124
7.9 Backup Generator Operational and Investment Constraints in OPL.....	125
7.10 Renewable Resource Parameters, Variables, Metrics, and Constraints in OPL	125
7.11 Battery Component Parameters, Variables, and Cost Calculations in OPL	126
7.12 Battery Component Operational and Investment Constraints in OPL.....	126
7.13 Power-Consuming Service Parameters and Constraints in OPL	127
7.14 Typical Yearly Load Profile [123].....	129
7.15 Hourly Prices for Power Generation [125]	130
7.16 Experimental Mean Resolution Time	131

ABSTRACT

A DECISION-GUIDED GROUP PACKAGE RECOMMENDER BASED ON MULTI-CRITERIA OPTIMIZATION AND VOTING

Hanan A. Mengash, PhD

George Mason University, 2016

Dissertation Director: Dr. Alexander Brodsky

Recommender systems are intended to help users make effective product and service choices, especially over the Internet. They are used in a variety of applications and have proven to be valuable for predicting the utility or relevance of a particular item and for providing personalized recommendations. State-of-the-art recommender systems focus on atomic (single) products or services and on individual users. This dissertation considers three ways of extending recommender systems: (1) to make composite (package) rather than atomic recommendations; (2) to use multiple rather than single criteria for recommendations; and, most importantly, (3) to support groups of diverse users or decision makers who might have different, even strongly conflicting, views on the weights of different criteria.

Complex group recommender systems with these features are important in such areas as public policy and budget recommendations, energy infrastructure investment,

and health care plan selection by organizations. However, the problem of how to develop such systems has not been adequately addressed. Package recommendations present a unique challenge because they require the recommendation space to be very large, even infinite, and to be implicitly rather than explicitly defined. And group recommenders are considerably more complex than individual user recommenders. One reason for this complexity is the need to effectively aggregate users' preferences in a way that maximizes the group's satisfaction, fairness, and user-friendliness.

In this dissertation, I propose and develop a decision-guided group package recommender framework based on multi-criteria decision-optimization and voting. This framework operates on a very large, even infinite, recommendation space, which is implicitly defined by mathematical constraints. It is designed to provide a diverse set of optimal or near-optimal package recommendations to groups of users while taking into account the influence of individuals within the group, the dissimilarity of interest among the group's members, and the size and homogeneity of the group. The framework applies six alternative decision-making (voting) methods to refine its recommendations. Five of these come from social choice theories, namely the instant runoff voting (IRV), hybrid Condorcet-IRV, average, least misery, and average without misery methods. In addition to them, I develop a new method, the structurally adjusted average. I also develop a technique for scaling up the group recommender system for very large, heterogeneous groups.

In addition, I demonstrate how the proposed framework applies to a real problem through a case study of the Power Microgrid Operation and Investment Recommender

(PMOIR). PMOIR supports (1) operational decisions on how to control each microgrid component on, say, a half-hourly basis, and (2) investment decisions on microgrid components (e.g., renewable sources of energy and power storage) over an investment time horizon. In order to implement PMOIR, I mathematically model different power components and formalize the overall optimization problem. I also implement the optimization model for PMOIR as a mixed-integer linear programming (MILP) model.

Finally, I validate the proposed framework with three experimental studies: (1) a study demonstrating that the proposed framework can produce a small set of recommendations that retain near-optimality, in terms of precision and recall, when compared with manual voting by human participants; (2) a study demonstrating the framework's ability to support very large, heterogeneous groups with only minor degradation in precision and recall; and (3) a study demonstrating the framework's feasibility, in terms of computational time, for applying PMOIR on microgrids involving 200 power components, over a five-year time horizon, with around 8 million binary variables.

CHAPTER 1: INTRODUCTION

1.1 Group Package Recommenders: The Problem

The Internet is now the greatest source of information that has ever existed. While information retrieval systems are helpful tools for guiding users to the information they want, they are limited to retrieving only the most relevant items. Users, however, need more personalized search systems that can return items more suited to their particular preferences and interests. This is the goal of recommender systems.

Recommender systems help users who are overwhelmed with the range of products and services available by identifying those that are likely to match their preferences. These systems learn each user's preferences and provide personalized recommendations.

Most existing recommender systems deal with single items rather than composite products and services ("packages"), and with individual users rather than groups of users. However, there are many situations in which individual recommender systems cannot be used because people are operating in groups—for example, friends going to see a movie together, family members planning a vacation, civic planners deciding on energy infrastructure investment, or committee members deciding on public policy and budget recommendations. In such scenarios, group recommendations would be the optimal way to provide specific suggestions of products and services to meet the needs of the diverse

users or decision makers. While all the users may have different, even strongly conflicting preferences, a group recommender system (GRS) should consider each user as a member of the group and generate recommendations that reflect the whole group's preferences.

Because of the social nature of human beings, group activities are an important part of our everyday life. This fact motivates the study of GRSs, which is still a new area in comparison to the study of individual user recommender systems. Moreover, there are many situations where in which a group of people needs package recommendations in which multiple criteria are taken into account, such as cost, quality, reliability, and risk.

While state-of-the-art recommender systems focus on atomic products and services and on individual users, this dissertation addresses group package recommenders (GPRs), which extend recommender systems in three ways:

- To consider composite rather than atomic recommendations.
- To use multiple rather than single criteria for recommendations.
- To support groups of diverse users and decision makers with different, even strongly conflicting, views on the weights for different criteria.

Some examples of this new class of recommendations are power-microgrid operational and investment recommendations, group-travel package recommendations, public policy and budget recommendations, and health-care plan selections by organizations. These recommendations are composite: a travel recommendation may involve interrelated air travel, accommodations, activities, and car rentals. They are also associated with multiple criteria, such as cost, benefit, enjoyment, satisfaction, and risk.

Finally, it is often necessary to satisfy a diverse group of individuals with different views on the relative importance of these criteria. This new class of recommender systems has not been addressed in previous research and is the focus of this dissertation.

1.2 Previous Research and Its Limitations

Extensive work has been done on recommender systems, most of it focused on single users rather than groups of users (e.g., [1-5]). Recently, though, researchers have proposed group recommenders in different domains and applications that use different strategies to aggregate individual preferences into group models. Some common uses of GRSs are in recommending TV programs and movies (e.g., [6-12]); finding songs to play in a shared public space (e.g., [13-15]); recommending video clips (e.g., [16]); recommending recipes to families (e.g., [17]); recommending photos (e.g., [18]); recommending web pages (e.g., [19, 20]); and finding tourist attractions for groups (e.g., [16, 21-25]). However, none of these applications was designed for packages of products and services, which require the recommendation space to be very large, even infinite, and implicitly rather than explicitly defined.

More recently, there has been a host of research supporting package recommendations [26-33], but it does not consider dynamic preference learning and decision optimization. The CARD framework [34] and the COD framework [35] support packages of product and service definitions and provide recommendations based on dynamic preference learning and decision optimization. However, both CARD and COD are recommender systems for individuals rather than groups.

In addition, the majority of recommender systems rely on a single ranking or

utility score, whereas in many applications there are multiple criteria, such as cost, quality, enjoyment, satisfaction, and risk, that need to be taken into account. Multi-criteria ranking has recently been explored in recommendation set retrieval [5, 36-38]. These methods choose a set of alternatives on the basis of a distance measure calculated for each of the criteria. Multi-criteria ranking can support both similarity- and diversity-based ranking. However, these methods are based on distance measures to increase the quality of each recommendation, which competes with the ability to diversify recommendations [34]. In addition, they focus on individual users rather than groups.

There has also been work on multi-criteria recommender systems, which have roots in multi-criteria optimization techniques (e.g., [39, 40]). However, these systems focus on atomic products rather than composite products and on individual users rather than groups.

Several approaches to aggregating individuals' utility functions have been suggested [41-43]. Some earlier, multi-attribute utility theory (MAUT) methods of group decision are reviewed in [41], including the use of weighted algebraic means, as proposed in [42], and the use of the sample additive theory to aggregate the individuals' utility functions, as proposed in [43]. The aggregated utility function, however, is only an approximation, and using it directly may limit decision makers' flexibility in refining their choices.

Furthermore, most existing GRSs require specific group characteristics rather than providing a general framework. For instance, the aggregation method in [10] works only when the group is very homogenous, and [6] works well only with small groups. And the

majority of these GRSs assume that individual preferences are already known [44]. The only exception is [45], which assumes they are not, but this method is based on members' critiques of desired package features, which requires experience with those features that is not always available.

Many GRSs are also intrusive and require significant feedback from users. For example, the Travel Decision Forum [23] and Collaborative Advisory Travel System (CATS) [45] require the group to negotiate the group model. Although feedback remains a primary factor in the recommender system concept, it might be better to extract information from users implicitly.

In addition, most current GRSs aggregate preferences without using fairness criteria. For instance, in [23] and [10], group members whose preferred features are not selected are simply "left out," not compensated with other desirable features. Using the average and plurality voting strategies, as in [9], does not avoid the fairness issue.

Furthermore, most previous work has been based on aggregation strategies that combine the group members' ratings in the same way without considering how they interact with each other. But it is natural for some members to have more influence in the aggregation of preferences than others—those with authority or expertise or who are more trusted, for example. These members must be treated differently to improve the group decision-making process.

Regarding which aggregation strategies are to be used in group recommender models, several researchers (e.g., [44, 46-49]) have concluded that taking certain main factors into account while developing the GRS will yield a significant improvement.

These factors include the influence of group size, the influence of group homogeneity, the personality of each member, and the relationships among them.

Several GRSs consider some of the main social factors when generating recommendations for groups (e.g., [8, 17, 21, 23, 45-47, 50]), but none was designed for packages, which require a large and implicitly defined recommendation space.

I provide further details on related work in Chapter 2.

Addressing these limitations of the previous research on recommender systems is the focus of my dissertation. To the best of my knowledge, based on popular group recommender surveys [44, 49, 51-54], there is no recommender system that involves both groups of users and composite recommendations using multiple criteria. The proposed framework will be the first recommender system to address all these issues.

1.3 Research Challenges

- A. **Recommendation space.** The recommendation space is very large and defined only implicitly. Therefore, multi-criteria optimization will be a key technique for supporting the decision-making process.
- B. **Group recommendations.** The challenges for GRSs are considerably more complex than for individual user recommenders. Some of the reasons for this complexity are:
- **Acquiring information about individual users' preferences.** While the usual individual recommender techniques can be used, such as collaborative and content-based filtering, it is difficult to infer an individual's preferences when a group uses the recommender system [44] because individuals' ratings may depend on their groups. For example, parents might be happy to watch a program with

their children but not want to watch it with their friends.

- **Effectively aggregating users' preferences in a way that maximizes group satisfaction, fairness, and user-friendliness.** In GRS, a group of users may share some preferences but not others. At least some preference conflicts are likely to occur, so some form of aggregation method is needed. This is the most obvious difference between group recommenders and individual recommenders. Choosing an aggregation method that maximizes the group's satisfaction is a critical step in designing a group recommender.
- **Estimating a group utility function taking multiple factors into account.** The problem of reducing the recommendation space can be addressed with an optimization technique for finding a small set of optimal or near-optimal recommendations. Before optimization can be applied, however, the GRS must be able to estimate a group utility function that captures the whole group's preferences, taking into account a number of factors such as the influence of individuals within the group, the dissimilarity of interests among group members, the size of the group, and its homogeneity.
- **Helping a group of users reach a final decision.** To refine the recommendations, one of the voting mechanisms from social choice theories can be used. These methods, however, apply only when there are a small number of alternatives to vote on. In the case of composite alternatives, the search space for recommendations is very large, making a voting method impractical. Therefore, we need to restrict the space of recommendations to a small subset that is highly

relevant to the whole group and which can then be refined through voting.

- **Determining actual group preferences for evaluative comparison to system recommendations.** Research on GRSs presents a significant additional challenge in the difficulty of evaluating the effectiveness of group recommendations [8, 55]—that is, determining an overall group preference to use as a ground truth for measuring the accuracy of a GRS. There are two main options: conducting live user studies and analyzing synthetic data sets. With either, it is difficult to determine an overall group preference to use as ground truth.

The former needs group discussion and interaction to make the final decision, which will help to determine the ground truth for the whole group’s preferences. However, this is impractical for large-scale evaluation even apart from the overhead involved in bringing groups together, as large groups are needed for such studies. Furthermore, even if we do interview real users, we need to decide how the individual evaluations are to be integrated. This is difficult because different aggregation methods will give different results, and there is no single best aggregation method to apply.

The latter option is based on generating synthetic groups from the users of a traditional individual recommender system. While this works well for large-scale evaluation, it is again difficult to determine “actual” group preferences (i.e., what a group’s preference outcome was) from individual users’ data [55].

- C. **Diversifying the recommendation set.** Because the group utility is only an estimate, it is important to have alternatives that are sufficiently diverse in terms of individual

decision makers' preferences. Hence there is a tradeoff to be made between the competing goals of optimization and diversity.

Addressing these challenges in a single recommender system involving composite recommendations, using multiple criteria, and supporting groups of users introduces a dimension of complexity that has not been addressed, which is the focus of this dissertation. I provide further details of the dissertation's contribution in the following section.

1.4 Thesis Statement and Summary of Contributions

Thesis Statement

A decision-guided recommender can be developed that will

- Automate the selection of packages of products and services in an optimal way,
- Deal with multiple criteria associated with the packages,
- Support large, heterogeneous groups of users, and
- Be feasible in terms of efficiency and scalability.

Summary of Key Contributions

A. Developing a framework for recommending composite products and services to groups. I have developed a decision-guided group package recommender framework based on multi-criteria decision optimization and voting. The framework operates on a very large, even infinite, recommendation space, which is implicitly defined. This framework is designed to do the following:

- Extend the existing recommender systems in three ways, to (1) consider

composite recommendations, (2) deal with multiple criteria for recommendations and most importantly (3) support groups of users.

- Take into account three important factors in group decision-making process: (1) the influence of individuals within the group, (2) the interest dissimilarity among group members, and (3) the size and homogeneity of the group.
- Diversify group recommendation sets. Because the group utility function is only an estimate, it is important to have diverse alternatives to meet individual decision makers' preferences. To do this I developed an extension of the diversity layering method used in the CARD framework [34].
- Apply six alternative decision-making (voting) methods to refine the recommendations to their final form. Five of these come from social choice theories: the instant runoff voting (IRV) method, the hybrid Condorcet-IRV method, the average method, the least misery method, and the average without misery method. In addition, I developed a new method, the structurally adjusted average, which takes into account the influence of decision makers within the group and the dissimilarity of opinions among them.
- Scale up the GRS for very large, heterogeneous groups. I did that by developing a technique based on clustering the whole group into smaller, homogeneous subgroups of decision makers with similar utilities.

B. A case study of the Power Microgrid Operation and Investment Recommender (PMOIR). I demonstrate how the proposed framework applies to a real problem by considering a realistic case study in which PMOIR is used to support a group of

decision makers by recommending an optimal set of operation and investment decisions regarding interrelated power components in a power microgrid on a university campus. The recommendations include optimal settings and values of decision control variables, such as the amount of power generated from the resource components (e.g., batteries, backup generators, utility contracts, and renewable resources) and the amount consumed by the power-consuming service components (e.g., heating, ventilation, air conditioning, and lighting) in each time interval. The goal is to maximize the net present value (NPV) within the required demand satisfaction ratio and within the bound for greenhouse gas (GHG) emissions. This is done while taking into account all components' interactions and satisfying a group of diverse decision makers who may have conflicting views on the weights of the relevant criteria.

C. Formalizing and implementing optimization models for power component operation and investment. To implement PMOIR, I mathematically modeled different power components and formalized the overall optimization problem. In addition, I implemented the power optimization model for PMOIR as a mixed-integer linear programming (MILP) model using IBM Optimization Programming Language (OPL) and CPLEX Studio.

D. Validating the effectiveness and efficiency of the proposed framework. This validation involved three experimental studies:

- A study with human participants in which the precision and recall of the proposed framework were compared with voting procedures run manually by human

participants. This study showed that the proposed framework can produce a small set of recommendations that remain nearly optimal in precision and recall.

- A study demonstrating the framework's scalability to very large, heterogeneous groups. Utility functions were generated synthetically for members of such groups, and the precision and recall of the framework were compared with the recommendations that would be generated through applying manual voting methods on the entire large groups. The study demonstrated an average precision ranging from 0.95 for top-1 recommendations to 0.80 for top-5 recommendations. For recall, it demonstrated an average value ranging from 0.19 for top-1 recommendations to 0.80 for top-5 recommendations.
- A study demonstrating the feasibility in computational time of the proposed framework through a realistic case study using PMOIR. This study used data sets of various sizes involving different numbers of microgrid energy components over different time horizons. It demonstrated that the proposed framework is feasible and practicable to operate on medium-sized and large microgrids to generate small sets of optimal and diverse solutions within a reasonable time. For example, the largest data set in this study, involving 200 components over a five-year time horizon, was solved in less than five hours. Note that this data set contained more than 23 million constraints and about 18 million variables, of which more than 8 million were binary and almost 10 million were continuous.

1.5 Organization of the Dissertation

This dissertation is organized as follows:

- Chapter 2: Related work
- Chapter 3: Group composite alternatives recommender (GCAR)
- Chapter 4: GCAR steps and evaluation under each voting method
- Chapter 5: Tailoring group package recommendations to large heterogeneous groups
- Chapter 6: Case study: Power microgrid operation and investment recommender (PMOIR)
- Chapter 7: PMOIR implementation and experimental study
- Chapter 8: Conclusions and future work

CHAPTER 2: RELATED WORK

2.1 Individual Recommender Systems

An individual recommender system is a system that attempts to recommend items (e.g., movies, music, TV programs, books, news, web pages) that will be of interest to a single user [56]. Individual recommender systems try to predict the rating a user would give to an item he or she has not yet encountered on the basis of some reference characteristics, which may belong to the item's content (the content-based approach) or to the user's social environment (the collaborative filtering approach).

Individual recommender systems have been extensively studied over the past two decades. Popular surveys (e.g., [1-5]) have classified individual recommenders into content-based, collaborative filtering (CF), and hybrid approaches. More recently, knowledge-based, utility-based, and demographic systems have been proposed, using different techniques to recommend alternatives to users.

A *content-based recommender* (e.g., [1, 57]) recommends items similar to the ones the user preferred in the past on the basis of features (contents) of those items. There are some limitations to the content-based approach: (1) limitation by the objects' features; (2) overspecialization, since this method returns similar items to those the user has liked; (3) the cold start (or new users) problem, the fact that when users have not yet rated enough items, the system is unable to predict their interests [1, 2].

Collaborative filtering (e.g., [1-3]) systems recommend items that have been preferred by similar users—that is, they reflect the preferences of the users rather than the features of the item. While CF is the most successful and widely used recommendation approach, it also has some limitations: (1) The cold start problem: in addition to the new users problem, explained above, the recommender system will not recommend an item until enough users have rated it (the new items problem). (2) Sparsity: the success of a recommender system depends on there being enough users in the system. In most recommender systems, each user rates only a small number of items, which make it difficult for the system to find similarities among the users or the items. (3) *Gray sheep*: users whose preferences do not consistently agree or disagree with those of any group of users [1, 2].

The *hybrid approach* (e.g., [4, 58-61]) combines methods of the CF and content-based approaches to minimize the limitations they have when used separately. There are many ways to combine methods into hybrid recommender systems [59].

2.2 Package Recommender Systems

Most existing recommender systems provide users with single items, such as movies, music, TV programs, and web sites. However, several applications need systems that can recommend packages of items. These include travel planning, health care planning, and course recommendations for students. In travel planning, for example, a user needs a package of recommendations including places to visit, accommodations, airline reservations, and car rental.

Researchers have recently considered package recommendations. For example,

some [62] have recommended fixed-size packages that are tuples of the entities included—cities, airlines, and hotels—with fixed associations among the entities to determine entity scores. They queried documents using keywords instead of querying recommender systems. By contrast, work [33] recommended packages of variable size, subject to budgetary constraints, with the associations among entities to be captured using the notion of compatibility of sets. This system combined recommendations from systems that provided ratings for items. Work [63] proposed a novel framework for automatically generating itineraries from user-generated data such as uploaded pictures. CourseRank [32, 64] is a package recommender system that provides a minimal number of courses to students that will satisfy requirements and accord with past ratings of the courses. FlexRecs [29] is a package recommender system that generates complex recommendations from relational data and specified recommendation requirements by using relational algebra extended with additional features and operators. The CARD [34] and COD [35] frameworks support packages of product and service definitions and provide recommendations based on dynamic preference learning and decision optimization. The packages of services in CARD are characterized by a set of sub-services, which in turn can be package or atomic. CARD uses a decision-guidance query language (DGQL) to define recommendation views, which specifies multiple utility metrics, in addition to the weighted utility function. COD was based on CARD and provides an efficient method for eliciting individuals’ utility functions.

However, all of the above systems recommend packages of items for individuals rather than groups. There has been little research on group package recommendations.

One instance is Travel Decision Forum [23], which allows each group member to view the preferences of other members to help them reach an agreement on the desired features of a joint holiday. And CATS [45] is a critique-based group recommender that helps groups of users plan a joint ski holiday by letting them view ski packages and critique their features, and then recommending a new ski package in response to these critiques. Intrigue [21] recommends packages of tourist attractions to groups by using the weighted average strategy to take into account the preferences of relatively homogeneous subgroups, such as children. However, these recommenders do not make use of dynamic preference-learning or decision optimization.

2.3 Group Recommender Systems

There are many activities that can be done by groups of users, such as watching a movie, listening to music, or travelling. For these kinds of activities, a recommender should suggest items that satisfy the preferences of the whole group based on the individual preferences of its members. Such a recommender system is known as a group recommender system (GRS).

A GRS is a system that attempts to recommend items that will satisfy a group of users with potentially competing interests [65]. Thus the success of a given recommendation depends not on the individual user's interests but on those of the group as a whole. This makes GRSs more complex than individual recommenders, as described in [23]. One reason for this complexity is the need to effectively aggregate users' preferences in a way that maximizes the group's satisfaction, is fair, and is easy to use.

Recently, researchers have proposed group recommenders in different domains

and applications that use different strategies to aggregate individual preferences into group models. Common uses of GRS include recommending TV programs, movies, and video clips (e.g., [6-12], [16]), finding songs to play in shared spaces (e.g., [13-15]), recommending meal recipes for families (e.g., [17]), recommending photos (e.g., [18]), recommending web pages (e.g., [19, 20]), and finding tourist attractions for groups of people (e.g., [16, 21-25]). The most popular GRSs in different domains are presented in the following subsection.

2.3.1 Overview of Existing Group Recommender Systems

A large number of works have addressed group recommenders in different domains in the past two decades. For example, PolyLens [6] is a group movie recommender that is extended from the MovieLens system. It uses the least misery strategy, which takes the minimums of individual ratings to avoid causing “misery” for members. The authors addressed some important issues for group recommenders, such as groups’ privacy, members’ rights, and social value functions. MusicFx [13] is a group recommender that chooses background music to suit a group in a fitness center. To aggregate a group preference, it uses an average without the minimum rating. Pocket Restaurant Finder [25] recommends restaurants for groups of people going out to eat together on the basis of the users’ locations and the restaurants’ characteristics (distance, cost, cuisine). Intrigue [21] recommends tourist attractions to groups of users by following the weighted average strategy and using socio-demographic information about the participants. It takes into account the preferences of relatively homogeneous subgroups, such as children, in which each subgroup may have a different degree of

influence on the estimation of group preferences. Yu's TV Recommender [10] selects a TV program for a group of users depending on the average of their ratings of program features. Travel Decision Forum [23] allows each group members to view the preferences of other members to help the group reach an agreement on the desired features of a joint holiday. I-Spy [20] is a community-based search engine that adapts queries and re-ranks its search results on the basis of community choices. E-Tourism [22] is a web-based service for recommending of group tourism activities. To compute group profiles, it uses three mechanisms: aggregation, intersection, and incremental intersection. CATS [45] is a critique-based group recommender that helps groups plan ski holidays by letting individual members view ski packages and critique their features and then recommending new packages in response to these critiques. Work [11] proposes voting mechanism for recommending TV shows to groups of people. It focuses on range voting, in which users give items ratings within a specified range, and the item with the greatest total rating is recommended to the group.

Some recent group recommenders have been implemented on Facebook. For example, GroupFun [14] recommends a common set of music items to groups. It uses voting algorithms to determine users' true preferences and aggregates those using the probabilistic weighted sum method. Happy Movie [7] is another application on Facebook that recommends movies to groups of users.

2.3.2 Group Modeling Approaches

As explained in popular group recommender surveys (e.g. [44, 49, 51-53]), there are two main approaches to group modeling: (a) aggregating individual predictions

(ratings or recommendations) into group predictions, and (b) aggregating individual preference models (profiles) into a model representing the preferences of the group as a whole. For both methods, there are many different strategies for the aggregation process (see Section 2.4).

Aggregating individual predictions (e.g., [6, 7, 45, 66]) is based on aggregating lists of recommendations for individual members of a group. There are two ways of doing this: the CF method, which is explained in Section 2.1, and the rank aggregation method, which generates recommendation lists for each individual and merges these into a single recommendation list for the group by using one of the social choice strategies for combining multiple rankings (see Section 2.4). The approach of aggregating individual predictions can be formulated as follows [67]:

- For each member m_j , predict the rating r_{ij} of each candidate c_i by m_j .
- Compute an aggregate rating R_i from the set $\{r_{ij}\}$.
- Recommend the set of candidates with the highest predicted ratings R_i .

Many authors (e.g., [8, 17, 44, 48, 49, 51]) agree that merging individuals' recommendations provides the worst GRS results in general. The problem of optimal solution search is the reason this approach is not used nowadays. However, it is useful when applied as an extension to the existing recommender systems, so that minimal changes to those systems are needed [48]. Some existing GRSs use this approach: PolyLens [6] and gRecs [66] for group movie recommendations, CATS [45] for group ski holidays, and [7] for group movie recommendations via Facebook.

Aggregating individual preference models (e.g., [13, 20, 24, 68]) is based on

merging individuals' profiles. On this approach, user profiles are usually represented as sets of weighted preferences or as sets of personal scores assigned by group members to the existing items [52]. Predictions for individual users are not needed in this approach. Instead, an aggregate model is built for the preferences of the group as a whole [67]. This approach can be formulated as follows [67]:

- Compute an aggregate preference model M that represents the whole group's preferences.
- For each candidate c_i , use model M to predict the whole group's rating R_i .
- Recommend the set of candidates with the highest predicted ratings R_i .

The preference-aggregation approach provides better results in general because multiple aspects of the group can be considered. In addition, it resolves some privacy concerns because individual users' preferences are not made visible [52]. Most contemporary GRSs use this approach rather than aggregating individual recommendations: Intrigue [24] recommends tourist attractions by dividing each tour group into homogeneous subgroups and specifying a preference model for each subgroup. The group model is built by computing the weighted average of the subgroup models, taking into account the importance of each subgroup. MusicFX [13], which chooses background music for fitness centers, computes a group preference for any given genre of music by taking the average of the squares of the users' ratings of that genre. I-Spy [20], a collaborative search framework, creates a group preference model without creating any individual preferences models first. It adapts search queries and re-ranks the results while taking into account the community (group) choices. HbbTV [68] is designed

for HbbTV browser, and its recommendations are based on user preferences and several filtering algorithms.

In the following section, I explain the group decision strategies based on social choice theory that are the most used in existing GRSs.

2.4 Overview of Voting Methods (Aggregation Strategies)

The main difficulty for group recommendation is effectively aggregating users' preferences into a common social welfare function that will maximize the group's satisfaction. Usually a number of different voting methods, called "aggregation strategies," "social choice rules," or "group decision rules," can be used to solve this problem. Eleven aggregation strategies are explained in works [44, 52, 69]. In addition, other lists of aggregation strategies called "group decision rules" are explained in works [56, 70]. I summarize most of them in Table 2.1.

Table 2.1: Overview of Voting Methods (Aggregation Strategies)

Voting Method (Aggregation Strategy)	Summary of How It Works
Majority Voting	The item receiving more than 50% of the votes is selected for the group.
Plurality Voting	The item with the most votes is selected for the group.
Plurality with Elimination	Proceed in (n-1) rounds. After each round, the least preferred item is eliminated; those who voted for it have to vote again, for a remaining item.
Instant Runoff Voting (IRV)	Items are ranked in members' preference lists. The least preferred item is eliminated, and any votes for that item are redistributed to the voters' next choices. This continues until an item has a majority (over 50%).

Borda Count	Points are counted from items' rankings in members' preference lists, with the bottom item getting 0, the next up getting 1, and so on. The item with the most points is selected for the group.
Condorcet Winner	If there is an item that is preferred in every one-to-one comparison with other items, that item is the Condorcet winner and should be selected for the group.
Copeland Method	For each item, count the number of times it beats other items (using majority votes) and subtract the number of times it loses. The item with the highest result is selected for the group.
Approval Voting	For each item, count the members who rate it above a chosen approval threshold. Selects the item with the highest result value for the group.
Group Satisfying Rule	On each trial, items are considered one at a time in a random order. The first items for which all individual ratings exceed a certain threshold is selected.
Average	Individuals' ratings for each item are averaged, and the one with the highest average is ranked first in the group's ranked list.
Weighted Average	Weights are attached to individual ratings for each item, and the item with the highest weighted average is ranked first in the group's ranked list.
Additive	Individuals' ratings for each item are added together, and the one with the highest total value is ranked first in the group's ranked list.
Multiplicative	Individuals' ratings for each item are multiplied together, and the item with the highest product is ranked first in the group's ranked list.
Median Winner	The median value of individuals' ratings for each item is computed, and the item with the highest median is ranked first in the group's ranked list.
Least Misery	For each item, the lowest of the individuals' ratings is selected, and the item with the highest minimum is ranked first in the group's ranked list.
Average Without Misery	Items with individual ratings below a certain threshold are excluded, and then individual ratings for each remaining item are averaged. The item with the highest average is ranked first in the group's ranked list.
Most Pleasure	For each item, the highest of the individuals' ratings is selected, and the item with the highest maximum is ranked first in the group's ranked list.
Fairness/Random Dictator Rule	Items are ranked as if members are choosing them in turn. On each turn, one member is selected randomly, and his first

	choices become the group's choices.
Most Respected Person/Dictatorship	For each item, the rating of the most respected member of the group is used as the group's rating for that item.

2.4.1 Aggregation Strategies Used in Related Work

In [69], Masthoff presents and evaluates some group aggregation strategies (most summarized in Table 2.1) in a TV-recommendation domain with a small group of users. Here I review the most common strategies for GRS, including the additive, average, least misery, average without misery, and most pleasure strategies. In addition, I cite representative GRSs that use each and illustrate the differences among them with an example, which assumes that each user has a preference rating, from 1 (really hate) to 10 (really like), for each item.

- **Additive strategy.** This strategy adds individuals' ratings for each item, and the item with the highest total value is ranked first in the group's ranked list (see Table 2.2). A major problem with this strategy is the fact that individuals' opinions become less significant as the group gets larger [52].

Pocket Restaurant Finder [25] uses the additive strategy to recommend restaurants to groups of people going out to eat, on the basis of their locations and the restaurants' characteristics, such as distance, cost, and cuisine.

- **Average strategy.** This strategy averages individuals' ratings for each item and sets the one with the highest average first in the group's ranked list (see Table 2.2). The main problem with this strategy is that it can give a high score to an

item that is disliked highly by some members if it is also liked highly by enough others.

Travel Decision Forum [23] uses multiple aggregation strategies, including the average strategy and the median strategy, which takes the middle value of the users' ratings rather than the average. It allows each group member to view the others' preferences to help the group reach an agreement on the desired features of a joint holiday. In addition, Yu's TV Recommender [10] selects a TV program for a group of users depending on the average of the individuals' ratings of program features.

In this strategy, weight can be assigned to individual preferences based on multiple criteria for some members in the group. For example, Intrigue [21] recommends tourist attractions to groups of users by assigning particular users' ratings weights based on socio-demographic information about the participants. It takes into account the preferences of relatively homogeneous subgroups, such as children, in which each subgroup may have a different degree of influence on the estimation of the group's preferences. Note that the group ranking list for the average strategy will be the same as for the additive strategy if no weights are assigned to users' ratings.

- **Least misery strategy.** This strategy takes the lowest individual rating for each item and ranks the item with the highest minimum first in the group's ranked list, so that a group is as satisfied as its least-satisfied member (see Table 2.2). This avoids "misery" for all members. The main problem with this strategy is it will

miss an item that would be liked by all but one member of the group.

PolyLens [6] is a group movie recommender that uses the least misery strategy.

- **Average without misery strategy.** This strategy averages individual ratings, like the average strategy, with the difference that items with individual ratings below a certain threshold are excluded from the group recommendations. Table 2.2 shows an example of preference aggregation using this strategy with a threshold value of 4 [52]. The main problem with this strategy is the same as with the least misery strategy: even if many members like a certain item, if one member really hates it, it will not appear in the group's recommendations.

MusicFx [13] is a group recommender that selects background music to suit a group in a fitness center. It uses the average without misery method to aggregate the group preference. The CATS system [45] also applies the misery strategy.

- **Most pleasure strategy.** This strategy takes the maximum of the individual ratings for each item and ranks the item with the highest maximum first in the group's ranked list, so that a group is as satisfied as its most satisfied member (see Table 2.2). The main problem with this strategy is that it can select an item for the group even if some members really hate it. However, this strategy may work well when the group members' social relationships are strong and tight (e.g., couples or close friends), as explained in work [46].

Table 2.2: Example of Group Ratings Using Five Different Strategies

Users		Items									
		I ₁	I ₂	I ₃	I ₄	I ₅	I ₆	I ₇	I ₈	I ₉	I ₁₀
U ₁		10	4	3	6	10	9	6	8	10	8
U ₂		1	9	8	9	7	9	6	9	3	8
U ₃		10	5	2	7	9	8	5	6	7	6
Group Ratings	Additive	21	18	13	22	26	26	17	23	20	22
	Average	7	6	4.3	7.3	8.7	8.7	5.7	7.7	6.7	7.3
	Least Misery	1	4	2	6	7	8	5	6	3	6
	Average Without Misery	-	6	-	7.3	8.7	8.7	5.7	7.7	-	7.3
	Most Pleasure	10	9	8	9	10	9	6	9	10	8
Group Ranked List	Additive	(I ₅ -I ₆ , I ₈ , I ₄ -I ₁₀ , I ₁ , I ₉ , I ₂ , I ₇ , I ₃)									
	Average	(I ₅ -I ₆ , I ₈ , I ₄ -I ₁₀ , I ₁ , I ₉ , I ₂ , I ₇ , I ₃)									
	Least Misery	(I ₆ , I ₅ , I ₄ -I ₈ -I ₁₀ , I ₇ , I ₂ , I ₉ , I ₃ , I ₁)									
	Average Without Misery	(I ₅ -I ₆ , I ₈ , I ₄ -I ₁₀ , I ₂ , I ₇)									
	Most Pleasure	(I ₁ -I ₅ -I ₉ , I ₂ -I ₄ -I ₆ -I ₈ , I ₃ -I ₁₀ , I ₇)									

In addition to these strategies, I review two other voting methods that I use in my research: the instant runoff voting (IRV) and the Condorcet voting methods.

- **Instant runoff voting (IRV).** If there exists an alternative that has majority support (more than 50%), then it is selected for the whole group of voters. Otherwise, the alternative with the fewest first-place votes is eliminated from the election, and votes for it are redistributed to the voters' next choices [71, 72]. If an exact tie exists for last place, the tied option that had the fewest votes in the previous round is eliminated.

The IRV method weakens the need to vote strategically for an alternative that is not a voter's first choice but has a better chance of winning, because

second and third choices still count if first choices are eliminated. However, it is well known that IRV does not meet the Condorcet winner criterion: it can result in a Condorcet winner being excluded from the choice set.

- **Condorcet-winner voting.** If there is an item that is preferred in every one-to-one comparison with other items, that item is the Condorcet winner and should be selected for the group. Although this method does make the Condorcet winner impervious to elimination, it ends in ties fairly easily.

2.4.2 Which Strategy Performs Best

Masthoff [44] conducted a number of experiments to determine which strategy is the best to use. She found that users cared about promoting fairness and about preventing misery. She also found that participants' decisions reflected several strategies, including average, least misery, and average without misery; while other strategies, such as Borda count and Copeland rule, were clearly not used.

Furthermore, she presented some item sequences chosen by the different aggregation strategies and asked participants to rate how satisfied they thought the group members would be with those sequences. She found that the multiplicative strategy worked the best, as all participants thought its sequences would keep all members satisfied. She judged that some strategies, including the Copeland rule, plurality voting, and least misery, could be discarded, as they were clearly found to make group members unhappy.

Work [12] evaluated some of the aggregation strategies on a large data set of TV viewing patterns. It found that the average, additive, and average without misery

strategies (the “consensus-based” strategies) provided the best recommendation results. In addition, work [18] evaluated their approach using different social choice strategies and like work [12] found that the average, additive, average without misery, and fairness strategies outperformed the least misery, most pleasure, and plurality voting strategies (the “borderline” strategies).

However, there is no perfect voting method that is fully fair. Mathematical economist Kenneth Arrow proved in 1950 [73] that there is no voting method that satisfies all consensus-desirable properties (fairness criteria), such as the following [71]:

- **The Condorcet criterion.** If there is a choice that is preferred in every one-to-one comparison with the other choices, that choice should be the winner. (Plurality voting, Borda count, and IRV methods violate this criterion.)
- **The majority criterion.** If a choice has a majority (more than 50%) of first-place votes, it should be the winner. (Plurality voting and Borda count methods violate this criterion.)
- **The monotonicity criterion.** If voters change their votes to increase the preference for a candidate, it should not harm that candidate’s chances of winning. (The IRV method violates this criterion.)
- **The independence of irrelevant alternatives (IIA) criterion.** If a non-winning candidate is removed from the ballot, it should not change the winner of the election. (The plurality voting, Borda count, IRV, and Copeland methods violate this criterion.)

It is because of Arrow’s impossibility theorem that different voting methods are

still used. The choice of which method to use usually depends on which property people want to satisfy and what seems most fair in the situation [71].

Several other properties of aggregation strategies in GRS are also important:

- Allowing the aggregation strategy to reflect the unequal influence of different individuals in the group depending on the situation, such as its being one user's birthday. The most respected person (dictatorship) strategy seems to reflect this goal, but other users' preferences should be taken into account too [74].
- Discouraging manipulation of the recommendation process. The IRV method is relatively resistant to strategic manipulation, as explained in [75].
- Taking characteristics of the group into account, such as its size and homogeneity.

2.5 Social Factors in Group Recommender Systems

Regarding which aggregation strategies should be used in group recommender modeling, several researchers (e.g., [44, 46-49]) have concluded that taking certain main factors into account when developing the GRS yields a significant improvement. These factors include the influence of group size, the influence of group homogeneity, the personality of each member, and the relationships among them within the group.

There is extensive work addressing group recommendations in different domains. MusicFx [13] recommends background music for gyms after users specify their preferences for specific genres. No group structure is considered, and only genres, not individual songs, are considered. PolyLens [6] is a small-group extension for the well known single-user recommender MovieLens. It recommends movies to groups while trying to satisfy at least the least-satisfied members, using the least misery aggregation

strategy. Pocket Restaurant Finder [25] recommends restaurants to a groups going out to eat together, generating its recommendations via the additive aggregation strategy. Yu's TV [10] is a TV recommender that uses a vector-space model of TV programs' features to find recommendations for groups. I-Spy [20] is a community-based search engine that adapts search queries and re-ranks the results in response to community choices. Every member is treated equally, which does not reflect real-life situations. FlyTrap [76] is recommends music to be played in public rooms. Its interface lets users control and easily see the reasons for its recommendations.

All the GRSs above consider each group member equally: members' personalities and the way each behaves in a group decision-making process are not taken into account. In addition, the relationships among group members and the possibly different relevance of members' preferences have not been taken into account. The work on these issues is limited.

Several existing GRSs have considered some of the main social factors when generating recommendations for groups of people [49]. For example, Work [17] is a family-based recipe recommender that focuses on the most appropriate recommendation strategy and user-weighting model. Its evaluation showed that the best performance of group recommendations is obtained when the individual data of group members are aggregated in a weighted manner. However, like most previous work, it focused on the content-relevance of the group members and ignored key characteristics of the group, such as its size and interest dissimilarity among its members, which resulted in sub-optimal recommendations. Travel Decision Forum [23] makes recommendations that take

into account the preferences of each member and reflect interactions among members by allowing each member to view the preferences of the others, in order to help the group reach an agreement on desired features. CATS [45] is a critique-based group recommender for ski holidays that lets users to view ski packages and recommends new packages based on their critiques. This work presumes that the members' current preferences depend on the preferences and behavior of other members. However, CATS users have to read the information on other members' choices to adjust their initial opinions, and this is possible only for users who vote later. Intrigue [21] recommends tourist attractions to groups by assigning weights to particular users' ratings on the basis of socio-demographic information about them. It takes into account the preferences of relatively homogeneous subgroups and allows each subgroup to have a different influence on the estimation of the group preferences.

Other works have focused not only on members' relevance preferences, but also on their disagreements over items. For example, GRec-OC [50] recommends books for online communities and tries to reduce individual members' dissatisfaction; its recommendations are based on what similar groups have purchased. By using rank aggregation techniques, work [8] addressed interest similarity and dissimilarity among group members and found that the more alike the members are, the more effective the group recommendations are. It also addressed the affect of the group's size on the GRS and showed that group recommendations are less effective than individual recommendations only for large groups (of size 8+) and that the difference is in fact very small for groups of moderate size (2, 3, and 4). Work [46] proposed a group

recommendation method that examines the key characteristics of groups and proposes a group consensus function for capturing social, expertise, and interest dissimilarity among group members. In work [47], the proposed GRS takes into account both an item's relevance to the group and the disagreements among group members.

However, none of the above GRSs was designed for packages of products and services, which would make the recommendation space very large and implicitly defined. It is necessary to modify the existing aggregation strategies to reflect other aspects, such as group size, within-group similarities, and each member's role in the group. Thus the group decision-making method developed in my research reflects the influence of each member on the group decision-making process, the size of the group, and the group's homogeneity to provide its composite group recommendations.

2.6 Multi-Criteria Recommender Systems

The majority of current recommender systems consider a single criterion, such as a single rating of an item by a user. While such systems provide successful recommendations in several applications, some recent works (e.g., [5, 38, 77]) have agreed that the incorporation of multiple criteria may produce more accurate recommendations because more complex users' preferences can be represented [77].

Multi-criteria decision-making (MCDM) techniques have been extensively studied in the field of decision science. The two main families of MCDM techniques are those based on multi-attribute utility theory (MAUT) [78, 79] and those based on the outranking methods [80, 81].

The MAUT methods are based on aggregating the different criteria into a function

that has to be maximized. They include the simple multi-attribute rating technique (SMART) [82, 83], the simplest of the MAUT methods; the technique for order preference by similarity to ideal solution (TOPSIS) [84]; and the analytic hierarchy process (AHP) [85], which uses pairwise comparisons to determine the criterion weights. The two main families of outranking methods are the ELECTRE [80, 81] and PROMETHEE methods [86, 87].

The vast majority of multi-criteria recommender systems (e.g., [35, 38, 39, 88]) take the MAUT approach, which provides prediction in the form of additive utility functions. Only a few (e.g., [40, 89]) use multi-objective mathematical programming methodologies, in which the goal is finding the Pareto-optimal solution to the optimization problem. As with the previous approach, only few multi-criteria recommender systems (e.g., [90]) use the outranking relations approach, on which preferences are expressed as a system of outranking relations between items.

Note that although this dissertation focuses on composite products and services and support for groups of users, all the examples of multi-criteria recommender systems above focus on atomic products and services and on individual users.

There are several approaches to aggregating individuals' utility functions. Some earlier MAUT methods of group decision are reviewed in [41] and used in a number of works: the simple additive theory for aggregating individuals' utility functions, proposed in [43]; the simple additive weighting (SAW) in [91]; the MAUT-group decision model in [92], which considers both preferential differences and preferential priorities to the model construction; and the use of weighted algebraic means, which is applied in the

WINGDSS software [93]. The group utility function computed in WINGDSS is appropriate in respect of satisfying the axioms given in [94].

In this research, I extract the group utility function based on the additive multi-attribute group utility function proposed on earlier literature (e.g., [94, 95]) to aggregate the individuals' utility functions. However, the aggregated utility function is only an approximation, and using it directly may limit the flexibility of decision makers to refine their choices.

In addition, the AHP [96] and outranking methods are extended to group decision support. For example, work [97] developed a PROMETHEE technique for group decision support, and [98] developed a method for group decision support based on ELECTRE methodology. However, most of the techniques mentioned above focus on decision-making problems in which the number of alternatives is finite and the alternatives are explicitly defined.

2.6.1 Recommendation as a Multi-Criteria Decision-Making Problem

Different multi-criteria decision-making (MCDM) methods can be applied to support multi-criteria recommender systems. These methods follow several steps [77], including:

- **Defining the object of decision**, which is an item in recommender systems.
- **Defining a consistent family of criteria**, which may be the multiple features of an item or the multiple dimensions along which it is being rated.
- **Developing a global preference model**, which provides a way to aggregate the criteria values to express preferences between different items.

The existing multi-criteria recommender systems follow several approaches to developing the global preference models [5]:

- **Value-focused models.** Marginal preferences on each criterion are synthesized into a utility function. These approaches are often called multi-attribute utility theory approaches. The vast majority of multi-criteria recommender systems (e.g., [35, 38, 39, 88]) use these approaches, which provide predictions in the form of additive utility functions.
- **Multi-objective optimization models.** The goal of these approaches is to find a Pareto-optimal solution to the optimization problem. They are also called multi-objective mathematical programming methodologies. Only a few multi-criteria recommender systems (e.g., [40, 89]) take this approach.
- **Outranking relations models.** Preferences are expressed as a system of outranking relations between items. As with the previous approach, only a few multi-criteria recommender systems (e.g., [90]) take this approach.

Note that while this research focuses on composite products and services and support for group of users, all the examples above focus on atomic products and services and on individual users.

2.6.2 Multi-Criteria Optimization

Cases in which there are a finite number of criteria and an infinite number of alternatives belong to the field of multi-criteria optimization. Multi-criteria optimization problems have rarely been addressed in the context of recommender systems. This approach helps decision makers choose the best alternatives when multiple criteria are in

conflict with each other. The following methods are often used to address multi-criteria optimization problems and can be applied in multi-criteria recommender systems, as discussed in [1, 77]:

- Finding Pareto-optimal solutions.
- Reducing the problem to a single-criterion optimization problem by taking a linear combination of multiple criteria.
- Optimizing the most important criterion and converting the other criteria to constraints.
- Optimizing one criterion at a time, converting the optimal solution to a constraint, and repeating the process for other criteria.

Few multi-criteria recommender systems have roots in multi-criteria optimization techniques; one example is work [39], which estimates the overall utility of a specific item for each user by adding the marginal utility of each criterion. Its model evaluation demonstrates that multi-criteria rating provides measurable improvements in modeling users' preferences. Similarly, work [40] proposes a method for calculating the total utility of an item by using an aggregation function. After a total ordering has been established on the items, the system recommends to each user the items that maximize this total utility. Both of these systems, however, focus on atomic products and individual users.

The CARD [34] and COD frameworks [35] do support packages of product and service definitions, and both provide recommendations based on multi-criteria optimization. CARD uses a decision-guidance query language (DGQL) [99, 100] to define recommendation views, which specify multiple utility metrics and the weighted

utility function. The CARD packages of services are described using the constraint representation, following [101-106]. COD is based on CARD and provides an efficient method of eliciting utility functions for individuals that are optimized to find diverse, multi-criteria recommendations. However, both CARD and COD are recommender systems for individuals rather than groups.

2.7 Summary of the Evaluation of Related Work

Extensive work has been done on recommender systems, most of it focused on single users rather than groups (e.g., [1-5]). Recently, researchers have proposed group recommenders in different domains and applications, using different strategies to aggregate individual preferences into group models (e.g., [6-25]). However, none of the above GRSs was designed for packages, which makes the recommendation space very large and implicitly defined.

More recently, there has been a great deal of research supporting package recommendations [26-33], but these studies do not consider dynamic preference learning and decision optimization. The CARD [34] and COD [35] frameworks support packages and provide recommendations based on dynamic preference learning and decision optimization, but both are recommender systems for individuals rather than groups.

In addition, the majority of recommender systems rely on a single ranking or utility score, whereas in many applications multiple criteria need to be considered, such as cost, quality, enjoyment, satisfaction, and risk. Multi-criteria ranking has recently been explored in recommendation-set retrieval [5, 36-38], but the methods in these studies were based on distance measures to increase the quality of each individual

recommendation, which competes with the ability diversify recommendations [34]. In addition, they focused on individual users rather than groups.

A few existing multi-criteria recommender systems have roots in multi-criteria optimization techniques (e.g., [39, 40]), but these systems focus on atomic products and on individual users rather than groups. Furthermore, most previous work is based on aggregation strategies that always combine members' ratings in the same way, without considering how group members interact with each other. In fact, it is natural for some members to have more influence than others on the aggregation of preferences—for example, users who have more authority or more expertise or who are more trusted. These members must be treated differently to improve the group decision-making process.

Regarding which aggregation strategies should be used in group recommender modeling, several researchers (e.g., [44, 46-49]) have concluded that taking certain factors into account while developing the GRS yields a significant improvement. These factors include the influence of the group size, the influence of group homogeneity, the personality of each member, and the relationships among them within the group. Several GRSs have considered some major social factors when generating recommendations for groups (e.g., [8, 17, 21, 23, 45-47, 50]), but none of these was designed for package recommendation, which makes the recommendation space very large and implicitly defined.

In the following section, I present a decision-guided group composite alternative recommender framework based on multi-criteria decision optimization and voting to

address the outlined limitations. This framework extends existing recommender systems in three ways: (1) to consider composite recommendations; (2) to deal with multiple criteria for recommendations, and (3) to support groups of users.

CHAPTER 3: GROUP COMPOSITE ALTERNATIVES RECOMMENDER (GCAR)

3.1 Introduction

As explained in Chapter 2, most of the work on recommender systems focuses on atomic products and services and on individual users. In this chapter, I focus on extending recommender systems in three ways: (1) to consider composite recommendations; (2) to deal with multiple criteria associated with recommendations; and, most importantly, (3) to support groups of users rather than individual users. Examples of this new class of recommender systems include recommenders for group-travel packages, public policy and budgets, energy infrastructure investments, and health care plan selection by organizations. Recommendations by these systems are composite; for example, a travel recommendation might involve interrelated air reservations, accommodations, activities, and car rentals. They are associated with multiple criteria, such as cost, benefit, enjoyment, satisfaction, and risk. Finally, there is often a need to support a group of diverse users who may have conflicting views on weights for different criteria. The challenges for GRSs are considerably more complex than for individual user recommenders, as explained in Chapter 1. One reason for this is the need to effectively aggregate users' preferences so as to maximize the group's satisfaction, fairness, and user-friendliness.

Addressing these challenges is the focus of this chapter. Here I develop and

propose the decision-guided group composite alternatives recommender (GCAR) framework based on multi-criteria decision optimization and voting to address the outlined limitations. The GCAR framework works on a very large recommendation space, which is implicitly defined by mathematical constraints. I consider six group decision-making methods, including three based on well known and commonly used aggregation strategies, the average, least misery, and average without misery strategies [44]. Two others are existing voting methods based on individuals' rankings: instant runoff voting (IRV) and a hybrid Condorcet-IRV method. I also develop a new method, the structurally adjusted average method, to take into account the influence of decision makers within the group and the dissimilarity of opinions among them.

However, group decision-making methods can be applied only when there are a small number of alternatives to vote on. In the case of composite alternatives, the search space is exponentially large, even infinite if some choices are continuous. It is thus impractical to use a voting method on such space directly. The idea of the proposed GCAR framework is to filter the very large recommendation space into a very small but diverse set of near-optimal choices, which can then be refined through group decision-making methods. On one hand, it is important that these alternatives be close to optimal in terms of the estimated group utility function. On the other hand, because this function is only an estimate, it is important to have alternatives that are reasonably diverse in terms of individual decision makers' preferences. To achieve this, I follow six steps: (1) Elicit the utility function for each member of the group; (2) Estimate the group utility function; (3) Use the group utility function to find an optimal recommendation

alternative; (4) Use diversity layering to generate a diverse set of l recommendations that contains the optimal recommendation alternative; (5) Rank this set for each individual; and (6) Apply a group decision-making method to refine the final top k recommendations. Note that the group utility estimation is parameterized on the basis of the target group decision-making method.

The remainder of this chapter is organized as follows: Section 3.2 presents an overview of the proposed GCAR framework. Section 3.3 gives an overview of group decision-making methods used in the framework and proposes the structurally adjusted average method that I developed. Section 3.4 summarizes the chapter. Some of the work reported in this chapter was published in [107, 108].

3.2 Overview of the Proposed GCAR Framework

In this section, I first describe the recommendation space and then explain the recommendation process implemented by the proposed GCAR framework and the intuition behind this process.

Recommendation space R , consists of composite products and services. Each recommendation alternative $a \in R$ is mapped to a utility vector $\vec{u} = (u_1 \dots, u_n)$ from an n -dimensional utility space such that $\forall_i, 1 \leq i \leq n, u_i: R \rightarrow [0,1]$. The components of a utility vector $\vec{u} = (u_1, u_2, \dots, u_n)$ are associated with criteria—such as enjoyment, saving, and location attractiveness in a traveling domain—that have been previously defined. Each criterion has an associated domain $D_i, 1 \leq i \leq n$, and each domain D_i has a total ordering “better than,” denoted by \succsim_{D_i} . For example, for the domain *Saving*, $a_1 \succsim_{\text{Saving}} a_2 \Leftrightarrow a_1 \geq a_2$.

For a given group of m decision makers, the utility of each decision maker j is denoted by $\forall_j, 1 \leq j \leq m$, $U_j: [0,1]^n \rightarrow [0,1]$, and the group utility is denoted by $U: [0,1]^n \rightarrow [0,1]$.

U_j and U define a utility associated with each alternative $a \in R$. Therefore, the user recommendation alternative utility for recommendation a is defined by $RU_j: R \rightarrow [0,1]$, where $RU_j(a) = U_j(u_1(a), \dots, u_n(a))$, and the group recommendation alternative utility is defined by $RU: R \rightarrow [0,1]$, where $RU(a) = U(u_1(a), \dots, u_n(a))$.

The recommendation process implemented by the proposed GCAR framework is depicted in Figure 3.1.

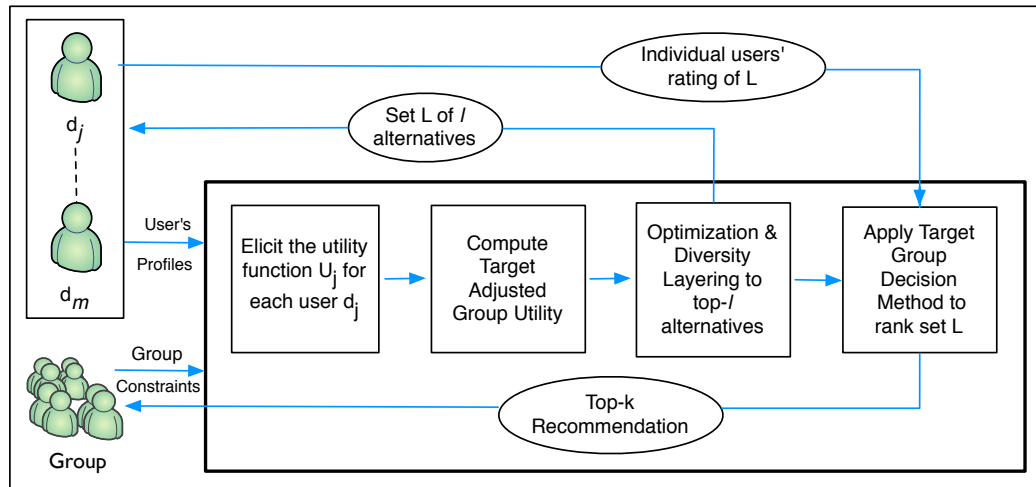


Figure 3.1: Group Composite Alternatives Recommender (GCAR) Framework

As shown in the diagram, the process starts when a group of decision makers submits a request to the group recommender. The request specifies the group's decision

constraints on recommendation alternatives. To generate the top k recommendations, the recommender follows six steps: (1) Elicit the utility function for each member of the group; (2) Estimate the group utility function; (3) Use this to find an optimal recommendation alternative; (4) Use diversity layering to generate a diverse set of l recommendations that contains the optimal alternative; (5) Rank or rate (depending on the method used in the last step) this set by each individual; and (6) Apply a group decision-making method to refine the final k recommendations.

Before discussing these steps in detail, I will describe the intuition behind the process. First, I apply alternative group decision-making methods to make the final recommendations for a group of decision makers. Different group decision-making methods are used by different people, and the choice of method usually depends on the domain, the group's characteristics, and what property people want to satisfy. No single method is considered to be generally superior to all others and fully fair [73].

In this chapter, I consider six group decision-making methods, any one of which can be used to instantiate the framework. Three are based on well known and commonly used aggregation strategies: the average, least misery, and average without misery strategies [44]. Two are voting methods based on individuals' rankings: instant runoff voting (IRV) and the hybrid Condorcet-IRV method. I also develop a new aggregation strategy, the structurally adjusted average method, which takes into account the influence of decision makers within the group and the dissimilarity of opinions among them. (Applying alternative group decision-making methods is the last step of the process in Figure 3.1.)

Group decision-making methods can be applied only when there are a small number of alternatives; composite alternatives, however, make the search space exponentially large or infinite, so that it is impractical to apply group decision-making methods directly. We first need to restrict the large search space to a small, highly relevant set that can then be refined through voting.

To carry out this reduction, I apply mathematical optimization to produce a small set of recommendations that are: (1) close to optimal and (2) sufficiently diverse that group members will have enough flexibility. This explains the second-last step in Figure 3.1 (Optimization and Diversity Layering). However, to complete the optimization and diversification, we need to estimate the group utility function that captures the whole group's preferences. This explains the second step in Figure 3.1. This group utility function is parameterized on the basis of the target group decision-making method, and it must be based on the utility functions of the individual users, which are not known to the system and need to be extracted from the individuals. This is the first step in Figure 3.1.

I now discuss each of these steps in detail, beginning with an overview of the group decision-making methods used in the last step of GCAR framework and leaving the other steps to be explained in the next chapter.

3.3 Overview of Group Decision Methods Used in GCAR

In this section, I review the most commonly used aggregation strategies for GRS and two voting methods based on individuals' rankings. I then propose the new group aggregation strategy that I developed, the structurally adjusted average method.

3.3.1 Aggregation Strategies

To illustrate how the existing aggregation strategies work, consider the following example: Let m be the number of users in a group and r_{ji} the rating of user j for alternative i , ranging from 1 (really hate) to 10 (really like). The group rating for alternative i , denoted by GR_i , is shown in Table 3.1, which can be computed using any of these strategies, as explained in the following subsections. Finally, the alternatives are ranked in descending order on the basis of the resulting group rating values.

Average Strategy

This strategy is the most straightforward. It assumes equal influence of all decision makers in the group. As shown in Table 3.1, it computes the group rating for alternative i by averaging the individual ratings, as follows:

$$GR_i = \frac{1}{m} \left(\sum_{j=1}^m r_{ij} \right) \quad (3.1)$$

Table 3.1 Example to Illustrate the Aggregation Strategies Used in GCAR

Users	Alternatives									
	<i>a1</i>	<i>a2</i>	<i>a3</i>	<i>a4</i>	<i>a5</i>	<i>a6</i>	<i>a7</i>	<i>a8</i>	<i>a9</i>	<i>a10</i>
U₁	10	8	3	7	10	8	6	6	10	6
U₂	1	7	8	6	7	10	6	4	3	7
U₃	10	5	2	8	9	10	5	5	7	9
<i>GR_i</i> (by Average)	7	6.7	4.3	7	8.7	9.3	5.7	5	6.7	7.3
<i>GR_i</i> (by Least Misery)	1	5	2	6	7	8	5	4	3	6
<i>GR_i</i> (by Average Without Misery)	-	6.7	-	7	8.7	9.3	5.7	5	-	7.3

Least Misery Strategy

This aggregation strategy is suitable when the GRS needs to avoid causing “misery,” which it might do by recommending items that are strongly disliked by any member of the group. This strategy computes the group rating for an alternative i as the lowest rating assigned for that alternative by any of the group members, as follows:

$$GR_i = \min_j(r_{ij}) \quad (3.2)$$

Average Without Misery Strategy

This strategy averages individual ratings, like the average strategy, with the difference that alternatives with any rating below a certain threshold are not considered for the group recommendations. For the example in Table 3.1, the threshold rating is 4.

3.3.2 Voting Methods

Because of Arrow’s impossibility theorem [73], which states that no voting method is fully fair, many voting methods are still used. The choice of method usually depends on the property people want to satisfy and what seems most fair in the situation [71]. In the GCAR framework, I consider IRV and a hybrid Condorcet-IRV method. Both are relatively resistant to strategic manipulation [75], which I believe is a critical feature.

But other voting methods, such as Borda, Kemeny, and Copeland, can also be applied in my framework.

Definitions

- *One-to-one comparisons.* Each pair of alternatives is compared to determine which is more preferred. Let $P(a_x, a_y)$ be the number of decision makers who prefer alternative a_x over a_y . If $P(a_x, a_y) > P(a_y, a_x)$, then a_x wins the one-to-one comparison and beats a_y [75] .
- *Condorcet winner criterion.* If some alternative is preferred in every one-to-one comparison, then it should be the winner. Formally: An alternative a_x is a Condorcet winner if and only if $P(a_x, a_y) > P(a_y, a_x)$, $\forall a_y \neq a_x$ [75].

Instant Runoff Voting Method

This is a method in which each voter ranks the alternatives in order of preference. For each alternative, the system then counts the number of voters who ranked it as their first choice. If any alternative has an outright majority, it is selected for the whole group. Otherwise, the alternative with the fewest first-place votes is eliminated, and its votes are redistributed to the voters' next choices. This procedure is repeated until one alternative obtains a majority of votes among the remaining choices [71, 72]. If there is a tie for last place at any stage, special tie-breaking rules are applied to select the alternative to eliminate [71, 109].

This method reduces the need for individuals to vote strategically for alternatives that have a better chance of winning than their actual first choices, because second and

third choices still count if first choices are eliminated.

In the GCAR framework, in order to conclude with a total ordering of the alternatives from which the final top k recommendations are selected and displayed to the group, I use the above IRV method with the difference that the system keeps eliminating last-place alternatives after the winner is declared. The total order associated with IRV is a list of eliminated alternatives ordered by the round in which each was eliminated. If a tie occurs for last place, the following tie-breaking rules apply:

- **Rule 1:** if the number of decision makers who vote for these alternatives as their first choice = 0, (i.e., the alternatives are not the first choices of any decision maker), then, the first alternative to eliminate is randomly selected.
- **Rule 2:** if the number of decision makers who vote for these alternatives as their first choice $\neq 0$, (i.e., the alternatives are the first choice of at least one decision maker), then the tied alternative that received the fewest votes in the previous round is eliminated. If there is still a tie, then look back to the next most recent round, and if necessary to progressively earlier rounds, until one alternative can be eliminated.

To illustrate how IRV works, suppose that we have a group of 9 decision makers who initially ranked a set of 3 recommendations as shown in Table 3.2. Alternative A_2 has the fewest first-choice votes, 2, so it is eliminated in the first round, and everyone's choices must shift to fill in the gaps (see Table 3.3).

Table 3.2 IRV Initial Votes

Total number of voters	4	2	3
1st choice	A ₁	A ₂	A ₃
2nd choice	A ₂	A ₃	A ₂
3rd choice	A ₃	A ₁	A ₁

Table 3.3 IRV Round 1

Total number of voters	4	2	3
1st choice	A ₁	A ₃	A ₃
2nd choice	A ₃	A ₁	A ₁

Finally, A₃ has the majority votes, and wins the election under the IRV method.

By analyzing the initial preferences in Table 3.2, the one-to-one comparisons are as follows:

- A₁ vs. A₂: A₂ beats A₁
- A₁ vs. A₃: A₃ beats A₁
- A₂ vs. A₃: A₂ beats A₃

So even though A₂ had the fewest first-place votes, it is the Condorcet winner, and thus IRV violates the Condorcet criterion. A₂ is preferred in every one-to-one comparison but was eliminated in the first round and lost the election. Applying the standard IRV method on the ranked set of l recommendations to refine the top k recommendations might thus result in a Condorcet winner being excluded from the choice set. To avoid this outcome, a hybrid Condorcet-IRV method can be applied instead.

Hybrid Condorcet-IRV Method

This method makes use of both Condorcet's pairwise comparison principle and the IRV method, like the Bentham method mentioned in work [75] but with some differences. The method checks whether an alternative exists that beats all other alternatives in one-to-one comparisons (a Condorcet winner). If so, that alternative is moved to the first place in a winner list (W); otherwise the IRV method is applied and the losing alternative is moved to an eliminated list (E). This process is repeated until no alternatives remain. The method thus ends with two lists: list W , in descending order by decision makers' preferences, and list E , ordered in the opposite way. By reversing list E 's entries and appending them to list W , we create a list of alternatives in descending order from which we can select the top k recommendations.

Figure 3.2 illustrates how the hybrid Condorcet-IRV method works. Suppose that we have a group of 9 decision makers who initially ranked 5 alternatives as shown in Table 3.4. In one-to-one comparisons, A_2 beats all the other alternatives. Even though it has the fewest first-place votes, it is the Condorcet winner and is moved to list W , and everyone's choices must be shifted to fill in the gaps (see Table 3.5).

In Round 2, there is no Condorcet winner, so the method looks for an alternative with the majority of first-place votes. No alternative has a majority, so the alternative with the fewest first-choice votes, A_4 , is moved to list E (see Table 3.6).

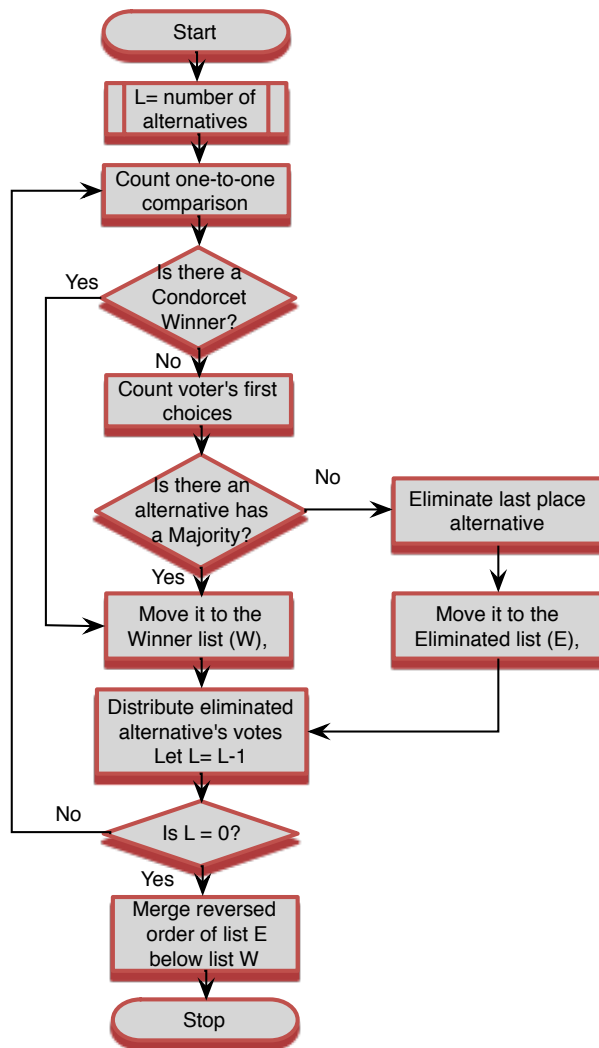


Figure 3.2: Flowchart Illustrates the Hybrid Condorcet-IRV Method

Table 3.4: Hybrid Condorcet-IRV, Initial Votes

Total number of voters	4	2	3
1st choice	A ₁	A ₂	A ₅
2nd choice	A ₂	A ₃	A ₄
3rd choice	A ₃	A ₅	A ₂
4th choice	A ₄	A ₁	A ₁
5th choice	A ₅	A ₄	A ₃

Table 3.5: Hybrid Condorcet-IRV, Round 1

Total number of voters	4	2	3
1st choice	A ₁	A ₃	A ₅
2nd choice	A ₃	A ₅	A ₄
3rd choice	A ₄	A ₁	A ₁
4th choice	A ₅	A ₄	A ₃

Table 3.6: Hybrid Condorcet-IRV, Round 2

Total number of voters	4	2	3
1st choice	A ₁	A ₃	A ₅
2nd choice	A ₃	A ₅	A ₁
3rd choice	A ₅	A ₁	A ₃

In Round 3, there is neither a Condorcet winner nor a majority winner, so A₃ is eliminated and moved to list *E* (see Table 3.7).

Table 3.7: Hybrid Condorcet-IRV, Round 3

Total number of voters	4	2	3
1st choice	A ₁	A ₅	A ₅
2nd choice	A ₅	A ₁	A ₁

In Round 4, A₅ is the Condorcet winner and is moved to list *W*. At that point only A₁ remains, so it is moved to list *W* in the final round. The method ends with list *W* ordered as A₂, A₅, A₁, and list *E* ordered as A₄, A₃. By reversing the order of the alternatives in list *E* and merging them with the alternatives in list *W*, we create an

ordered recommendation list, $A_2 \succ A_5 \succ A_1 \succ A_3 \succ A_4$, from which we can select the top k recommendations. As this shows, one of the advantages of the hybrid Condorcet-IRV method is that it makes the Condorcet winner impervious to elimination.

In the following subsection I propose a new group aggregation strategy, which I call the structurally adjusted average method.

3.3.3 The Structurally Adjusted Average Method

When aggregating individuals' ratings into a group rating, this strategy takes into account two main factors: (a) the influence of individuals within the group and (b) the dissimilarity of opinion among group members.

To consider the influence of individuals, I assign users different weights when I aggregate their ratings to compute the group rating. These weights reflect the expertise of the members. Because experts often attempt to persuade other group members, their opinions may be weighted more highly than others'. Depending on the domain of the group recommender, expertise can be divided into different quantitative levels into which each member will be assigned. For example, in the domain of movies, expertise levels will depend on the number of movies a member has watched from a list of popular movies [46]; in travel, levels will be assigned on the basis of average yearly travelling by each member (see Table 3.8).

Table 3.8: Categorization of Expertise Levels Based on Average Yearly Travel

The average yearly travelling	≤ 1	2 - 3	4 - 5	> 5
Expertise level	I	II	III	IV

The normalized expertise level of group member j is defined as

$$E(j, G) = \frac{e_j}{\sum_{u=1}^m e_u} \quad (3.3)$$

where e_u is the absolute expertise level of group member u and the sum of the relative expertise levels of the group = 1.

To take expertise into account, I compute the weighted average of the individual ratings for alternative i as follows:

$$\text{EGR}_i = \frac{1}{m} \left(\sum_{j=1}^m E_j \cdot r_{ij} \right) \quad (3.4)$$

where E_j is the expertise of each decision maker j , $m = |G|$, and r_{ij} is the rating of user j for alternative i , and $0 \leq r_{ij} \leq 1$

As suggested in works [46, 47], the overall group rating of an alternative should reflect the degree of consensus in the ratings for that alternative among the group's members. Thus if the weighted average of the individual ratings is the same for two alternatives, the one that has more similarity among its ratings should have a higher

group rating, in order to avoid misery for some members. Thus suppose that two alternatives, a_1 and a_2 , have the same weighted average of individual ratings, as computed with Equation 3.4, but the group members' similarity of opinion on a_1 is greater than on a_2 , and we want only one of these two alternatives to be included in the small set of top alternatives. Intuitively, we will choose a_1 to avoid causing misery to any members who strongly dislike a_2 . This kind of dissimilarity of opinion on an alternative tends to be more significant the larger the group is.

To describe group dissimilarity over an alternative i , I use the standard deviation,

$$(r_{i1}, \dots, r_{im}) = \sqrt{\frac{1}{m-1} \sum_{j=1}^m (r_{ij} - avg_i)^2} \quad (3.5)$$

where avg_i is the average of all the individual ratings for alternative i .

Finally, to reflect both the influence of individuals within the group and the dissimilarity of opinion among them, I compute the group rating for alternative i as

$$GR_i = EGR_i \cdot (1 - \delta) \quad (3.6)$$

where EGR_i is the weighted average of the individual ratings for alternative I , as defined in Equation 3.4, and δ represents the dissimilarity penalty, defined as

$$\delta = \alpha \cdot \frac{\sigma}{\sigma_{\max}} \quad (3.7)$$

where α , $0 \leq \alpha \leq 1$, is a parameter representing an upper bound on the dissimilarity penalty (see Figure 3.3), σ is the standard deviation, as in Equation 3.5, and σ_{\max} is the maximum possible σ ; that is,

$$\sigma_{\max} = \max \sigma(r_{i1}, \dots, r_{im}) = \frac{1}{2} \sqrt{\frac{m}{m-1}} \quad (3.8)$$

where $0 \leq r_{i1}, \dots, r_{im} \leq 1$, and clearly $0 \leq \sigma \leq \sigma_{\max}$.

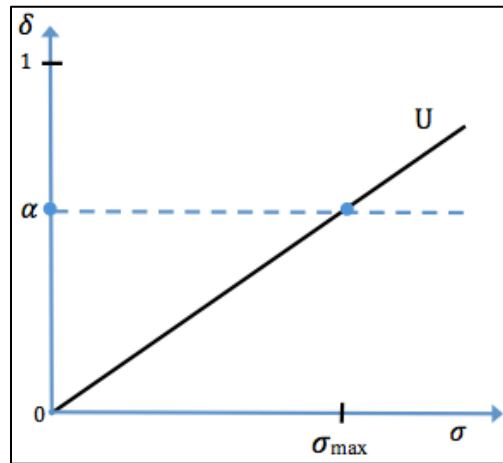


Figure 3.3: Adjusted Group Utility

I explain the other steps of the GCAR framework in the next chapter.

3.4 Summary

In this chapter, I described the proposed decision-guided group composite alternatives recommender (GCAR) framework, which is based on multi-criteria decision optimization and voting. The GCAR framework works on a very large, even infinite, recommendation space, which is implicitly defined by mathematical constraints. GCAR extends existing recommender systems in three ways: (1) it considers composite recommendations, (2) it deals with multiple criteria for recommendations, and (3) it supports a group of diverse users with conflicting views on weights for different criteria.

I considered six group decision-making methods: three based on commonly used aggregation strategies, two voting methods based on individuals' rankings, and a new aggregation strategy that I developed. The idea of this GCAR framework is to filter a very large recommendation space into a small, diverse set of near-optimal alternatives, which can then be refined through group decision-making methods. It is important that these alternatives be nearly optimal in terms of the estimated group utility function but also to be sufficiently diverse to reflect individual decision makers' preferences.

In the following chapter, I discuss each of the GCAR framework's steps in detail. I also explain the preliminary experimental real-world user study that I conducted to evaluate the framework's performance under each group decision-making method.

CHAPTER 4: GCAR STEPS AND EVALUATION UNDER EACH VOTING METHOD

4.1 Introduction

In the previous chapter, I described the decision-guided group composite alternatives recommender (GCAR) framework based on multi-criteria decision optimization and voting.

To generate the top k recommendations, GCAR follows six steps: (1) eliciting a utility function for each group member, (2) estimating the group utility function, (3) using this to find an optimal recommendation, (4) generating a diverse recommendation set containing the optimal recommendation, (5) ranking or rating this set for each individual, and (6) using a group decision-making method to refine the top k recommendations. An overview of the methods used in step 6 was given in the previous chapter. In this chapter, I discuss each of the steps in detail.

I also describe a preliminary experimental study I conducted with the approval of the Human Subjects Review Board (HSRB) of George Mason University to evaluate the GCAR framework's performance with each group decision-making method. In this study, I used alternative methods of modeling the "actual" group preferences in order to fit the decision-making method being used. The study showed that for each decision-making method, the average precision and recall achieved by the proposed GCAR framework for top-1 recommendations are exactly the same as the ideal precision and recall (which are

obtained under the assumption of complete knowledge), and that they are 0–15% off the ideal for top-2 recommendations, 8–27% off for the top 3, and 23–34% off for the top 4.

The remainder of this chapter is organized as follows: Section 4.2 explains the extraction of the user utility functions. Section 4.3 explains the group utility estimation. Section 4.4 presents the optimization and diversity layering. Section 4.5 discusses the preliminary experimental study used to evaluate the framework. Section 4.6 summarizes the chapter. Some of the work reported in this chapter was published in [110].

4.2 Eliciting User Utility Functions

The GCAR framework adopts the COD method [35] for eliciting the utility function of each decision maker. This method starts by representing a number of distinguishable recommendations in terms of utility vectors to each decision maker. Each returned recommendation stretches the dimension it represents (e.g., saving), and relaxes the other dimensions (e.g., enjoyment, attractiveness of location). The process iteratively updates the utility vector in response to the decision maker’s feedback until an exit point is reached (e.g., when there is “no difference” between the recommendations presented). Upon exit, the recommendation space will be constructed according to the utility vector learned.

Recommendation space R , consists of composite products and services. Each recommendation is mapped to a utility vector \vec{u} , from an n -dimensional utility space U , which is presented as R_+^n . This mapping is denoted by $\vec{U} : R \rightarrow R_+^n$. The components of a utility vector $\vec{u} = (u_1, u_2, \dots, u_n)$ are associated with criteria such as enjoyment, saving, or attractiveness of location, that have been previously defined. Each criterion has an

associated domain D_i , $1 \leq i \leq n$, and each domain D_i has a total ordering “better than,” denoted by \succsim_{D_i} . For example, for the domain *Saving*, $a_1 \succsim_{\text{Saving}} a_2 \Leftrightarrow a_1 \geq a_2$.

The relative importance the user places on each dimension is modeled by a vector of weights $\vec{w} = (w_1, w_2, \dots, w_n)$, where $|\vec{w}| = \sqrt{\sum_{i=1}^n w_i^2} = 1$, which is called an axis. Each component w_i captures the weight of the i -th dimension according to a decision maker j . Therefore for each decision maker j , the total utility of a recommendation a_k , with regard to the vector \vec{w}_j , is defined as:

$$U_j(\vec{u}) = w_{j1}u_1 + w_{j2}u_2 + \dots + w_{jn}u_n \quad (4.1)$$

where $\sum_{i=1}^n u_i = 1$.

4.3 Estimating the Group Utility Function

In this research, I assume that the members of a group have agreed on the overall set of criteria but not on their weights. The problem of arriving at a unified set of criteria for multiple decision makers has been studied (e.g., [41, 79, 93]) and is outside the scope of this research.

Estimating the group utility function is the second step in Figure 3.1. Recall that a group utility function $U: [0,1]^n \rightarrow [0,1]$ maps a vector of criteria $u_1 \dots, u_n \in [0,1]$ into a combined group utility $U(u_1 \dots, u_n) \in [0,1]$. This group utility estimation is parameterized on the basis of the target group decision-making method that is applied in the last step, as explained in Chapter 3. I now discuss in detail how I estimate the group utility using each of these methods.

4.3.1 Average Strategy

I use the additive utility function, which has been used in many early works (e.g. [94, 95]) and some recent ones (e.g. [35, 92]). I estimate the group utility of an alternative a_k as follows: for the i -th dimension, the individual weights important to that dimension are aggregated into the group weight w_i by calculating the algebraic mean of the individual weights as:

$$w_i = \frac{1}{m} \left(\sum_{j=1}^m w_{ij} \right) \quad (4.2)$$

where $j = 1, \dots, m$, and m is the number of decision makers in the group. The group utility of alternative a with regard to axis \vec{w}_i is defined as

$$U(\vec{u}) = w_1 u_1 + w_2 u_2 + \dots + w_n u_n \quad (4.3)$$

where $\sum_{i=1}^n w_i = 1$, and $\sum_{i=1}^n u_i = 1$.

4.3.2 Least Misery Strategy

In this strategy, the group utility is computed as the minimum utility value for any alternative among group members, as follows:

$$U(\vec{u}) = \min_j (U_j(\vec{u})) \quad (4.4)$$

where U_j is the utility of decision maker j , $1 \leq j \leq m$, for an alternative a , as defined in Equation 4.1.

4.3.3 Average Without Misery Strategy

In this strategy, the group utility is computed as in the average strategy, but those alternatives with any individual utilities below a certain threshold t are not considered in the group recommendations. More formally,

$$U(\vec{u}) = \frac{1}{m} \sum_j^m U_j(\vec{u}), \quad \text{such that } \forall_j, 1 \leq j \leq m, U_j(\vec{u}) \geq t \quad (4.5)$$

4.3.4 Structurally Adjusted Average Strategy

First, to estimate the group utility while taking the expertise factor into account, I compute the algebraic mean of the individual criteria-weights as

$$w'_i = \frac{1}{m} \left(\sum_{j=1}^m E_j \cdot w_{ij} \right) \quad (4.6)$$

where E_j is the expertise of each decision maker j , $m = |G|$, and w_{ji} is the weight of i -th criterion for individual decision maker j . Then, for an alternative a , I define the weighted group utility that takes the expertise factor into account, denoted by (EU) , as

$$EU(\vec{u}) = w'_1 u_1 + w'_2 u_2 + \cdots + w'_n u_n \quad (4.7)$$

where $\sum_{i=1}^n w_i = 1$, and $\sum_{i=1}^n u_i = 1$.

Second, to describe dissimilarity of opinion among group members over an alternative, I use the standard deviation,

$$\sigma(U_1, \dots, U_m) = \sqrt{\frac{1}{m-1} \sum_{j=1}^m (U_j - AU)^2} \quad (4.8)$$

where $m = |G|$, U_j is the decision maker j 's utility for alternative a , as defined in Equation 4.1, and AU is the average utility for an alternative a , as defined in Equation 4.3.

Finally, to reflect both the influence of individuals within the group and the dissimilarity of opinion among them, I compute the adjusted group utility as:

$$U = EU \cdot (1 - \delta) \quad (4.9)$$

where EU is the weighted group utility defined in Equation 4.7 and δ represents the dissimilarity penalty defined in Equation 3.7.

4.3.5 Instant Runoff Voting (IRV) Method

First, for each decision maker, I rank the set of alternatives in descending order by her extracted utility U_j . Second, I apply the IRV method to obtain the group-ranked list of alternatives. Finally, I estimate the group utility of each alternative $a \in R$ as

$$RU(a) = \frac{n - i}{n - 1} \quad (4.10)$$

where $RU(a) = U(u_1(a), \dots, u_n(a))$, n is the number of ranked alternatives, and i is the position of alternative a in the ranked set resulting from the IRV method.

4.3.6 Hybrid Condorcet-IRV Method

I estimate the group utility of each alternative $a \in R$ in a similar process to the IRV method, except that I apply the hybrid Condorcet-IRV method instead.

4.4 Optimization and Diversity Layering

Because it is not practical for decision makers to focus on more than a small set of alternatives, the goal of this step is to produce this small set. On the one hand, it is

important that the alternatives be nearly optimal in terms of the group utility function. On the other hand, since this function is only an estimate, it is also important that the alternatives be diverse in terms of the individual decision makers' preferences.

Note that the optimal choices according to the estimated utility may limit the flexibility to diversify recommendations. There is thus a tradeoff to be made between two competing goals: optimization and diversity. To find the right balance, I follow two steps. First, for optimization, I find the optimal choice A_1 by maximizing the estimated group utility, $A_1 \in \operatorname{argmax}_A U(\vec{u}(A))$, where $A \in R$, $\vec{u}(A)$ is the utility vector, and $U(\vec{u}(A))$ is the estimated group utility corresponding to vector $\vec{u}(A)$, which is computed by using any of the six group decision-making methods explained above.

Second, for diversification, I adapt the diversity layering method from CARD [34]. However, the dimensions of the utility space in [34] are the original criteria, whereas, I am advocating using the space of the extracted utilities of the individual decision makers instead.

The motivation for this choice is that individuals may not be satisfied if none of the options presented to them for voting are closely related to their own preferences. We would like to mimic as closely as possible the popular group decision-making mechanisms in which the alternatives are proposed by individual members and thus reflect their preferences. However, the diversity layering method, described below, will still provide options that are either optimal or fall within a bounded distance of the optimal group utility.

The key idea is to create a diverse subset of recommendations that correspond to

different individual's utility functions while remaining within a bounded distance of the optimal group utility score to provide a balance between optimality and diversity. I partition the recommendation space into q layers, starting with the layer that includes the optimal recommendation and maximizes the group utility U . The second layer includes the recommendations close to the optimal one and with total utility of at least the maximum group utility minus ε (a percentage of the maximum group utility score). The third layer includes recommendations with a total utility values of not less than the maximum minus 2ε . In general, a recommendation in the i -th layer will have a utility value of at least the maximum group utility minus $(i-1)\varepsilon$. Within each layer, I select n recommendations to maximize each dimension of the recommendation space in turn.

The example depicted in Figure 4.1 illustrates the diversity layering method. Here, RU_1 and RU_2 are two individual decision makers' utilities, and U is the group utility, which is defined as a linear combination of RU_1 and RU_2 . The polygon set in the figure depicts all the possible utility vectors of the recommendations. Among these, A_1 is the optimal recommendation for maximizing U .

The second layer includes recommendations for which $U \geq \max\{U\} - \varepsilon$, where ε is a percentage of $\max\{U\}$, say 2%. The selected recommendations in this layer are A_2 and A_3 because they maximize RU_1 and RU_2 , which provides diversity while restricting the group utility within its layer preserves the distance from the optimal recommendation. The third layer includes recommendations for which $U \geq \max\{U\} - 2\varepsilon$, and the selected recommendations are A_4 and A_5 , which maximize RU_1 and RU_2 respectively.

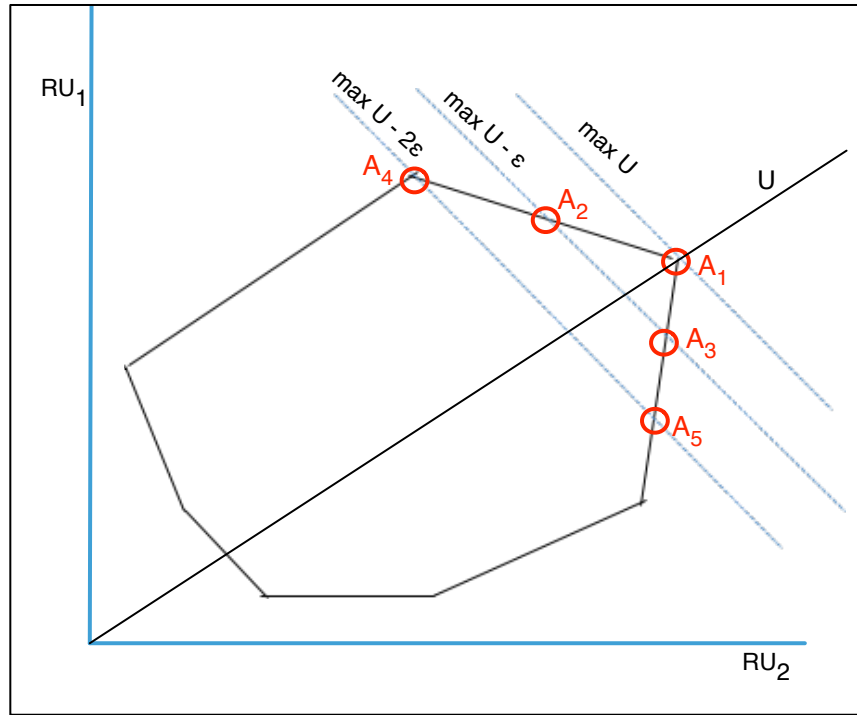


Figure 4.1: Diversity Layering

The diversity layering method generates a set of alternatives by optimizing each user's utility function in each layer. It may not scale well for large groups—for example, where the number of individuals is greater than the number of diversity recommendations needed. In such cases, the system may complete the set of recommendations early, before completing optimization over all member's utility functions, even while still in the first layer. We can solve this problem by clustering large groups into homogenous subgroups and diversifying recommendations across subgroups rather than individuals. This is considered in Chapter 5.

After the diversity set of recommendations has been generated, they are presented to each individual decision maker in descending order of group utility, and each decision

maker ranks (or rates, depending on the decision-making method being used) the recommendations in a way that reflects her own preferences. Allowing each member to rank the pre-final results individually avoids the effects of an incorrect estimation of individual decision makers' utility functions in the first step.

These individual rankings of the optimal and diverse recommendations are the input for the group decision-making method that is applied in the last step to determine the final top k recommendations.

4.5 Initial Experimental Evaluation

Experimental Setting

I conducted a preliminary experimental study to evaluate the GCAR framework's precision and recall under each group decision-making method. Precision and recall metrics are widely used in information-retrieval scenarios. Recall is the proportion of truly good recommendations that appear in the top recommendations, and precision is the proportion of the recommendations that are truly good ones [1].

The study involved 67 human participants, all of them graduate students, in 13 groups of different sizes, as follows: 1 group of 2 participants; 2 groups of 3 participants; 2 groups of 4 participants; 4 groups of 5 participants; 2 groups of 6 participants; 1 group of 9 participants; and 1 group of 10 participants. The hypothesis of this study was that the GCAR framework can produce a small set of recommendations that remain nearly optimal in precision and recall.

The data for the study were actual data on vacation packages, which I extracted

from a popular commercial travel website by submitting a request for a two-week vacation in Los Angeles, California, including non-stop round-trip airfare, from Washington Dulles Airport. All the packages returned by this website were extracted keeping only the cost and number of stars (enjoyment) of each package.

Experimental Methodology

I undertook to study the framework’s performance with different decision-making methods, not to determine which method is the best. Extensive work has been done on GRSs, showing that some methods are better than others in different situations (e.g., [12, 44]). For this research, in which the method that best fits the situation is chosen externally, I am studying the accuracy of my own system, in the face of the approximations necessary in the large selection space, and in comparison to ideal accuracy, which is obtained under the assumption of complete knowledge (without approximations).

For evaluation purposes, I consider for each group only the packages that have been evaluated by all the group’s members. These evaluations are based on ratings on a scale of 1 (strongly disagree) to 5 (strongly agree). For each group, I then apply the GCAR framework using each group decision-making method on the same dataset to repeatedly generate the top 4 recommendations. Finally, to generate the ground truth for each group, I aggregate the actual individual preferences into the group’s actual overall preferences using the alternative group decision-making methods so as to fit the choice of the method used in the framework.

To estimate the recall of the GCAR framework at a given rank k , I gather all the

packages rated 4 or above from the group ground truth into a set called “Good.” Then, for each group, I calculate the estimated recall as:

$$Recall(k) = \frac{|\{r \in Good | rank(r) \leq k\}|}{|\{Good\}|} \quad (4.11)$$

Next, I compute the average recall at each rank k , for each group decision-making method, by taking the average of recall k among all the 13 groups. The results are shown in Figures 4.2 to 4.7.

Similarly, I estimate the precision of the GCAR framework at a given rank k for each group as

$$Precision(k) = \frac{|\{r \in Good | rank(r) \leq k\}|}{k} \quad (4.12)$$

Finally, I compute the average precision at each rank k , for each group decision-making method, by taking the average of precision k among all the 13 groups. The results are also shown in Figures 4.2 to 4.7.

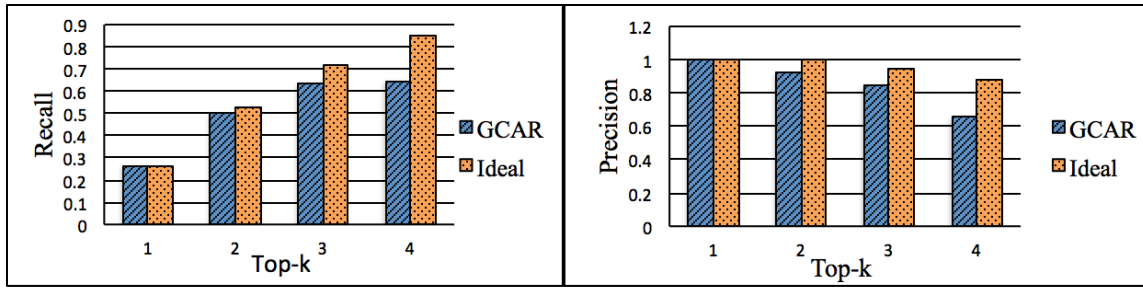


Figure 4.2: Average Recall and Precision for GCAR vs. Ideal (Average Strategy)

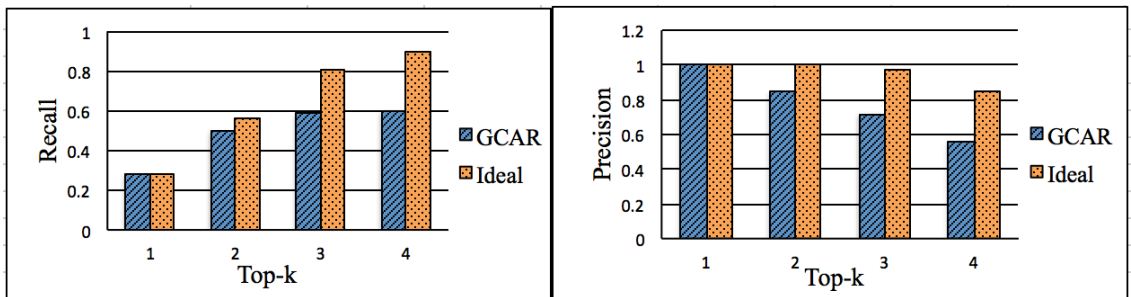


Figure 4.3: Average Recall and Precision for GCAR vs. Ideal (Least Misery Strategy)

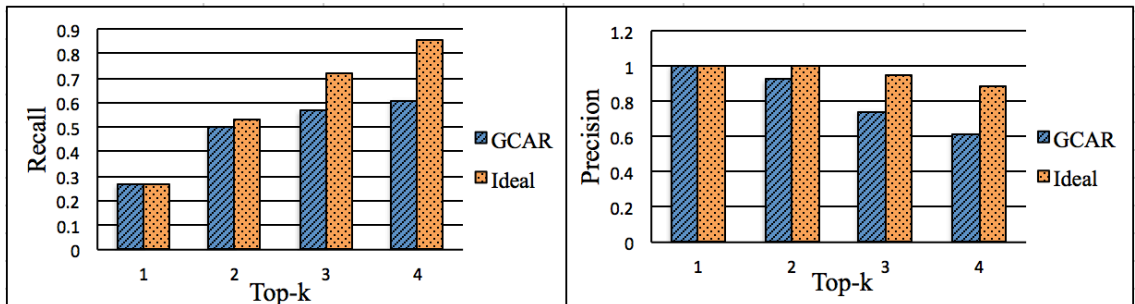


Figure 4.4: Average Recall and Precision for GCAR vs. Ideal (Average without Misery)

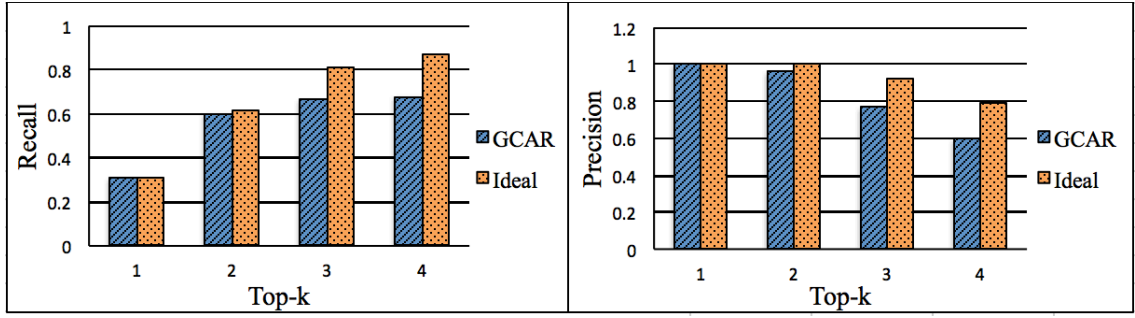


Figure 4.5: Average Recall and Precision for GCAR vs. Ideal (Structurally Adjusted Average)

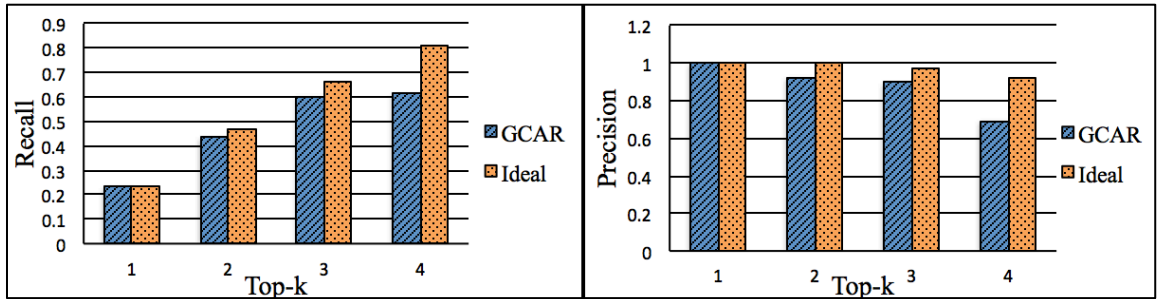


Figure 4.6: Average Recall and Precision for GCAR vs. Ideal (IRV Method)

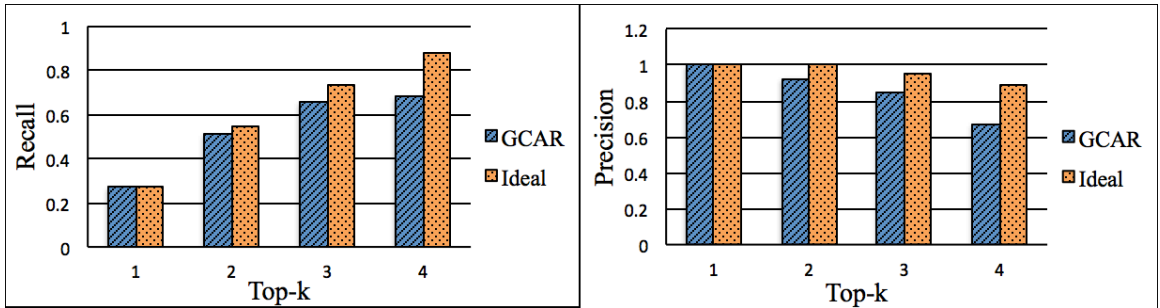


Figure 4.7: Average Recall and Precision for GCAR vs. Ideal (Condorcet-IRV Method)

Experimental Results

The study shows that for top-1 recommendations, the average recall and precision achieved by the proposed GCAR framework under each decision-making method are exactly the same as the ideal.

For top-2 recommendations, the GCAR framework obtained recall and precision within 90% of the ideal under all the methods except least misery method, for which the recommendations were within 85% of the ideal.

For top-3 recommendations, GCAR framework's recall and precision were within 80% of the ideal for all methods except least misery and average without misery, for which they were 20 to 27% off from the ideal. Recall and precision were 23 to 34% off from the ideal for top-4 recommendations under all methods.

Statistical Analysis

For the statistical analysis, I calculate the confidence interval (CI) for the estimated mean of the percentage differences between the GCAR framework's accuracy and the ideal, in terms of recall and precision, using alternative group decision methods. For this calculation, I apply the following formula:

$$\text{Estimated mean} = \text{sample mean} \pm (se . t_{crit}) \quad (4.13)$$

where se is the standard error of the mean and t_{crit} is the two-tailed critical value of t for the 0.05 level of significance. The results are illustrated in Figure 4.8.

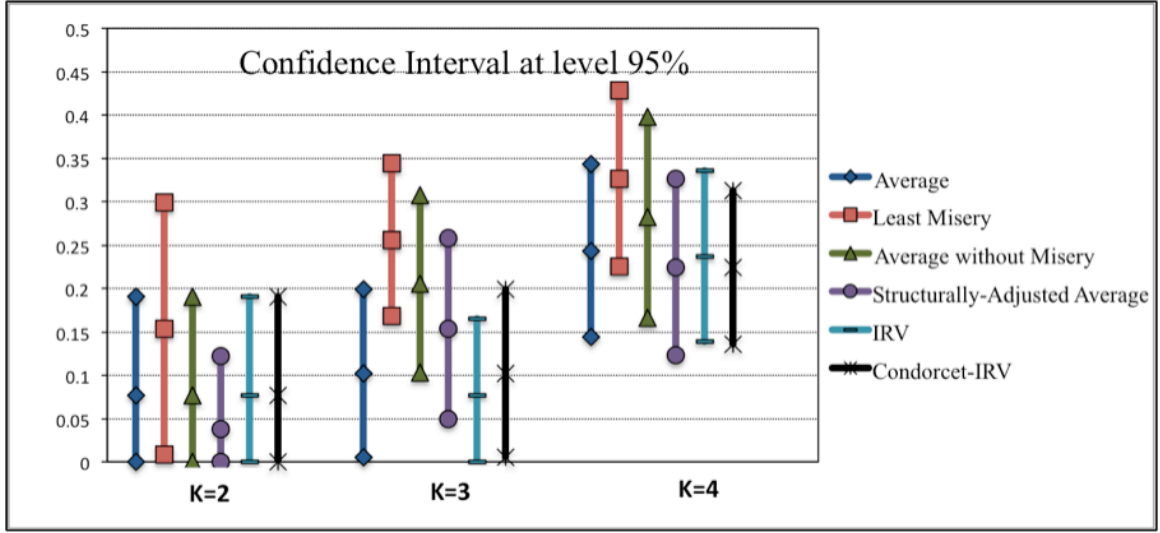


Figure 4.8: Confidence Interval at Level 95% for the Estimated Mean of the Percentage Differences Between GCAR Accuracy and Ideal, in Terms of Recall and Precision

Note that the CI for the top-1 recommendation (i.e., when $k = 1$) is not shown in Figure 4.8. This is because it is equal to 0; that is, we are 95% confident that the framework's accuracy under all six methods, will match the ideal accuracy in recall and precision for top-1 recommendations.

In Figure 4.8, the markers in the middle of the vertical lines represent the mean of the percentage differences between the GCAR framework's accuracy and the ideal. The vertical lines indicate the confidence interval. For instance, under the average strategy, for top-2 recommendations, the mean percentage difference between the proposed framework's accuracy and the ideal will be between 0 and 20%. It will also not be off by more than 30% from the ideal for top-3 recommendations (40% for top-4) under any method but least misery, for which the accuracy may decrease.

As I mentioned, extensive work on GRSs has explained that some methods are

better than others in different situations. I think that the accuracy of the proposed framework decreases under the least misery strategy because that method works well only with small groups [6], whereas the groups in this study were relatively large.

4.6 Summary

In this chapter, I explained the steps of the proposed GCAR framework. I also described a preliminary experimental study I conducted to evaluate the framework's performance under each of the six group decision-making methods. I used alternative methods to model the “actual” group preferences so as to fit the decision-making method used in the framework. The study showed that GCAR framework can produce a small set of recommendations that remain nearly optimal under any of the methods: the average precision and recall it achieved ranged from ideal, for top-1 recommendations, to between 23 and 34% off the ideal for top-4 recommendations.

Although the framework is designed to be high scalable in terms of alternatives through utility optimization, it may not scale well for large numbers of decision makers. One reason is the fact that eliciting the utility function for each user may not be practical for large groups. In addition, aggregation methods for large groups can lose accuracy, as indicated in many works (e.g., [8, 47]).

Therefore, in the following chapter I propose a technique for improving the quality of the GCAR framework with large, heterogeneous group of users who may have strongly conflicting views on weights for different criteria.

CHAPTER 5: TAILORING GROUP PACKAGE RECOMMENDATIONS TO LARGE HETEROGENEOUS GROUPS

5.1 Introduction

In the previous chapter, I described the proposed GCAR framework, which provides a diverse set of group package recommendations based on multi-criteria decision optimization. However, this framework is designed for small groups of users and is not flexible enough to support very large, heterogeneous groups with multiple conflicting views on the weights of different criteria. In many areas, recommendations affect large numbers of users. For example, many employees of a company might like to go to a conference, or a large number of people in a county or a city might want to influence its infrastructure investments. A few group recommenders have applied algorithms to improve large-group recommendations (e.g., [24, 111-113]), but none of them was designed to support package recommendations that are implicitly defined.

The focus of this chapter is extending the GCAR framework to support large heterogeneous groups. The contributions of the chapter are twofold. First, I propose a technique for scaling up GCAR for such groups. The idea is to randomly select a representative sample of the entire group and elicit the utility function of each member in the sample. Then, these utility functions are clustered into a number of relatively small homogeneous subgroups of users with similar utilities, and the representative utility function for each cluster is used to find the optimal recommendation for the subgroup.

These subgroup utility functions are then combined to estimate the utility function of the entire group, U . However, using U directly may limit the flexibility of users to refine their choices. Therefore I use the estimated U to derive a small set of near-optimal recommendations that are also diverse in terms of the subgroups' utility functions. Finally, one of the voting methods is used to determine the top k recommendations.

Second, I conduct an experimental study to demonstrate the framework's scalability to very large heterogeneous groups. In this study, the utility functions of members were synthetically generated, and the precision and recall of the proposed framework were compared to the recommendations that would be generated through manual voting methods by the entire large groups. The study demonstrated an average precision ranging from 0.95 for top-1 recommendations to 0.80 for top-5 recommendations. For recall, it demonstrated an average ranging from 0.19 for top-1 recommendations to 0.80 for top-5 recommendations.

The remainder of this chapter is organized as follows: Section 5.2 gives a high-level overview of the proposed technique. Section 5.3 explains the entire group sampling and the eliciting of user utilities. Section 5.4 explains the clustering process. Section 5.5 explains the estimation of the group utility from the subgroups' utilities. Section 5.6 discusses the experimental evaluation of the GCAR framework's scalability to large groups. Finally, Section 5.7 summarizes the chapter. Some of the work reported in this chapter was published in [114].

5.2 GCAR with a Technique to Support Very Large Groups

For a group of m users, the utility of each user j , denoted by $\forall_j, 1 \leq j \leq m$,

$U_j: [0,1]^n \rightarrow [0,1]$, maps a vector of criteria $u_1 \dots, u_n \in [0,1]$ into a user utility $U_j(u_1 \dots, u_n) \in [0,1]$. Similarly, the utility of each subgroup z is denoted by $\forall_z, 1 \leq z \leq k$, $U_z: [0,1]^n \rightarrow [0,1]$, where k is the total number of subgroups. In addition, the entire group sample utility is denoted by $U: [0,1]^n \rightarrow [0,1]$.

U_z and U define a utility associated with each alternative $a \in R$. Therefore, the subgroup recommendation alternative utility for recommendation a is defined by $RU_z: R \rightarrow [0,1]$, where $RU_z(a) = U_z(u_1(a), \dots, u_n(a))$, and the recommendation alternative utility for the entire group sample is defined by $RU: R \rightarrow [0,1]$, where $RU(a) = U(u_1(a), \dots, u_n(a))$.

The recommendation process implemented by the proposed technique is depicted in Figure 5.1.

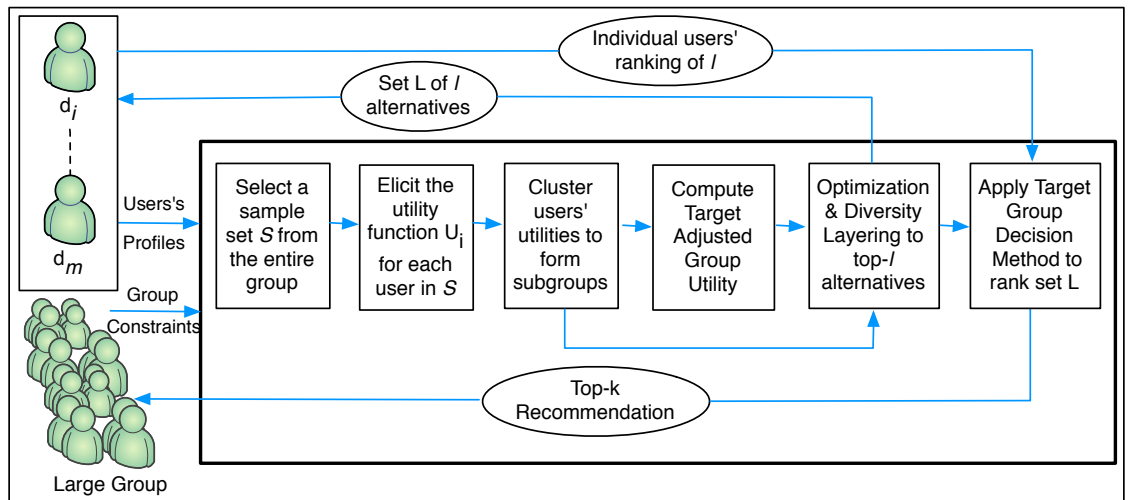


Figure 5.1: GCAR Framework with the Proposed Technique to Support Very Large Groups

As shown in the diagram, to generate the top k recommendations for a large heterogeneous group, the recommender follows seven steps: (1) sampling the entire large group; (2) eliciting the utility function for each member in the sample; (3) clustering these utility functions into small homogeneous subgroups of similar utilities; (4) extracting the representative utility function for each subgroup; (5) estimating the utility function of the entire group and using it to find an optimal recommendation; (6) diversifying the recommendations across the subgroups; and (7) applying a group decision-making method to refine the top k recommendations.

Some of these steps are similar to steps in the basic GCAR framework. The sampling and clustering steps are added for very large groups, and the diversity-layering step in Figure 5.1 is applied to the extracted utilities of the subgroups rather than those of the individual decision makers. In this chapter, I explain in detail only the new steps.

5.3 Sampling the Entire Group and Eliciting User Utilities

In the real world, resources are often limited, which makes it impractical to elicit the utility function of every user in a large group. This is why the proposed framework begins by randomly selecting a representative sample of users. This sampling phase can be skipped in situations with sufficient resources. For details on eliciting the utility function of each user, refer to Section 4.2.

5.4 Clustering Users' Utilities

The goal is to cluster the numerous individual utility functions of the sample into a number of small subgroups in such a way that the utility functions in each subgroup are

more similar to each other than to those in other subgroups. Formally, I aim to partition the m utility functions into a set of k clusters $C = \{c_1, c_2, \dots, c_k\}$ so as to minimize the within-cluster sum of squares, which is defined as

$$\operatorname{argmin}_C \sum_{Z=1}^k \sum_{U_j \in C_Z} \|U_j - \mu_Z\|^2 \quad (5.1)$$

where U_j is the utility of decision maker j , $1 \leq j \leq m$, for an alternative a , as defined in Equation 4.1, and μ_Z is the mean of cluster C_Z . For clustering, I choose to use the k -means algorithm for its simplicity and popularity [115]. By definition, $k = l - 1$, where l is the number of alternatives needed from the optimization and diversity layering step. To compute the distance between the input vector and the clustering center, the algorithm uses the Euclidean distance [116].

With very large groups, it is normal for individual members to represent either just themselves or a number of users. For example, in the case of public infrastructure investments, we might have individual members representing the government sector, the private sector, expert decision makers, and environmental protection organizations. Therefore, I introduce a representation factor r_j , $\forall j, 1 \leq j \leq m$, which is the number of users member j represents. This factor figures in the estimation of the utility functions for both the subgroups and the entire group. Consequently, the utility function of a given subgroup C_Z is defined as

$$U_Z(\vec{u}) = \frac{1}{|C_Z|} \sum_{j=1}^p (U_j(\vec{u}) \cdot r_j) \quad (5.2)$$

where p is the number of users in subgroup C_Z and $|C_Z|$ is the normalized size of subgroup C_Z (the total number of users represented by this subgroup), which is defined as $|C_Z| = \sum_{j=1}^p r_j$.

5.5 Estimating Group Utility on the Basis of Subgroup Utilities

This group utility estimation is parameterized on the basis of the group decision-making method to be applied in the last step of Figure 5.1. Six such methods were explained in Chapters 3 and 4. I now discuss how I estimate the group utility from the subgroups' utilities with some of these methods. For the IRV and the Condorcet-IRV methods, see Chapter 4.

5.5.1 Average Strategy

The influence Inf_Z of subgroup C_Z on the group's preferences is computed as

$$Inf_Z = \frac{|C_Z|}{|G|} \quad (5.3)$$

where $|C_Z|$ is the normalized size of subgroup C_Z and $|G|$ is the normalized size of the

entire group sample, which is defined as $|G| = \sum_{j=1}^m r_j$, and $(\sum_{z=1}^k Inf_z) = 1$.

Taking Inf_z into account for each subgroup, the group utility is then defined as

$$U(\vec{u}) = \sum_{z=1}^k (U_z(\vec{u}) \cdot Inf_z) \quad (5.4)$$

5.5.2 Least Misery Strategy

In this strategy, the group utility is computed as the lowest utility given to any alternative by the subgroups, as follows:

$$U(\vec{u}) = \min_z (U_z(\vec{u})) \quad (5.5)$$

where U_z is defined as in Equation 5.2.

5.5.3 Average without Misery Strategy

In this strategy, the group utility is computed as in the average strategy, but those alternatives with any subgroup utility below a certain threshold t are excluded from the group recommendations. More formally,

$$U(\vec{u}) = \sum_{z=1}^k (U_z(\vec{u}) \cdot Inf_z), \text{ such that } \forall_z, 1 \leq Z \leq k, \min_z(U_z(\vec{u})) \geq t \quad (5.6)$$

5.5.4 Structurally Adjusted Average Strategy

This strategy was developed in this research and was explained in Chapter 3. It computes the group utility by taking into account two main GRS factors: (a) the influence of individuals within the group; and (b) the dissimilarity of opinion among group members. To describe dissimilarity of opinion among subgroups over an alternative, I use the standard deviation:

$$\sigma(U_1, \dots, U_k) = \sqrt{\frac{1}{Z-1} \sum_{z=1}^k (U_z - MU)^2} \quad (5.7)$$

where MU is the mean of subgroup utilities for alternative a . Finally, to reflect both the influence of subgroups in the entire group and the dissimilarity of opinion among them, I compute the adjusted group utility as

$$U = WU \cdot (1 - \delta) \quad (5.8)$$

where WU is the weighted average group utility, as defined in Equation 5.4, and δ

represents the dissimilarity penalty, as defined in Equation 3.7.

5.6 Experimental Evaluation of GCAR Scalability

5.6.1 Experimental Setting

The first three steps of the technique involve approximating group utility, because working with all the members individually is impractical with very large groups. But this raises the important question of how much accuracy (in terms of precision and recall) we lose in these approximations. I conducted an experiment to answer this question. I compared the precision and recall of the proposed framework, as explained in Section 5.2, to the results that would be generated by manual voting by the entire large groups (the baseline).

In this experiment, I set the group size m to three different values, 1000, 10,000, and 100,000, with 20 groups of each, and I assume that each alternative i , $1 \leq i \leq N$, is associated with only two utilities, u_1 and u_2 . For the recommendation space, the number of alternatives N is set to 1000, and I generate these alternatives by assuming that the recommendations, in terms of u_1 and u_2 , are uniformly distributed on a quarter circle, as shown in Figure 5.2, where $\forall i$, $u_1 = \cos \frac{\pi \cdot i}{2 \cdot N}$, and $u_2 = \sin \frac{\pi \cdot i}{2 \cdot N}$. Note that these correspond to the Pareto-optimal selection, and that if there are additional alternatives located inside the circle, they will not be considered because they will be dominated.

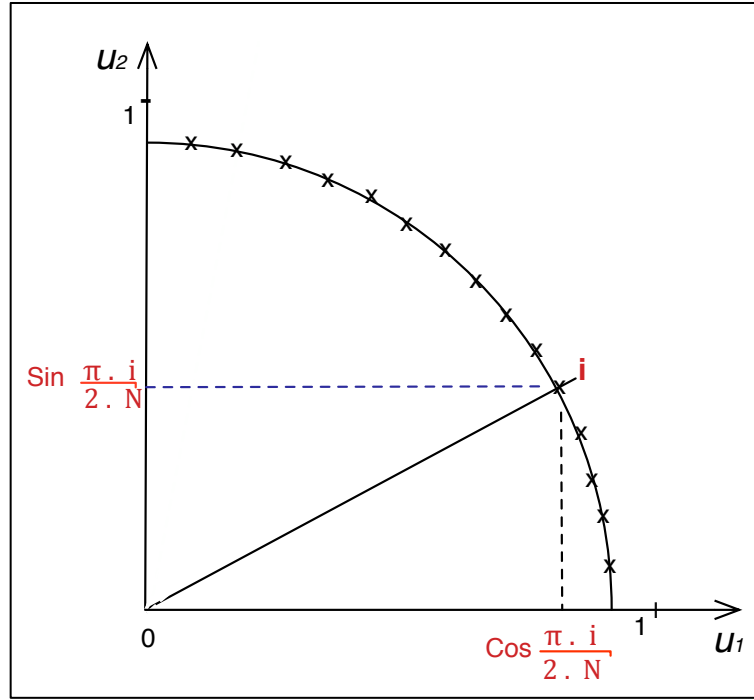


Figure 5.2: Alternatives Generation

In this study, large enough groups of people are considered that it is impractical to consider real user utility functions. Therefore, I assume that for each user j , $1 \leq j \leq m$, the proposed framework has correctly extracted her utility function, U_j , and that each user j represents only herself, for simplicity. These user utilities are simulated as shown in Figure 5.3, for which I formulated the utility functions so that they are geometrically arranged on a quarter circle.

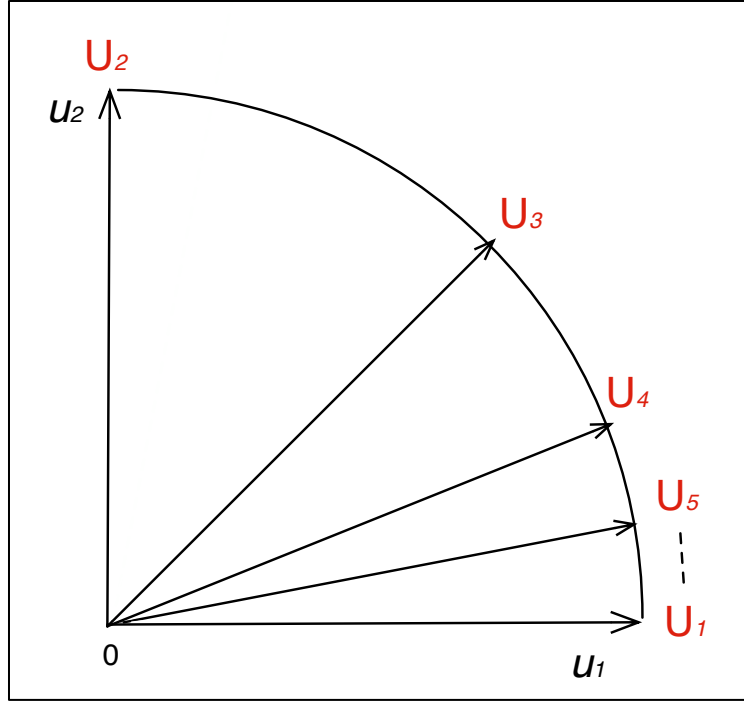


Figure 5.3: User Utilities Generation

5.6.2 Experimental Methodology

For both the baseline and the proposed framework, I used the weighted average method, as explained in Section 5.5, to generate the top 5 recommendations for each group. To answer the question above, I calculated the recall and precision metrics of the top 5 recommendations returned by the proposed framework and by the baseline. Recall is the proportion of relevant recommendations that appear among the top recommendations, and precision is the proportion of the recommendations that are relevant [1]. In this study, any of the framework's top recommendations (F) is considered relevant if it is one of the baseline system's top recommendations (B).

For each group, I estimated the recall of the framework at a given rank k as

$$Recall(k) = \frac{|F_k \cap B_k|}{|B|} \quad (5.9)$$

Then I computed the average recall at each rank k for the proposed framework by taking the average of recall (k) among all groups. The result is shown in Figure 5.4.

Similarly, for each group I estimated the precision of the proposed framework at rank k as

$$Precision(k) = \frac{|F_k \cap B_k|}{k} \quad (5.10)$$

and then computed the average precision at each rank k by taking the average of precision(k) among all groups. The result is shown in Figure 5.5.

5.6.3 Experimental Results

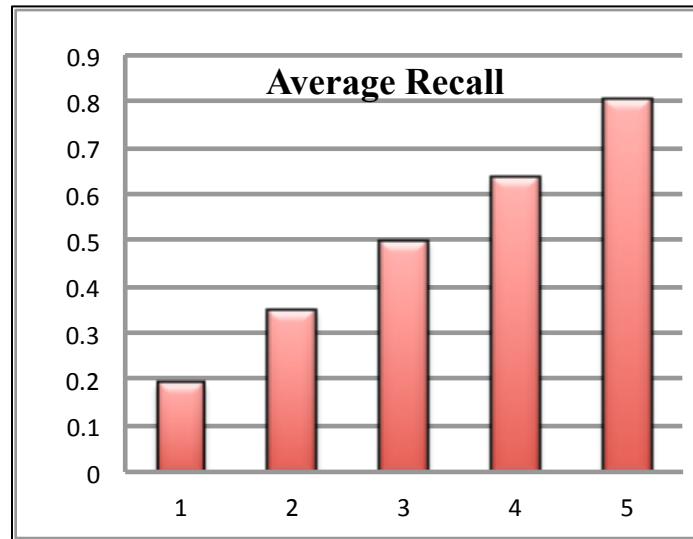


Figure 5.4: Average Recall vs. Rank (k) for the Proposed Framework

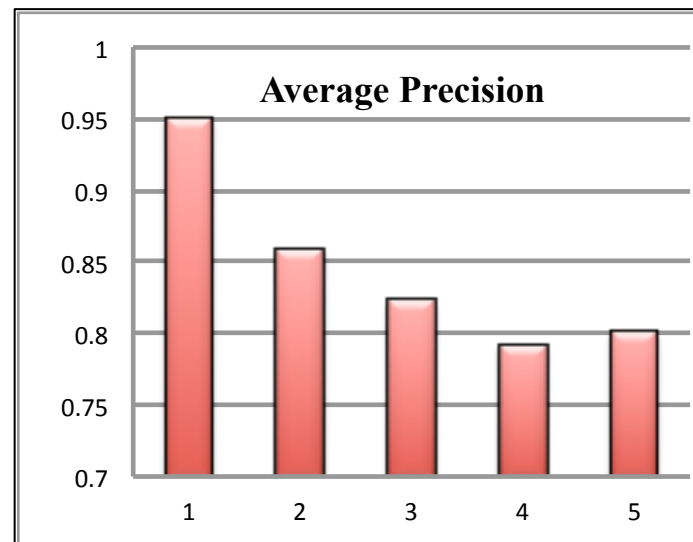


Figure 5.5: Average Precision vs. Rank (k) for the Proposed Framework

As shown in Figures 5.4 and 5.5, at rank 1 almost all the top-1 recommendations returned by the proposed framework were relevant. This indicates that we lost almost no accuracy among top-1 recommendations by applying the technique, even with the approximations. For top-2 recommendations, 86% of the returned recommendations are relevant. However, for top-3, -4, and -5 recommendations, only about 20% of the returned recommendations were irrelevant.

For the statistical analysis, I calculated the CI at level 95% for the estimated mean of the proposed framework's recall and precision. The results are given in Table 5.1. We are 95% confident that we will not lose more than 25% of our outcomes' recall and precision for top-5 recommendations by applying the proposed technique, in which the original large groups are approximated.

Table 5.1: Confidence Interval at Level 95% for the Accuracy Estimated Mean

	k = 1	k = 2	k = 3	k = 4	k = 5
Recall	0.19 ± 0.01	0.35 ± 0.02	0.50 ± 0.03	0.64 ± 0.04	0.80 ± 0.05
Precision	0.95 ± 0.06	0.86 ± 0.06	0.82 ± 0.05	0.79 ± 0.05	0.80 ± 0.05

5.7 Summary

In this chapter, I proposed a technique for scaling up GCAR for very large heterogeneous groups. The idea is to select a representative sample of the large group, elicit utility functions from its members, and cluster these functions into small

homogeneous subgroups of users with similar utilities. The representative utility function for each cluster is then used to find the optimal recommendation for the subgroup and to diversify the final recommendations.

I also described a study I conducted to demonstrate this scalability. In this study, the utility functions of members of very large groups were synthetically generated, and the precision and recall of the proposed framework were measured against the recommendations that would be generated by manually voting by the entire groups. The study showed that the proposed framework, which approximated the original large groups, can produce a small set of near-optimal recommendations. The experiment also shows that in applying the framework we did not lose more than 21% of the accuracy for top-5 recommendations.

In the next chapter, I demonstrate how the GCAR framework applies to a real problem by considering a realistic case study.

CHAPTER 6: CASE STUDY: POWER MICROGRID OPERATION AND INVESTMENT RECOMMENDER (PMOIR)

6.1 Introduction

In the previous chapters, I described the proposed GCAR framework, which provides a diverse set of package recommendations to a group of users. I also described a proposed technique for scaling the GCAR up to handle very large, heterogeneous groups, and I described the experiments I conducted to evaluate the precision and recall of the framework's outcomes.

In this chapter, I demonstrate how the proposed framework applies to a real problem by considering a realistic case study of a power microgrid operation and investment recommender (PMOIR) used to recommend to a group of decision makers a set of optimal operation and investment decisions involving interrelated power components in a power microgrid of a university campus. A microgrid is an integrated system of energy resources together with a sophisticated decision system for controlling them, such as the power microgrid of a university campus, an industrial facility, or a building complex (see Figure 6.1).

The energy operation and investment recommendations include optimal settings and values for decision control variables, such as the amount of power generated from the resource components (e.g., batteries, backup generators, utility contracts, and renewable resources) and the amount consumed by service components (e.g., heating, ventilation,

air conditioning, and lighting) in each time interval. The goal is to maximize the net present value (NPV) within the required demand satisfaction ratio and within the bound for greenhouse gas (GHG) emissions. This is to be done while taking into account all components' interactions and satisfying a diverse group of decision makers (e.g., the general staff, executives, energy managers, and environment protection officers) who may have conflicting views on weights for the relevant criteria. As indicated in works [117, 118], making optimal planning and investment decisions for interrelated energy components is a complex problem because of the variety of energy resources, the interdependencies of the energy components, the complex interactions between old and new equipment in every time interval over an investment time horizon, the huge diurnal and annual variations in energy consumption, and environmental problems.

In order to implement PMOIR, I mathematically modeled specific power components based on the models described in work [119]. These components include renewable resources, battery units, backup generators, utility contracts, and power-consuming services. I also formalized the optimization problem, which involves all these energy components and their complex operational interdependencies. This is the focus of this chapter; the implementation of the PMOIR case study is described in Chapter 7.

The rest of this chapter is organized as follows: Section 6.2 gives a high-level description of a group package recommender for the renewable energy operation and investment problem. Section 6.3 describes the optimization formalization. Section 6.4 describes the power component modeling. Section 6.5 summarizes the chapter.

6.2 Renewable Energy Operation and Investment Group Recommender

Consider a realistic case study of a power microgrid of a university campus like the one in Figure 6.1, which involves a number of components, such as services like: lighting, cooling, water heating, etc., utility contracts, and backup generators. Now suppose that a group of decision makers are planning to invest in the renewable energy components of this microgrid, namely the solar photovoltaic cells and wind turbines. Investing in these components is not trivial because their power supply is unpredictable and may drop suddenly, which will require either immediately supplementing it from other power components like batteries or reducing the demand. Thus the components interact with each other as a package: whenever the supply from the renewable components drops, other components must compensate for it. In addition, the microgrid system needs to decide on an hourly basis how much power is to be supplied or used by each component.

This microgrid consists of both the installed energy components and others that may have to be purchased later in the time horizon to meet the entire campus's future power demands. The energy managers have recently seen a significant growth in power demand, and because the campus is continuing its expansion rapidly, they realize that the existing energy components will not be able to satisfy future power demand.

For these reasons, a group of decision makers from different departments must determine the best investment and operation options to satisfy current and future power demand while also addressing the optimal operation of the new components with the

installed ones. The goal is to maximize their NPV within the bounds for GHG emissions. This should be done while satisfying the group of decision makers, who have different views on appropriate weights for the criteria. For example, while the general staff and the executives give the highest weight to the NPV criterion, the energy managers consider the demand–satisfaction ratio the most important, and the environmental protection people focus on reducing the GHG emissions.

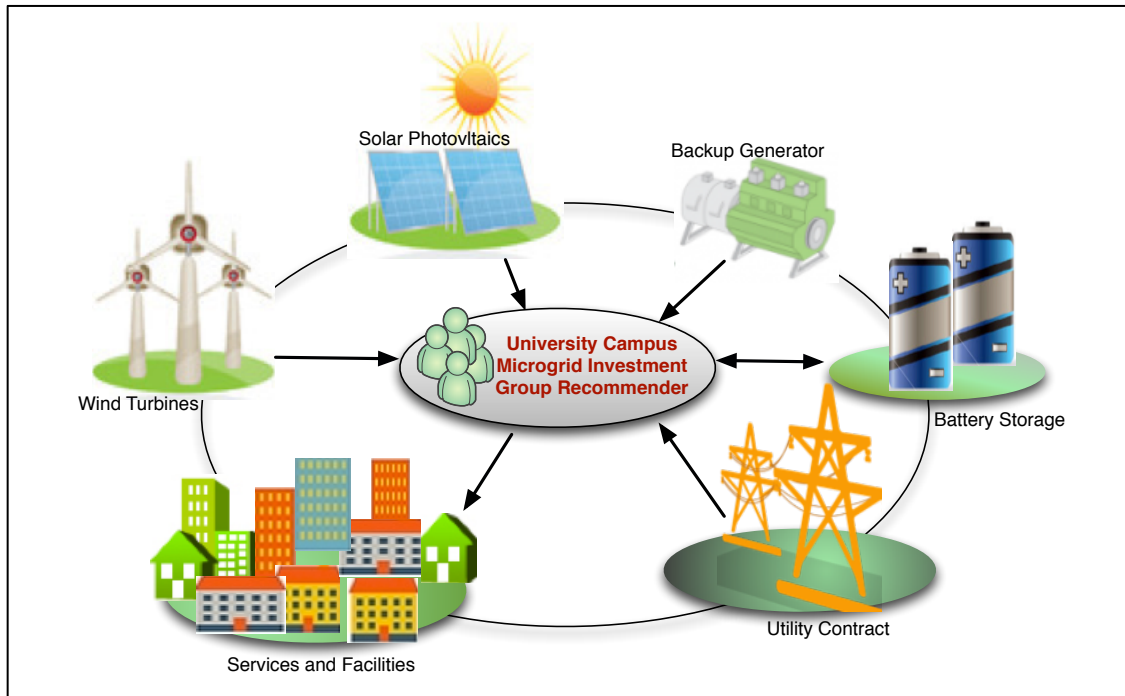


Figure 6.1: Example of a University Campus Microgrid

To solve this problem with the proposed GCAR framework, I mathematically formalized the overall optimization problem, as described in the next section.

6.3 Optimization Formalization

Assume that a microgrid consists of a set of power components $C = \{c_1, \dots, c_k\}$ that includes a subset of components available at the initial time horizon, denoted by $(initAvaC)$, and a subset of components that are not yet available but might be purchased later in the time horizon, denoted by $(notInitAvaC)$, where $C = initAvaC \cup notInitAvaC$.

These components can be considered power-producing resources, such as backup generators and solar panels, or power-consuming services, such as lighting and heating.

The time horizon T is a set of discrete hourly time intervals, $T = \{1, \dots, N\}$ where $N = 24$ if the time horizon is one operational day with interval length, and $N = 8760$ if the time horizon is one operational year with interval length, and so on. An interval length of 1 means that each time interval is an hour long.

Generally, every component $i \in C$ is associated with the following:

1. A vector of controls $\vec{a}_i = (a_{i1} \dots, a_{iN})$, which represents the control actions that component i takes over time horizon T , and where each a_{it} , $1 \leq i \leq k$, $1 \leq t \leq N$, is the control action that component i takes at time t . Therefore, the control actions for all components over the time horizon T is represented as matrix A :

$$A = \begin{pmatrix} \vec{a}_1 \\ \vdots \\ \vec{a}_k \end{pmatrix} = \begin{pmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{k1} & \cdots & a_{kN} \end{pmatrix}$$

These control actions are defined as follows:

- $avail[i]$ indicates whether component i is available at the initial time interval; $\forall i \in C$, $avail[i] = 1$ if component i is available and 0 otherwise.
- $buyFlag[i]$ indicates whether new component i should be purchased, where $\forall i \in notInitAvaC$, $buyFlag[i] = 1$ if component i is to be purchased and 0 otherwise. In addition, $\forall i \in initAvaC$, $buyFlag[i] = 0$.
- $on[i,t]$ indicates whether component i supplies or uses an amount of power at time interval t , such that \forall component $i \in C$, $on[i,t] = 1$ if i supplied or used power at t and 0 otherwise.
- $kw[i,t] \in \mathbb{R}$ indicates how much power should be supplied by each component $i \in C$ in each time interval $t \in T$.

2. A number of metrics, such as cost, GHG emissions, and number of payments during the time horizon, where each payment consists of a specific amount paid at a specific time interval. These metrics are discussed for each component type in the following subsections.
3. A number of operational and investment constraints, in terms of control actions \vec{a}_i . For example, the capacity in kw of a power generator is a constraint on the amount of power that this component can generate.

Every operation and investment decision option for this problem will involves the values of all the component decision control vectors over the time horizon. In addition, each option is associated with a utility vector $\vec{u} = (u_1 \dots, u_n)$ from an n -dimensional utility space, such that $\forall_i, 1 \leq i \leq n$, $u_i: R \rightarrow [0,1]$, where each utility u_i has a specific

domain D_i and represents a specific criterion. For example, u_1 represents the NPV, u_2 represents the GHG emissions, and u_3 represents the demand-satisfaction ratio, where $\sum_{i=1}^n u_i = 1$. Furthermore, each utility has a global weight, $utilityWeight_i$, such that $\sum_{i=1}^n utilityWeight_i = 1$. Therefore, the total utility of all components, given their control action matrix A , denoted by $TotalUtility : R \rightarrow [0,1]$, is defined as

$$TotalUtility(A) = \sum_{i=1}^n utilityWeight_i \times utility_i$$

The overall optimization problem is to maximize the total utility over the action matrix A , subject to three kinds of constraints: the generic balance constraints for the entire problem model, the utility-definition constraints for the entire model, in terms of the component metrics, and the specific constraints for each power component. Formally, the optimal control action matrix A^o is the one that maximizes the utility the microgrid system can achieve:

$$A^o \in \arg \max_A TotalUtility(A)$$

subject to

$$Generic\ Constraints(A) \wedge$$

$$Utility\ Definition\ Constraints(A) \wedge$$

$$Component\ Constraints_i(\vec{a_i}) \forall i \in C$$

where:

- Matrix A^o represents the control actions that all components must take to achieve optimal utility over the entire time horizon, and
- Vector (\vec{a}_i) represents the control actions that component i takes over the time horizon T .

In the following sections, I describe in details each of these constraints, starting with the balance and utility definition constraints in this section, and leaving the power component constraints for the next.

6.3.1 Generic Constraints

In addition to the specific constraints for each component i given its control actions vector, $(\vec{a}_i) \forall i \in C$, there are generic constraints for the entire model, given the control action matrix A , that need to be satisfied. These are as follows:

- For a stable power supply to be maintained, the sum of power supply and power demand for any time interval must equal 0; that is,

$$\forall t \in T \sum_{i=1}^k power_i(a_{it}) = 0$$

where $power_i$ is the power in kW that component i produces or consumes, given the control actions vector \vec{a}_i , in any time interval t . This value is positive if the component supplies power, negative if it receives power, and zero if it is unavailable at t or is turned off.

- In any time interval, if a component i is *OFF*, the output or input power from or to i for this time interval must equal 0; that is,

$$\forall (i \in \text{Components}, t \in T)$$

$$on[i, t] = 0 \Rightarrow kw[i, t] = 0$$

- For each component, if any amount of power is generated or used by the component in a time interval, the component must be *ON* for this time interval:

$$\forall (i \in \text{Components}, t \in T),$$

$$kw[i, t] \geq m \Rightarrow on[i, t] = 1$$

where $m \in \mathbb{R}$, and $m > 0$

- For each component, at any time interval, the value of control action $on[i, t]$ cannot exceed the value of control action $avail[i]$:

$$\forall (i \in \text{Components}, t \in T) \quad on[i, t] \leq avail[i]$$

- At any time interval, the amount of power produced or consumed by any component must be bounded by a minimum and a maximum value:

$$\forall (i \in \text{Components}, t \in T) \quad -M \times on[i, t] \leq kw[i, t] \leq M \times on[i, t]$$

where M is a constant.

- For each initially available component, the value of the control action $buyFlag[i]$ is 0 and that of $avail[i]$ is 1:

$$\forall(i \in \text{initAvaC}) \text{ buyFlag}[i] = 0$$

$$\forall(i \in \text{initAvaC}) \text{ avail}[i] = 1$$

- For each component that is not initially available, the value of the control action $avail[i]$ is equal to the value of the control action $buyFlag[i]$:

$$\forall(i \in \text{notInitAvaC}) \text{ avail}[i] = \text{buyFlag}[i]$$

- The demand-satisfaction ratio, which is explained in the next subsection, must be greater than or equal to an accepted value. For example, the microgrid energy system must satisfy at least 95% of the total power demand:

$$\text{demSatRatio}(A) \geq \text{accepted Value}$$

6.3.2 Utility Definition Constraints

In this subsection, I describe the global utility definition constraints for the entire model in terms of the component metrics. The global utility is the additive combination

of the net present value (NPV), the GHG emission, and the demand-satisfaction ratio for the entire model.

Net present value (NPV) utility. NPV is the difference between the present value of cash inflows and the present value of cash outflows, which is defined as

$$NPV = \sum_{t=1}^N \frac{F_t}{(1+r)^{t-1}}$$

where F_t is the net cash flow during an interval $t \in T$ and r is the discount rate, per interval t , used to determine the present value of future cash flows by taking into account the time-value of money.

For the proposed model, F is the total payments per each interval t , denoted by $payPerInt[t]$, which is defined as

$$\forall (i \in Components, t \in T) \text{ payPerInt}[t] = \sum_{i=1}^k \text{payPerComp}[i, t]$$

where $payPerComp[i, t]$ is the total number of payments for each component i at interval t , defined as

$$\text{payPerComp}[i, t] = \sum_{costCateg} \text{payPerCompCostCateg}[\langle i, costCateg \rangle, t]$$

and $payPerCompCostCate[< i, costCate >, t]$ is the total payments for each cost category of component i at interval t , defined as

$$\forall (i \in Components, t \in T, p \in \{1, \dots, noPayment[< i, costCate >]\}):$$

$$payPerCompCostCate[< i, costCate >, t] =$$

$$\begin{cases} payAmount[<< i, costCate >, p >] & : t \text{ in } payInterval[<< i, costCate >, p >] \\ 0 & : otherwise \end{cases}$$

Note that $payAmount[<< i, costCate >, p >]$, for each component type, is calculated as described in Section 6.4.

GHG emissions utility. The total GHG emissions, denoted by $totalCo2Value$, is the sum of the emissions generated by all components i . Formally,

$$totalCo2Value(A) = \sum_{i=1}^k compTotalCo2[i]$$

where $compTotalCo2[i]$, for each component type, is calculated as described in Section 6.4.

Demand-satisfaction ratio utility. The demand-satisfaction ratio, denoted by $demSatRatio$, is the ratio of the total supplied power to the total needed power:

$$demSatRatio(A) = totalSuppliedKW(A) \div totalDemandKW$$

where

$$totalSuppliedKW(A) = \sum_{i \in Services} kwPerSer[i]$$

$$kwPerSer[i] = \sum_{t=1}^N kw[i, t]$$

and

$$totalDemandKW = \sum_{i \in Services} demandPerSer[i]$$

$$demandPerSer[i] = \sum_{t=1}^N serPredictedDemand[i, t]$$

6.4 Power Component Modeling

In this section, I describe how the metrics are derived for each component type and the constraints based on the types of the components.

6.4.1 Utility Power Contract

A utility contract is a service agreement between a utility company and its business partners that defines all the commercial terms for the sale of power between the two parties, including the cost of using the power, the cost of the maximum peak demand

bound, and the penalty charge for exceeding this bound.

Cost:

Each contract component i has only one cost category ($conCost$) for each time interval. This typically includes both the peak demand charge and the total power consumption charge. The peak demand charge, denoted by $peakDemand$, measures the maximum rate of power consumption for any time interval. The total power consumption charge, denoted by $powerConsum$, measures the rate of power consumption in the specific billing period. Formally, $\forall i \in Contracts$, such that $Contracts \subset C$, and $\forall t \in T$, $conCost[i, t]$ is defined as

$$conCost[i, t] = powerConsum[i, t] + peakDemand[i, t],$$

where:

$$powerConsum[i, t] = kw[i, t] \times powerPricePerKw(kw[i, t]), \text{ and}$$

$$peakDemand[i, t] = kw[i, t] \times peakPricePerKw(kw[i, t])$$

The $powerPricePerKw$ and the $peakPricePerKw$ are functions that are represented as piece-wise or step-wise linear functions [119], for example:

$$powerPricePerKw(x) = \begin{cases} 12\text{¢}, & 0 < x \leq 3000 \\ 10\text{¢}, & 3000 < x \leq 6000 \\ 8\text{¢}, & x > 6000 \end{cases}$$

$$peakPricePerKw(x) = \begin{cases} 1.50\$, & 0 < x \leq 1000 \\ 3.00\$, & 1000 < x \leq 2000 \\ 6.00\$, & x > 2000 \end{cases}$$

Payments:

This cost category (*conCost*) has number of payments (*noPayment*) during the time horizon, and each payment *p* has an amount (*payAmount*) that is paid at a specific time interval (*payInterval*). For example, if this cost category is paid monthly, there will be 12 payments in a one-year time horizon. On an hourly-basis time horizon, the first payment (*p* = 1) will be made at *t* = 730, the second (*p* = 2) at *t* = 1460, and so on. Formally,

$\forall (i \in Contracts, t \in T, \text{ and } p \in \{1, \dots, noPayment[< i, costCateg >]\})$:

$noPayment[< i, conCost >] = \text{total number of months in } T \text{ (i.e., } N \div 730)$

$payInterval[< < i, conCost >, p >] = p \times 730,$

$payAmount[< < i, conCost >, p >] =$

$$\begin{cases} \sum_{payInterval[p-1] < t \leq payInterval[p]} conCost[i, t], & 1 < p \leq noPayment[< i, costCateg >] \\ \sum_{1 < t \leq payInterval[p]} conCost[i, t], & p = 1 \end{cases}$$

where $< < i, costCateg >, p >$ is a tuple associating a payment number with its cost category for each utility contract, and the constant 730 is the total number of hours in a month.

GHG emissions:

The total GHG emission of each contract component i , denoted by $conTotalCO2[i]$, is the sum of $conCO2$, which is the GHG emission of component i in each time interval. Formally, $\forall i \in Contracts$, $conTotalCO2[i]$ is defined as

$$conTotalCO2[i] = \sum_{t=1}^N conCO2[i, t]$$

where:

$$conCO2[i, t] = kw[i, t] \times conCO2perKw,$$

and $conCO2perKw$ is the GHG emission (in Btu) produced in the consumption of each kilowatt of power.

Operational and investment constraints:

At any time interval, the total kw consumption must not exceed the contract peak demand bound:

$$\forall (i \in Contracts, t \in T) \quad kw[i, t] \leq peakDemand[i]$$

6.4.2 Backup Power Generator

Large organizations usually have backup generators in case of power outages or for peak demand times. A generator requires fuel to operate and typically has an efficiency function of fuel consumption ($genEfficiency[i]$) based on the amount of power it generates. This function, which can be defined as a piece-wise linear function or a step-

wise linear function, determines the amount of fuel needed for each kilowatt generated.

Cost:

Every backup generator has three cost categories: fuel, maintenance, and depreciation, the difference between its present value and its residual value at the end of the time horizon. There is also a fourth cost category for each new generator purchased during the time horizon, which is equal its price. The fuel and depreciation cost categories are defined as follows:

$\forall i \in Generators$, such that $Generators \subset C$, and $\forall t \in T$:

$$genFuelCost[i, t] = fuelPrice[i, t] \times genEfficiency[i](kw[i, t]) \times kw[i, t]$$

$$genDeprCost[i] = genPresenValue[i] - genResidualValue[i]$$

Payments:

As indicated previously, each cost category has a number of payments during the time horizon, and each payment has an amount paid at a specific time interval. For example, while the *genFuelCost* could be paid monthly, with 12 payments in a time horizon of one year, the *genMaintCost* is paid annually and the *genDeprCost* is considered only at the end of the time horizon; that is, when $t = N$. Formally,

$\forall (i \in Generators, t \in T, p \in \{1, \dots, noPayment[< i, costCateg >]\})$:

$$noPayment[<< i, Fuel >>] = \text{total number of months in } T \text{ (i.e. } N \div 730)$$

$$noPayment[<< i, Maintenance >>] = \text{total number of years in } T \text{ (i.e. } N \div 8760)$$

$noPayment[<< i, Depreciation >>] = 1$ (i.e. only paid when $t = N$)

$noPayment[<< i, NewEquip >>] = 1$ (i.e. only paid when $t = 1$)

$payInterval[<< i, Fuel >, p >] = p \times 730$

$payInterval[<< i, Maintenance >, p >] = p \times 8760$

$payInterval[<< i, Depreciation >, p >] = N$

$payInterval[<< i, NewEquip >, p >] = 1$

$payAmount[<< i, Fuel >, p >] =$

$$\begin{cases} \sum_{payInterval[p-1] < t \leq payInterval[p]} genFuelCost[i, t], & 1 < p \leq noPayment[< i, costCateg >] \\ \sum_{1 < t \leq payInterval[p]} genFuelCost[i, t], & p = 1 \end{cases}$$

$payAmount[<< i, Maintenance >, p >] = genMaintCost[i] \times avail[i]$

$payAmount[<< i, Depreciation >, p >] = genDeprCost[i] \times avail[i]$

$payAmount[<< i, NewEquip >, p >] = genNewEquipCost[i] \times buyFlag[i]$

Note that constant 730 represents the total number of hours per month, and constant 8760 represents the total number of hours per year.

GHG emissions:

The total GHG emission of generator component i , denoted by $genTotalCO2[i]$, is the sum of $genCO2$, which is the GHG emission of component i in every time interval.

Formally, $\forall i \in \text{Generators}$, $genTotalCO2[i]$ is defined as

$$genTotalCO2[i] = \sum_{t=1}^N genCO2[i, t]$$

where:

$$genCO2[i, t] = kw[i, t] \times genCO2perKw,$$

and $genCO2perKw$ is the GHG emission (in *Btu*) produced by the consumption of each kilowatt of power.

Operational and investment constraints:

For any generator component, at any time interval, i.e., $\forall(i \in \text{Generators}, t \in T)$ the output power must not exceed the generator's capacity:

$$kw[i, t] \leq genCapacity[i]$$

6.4.3 Renewable Energy Resources

Renewable energy systems (e.g., photovoltaic systems, and wind turbines) generally supply energy that comes from natural sources, such as sunlight and wind. While these are much more environmentally friendly than non-renewable energy resources, they are much more expensive to use and depend on environmental factors like sunshine or wind activity, which makes it difficult to control their output power. The output of these components is represented as the predicted power (in kW) generated over

a time horizon, denoted by $predictedOutput[i, t] \forall (i \in Renewables, t \in T)$, such that $Renewables \subset C$.

Cost:

Each renewable resource has two main cost categories: annual maintenance and depreciation. As with the generators, there is also a cost of new equipment category for resources newly purchased during the time horizon. These categories are defined and calculated like the generator categories.

Payments:

$\forall (i \in Renewables, t \in T, p \in \{1, \dots, noPayment[< i, costCateg >]\})$, the $payAmount[<< i, costCateg >, p >]$, $payInterval[<< i, costCateg >, p >]$, and $noPayments[< i, costCateg >]$ are defined and calculated like the values for the generator components.

GHG emissions:

Because renewable resources do not typically use fuel to produce power, the total GHG emission of each renewable component i , denoted by $renTotalCO2[i]$, is equal to 0.

Operational and investment constraints:

In any time interval, the power used from any renewable resource component can not exceed the power generated by it:

$$\forall (i \in Renewables, t \in T) kw[i, t] \leq predictedOutput[i, t]$$

6.4.4 Battery Storage Units

A battery storage component is in one of three performance states at a given time: discharging (supplying power), charging (consuming power), or idle; that is, the value of

$kw[i,t]$ can be positive, negative, or 0. A battery typically has a limited number of design charge–discharge cycles, denoted by $batLifeCycles$, before it is considered inefficient and needs to be replaced.

Cost:

Like the renewable components, a battery has two main cost categories, maintenance and depreciation, and a third category for new battery purchases. The depreciation cost is the cost of wear caused by using the battery. To determine the cumulative number of cycles used, we divide the value of the used cumulative charge/discharge cycles of the battery, $cumChargeDischarge$, by the power in kilowatts for a single cycle life energy, $cycleEnergy$. This result, denoted by $cumCycles$, is multiplied by the new battery cost and divided by $batLifeCycles$. Formally, the depreciation cost is defined as follows:

$\forall i \in Batteries$, such that $Batteries \subset C$, and $\forall t \in T$:

$$batDeprCost[i] =$$

$$batNewEquipCost[i] \times cumCycles[i][t = N + 1] / batLifeCycles[i]$$

where:

$$cumCycles[i][t] = cumChargeDischarge[i] / cycleEnergy[i] ,$$

and

$$cumChargeDischarge[i] = \sum_{t=1}^N |kw[i,t]|$$

Payments:

$\forall (i \in Batteries, t \in T, p \in \{1, \dots, noPayment[< i, costCateg >]\})$; the values of $payAmount[<< i, costCateg >, p>]$, $payInterval[<< i, costCateg >, p>]$, and $noPayments[< i, costCateg >]$ are defined and calculated similarly to the values for the generator components.

GHG emissions:

As with the renewable resources, the total GHG emission of each battery component i , denoted by $batTotalCO2[i]$, is equal to 0.

Operational and investment constraints:

For any battery component, at any time interval, $\forall (i \in Batteries, t \in T)$,

- The discharge power amount must equal at least the minimum discharge rate:

$$kw[i, t] \geq minDischargeRate[i]$$

- The charging power amount must not exceed the maximum charge rate:

$$kw[i, t] \leq maxChargeRate[i]$$

- At the beginning of the time horizon, when $t = 1$, the discharge power amount cannot exceed the battery's initial energy:

$$kw[i, 1] \leq batInitialEnergy[i]$$

- For the remaining time intervals, where $t \in \{2, \dots, N\}$, the discharge power amount cannot exceed the battery's current charge. That is,

$$kw[i, t] \leq batCurrentCharge[i, (t - 1)], \text{ where}$$

$$batCurrentCharge[i, (t + 1)] = batCurrentCharge[i, t] - kw[i, t]$$

- The battery's current charge must not exceed its capacity:

$$batCurrentCharge[i, t] \leq batCapacity[i]$$

- At the beginning of the time horizon, when $t = 1$, the cumulative charge–discharge power is equal to zero:

$$cumChargeDischarge[i, 1] = 0$$

6.4.5 Power-Consuming Services

Power-consuming components are services that contribute to the power demand, such as HVAC, lighting, and water heating. Each service component i , requires an amount of power, denoted by *serPredictedDemand*, to run during each time interval.

Cost:

Because there are no power-related costs to operating a service other than the cost of supplying power, the cost of any service component for any time interval is 0:

$$\forall (i \in Services, t \in T) \text{ serCost}[i, t] = 0$$

Payments:

Because the cost is always 0, there are no payments for these components.

GHG emissions:

The total GHG emissions of each service component i , denoted by *serTotalCO2*[i], is 0.

Operational and Investment Constraints:

In any time interval, the power supplied to component i must be equal to the power needed if the service is *ON* and 0 otherwise:

$$\forall (i \in Services | Services \subset C, t \in T):$$

$$kw[i, t] = serPredictedDemand[i, t] \times on[i, t]$$

6.5 Summary

In this chapter, I showed how the proposed framework applies to a real problem by considering a case study in which PMOIR is used to support a group of decision makers by recommending a set of optimal operation and investment decisions regarding interrelated power components in the power microgrid of a university campus.

The energy operation and investment recommendations included optimal values for decision control variables, such as the amount of power generated from the power resource components and consumed by the power consuming services components at every time interval. The goal is to maximize the total NPV, within constraints on the demand-satisfaction ratio and the GHG emissions, while taking into account the components' interactions and satisfying a group of decision makers with possibly different views on the weights of various criteria. To implement PMOIR, I mathematically modeled renewable resources, batteries, backup generators, utility contracts, and power consuming services, and formalized the optimization problem, which involves all these components and their inter dependencies.

In the next chapter, I explain the implementation of the power optimization model for PMOIR. I also describe the experimental study I conducted to demonstrate the proposed framework's feasibility for applying PMOIR on medium and large power microgrids.

CHAPTER 7: PMOIR IMPLEMENTATION AND EXPERIMENTAL STUDY

7.1 Introduction

In the previous chapter, I mathematically described and modeled a library of specific power components. I also formalized the optimization for the entire problem, which involves the interrelations between all these components. In this chapter, I describe the implementation of the power optimization model for PMOIR that was formalized in the last chapter. I also describe an experimental study I conducted to demonstrate the proposed GCAR framework's feasibility, in terms of computational time, for applying PMOIR on power microgrids. The aim of this study is to determine whether the framework is practical for applying to realistically large problems to achieve near-optimal financial and operational results within a reasonable amount of time.

The specific contributions of this chapter are as follows: First, I implement the power optimization model for PMOIR as a mixed-integer linear programming (MILP) model using IBM's Optimization Programming Language (OPL) [120] and CPLEX Studio to decide on the power resource investment and operation. Second, I conduct an experimental study to demonstrate the GCAR framework's feasibility in computational time for recommending a small set of optimal investment and operation decisions in a case study using PMOIR and realistic data sets. This study is conducted with data set of multiple sizes involving different numbers of energy components over different time

horizons. The largest data set, which involves 200 components over a five-year time horizon, contains more than 23 million constraints and about 18 million variables, of which more than 8 million are binary and nearly 10 million are continuous. This data set is solved in less than five hours, which shows that the framework is feasible for use in medium-sized and large microgrids.

The remainder of this chapter is organized as follows: Section 7.2 demonstrates the OPL implementation of the entire optimization model and the optimization of the five power components. Section 7.3 presents the experimental study and its results. Section 7.4 summarizes the chapter.

7.2 OPL Implementation

In this section, I describe the OPL implementation of the entire optimization model and of the five power component models formalized in the last chapter. I begin by describing the implementation of the entire optimization model, and then for each power component, I describe how its general variables and parameters are defined, how the operational and investment metrics are driven, and how its specific constraints are modeled in OPL.

7.2.1 Entire Model Implementation

Figure 7.1 shows the general declarations of the time horizon, the five power components, the general metrics and weights, and the general decision variables. These correspond to the declarations described in Section 6.3.

```

// generic structures: time horizon, comps
float m = 000.1; // the minimal possible value
float M = 1000000.0; // the maximal possible metric value
float minDemSat = ...; // minimum accepted demand satisfaction ratio
int noTimeIntervals = ...;
range TimeHorizon = 1..noTimeIntervals;
float intervalLength = ...; // in hours

{string} initAvaContracts = ...; // set of ids; must be a subset of Components
{string} initAvaGenerators = ...;
{string} initAvaBatteries = ...;
{string} initAvaRenewables = ...;
{string} initAvaServices = ...;

// initial available components, subset of Components
{string} initAvaComponents = initAvaContracts union initAvaGenerators
    union initAvaBatteries union initAvaRenewables union initAvaServices;

{string} NotInitAvaGenerators = ...;
{string} NotInitAvaBatteries = ...;
{string} NotInitAvaRenewables = ...;

// Not initial available components, subset of Components
{string} NotInitAvaComponents = NotInitAvaGenerators union NotInitAvaBatteries
    union NotInitAvaRenewables;

{string} Contracts = initAvaContracts;
{string} Generators = initAvaGenerators union NotInitAvaGenerators ;
{string} Batteries = initAvaBatteries union NotInitAvaBatteries;
{string} Renewables = initAvaRenewables union NotInitAvaRenewables ;
{string} Services = initAvaServices;
{string} Components = initAvaComponents union NotInitAvaComponents;

// additive metrics' names, e.g., NPV, Co2, Demand Satisfaction Ratio, etc
{string} Metrics = ...;
float metricWeight[Metrics] = ...; // users weights for metrics

dvar float metricValue[Metrics];
dvar boolean on[Components][TimeHorizon]; // it is 1 if the component is ON
dvar boolean avail[Components]; // it is 1 if the component is available
dvar boolean buyFlag[Components];
dvar float kw[Components][TimeHorizon]; // Power in kw that a component gives/takes

```

Figure 7.1: General Declarations in OPL

Figure 7.2 describes the common component cost categories, as formally defined in the last chapter.

```

float newEquipCost[NotInitAvaComponents] = ...; // new equipment cost when purchased
{string} costCategory = ...; //e.g. maintenance, depreciation, fuel cost, etc.
int noPaymentPerCostCategory[costCategory] = ...;
tuple compCostCategory {
    string comp ; // subset of component IDs
    string costCtg; // subset of CostCategory
};
{compCostCategory} contractCostCategories =
    {<c,"ContCost"> | c in Contracts};
{compCostCategory} initAvaGeneratorsCostCategories =
    {<c,cc> | c in initAvaGenerators, cc in {"Fuel","Maint","Depr"}};
{compCostCategory} NotInitAvaGeneratorsCostCategories =
    {<c,cc> | c in NotInitAvaGenerators,
    cc in {"Fuel","Maint","Depr","NewEquipCost"}};
{compCostCategory} initAvaBatteriesCostCategories =
    {<c,cc> | c in initAvaBatteries, cc in {"Maint","Depr"}};
{compCostCategory} NotInitAvaBatteriesCostCategories =
    {<c,cc> | c in NotInitAvaBatteries,
    cc in {"Maint","Depr","NewEquipCost"}};
{compCostCategory} initAvaRenewablesCostCategories =
    {<c,cc> | c in initAvaRenewables, cc in {"Maint","Depr"}};
{compCostCategory} NotInitAvaRenewablesCostCategories =
    {<c,cc> | c in NotInitAvaRenewables,
    cc in {"Maint","Depr","NewEquipCost"}};

{compCostCategory} CompCostCategories = contractCostCategories union
    initAvaGeneratorsCostCategories union
    NotInitAvaGeneratorsCostCategories union
    initAvaBatteriesCostCategories union
    NotInitAvaBatteriesCostCategories union
    initAvaRenewablesCostCategories union
    NotInitAvaRenewablesCostCategories;

```

Figure 7.2: Common Component Cost Category Declarations in OPL

Figure 7.3 shows the payment amounts and the payment intervals declarations, which are needed for the NPV calculations. Figure 7.4 shows the declarations for other common metrics, such GHG emissions and the demand-satisfaction ratio, as well as the optimization statement. These correspond to the declarations described in Section 6.3.2.


```

tuple compPayPair {
    compCostCategory ccostctg ; // subset of compCostCategorys
    int paymentNo;
};
{compPayPair} CompPayPairs = {<<c,cc>,p> | <c,cc> in CompCostCategories,
    p in 1..noPayments[<c,cc>]};

int payInterval [CompPayPairs]= ...;
dvar float payAmount [CompPayPairs];
dvar float payPerCompCcInt[CompCostCategories][TimeHorizon];
dexpr float payPerCompInt[c in Components][t in TimeHorizon] =
    sum (<c,cc> in CompCostCategories) payPerCompCcInt[<c,cc>][t];
dexpr float payPerInt[t in TimeHorizon] =
    sum (c in Components) payPerCompInt[c][t];

float discountRate = ...;
// discount rate per each interval
float intDiscountRate = discountRate / noTimeIntervals;
float npvInterestPerInt = 1 + intDiscountRate;
float npvIndex[t in TimeHorizon] = npvInterestPerInt ^ (t - 1);
dexpr float deltaNpv[t in TimeHorizon] = payPerInt[t] / npvIndex[t];
dexpr float npv = sum (t in TimeHorizon)deltaNpv[t];

```

Figure 7.3: NPV Calculations in OPL

```

// Metrics general calculations

// CO2 emission metric
dvar float co2ValuePerComp[Components][{"CO2"}];
dexpr float totalCo2Value =
    sum(c in Components) co2ValuePerComp[c][{"CO2"}];
// Demand satisfaction ratio metric
dexpr float totalKwPerSer[s in Services] =
    sum (t in TimeHorizon) serKW[s][t];
dexpr float totalKwAllSer =
    sum (s in Services) totalKwPerSer[s];
dexpr float totalDemandPerSer[s in Services] =
    sum (t in TimeHorizon) serPredictedDemand[s][t];
dexpr float totalDemandAllSer =
    sum (s in Services) totalDemandPerSer[s];
dexpr float demandSatisRatio = totalKwAllSer / totalDemandAllSer;

dexpr float utility = sum(m in Metrics) metricWeight[m] * metricValue[m];

// optimization statement
maximize utility;

```

Figure 7.4: General Metrics Calculations in OPL

Finally, Figure 7.5 shows the common operational and investment constraints that need to

be satisfied, which were formally defined in Section 6.3.1.

```
// general and balancing constraints

constraints {

metricValue ["NPV"] == -npv;
metricValue ["DemSatisRatio"] == demandSatisRatio;
metricValue ["CO2"] == -totalCo2Value;

demandSatisRatio >= minDemSat;

forall (c in Components, t in TimeHorizon) {
    on[c][t] <= avail[c];
    -M * on[c][t] <= kw[c][t] && kw[c][t] <= M * on[c][t];
    (on[c][t] == 0) => (kw[c][t] == 0.0);
    (abs(kw[c][t]) >= m) => on[c][t] == 1;
};
forall (t in TimeHorizon) sum(c in Components)
    kw[c][t] == 0.0; // power balancing

forall (c in initAvaComponents) buyFlag[c] == 0;
forall (c in initAvaComponents) avail[c] == 1;
forall (c in NotInitAvaComponents) avail[c] == buyFlag[c];

forall (c in Components, cc in costCategory:
    <c,cc> in CompCostCategories, p in 1..noPayments[<c,cc>])
    payPerCompCcInt[<c,cc>][payInterval[<c,cc>,p]] == payAmount[<c,cc>,p];
forall (c in Components, cc in costCategory: <c,cc> in CompCostCategories,
    p in 2..noPayments[<c,cc>],
    t in payInterval[<c,cc>,p-1]+1..payInterval[<c,cc>,p] - 1)
    payPerCompCcInt[<c,cc>][t] == 0.0;
forall (c in Components, cc in costCategory: <c,cc> in CompCostCategories,
    t in 1..payInterval[<c,cc>,1] - 1)
    payPerCompCcInt[<c,cc>][t] == 0.0;
forall (c in NotInitAvaComponents, t in 2..noTimeIntervals)
    payPerCompCcInt[<c,"NewEquipCost">][t] == 0.0;
forall (c in Services, t in TimeHorizon)
    payPerCompInt[c][t] == 0.0;
forall (c in NotInitAvaComponents)
    payAmount[<c,"NewEquipCost">,1] == newEquipCost[c] * buyFlag[c];
```

Figure 7.5: Common Operational and Investment Constraints in OPL

7.2.2 Power Component Implementation

In this subsection, I describe the implementation of the five power component models formally defined in the last chapter. These components are utility contracts, backup generators, renewable resources, batteries, and power consuming services.

Utility Power Contract:

Figure 7.6 shows the declarations for the variables and parameters of the contract and the way its specific costs and GHG emissions are driven. The specific constraints for the utility power contract are shown in Figure 7.7. These all correspond to the description in Section 6.4.1.

```
// structure for contracts:
float conPeakDemandBound [Contracts]= ...;
float conPricePerKW [Contracts]= ...;
float conCo2PerKW [Contracts]= ...; //grams CO2 Emissions /kwh
dvar float+ conKW [Contracts][TimeHorizon];
// Contract Cost
dexpr float conCost[co in Contracts][t in TimeHorizon]=
    conKW[co][t] * conPricePerKW[co];
// Contract CO2 Emissions metric
dexpr float conCo2[co in Contracts][t in TimeHorizon]=
    conKW[co][t] * conCo2PerKW[co];
dexpr float conTotalCo2[co in Contracts]=
    sum (t in TimeHorizon) conCo2[co][t];
```

Figure 7.6: Utility Power Contract Parameters, Variables, and Metric Calculations in OPL

```
// Contract Constraints
forall (co in Contracts, t in TimeHorizon)
    conKW[co][t] <= conPeakDemandBound[co];
forall (co in Contracts, p in 2..noPayments[<co,"ContCost">])
    payAmount[<<co,"ContCost">,p>]==
        sum (t in payInterval[<<co,"ContCost">,p-1>] + 1..payInterval[<<co,"ContCost">,p>])
            conCost[co][t];
forall (co in Contracts) payAmount[<<co,"ContCost">,1>]==
    sum (t in 1..payInterval[<<co,"ContCost">,1>]) conCost[co][t];
//connections
forall (co in Contracts, t in TimeHorizon) kw[co][t] == conKW[co][t];
forall (co in Contracts) co2ValuePerComp[co]["CO2"] == conTotalCo2[co];
```

Figure 7.7: Utility Contract Operational and Investment Constrains in OPL

Backup Power Generator:

Figure 7.8 shows the implementation details of the parameters and decision variables for the backup generator and the calculations for its cost and GHG emission metrics. These correspond to the explanations in Section 6.4.2. The specific operational and investment constraints to be satisfied for each backup generator are shown in Figure 7.9.

```
// structure for generators

float genCapacity [Generators]= ...;
float fuelPrice [Generators][TimeHorizon]= ...;
float genEfficiency [Generators]= ...;
float genPresentValue [Generators]= ...;
float genResidualValue [Generators]= ...;
float genAnnualMaintCost [Generators]= ...;
float genCo2PerKW [Generators]= ...; //grams CO2 Emissions /kwh
dvar float+ genKW[Generators][TimeHorizon];
// generator Cost
dexpr float genFuelCost[g in Generators][t in TimeHorizon]=
    fuelPrice[g][t] * genEfficiency[g] * genKW[g][t];
dexpr float genMaintCost[g in Generators]= genAnnualMaintCost[g];
dexpr float genDeprCost[g in Generators] =
    genPresentValue[g] - genResidualValue[g];
// generator CO2 Emissions
dexpr float genCo2[g in Generators][t in TimeHorizon]=
    genKW[g][t] * genCo2PerKW[g];
dexpr float genTotalCo2[g in Generators]=
    sum (t in TimeHorizon) genCo2[g][t];
```

Figure 7.8: Backup Generator Parameters, Variables, and Metric Calculations in OPL

```

// Generator Constraints
forall (g in Generators, t in TimeHorizon) genKW[g][t] <= genCapacity[g];
forall (g in Generators, p in 2..noPayments[<g,"Fuel">])
    payAmount[<<g,"Fuel">,p>]==
        sum (t in payInterval[<<g,"Fuel">,p-1>] + 1..payInterval[<<g,"Fuel">,p>])
            genFuelCost[g][t];
forall (g in Generators) payAmount[<<g,"Fuel">,1>]==
    sum (t in 1..payInterval[<<g,"Fuel">,1>]) genFuelCost[g][t];
forall (g in Generators, p in 1..noPayments[<g,"Maint">])
    payAmount[<<g,"Maint">,p>] == genMaintCost[g] * avail[g];
forall (g in Generators, p in 1..noPayments[<g,"Depr">])
    payAmount[<<g,"Depr">,p>] == genDeprCost[g] * avail[g];
//connections
forall (g in Generators, t in TimeHorizon) kw[g][t] == genKW[g][t];
forall (g in Generators) co2ValuePerComp[g]["CO2"] == genTotalCo2[g];

```

Figure 7.9: Backup Generator Operational and Investment Constraints in OPL

Renewable Energy Resources:

The OPL implementation of the renewable components' parameters, decision variables, metrics, and constraints is shown in Figure 7.10, which corresponds to the explanations in Section 6.4.3.

```

// structure for renewable resources
float predictedOutput [Renewables][TimeHorizon]= ...;
float renPresentValue [Renewables]= ...;
float renResidualValue [Renewables]= ...;
dvar float+ renKW [Renewables][TimeHorizon];
float renAnnualMaintCost [Renewables]= ...;
// renewable NPV
dexpr float renMaintCost[r in Renewables]= renAnnualMaintCost[r];
dexpr float renDeprCost[r in Renewables]=
    renPresentValue[r] - renResidualValue[r];

// Renewable Constraints
forall (r in Renewables, t in TimeHorizon)
    renKW[r][t] <= predictedOutput[r][t];
forall (r in Renewables, p in 1..noPayments[<r,"Maint">])
    payAmount[<<r,"Maint">,p>] == renMaintCost[r] * avail[r];
forall (r in Renewables, p in 1..noPayments[<r,"Depr">])
    payAmount[<<r,"Depr">,p>] == renDeprCost[r] * avail[r];
//connections
forall (r in Renewables, t in TimeHorizon) kw[r][t] == renKW[r][t];
forall (r in Renewables) co2ValuePerComp[r]["CO2"] == 0.0;

```

Figure 7.10: Renewable Resource Parameters, Variables, Metrics, and Constraints in OPL

Battery Storage Units:

Figure 7.11 shows the battery parameters, decision variables, and metric calculations. The operational and investment constraints are shown in Figure 7.12. All of these correspond to the explanations in Section 6.4.4.

```
// structure for batteries

float batCapacity [Batteries]= ...;
float batInitialEnergy [Batteries]= ...;
float batMinDischargeRateKW [Batteries]= ...;
float batMaxChargeRateKW [Batteries]= ...;
float batAnnualMaintCost [Batteries]= ...;
float batLifeCycles [Batteries]= ...;//battery design number of lifecycles
float cycleEnergy [Batteries]= ...;//single cycle life energy
float newBatCost [Batteries]= ...;
dvar float+ batCurrentCharge[Batteries][1..noTimeIntervals+1];
dvar float batKW[Batteries][TimeHorizon];
dvar float+ cumChargeDischarge[Batteries][1..noTimeIntervals+1];
// battery Cost
dexpr float cumCycles[b in Batteries][t in 1..noTimeIntervals+1] =
    cumChargeDischarge[b][t] / cycleEnergy[b];
dexpr float batMaintCost[b in Batteries]= batAnnualMaintCost[b];
dexpr float batDeprCost[b in Batteries]=
    newBatCost[b] * npvIndex[noTimeIntervals]
    * cumCycles[b][noTimeIntervals+1] / batLifeCycles[b];
```

Figure 7.11: Battery Component Parameters, Variables, and Cost Calculations in OPL

```
// Battery Constraints

forall (b in Batteries, t in TimeHorizon) {
    batKW[b][t] >= batMinDischargeRateKW[b];
    batKW[b][t] <= batMaxChargeRateKW[b];
};
forall (b in Batteries) batCurrentCharge [b][1] == batInitialEnergy[b];
forall (b in Batteries, t in TimeHorizon)
    batCurrentCharge[b][t+1] == batCurrentCharge[b][t]- batKW[b][t];
forall (b in Batteries, t in TimeHorizon)
    batCurrentCharge[b][t+1] <= batCapacity[b];
forall (b in Batteries) {cumChargeDischarge[b][1] == 0.0;
    forall (t in TimeHorizon) {cumChargeDischarge[b][t+1] ==
        cumChargeDischarge[b][t] + abs(batKW[b][t]) * intervalLength;}}
forall (b in Batteries, p in 1..noPayments[<b,"Maint">])
    payAmount[<<b,"Maint">,p>] == batMaintCost[b] * avail[b];
forall (b in Batteries, p in 1..noPayments[<b,"Depr">])
    payAmount[<<b,"Depr">,p>] == batDeprCost[b] * avail[b];
//connections
forall (b in Batteries, t in TimeHorizon) kw[b][t] == batKW[b][t];
forall (b in Batteries) co2ValuePerComp[b][<"CO2">] == 0.0;
forall (b in Batteries,t in TimeHorizon)(on[b][t]==0) =>(batKW[b][t]==0.0);
```

Figure 7.12: Battery Component Operational and Investment Constraints in OPL

Power-Consuming Services:

Finally, Figure 7.13 shows the parameters and decision variables of the power-consuming components. It also shows how the related metrics are calculated, and how the specific operational and investment constraints are implemented in OPL. All of these correspond to the explanations in Section 6.4.5.

```
// structure for services

float serPredictedDemand [Services][TimeHorizon]= ...;
dexpr float serKW[s in Services][t in TimeHorizon]=
    serPredictedDemand[s][t] * on[s][t];

// Services Constraints

//connections
forall (s in Services) co2ValuePerComp[s]["CO2"] == 0.0;
forall (s in Services, t in TimeHorizon) kw[s][t] == -serKW[s][t] ;

}
```

Figure 7.13: Power-Consuming Service Parameters and Constraints in OPL

7.3 Experimental Study

The question I address in this experiment is whether the proposed GCAR formulation for a realistically large case study is practical for generating a small set of optimal and diverse solutions within a reasonable amount of time. To answer this question, I apply the GCAR framework to a realistic case study using PMOIR.

7.3.1 Experimental Settings and Methodology

In this study, the utility function of each member of a group of three decision

makers is generated synthetically. The group utility function is estimated using the average group decision-making method that was detailed in Chapter 3.

The study considers three different microgrid sizes: a small microgrid of up to 50 power components, a medium-sized microgrid of up to 100 components, and a large microgrid of up to 200 components. All the power components belong to the types described and modeled in Chapter 6. In addition, all three microgrids are operated over three different time horizons: one year, three years, and five years.

Ten data sets are generated and tested for each microgrid over each time horizon, in order to measure the efficiency of the branch and bound, and branch and cut of MILP Solver. These data sets are generated with real data from [121], which provides annual energy usage information (in kWh) from 1989 to 2010 at the University of Texas at Austin.

For the optimization step, OPL is used with the ILOG CPLEX Solver engine [122], and a 1% relative gap is set as the stopping criteria for CPLEX. The testing is performed on 2.6 workstation with Intel Core i7 processor and a memory of 16 GB 1600 MHz DDR.

The optimization model, which was formalized in the last chapter, is applied in this case study with a variation to support the diversity step of the GCAR framework. After the group's total utility has been maximized, each decision maker's utility is also maximized in turn, subject to an additional constraint: the total utility is bounded by the maximum group utility minus ϵ , where ϵ corresponds to 2% of the maximum group utility score. The idea is to create a subset of diverse recommendations, each of them

based on a different individual's utility function, while preserving a bounded distance from the optimal group utility score to provide a balance between optimality and diversity, as described in Chapter 4.

The generated data follows a typical daily system-demand profile, as shown in Figure 7.14 [123]. To estimate the demand for the next five years, a 1.4% annual growth rate is assumed, which is consistent with work [124].

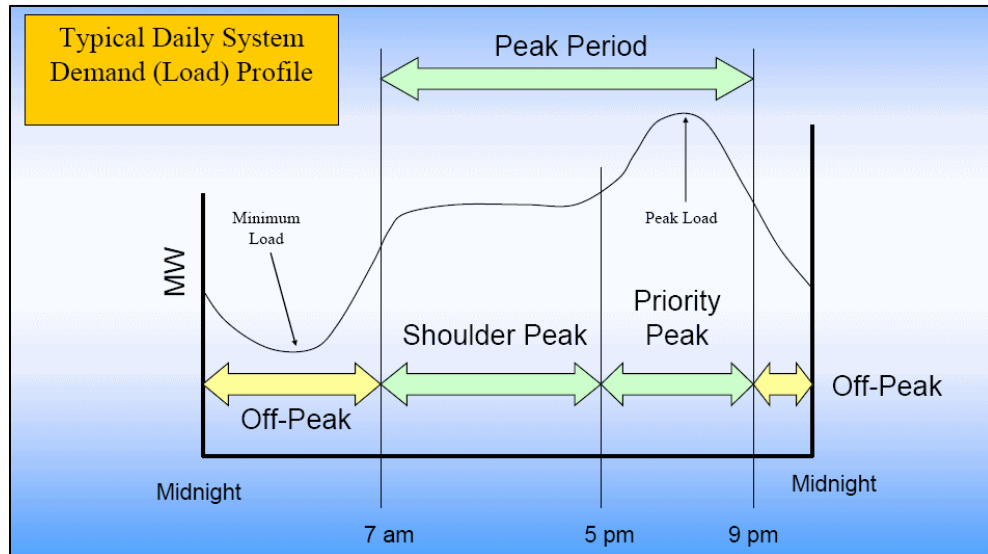


Figure 7.14: Typical Yearly Load Profile [123]

Following Figure 7.14, the peak hours are from 5 p.m. to 9 p.m., the mid-peak hours from 7 a.m. to 5 p.m., and the off-peak hours from 9 p.m. to 7 a.m. The hourly prices for power generation are generated by following the curve in Figure 7.15 from work [125]. They show that on a typical day, the price varies between 2 and 4 ¢ per kilowatt-hour in the early morning and reaches 8¢ at peak usage time.

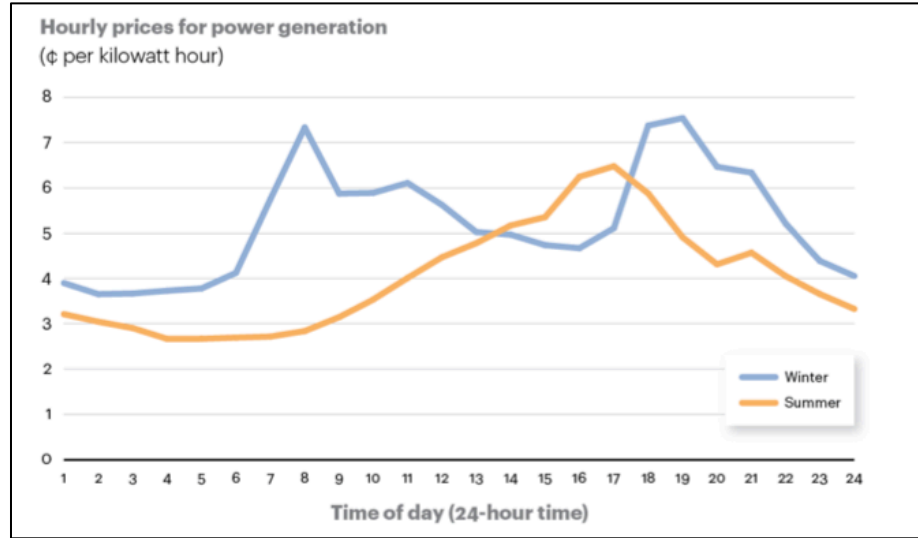


Figure 7.15: Hourly Prices for Power Generation [125]

7.3.2 Experimental Results

The mean resolution times for all three microgrid sizes over the three time horizons are depicted in Figure 7.16. The largest data set in this study, which involves 200 components over a five-year time horizon, contains over 23 million constraints, and about 18 million variables, of which over 8 million variables are binary and nearly 10 million variables are continuous. This largest dataset is solved in less than five hours of solver time, meaning the proposed framework is feasible for use with medium-sized and large microgrids to generate a small set of optimal and diverse solutions within a reasonable time. Note that the resolution time includes the time required for the group utility optimization and the optimization over each decision maker's utility to diversify the recommendation set.

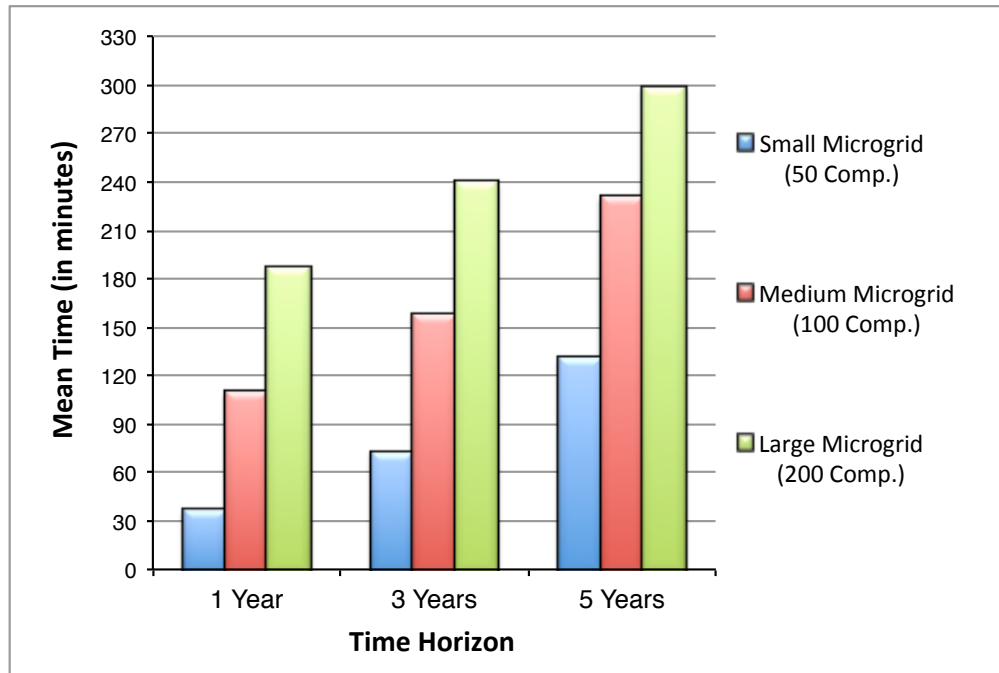


Figure 7.16: Experimental Mean Resolution Time for Three Microgrid Sizes over Three Time Horizons

For the statistical analysis, I calculate the CI at a 95% level for the estimated mean of the time resolution for each microgrid size and over each time horizon. The results are illustrated in Table 7.1: the mean resolution time for the smallest data set (50 components over a one-year time horizon) is 36.49 minutes, with upper and lower bounds of 1.22 minutes. The mean resolution time for the largest data set (200 components over a five-year time horizon) is 299.02 minutes, with upper and lower bounds of 4.22 minutes.

Table 7.1: Confidence Interval at Level 95% for the Estimated Mean of the Time Resolution (in Minutes)

Microgrid Size	Time Horizon (in years)		
	1	3	5
Small (50 Components)	36.49 ± 1.22	73.09 ± 2.17	130.38 ± 2.13
Medium (100 Components)	109.06 ± 1.47	158.15 ± 2.34	230.21 ± 3.28
Large (200 Components)	187.06 ± 2.12	239.36 ± 3.36	299.02 ± 4.22

7.3.3 Post-Processing Phase

After the diverse set of recommendations is generated, they are to be presented to each decision maker in descending order of group utility, and each decision maker ranks the set in accordance with his or her preferences. Finally, one of the voting methods described in Chapter 3 is applied on the ranked set of recommendations to determine the final top k recommendations.

7.4 Summary

In this chapter, I described the implementation of the power optimization model for PMOIR as a MILP model using OPL and the CPLEX solver to decide on power resource investment and operations on an hourly-basis time horizon. I also described a study I conducted to show the framework's feasibility in actual scenarios. This study involved multiple data set sizes and time horizons. The largest data set was solved in less than five hours, showing that the framework is practicable for its intended purposes.

CHAPTER 8: CONCLUSIONS AND FUTURE WORK

In this chapter, I summarize the research presented in this dissertation and suggest directions for future work.

8.1 Conclusions

State-of-the-art recommender systems focus on atomic products and services and on individual users. This research deals with extending recommender systems in three ways: (1) to consider composite recommendations; (2) to deal with multiple, rather than single, criteria for recommendations; and (3) to support groups of diverse users with different, even conflicting, views on weights for different criteria. To the best of my knowledge, based on popular group recommender surveys [44, 49, 51-54], no existing recommender system works with groups of users and composite, multi-criteria recommendations. The proposed system is the first to address these issues.

I have proposed a decision-guided group composite alternatives recommender framework based on multi-criteria decision optimization and voting. This framework works on a very large, or even infinite, recommendation space, which is implicitly defined by mathematical constraints. It is designed to provide a diverse set of near-optimal package recommendations to groups of users, while taking into account the different influence of individuals in the group, the interest dissimilarity among them, and the size and homogeneity of the group. The framework applies six decision-making

methods to refine its recommendations. Five of these come from social choice theories— instant runoff voting (IRV), hybrid Condorcet-IRV, average, least misery, and average without misery—and the last, the structurally adjusted average method, I developed. In addition, I propose a technique for scaling this framework up to handle very large, heterogeneous groups. This technique involves clustering the large group into small, homogeneous subgroups of decision makers with similar utilities.

I also developed a power microgrid operation and investment recommender (PMOIR) as a case study to demonstrate the applicability of the framework. PMOIR recommends a set of operation and investment decisions involving the components of a power microgrid to a group of decision makers who need to maximize the NPV within bounds set on the demand-satisfaction ratio and the GHG emissions while taking into account all the component's interactions, and multiple potentially conflicting views on the importance of various criteria. To implement this, I modeled the different components, formalized the optimization problem, and implemented the power optimization model as a mixed-integer linear programming (MILP) model using OPL and CPLEX Studio.

Finally, I validated the proposed framework with three experimental studies:

1. A study comparing the precision and recall of the framework to the results of manual voting by human participants. This study showed that for each decision-making method, the average precision and recall achieved by the framework ranged from being the same as the ideal (for top-1 recommendations) to being up to 34% off it (for top-4 recommendations).

2. A study demonstrating the framework's scalability to very large, heterogeneous groups. This study compared the precision and recall of the framework to recommendations that would be generated by manual voting by the entire groups. Utility functions for the large groups' members were generated synthetically. The study found an average precision ranging from 0.95 (for top-1 recommendations) to 0.80 (for top-5 recommendations) and an average recall of 0.19 (for top-1 recommendations) to 0.80 (for top-5 recommendations).
3. A study demonstrating the feasibility, in terms of computational time, of the framework in a realistic case study using PMOIR. This study used multiple data set sizes and time horizons to demonstrate that the proposed framework is feasible for generating small sets of optimal and diverse solutions in a reasonable time with medium-sized and large microgrids.

8.2 Future Work

Further exploration is possible in many areas:

- The recommendation process could be sped up by leveraging historical data. The goal would be to identify a new user's utility function by analyzing a database of previously collected utility functions.
- Different group decision-making methods are used by different people, and the choice usually depends on the domain, the group's characteristics, and the property people want to satisfy. No single voting method is superior to all others and fully fair. In the GCAR framework, I consider six methods, but others, such as Borda, Kemeny, and Copeland, could be applied to it in future work.

- I have assumed that the individuals in a group have already agreed on the overall set of criteria, simply not on their weights. However, in future work this assumption could be relaxed and the criteria themselves examined and refined.
- Future research can model a range of power and renewable energy components going beyond the five formalized and modeled in this work.
- In the microgrid problem I considered, I assumed for simplicity that new investment equipment was bought in the first interval of the time horizon. In future work, we could decide not only whether to buy a new component, but when to buy it.
- I addressed only the net present value of the microgrid power components; however, it would be worth addressing the possibility of leveraging the existing energy market to sell excess capacity during time intervals with low demands.
- A good user-interface design can be incorporated into the framework to increase group members' satisfaction.

APPENDIX A: GMU INSTITUTIONAL REVIEW BOARD (IRB) APPROVAL LETTER



Office of Research Integrity and Assurance

Research Hall, 4400 University Drive, MS 6D5, Fairfax, Virginia 22030
Phone: 703-993-5445; Fax: 703-993-9590

DATE: December 12, 2013

TO: Alexander Brodsky
FROM: George Mason University IRB

Project Title: [486276-1] GCAR: A Group Composite Alternatives Recommender Based on Multi-Criteria Optimization and Voting

SUBMISSION TYPE: New Project

ACTION: APPROVED
APPROVAL DATE: December 12, 2013
EXPIRATION DATE: December 11, 2014
REVIEW TYPE: Expedited Review

REVIEW TYPE: Expedited review category #7

Thank you for your submission of New Project materials for this project. The George Mason University IRB has APPROVED your submission. This submission has received Expedited Review based on applicable federal regulations.

Please remember that all research must be conducted as described in the submitted materials.

Please remember that informed consent is a process beginning with a description of the project and insurance of participant understanding followed by a signed consent form. Informed consent must continue throughout the project via a dialogue between the researcher and research participant. Federal regulations require that each participant receives a copy of the consent document.

Please note that any revision to previously approved materials must be approved by the IRB prior to initiation. Please use the appropriate revision forms for this procedure.

All UNANTICIPATED PROBLEMS involving risks to subjects or others and SERIOUS and UNEXPECTED adverse events must be reported promptly to the Office of Research Integrity & Assurance (ORIA). Please use the appropriate reporting forms for this procedure. All FDA and sponsor reporting requirements should also be followed (if applicable).

All NON-COMPLIANCE issues or COMPLAINTS regarding this project must be reported promptly to the ORIA.

The anniversary date of this study is December 11, 2014. This project requires continuing review by this committee on an annual basis. You may not collect data beyond this date without prior IRB approval. A continuing review form must be completed and submitted to the ORIA at least 30 days prior to the

anniversary date or upon completion of this project. Prior to the anniversary date, the ORIA will send you a reminder regarding continuing review procedures.

Please note that all research records must be retained for a minimum of three years, or as described in your submission, after the completion of the project.

If you have any questions, please contact Bess Dieffenbach at 703-993-4121 or edieffen@gmu.edu. Please include your project title and reference number in all correspondence with this committee.

This letter has been electronically signed in accordance with all applicable regulations, and a copy is retained within George Mason University IRB's records.

APPENDIX B: PUBLICATIONS RELATED TO THIS DISSERTATION

This research has resulted in the following publications:

- Mengash, H., and Brodsky, A., "GCAR: A Group Composite Alternatives Recommender Based on Multi-Criteria Optimization and Voting", In: Proceedings of the 47th Hawaii International Conference on System Science (HICSS), January 6-9, 2014, IEEE Computer Society Press, 2014, pp.1113-1121.
- Mengash, H., and Brodsky, A., "DG-GPR: A Decision-Guided Group Package Recommender with Hybrid Condorcet-Instant Runoff Voting", in DSS 2.0, Supporting Decision Making with New Technologies. Frontiers in Artificial Intelligence and Applications, IOS Press, Paris, France, 2014, pp. 317 - 328.
- Mengash, H., and Brodsky, A., "A Group Package Recommender Based on Learning Group Preferences, Multi-Criteria Decision Analysis, and Voting", EURO Journal on Decision Processes, Springer Berlin Heidelberg, vol. 3, pp. 275-304, 2015.
- Mengash, H., and Brodsky, A., "Tailoring Group Package Recommendations to Large Heterogeneous Groups Based on Multi-Criteria Optimization", In: Proceedings of the 49th Hawaii International Conference on System Science (HICSS), January 5-8, 2016, IEEE Computer Society Press, 2016, pp.1537-1546.

REFERENCES

- [1] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Trans. on Knowl. and Data Eng.*, vol. 17, pp. 734-749, 2005.
- [2] F. Cacheda, V. Carneiro, D. Fernandez, and V. Formoso, "Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems," *ACM Trans. Web*, vol. 5, pp. 1-33, 2011.
- [3] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative Filtering beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges," *ACM Comput. Surv.*, vol. 47, pp. 1-45, 2014.
- [4] R. Burke, "Hybrid Recommender Systems: Survey and Experiments," *User Modeling and User-Adapted Interaction*, vol. 12, pp. 331-370, 2002.
- [5] N. Manouselis and C. Costopoulou, "Analysis and Classification of Multi-Criteria Recommender Systems," *World Wide Web*, vol. 10, pp. 415-441, 2007.
- [6] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl, "PolyLens: a recommender system for groups of users," presented at the Proceedings of the seventh conference on European Conference on Computer Supported Cooperative Work, Bonn, Germany, 2001.
- [7] L. Quijano-Sanchez, J. A. Recio-Garcia, and B. Diaz-Agudo, "HappyMovie: A Facebook Application for Recommending Movies to Groups," presented at the Proceedings of the 2011 IEEE 23rd International Conference on Tools with Artificial Intelligence, 2011.
- [8] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," presented at the Proceedings of the fourth ACM conference on Recommender systems, Barcelona, Spain, 2010.
- [9] L. M. Campos, J. M. Fernandez-Luna, J. F. Huete, and M. A. Rueda-Morales, "Managing uncertainty in group recommending processes," *User Modeling and User-Adapted Interaction*, vol. 19, pp. 207-242, 2009.

- [10] Z. Yu, X. Zhou, Y. Hao, and J. Gu, "TV Program Recommendation for Multiple Viewers Based on user Profile Merging," *User Modeling and User-Adapted Interaction*, vol. 16, pp. 63-82, 2006.
- [11] L. N. Dery, M. Kalech, L. Rokach, and B. Shapira, "Iterative voting under uncertainty for group recommender systems," presented at the Proceedings of the fourth ACM conference on Recommender systems, Barcelona, Spain, 2010.
- [12] C. Senot, D. Kostadinov, M. Bouzid, J. Picault, A. Aghasaryan, and C. Bernier, "Analysis of Strategies for Building Group Profiles," in *User Modeling, Adaptation, and Personalization*. vol. 6075, P. De Bra, A. Kobsa, and D. Chin, Eds., ed: Springer Berlin Heidelberg, 2010, pp. 40-51.
- [13] J. F. McCarthy and T. D. Anagnost, "MusicFX: an arbiter of group preferences for computer supported collaborative workouts," presented at the Proceedings of the 1998 ACM conference on Computer supported cooperative work, Seattle, Washington, USA, 1998.
- [14] G. Popescu and P. Pu, "What's the best music you have?: designing music recommendation for group enjoyment in groupfun," presented at the Proceedings of the 2012 ACM annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts, Austin, Texas, USA, 2012.
- [15] D. L. Chao, J. Balthrop, and S. Forrest, "Adaptive radio: achieving consensus using negative preferences," presented at the Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work, Sanibel Island, Florida, USA, 2005.
- [16] J. Masthoff, "The Pursuit of Satisfaction: Affective State in Group Recommender Systems," in *User Modeling 2005*. vol. 3538, L. Ardissono, P. Brna, and A. Mitrovic, Eds., ed: Springer Berlin Heidelberg, 2005, pp. 297-306.
- [17] S. Berkovsky and J. Freyne, "Group-based recipe recommendations: analysis of data aggregation strategies," presented at the Proceedings of the fourth ACM conference on Recommender systems, Barcelona, Spain, 2010.
- [18] I. Cantador and P. Castells, "Extracting multilayered Communities of Interest from semantic user profiles: Application to group modeling and hybrid recommendations," *Comput. Hum. Behav.*, vol. 27, pp. 1321-1336, 2011.
- [19] S. Pizzutilo, B. D. Carolis, G. Cozzolongo, and F. Ambruso, "Group modeling in a public space: methods, techniques, experiences," presented at the Proceedings of the 5th WSEAS International Conference on Applied Informatics and Communications, Malta, 2005.
- [20] B. Smyth, E. Balfe, J. Freyne, P. Briggs, M. Coyle, and O. Boydell, "Exploiting Query Repetition and Regularity in an Adaptive Community-Based Web Search

- Engine," *User Modeling and User-Adapted Interaction*, vol. 14, pp. 383-423, 2004.
- [21] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso, "Tailoring the Recommendation of Tourist Information to Heterogeneous User Groups," presented at the Revised Papers from the International Workshops OHS-7, SC-3, and AH-3 on Hypermedia: Openness, Structural Awareness, and Adaptivity, 2002.
 - [22] I. Garcia, L. Sebastia, E. Onaindia, and C. Guzman, "A Group Recommender System for Tourist Activities," presented at the Proceedings of the 10th International Conference on E-Commerce and Web Technologies, Linz, Austria, 2009.
 - [23] A. Jameson, "More than the sum of its members: challenges for group recommender systems," presented at the Proceedings of the working conference on Advanced visual interfaces, Gallipoli, Italy, 2004.
 - [24] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso, "INTRIGUE: Personalized recommendation of tourist attractions for desktop and handset devices," *Applied Artificial Intelligence*, vol. 17, pp. 687-714, 2003.
 - [25] J. F. McCarthy, "Pocket Restaurant Finder: A situated recommender systems for groups," in *Proceeding of the ACM CHI 2002 International Workshop on Mobile Ad-Hoc Communication*, 2002.
 - [26] F. Lorenzi, S. Loh, and M. Abel, "PersonalTour: A Recommender System for Travel Packages," presented at the Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 02, 2011.
 - [27] T. Deng, W. Fan, and F. Geerts, "On the complexity of package recommendation problems," presented at the Proceedings of the 31st symposium on Principles of Database Systems, Scottsdale, Arizona, USA, 2012.
 - [28] R. Interdonato, S. Romeo, A. Tagarelli, and G. Karypis, "A Versatile Graph-based Approach to Package Recommendation," 2013.
 - [29] G. Koutrika, B. Bercovitz, and H. Garcia-Molina, "FlexRecs: expressing and combining flexible recommendations," presented at the Proceedings of the 2009 ACM SIGMOD International Conference on Management of data, Providence, Rhode Island, USA, 2009.
 - [30] T. Lappas, K. Liu, and E. Terzi, "Finding a team of experts in social networks," presented at the Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, Paris, France, 2009.

- [31] A. G. Parameswaran, H. Garcia-Molina, and J. D. Ullman, "Evaluating, combining and generalizing recommendations with prerequisites," presented at the Proceedings of the 19th ACM international conference on Information and knowledge management, Toronto, ON, Canada, 2010.
- [32] A. Parameswaran, P. Venetis, and H. Garcia-Molina, "Recommendation systems with complex constraints: A course recommendation perspective," *ACM Trans. Inf. Syst.*, vol. 29, pp. 1-33, 2011.
- [33] M. Xie, L. V. S. Lakshmanan, and P. T. Wood, "Breaking out of the box of recommendations: from items to packages," presented at the Proceedings of the fourth ACM conference on Recommender systems, Barcelona, Spain, 2010.
- [34] A. Brodsky, S. M. Henshaw, and J. Whittle, "CARD: a decision-guidance framework and application for recommending composite alternatives," presented at the Proceedings of the 2008 ACM conference on Recommender systems, Lausanne, Switzerland, 2008.
- [35] K. Alodhaibi, A. Brodsky, and G. A. Mihaila, "COD: Iterative Utility Elicitation for Diversified Composite Recommendations," in *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, 2010, pp. 1-10.
- [36] K. Bradley and B. Smyth, "Improving Recommendation Diversity," ed, 2001.
- [37] B. Smyth and P. McClave, "Similarity vs. Diversity," presented at the Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development, 2001.
- [38] G. Adomavicius and Y. Kwon, "New Recommendation Techniques for Multicriteria Rating Systems," *IEEE Intelligent Systems*, vol. 22, pp. 48-55, 2007.
- [39] K. Lakiotaki, S. Tsafarakis, and N. Matsatsinis, "UTA-Rec: a recommender system based on multiple criteria analysis," presented at the Proceedings of the 2008 ACM conference on Recommender systems, Lausanne, Switzerland, 2008.
- [40] N. Manouselis and C. Costopoulou, "Experimental Analysis of Design Choices in multiattribute Utility Collaborative Filtering," *International Journal of Pattern Recognition and Artificial Intelligence - IJPRAI*, vol. 21(2), pp. 311-332, 2007.
- [41] U. Bose, A. M. Davey, and D. L. Olson, "Multi-attribute utility methods in group decision making: Past applications and potential for inclusion in GDSS," *Omega*, vol. 25, pp. 691--706, 1997.
- [42] P. Csáki, T. Rapesák, P. Turchányi, and M. Vermes, *Research and development for group decision aid in Hungary by WINGDSS, a Microsoft Windows based group decision support system. (Working paper of the Laboratory of Operations Research and Decision Systems (LORDS) WP 93-9.)*: MTA SZTAKI, 1993.

- [43] N. Arora and G. Allenby, "Measuring the Influence of Individual Preference Structures in Group Decision Making," *Journal of Marketing Research*, vol. 36, 1999.
- [44] J. Masthoff, "Group Recommender Systems: Combining Individual Models," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., ed: Springer US, 2011, pp. 677-702.
- [45] K. McCarthy, L. McGinty, B. Smyth, M. Salam, and #243, "The needs of the many: a case-based group recommender system," presented at the Proceedings of the 8th European conference on Advances in Case-Based Reasoning, Fethiye, Turkey, 2006.
- [46] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, *et al.*, "Enhancing group recommendation by incorporating social relationship interactions," presented at the Proceedings of the 16th ACM international conference on Supporting group work, Sanibel Island, Florida, USA, 2010.
- [47] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu, "Group recommendation: semantics and efficiency," *Proc. VLDB Endow.*, vol. 2, pp. 754-765, 2009.
- [48] M. Kompan and M. Bielikova, "Group Recommendations: Survey and Perspectives," *Computing and Informatics*, pp. 1-31, 2013.
- [49] L. Quijano-Sanchez, J. A. Recio-Garcia, B. Diaz-Agudo, and G. Jimenez-Diaz, "Social factors in group recommender systems," *ACM Trans. Intell. Syst. Technol.*, vol. 4, pp. 1-30, 2013.
- [50] J. K. Kim, H. K. Kim, H. Y. Oh, and Y. U. Ryu, "A group recommendation system for online communities," *International Journal of Information Management*, vol. 30, pp. 212-219, 2010.
- [51] L. Boratto and S. Carta, "State-of-the-Art in Group Recommendation and New Approaches for Automatic Identification of Groups," in *Information Retrieval and Mining in Distributed Environments*. vol. 324, A. Soro, E. Vargiu, G. Armano, and G. Paddeu, Eds., ed: Springer Berlin Heidelberg, 2011, pp. 1-20.
- [52] I. Cantador and P. Castells, "Group Recommender Systems: New Perspectives in the Social Web," in *Recommender Systems for the Social Web*. vol. 32, ed: Springer Berlin Heidelberg, 2012, pp. 139-157.
- [53] P. Pu, L. Chen, and R. Hu, "Evaluating recommender systems from the user's perspective: survey of the state of the art," *User Modeling and User-Adapted Interaction*, vol. 22, pp. 317-355, 2012.

- [54] M. Kompan and M. Bielikova, "Group Recommendations: Survey and Perspectives," *Computing and Informatics*, vol. 33, pp. 446–476, 2014.
- [55] N. A. Najjar and D. C. Wilson, "Evaluating group recommendation strategies in memory-based collaborative filtering," in *Joint Workshop on Human Decision Making in Recommender Systems, Decisions@RecSys 2011 and User-Centric Evaluation of Recommender Systems and Their Interfaces-2, UCERSTI 2 - Affiliated with the 5th ACM Conference on Recommender Systems, RecSys 2011*, Chicago, IL, United states, 2011, pp. 43 - 50.
- [56] G. Popescu and P. Pu, "Group Recommender Systems as a Voting Problem," Semester report. Swiss Federal Institute of Technology in Lausanne (EPFL), Lausanne, Switzerland 2011.
- [57] R. J. Mooney and L. Roy, "Content-based book recommending using learning for text categorization," presented at the Proceedings of the fifth ACM conference on Digital libraries, San Antonio, Texas, USA, 2000.
- [58] A. Said, "Identifying and utilizing contextual data in hybrid recommender systems," 2010, pp. 365-368.
- [59] A. Gunawardana and C. Meek, "A unified approach to building hybrid recommender systems," presented at the Proceedings of the third ACM conference on Recommender systems, New York, New York, USA, 2009.
- [60] L. Pizzato, T. Rej, T. Chung, I. Koprinska, and J. Kay, "RECON: a reciprocal recommender for online dating," presented at the Proceedings of the fourth ACM conference on Recommender systems, Barcelona, Spain, 2010.
- [61] S. Zhao, M. X. Zhou, Q. Yuan, X. Zhang, W. Zheng, and R. Fu, "Who is talking about what: social map-based recommendation for content-centric social websites," presented at the Proceedings of the fourth ACM conference on Recommender systems, Barcelona, Spain, 2010.
- [62] A. Angel, S. Chaudhuri, G. Das, and N. Koudas, "Ranking objects based on relationships and fixed associations," presented at the Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, Saint Petersburg, Russia, 2009.
- [63] M. D. Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu, "Automatic construction of travel itineraries using social breadcrumbs," presented at the Proceedings of the 21st ACM conference on Hypertext and hypermedia, Toronto, Ontario, Canada, 2010.
- [64] A. G. Parameswaran and H. Garcia-Molina, "Recommendations with prerequisites," presented at the Proceedings of the third ACM conference on Recommender systems, New York, New York, USA, 2009.

- [65] G. Popescu, "Group Recommender Systems as a Voting Problem," in *Online Communities and Social Computing*. vol. 8029, A. A. Ozok and P. Zaphiris, Eds., ed: Springer Berlin Heidelberg, 2013, pp. 412-421.
- [66] I. Ntoutsi, K. Stefanidis, K. Norvag, and H.-P. Kriegel, "gRecs: A Group Recommendation System Based on User Clustering," in *Database Systems for Advanced Applications*. vol. 7239, S.-g. Lee, Z. Peng, X. Zhou, Y.-S. Moon, R. Unland, and J. Yoo, Eds., ed: Springer Berlin Heidelberg, 2012, pp. 299-303.
- [67] A. Jameson and B. Smyth, "What a Difference a Group Makes: Web-Based Recommendations for Interrelated Users." vol. 4321, P. Brusilovsky, Kobsa, A., Nejd, W. (eds.) *The Adaptive Web*. LNCS, Ed., ed: Springer, Heidelberg, 2007.
- [68] O. Van Deventer, J. De Wit, J. Vanattenhoven, and M. Gualbahar, "Group Recommendation in a Hybrid Broadcast Broadband Television Context," in *GroupRS 2013: Group Recommender Systems: Concepts, Technology, Evaluation*, 2013, pp. 12 - 18.
- [69] J. Masthoff, "Group Modeling: Selecting a Sequence of Television Items to Suit a Group of Viewers," *User Modeling and User-Adapted Interaction*, vol. 14, pp. 37-85, 2004.
- [70] R. Hastie and T. Kameda, "The robust beauty of majority rules in group decisions," *Psychological Review*, vol. 112, pp. 494 - 508, 2005.
- [71] Lippman and David, *Math in Society*: CreatSpace Independent Publishing Platform, 2012.
- [72] D. Cary, "Estimating the margin of victory for instant-runoff voting," presented at the Proceedings of the 2011 conference on Electronic voting technology/workshop on trustworthy elections, San Francisco, CA, 2011.
- [73] K. Arrow, "A Difficulty in the Concept of Social Welfare," *Journal of Political Economics*, vol. 58, pp. 328-346, 1950.
- [74] A. Jameson and B. Smyth, "Recommendation to Groups," in *The Adaptive Web*. vol. 4321, P. Brusilovsky, A. Kobsa, and W. Nejd, Eds., ed: Springer Berlin Heidelberg, 2007, pp. 596-627.
- [75] J. Green Armytage, "Four Condorcet-Hare Hybrid Methods for Single-Winner Elections," *Voting Matters*, vol. 29, pp. 1-14, 2011.
- [76] A. Crossen, J. Budzik, and K. J. Hammond, "Flytrap: intelligent group music recommendation," presented at the Proceedings of the 7th international conference on Intelligent user interfaces, San Francisco, California, USA, 2002.

- [77] G. Adomavicius, N. Manouselis, and Y. Kwon, "Multi-Criteria Recommender Systems," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., ed: Springer US, 2011, pp. 769-803.
- [78] P. C. Fishburn, *Utility Theory for Decision Making*: John Wiley & Sons, New York, 1970.
- [79] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*: John Wiley & Sons, New York, 1976.
- [80] B. Roy, "Classement et choix en présence de points de vue multiples," *RAIRO - Operations Research - Recherche Opérationnelle*, vol. 2, pp. 57-75, 1968.
- [81] B. Roy, "ELECTRE III: un algorithme de classement fondé sur une représentation floue des préférences en présence de critères multiples," *Cahiers du CERO*, vol. 20, pp. 3-24, 1978.
- [82] W. Edwards, "How to Use Multiattribute Utility Measurement for Social Decisionmaking," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 7, pp. 326-340, 1977.
- [83] W. Edwards and F. H. Barron, "SMARTS and SMARTER: Improved Simple Methods for Multiattribute Utility Measurement," *Organizational Behavior and Human Decision Processes*, vol. 60, pp. 306-325, 1994.
- [84] C.-L. Hwang and K. Yoon, *Multiple Attribute Decision Making Methods and Applications: A State-of-the-Art Survey*: Springer Berlin Heidelberg, New York, 1981.
- [85] T. L. Saaty, *The Analytic Hierarchy Process*: McGraw-Hill, New York, 1980.
- [86] J. P. Brans and P. Vincke, "A preference ranking organization method," *Management Science*, vol. 31, pp. 647-656, 1985.
- [87] J. P. Brans, P. Vincke, and B. Mareschal, "How to select and how to rank projects: the PROMETHEE method," *European Journal of Operational Research*, vol. 24, pp. 228-238, 1986.
- [88] Q. Li, C. Wang, and G. Geng, "Improving personalized services in mobile commerce by a novel multicriteria rating approach," presented at the Proceedings of the 17th international conference on World Wide Web, Beijing, China, 2008.
- [89] L. Hsin-Hsien and T. Wei-Guang, "Incorporating Multi-Criteria Ratings in Recommendation Systems," in *Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference on*, 2007, pp. 273-278.

- [90] E. Yano, E. Sueyoshi, I. Shinohara, and T. Kato, "Development of a recommendation system with multiple subjective evaluation process models," in *Cyberworlds, 2003. Proceedings. 2003 International Conference on*, 2003, pp. 344-351.
- [91] C. W. Kirkwood and J. L. Corner, "The Effectiveness of Partial Information about Attribute Weights for Ranking Alternatives in Multiattribute Decision Making," *Organizational Behavior and Human Decision Processes*, vol. 54, pp. 456-476, 1993.
- [92] Y.-S. Huang, W.-C. Chang, W.-H. Li, and Z.-L. Lin, "Aggregation of utility-based individual preferences for group decision-making," *European Journal of Operational Research*, vol. 229, pp. 462-469, 2013.
- [93] P. Csáki, T. Rapsák, P. Turchányi, and M. Vermes, "Research and development for group decision aid in Hungary by WINGDSS, a Microsoft Windows based group decision support system," *Decision Support Systems 14*, pp. 205-221, 1995.
- [94] R. L. Keeney, "A Group Preference Axiomatization with Cardinal Utility," *Management Science*, vol. 23, pp. 140-145, 1976.
- [95] J. S. Dyer and R. K. Sarin, "Group Preference Aggregation Rules Based on Strength of Preference," *Management Science*, vol. 25, pp. 822-832, 1979.
- [96] R. F. Dyer and E. H. Forman, "Group decision support with the Analytic Hierarchy Process," *Decision Support Systems*, vol. 8, pp. 99-124, 1992.
- [97] C. Macharis, J. P. Brans, and B. Mareschal, "The GDSS PROMETHEE procedure -- A PROMETHEE-GAIA based procedure for group decision support.." *Journal of Decision Systems*, vol. 7, pp. 283-307, 1998.
- [98] J. C. Leyva-López and E. Fernández-González, "A new method for group decision support based on ELECTRE III methodology," *European Journal of Operational Research*, vol. 148, pp. 14-27, 2003.
- [99] A. Brodsky, N. Egge, and X. Wang, "Supporting Agile Organizations with a Decision Guidance Query Language," *J. Manage. Inf. Syst.*, vol. 28, pp. 39-68, 2012.
- [100] A. Brodsky, M. M. Bhot, M. Chandrashekar, N. E. Egge, and X. S. Wang, "A decisions query language (DQL): high-level abstraction for mathematical programming over databases," presented at the Proceedings of the 2009 ACM SIGMOD International Conference on Management of data, Providence, Rhode Island, USA, 2009.

- [101] A. Brodsky, M. Al-Nory, and H. Nash, "Service Composition Language to Unify Simulation and Optimization of Supply Chains," in *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, 2008, pp. 74-74.
- [102] A. Brodsky, L. Kerschberg, and S. Varas, "Resource Management in Agent-based Distributed Environments," in *Cooperative Information Agents III*. vol. 1652, M. Klusch, O. Shehory, and G. Weiss, Eds., ed: Springer Berlin Heidelberg, 1999, pp. 61-85.
- [103] A. Brodsky and H. Nash, "CoJava: A Unified Language for Simulation and Optimization," in *Principles and Practice of Constraint Programming - CP 2005*. vol. 3709, P. van Beek, Ed., ed: Springer Berlin Heidelberg, 2005, pp. 877-877.
- [104] A. Brodsky and H. Nash, "CoJava: Optimization Modeling by Nondeterministic Simulation," in *Principles and Practice of Constraint Programming - CP 2006*. vol. 4204, F. Benhamou, Ed., ed: Springer Berlin Heidelberg, 2006, pp. 91-106.
- [105] A. Brodsky, "Constraint Databases: Promising Technology or Just Intellectual Exercise?," *Constraints*, vol. 2, pp. 35-44, 1997.
- [106] A. Brodsky and V. E. Segal, "The C3 Constraint Object-Oriented Database System: An Overview," presented at the Second International Workshop on Constraint Database Systems, Constraint Databases and Their Applications, 1997.
- [107] H. Mengash and A. Brodsky, "GCAR: A Group Composite Alternatives Recommender Based on Multi-Criteria Optimization and Voting," in *47th Hawaii International Conference in System Sciences (HICSS)*. 2014, pp. 1113 - 1121.
- [108] H. Mengash and A. Brodsky, "DG-GPR: A decision-guided group package recommender with hybrid condorcet-instant runoff voting" in *DSS 2.0 - Supporting Decision Making with New Technologies*, Paris, France, 2014, pp. 317 - 328.
- [109] T. R. Magrino, R. L. Rivest, E. Shen, and D. Wagner, "Computing the margin of victory in IRV elections," presented at the Proceedings of the 2011 conference on Electronic voting technology/workshop on trustworthy elections, San Francisco, CA, 2011.
- [110] H. Mengash and A. Brodsky, "A group package recommender based on learning group preferences, multi-criteria decision analysis, and voting," *EURO Journal on Decision Processes*, vol. 3, pp. 275 - 304, 2015.
- [111] M. Komkhao, J. Lu, Z. Li, and W. Halang, "Improving Group Recommendations by Identifying Homogenous Subgroups," in *Knowledge Engineering and Management*. vol. 214, F. Sun, T. Li, and H. Li, Eds., ed: Springer Berlin Heidelberg, 2014, pp. 453-462.

- [112] L. Boratto and S. Carta, "Modeling the Preferences of a Group of Users Detected by Clustering: a Group Recommendation Case-Study," presented at the Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14), Thessaloniki, Greece, 2014.
- [113] E. Ntoutsi, K. Stefanidis, K. Nørvag, and H.-P. Kriegel, "Fast group recommendations by applying user clustering," presented at the Proceedings of the 31st international conference on Conceptual Modeling, Florence, Italy, 2012.
- [114] H. Mengash and A. Brodsky, "Tailoring Group Package Recommendations to Large Heterogeneous Groups Based on Multi-Criteria Optimization," in *49th Hawaii International Conference on System Science (HICSS)*, 2016, pp. 1537 - 1546.
- [115] K. j. Kim and H. Ahn, "A recommender system using GA K-means clustering in an online shopping market," *Expert Systems with Applications*, vol. 34, pp. 1200-1209, 2008.
- [116] X. Wang, Y. Wang, and Y. Zhan, "Optimization of K-means Clustering by Weight Learning," *Journal of Computer Research and Development*, vol. 40.
- [117] S. Fazlollahi, P. Mandel, G. Becker, and F. Maréchal, "Methods for multi-objective investment and operating optimization of complex energy systems," *Energy*, vol. 45, pp. 12-22, 2012.
- [118] C.-K. Ngan, A. Brodsky, N. E. Egge, and E. Backus, "A Decision-Guided Energy Framework for Optimal Power, Heating, and Cooling Capacity Investment," in *ICEIS (I)*, 2013, pp. 357-369.
- [119] H. Altaieb, "Market-based decision guidance framework for power and alternative energy collaboration," Ph.D., Computer Science, George Mason University - WRLC, ProQuest Dissertation & Theses Global, 2015.
- [120] The IBM Corporation. (2012). *Optimization Programming Language (OPL)*. Available: http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r4/index.jsp?topic=%2Filog.odms.ide.help%2FOPL_Studio%2Fmaps%2Fgroupings_Eclipse_and_Xplatform%2Fps_opl_Language_1.html
- [121] Open Energy Info. (OpenEI). (2015). *Commercial and Residential Hourly Load Profiles for All TMY3 Locations in the United States*. Available: http://en.openei.org/data_sets/data_set/commercial-and-residential-hourly-load-profiles-for-all-tmy3-locations-in-the-united-states/resource/b341f6c6-ab5a-4976-bd07-adc68a2239c4.
- [122] ILOG S. A. and ILOG Inc. (2007). *ILOG CPLEX 11.0 User's Manual*. Available: <http://www-eio.upc.es/lceio/manuals/cplex-11/html/>.

- [123] Hawaii Electric Company Inc. *HecoDocReview*. *HECO*. Available: <https://code.google.com/p/openmicrosmartgrid/wiki/HecoDocReview>
- [124] California Energy Commission. (2001). *Peak electricity demand forecasts*. Available: http://www.energy.ca.gov/electricity/comission_demand_forecast.html
- [125] A. T. Kearney. (2015). *Confronting Electricity Costs in the United States*. Available: http://www.atkearney.com/oil-gas/featuredarticle//asset_publisher/S5UkO0zy0vnu/content/confronting-electricity-costs-in-the-united-states/10192?_101_INSTANCE_S5UkO0zy0vnu_redirect=/oil-gas.

BIOGRAPHY

Hanan A. Mengash received her bachelor of computer science from King Saud University in Riyadh, Saudi Arabia, in 1993. In 1999, she graduated from George Washington University in Washington, DC, USA, with a master of science in engineering management.

Hanan has more than 20 years of professional experience teaching and working in IT. After completing her bachelor's degree, she worked as a systems analyst at King Khaled Eye Specialist Hospital in Riyadh. In 1995, she joined the staff of Princess Nourah bint Abdulrahman University in Riyadh, where she was awarded a scholarship to complete her master's degree in the United States. Since 1999, Hanan has been on the faculty of Princess Nourah bint Abdulrahman University, where she was awarded a second scholarship to complete her doctoral degree in information technology at George Mason University in Virginia, USA.

She has conducted research on group decision guidance for composite alternatives recommenders and has published a number of research papers during her PhD studies. Her research interests include group decision support, group recommender systems, and recommender systems for optimal energy operation and investment.